Amy LaViers
Magnus Egerstedt *Editors*

Controls and Art

Inquiries at the Intersection of the Subjective and the Objective

Springer

Controls and Art

Amy LaViers · Magnus Egerstedt
Editors

Controls and Art

Inquiries at the Intersection of the Subjective and the Objective

Springer

Editors
Amy LaViers
Systems and Information Engineering Department
University of Virginia
Charlottesville, VA
USA

Magnus Egerstedt
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, GA
USA

ISBN 978-3-319-03903-9 ISBN 978-3-319-03904-6 (eBook)
DOI 10.1007/978-3-319-03904-6
Springer Cham Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013957127

Printed on acid-free paper

Photo used in cover design is by Christian Moreno

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Several years ago, we started thinking about what it means for a robot to move in a particular "style of motion." This question arose out of a desire to be able to make robotic movements more meaningful and relatable to humans and a curiosity about what distinguishes different genres of movement. It quickly became clear that this line of inquiry required a whole new set of tools for understanding motion generation in particular, and control design in general—tools that related to our perception of human motion and to the performing arts.

In conjunction with the technical development of style-based motions, we began searching for a community of researchers who were interested in questions pertaining to "Controls and Art." This search resulted in two Invited Sessions at the American Control Conference (2011 in San Francisco and 2012 in Montreal), with the second one being co-organized with Naomi Leonard. We were pleasantly surprised by the strong interest in these sessions, and by the many creative and interesting approaches to using control-theoretic frameworks to investigate, bolster, and create art. This book is a logical extension of the two invited sessions, where a large subset of our authors contributed papers.

The different chapters in this book fall into one or more of three main camps, namely research that (1) uses artistic ideas for the purpose of control design and analysis, (2) uses control theoretic ideas to understand and analyze art, and (3) uses control theory as a generator of artistic expressions. It is, however, not our ambition to paint the complete picture of the research that can be found in the intersection between Controls and Art, or to imply that this field is close to maturity. Instead, we hope that this book will inspire new research in this exciting line of inquiry.

The marriage between control theory and art is challenging because it requires a union between subjective and objective analysis. In fact, while control theory thrives under well-defined performance specifications and clear metrics of success, art thrives on multiple interpretations and evaluations of a single work. As such, when embarking on an endeavor connecting the two, a shift in perspective is needed. This book represents such a shift: the list of authors includes artists and engineers and the list of topics includes dancing robots, swarming dancers, automated puppetry, reactive museum installments, salsa dancing, intricate geometric patterns, aerial quadrotor ballets, music generated from math, and amorphous blobs that dance the Bhangra. More importantly, we ask our readers to

consider the matters that arise for putting together what may seem like kitschy combos: How do dancers communicate to execute such highly coordinated movements? What does it mean for a robot to dance? What do viewers watching movement notice most? Can we augment human capabilities with technology? What makes a composition of curves aesthetic? How do people interact with curated spaces? And what do the answers to these questions teach us about human creativity?

To answer these questions, the chapters in this book are arranged around different themes, with the first theme investigating fundamental questions such as *How is movement perceived?* (Kingston et al. Chap. 1), *What do performers actually do?* (Leonard et al. Chap. 2), and *What is in a dance?* (Baillieul and Ozcimder, Chap. 3). The next theme involves the construction of control laws based on artistic principles, including aerial robots (Schoellig et al. Chap. 4), robot puppets (Jochum et al. Chap. 5), and teams of mobile robots (Tsiotras and Castro, Chap. 6). The last theme turns the relationship between Controls and Art around by asking how Controls tools can be used in the Arts themselves, including algorithmic composition using swarm theory (Huepe et al. Chap. 7), automation for enhancing museum installations (Godbehere and Goldberg, Chap. 8), and the incorporation of robots in performance (LaViers et al. Chap. 9).

This book would not have been possible without the support from the U.S. National Science Foundation, the U.S. Office of Naval Research, the U.S. Office of Scientific Research, the Swiss National Science Foundation, the Chilean National Council of Culture and Arts, and numerous private foundations, Oliver Jackson at Springer who got as excited about this idea for a book as we were, and Naomi Leonard who helped connect us to several of the authors here in the Invited Session she co-organized for the American Control Conference in 2012.

Charlottesville Amy LaViers
Atlanta, October 2013 Magnus Egerstedt

Contents

Chapter 1
Metric Preference Learning with Applications to Motion Imitation

Peter Kingston, Jason von Hinezmeyer and Magnus Egerstedt

1.1 Introduction to Motion Imitation Through Puppetry

In entertainment and other artistic endeavors, subjective notions such as "style" and "aesthetics" play a key role. When control systems are incorporated in these settings, they are asked to generate behaviors that, rather than achieving a well-defined physical outcome, serve an esthetic or communicative purpose. In these situations, the effectiveness of the control strategy is ultimately the degree to which it aligns with the subjective judgments of human observers.

One specific control application in which this issue has arisen is *robotic puppetry* (see, e.g., [18, 27]), where marionettes with actuated strings—highly complex mechanical systems—are asked to perform expressive and esthetically pleasing motions in the context of puppetry plays, as in Chap. 5, by Jochum et al., and as shown in Fig. 1.1. Given a particular human motion, how should the vastly more limited marionette move to not just mimic the human motion but also communicate the same emotional intent? Similar issues are encountered in controls applications like [8, 16], as well as in inverse reinforcement learning settings, e.g., [1, 32, 35].

In puppetry, as in other performing arts, not just any movement is artistically interesting or meaningful; What is the difference between someone just moving an object and someone animating it? What is the difference between someone folding their laundry and someone moving a shirt in a manner that gives it a personality and brings it to life? The movements of the puppet have the intention to give the puppet

P. Kingston · M. Egerstedt (✉)
School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA
e-mail: magnus@gatech.edu

P. Kingston
e-mail: kingston@gatech.edu

J. von Hinezmeyer
Center for Puppetry Arts, Atlanta, GA 30309, USA
e-mail: Jasonhines@puppet.org

A. LaViers and M. Egerstedt (eds.), *Controls and Art*,
DOI: 10.1007/978-3-319-03904-6_1,

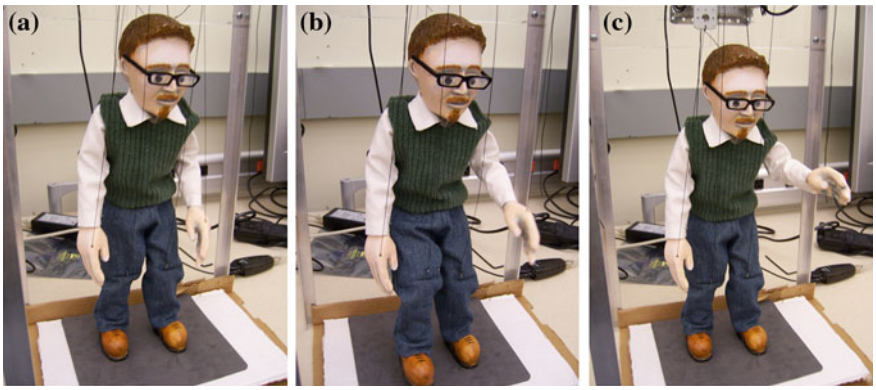


Fig. 1.1 A robotic puppet executing a wave motion

life and through those movements to communicate an idea. It is the movement of that puppet that is the heart of the communication. But just what is it about the movement that enables effective communication? Similar questions are pursed in Chap. 4 by Scholleing et al. and Chap. 9, by LaViers et al.

To answer this question in a quantitative manner, this chapter draws inspiration from puppetry. And, it is important to note that puppets are defined by their limitations more than their abilities [7, 10, 19]. In fact, the limited movements of the puppet are a distillation of human movement—so-called motion caricature. Simply put, a puppet cannot do all of the movement of a living person, no matter how simple or complicated the puppet is in its design or construction. It is therefore necessary to distil the movement that the puppet can do to its very essence by first imitating the human motion, then simplifying it to make it executable on the puppet, and lastly, by exaggerating the motion to make it more expressive.

In order to produce such expressive movements, some guidelines have been developed through three main classes of puppet movements [33]. The first class of movements—the primary movements—deals with the overall puppet body and it captures basic movements such as "up," "down," "left," and "right." Any angling or spinning of the puppet body is also considered a primary movement. "Breath" is another major, primary movement that the puppet must perform to seem alive. In fact, most of the puppet's emotions and attitudes are expressed through the primary movement class, including the speed at which the puppet starts and stops or whether the movement is smooth and continuous or jerky and broken.

The secondary movement class is the next class of puppet movement. It considers the moving parts that are attached to the puppet, such as head, arms, and legs. The movements of these parts can multiply the emotions expressed by the puppet through, for example, a subtle tilt of the head or various positions of the arms. Finally, the tertiary movement class concerns puppets that have elements in their designs that are not directly controlled by the puppeteer. These can include, but are not limited to, costume elements such as capes or long sleeves, or hair which can be made of

lightweight fibrous materials or feathers. In short, the tertiary class contains anything that can move on the puppet that can help it come alive.

In this chapter, we try to take these observations about the Puppetry Arts and make then quantitative and mathematically precise. But, any such endeavor must ultimately allow for subjective, human observers to enter into the discussion. To accommodate this, this paper will indeed focus on how to imitate motions based on human preferences for which synthetic motions best capture the original, human motion, with the ultimate objective of producing artistically meaningful motions. And we achieve this by making so-called preference learning applicable to motion sequences.

1.2 Preference Learning

The role of human preferences in these problems is unavoidable in that a system's output is esthetically pleasing only if we think it is pleasing. In this work, we address is this rather unusual issue by developing techniques both for using empirical measurements to learn cost functions that are consistent with humans' esthetic preferences, and for generalizing from these preference measurements to determine a globally best alternative. For the example of the marionette asked to mimic a human, this would mean finding the one marionette motion that best captures the subjective "essence" of a given human motion.

The idea of learning costs or rating functions from expressed preferences has received significant attention, and a number of related approaches and problems exist. We sketch a taxonomy of these approaches in the next few paragraphs, before highlighting their potential drawbacks, together with the novelties of the approach pursued here.

In *instance preference learning*, one is given a set of objects called *instances* or *alternatives* (usually points in a real vector space)[1] together with information about humans' preferences among them. The problem is to learn functions that generalize these expressed preferences to the entire space in some way. When the preference data take the form of values from an ordinal scale—e.g., "Good," "Fair," "Poor"—the problem is known as *ordinal regression* (e.g., [12, 23]). When they take the form of a collection of *pairwise comparisons* (i.e., answers to questions of the form "Which of these two options is better?"), we will refer to the problem as *preference learning* (e.g., [2, 11, 13, 15, 20, 22, 25]). Often, preference learning is done by constructing a real-valued ranking function over the instances (e.g., [20]), but in some cases, particularly when one wishes to allow intransitive preferences, one can seek merely to solve a binary classification problem that determines, for a given pair of instances, which is preferred (as in, e.g., [22]). Applications have included route selection by

[1] To the extent that a distinction is made between "instances" and "alternatives," it is that "instances" are the points that were shown to human judges, whereas "alternatives" may also include other points in the space besides those that were seen.

automotive GPS systems [20], food preference studies [6, 17], and the sorting of search results [15, 26], among many others.

It should be noted that pairwise comparison studies have the advantage over numerical or ordinal-ranking experiments of being less prone to *batch effects*, a psychological phenomenon in which people's rankings are only accurate among objects compared at around the same time [17]. Specific experimental protocols include two-alternative forced choice (2AFC), in which, when comparing two objects A and B, subjects must either respond "A is better than B" or "B is better than A;" three-alternative forced choice (3AFC), in which "the two are the same" is also an acceptable answer; and 4AFC, which also includes "the two are incomparable" as an answer. Our attention will be on 2 and 3AFC experiments. In the related problem of *label preference learning* (e.g., [4, 24]), one attempts to learn a function that, when presented with an instance, returns a partial order over *labels* for that instance. That problem will fall somewhat outside the scope of this work.

For solving instance- and label-preference problems, *large-margin* approaches (e.g., [20, 23]) dominate in the literature. The *margin*, which these methods maximize, is the size of the largest error in the data that can be accommodated without contradicting the learned model. For example, for linear classifiers, it is the distance from the decision hyperplane to the nearest instance, and for linear rating functions, it is the distance between level sets of the learned function. Ultimately, Support Vector Machine (SVM) algorithms (e.g., SMO [31]) are used to solve these problems. That said, competitive Gaussian-process [11] and least-squares [25] approaches also exist, which bring Bayesian and algebraic-topological interpretations, respectively.

The essential idea underlying the large-margin approaches is to develop algorithms that learn linear rating functions or classifiers, and to then generalize these algorithms to the nonlinear case through the use of *Mercer kernels* (the so-called *kernel trick* [5, 9]). It is an elegant and general approach, but what it lacks is the ability to easily guarantee that the resulting cost functions are convex, which is an obstacle to the efficient determination of globally best alternatives. We will investigate techniques that sacrifice some of the generality of the kernel-linear approach in exchange for formulations that allow for the efficient determination of globally optimal alternatives. This chapter unifies and builds on our work [28, 29], and, of earlier work on preference learning, is most closely related to the large-margin instance-preference-learning approaches of [17, 22, 23]. One of the methods we propose—a Chebyshev estimation scheme—is similar in that it also employs a constrained optimization approach and in a particular limiting case (but only then) can be reduced to an equivalent SVM classification problem. Our proposed approach differs from [17, 22, 23] in that it aims to find not only a rating function but also a globally best alternative, and to solve only computationally efficient convex programs, which in turn motivates different (and sometimes more efficient) problem formulations and solutions.

The resulting algorithms operate on batch data, and simultaneously use all of the information available. We give a graph-theoretic method, based on the work of [3] on transitive reductions, to efficiently simplify the resulting optimization problems prior to their solution. Additionally, for situations in which a great deal more data

are available in real-time, we also present a limited memory asymptotic observer, which trades some efficiency in the use of data for constant memory requirements and very cheap measurement updates, and which nevertheless guarantees probabilistic convergence to a global optimum.

The chapter is organized as follows: After giving a problem formulation, we describe an algebraic-graph-theoretic simplification method for the resulting systems of constraints, before introducing the optimization problems in more detail in both *direct* and *instance vector expansion* forms, and an example is given to demonstrate the applicability of the methods to LQ-type optimal control problems. Next, we show how in a particular limiting case (but only then) a natural generalization is equivalent to a certain SVM classification problem, before describing a novel limited memory asymptotic observer. The approaches are demonstrated with two examples; the first of these compares apples relative to an orange, and the second involves the comparison of human and synthetic motions.

1.3 Problem Formulation

At the core of preference learning is a collection of empirical, pairwise comparisons. The assumption is that these comparisons reflect an underlying rating function. Hence, given a sequence of pairwise comparisons between points in an inner product space, we wish to find (1) a real-valued rating function that is consistent with those preferences, and (2) a global optimum to this function—the best point in the metric space. By solving these two problems we would have recovered what the underlying source for the comparisons is.

Formally, let $(X, \langle\cdot,\cdot\rangle)$ be the inner product space, and $S = \{(x_i^1, x_i^2)\}_{i=1}^N = \{s_1, \cdots, s_N\} \subset X \times X$ the sequence of comparisons; a pair (x_i^1, x_i^2) appears in the sequence S if and only if x_i^1 is preferred to x_i^2. The first item we seek, given some assumptions about its parametric form, is a function $f : X \to \mathbb{R}$ such that

$$f(x^1) \leq f(x^2) \Leftrightarrow (x^1, x^2) \in S. \tag{1.1}$$

That is, we adopt the convention that lower scores are better; hence we will refer to f as a *cost function*.

The second item we seek is a global minimizer to f,

$$\bar{x} \triangleq \operatorname{argmin}_x f(x) \tag{1.2}$$

which represents the best possible point in the inner product space.

Crucially, we would like to be able to determine f and $\bar{x}$ entirely by convex optimization—both so that the resulting problems are computationally efficient, and to ensure that any minima we find are in fact global optima. Although the SVM methodology employed in, e.g., [17, 23] finds f as the solution to a convex program, its

use of *Mercer kernels* (the so-called *kernel trick*—see, e.g., [5, 9]) introduces nonlinearities that prevent the determination of $\bar{x}$ by convex programming. Yet without the kernel trick and using the SVM approach, one arrives at linear cost functions for which minima do not exist. What we will present in this chapter is instead a set of convex programs that provide a useful compromise between these extremes, and which only reduce to an SVM classification problem in a particular limiting case. These formulations will allow us to entertain the idea of a unique "best" point in X, and at the same time determine what it is by convex programming.

1.4 The Preference Graph

The *preference graph* $\mathcal{G} = (V, S)$ corresponding to the comparison sequence S is the directed graph whose vertex set $V = \{x_1^1, x_1^2, \cdots, x_N^1, x_N^2\} \subset X$ is the collection of all unique points that have been compared, and whose edge set is S. We will index the vertices as $V = \{x_1, \cdots, x_M\}$, where $M \leq 2N$ is the cardinality of V.

If (1.1) is to hold with strict inequality, then we note immediately that the graph $\mathcal{G}$ must be *acyclic*, and thus represent a *partial order*. When nonstrict inequalities are allowed, however, then we may permit cycles, and moreover $\mathcal{G}$ can be replaced by a smaller, equivalent acyclic graph. This has the practical significance of allowing redundant constraints to be eliminated on purely graph-theoretic grounds, thereby speeding up later optimization steps. This is constructed, following [3], in the following way:

A *cell* is defined to be an equivalence class of vertices; two vertices $v_1, v_2 \in V$ belong to the same cell (denoted $v_1 \sim v_2$) if and only if there exist directed paths in $\mathcal{G}$ from v_1 to v_2 *and* from v_2 to v_1. The quotient graph $\mathcal{G}/\sim$ is the directed acyclic graph whose vertices are these equivalence classes, and in which the directed edge (C_1, C_2) exists between two cells C_1 and C_2 whenever there exist vertices $v_1 \in C_1$ and $v_2 \in C_2$ such that there is a directed path in $\mathcal{G}$ from v_1 to v_2.

Since any two vertices in the same cell must by (1.1) have the same cost, one may optimize using only the constraints represented by the edges of this quotient graph, and discard the rest. Hence without loss of generality we will assume that $\mathcal{G}$ is acyclic; when it is not it should be understood that we will actually work with $\mathcal{G}/\sim$.

Additional constraints can be eliminated via the *transitive reduction*. Formally, using Aho's definition [3], $\mathcal{G}^t$ is the transitive reduction of a graph $\mathcal{G}$ if,

1. there is a directed path from vertex u to vertex v in $\mathcal{G}^t$ if and only if there is a directed path from u to v in $\mathcal{G}$, and
2. there is no graph with fewer arcs than $\mathcal{G}^t$ satisfying condition 1.

In the case of a directed acyclic graph, the reduction $\mathcal{G}^t$ (which is unique) is a subgraph of $\mathcal{G}$. It was shown in [3] that computation of the transitive reduction is of the same complexity as transitive closure, and hence matrix multiplication; thus, the transitive reduction can be found in $O(n^{\log_2 7})$ steps using Strassen's algorithm [34], or, in principle, $O(n^{2.376})$ steps using the Coopersmith-Winograd algorithm [14]. (See,

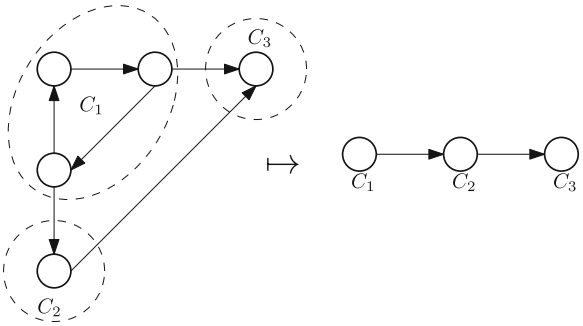


Fig. 1.2 The original preference graph $\mathcal{G}$ (*left*), and the corresponding transitively reduced quotient graph, $(\mathcal{G}/\sim)^t$ (*right*). The vertices of $(\mathcal{G}/\sim)^t$, labeled $C_1, \cdots, C_3$, are sets of vertices in $\mathcal{G}$ called *cells* (*dashed circles and ellipses, left*)

e.g., [21, 30]). Moreover, if $\mathcal{G}$ contains cycles, then the algorithm given in [3] can compute $(\mathcal{G}/\sim)^t$ with the same complexity.

In short, by working with the transitive reduction of the quotient graph, we are able to eliminate redundant constraints on purely graph-theoretic grounds, before even knowing the form of the cost function f. The reduction is shown in Fig. 1.2.

1.5 Metric Costs

Colloquially, when comparing various alternatives, we often speak of options as being "closer to what we would like," or of being "far from perfect." Motivated by this everyday use of geometric language, in [28] we considered *metric costs*, which have the form,

$$f(x) = ||x - \bar{x}||^2. \tag{1.3}$$

In short, it is assumed that there exists some single best point $\bar{x}$ in X, and one alternative is preferred over another if and only if it is closer to that point. Moreover, costs of this type are nicely compatible with the infinite-dimensional alternatives encountered in control applications, as demonstrated in the following, motivating example:

Example 1 (LQ Tracking).

We consider a situation in which a human user would like a linear system

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \tag{1.4}$$

to perform a tracking task, with an LQ-type cost functional,

$$J(y,u) = \frac{1}{2}\int_0^T \Big[(y-\bar{y})^T Q(y-\bar{y}) + (u-\bar{u})^T R(u-\bar{u})\Big]dt \tag{1.5}$$

where $Q = Q^T \succ 0$, $R = R^T \succ 0$, and $T > 0$. Subjectivity enters because the pair $\bar{x} \triangleq (\bar{y}, \bar{u})$ is known only implicitly to the user, and it is the goal of preference learning to estimate what it is. It is well-known that (1.5) can be written as

$$\frac{1}{2}||(y,u) - (\bar{y},\bar{u})||^2_{\text{LQR}} \tag{1.6}$$

where $||\cdot||_{\text{LQR}}$ is the norm induced by the inner product

$$\langle (y_1,u_1),(y_2,u_2)\rangle_{\text{LQR}} = \int_0^T \Big[y_1^T Q y_2 + u_1^T R u_2\Big]dt. \tag{1.7}$$

If both y and u are measured, this is enough to apply directly the preference learning methods to be introduced in the subsequent sections, without knowledge of the dynamics (1.4). However, if these dynamics are known, then together with (1.7), they induce an inner product on the control inputs, which enables us to work in that smaller vector space. This is done in the following way:

Defining the linear operator $\mathcal{L} : L^2([0,T],\mathbb{R}^m) \to L^2([0,T],\mathbb{R}^p)$ by,

$$(\mathcal{L}u)(t) \triangleq C\int_0^t e^{A(t-\tau)Bu(\tau)}d\tau \tag{1.8}$$

we define a second inner product,

$$\langle u_1,u_2\rangle_{\text{LQR}'} \triangleq \langle(\mathcal{L}u_1,u_1),(\mathcal{L}u_2,u_2)\rangle_{\text{LQR}}. \tag{1.9}$$

With these definitions, the cost can be written,

$$J(u) = \frac{1}{2}||u-\bar{u}||_{\text{LQR}'} \tag{1.10}$$

and the problem, by taking advantage of knowledge of the system dynamics, has been reduced from that of finding a pair $(\bar{y},\bar{u})$, to that of simply finding $\bar{u}$. Moreover, note that (1.9) can be written in a more standard double-integral form as,

$$\langle u_1,u_2\rangle_{\text{LQR}'} = \int_0^T\int_0^T u_1^T(\tau_1)M(\tau_1,\tau_2)u_2(\tau_2)d\tau_1 d\tau_2, \tag{1.11}$$

where

$$M(\tau_1,\tau_2) = \delta(\tau_1-\tau_2)R + \int_{T-|\tau_2-\tau_1|}^T B^T e^{A^T(t-\tau_1)}c^T Q c e^{A(t-\tau_2)}B dt \tag{1.12}$$

and δ denotes the Dirac delta distribution. In this way, system dynamics are incorporated directly into the preference learning framework.

Under an assumption of metric costs, what does an individual response (x^1, x^2) tell us about the location of $\bar{x}$? The following are equivalent:

1. $(x_i^1, x_i^2) \in S$
2. $f(x^1) \leq f(x^2)$
3. $\langle x_i^2 - x_i^1, \bar{x} \rangle - \frac{1}{2}\langle x_i^2 - x_i^1, x_i^2 + x_i^1 \rangle < 0.$

In words, each comparison constrains $\bar{x}$ to lie within a particular halfspace of X. Defining,

$$d_i \triangleq x_i^2 - x_i^1 \tag{1.13}$$

$$\mu_i \triangleq \frac{1}{2}\left(x_i^1 + x_i^2\right) \tag{1.14}$$

$$b_i \triangleq \langle d_i, \mu_i \rangle, \tag{1.15}$$

the totality of what we know, then, about where $\bar{x}$ might lie is summarized by the inclusion over all the comparison halfspaces,

$$\bar{x} \in P \triangleq \bigcap_{i=1}^{N} \{x \mid \langle d_i, x \rangle - b_i < 0\}. \tag{1.16}$$

The set P, if it is bounded, is a polytope in X. In [28], we stated this system of inequalities and gave an asymptotic observer that converges to $\bar{x}$ under certain assumptions. Here, we ask another question: Out of all the points in this polytope, which is "best?" Two cases will be relevant: Either a particular linear program is bounded, in which case such a platonic ideal point $\bar{x}$ exists; or it is unbounded, in which case only an ideal *direction* or *ray* exists. The next subsections introduce this linear program and address these two cases in turn.

1.5.1 Bounded Case

When P is bounded, a natural choice for $\bar{x}$ is the *incenter* or *Chebyshev center* of the polytope,

$$\bar{x} = \text{argmin}_x \max_i \frac{1}{||d_i||}(\langle d_i, x \rangle - b_i). \tag{1.17}$$

This is the point that is maximally far away from the closest constraint plane, as shown in Fig. 1.3. In other words, when P is nonempty, $\bar{x}$ is the point that can be perturbed as much as possible without contradicting any of the preferences expressed in S; and when P is empty, it is the "compromise" point whose worst constraint violation

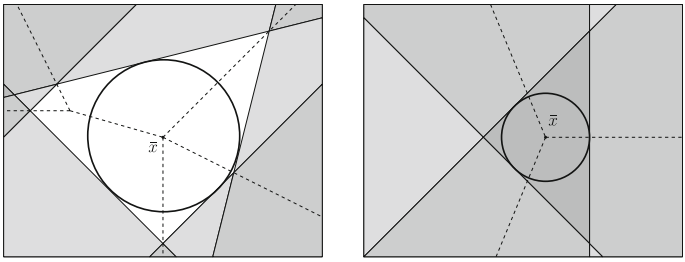


Fig. 1.3 Two examples for $X = \mathbb{R}^2$. Shades of *gray* indicate the number of violated constraints (points in darker regions violate more constraints), and discontinuities in the derivative of the piecewise-linear function $x \mapsto \max_i \frac{1}{||d_i||}(\langle d_i, x\rangle - b_i)$ are indicated by *dashed lines*. In the first example (*left*), $P \neq \emptyset$ (*white region*), and $\bar{x}$ is its incenter, the point maximally far away from the closest of the constraint surfaces (*thin*, *solid lines*) - i.e., it is the center of the largest inscribed sphere (*thick*, *solid curve*). In the second example (*right*), $P = \emptyset$, and the resulting optimum, $\bar{x}$, is the point whose worst constraint violation is minimal

is minimal. That is, with the definition (1.17), if the constraints are feasible (i.e., if $P \neq \emptyset$), then $\bar{x} \in P$. This can be viewed as minimizing the ∞-norm of the vector of constraints. The minimization problem (1.17) is feasible even when P is empty, in which case its solution is the point whose worst constraint violation is as small as possible. Equivalently, the problem (1.17) can be rewritten in epigraph form as the linear program,

$$\begin{aligned}(\bar{z}, \bar{x}) = \text{argmin}_{(z,x)} z \\ \text{s.t.} ||d_i|| z \geq \langle d_i, x\rangle - b_i\end{aligned} \tag{1.18}$$

which is always feasible (but possibly unbounded), and satisfies $\bar{z} > 0 \iff P = \emptyset$.

The formulation (1.17) also has the advantage that it can be solved with a complexity that grows not with the dimensionality of the space X containing the points, which may be quite large, but instead with the number of instances seen, which is typically much smaller; Theorem 1 states the fact that enables this simplification:

Theorem 1. *If* (1.17) *has a global minimizer, then it has a global minimizer in* $\text{aff}\left\{x_1^1, x_1^2, \cdots, x_N^1, x_N^2\right\}$.

Proof. Let x be a global minimum to (1.17), and $\bar{x}$ be the projection of x onto $\text{aff}\left\{x_1^1, x_1^2, \cdots, x_N^1, x_N^2\right\}$; i.e., $\bar{x} = x + \delta$ with $\delta \perp \text{span}\{d_1, \cdots, d_N\}$. Then for all $i \in \{1, \cdots, N\}$, since $\langle d_i, \delta\rangle = 0$ and by linearity of the inner product, $\frac{1}{||d_i||}\langle d_i, \bar{x}\rangle - b_i = \frac{1}{||d_i||}\langle d_i, x\rangle - b_i$, and hence the value of the objective function in (1.17) is the same at either x or $\bar{x}$.

In Sect. 5.1, which follows, we describe an efficient *Instance Vector Expansion* form suggested by Theorem 1, and obtain problems that have optima when P is

bounded. For the case when the set P is unbounded, we will generalize the solution concept and find optimal *directions* rather than points, in Sect. 5.2.

Instance Vector Expansion

Since $\bar{x} \in \text{aff}\left\{x_1^1, x_1^2, \cdots, x_N^1, x_N^2\right\}$, i.e., the affine span of the constituent points, the optimization problem (1.17) can be solved as a finite-dimensional problem even when X is not finite-dimensional, by expanding $\bar{x}$ in terms of a finite-dimensional basis, as described by the following theorem:

Theorem 2. *The point*

$$\bar{x} = \sum_{k=1}^{N} \bar{c}_k d_k + x^* \tag{1.19}$$

solves the optimization problem (1.17)*, where*

$$x^* = argmin_x \left\{ ||x||^2 \mid x \in \text{aff}\left\{x_1^1, x_1^2, \cdots, x_N^1, x_N^2\right\}\right\}, \tag{1.20}$$

and $\bar{c}$ *is found by solving*

$$\begin{aligned} (\bar{z}, \bar{c}) =& argmin_{(z,c)} z \\ & \text{s.t.} G^{dd} c - Dz \leq \beta, \end{aligned} \tag{1.21}$$

with $D = (||d_1||, \cdots, ||d_N||)$, $\beta \in \mathbb{R}^N$ *defined by*

$$\beta_i \triangleq \langle d_i, \mu_i \rangle \tag{1.22}$$

and $G^{dd} \in \mathbb{R}^{N \times N}$ *being the Gramian,*

$$G^{dd} \triangleq \begin{bmatrix} \langle d_1, d_1 \rangle & \cdots & \langle d_1, d_N \rangle \\ \vdots & \ddots & \vdots \\ \langle d_N, d_1 \rangle & \cdots & \langle d_N, d_N \rangle \end{bmatrix}. \tag{1.23}$$

Proof. Defining x^* by (1.20), one can write any x in the affine span of the data in the form (1.19). Substituting the expansion (1.19) into (1.18) and noting that by Hilbert's Projection Theorem $x^* \perp d_i$ for all $i \in \{1, \cdots, N\}$, one obtains (1.21).

Remark 1. We also note at this point that (1.19) can be written,

$$x = \sum_{k=1}^{M} \left(\text{indeg}_c(x_k) - \text{outdeg}_c(x_k)\right) x_k + x^* \tag{1.24}$$

$$\triangleq \sum_{k=1}^{M} \xi_k x_k + x^* \tag{1.25}$$

by treating c as a vector of edge weights to the preference graph, and denoting the weighted in- and out-degrees of a given node x_k by $\text{indeg}_c(x_k)$ and $\text{outdeg}_c(x_k)$ respectively. Precisely,

$$\text{indeg}_c(x_k) \triangleq \sum_{i|x_i^2 = x_k} c_i \tag{1.26}$$

$$\text{outdeg}_c(x_k) \triangleq \sum_{i|x_i^1 = x_k} c_i . \tag{1.27}$$

Remark 2. Moreover, β can be written,

$$\beta_i = e_i^T G^{\mu d} e_i , \tag{1.28}$$

where $G^{\mu d} \in \mathbb{R}^{N \times N}$ is the cross-Gramian

$$G^{\mu d} \triangleq \begin{bmatrix} \langle d_1, \mu_1 \rangle & \cdots & \langle d_1, \mu_N \rangle \\ \vdots & \ddots & \vdots \\ \langle d_N, \mu_1 \rangle & \cdots & \langle d_N, \mu_N \rangle \end{bmatrix} \tag{1.29}$$

and e_i denotes the i-th element of the natural basis.

Remark 3. Note that the problem (1.21) depends only on inner products of the various d_i and u_i vectors, and hence the problem can be solved even when X is infinite-dimensional. Precisely, $\frac{N(N+1)}{2} + N^2 \sim O(N^2)$ inner products must be computed to build the matrices G^{dd} and $G^{\mu d}$, where N is the number of comparisons. Alternatively, the relevant matrices can also be produced directly from inner products of elements of S, as

$$G^{dd} = K^{22} - K^{21} - K^{12} + K^{11} \tag{1.30}$$

$$G^{\mu d} = \frac{1}{2}(K^{22} + K^{21} - K^{12} - K^{11}) \tag{1.31}$$

where each matrix $K^{lm} \in \mathbb{R}^{N \times N}$ is defined by

$$K_{ij}^{lm} = \langle x_i^l, x_j^m \rangle \tag{1.32}$$

and can be built by indexing into the single Gramian (or kernel) matrix $K \in \mathbb{R}^{M\times M}$ defined,

$$K_{ij} = \langle x_i, x_j \rangle. \tag{1.33}$$

Moreover, $D = (\sqrt{G^{dd}_{11}}, \sqrt{G^{dd}_{22}}, \sqrt{G^{dd}_{33}}, \cdots, \sqrt{G^{dd}_{NN}})$.

Finally, $\bar{x}$ can be reconstructed using (1.19) and

$$x^* = \sum_{i=1}^{M} \alpha_i x_i \tag{1.34}$$

$$\alpha = \frac{1}{\mathbf{1}^T K^\dagger \mathbf{1}} K^\dagger \mathbf{1} \tag{1.35}$$

where $K^\dagger$ denotes the Moore-Penrose pseudoinverse of K, and $\mathbf{1} = (1, 1, \cdots, 1) \in \mathbb{R}^M$.

In particular, the costs of the presented instances can be reconstructed as,

$$f(x_k) = (e_k - \xi - \alpha)^T K (e_k - \xi - \alpha) \tag{1.36}$$

where ξ is related to c by (1.24), (1.26), and (1.27).

With a method thus in hand to efficiently compute Chebychev centers when P is bounded, we now turn our attention to the case when it is not, so that we have a meaningful solution in all cases.

1.5.2 Unbounded Case: The Minimax-Rate Problem

When P is nonempty but unbounded, we ask a slightly different question: What is the "point at infinity," or *direction*, that is best? More precisely, what we seek in this case is a unit vector

$$\bar{v} = \text{argmin}_{v\in X | ||v||=1} \lim_{t\to\infty} \frac{1}{t} \left[\max_i \frac{1}{||d_i||} (\langle d_i, tv \rangle - b_i) \right] \tag{1.37}$$

$$= \text{argmin}_{v\in X | ||v||=1} \max_i \frac{1}{||d_i||} \langle d_i, v \rangle \tag{1.38}$$

or equivalently,

$$(\bar{p}, \bar{v}) = \text{argmin}_{v\in X, p\in\mathbb{R}} p \tag{1.39}$$

$$\text{s.t.} \begin{cases} ||d_i|| p \geq \langle d_i, v \rangle \ \ \forall i \in \{1, \cdots, N\} \\ ||v||^2 \leq 1 \end{cases}.$$

As before, an instance vector expansion is possible:

Theorem 3. *Letting* $v = \sum_{k=1}^{N} c_k d_k$, *the optimization problem* (1.39) *is equivalent to*

$$(\bar{p}, \bar{c}) = \operatorname{argmin}_{(p,c)} p \quad \text{s.t.} \begin{cases} G^{dd} c - Dp \leq 0 \\ c^T G^{dd} c \leq 1 \end{cases} \tag{1.40}$$

with the matrices G^{dd} *and* D *as defined in the previous subsection.*

Proof. The proof takes the form of that of Theorem 2.

The problem (1.40) is a finite-dimensional second-order cone program (SOCP), which can be solved efficiently.

The cost function for the unbounded case arises from a similar limit process to (1.38), as

$$f(x) = \lim_{t \to \infty} \left(\frac{1}{t} ||x - vt||^2 - t \right) \tag{1.41}$$

$$= \lim_{t \to \infty} \left[\frac{1}{t} \left(||x||^2 - 2\langle x, vt \rangle + ||vt||^2 \right) - t \right] \tag{1.42}$$

$$= -2\langle x, v \rangle \tag{1.43}$$

which can be evaluated at the instances as,

$$f(x_k) = -2e_k^T K \xi. \tag{1.44}$$

QP Form and Relation to SVMs

In the unbounded case, so long as int P is nonempty, the minimization problem (1.38) can be rewritten as an equivalent quadratic program (QP), which will make the relationship to the usual SVM approach very clear. In fact, (1.38) is equivalent in this case to a particular SVM classification problem (which differs from but is related to that studied in, e.g., [17] and [23]).

Defining,

$$w = \frac{1}{p} v \tag{1.45}$$

and restricting our attention to negative values for p (since when int P is nonempty, $p^* < 0$), we note that

$$\operatorname{argmin}\ p = \operatorname{argmax}\ p^2 = \operatorname{argmin} \frac{1}{p^2} = \operatorname{argmin}\ ||w||^2. \tag{1.46}$$

Additionally, the constraints in (1.39) can be replaced by,

$$\left\langle \frac{d_i}{||d_i||}, w \right\rangle \geq 1 \tag{1.47}$$

which results in the standard unbiased SVM problem,

$$\begin{aligned} \bar{w} =& \text{argmin}_w ||w||^2 \\ \text{s.t.} & \left\langle \frac{d_i}{||d_i||}, w \right\rangle \geq 1 \ \ \forall i \in \{1, \cdots, N\}. \end{aligned} \tag{1.48}$$

This is equivalent to (1.39) in the unbounded case except when int $P = \emptyset$; then, since $\bar{p} = 0$, $\bar{w}$ from (1.45) is undefined, but the solution to the SOCP problem (1.40) nevertheless exists.

The *minimax-rate* problem (1.48) differs from the SVM problem considered in, e.g., [17] and [23] by the factor of $\frac{1}{||d_i||}$ included in each constraint. The difference is that whereas the standard SVM approach attempts to classify differences using a maximum-margin separating hyperplane, the minimax-rate approach finds the direction that maximizes the rate of constraint satisfaction; this is shown in Fig. 1.4.

1.6 An Asymptotic Observer for Metric Cost Models

The preference learning methods described so far have been batch processes. Preference data are collected, and then an optimization problem is solved to produce optimal alternatives. Although the convex formulations chosen allow fairly large problems to be solved efficiently in this way, the memory requirements do grow linearly in the number of observations. In this section, instead of a batch algorithm, we consider a stream algorithm that continually adjusts its estimate of the optimal alternative. This algorithm has constant memory and time-per-update requirements, so it is more suitable for use in embedded applications, where, e.g., a small microcontroller can continually adjust system operation to improve subjective performance.

Formally, suppose we have access to a very long (infinite) sequence of comparisons $S = \{(x_k^1, x_k^2)\}_{k=1}^{\infty} = \{s_1, s_2, \cdots\} \subset X \times X$, perhaps as the result of passive monitoring over an extended period of time, and we would like to know the features $\bar{x}$ of the ideal alternative. If alternatives are presented at random to the human, can we construct an asymptotic observer for $\bar{x}$ which can avoid storing all of the very (infinitely) many constraints implied by this sequence? It turns out that the answer is yes, and exactly such an observer is given by,

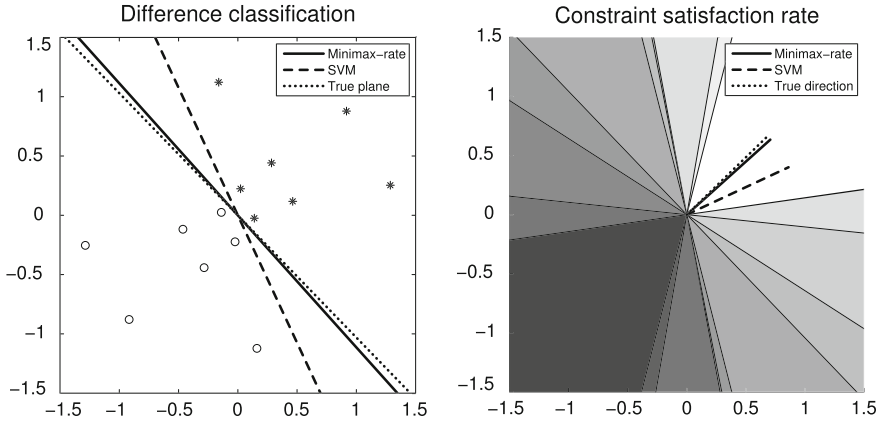


Fig. 1.4 A number of uniformly randomly selected points in $[-1, 1] \times [-1, 1] \subset \mathbb{R}^2$ are compared according to a point at infinity (i.e., a linear cost function) (*dotted*), and both the traditional SVM (*dashed*) and the minimax-rate (*solid*) approaches are used to produce estimates of this direction from the comparisons. From the difference-classification point of view (*top*), one wishes to separate the vectors $\{d_i\}_{i=1}^N$ (displayed as "o"s) from the vectors $\{-d_i\}_{i=1}^N$ (displayed as "*"s). From the minimax-rate point of view (*bottom*), one wishes to find the direction that maximizes the rate of constraint satisfaction (the numbers of violated constraints are represented by shades of *gray*; the *white region* is feasible). The traditional SVM solution separates the positive from the negative differences with a larger margin (*top*), but the minimax-rate solution stays as far from the edge of the constraint cone as possible (*bottom*)

$$\tilde{x}_{k+1} = \begin{cases} P^k \tilde{x}_k + \frac{\alpha_k b_k}{d_k^T d_k} d_k & \text{if } d_k^T \tilde{x}_k - b_k > 0 \\ \tilde{x}_k & \text{otherwise} \end{cases} \tag{1.49}$$

$$P_k = I - \alpha^k \frac{d_k d_k^T}{d_k^T d_k} \tag{1.50}$$

for any sequence of observer gains $\alpha_k \in (0, 2)$ (and d_k, b_k defined by (1.13–1.15)), regardless of $\tilde{x}_0$. That is, $\tilde{x}_k$ converges to $\bar{x}$ in probability as $k \to \infty$, given a few assumptions; we will prove this shortly in Theorem 4. Moreover, note that, although (1.49–1.50) are broken down into separate expressions for clarity of presentation, they are in fact all functions of $\tilde{x}^k$, so this observer can be implemented with only $\dim\{X\}$ real memory elements.

Geometrically, the observer (1.49–1.50) operates through a series of projections (or under/over-projections, if $\alpha_k \neq 1$), as shown in Fig. 1.5, with each projection bringing the estimate $\tilde{x}_k$ of the ideal closer to the true ideal, $\bar{x}$. That the resulting sequence does indeed converge to $\bar{x}$ is guaranteed by Theorem 4, which follows.

Before continuing, we now state a useful lemma, whose geometric interpretation is that comparisons between distances relative to reference points can be interchanged with signed point-plane distance tests.

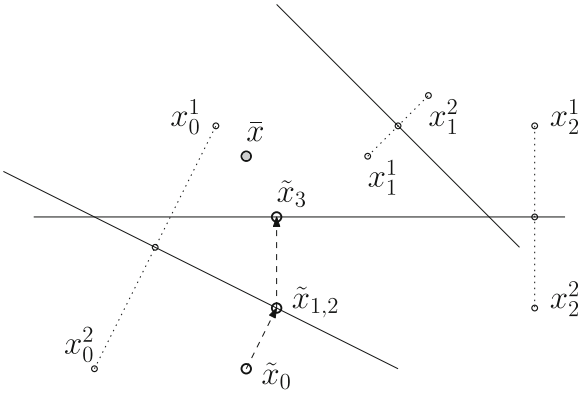


Fig. 1.5 A series of the observer's estimates, with $\alpha_k = 1\ \forall k$. The initial estimate is $\tilde{x}_0$, and the true ideal is given by $\bar{x}$. In step 0, the observer projects $\tilde{x}_0$ onto the plane (*solid line*) corresponding to the measured output $s_0 = (x_0^1, x_0^2)$ to produce $\tilde{x}_1$. In step 1, the observer makes no changes to its estimate, because $\tilde{x}_1$ is on the correct side of the plane corresponding to s_1; hence $\tilde{x}_2 = \tilde{x}_1$. In step 2, the observer projects $\tilde{x}_2$ onto the plane corresponding to s_2 to create the estimate $\tilde{x}_3$, which is yet closer to $\bar{x}$

Lemma 1. Let $x^1, x^2, \bar{x}$ be any vectors in an inner product space $(X, \langle\cdot,\cdot\rangle)$, and let $\bowtie$ be a binary relation from the set, $\{=, <, >, \leq, \geq\}$. Then,

$$\bar{x} \in \{x \mid \langle d, x\rangle - b \bowtie 0\} \Longleftrightarrow ||x^1 - \bar{x}|| \bowtie ||x^2 - \bar{x}||$$

where $d = x^2 - x^1$, and $b = \frac{1}{2}\langle d, x^1 + x^2\rangle$.

Proof. The proof of this is based on the Polarization Identity and is straightforward.

Theorem 4. *Let $\bar{x} \in X$ be the ideal alternative, and $S = \{(x_k^1, x_k^2)\}_{k=1}^{\infty} = \{s_1, s_2, \cdots\}$ a sequence of pairs of i.i.d. random vectors drawn according to a probability density function p on $\{(x^1, x^2) \in X \mid ||x^1 - \bar{x}|| < ||x^2 - \bar{x}||\}$ which is nonzero in an open ball $B(\bar{x}, r) = B_r$ around $\bar{x}$. Then, the asymptotic observer given by* (1.49–1.50) *converges to $\bar{x}$ in probability.*

Proof. 1. *If $\langle d_k, \tilde{x}_k\rangle - b_k > 0$, then $||\tilde{x}_{k+1} - \bar{x}|| < ||\tilde{x}_k - \bar{x}||$.* The distances $||\tilde{x}_k - \bar{x}||$ and $||\tilde{x}_{k+1} - \bar{x}||$ are related through the Polarization Identity by (where $\Delta_k = \tilde{x}_{k+1} - \tilde{x}_k$),

$$||\tilde{x}_{k+1} - \bar{x}||^2 = ||\tilde{x}_k + \Delta_k - \bar{x}||^2 =$$
$$||\tilde{x}_k - \bar{x}||^2 + ||\Delta_k||^2 + 2\langle \tilde{x}_k - \bar{x}, \Delta_k\rangle$$

so, it order to show that $||\tilde{x}_{k+1} - \bar{x}|| < ||\tilde{x}_k - \bar{x}||$, it is sufficient to demonstrate

$$||\Delta_k||^2 + 2\langle \tilde{x}_k - \bar{x}, \Delta_k\rangle < 0. \tag{1.51}$$

From (1.49, 1.50),

$$\begin{aligned}\Delta_k &= \left(I - \alpha^k \frac{d_k d_k^T}{d_k^T d_k}\right)\tilde{x}_k + \frac{\alpha_k b_k}{d_k^T d_k} d_k - \tilde{x}_k \\ &= \frac{\alpha}{\langle d_k, d_k\rangle}\left(b_k - \langle d_k, \tilde{x}_k\rangle\right) d_k \end{aligned} \tag{1.52}$$

so, substituting Δ_k into (1.51) (and dropping the superscript indices k),

$$\frac{\alpha^2}{\langle d, d\rangle}\left(b - \langle d, \tilde{x}\rangle\right)^2 + 2\frac{\alpha}{\langle d, d\rangle}\left(b - \langle d, \tilde{x}\rangle\right)\langle d, \tilde{x} - \bar{x}\rangle < 0$$

or equivalently, so long as $\alpha > 0$ (as we require),

$$-\left(\langle d, \tilde{x}\rangle - b\right)\left[\alpha\left(b - \langle d, \tilde{x}\rangle\right) + 2\langle d, \tilde{x} - \bar{x}\rangle\right] < 0. \tag{1.53}$$

Since by assumption $\langle d, \tilde{x}\rangle - b > 0$, this is satisfied iff the second factor is positive; that is,

$$\begin{aligned}&\alpha\left(b - \langle d, \tilde{x}\rangle\right) + 2\langle d, \tilde{x} - \bar{x}\rangle = \\ &\alpha b + (2 - \alpha)\langle d, \tilde{x}\rangle - 2\langle d, \bar{x}\rangle > 0.\end{aligned} \tag{1.54}$$

Since $\langle d, \tilde{x}\rangle > b$, and by Lemma 1, $\langle d, \bar{x}\rangle \leq b$, this is satisfied so long as $\alpha \in (0, 2)$, as we require.

2. *The sequence* $e_k = ||\tilde{x}_k - \bar{x}_k||,\ k = 0, 1, 2, \ldots$ *is nonincreasing.* In the second case of (1.49), $\tilde{x}_{k+1} = \tilde{x}_k$; this is nonincreasing. In the first case, $\langle d_k, \tilde{x}_k\rangle - b_k > 0$, so $e_{k+1} < e_k$ by point 1 above.
3. g.l.b.$(e_k) = 0$ *with unit probability.* By positivity of $||\cdot||$, zero is a lower bound. To show that this is the greatest such bound, consider some $\epsilon > 0$ and suppose that, at iteration m, $||\tilde{x}_m - \bar{x}|| = \epsilon$. Now, let $z = \min(r, \epsilon/2)$, and consider the open balls $B_1 = B(c_1, z/4)$, $B_2 = B(c_2, z/4)$, where the center points c_1, c_2 are defined,

$$c_j = \bar{x} + \frac{\tilde{x} - \bar{x}}{||\tilde{x} - \bar{x}||}\frac{(2j - 1)}{4}z$$

for $j \in \{1, 2\}$ (see Fig. 1.6); additionally, let $x^1 \in B_1$, $x^2 \in B_2$. Then by Lemma 1, we can confirm that $\bar{x}$ and $\tilde{x}$ are on opposite sides of the plane corresponding to (x^1, x^2) (and hence, that a projection will occur) by verifying that,

$$||x^2 - \tilde{x}|| < ||x^1 - \tilde{x}|| \tag{1.55}$$

$$||x^2 - \bar{x}|| > ||x^1 - \bar{x}||. \tag{1.56}$$

Considering the first of these, we note by the triangle inequality,

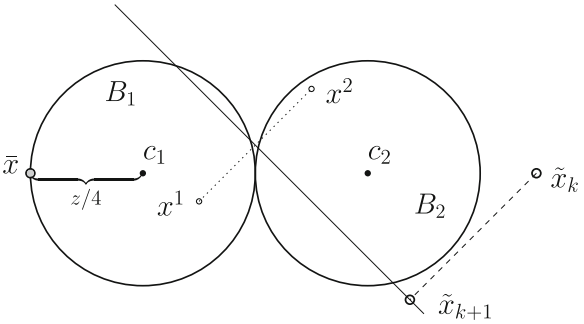


Fig. 1.6 If $x^1 \in B_1$ and $x^2 \in B_2$, then $||\tilde{x}_{k+1} - \bar{x}|| < ||\tilde{x}_k - \bar{x}||$

$$||x^2 - \tilde{x}|| \leq ||x^2 - c_2|| + ||c_2 - \tilde{x}|| < \frac{1}{4}z + ||c_2 - \tilde{x}||$$

whereas, by the inverse triangle inequality,

$$\begin{aligned} ||x^1 - \tilde{x}|| &\geq | \, ||x^1 - c_1|| + ||c_1 - \tilde{x}|| \, | \\ &\geq ||c_1 - \tilde{x}|| = \frac{1}{2}z + ||x^2 - c_2|| \end{aligned}$$

so this is indeed the case. Considering the second inequality (1.56), we have likewise,

$$||x^1 - \bar{x}|| \leq ||x^1 - c_1|| + ||c_1 - \bar{x}|| < \frac{1}{4}z + \frac{1}{4}z = \frac{1}{2}z$$

and

$$||x^2 - \bar{x}|| \geq | \, ||x^2 - c_2|| - ||c_2 - \bar{x}|| \, | \geq \frac{3}{4}z$$

so this inequality holds as well. Therefore, *any* x^1, x^2 from B_1, B_2 are associated with a plane that separates $\tilde{x}$ from $\bar{x}$ and hence triggers a projection. Since B_1 and B_2 have nonzero measure, and are subsets of B_r in which $p(\cdot)$ is nonzero, then the probabilities for this iteration $P_1 = \Pr(\text{“a point is selected in } B_1\text{”})$ and $P_2 = \Pr(\text{“a point is selected in } B_2\text{”})$ are both nonzero, and therefore, since the s^k are independent, $P_{\text{both}} = \Pr(\text{“one point is selected in } B_1 \text{ and the other is selected in } B_2\text{”}) = P_1 P_2$ is nonzero, and the probability that this occurs for *at least* one iteration $k > m$ is given by $1 - \prod_{k=m}^{\infty} \left(1 - P_{\text{both}}^k\right) = 1$ or in other words, with probability one, there exists a $q > m$ such that $\langle d_q, \tilde{x}_q \rangle - b_q > 0$. Then, by point 1, $||\tilde{x}_q - \bar{x}|| < ||\tilde{x}_m - \bar{x}|| = \epsilon$, and so ϵ, with unit probability, cannot be a lower bound. Since e_k is a nonincreasing sequence in $\mathbb{R}$ and g.l.b.$(e_k) = 0$, e_k converges to 0 and thus $\tilde{x}$ converges to $\bar{x}$ in probability.

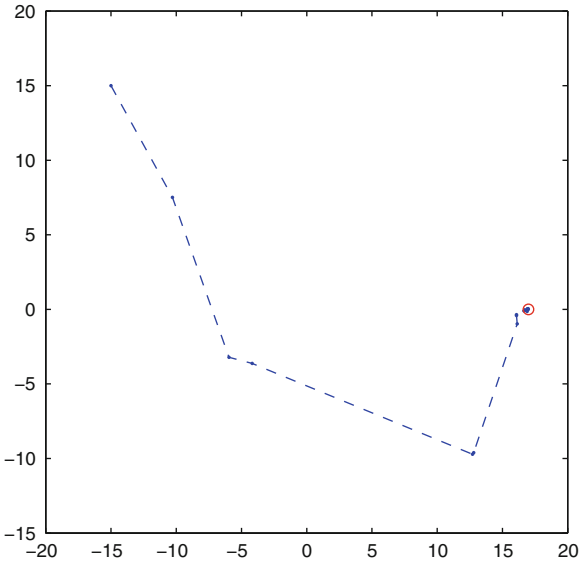


Fig. 1.7 Example estimate trajectory for observer (1.49–1.50) for $\alpha^k = \alpha = 1$, with $X = \mathbb{R}^2$. The estimate begins at $\tilde{x}_0 = (-15, 15)$, and approaches the ideal $\bar{x} = (17, 0)$

An example of the estimate trajectory in feature space generated by such an observer is shown in Fig. 1.7. For this example, $X = \mathbb{R}^2$, and features were drawn from a uniform distribution in the square $[-20, 20] \times [-20, 20]$. The estimate evolves from its initial condition, $\tilde{x}_0 = (-15, 15)$ to near the ideal $\bar{x} = (17, 0)$.

1.7 Applications

1.7.1 Apples and Oranges

To demonstrate the application of the metric preference learning formulation, photos of nine apples were shown to an audience of 13 people in a number of pairwise experiments. (The fruit is shown in Fig. 1.8.)

Each apple was described by a 15-dimensional feature vector, containing (1–3) the average color in HSB (hue, saturation, brightness) color space, (4–6) the average color in RGB color space, (7) the color variance, (8–10) width, height, and the ratio of the two, (11–12) stem length, and angle relative to apple, (13–14) dimple angle and depth, and (15) roundness. This represents a collection of many conceivable characteristics of an apple that may make it more or less orangelike, and the idea is to learn a cost function without making assumptions a priori about which of these features are most salient.

Fig. 1.8 Depicted are the nine apples used to generate comparisons with the single orange

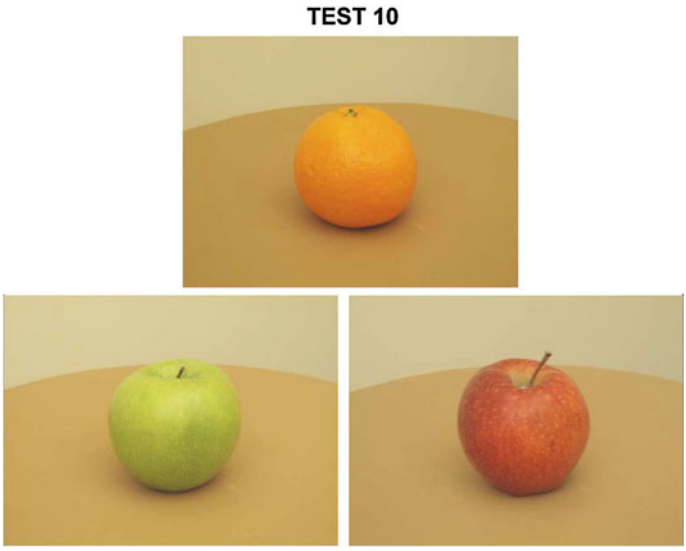


Fig. 1.9 An example of a pairwise comparison between two apples, relative to the orange

The partial order over the apples was thus generated by having a number of people make a number of randomly selected, pairwise comparisons (as the one shown in Fig. 1.9). Represented as a preference graph, the results of these experiments are shown in Fig. 1.10.

For these data, the minimization problem (1.21) is unbounded and hence we find an optimal direction via (1.19). Solving (1.40), we obtain the optimum,

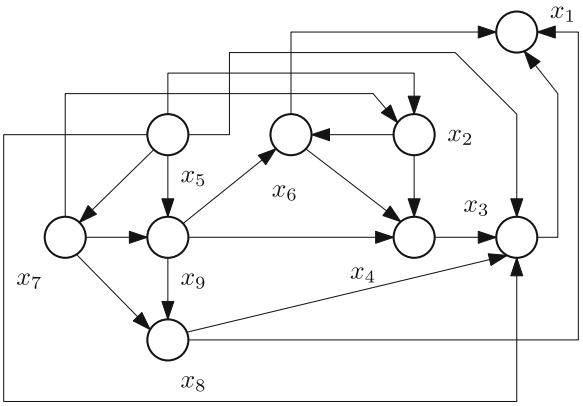


Fig. 1.10 The preference graph corresponding to the apple experiments

$\bar{v}_1 = -0.0252$	(Hue)	$\bar{v}_9 = -0.2380$	(Height)
$\bar{v}_2 = -0.0844$	(Saturation)	$\bar{v}_{10} = -0.0472$	(Width/Height)
$\bar{v}_3 = 0.1374$	(Brightness)	$\bar{v}_{11} = -0.0409$	(Stem Length)
$\bar{v}_4 = 0.3572$	(Red)	$\bar{v}_{12} = -0.5017$	(Stem Angle)
$\bar{v}_5 = 0.1137$	(Green)	$\bar{v}_{13} = 0.6683$	(Dimple Angle)
$\bar{v}_6 = 0.1856$	(Blue)	$\bar{v}_{14} = 0.0996$	(Dimple Depth)
$\bar{v}_7 = 0.0442$	(Variance)	$\bar{v}_{15} = -0.0472$	(Roundness)
$\bar{v}_8 = -0.1593$	(Width)		

which has the interpretation that dimple angle and redness are important orangelike qualities, and that large stem angles are perceived as un-orangelike.

1.7.2 Amoebas and Humans

To understand the comparison of higher dimensional objects and in particular motions, another experiment was performed in which an audience of 25 people was asked to perform pairwise comparisons of different motions of a computer-animated amoeba, relative to the motion-captured movement of a human who danced the bhangra (Fig. 1.11). An example of one such question is shown in Fig. 1.12. In this manner, a preference graph was generated as before, with 12 vertices (the amoeba motions) and 20 edges; this is shown in Fig. 1.11.

Inner products between the various amoeba motions were computed by rasterizing the motions to binary videos, blurring each frame of the result, and computing the standard Euclidean inner product of these (extremely large) `[Frame Width]×[Frame Height]×[Number of Frames]`-dimensional vectors. The resulting inner product is a relaxation of set overlap area, and its corresponding metric returns small distances between videos in which mass is located at nearly, but

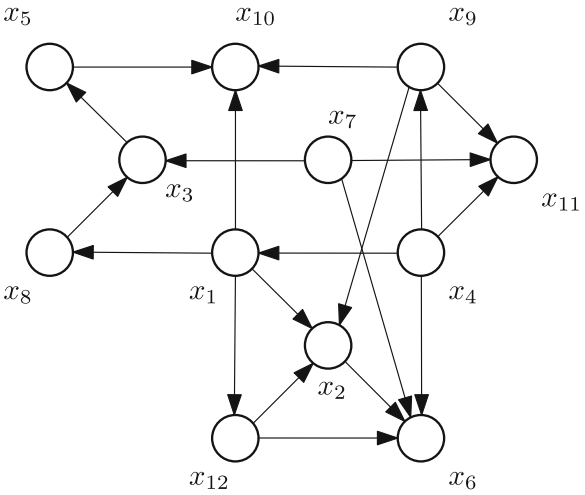


Fig. 1.11 The DAG corresponding to the amoeba experiments

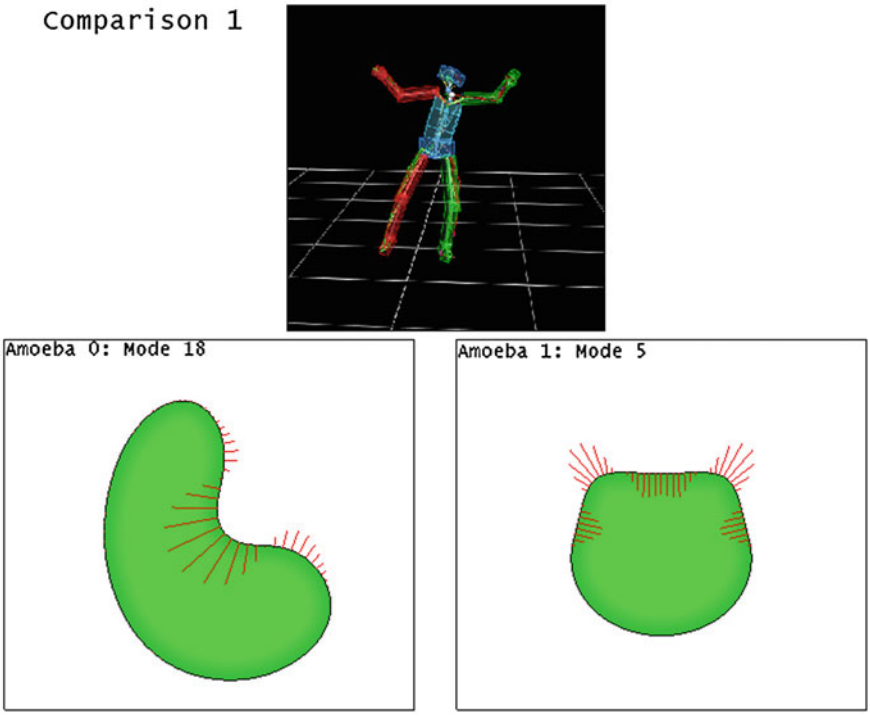


Fig. 1.12 Each question took the form, "Which of the two 'amoebas' (*bottom*) looks more like the [motion capture data from a] human dancer (*top*)?"

not necessarily exactly, the same locations at the same times. We note that the sheer size of the rasterized representation highlights the advantage of the instance vector expansion described in Sect. 1.5.1, without which the optimization problem simply could not be realistically solved.

The minimization problem (1.21) with the resulting data turns out to be unbounded and hence we again find an optimal direction via (1.19). We obtain the coefficient expansion for the optimal direction,

$$\bar{v} = \sum_{k=1}^{M} \xi_k x_k \tag{1.57}$$

where

$$\begin{aligned} \xi = 10^3 (& 1.4918, \quad -3.6556, \quad -0.1390, \quad 0.3113, \\ & -1.1243, -0.1771, \quad 2.6335, \quad 0.5878, \\ & 1.8362, \quad -1.7319, \quad -0.2999, \quad 0.2672). \end{aligned}$$

What this means is that, in order to look as much like it is dancing the bhangra as possible, an amoeba should as its first priority aspire to be as much like amoeba 7 ($\xi_7 = 2.6335$) and as dissimilar from amoeba 2 ($\xi_2 = -3.6556$) as possible, and that it should to a lesser extent model itself after amoebas 1 and 9 ($\xi_1 = 1.4918$, $\xi_9 = 1.8362$) while avoiding the esthetically unappealing moves of amoebas 5 and 10 ($\xi_5 = -1.1243$, $\xi_{10} = -1.7319$). Although this does not explain why, psychologically, e.g., amoeba 7—which pulses with two upward-pointing, armlike protrusions—is preferred to amoeba 2—in which a bulge on one side moves in and out—it does produce both a consistent cost structure, and an estimate for an amoeba motion that will be preferred to all others in the larger space of motions.

1.8 Concluding Remarks

In this chapter, we investigated the problem of motion preference learning under the assumption of an underlying *metric cost model*; here, the alternatives being compared are points in a metric space, and human judges are assumed to prefer one point to another if and only if it is closer to some fixed but unknown best alternative that they may not have been shown. This assumption appears to be a good one for the examples considered and the features chosen, in that the feasible set P in this case is nonempty.

Based on the metric cost assumption, a Chebyshev estimator was given for the best point for the case when P is bounded, and a natural generalization, the minimax-rate estimator, was developed for when P is unbounded. In the first case, the solution was found, with an efficiency rivaling standard quadratic SVMs, as the solution to a linear program; and in the second case the problem was shown to in fact reduce a particular SVM classification problem.

In order that the estimators for the bounded and unbounded cases be applicable to situations in which the compared alternatives inhabit high- or infinite-dimensional metric spaces, the optimization problems were additionally given in an instance vector expansion form, which results in optimization problems whose size is proportional not to the dimensionality of the metric space, but only to the number of comparisons available. This is particularly relevant in a controls context, when the alternatives being compared are the signals input to or produced by continuous-time systems, which inhabit infinite-dimensional function spaces.

Finally, for the case when a large amount of data are available, or stream rather than batch processing is desired, a limited memory asymptotic observer was given that avoids the need to store all the (infinitely) many constraints.

In all cases, optimal cost functions *and* points/directions were found efficiently by convex programming. The result is an efficient minimax estimator for the best possible alternative.

Acknowledgments This work was supported by the U.S. National Science Foundation through Creative IT Grant #0757317. The human-study was conducted within the Georgia Institute of Technology, Institute Review Board Protocol H08162 - "Perceived Similarity Study." We would like to thank Akhil Bahl for his assistance in producing the motion capture data used in the synthetic amoeba experiment.

References

1. Abbeel P, Ng AY (2004) Apprenticeship learning via inverse reinforcement learning. In: Proceedings of the international conference on machine learning
2. Agarwal S (2006) Ranking on graph data. ICML,ACM Press, pp 25–36
3. Aho AV, Garey MR, Ullman JD (1972) The transitive reduction of a directed graph. SIAM J Comput 1:131–137
4. Aiolli F, Sperduti A (2004) Learning preferences for multiclass problems. Advances in neural information processing systems 17, MIT Press, pp 17–24
5. Aizerman M, Braverman E, Rozonoer L (1964) Theoretical foundations of the potential function method in pattern recognition learning. Autom Remote Control 25:821–837
6. Bahamonde A, Diez J, Quevedo J, Luances O, del Coz J (2007) How to learn consumer preferences from the analysis of sensory data by means of support vector machines. Trends Food Sci Technol 18:20–28
7. Baird B (1965) The art of the puppet. Mcmillan Company, New York
8. Black J, Iyasere E, Wagner J (2011) Creation of a driver preference objective metric to evaluate ground vehicle steering systems. In: American control conference
9. Boser B, Guyon I, Vapnik V (1992) An training algorithm for optimal margin classiers. In: Proceedings of the fifth annual workshop on computational learning theory, pp 144–152
10. Callahan P (1994) Puppets, stop the flap!: the Arpeggio method of mouth puppet manipulation. Arpeggio, Atlanta
11. Chu W, Ghahramani Z (2005) Preference learning with gaussian processes. In: Proceedings of the 22nd international conference on machine learning, New York, pp 137–144
12. Chu W, Keerthi SS (2007) Support vector ordinal regression. Neural Comput 15(9):2227–2254
13. Cohen WW, Schapire RE, Singer Y (1999) Learning to order things. J Artif Intell Res 10:243–270

14. Coppersmith D, Winograd S (1987) Matrix multiplication via arithmetic progressions. In STOC '87: Proceedings of the nineteenth annual ACM symposium on theory of computing, New York, pp 1–6
15. Cortes C, Mohri M, Rastogi A (2007) Magnitude-preserving ranking algorithms. In: Proceedings of the twenty-fourth international conference on machine learning
16. Canudas de Wit C (2005) Fun-to-drive by feedback. Eur J Control: Fundam issues Control 11/4–5:353–383
17. Diez J, Jose del Coz J, Luaces O, Bahamonde A (2008) Clustering people according to their preference criteria. Expert Syst Appl 34:1274–1284
18. Egerstedt M, Murphey T, Ludwig J (2007) Motion Programs for Puppet Choreography and Control. Springer-Verlag, Berlin, pp 190–202
19. Engler L, Fijan C (1973) Making puppets come alive. Taplinger Publishing Company, New York
20. Fiechter CN, Rogers S (2000) Learning subjective functions with large margins. In: Proceedings of the seventeenth international conference on machine learning, pp 287–294
21. Fischer MJ, Meyer AR (1971) Boolean matrix multiplication and transitive closure. In: Twelfth annual symposium on switching and automata theory, pp 129–131
22. Herbrich R, Graepel T, Bollmann-Sdorra P, Obermayer K (1998) Supervised learning of preference relations. In: Proceedings FGML-98, German national workshop on machine learning, pp 43–47
23. Herbrich R, Graepel T, Obermayer K (2000) Large margin rank boundaries for ordinal regression. Advances in large margin classifiers. MIT Press, Cambridge
24. Hullermeier E, Frankranz J, Cheng W, Brinker K (2008) Label ranking by learning pair-wise preferences. Artif Intell 172:1897–1917
25. Jiang X, Lim L-H, Yao Y, Ye Y (2011) Statistical ranking and combinatorial hodge theory. Math Program 127:203–244
26. Joachims T (2002) Optimizing search engines using click-through data. In: ACM SIGKDD conference on knowledge discovery and data mining (KDD), pp 133–142
27. Johnson E, Murphey T (2007) Dynamic modeling and motion planning for marionettes: rigid bodies articulated by massless strings. In: IEEE international conference on robotics and automation, pp 330–335
28. Kingston P, Egerstedt M (2009) Comparing apples and oranges through partial orders: an empirical approach, In: American control conference
29. Kingston P, Egerstedt M (2011) Motion preference learning. In: American control conference
30. Munro I (1971) Efficient determination of the strongly connected components and the transitive closure of a graph
31. Platt JC (1998) Sequential minimal optimization: a fast algorithm for training support vector machines. Advances in kernel methods. MIT Press, Cambridge
32. Qiao Q, Beling PA (2011) Inverse reinforcement learning with gaussian process. In: American control conference
33. Roser A (1979) Gustaf Und Sein Ensemble: Beschreibungen eines Puppenspielers. Bleicher Verlag, Gerlingen
34. Strassen V (1969) Gaussian elimination is not optimal. Numer Math 13:354–356
35. Syed U, Bowling M, Schapire RE (2008) Apprenticeship learning using linear programming. In: Proceedings of the international conference on machine learning

Chapter 2
In the Dance Studio: An Art and Engineering Exploration of Human Flocking

Naomi E. Leonard, George F. Young, Kelsey Hochgraf, Daniel T. Swain, Aaron Trippe, Willa Chen, Katherine Fitch and Susan Marshall

2.1 Flock Logic

The *Flock Logic* project [1] was conceived by engineering professor/control theorist Naomi Leonard and dance professor/choreographer Susan Marshall. Marshall approached Leonard in 2010 about the possibility of an investigation to find out what would happen if a group of dancers were in possession of the rules governing the motion of groups of flocking animals. The project was initiated as a joint exploration with professional dancers in July 2010 and then subsequently as a semester-long Princeton University Atelier course co-taught by Leonard and Marshall in Fall 2010. The professional dancers and the students, most of whom had previous dance training, participated in collaborative artistic and scientific investigations and experiments inspired by the complex and beautiful group motion that emerges in bird flocks and fish schools. The aim was to explore artistically and scientifically how individual rules of interaction and response within a network of dancers yield complex emergent collective motion of the group.

The emergent nature of flocking and schooling was a central driver for the project: the remarkable collective motion of flocks and schools results not from a prescribed choreography nor even from a designated leader, but rather from simple rules of response that each individual obeys [2]. These feedback rules govern how each individual moves in response to the relative position or motion of its close neighbors. For instance, basic flocking rules typically have a cohesive element and a repulsive

This effort was supported in part by Princeton University's Essig Enright Fund, Lewis Center for the Arts, Keller Center for Innovation in Engineering Education, and Mechanical and Aerospace Engineering Department, and by NSF grant ECCS-1135724, AFOSR grant FA9550-07-1-0-0528 and ONR grant N00014-09-1-1074.

N. E. Leonard (✉) · G. F. Young · K. Hochgraf · D. T. Swain · A. Trippe · W. Chen · K. Fitch · S. Marshall
Princeton University, Princeton, NJ 08544, USA
e-mail: naomi@princeton.edu

A. LaViers and M. Egerstedt (eds.), *Controls and Art*,
DOI: 10.1007/978-3-319-03904-6_2,

element [3]. The cohesive element requires that while each individual moves around it should remain a comfortable distance from a few others; the repulsive element requires that each individual should move away from others that get too close. An active area of research is focused on explaining how the observed complex collective motion of animal groups emerges from, and is influenced by, the feedback rules, the dynamics of the social interactions within the group, the distribution of information across the group, the features in the spatial surrounding, the differences among individuals, the noise in measurements, and the uncertainty in decision making [4, 5]. Analytical and numerical studies and laboratory and field experiments have all been used to investigate; for example, see [6–8] for a range of studies on flocking of starlings.

Flock Logic explored what happens when a group of *dancers* apply these and related feedback laws as they move around a space together. In the Flock Logic explorations the flocking rules were prescribed, but neither how the dancers applied the rules nor how faithfully they followed the rules were controlled. For example, the number of neighbors and the distance from neighbors to maintain in the coherence rule were prescribed, but the dancers were not instructed how to choose with whom to cohere, how to prioritize among neighbors moving in diverging directions, nor how to handle conflicts such as when cohering with one dancer meant getting too close to another dancer. It was also possible that dancers broke the rules at times. Thus, the emergent human flocking resulted from both prescribed and individualized, and thus unknown, features of dancers' choices and dynamics. In this way, the Flock Logic project provided a framework for exploring emergent collective behavior somewhere between studying animal aggregations in the wild, with all that is unknown, and examining computer simulated flocking, with its exclusive reliance on a prescription.

This aspect of Flock Logic made it particularly well suited to an integrated art, engineering, and science agenda. On the one hand, the Flock Logic framework made it possible to observe the influence on collective motion of natural biases, in this case human biases, and heterogeneity across the group. Dancers with different physical features, personalities, dance training, etc., would respond differently to one another and would prioritize rules and resolve conflicts differently. This would affect how information would pass through the group and how the group as a whole would respond to external forces. On the other hand, the Flock Logic framework made it possible to systematically examine the influence on collective motion of parameters of the prescribed rules. This applied to the parameters of rules and environments meant to represent animal groups, e.g., number of others with whom to cohere, total population of the group, availability of information or preferences across the group, as well as the shape, size, and placement of obstacles. This also applied to the parameters of rules and environments not necessarily intended to represent animal groups but rather motivated by artistic and engineering design goals. By varying rules and environmental features beyond what one would expect in animal groups, it was possible to explore how individual-level behaviors connect more generally to the aesthetics and the functionality of the emergent group-level behaviors. And this led to the creation of a wide range of artistic and engineering design possibilities.

The engineering goal was to use the explorations with dancers to gain insight into the mechanisms of animal group and human crowd dynamics and into design principles for control of natural and robotic groups. Could the dance studio be viewed as an experimental test-bed in this regard? Could the human data collected be used to help explain a range of collective behaviors? The dancers could represent a human crowd moving in a bounded space, trying to avoid colliding. Likewise, dancers moving in a studio, responding to local neighbors and the environment, provide a reasonable approximation to the collective motion of a herd. The walls of the studio are like trees or topography, and the heterogeneity among the dancers (experience, height, confidence) is similar to that in a herd [9]. Further, dancers are particularly well suited to these kinds of explorations because they are trained to be physically aware and can comfortably handle a number of feedback rules. Thus, the setting provided enormous flexibility in the kinds of questions that could be addressed. For example, in this chapter, the human motion data are used to rigorously study how influence among individuals in the network is distributed and how that is reflected in the changing spatial distribution of individuals and in the group-level shape and motion dynamics. This could, for example, lead to insights on how human crowds move in cluttered spaces and how animals organize themselves to reduce vulnerability to predators [10]. This could also lead to bioinspired methods for designing robust and responsive networks of heterogeneous robots [11].

There are a number of motivating and complementary scientific studies of human collective motion, many of which focus on crowd dynamics. Experiments on leadership and decision making in human crowds were described in Dyer et al. [12]. In Moussaid et al. [13], analysis of natural pedestrian group motion revealed the influence of social interactions on crowd dynamics. In Funes et al. [14] a design method for human collective behaviors used evolutionary dynamics. Simplified models described in Silverberg et al. [15] predicted collective behavior of humans in mosh and circle pits as observed from video data of heavy metal concerts. Altenberg [16] developed a set of rule-based movement experiments as a concrete way of teaching emergence.

In Flock Logic, the engineering goal was tightly integrated with the artistic goal, where the idea was to see how applying work on decentralized control of collective motion, to dancers, could potentially result in choreographic tools or training tools for developing individual and group awareness. It was also imagined that a site-specific large group performance work could be developed with little more than a site and the rules.

The artistic interest centered on a desire to translate flocking rules such as those related to group cohesion and response to external pressures into improvisational instructions for dancers. Could these rules support unexpected and complexly orchestrated collective motion to emerge from individual interactions? How might the local sensing rules be altered choreographically to make emergent choreography that didn't resemble the familiar look of organic flocking? Could these rules be learned quickly by nondancers to create a kind of flash mob performance?

In theater and dance, there is a long history of movement practice and performance based on structured improvisation and rules and games [17]. Contemporary chore-

ographer Forsythe has studied synchrony and pattern in dance, as in, for example, "Synchronous Objects" [18]. Choreographer Sgorbati has explored dance through "Emergent Improvisation," which is modeled after ordering principles observed in nature [19]. Hagendoorn [20] designed rules for dancers to explore complexity, emergent patterns and emergent choreography. Carlson [21] introduced constraints in movement generation to study creative decision making in choreographic practice.

Often in improvisational dance work, the individual has a wide range of choices open to them and takes compositional responsibility for the entire stage as well as their own body. The Flock Logic rules tend to limit the individual's choices to their immediate neighbors and to ask the individual to relinquish group choreographic responsibility; nonetheless, rich group choreography results. Could rules be designed that would allow dancers, ignorant of any overarching choreographic goals, to create complex and organized patterns using these tools?

In Flock Logic, to generate human flocking, the dancers were asked to move about a space and follow rules that were defined in advance. To enable *cohesion*, each dancer was given the rule to keep m of their neighbors at a distance of arm's length with the selection of the m neighbors freely changeable. To enable *repulsion*, each dancer was asked to avoid letting any dancer get closer than arm's length. To prevent tripping, the dancers were asked to avoid moving backwards.

These three rules (cohesion, repulsion, backwards avoidance) were among the most fundamental rules examined. Variations on the three fundamental rules were prescribed as well as a range of additional and alternative rules. For example, rules for alignment with neighbors, response to obstacles and walls, options to initiate or imitate specific movements, etc., were implemented. More complex informational structures were imposed—for example, two or three dancers in the group were secretly given additional rules, such as to move to a particular location or according to a particular pattern. The dancers also performed rules for other kinds of behaviors such as dynamic coverage and pursuit and evasion. In part because each dancer's motion was relatively under-prescribed, there was considerable room for variation among individuals, e.g., in speed, facing direction relative to motion, selection of neighbors, positioning relative to neighbors, and response to walls or obstacles.

Complex and artistically satisfying collective behaviors were routinely observed. As part of the 2010 Princeton University Atelier course, approximately 50 volunteers participated in two flocking performances, each at a different site, after having been briefly instructed in a few local rules of cohesion, repulsion and alignment as well as responses to obstacles, to walls, and to "predators." From both artistic and engineering perspectives, these were highly satisfying performances. A snapshot from one of the performances is shown in Fig. 2.1. Video clips from the events are publicly available and can be accessed from the Flock Logic website [1] or directly at the following links:
http://vimeo.com/19361231 (Peter Richards);
http://www.princeton.edu/main/news/archive/S29/62/38K80/ (Evelyn Tu);
http://www.youtube.com/watch?v=Mg29hawdcMw (Jeffrey Kuperman).

In this chapter, we describe the Flock Logic explorations and the tools used for our artistic and engineering investigations. As an illustration, we examine one experiment

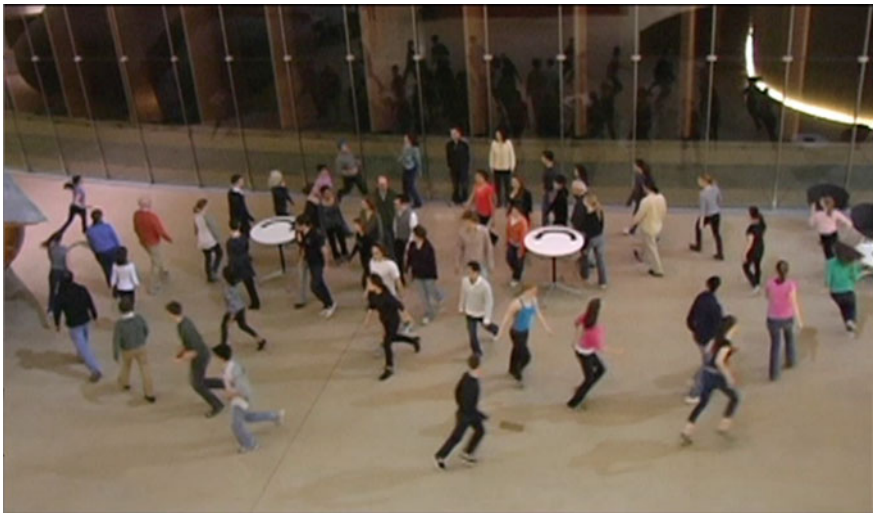


Fig. 2.1 Snapshot from a Flock logic performance at Princeton University in December 2010

with 13 dancers who followed the flocking rules of cohesion and repulsion as they moved around the dance studio. Using the trajectories tracked from an overhead video camera and the prescribed interaction rules, we estimate the time-varying graph that encodes who is sensing whom as a function of time. We compute the time-varying status of each node in the graph, defining how much attention a dancer receives from the rest of the dancers, and use these to infer emergent leaders. We discuss implications, open questions, and further directions both artistic and scientific.

The work described in this chapter connects with the work described in several other chapters in this collection. In Chap. 7, Heupe et al. [22], similarly inspired, investigate how flocking dynamics can be used to generate music, and they use performance to explore the rich interplay between coherence and decoherence. In Chap. 9, LaViers et al. [23] also use performance to study dance and address engineering design questions similar to ours by investigating how to translate information about human dance styles into design for robotic motion. Although in our work, we only track dancers offline after an event, real-time tracking of humans offers the potential for more complex interactions between humans and machines, as shown in Chap. 8 by Godbehere and Goldberg [24]. The work in Chap. 6 of Tsiotras and Castro [25] further demonstrates the richness of multiagent geometries that can result from modifications to the basic rules of flocking.

In Sect. 2.2, we describe our human flocking explorations, including our on-line FlockMaker software tool, and the human flocking experiments. Trajectory tracking is described in Sect. 2.3. In Sect. 2.4, we review graphs and FlockGrapher, our tool for visualizing graphs. In Sect. 2.5, we estimate the time-varying graph of the network. In Sect. 2.6, we estimate node status and discuss the influence of individuals. We

conclude with a discussion of the results and a reflection on further artistic and engineering opportunities that build on the Flock Logic project in Sect. 2.7.

An earlier version of this paper appeared in the proceedings of the American Control Conference, held in Montreal, Canada in June 2012 [26]. At this same conference, a special interactive session was held in which 100 conference attendees participated in a human flocking performance event. For this event, original accompanying music was composed and performed live by Cristián Huepe.

2.2 Human Flocking

2.2.1 Explorations

A typical Flock Logic exploration involved on the order of 10–15 dancers who moved around the dance studio for a few minutes applying flocking rules prescribed in advance. Many of these explorations were run in series during a single session, with a wide variety of flocking rules prescribed. The dancers were given frequent opportunities to watch the group from the outside, and to discuss how it felt from within the group and how it looked from without the group. The process was highly collaborative: dancers made suggestions routinely and during a number of sessions small groups of dancers would design a set of rules for themselves and for the rest of the group.

The explorations evolved over time as the dancers gained more experience with moving according to the rules of flocking. This meant that level of experience played into the emergent collective motion, especially later in the semester when volunteers were briefly "trained" and joined the group for flocking. By adding dancers to the group, the role of the number of dancers was also explored. As many as 24 dancers participated in some of the experiments in the studio, described in Sect. 2.2.3. In one of the Flock Logic performances in December 2010 , more than 50 people were involved and in later flocking events, such as at the special session in Montreal at the American Control Conference, as many as 100 people participated. Sessions were also held outdoors which provided the opportunity for explorations in a space without boundaries.

The basic flocking rules of cohesion, which meant keeping m neighbors at arm's length, repulsion, which meant moving away from anyone closer than arm's length, and backwards avoidance, formed the basis of many explorations. Two-person cohesion ($m = 2$) was enough to create what looked like a planar school or flock. With $m = 2$ the dancers were regularly spaced and exhibited polarized motion, i.e., everyone moving together in a single direction, as well as circular motion, i.e., the group moving around a circle. The circular motion sometimes drifted and sometimes remained fixed about a single stationary dancer. The group also experienced fissions and fusions as well as significant changes in momentum. When m was decreased to

$m = 1$, i.e., one-person cohesion, the result was a lot more "parading" and much less of the distributed look of a natural flock.

Explorations made use of the presence of walls and the introduction of obstacles. With no special rules attached to walls or obstacles, the dancers treated them much as animals would likely treat them, i.e., they deftly avoided colliding with them. In one case, a row of chairs was extended in a line from one wall into the middle of the room. When the dancers moved into the smaller space created by the chairs and the parallel wall, they would remain there temporarily, as if caught in a tidal pool, and only move out once they had reversed direction. When rules were prescribed with respect to walls and obstacles, all sorts of interesting, and less biologically motivated, collective behavior emerged. For example, the walls were given an attractive pull as well as a "stickiness." As a result, dancers getting close to a wall got pulled away from the group and stuck there (see Fig. 2.1). They were released from the wall by cohesion to other dancers when at some later point the group passed by them. This looked like the peeling off and adhering back of dancers in an ordered way since these dancers still applied the cohesion and repulsion rules among themselves.

In another case, a round table was moved into the open space and endowed with the properties of a sling shot. When dancers got close to the table they would circle around it in a fixed direction, e.g., counter-clockwise, at an increased speed and then get "flung off." This led to a variety of different outcomes since different dancers resolved conflicts differently. For example, in early runs with the round table, the first few dancers moved around the table at elevated speed, but then they tended to slow down and congregate in a slow moving flock nearby. The dancers who went around the table subsequently either stopped short to avoid cutting through this congregating group or broke the repulsion rule and charged right through it. In the performance events in December 2010, a "waterfall" effect was sustained with two slingshot tables (see Fig. 2.1).

The ability to prescribe individualized preferences and objectives was also explored. These explorations were motivated by an interest in understanding the role of heterogeneity in preference and objective in groups, if and how individuals can exert leadership through motion, and the range of emergent collective motion patterns that can result. Individualized objectives were typically prescribed secretly: all dancers were told to follow the three basic flocking rules and two or three of the dancers, unbeknownst to the others, were directed to follow certain additional rules. For example, the two or three selected dancers were sometimes given the same additional rule, such as to head for one corner of the room or out a door. Alternatively, the two or three dancers were given conflicting rules, such as one told to aim for one corner of the room and another to aim for the opposite corner. Or the two or three dancers were given a joint objective such as to split the group into subgroups. How dancers attempted to attain their additional objectives was explored, and it was observed and discussed how some dancers were successful and some were not successful in influencing the other dancers through their motion.

In some explorations, one or two dancers were instructed to behave as a predator or pursuer by waving a hand or t-shirt or flashing a bicycle light. In this case, the dancers were given the rule to keep a safe distance from the pursuers, e.g., 5 or 10 ft

Fig. 2.2 Snapshot of a cyclic pursuit experiment with 24 dancers. The position of each dancer is marked with a *green dot* and a *dashed line* connects each pair of pursuer and pursued. The collection of *dashed lines* show that the dancers move around a single *closed curve* that loops around, intersecting itself three times

depending on the size of the studio. The pursuers could thus put pressure on the group and create a variety of beautiful patterns by trapping the flock, shaping the flock, and restricting the flock's motion to changing corridors of space in the studio. Cyclic pursuit and evasion was also explored in which case every dancer was assigned one other dancer to pursue so that the group made a closed cycle with each dancer having one person to pursue and one person to evade. The motion patterns were constantly changing loops with multiple intersections, with qualitative features as predicted in [27]. Figure 2.2 shows a snapshot of 24 dancers in a cyclic pursuit experiment in the dance studio.

It was also found that adding an optional alignment rule created further artistic variation and options. Many other rules were explored in place of the basic flocking rules; these contributed significantly to artistic, engineering and scientific goals. For example , a rule was applied in which each dancer moved with oscillating speed, i.e., accelerating and decelerating repetitively, and such that the oscillations were out of phase with others nearby. This was motivated by the oscillating speed observed in fish schools and the rich family of motion patterns that could be designed using this rule [28]. In another example, rules for dynamic coverage were explored; these rules were motivated by problems of foraging over spaces of distributed resource [29].

2.2.2 FlockMaker

FlockMaker is a Java WebStart application developed to aid the Flock Logic project and designed for simulation and exploration of collective motion [30]. FlockMaker is inspired by the original work of Reynolds [31] on rule-based simulation of flocking particles. It is similar in spirit to the "Counterpoint Tool" in Synchronous Objects [18], in which the dynamic motion of widgets are animated and the user can modify the number and scale of widgets as well as the parameters that define the extent of their alignment and synchrony of motion.

FlockMaker is intuitive for a curious layperson, and it can be used to model complex combinations of flocking rules and initial configurations. Each dancer is represented as a single particle, modeled as a colored dot with a directional arrow, moving in the horizontal plane with variable velocity. Speed and facing angle (but not acceleration) are taken to be approximately continuous in time.

The user can assign a variety of flocking rules to the dancers, such as "Pursue Someone," "Repel Neighbors," and "Slow Down Near Neighbors." To further control behavior, the user can set values for a wide range of parameters pertaining to a dancer's rules or initial configuration, including radius of sensing, number of neighbors sensed, maximum speed of rotation, and magnitude of additive random noise. Different rules can be assigned to different dancers. Furthermore, each dancer can be assigned to follow multiple rules at a time, each rule potentially carrying a different relative weight representing its level of priority.

Dancers interact not only with one another, but also with the room in which they are moving, represented as a rectangular space contained within four walls. The FlockMaker user can change the size of the room, add obstacles to the room, and add rules applicable only within certain zones of the room.

After several weeks of work in the studio, the students in the Princeton University Atelier course spent time using FlockMaker, both to test ideas that had been tried in the studio and to investigate new ideas. Several of the rule sets and emergent behaviors investigated in FlockMaker were subsequently explored in the studio.

2.2.3 Experiments

A series of human flocking experiments was run in mid December 2010 in the $62'7'' \times 28'4''$ New South dance studio at Princeton University. Groups of dancers carried out the three basic rules of flocking with manipulations on initial conditions, number of dancers N (either 13 or 24), and number of neighbors m for cohesion (either 1 or 2). Alignment with neighbors was tested as was the assignment of an additional rule to two of the dancers (of which the others were not aware), which was to try to split the group. Several experiments were also run with dancers implementing the rules for cyclic pursuit; see Fig. 2.2.

Six Trendnet IP-600 cameras, synchronized over a local wired network, were set up in fixed locations to record the motion of the dancers. Two cameras were hung on the ceiling near either end of the studio, facing inward towards each other, and four were mounted high up on one side wall. Camera views covered the majority of the space in the studio and overlapped significantly. Using built-in software, the cameras recorded video and stored it on a laptop. The video provided 640×480 resolution and 20 frames per second.

For the December 2010 series of experiments, part of the room was blocked off so that the motion of all of the dancers could be fully captured by one of the six cameras (one of the two fixed to the ceiling). The dancers wore bright colored hats, black clothing and bare feet to aid trajectory tracking.

In this chapter, we examine one experiment from the series in which there were $N = 13$ dancers—two professional dancers and 11 students. All 13 dancers were asked only to follow the three basic rules of flocking with cohesion to $m = 2$ neighbors. The total time for the experiment was 185 s, corresponding to the period from the start to the stop of the music. We study the tracked trajectories of the dancers from the first 72 s of this experiment.

2.3 Trajectory Tracking

Trajectories were estimated using custom tracking software applied to the overhead video from one camera for the first 72 s of the experiment. The tracked trajectories comprise an ordered set of 1440 planar position vectors (x, y) for each of the 13 dancers. A velocity vector is computed for each dancer at every time step by differencing the position vectors. Speed and heading are computed as the magnitude and angle of the velocity vector. Figure 2.3 shows one frame from the video with superimposed tracked positions and directions of motion.

The custom tracking software uses a modified version of a real-time tracking algorithm that was developed at Princeton and used successfully for experiments involving multiple fish and robots [32]. The algorithm is implemented using the MADTraC C++ library [33], which in turn relies upon OpenCV [34] for low-level image processing routines. The original tracking software was designed to address the challenges of tracking potentially densely distributed objects that are very similar to one another in appearance. It was therefore applicable to the task of estimating dancers' trajectories.

The tracking algorithm follows three steps that are iterated for each video frame, and described in greater detail in [32]. In the first step, image segmentation produces a set of "blobs," such that each blob is a collection of contiguous pixels with high likelihood of belonging to any dancer's hat. Likelihood is determined by thresholding each pixel's value in HSV color space and mapping to a binary image. Blobs are extracted from the binary image using OpenCV's built-in blob labelling algorithm, which is based on [35]. A blob is often associated with more than one dancer because

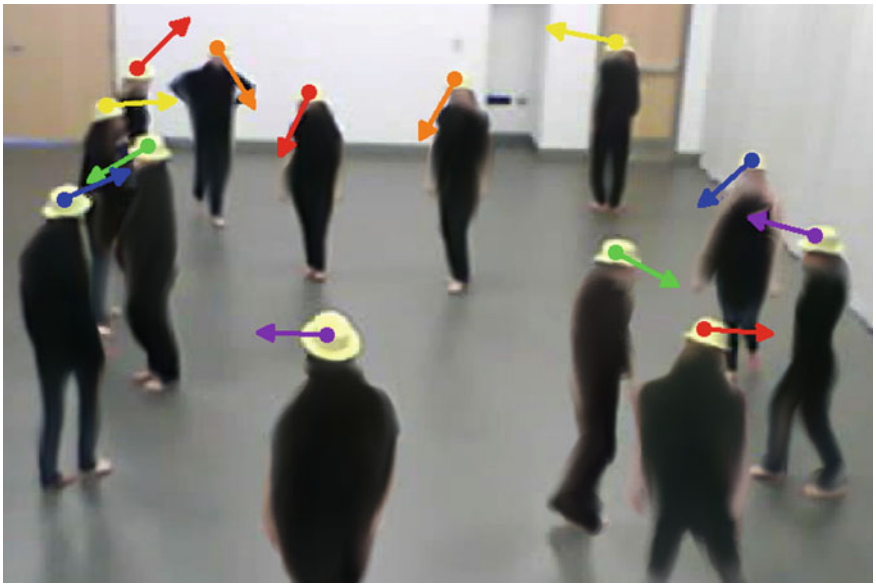


Fig. 2.3 From [26]. One frame from the video of the experiment with superimposed tracked positions (*colored dots*) and normalized velocity vectors in the image plane (each *colored arrow* indicates the direction in which the centroid of the corresponding hat is moving in the image). Images of dancers are deliberately blurred

of the physical proximity of dancers to one another, the proximity of dancers in the image due to the angle of the camera, and noise in the image.

In the second step, the blobs are analyzed in order to extract a noisy measurement for the position of each dancer. If a single dancer is associated with a blob, then the measurement of that dancer's position is taken as the centroid of all pixels in that blob. Otherwise, to resolve multidancer blobs or clusters of densely-spaced blobs, an expectation-maximization mixture-of-gaussian (EMMG) algorithm is used, which iteratively adjusts dancer positions for a given cluster and provides position measurements as output.

In the third step, the noisy position measurements are used with an unscented Kalman filter (UKF) for each dancer to provide a more accurate estimated position (x, y) in the current frame and to predict the position in the next frame. The estimated position of each dancer is stored as the current point in the dancer's tracked trajectory. The predicted positions are used to inform the next tracking iteration. The (x, y) position vector is expressed in a coordinate frame that is parallel to the floor. The transformation to these coordinates from image plane coordinates was determined by applying camera calibration techniques to an image of several objects placed at known locations in the scene. The average height of each dancer is assumed to be 1.65 m.

2.4 Graph Theory and Visualization

2.4.1 Background on Graphs

Let N be the number of dancers. For each dancer i, we define the set of neighbors, $\mathcal{N}_i$, to be the set of dancers whose positions are observed and used for cohesion by dancer i.

We associate to the system a *sensing graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E}, A)$, where $\mathcal{V} = \{1, 2, \ldots, N\}$ is the set of nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges and A is the $N \times N$ adjacency matrix with $a_{i,j} = 1$ when edge $(i, j) \in \mathcal{E}$ and $a_{i,j} = 0$ otherwise. Every node in the graph corresponds to a dancer, and the graph contains edge (i, j) when $j \in \mathcal{N}_i$. An edge $(i, j) \in \mathcal{E}$ is said to be *undirected* if (j, i) is also in $\mathcal{E}$; otherwise it is *directed*. A graph is undirected if every edge is undirected, that is, if A is symmetric; otherwise it is directed.

A graph can be represented visually by drawing a dot for each node and a line between the appropriate pair of nodes for each edge. An undirected edge is typically drawn as a simple line, while a directed edge (i, j) will have an arrow head pointing from node i to node j.

A *path* in $\mathcal{G}$ is a (finite) sequence of nodes containing no repetitions and such that each node is a neighbor of the previous one. The length of a path is given by the number of edges traversed by the path. The *distance*, $d_{i,j}$, between nodes i and j in a graph is the shortest length of any path from i to j. If no such path exists, $d_{i,j}$ is infinite. This distance is not a metric since $d_{i,j}$ is not necessarily equal to $d_{j,i}$.

The graph $\mathcal{G}$ is *connected* if it contains a globally reachable node k; i.e., there is a path in $\mathcal{G}$ from i to k for every node i. $\mathcal{G}$ is said to be *strongly connected* if there is a path between every pair of nodes in the graph. A strongly connected component of $\mathcal{G}$ is a maximal subset of nodes such that there is a path in $\mathcal{G}$ between every pair of nodes in the subset. $\mathcal{G}$ is *weakly connected* if it is connected when every directed edge is replaced by an undirected edge. A weakly connected component is a maximal subset of nodes that forms a connected component when every directed edge in $\mathcal{G}$ is replaced by an undirected edge.

The *status*, s_k, of a node k is the average inverse distance between every other node and k. That is, $s_k = \frac{1}{N-1} \sum_{j \neq k} \frac{1}{d_{j,k}}$. s_k will be maximum (equal to 1) if there is an edge from every other node to node k, and minimum (equal to 0) if there are no edges leading to node k.

2.4.2 Visualization of Graphs

FlockGrapher is a Matlab tool developed to compute, visualize and evaluate different kinds of graphs derived from flock position data. Using a graphical user interface, the tool accepts tracked position and direction of motion data for individuals in a flock in two or three dimensions. It can visualize data from one specific instant in time or

create a time series animation of data sets corresponding to successive time steps. The user can create graphs from the data by defining an individual's neighborhood in terms of either a prescribed number of nearest neighbors or a prescribed sensing radius. For data that includes the direction of motion of nodes, FlockGrapher can incorporate a limited viewing angle, assumed to be symmetric about the individual's direction of motion. In the case of a fixed number of nearest neighbors and a limited viewing angle, if there are fewer than the required number of neighbors visible to a node, the viewing angle will be rotated with respect to the direction of motion until enough neighbors are visible. Edge weights can be automatically manipulated, e.g., as a function of distance between nodes, or they can be prescribed by the user.

Once a sensing graph has been computed, FlockGrapher can evaluate a range of graph properties, including number of strongly and weakly connected components, algebraic connectivity, speed of convergence and node status. The tool also displays some properties on the graph visualization; for example, directed and undirected edges can be distinguished with different colors. For sets of data corresponding to successive time steps, the time-varying values of these properties will be displayed as the graph visualization changes. In the case of the human flocking experiment, this dynamic graph visualization can be run at the same time as the video of the dancers to compare computed and observed behavior. FlockGrapher can save all the computed data to allow for further analysis. A screenshot of FlockGrapher is shown in Fig. 2.4; the graph and its properties correspond to the frame from the video shown in Fig. 2.3.

2.5 Sensing Model and Graph Computation

Since each dancer was given the same specific rules to follow, it is in principle possible to apply the same rules to the tracked data and reconstruct the sensing graph used by the dancers. However, certain aspects of both the rules and human behavior make this task challenging. Although the dancers were each told to stay arm's length from two other dancers, no instruction was given for how they were to choose these two neighbors. In addition, although humans have a field of view of up to 200° [37], there was no compulsion for the dancers to keep both of their neighbors visible at all times.

Given these limitations, two key assumptions were made in order to estimate the dancers' sensing graph. First, it was assumed that each dancer only chose neighbors from within a limited angular range centered about the direction they were traveling. Since no dancer was observed to be rapidly moving their head, the direction of motion was assumed to be a reasonable proxy for direction of the head and therefore for center of viewing range. Although this assumption is generally applicable to the data, there were instances observed in which a dancer would either move in a different direction to where they were facing, or move with their head turned at a constant angle to their body. These occurrences are impossible to detect with our point-tracking approach, but could be accounted for with a more sophisticated tracker with the ability to detect

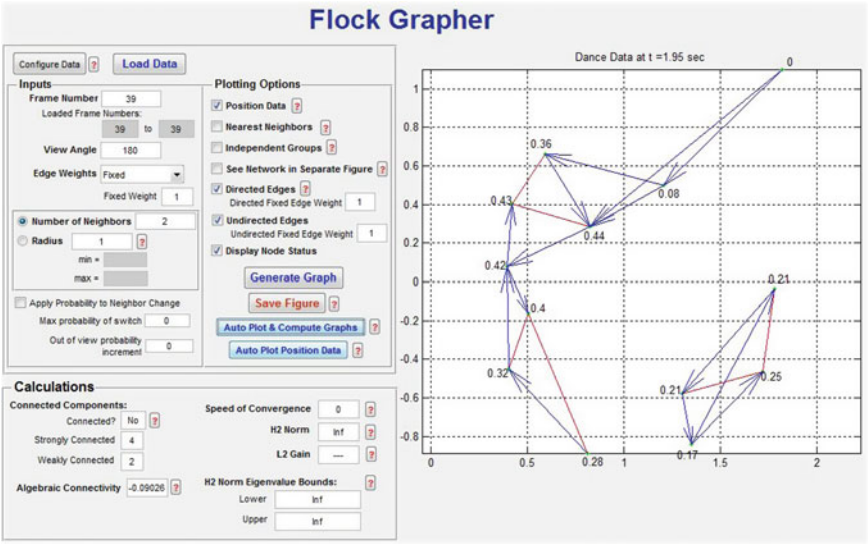


Fig. 2.4 Screenshot of FlockGrapher using dancer data corresponding to the instant shown in Fig. 2.3. Nodes are shown as *small green circles* connected by edges. Directed edges are *blue* with *arrow heads* and undirected edges are *red*. The number next to each node is its status. Other computed graph properties are displayed on the *left*

the orientation of each dancer's face. Second, it was assumed that each dancer was applying the cohesion rule with the two *nearest* neighbors within this range. Since every dancer was trying to keep two neighbors at arm's length (and let no dancers closer than arm's length), a dancer's neighbors would naturally be among the closest of the other dancers.

With these assumptions, we used FlockGrapher to estimate the sensing graph at each time (frame) by computing the two-nearest neighbor graph with a limited viewing angle. When fewer than two other dancers were visible using the direction of motion to center the viewing region, this region was allowed to rotate until two dancers became visible. However, it was not known *a priori* what viewing angle to choose to best represent the dancers' behavior.

For collective behavior, it is impossible to guarantee that a group will remain together if the communication graph is not connected [38]. When the graph is disconnected, there is nothing to prevent different subgroups from moving in different directions and splitting the group. However, other features of the environment (such as the limited space in the room) can drive the group back together. Since fissions and fusions of the group were observed, we selected the viewing angle for our sensing model as the one that produced a graph that was disconnected when the group of dancers was observed to split and remained connected when the group of dancers was observed to be cohesive.

Table 2.1 Effects of viewing angle on graph connectedness over the whole tracked period

Total viewing angle	Percentage of time connected (%)	Number of disconnection events
360°	59.58	40
270°	91.67	43
210°	97.5	10
180°	98.47	3
150°	98.68	5
120°	99.65	3
90°	99.58	3

Table 2.1 shows the results of estimating the sensing graph across the whole tracked period using different viewing angles. It can be seen that reducing the viewing angle from 360° to 270° significantly improves the amount of time the graph is connected, with the maximum connectedness occurring with a viewing angle of 120°. However, our goal was not simply to maximize connectedness but rather to match the observed behavior of the group.

Early in the experiment, between about 1 and 3 s, a small group of four dancers split from the rest of the group. The dancers within this group appeared to be observing only one another. Eight of the remaining dancers also formed a group, only observing one another. The thirteenth dancer was originally able to observe both groups before turning to face the larger group, but since no other dancer was observing this individual, the group was split during this whole period. Eventually, the dancers in the larger group turned and observed the smaller group, leading to a single "flock" again. This disconnection event was reflected in the estimated graphs for viewing angles of 150° and greater, but not for the smaller angles. However, with a viewing angle of 150° the graph became connected at a few points within this interval when direct observation of the video suggests that the group was still split. This was not the case with a viewing angle of 180°; thus, 180° was chosen as providing the best match of the splitting behavior of the dancers. Figure 2.3 shows the group during this disconnection event and the graph in Fig. 2.4 (corresponding to the frame of Fig. 2.3) is computed using a viewing angle of 180°.

Although our first estimate of the sensing graph captures a split in the group and stays connected during the rest of the tracked period, it remains a crude approximation to the true sensing graph. For example, some nodes change their neighbors rapidly in our estimated graph, which is likely an overestimation of the rate at which dancers switch neighbors. Instead, if an individual has just been chosen as a neighbor, that individual is likely to stay a neighbor for a period of time rather than being immediately discarded as another individual comes closer in view. Two steps were taken in an effort to reduce rapid neighbor fluctuations. First, the tracked position data were passed through a low-pass filter, which consequently smoothed out node headings. Then, to account for an individual's reluctance to change neighbors soon after they are chosen, a term was added to the estimation model representing the probability of switching from a current neighbor to a closer dancer. The lower was

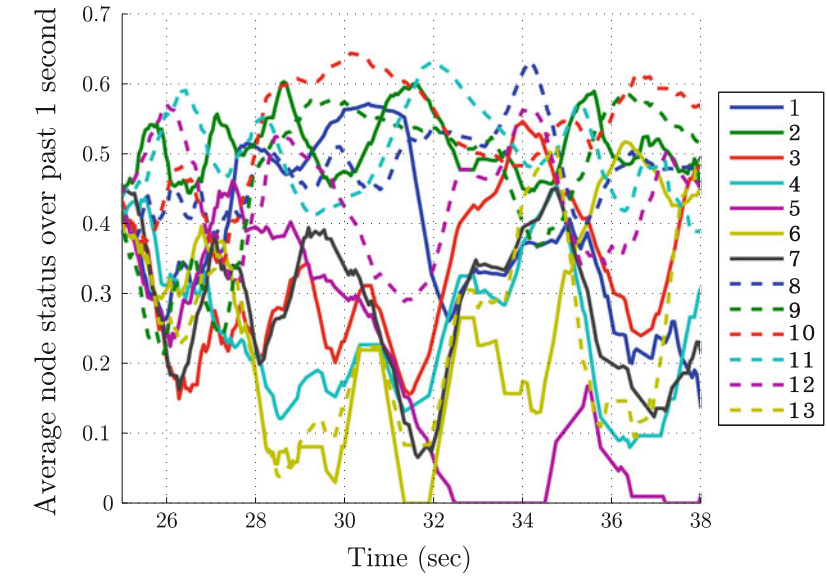
the probability the greater was the "inertia" of the dancer to switch to a closer dancer, equivalently, the greater was the commitment of the dancer to its current neighbor. By allowing this probability to reset to a low value whenever a new neighbor was chosen and then increase with time, we could capture the inertia of edges in the sensing graph.

2.6 Analysis of Individual Influence

We used the estimated time-varying sensing graph to begin investigating the influence of each individual dancer within the group. Our method computes and compares node status for all dancers. Without knowing precisely how each individual implemented the flocking rules, node status can provide an estimate of an individual's importance within the group. A dancer with a status of 0 has no influence since no one else in the group is observing that dancer. A dancer with a status of 1 has the maximum possible influence as every other individual is directly observing that dancer. However, due to the time-varying nature of the graph, an individual's importance depends not only on its current node status but also on its node status in the past. Therefore, we took as a first estimate of instantaneous importance each node's average status over the past 1 s. A plot of average node status for all nodes for part of the tracked period is shown in Fig. 2.5.

Although node status provides a measure of an individual's potential to influence the group, it does not indicate whether that influence was actually exercised. Therefore, to examine if node status is indeed related to the influence of a dancer in this data set, we investigated a quantitative measure of an individual's influence on the rest of the group. This quantity is the time, referred to as *lead time*, at which a peak occurred in the cross-correlation function between an individual's direction of motion and the direction of the group's motion. Positive values for this lead time indicate that an individual tended to *lead* the group (i.e., change direction and then have the group follow) while negative values indicate that an individual tended to *lag* the group (i.e., change direction to follow the group after the group had changed). This lead time measure was found to correlate strongly with average node status, with the nodes with highest average status having the largest lead times and the nodes with lowest average status having the largest lag times. Thus, we argue that node status computations do indeed provide insight into leadership roles within the group. Importantly, the ability to calculate node status at any point in time allows for investigation of instantaneous and changing leadership throughout the flocking event.

By examining the plot of averaged node status, "leadership events" can be identified, where one particular node achieved the greatest importance within the group (with a high status value) for an extended period of time. Figure 2.5 reveals one such event when node 10 became a leader between approximately 28.75 and 31.45 s. Looking at the video, it can be observed that during this time the group was moving from the back left corner of the room toward the front right corner, with node 10 at



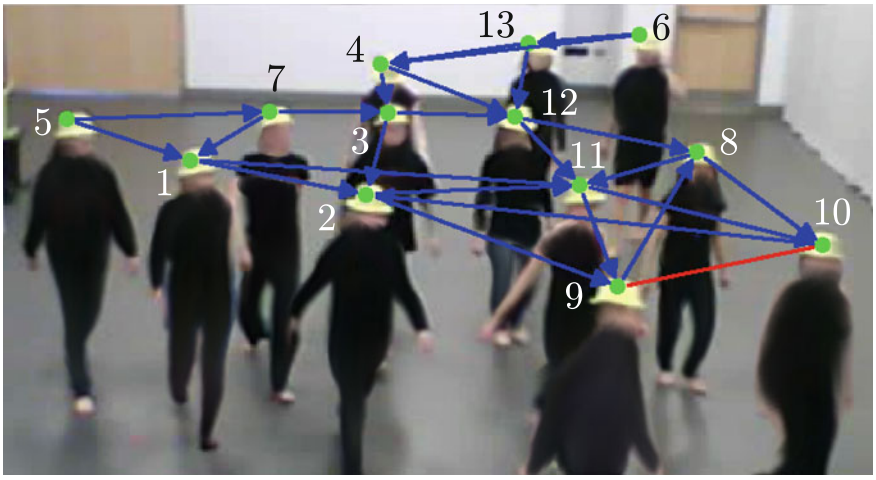


Fig. 2.5 From [26]. Plot of 1 s running average of node status for all nodes, along with a sample video frame and sensing graph near the end of the leadership event from $t = 28.75$ s to $t = 31.45$ s. The *red edge* is undirected while all *blue edges* are directed. Node 10, with the highest status, corresponds to the dancer in the front of the group

the front of the group. This suggests that the node status measurements can capture emergent leadership.

Another leadership event can be observed from the data during a period when one dancer stopped moving and the remaining dancers started circling around this

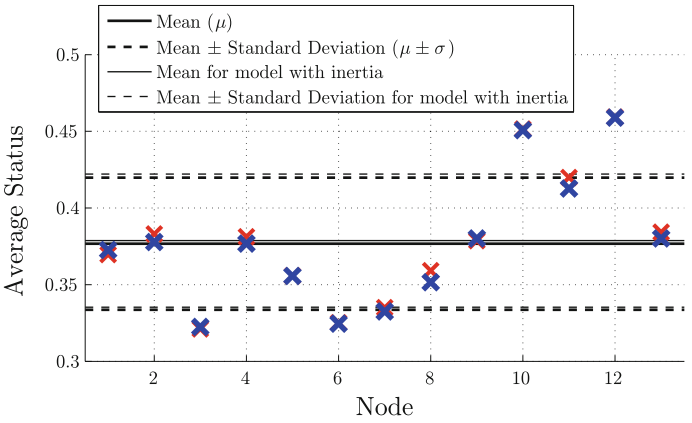


Fig. 2.6 Average node status over the whole tracked period. The *blue crosses* represent the status values without any inertia term, while the *red crosses* represent the status values with an inertia term where the probability resets to 0 when new neighbors are chosen and grow back to 1 with an increment of 0.14 per frame

individual. However, the individual with the highest status during this event was not the stationary one, but rather one who was very close by the stationary one and who kept moving in a circle. This seems particularly interesting since at other times (however, not during our tracked period) one dancer would stop and the whole group would eventually stop too. The difference between these two kinds of events (circling vs. stationary group motion) may be due to the differences between the status of the stationary and nearby dancers in the first case as compared to the second case.

By averaging each individual's status over the whole tracked period, we evaluated whether or not some individuals had a disproportionate influence on the group. Figure 2.6 shows the average of each node's status over the tracked period. Nodes 12 and 10 had the highest average status, with values 1.9σ and 1.7σ higher than the group mean, where σ is the standard deviation of the average status values over all nodes.

The average of an individual's status over the whole tracked period can be similarly computed in the case that the graph estimation model includes a probability-based reluctance to switch neighbors, as described above to model switching inertia or, equivalently, commitment to neighbors. Interestingly, while this inertia term does lead to a significant decrease in average neighbor changes per second, the overall structure of average node status values does not change significantly. Figure 2.6 displays the average status of each node both for the original model (blue) and the filtered model with the inertia term (red). Although there is some variation between the average node status values for the two models, the same nodes represent upper and lower outliers. This suggests that the incorporation of reluctance to switch neighbors into the model does not alter the overall sensing structure, notably the emergence of leaders, even as it potentially smooths out unrealistic fluctuations of node neighbors.

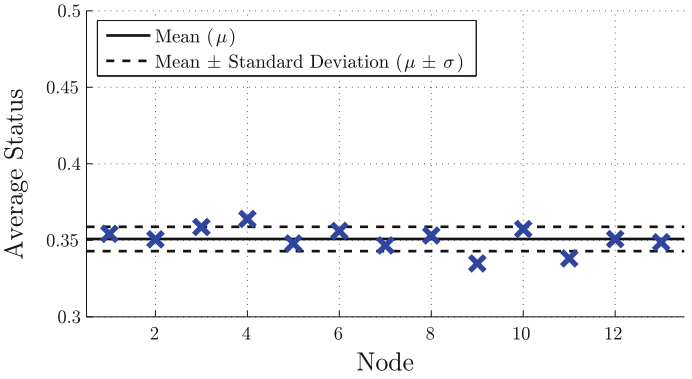


Fig. 2.7 Average node status, with no inertia, from a simulation over an equivalent period to the tracked dancer data

We hypothesize that the emergence of the outlier nodes in Fig. 2.6, and in particular those with very high status, is due to human bias. To test this, we developed an agent-based flocking simulation which lacks any human bias. Given our hypothesis, we would expect that the simulated agents, without human bias, would not exhibit outlier nodes. The simulation models particles that move in the plane, in a space with boundaries like in the dance studio, and follow the rules and parameter values close to those given to the dancers. The simulation was run in Matlab with particle positions updated synchronously to move in the direction to maintain one arm's length (assumed to be 0.80 m) from its two closest nodes within a viewing angle of 180°, while also repelling from all other nodes within an arm's length. Additional functions were incorporated to provide limits on velocities, turning rates and response to boundaries. The corresponding node status of each of these particles was calculated analogously to those of the dancers.

The simulated particle system was initialized with positions and headings matching those of the dancers in our experiment and then the average status of each node was calculated over the following 72 s. The average node status over the first 72 s is shown in Fig. 2.7 and can be compared to the plot in Fig. 2.6. It can be seen in Fig. 2.7 that both the mean and the standard deviation are smaller than in the case of the human dancers and furthermore there are no significant outliers. Every node in the simulation has a status in the range of values below the human dancer that ranked eighth in terms of highest node status and above the human dancer that ranked tenth. Additional simulations were also run with random initial conditions. The average mean status and average standard deviation of 40 simulations run for 72 s each with random initial conditions was found to be approximately 0.352 and 0.0094, respectively. These mean and standard deviation values are very similar to those from the simulation of Fig. 2.7 with the dancers' initial conditions (0.351 and 0.0080, respectively) and are significantly smaller than what is computed from the tracked dancer data (0.377 and 0.043, respectively).

This comparison suggests that one consequence of human bias in the behavior of the dancers is that some individuals were less influential and other individuals significantly more influential as compared to a group of identical particles. This implies that rather than leadership simply arising as a result of random mixing within the group, the behavior of some individuals makes them more likely to assume positions of high influence. We note that the dancers corresponding to nodes 10, 11, and 12 (the three nodes with highest average status) are three of the four dancers in the small disconnected group of Figs. 2.3 and 2.4, suggesting further possible consequences of emergent leaders.

2.7 Final Remarks

The Flock Logic project, conceived at the intersection of dance and control theory, produced a novel and generative framework for artistic, engineering, and scientific investigation of collective motion. The project centered around explorations and experiments with the motion patterns that emerge when dancers apply feedback rules modeled after those attributed to flocking birds or schooling fish. The framework combined the prescribed rules of individual behavior and response with the unknown choices and actions of living agents, yielding opportunities for exploration that was part systematic and part uncontrolled. As a result, the Flock Logic framework proved useful for artistic exploration of dance and tools for choreography, for engineering exploration of decentralized control laws for multiagent system dynamics, as well as for scientific investigation of collective animal behavior and crowd dynamics.

The explorations built off of a set of basic "flocking" rules of cohesion and repulsion: dancers were instructed to move around while maintaining an arm's length distance from a prescribed number of other dancers and not letting anyone come closer than arm's length. Rules for walls, obstacles, and zones were added. Additional objectives and preferences were imposed selectively and secretly so that a small subset of dancers were asked to influence the group only through their motion and without explicit signaling. Dancers behaving as pursuers or predators applied pressure dynamically on the group, often to beautiful effect. Synchrony and anti-synchrony of directionality were explored using alignment rules. Many artistic explorations made use of different kinds of rules that were originally motivated from observations or analysis of animal behavior or from engineering design objectives, such as foraging and coverage. Other explorations were motivated purely by artistic goals, such as designed responses to specific obstacles.

To address questions concerning the role of the heterogeneity of the group of dancers and specifically the relative influence of the different dancers on the collective motion, we analyzed video data of an experiment with thirteen dancers applying the basic rules of flocking with two-person cohesion. From the video we tracked, the trajectories of the dancers over a 72 s segment. Then, we applied the flocking rules to the data to estimate the network graph at each frame of the video, that is, who was paying attention to whom at each time step. From the resulting time-varying

graph, we computed node status for each dancer at each frame; node status provides a measure of how much attention a dancer received from the rest of the dancers. We discussed how node status was strongly correlated with lead time in turning, i.e., dancers with high status would typically turn before the rest of the group. From this we argued that high node status suggests high influence and therefore leadership. By examining the average status of each dancer over the whole tracked segment, we found two of the dancers with status higher than the mean value by nearly twice the standard deviation. We showed how this result is robust to the addition of an inertia term that models a dancer's commitment to its newly acquired neighbors. We also showed evidence that human bias explains the large variation in influence among the dancers, and in particular the outliers, by comparing the data with analogous results from a simulation of dancers modeled as particles without human bias.

These results raise many more interesting questions and possibilities for future investigation. For example, how does human bias produce leadership, where no such leadership was assigned? Do certain dancers move in ways that attract the attention of the others? Or do individuals who emerge as leaders break the rules, for example, by paying less attention to others than instructed? The results suggest the possibility of an interesting tension between following rules and breaking rules. This could be explored scientifically using evolutionary game theory in which there is a benefit to breaking the rules associated with influencing the group toward one's own preferences but also a cost to breaking the rules associated with losing the advantages of group living.

Other questions concern the relationship between the rules and environment and the resulting shape and momentum of the group. What accounts for polarized versus circular motion? What accounts for fissions and fusions of the group? What role do leaders play in these dynamic transitions? These questions address the fundamental interplay between how an individual influences the group and how the group influences an individual. In [39] leadership in a dynamic network evolves according to distributed adaptive dynamics driven by a metric that rewards efficient tracking of an external signal; it can be shown in this context that optimal leader sets are defined by their joint centrality, a property of the network graph much like node status [40]. Our present results suggest an important link between the spatial distribution of more influential individuals, i.e., those with high node status, and the group-level dynamics. These results may inform design of distributed multiagent adaptive dynamics of rules, rule-breaking or network interconnections, to control leadership and thus the collective behavior of the multiagent system.

Many further artistic, engineering, and scientific investigations are possible, even extending the basic flocking rules into more abstract, nonspatial, domains. Such domains could include music (as shown in Chap. 7 [22]), personal preferences, styles, language, and more. Human flocking for recreation or therapy might also be explored—participants in the Flock Logic performances described finding it calming to engage with a group without a goal and rewarding to be part of creating something new.

Acknowledgments The authors thank Alex Holness for his contributions to the study of the lead/lag time and its correlation with node status.

References

1. Leonard NE, Marshall S (2010) Flock Logic [Online]. Available: http://www.princeton.edu/~flocklogic
2. Parrish JK, Edelstein-Keshet L (1999) Complexity, pattern, and evolutionary trade-offs in animal aggregation. Science 284(5411):99–101
3. Breder JCM (1954) Equations descriptive of fish schools and other animal aggregations. Ecology 35(3):361–370
4. Couzin ID, Krause J (2003) Self-organization and collective behavior in vertebrates. Adv Study Behav 32:1–75
5. Sumpter DJT (2010) Collective animal behavior. Princeton University Press, Princeton
6. Ballerini M, Cabibbo N, Candelier R, Cavagna A, Cisbani E, Giardina I, Lecomte V, Orlandi A, Parisi G, Procaccini A et al (2008) Interaction ruling animal collective behavior depends on topological rather than metric distance: evidence from a field study. Proc Nat Acad Sci 105(4):1232–1237
7. Young GF, Scardovi L, Cavagna A, Giardina I, Leonard NE (2013) Starling Flock networks manage uncertainty in consensus at low cost. PLoS Comput Biol 9(1):1–7
8. Hildenbrandt H, Carere C, Hemelrijk CK (2010) Self-organized aerial displays of thousands of starlings: a model. Behav Ecol 21(6):1349–1359
9. Gueron S, Levin SA, Rubenstein DI (1996) The dynamics of herds: from individuals to aggregations. J Theor Biol 182:85–98
10. Fischhoff IR, Sundareson SR, Cordingley J, Rubenstein DI (2007) Habitat use and movements of plains zebra (equus burchelli) in response to predation danger from lions. Behav Ecol 18(4):725–729
11. Leonard NE (2013) Multi-agent system dynamics: bifurcation and behavior of animal groups. In: Proceedings of 9th IFAC symposium on nonlinear control systems, Elsevier, pp 1–11
12. Dyer JRG, Johansson A, Helbing D, Couzin ID, Krause J (2009) Leadership, consensus decision making and collective behaviour in humans. Phil Trans R Soc B 364:781–789
13. Moussaid M, Perozo N, Garnier S, Helbing D, Theraulaz G (2010) The walking behaviour of pedestrian social groups and its impact on crowd dynamics. PLoS ONE 5(4):1–7
14. Funes P, Orme B, Bonabeau E (2004) Shaping collective behavior: an exploratory design approach. In: Artificial life IX: Proceedings of Ninth International Conference on the simulation and synthesis of artificial life, MIT Press, pp 232–237
15. Silverberg JL, Bierbaum M, Sethna JP, Cohen I (2013) Collective motion of humans in mosh and circle pits at heavy metal concerts. Phys Rev Lett 100(228701):1–5
16. Altenberg L (2000) Emergent movement: group experiments in emergent properties from simple rules. In: Short courses on the mathematics of biological complexity, University of Tennessee [Online]. Available: http://dynamics.org/Altenberg/TALKS/UTK_2000/EmergentMovement.html
17. Clemente K (1990) Playing with performance: the element of the game in experimental dance and theater. J Popular Cult 24:1–10
18. Forsythe W, Palazzi M, Shaw NZ (2009) Synchronous objects [Online]. Available: http://synchronousobjects.osu.edu/
19. Sgorbati S (2005) Emergent improvisation [Online]. Available: http://emergentimprovisation.org/essay.html
20. Hagendoorn I (2008) Emergent patterns in dance improvisation and choreography. Unifying themes in complex systems IV. Springer, Berlin, pp 183–195

21. Carlson K (2011) Exploring creative decision-making in choreographic practice: a phenomenological study of situated cognition. MSc Thesis, Simon Fraser University
22. Huepe C, Colasso M, Cádiz RF (2014) Generating music from Flocking dynamics. Controls and Art, Chapter 7. Springer, Heidelburg
23. LaViers A, Teague L, Egerstedt M (2014) Syle-based robotic motion in contemporary dance performance. Controls and Art, Chapter 9. Springer, Heidelburg
24. Ray Jaideep, Pinar Ali, Seshadhri C (2012) Are we there yet? When to stop a Markov chain while generating random graphs. In: Bonato Anthony, Janssen Jeannette (eds) WAW 2012, vol 7323., LNCSSpringer, Heidelberg, pp 153–164
25. Tsiotras P, Castro L I R (2014) Generation of intricate geometric patterns on plane surface using consensus protocols. Controls and Art, Chapter 6. Springer, Heidelburg
26. Leonard NE, Young G, Hochgraf K., Swain D, Trippe A, Chen W, Marshall S (2012) In the dance studio: analysis of human Flocking. In: Proceedings of American control conference on IEEE, pp 4333–4338
27. Pais D, Leonard NE (2010) Pursuit and evasion: evolutionary dynamics and collective motion. In: Proceedings of AIAA guidance, navigation, and control conference on AIAA, pp 1–14
28. Swain DT, Leonard NE (2009) On the trajectories and coordination of steered particles with time-periodic speed profiles. In: Proceedings of American control conference on IEEE, pp. 1286–1291
29. Caicedo-Nunez CH, Leonard NE (2011) Symmetric coverage of dynamic mapping error for mobile sensor networks. In: Proceedings of American control conference on IEEE, pp 4661–4666
30. Chen W, Leonard NE, Marshall S (2010) FlockMaker: simulator for collective motion [Online]. Available: http://dcsl.princeton.edu/dance
31. Reynolds CW (1987) Flocks, herds, and schools: a distributed behavioral model. In: SIGGRAPH '87 Proceedings of 14th annual conference on computer graphics and interactive techniques on ACM, pp 25–34
32. Swain DT, Couzin ID, Leonard NE (2012) Real-time feedback-controlled robotic fish for behavioral experiments with fish schools. Proc IEEE 100(1):150–163
33. Swain DT (2011) MADTraC framework source documentation [Online]. Available: https://github.com/leonard-lab/MADTraC
34. Bradski G (2011) OpenCV (Open Computer Vision) Library [Online]. Available: http://opencv.willowgarage.com/wiki/Welcome
35. Chang F, Chen C-J, Lu C-J (2004) A linear-time component-labeling algorithm using contour tracing technique. Comput Vis Image Underst 93(2):206–220
36. Mohar B (1991) The Laplacian spectrum of graphs. Graph Theory Combinatorics Appl 2:871–898
37. Werner ER (1991) Manual of visual fields. Churchill Livingstone, New York
38. Ren W, Beard R, Atkins E (2005) A survey of consensus problems in multi-agent coordination. In: Proceedings of American control conference on IEEE, pp 1859–1864
39. Pais D, Leonard NE (2013) Adaptive network dynamics and evolution of leadership in collective migration. Physica D [Online]. Available: http://dx.doi.org/10.1016/j.physd.2013.04.014
40. Fitch K, Leonard NE (2013) Information centrality and optimal leader selection in noisy networks. In: Proceedings of IEEE conference on decision and control, IEEE

Chapter 3
Dancing Robots: The Control Theory of Communication Through Movement

John Baillieul and Kayhan Özcimder

3.1 Dance and Motion Primitives

This chapter reports recent extensions of our work [1] on *motion-based communication* mediated by the dynamics of a control system. The discussion was focused on a form of dance involving a small set of motion primitives—beginners salsa. While the setting of this simple form of the dance has provided a useful setting for the discussion of key ideas, it is also of interest to consider the enhanced artistry to be found in more advanced level dance sequences. Hence, in this chapter, the basic motion primitives of *beginner's level salsa* (BLS) will serve as the foundation for introducing additional primitives that correspond to *intermediate level salsa* (ILS). As described in [1] and as will be recalled below, there are four motion primitives (dance steps) associated with beginner's salsa constitute the alphabet from which dance routines are constructed. Any sequence of steps from this four letter alphabet is admissible. This is to say that any of the four motion primitives can follow any other in the dance sequence. In constructing intermediate dance sequences, the alphabet of motion primitives will be larger, and it will no longer be the case that any move can follow any other move. This is related to another key difference between BLS and ILS. Motions in the latter involve upper body movements. The four basic steps A, B, C, D in beginner's salsa can be distinguished entirely in terms of foot movements of the dancers. In intermediate salsa, arm positions of the leader (male dance partner) and follower (female

This chapter reports results that have grown out of our paper in the 2012 American Control Conference [2]. Parts of that paper are reproduced here for the sake of completeness.
The authors gratefully acknowledge support for this work provided by the U.S. Office of Scientific Research through ODDR&E MURI07 Program Grant Number FA9550-07-1-0528, and also to the Office of Naval Research, ODDR&E MURI10 Program Grant Number N00014-10-1-0952.

J. Baillieul (✉) · K. Özcimder
Boston University, Boston, MA, USA
e-mail: johnb@bu.edu

A. LaViers and M. Egerstedt (eds.), *Controls and Art*,
DOI: 10.1007/978-3-319-03904-6_3,

dancer) play a key role in defining the step as well as in communicating the intended next step in the sequence. In what follows, we shall discuss the step transitions and characteristics of the interlocked arm configurations of the dancers in the language of topological knot theory. This language is useful in relating physical constraints on the dancers' body movements to the allowable step transitions in the dance.

The research described below is aimed at understanding how we might go about characterizing artistic communication through controlled motions. In computer science and linguistics, the concept of *expressiveness* characterizes the difficulty of recognizing a sentence in a language or grammar [5]. This chapter considers the problem of understanding expressiveness as it relates to perceived artistry in the movement sequences that enable communication in dance. When the objective of robot control and motion planning involves gesture-based information propagation, expressiveness can be thought of as a proxy for *controllability*. In [2], it was shown that a motion description language consisting of only two motion primitives is sufficiently expressive that it can be used to prescribe motions to accomplish robotic reconnaissance missions within an impotent class in two-dimensional domains. For the robotic dance described application below, the way in which expressivity depends on both the size of the motion primitive alphabet and the syntactic rules governing the construction of sequences of motion primitives will be explored.

The work that we report contributes to a growing literature on robotic emulation of human activities. A complete survey of this literature is not possible, but several efforts are notable. The use of advanced computer vision to detect and recognize movements is reported in Chap. 8. Leaving aside the important question of autonomous motion recognition, the present chapter focuses on generative models and their use in creating artistically pleasing motion sequences. The chapter that follows, together with the work reported in [1], describes the results of out effort to understand the creation of motion sequences that interpret musical phrases in salsa. An interesting counterpoint to this work is reported in Chap. 7 where the authors discuss blending audio segments to create musical phrases that interpret the detected flocking behaviors of a group of dancers. The perceived artistic characteristics of these musical phrases are shown to depend on both positions of the dancers and the communication between neighbors in the flock. In the work reported below, the artistic goal is always to create motion executions that utilize perceived movements of a dancer's partner to create a desired collaborative performance. While we focus on pairs of dancers, larger groups can certainly be studied. (C.f. [9, 10]). Work of LaViers et al. (Chap. 9) finds that additional stylistic characteristics such as the perceived energy expended in the dance routine can influence an observer's perception of artistry. While the work in this chapter (and the book as a whole) has aimed to articulate elements of a theory of robotic dance, the idea of coordinating group motions by means of regular signal patterns has a much wider range of potential applications. We call the reader's attention to Chap. 4 which treats the use of music to coordinate the movements of UAV formations.

The chapter is organized as follows. In the next section, we discuss some elementary ideas and concepts in topological knot theory and provide a brief review of the history of attempts to use formal languages to describe dance. Sections 3.3 and 3.4

will review earlier work on the relationship between motion complexity and artistic merit in beginner's salsa. Section 3.5 will discuss dance primitives and poses in terms of knot theory, and Sect. 3.6 will revisit figures of merit that have been introduced in Sects. 3.3 and 3.4. It will be shown that even though ILS is based on a larger alphabet of motion primitives, topologically enforced syntactic constraints cause the movements to be less random. Dance phrases emerge in a natural way. Section 3.7 provides some concluding remarks on quantifying artistic merit. We note that this chapter is an outgrowth of work reported in an earlier publication [1], and some of this earlier works are repeated for the sake of completeness in Sects. 3.3 and 3.4.

3.2 The Rudiments of Knot Theory

We recall that *knots* are simple closed curves in $\mathbb{R}^3$, and *links* are finite sets of knots that may be entangled with one another. Although interest in knots dates to antiquity, the formal study of knots as mathematical objects may be traced to Vandermonde's 1771 paper " Remarques sur les problèmes de situation", [12]. The modern introduction of polynomial invariants and other algebraic tools have significantly deepened the theoretical foundations, while the simultaneous proliferation of applications to statistical mechanics, molecular biology, and chemistry has secured the place of knot theory as an important mathematical discipline. Knot theory is of interest in the enhanced version of salsa that we shall examine below where the dancers' joined hands both enable artistic expression and constrain the grammar of motion sequences in the dance. For the purpose of our discussion, we shall provide a rudimentary conceptual introduction to the language of knot theory.

For any two knots, the linking number specifies how many times each curve (knot) winds around the other. The *linking number* is always an integer, and since the curves are oriented, the linking number may be positive or negative. A simple way to determine the linking number is to "project" the curves onto the plane by an *immersion* $f : \mathbb{R}^3 \to \mathbb{R}^2$. Under this mapping a knot (or link) is in one-to-one correspondence with its image except at double points (called *crossings*) where a distinction needs to be made between the top and bottom segments of the knot (link). (Note, that we do not allow image points of multiplicity higher than two [no triple points, for instance], and all crossings are assumed to be transverse.) To keep track of which segment crosses over and which crosses under, we represent the segment that is under by means of a break, as illustrated in Fig. 3.1. The planar image (with respect to f) of the knot (or link) together with the labeling of "over" and "under" segments at crossings constitutes the *link diagram*.

Two knots (or links) are *equivalent* if there exists an orientation preserving homeomorphis of a neighborhood of the first in $\mathbb{R}^3$ onto a neighborhood of the second (also in $\mathbb{R}^3$) such that the second knot (link) is the image of the first.

Well-known results of *K*. Reidemeister [13] have shown that equivalent links may be transformed into one another by a finite sequence of three elementary moves—the so-called *Reidemeister moves*. These are depicted in Fig. 3.2. A type 1 move simply

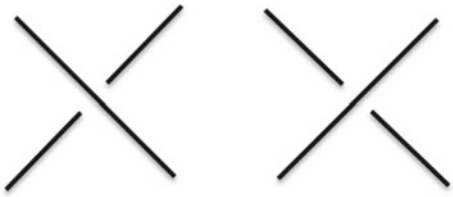


Fig. 3.1 Over and under pass information in 2-D regular diagrams

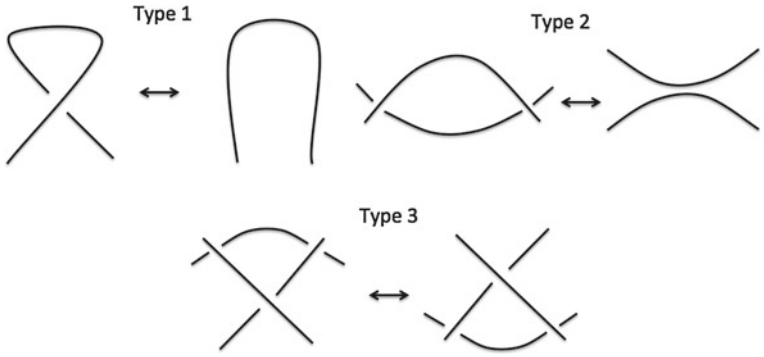


Fig. 3.2 Three Reidemeister moves preserving the link equivalence

removes or adds a kink, a type 2 move is a separation and a type 3 move is preserving the number of crossings.

Consider two links—colored, say, black and red. Assuming that each is oriented, there are precisely four possible crossings, as depicted in Fig. 3.3. Let the numbers of each type of crossing be n_1, n_2, n_3, n_4, respectively. The *linking number* is then defined to be

$$\#(L_1, L_2) = \frac{n_1 + n_2 - n_3 - n_4}{2}.$$

If L is a link with n components $L_1, L_2, \ldots L_n$ then the linking number of L is [11],

$$\sum_{1 \le i \le j \le n} \#(L_i, L_j) = \#(L). \tag{3.1}$$

Figure 3.4 gives the linking numbers for some simple links. Any two unlinked knots have linking number zero (Fig. 3.4). However, linking number zero for two knots does not necessarily mean that the knots are unlinked. One well-known example is Whitehead link which has linking number zero but for which the two components are linked (Fig. 3.5). This example is important because it shows that linking numbers do not determine the topology of knots and links. The related concept of an *unknot* will be important in what follows.

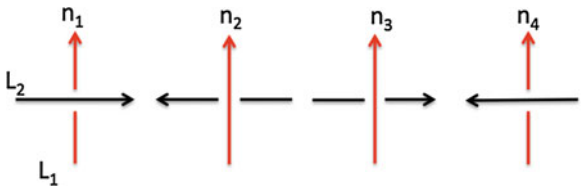


Fig. 3.3 Four possible crossings between the link edges

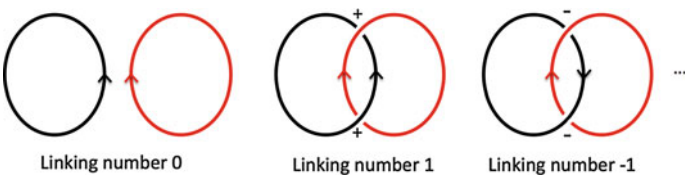


Fig. 3.4 Linking numbers for some of the 2-component links

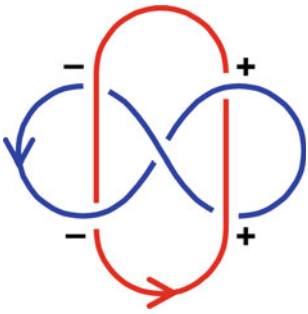


Fig. 3.5 Whitehead link with $\#(L_1, L_2) = 0$

Definition 3.1. *A knot is called an* unknot *(trivial knot) if it bounds an embedded disc. (That is to say, it is equivalent to a circle.)*

An important operation in knot theory is that of cutting and splicing. This operation can be used to alter the local over and under passing at crossings, and thus, it can be used to change the topology of a knot or link. The final basic concept of elementary knot theory that we shall make use of is the *knot sum*. Two oriented knots/links L_1 and L_2 can be added together to form their sum $L_1 \# L_2$ by placing them side by side and cutting each one once and splicing in two line segments such that the orientation is preserved [11] (Fig. 3.6). This is depicted in Fig. 3.7. Below (in Sect. 3.5), we shall use this circle of ideas to study poses and motion sequences in a simple extension of beginner's salsa that will be introduced next.

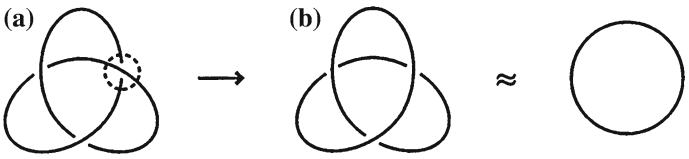


Fig. 3.6 *Unknotting the trefoil*: The cutting and splicing operation locally changes the over and under passes at a crossing

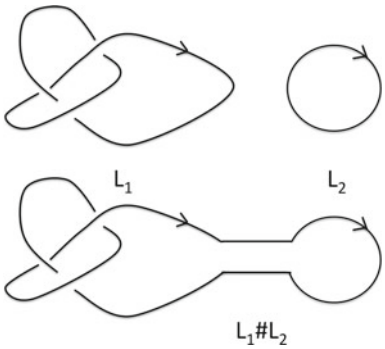


Fig. 3.7 Orientation preserving knot sum operation for the links L_1 and L_2

3.3 Salsa: Energy, Complexity, and Perceived Artistic Merit

There are many domains of human endeavor in which it is natural to decompose complex activities in sequences of motion primitives. Throughout many cultures, dance consists of sequences of body movements that are known to expert dancers, and passed on through formal instruction to beginners and students. The artistic content of formalized movements that occur in dance is central to what must be expressed in the motion-based language associated with each dance vernacular. It would seem natural, then, to develop a formal means of transcribing basic motion primitives for dance, but attempts to do this have not led to widespread use among dance professionals. Perhaps the best known effort in this direction was the development in the 1920s of *labanotation* [6, 15]. Rolf Von Laban attempted to develop a scripting language that was sufficiently expressive that all human movement could be described and recorded on paper. This has never been widely used, probably because in its attempt to be universal, it became complex and nonintuitive. (This is supported by noting the "more than 700 symbols that indicate parts of the body, direction, levels, and types of movement and the durations of each action." (Quoted from the web page [7].)

In this chapter, we avoid dealing with such expressive complexity by restricting our attention to a form of dance involving only a small set of motion primitives—beginner's salsa. As is always the case in the performing arts, there are distinct levels of proficiency in salsa. Because our goal is to analyze and deconstruct component motions in order to reinterpret them as controlled motions of simple wheeled robots,

Table 3.1 The ten dance sequences

Dance 1	*DDADBBBBACCCDDDDDBDAAAA*
Dance 2	*AAAAAAAADDDDDDDDDDDBDBB*
Dance 3	*ADBCDACBDADBCDABACDACBD*
Dance 4	*DBCADBCADBCADBCADBCADBC*
Dance 5	*ACBDACBDACBDACBDACBDACB*
Dance 6	*ABCDBCDACDABDABCABADBCD*
Dance 7	*DBADACBDDBABDDAACDBBDAD*
Dance 8	*AAAABAAADAAAAAAACAAADAA*
Dance 9	*DBCDCBBDCBDDDBDDDAABCCC*
Dance 10	*DBDCCBDDBBDDDCCCCABDDDB*

we consider a version of beginner's salsa that uses only four basic steps which we label A, B, C, and D. (See Fig. 3.17) In salsa, as in many forms of dance, each motion primitive (dance step) begins and ends in accordance with the rhythm of the music to which it is set. Specifically, in a sequence of elementary steps making up a salsa performance, each of the four motion primitives is executed for a period of eight beats (two bars) of the music. In what follows, we describe the salsa motion primitives in terms of their two-bar durations. We assume each primitive has the dance partners standing in a standard initial pose as illustrated in Fig. 3.17.

In an attempt to understand something about how people perceive the artistic merit of a dance performance, two dancers were asked to perform a number of short salsa segments using the four basic dance primitives in different sequences. Digital video recordings of the salsa segments were shown to 20 "judges" who were asked to rank the performances in order of artistic merit. The judges included both trained dancers as well as people with no formal training in dance. All judges were instructed to use standard criteria in their rankings, including artistic conent, dance routine difficulty, partner synchronization, and complexity of the choreography. Ten dance sequences, each comprised of 23 basic dance primitives were selected to be ranked by each of the judges. Using the motion primitives (dance steps) A, B, C, D described in the previous section, the ten performances are given in Table 3.1.

The average scores of the 20 judges are given in the first row of Table 3.3. Dance sequence 9 was preferred, while almost no one liked dance number 2. It is noted that although the judges were in substantial agreement regarding dance number 2 (rated as poor) there was comparatively high variance in the judges scores on other dances.

Having thus tabulated the judges' rankings, we were led to the question of whether the artistic qualities in terms of which the performances were differentiated could be indentified in a precise and even quantitative way. The late Dennis Dutton identified *complexity* as one of the four central characteristics of great art.[1] Dutton [4] To evaluate the complexity of a sequence of symbols such as those in Table 3.1, we considered metrics suggested by the well-known Shannon Entropy. The simplest possible metric may be arrived at by recording the number of occurrences of each

[1] Dutton identifies the four central characteristics of high art as 1. complexity, 2. serious content, 3. purpose, and 4. distance [4].

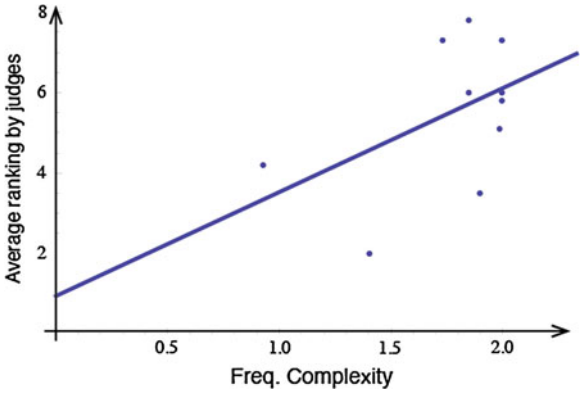


Fig. 3.8 A scatter plot of Judges' rankings as a function of the frequency complexity using the data from Table 3.1 and listed on rows 1 and 2 of Table 3.3

of the symbols in the symbol set $\mathcal{S} = \{A, B, C, D\}$. Each dance is exactly 23 symbols in length, and thus the relative frequency of occurrence of the k-th symbol is $f_k = (\#\text{of occurrences of } k-th \text{ symbol})/23$. The metric

$$\begin{pmatrix} \text{symbol} \\ \text{frequency} \\ \text{complexity} \end{pmatrix} = -\sum_{k=1}^{4} f_k \log_2 f_k \tag{3.2}$$

is then a measure of the variability of the component steps that make up the dance. Because there are only four symbols involved, the maximum value this measure could take is $\log_2 4 = 2$, which would be attained if each symbol appeared in the sequence equally often. (Since the sequence lengths are all 23, this bound is never achieved.) On the other hand, if any single symbol were to appear in all 23 places in the sequence, the complexity (3.2) would have the value 0. When the complexity metric (3.2) is evaluated on the ten dance sequences of Table 3.1, the values are strictly between the two extremes, and they are given in row three of Table 3.3.

A simple linear regression in which the average judges' scores were regressed on the computed symbol frequency suggests only a modest correlation. (See Fig. 3.8) Indeed, the value of the coefficient of correlation for the sequences is only 0.48, indicating a weak correlation. The following section describes some refined notions of complexity that may more faithfully reflect the artistic quality of the sequences.

3.4 Deconstructing the Dances into Four-Step Phrases

It is an interesting exercise to attempt to fit four-state Markov chain models to the symbol sequences of Table 3.1. While the sequences are long enough and the sets of transitions are rich enough in some cases to construct such models, any model of the

dance in which the next step depends only on its immediate predecessor step will probably seem a bit aimless and not reflective of the artistic quality of the sequence of steps that the dance actually contained. As has been noted in the computer music literature, [14], higher order Markov chain models can be used to capture the phrasal nature of music. While fitting higher order Markov models to the sequences of Table 3.1 is beyond the scope of the chapter, we shall briefly examine the phrasal structure of the sequences. In Sect. 3.5, it will be noted that physical and topological constraints impose a natural phrase structure on motion sequences in our ILS model.

As noted above, each of the four motion primitives in BLS is executed over a period of eight beats of music. Each phrase is thus eight musical measures in length. Since there are four beats to a measure, it is natural to group the letters in the sequences into four letter phrases. Several phrase centric complexity metrics can then be considered. One such metric is based on viewing each four symbol phrase as a complete dance sequence in its own right. In terms of the symbol set $\mathcal{S}$, every four letter phrase has a complexity given by (3.2) where now f_k = (# number of occurrences of the k-th symbol)/4. Clearly, there are five possible values that this phrase complexity metric can take on phrases made up of the four letters in $\mathcal{S}$. They are 0, $-\frac{1}{4}\log\frac{1}{4} - \frac{3}{4}\log\frac{3}{4} = 0.811278$, $-\log\frac{1}{2} = 1$, $-\frac{1}{2}\log\frac{1}{4} - \frac{1}{2}\log\frac{1}{2} = 1.5$, and $\log 4 = 2$ in the respective cases that all letters in the phrase are equal, three letters in the phrase are equal, there are two distinct pairs of equal letters, there are exactly three letters in the sequence, and finally in the case that there are four distinct letters in the sequence. Based on this phrase metric, we prescribe an *averaged phrase complexity* metric for each of the 23 letter sequences. Ignoring the final three letters in each sequence, the right hand column in Table 3.2 lists the number of distinct four letter phrases that make up the dance. The third row of Table 3.3 lists the averaged phrase complexity of the dance.

A further metric in terms of which to evaluate dance complexity is what we shall call the *number-of-phrases complexity*. We omit details but note that this metric is based on the number of distinct phrases and their frequency of occurrence among the first 20 letters in each dance sequence (a number between 1 and 5). The possible values of the number-of-phrases complexity in terms of the appropriately restated formula (3.2) range between 0 and $\log_2 5 \approx 2.344$. The values taken on by this metric for our ten dances are listed in row 4 on Table 3.3. Note that while dances 4 and 5 have the highest averaged phrase complexities (being comprised of four distinct letters), they also have the lowest complexity measured in terms of number-of-phrases.

Comparing the average judges' scores with the averaged phrase complexity showed a discernible correlation, with the coefficient of correlation being 0.75. On the other hand, the number-of-phrases complexity had no meaningful correlation with the judges rankings (correlation coefficient -0.099). It is interesting to note, however, that a convex combination of these complexity metrics in which the relative weightings are 90 % averaged-phrase complexity and 10 % number-of-phrases complexity has a slightly higher value of 0.764 coefficient of correlation with the judges rankings. This metric slightly discounts dance routines that repeat the same four steps over and over. It is also interesting to note that both these complexity metrics are identical on and do not discriminate between dances 4 and 5, and yet the

Table 3.2 The dance sequences grouped into four-letter phrases

		# phrases
Dance 1	$(DDAD)(BBBB)(ACCC)(DDDD)(DBDA)AAA$	5
Dance 2	$(AAAA)(AAAA)(DDDD)(DDDD)(DDDB)DBB$	3
Dance 3	$(ADBC)(DACB)(DADB)(CDAB)(ACDA)CBD$	5
Dance 4	$(DBCA)(DBCA)(DBCA)(DBCA)(DBCA)DBC$	1
Dance 5	$(ACBD)(ACBD)(ACBD)(ACBD)(ACBD)ACB$	1
Dance 6	$(ABCD)(BCDA)(CDAB)(DABC)(ABAD)BCD$	5
Dance 7	$(DBAD)(ACBD)(DBAB)(DDAA)(CDBB)DAD$	5
Dance 8	$(AAAA)(BAAA)(DAAA)(AAAA)(CAAA)DAA$	4
Dance 9	$(DBCD)(CBBD)(CBDD)(DBDD)(DAAB)CCC$	5
Dance 10	$(DBDC)(CBDD)(BBDD)(DCCC)(CABD)DDB$	5

The right-hand column lists the number of distinct four letter phrases in the sequence, and the final three letters in each sequence were not counted as a phrases

Table 3.3 There were 20 judges; numbers in parentheses in the average score row are standard deviations

Dance no.	1	2	3	4	5	6	7	8	9	10
Average score by judges	3.6 (2.0)	1.9 (1.5)	5.1 (3.0)	5.7 (2.1)	7.3 (1.8)	6 (2.6)	6 (2.2)	4.2 (3.1)	7.8 (2.4)	7.3 (2.2)
Symbol frequency complexity	1.897	1.403	1.985	1.996	1.996	1.996	1.848	0.927	1.848	1.731
Averaged phrase complexity	0.625	0.162	1.8	2	2	1.9	1.5	0.487	1.362	1.362
Number of phrases complexity	2.322	1.522	2.322	0	0	2.322	2.322	1.922	2.322	2.322
Robot dance energy	13727	12945	14326	14567	14547	14248	13349	13181	14627	14647

Robot dance energy is computed as a function of the total distance traveled on the dance floor by wheeled robots simulating the motions of the dancers

judges had a clear preference for dance 5. There is clearly some aspect of artistic merit that is not captured by the complexity metrics.

3.5 The Topological Knot Theory of Intertwined Arms

In this section, we consider an enhanced form of salsa, which we refer to as ILS and in which there are additional motion primitives (dance steps) as well as a physical constraint imposed by requiring the dance partners to maintain hand contact. We

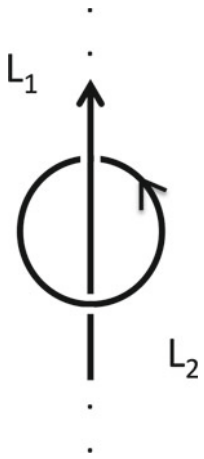


Fig. 3.9 The link diagram of L with its components L_1 and L_2

start our analysis by assuming that in an abstract model a dancer and his/her arms will be represented by a two component link L. The components are an infinitely long cylinder (or line) L_1 (representing the body or torso of the dancer) and a trivial knot L_2 (representing the arms) with the orientations given as in the Fig. 3.9.

It is assumed that the link representing the initial pose of a dance couple (Fig. 3.11) is a knot sum of the links representing each dancer. In Fig. 3.10, one can see the knot sum $L\#F$ of the links L and F with the orientations preserved and its three components, L_1, L_2 and L_3. We retain the color coding, blue and red, of the link segments to represent the male (leader) and female (follower) dance partners respectively.

It is easy to conclude from Fig. 3.10 that linking number of the link sun $L\#F$ is $\#(L, F) = 2$ and the number of crossings $cr = 4$. This link diagram above is assumed to be the starting link since it represents the link diagram for the initial pose of the dancers when they start to dance [1].

We shall address several questions: (1) How many non equivalent links are required to represent the poses that occur in a salsa performance? (2) What are the changes of link attributes corresponding to the physical movements that define the dance steps? To answer these questions, we define the set of dance moves (motion primitives) that will be the alphabet from which dance sequences of intermediate level salsa (ILS) are constructed. The key distinction that will be drawn with respect to BLS treated above is that in ILS, the dancers never break hand contact. Hence their arms and bodies remain knotted in the sense described above such that the link diagram always has three components. To the basic moves A, B, C, D that we have already defined (and which are pictorially represented in Fig. 3.17) we add seven additional moves $\{J, K, M, N, O, P, T\}$, each of which continues to be performed over the course of eight musical beats. We do not describe the foot movement used to execute these, but rather identify the distinguishing characteristic as simply the beginning and ending poses. It will turn out that the knottedness of the dancers' arms will constrain the sequencing of dance steps. Hence, unlike the beginners' salsa

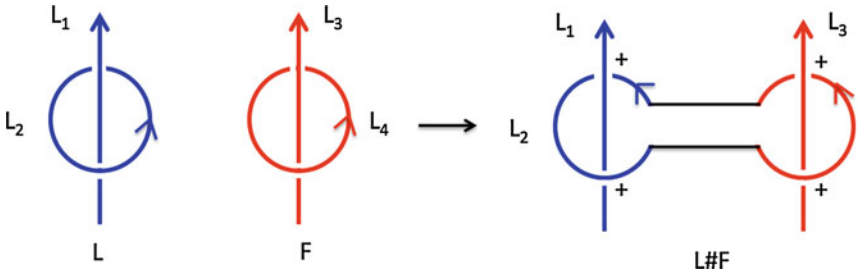


Fig. 3.10 The knot sum of the links L and F

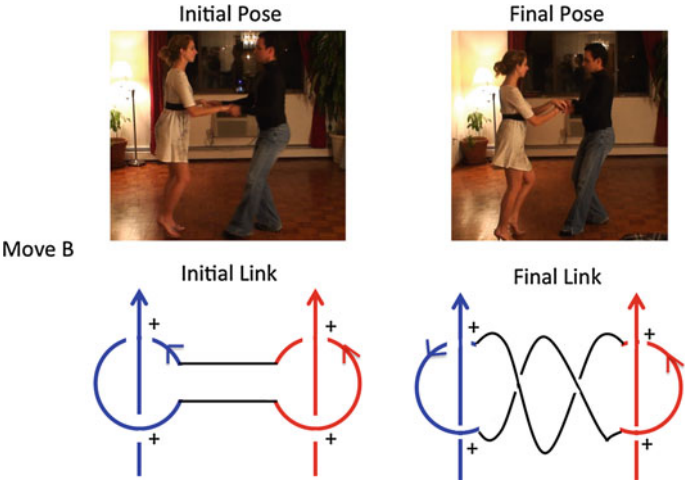


Fig. 3.11 The illustration of the initial and final poses and link diagram representations for move B

considered in Sects. 3.3 and 3.4, ILS does not allow complete freedom in the choice of letter sequences. This will, of course, affect the information theoretic metrics that we have used to discuss complexity of dance routines.

The focus here is understanding how dance movement transitions are constrained by the arm positions. The link diagrams corresponding to the poses illustrated in (Fig. 3.18) are shown in Fig. 3.12. The dancers' arms will cross in different patterns, and these patterns are shown in the usual way as under- and over-crossings in the figure.

Remark 3.1. Let q_{ji} and q_{jf} be the initial and final link diagrams representing the initial and final poses p_{ji} and p_{jf} respectively for the moves $j \in \{A, B, C, \ldots, T\}$. We can classify the links into three groups with respect to both the linking number (lk) and number of crossings (cr) (Fig. 3.12).

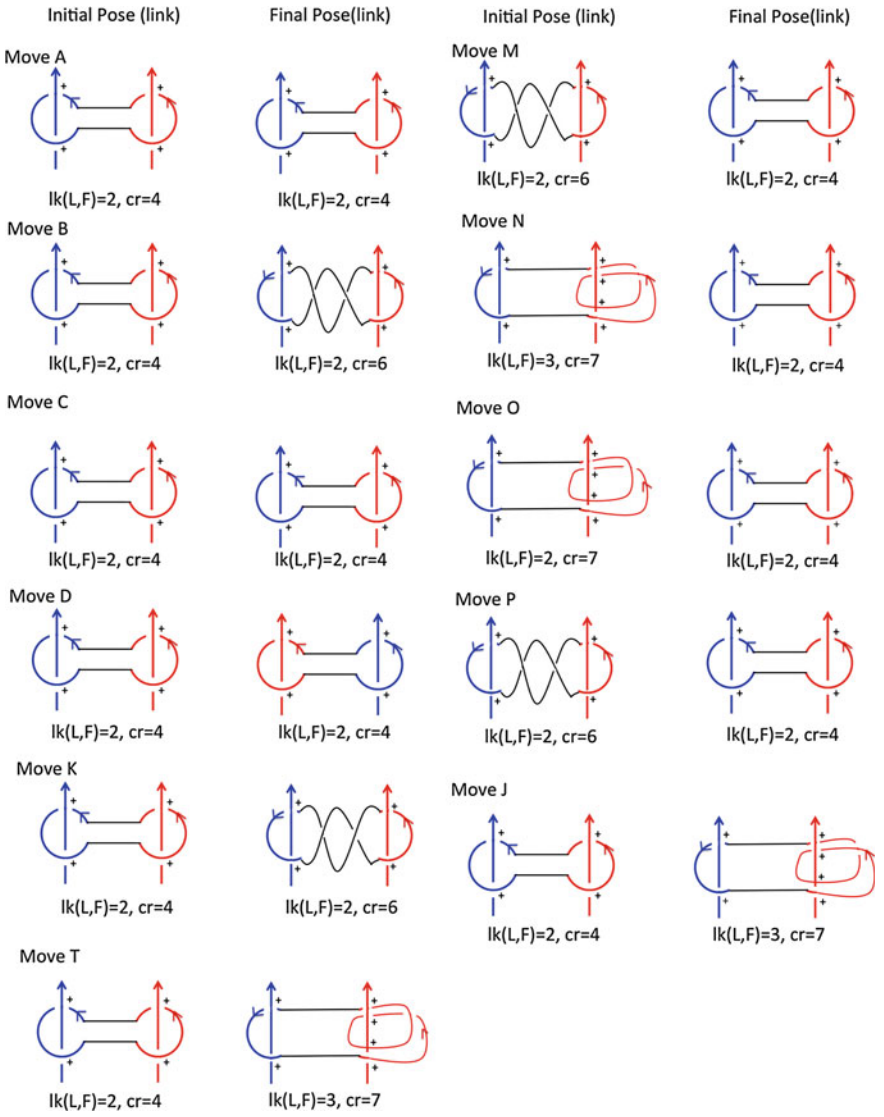


Fig. 3.12 Initial and final link diagrams for eleven moves in Intermediate Level Salsa (ILS)

- $lk = \#(L, F) = 2$ and $cr = 4$ (e.g. q_{Ai}, q_{Nf},...) : 14 link diagrams (including starting link),
- $lk = \#(L, F) = 2$ and $cr = 6$ (e.g. q_{Bf}, q_{Mi},...): 4 link diagrams,
- $lk = \#(L, F) = 3$ and $cr = 7$ (e.g. q_{Jf}, q_{Ni},...): 4 link diagrams.

Remark 3.2. The purpose of the discussion here is to illuminate the constraints on motion sequences that are imposed by the topologies of the links depicted in Fig. 3.12.

These constraints are completely determined by the beginning and ending poses of the dance steps. The significance of the constraints in terms of our complexity metrics will be discussed in the next section. Refined enhancements to the sets of dance primitives involving finely detailed descriptions of body movement, foot placement, and the artistic disconnecting and reconnecting of the dancers' hands will be discussed elsewhere. We point out that for motion primitives A, B, C, D, the foot motions of the dancers are shown in (Fig. 3.17).

We note that the crossing number (cr) is not a knot (link) invariant, but it will be shown to usefully characterize the poses corresponding to the link diagrams of Fig. 3.12. To understand this correspondence, we define a motion operator that describes the movement of the dancers in the transition from their initial to final pose in each of the eleven steps. As apparent from Fig. 3.12, the motions of the male (blue) and female (red) partners are qualitatively different from one another. The link crossings and uncrossings arise from the female rotating her body through angles of π or 2π with respect to the initial pose. The dance motions from beginning to ending poses in our eleven dance primitive can be labeled as follows in terms of a motion operator, $*(.,.)$:

$$
\begin{aligned}
*(0,0) &\sim \text{the female partner begins and ends facing the male partner in the move } (A, C, D).\\
*(\pi, CW) &\sim \text{the female partner rotates by } \pi \text{ in the clockwise direction}(J, O).\\
*(\pi, CCW) &\sim \text{the female partner rotates by } \pi \text{ in the counterclockwise direction}(N, T).\\
*(2\pi, CW) &\sim \text{the female partner rotates by } 2\pi \text{ in the clockwise direction}(B, P).\\
*(2\pi, CCW) &\sim \text{the female partner rotates by } 2\pi \text{ in the counterclockwise direction}(K, M).
\end{aligned}
$$

Consider move B. This has the follower rotating 2π in the CW direction. The change from the initial pose p_{Bi} to the ending pose p_{Bf} in move B is represented by the notation,

$$p_{Bi} * (2\pi, CW) \rightarrow p_{Bf}. \tag{3.3}$$

The $*(.,.)$ operator will be used to describe the rotations in ILS. The move descriptions are listed in Fig. 3.13.

From this figure, it is clear that the physical actions of the follower is one of the four $*(2\pi, CW)$, $*(2\pi, CCW)$, $*(\pi, CW)$, $*(\pi, CCW)$ together with the null

$$
\begin{array}{rl}
p_{Ai} * (0,0) & \rightarrow p_{Af}, \\
p_{Ci} * (0,0) & \rightarrow p_{Cf}, \\
p_{Di} * (0,0) & \rightarrow p_{Df}, \\
p_{Bi} * (2\pi, CW\ ) & \rightarrow p_{Bf}, \\
p_{Ji} * (\pi, CW\ ) & \rightarrow p_{Jf}, \\
p_{Ki} * (2\pi, CCW\ ) & \rightarrow p_{Kf}, \\
p_{Ti} * (\pi, CCW\ ) & \rightarrow p_{Tf}, \\
p_{Mi} * (2\pi, CCW\ ) & \rightarrow p_{Mf}, \\
p_{Ni} * (\pi, CCW\ ) & \rightarrow p_{Nf}, \\
p_{Oi} * (\pi, CW\ ) & \rightarrow p_{Of}, \\
p_{Pi} * (2\pi, CW\ ) & \rightarrow p_{Pf}.
\end{array}
$$

Fig. 3.13 10 moves in which there is follower rotation

rotation (of move A etc.). The link transformations associated with the five motion operations are characterized as follows.

Proposition 3.1. *In ILS, the physical transformations* $*(2\pi, CW)$, $*(2\pi, CCW)$ *will result in topologically equivalent (under Reidemeister transformations) initial and final link representations.*

$$p_{\ell i} * (2\pi, .) \rightarrow p_{\ell f}, \quad q_{\ell i} \rightarrow q_{\ell f}, \quad q_{\ell i} \approx q_{\ell f}, \quad \ell = B, K, M, P. \tag{3.4}$$

However, the physical transformations $*(\pi, CW)$, $*(\pi, CCW)$ *will have non equivalent initial and final links.*

$$p_{\ell i} * (\pi, .) \rightarrow p_{\ell f}, \quad q_{\ell i} \rightarrow q_{\ell f}, \quad q_{\ell i} \not\approx q_{\ell f}, \quad \ell = J, N, O, T. \tag{3.5}$$

Proof. For the proof we begin by proving the second part of our result. Links whose linking numbers differ cannot be topologically equivalent. Hence in Fig. 3.12, one may observe the moves J, N, O, T, which involve applications of the $*(\pi, \cdot)$ operator are such that the linking number changes by 1, proving that the links are not equivalent. In order to make the link diagrams of the beginning and ending pose equal, one would need to cut need to cut and splice one of the link components as illustrated in Fig. 3.14—the black circle showing the location of the cutting operation. In order to prove the equivalence of the links representing the poses after the $*(2\pi, .)$ operator is applied, it is enough to show the proper elementary Reidemeister transformation between the initial and final links. One can apply a Reidemeister *Type 1* move (Fig. 3.2) (inverting the right or left vertical link segment L or F) to decrease the number of crossings in the connecting link while preserving the link invariants, and showing the equivalence with the link diagram corresponding to move A.

Remark 3.3. It is important to note that the Reidemeister moves that render the link diagrams corresponding to $*(2\pi, \cdot)$ motions equivalent to the $*(0, 0)$ motion

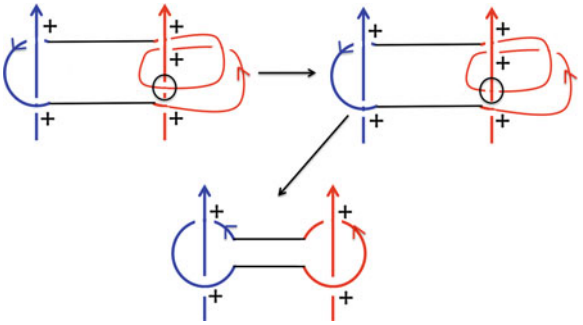


Fig. 3.14 The cut-and-splice operation of Proposition 1 that transforms the ending to the starting link diagram (*black circle* in the figure is the crossing at which the cutting operation is applied in order to alter the over and under passes of the link segments)

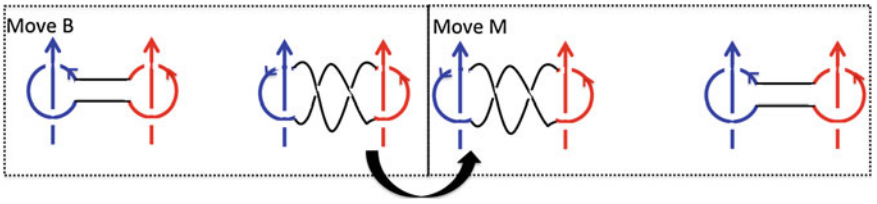


Fig. 3.15 Transition from move B to the move M with link diagram representations

in Proposition 1 would be impossible to physically realize by the dancers. Thus, both the $*(\pi, \cdot)$ and $*(2\pi, \cdot)$ operators play an equivalent role in constraining the motion transitions. If we relax the continual hand holding requirement, however, the distinction between linking number changes under the $*(\pi, \cdot)$ and $*(2\pi, \cdot)$ operators becomes significant. After the $*(2\pi, \cdot)$ motions, the female dancer does not need to move to return to the initial pose. The dancers can simply release and regrasp each other's hands. After the $*(\pi, \cdot)$ motions, however, the female dancer must rotate her body by $\pm\pi$ to return to the starting pose.

Proposition 3.1 characterizes the change of the link topology produced by the corresponding motions in each dance step. These changes and the fact that ending and beginning poses may have topologically distinct link diagrams places constraints on assembling admissible sequences of moves in a dance. These may be understood in terms of differences the among the moves illustrated in Fig. 3.12. Moves in the set $\{A, C, D\}$ start and end with the starting link. Moves in the set $\{T, J, K, B\}$ only start with the starting link, and $\{O, N, P, M\}$ only end with the starting link. The move transition pairs BM, JN, KP, TO also start and end with starting link. Move M can be thought as an "inverse" move of move B—one that transforms its final link back to the starting link (Fig. 3.15). This raises the question of finding all possible move transitions in ILS. We shall describe this feasible set in terms of the link diagrams of the beginning and ending poses.

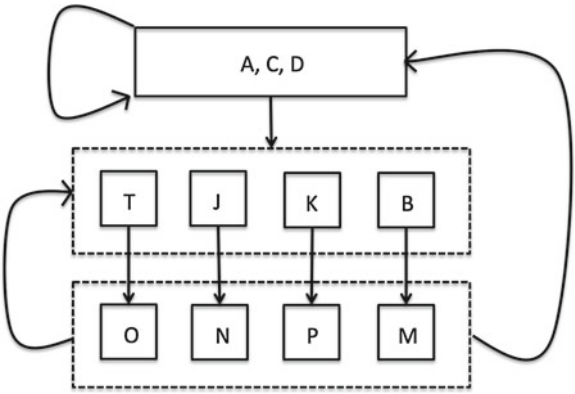


Fig. 3.16 The allowable transitions for each move in ILS based on the physical constraints

Proposition 3.2. *Assume that q_{ji} and q_{jf} are the initial and final link diagram representations for the initial and final poses p_{ji} and p_{jf},*

$$j \in \{A, B, C, D, J, K, M, N, O, P, T\},$$

respectively. Then any admissible dance sequence can be represented by a finite concatenation of brackets $[q_{ji}, q_{jf}]$ in which any pair of successive brackets has the final link diagram of the first bracket equal to the initial link diagram of the second bracket.

Discussion: It is obvious that with respect to the physical constraints on the dance, the final pose of a move performed and the initial pose of the next must be exactly the same for the dancers to transition from one to another. Because of these physical constraints, move transitions in a dance sequence $\ldots [q_{ji}, q_{jf}][q_{ki}, q_{kf}] \ldots$ are allowed only when the corresponding final link diagram q_{jf} of the previous move and the initial link diagram of the next move q_{ki} where $k, j \in \{A, B, C, \ldots T\}$ are the same.

By looking at the link diagram representations of the poses given in the Fig. 3.12, one may conclude that there are three types of moves in an admissible dance sequence. We can summarize the allowable transitions $\ldots [q_{ji}, q_{jf}][q_{ki}, q_{kf}] \ldots$ as follows.

- if $j \in \{A, C, D\}$ then, either $k \in \{A, C, D\}$ or $k \in \{T, J, K, B\}$.
- if $j \in \{T, J, K, B\}$ then the transitions are deterministic such that, BM, JN, KP, TO must appear in the sequence. (This is similar to the need to have the letter u follow the letter q in English).
- if $j \in \{O, N, P, M\}$ then either $k \in \{A, C, D, \}$ or $k \in \{T, J, K, B\}$.

This may be summarized in the state transition diagram of Fig. 3.16.

3.6 Complexity Merit of an Intermediate Level Salsa Performance

The transition constraints imposed by the knotting and unknotting of the dancers' arms will affect the complexity metrics of Sect. 3.3. The empirical comparison of rankings based on these metrics with human (e.g. judges') perceptions of artistic merit is work in progress. The intrinsic complexity of the enhanced (ILS) dance sequences can be discussed in terms of a Markov model of the step transitions, however, and this complexity can be compared with the corresponding model of BLS discussed above. Assuming no particular biases in the choice of step sequences allowed according to Fig. 3.16, we let the probabilities of steps A, C, D, B, J, K, T following step A in a sequence be $1/7$. Assigning transition probabilities in a similar way in accordance with the given transition constraints, we can model dance sequences as a Markov chain with transition matrix

$$P = \begin{pmatrix} 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 0 & 0 & 0 & 0 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 0 & 0 & 0 & 0 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 0 & 0 & 0 & 0 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 0 & 0 & 0 & 0 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 0 & 0 & 0 & 0 \\ 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{3.6}$$

It is easy to see that this is a doubly stochastic matrix, and since it is aperiodic and irreducible, the associated invariant distribution is the uniform distribution on the 11 dance steps. That is, the invariant probabilities are $\mu_i = 1/11$ for $i \in \{A, B, C, D, J, K, M, N, O, P, T\}$. The entropy of this is $\log_2 11 = 3.46$, whereas the entropy of the four-letter sequences of beginner's salsa is $\log_2 4 = 2$. What is perhaps more revealing is to compare the entropy rates of the unconstrained beginner's salsa and the intermediate level salsa. Recall [3] that given a random walk X_i on a graph described by a Markov transition matrix P having stationary distribution $\mu_1, \ldots, \mu_{11}$, the entropy rate is given by

$$-\sum_i \mu_i, \sum_j P_{ij} \log_2 P_{ij}.$$

For the systems described by (3.6) this is 1.7865, which is less than $\log_2 4 = 2$ for the unconstrained four state system of beginner's salsa.

In thinking about the complexity metrics of Sect. 3.3, we note that some differences can be expected with intermediate level salsa, but some metrics will be similar. For the intermediate level case, we have

$$\begin{pmatrix} \text{symbol} \\ \text{frequency} \\ \text{complexity} \end{pmatrix} = -\sum_{k=1}^{11} f_k \ \log_2 f_k. \tag{3.7}$$

Since there are eleven symbols in ILS the maximum value of this measure is $\log_2 11 = 3.45$ which is simply performing each move equally often in a given n symbols length dance sequence. On the other hand this value remains 0 when only one move appears the sequence. The *average phrase complexity* metric can be calculated by the eq. 3.7 where

$$f_k = (\#\text{of occurences of the k-th symbol})/4.$$

Thus, the complexity value for the whole sequence is calculated by taking the average of the total number of phrases. There are four distinct possibilities of phrase complexities, 0, 0.811278, 1, 1.5, 2 when all four symbols are equal, three symbols are equal, two symbols are equal and when there are four distinct symbols in a phrase respectively. These cases are exactly the same as in beginner's salsa where there were a total of four distinct symbols. The *number-of-phrases complexity* which takes values that range between 0 to $\log_2$(total number of phrases) will also remain roughly the same in ILS, although the set of possible distinct phrases is much larger.

It was noted that the energy consumed in performing a particular dance sequence in our beginner's salsa performances had a measurable influence on the judges rankings [1]. Our measure of the energy consumed was simply the distance covered by an equivalent wheeled robotic dancer that followed the planar path of the human performers. The extension of this idea to the ILS needs to account for additional movement since the majority of the motions in ILS are based on the dancers' rotations and arm movements. New sets of experiments with an enhanced set of performance metrics are work in progress.

3.7 Conclusion

The chapter has described a number of figures of merit of dance performance and considered both a beginner level and intermediate level salsa. The particular intermediate salsa discussed in Sects. 3.5 and 3.6 entails dance movements in which the dance partners remain in physical contact—holding each other's hands. This physical contact restricts the possible motion sequences that can be executed insofar as certain eight beat dance moves cannot be followed by certain other moves without breaking hand contact. It was shown that because of this, the entropy rate of a Markov model of the intermediate level salsa was less than that of the beginner's salsa, even

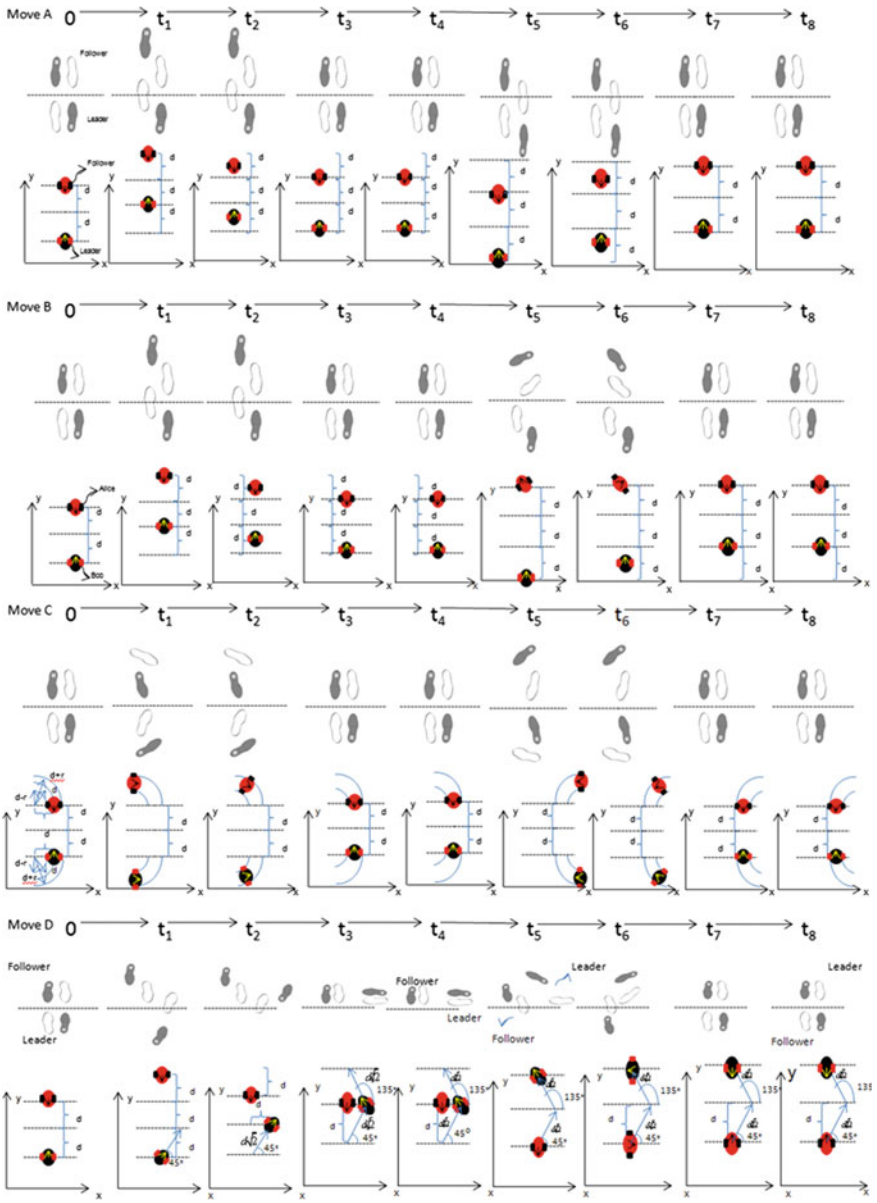


Fig. 3.17 Basic Salsa—four dance steps: from *top* to *bottom A*, *B*, *C*, *D*

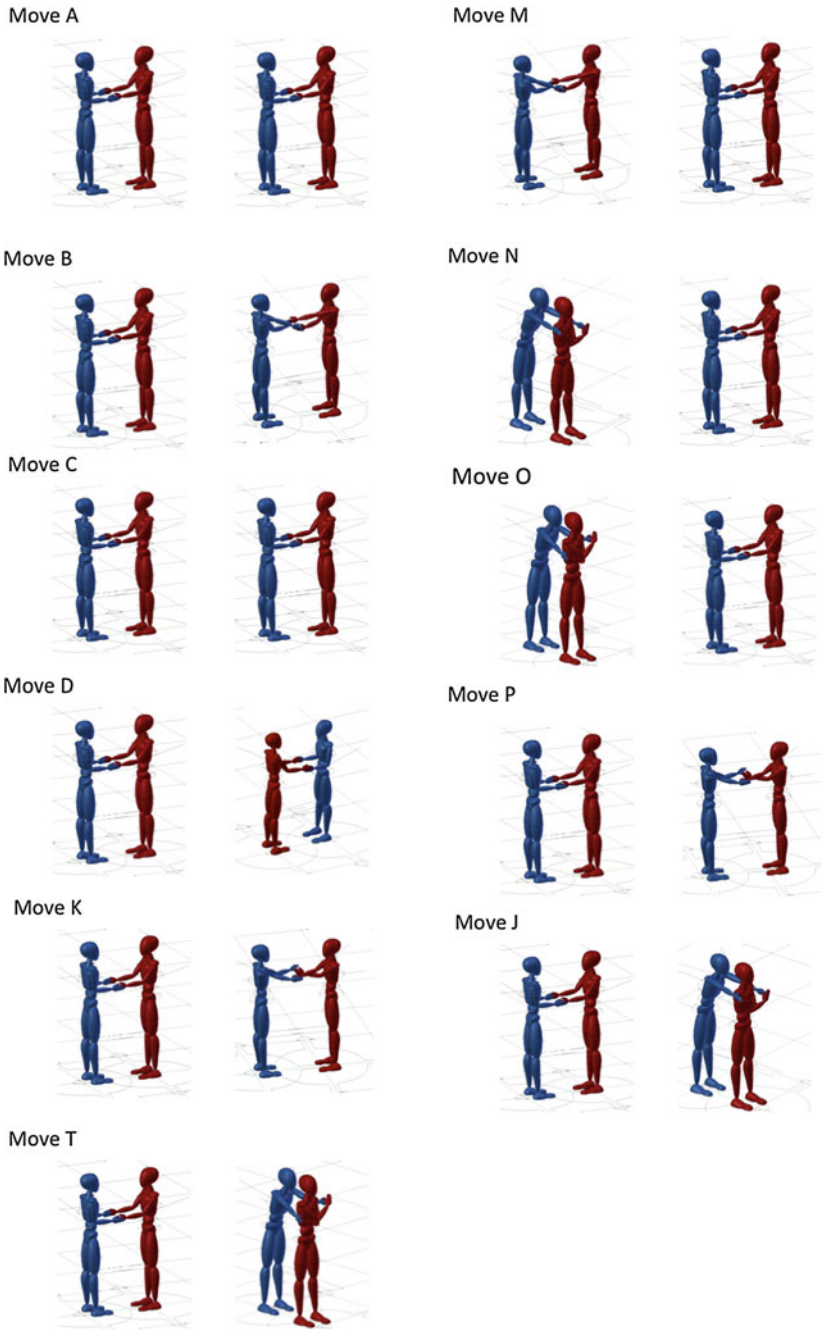


Fig. 3.18 Initial and final poses for eleven moves in *intermediate level salsa* (ILS)

though the ILS uses an alphabet of eleven motion primitives as compared with four motion primitives for BLS. This smaller entropy rate implies that dance phrases as opposed to purely random sequences of steps appear naturally in ILS. It was also noted in Sect. 3.4, that *averaged phrase complexity* of dance routines correlated well with judges rankings.

While the topological constraints on movement are useful in developing a grammar of movement in dance, trained dancers do not always maintain the physical contact that we have studied in Sects. 3.5 and 3.6. It is hoped that further research will illuminate the role of making and breaking physical contact in determining the perceived artistry of dance. Even with such expressive freedom, we conjecture that artistic performances will exhibit an underlying syntax of motion sequences. Some evidence of this appears in the work of LaViers et al. [8].

References

1. Baillieul J, Özcimder K (2012) The control theory of motion-based communication: problems in teaching robots to dance. In: 2012 American control conference, Montreal, pp 4319–4326, 27–29 June 2012
2. Baronov D, Baillieul J (2011) A motion description language for robotic reconnaissance of unknown fields. European J. Control 17:5–6, 512–525. doi:10:3166/EJC.17.512-525
3. Cover TM, Thomas JA (2006) Elements of information theory, 2nd edn. Wiley-Interscience, Hoboken
4. Dutton D (2010) The art instinct: beauty, pleasure, and human evolution. Bloomsbury Press, New York, 288 pp. ISBN-10:1608190552
5. Hopcroft JE,Ullman JD (1969) Formal languages and their relation to automata. Addison-Wesley, Reading. ISBN-13: 978–0201029833
6. Hutchinson Guest A (2004) Labanotation—the system of analyzing and recording movement. Routledge, New York. ISBN 0-415-96562-4
7. LabanWriter software. http://dance.osu.edu/3_research_gallery/laban_writer.html
8. LaViers A, Teague L, Egerstedt M (2014) Style-based robotic motion in contemporary dance performance. In: LaViers A, Egerstedt M (eds) Control and art. Springer, Berlin (to appear)
9. Leonard NE, Young G, Hochgraf K, Swain DT, Trippe A, Chen W, Fitch K, Marshall S (2014) The dance studio: an art and engineering exploration of human flocking. In: LaViers A, Egerstedt M (eds) Control and art. Springer, New York (to appear)
10. Leonard NE, Young G, Hochgraf K, Swain D, Chen W, Marshall S (2012) In the dance studio: analysis of human flocking. In: 2012 American control conference, Montreal, pp 4333–4338, 27–29 June 2012
11. Lickorish WBR (1997) An introduction to knot theory. Graduate texts in mathematics. Springer-Verlag, New York. ISBN 0-387-98254-X
12. Przytycki JH (2007) History of knot theory, http://arxiv.org/pdf/math/0703096v1.pdf
13. Reidemeister K (1932) Knotentheorie, Ergebn. Math. Grenzgeb, 1983. Bd. 1; Berlin: Springer-Verlag (English translation: Knot theory, BSC Associates, Moscow, Idaho, USA)
14. Roads C (ed) (1996) The computer music tutorial, MIT Press, Cambridge. ISBN 0262181584
15. von Laban R (1995) Kinetografie Labanotation. Einfhrung in die Grundbegriffe der Bewegungs- und Tanzschrift Herausgegeben von Claude Perrottet. Noetzel, Wilhelmshaven. ISBN 3-7959-0606-7

Chapter 4
So You Think You Can Dance? Rhythmic Flight Performances with Quadrocopters

Angela P. Schoellig, Hallie Siegel, Federico Augugliaro and Raffaello D'Andrea

4.1 Rhythmic Flight with Quadrocopters

Fly with the music. —Song title by DJ Grande

This chapter presents a set of algorithms that enable quadrotor vehicles (such as the ones depicted in Fig. 4.1) to "fly with the music"; that is, to perform rhythmic motions that are aligned with the beat of a given music piece.

We design feasible periodic motion patterns based on a model of the quadrocopter, which describes the dynamic capabilities of the vehicle. Control algorithms based on the vehicle model stabilize the vehicle in the air and guide it along the desired flight paths. However, without additional adaptation algorithms, the quadrocopter does not follow the desired path with the required accuracy resulting in a motion that is not in sync with the music. To perfect the vehicle's flight performance, measurements obtained from flight experiments are used to adapt the motion parameters sent to the vehicle ('commanded trajectory' in Fig. 4.2). This adaptation can be done online (during a flight performance) or offline (before a flight performance). The results are

This chapter summarizes results that have previously been published in [1–5]. Parts of those papers are reproduced here for the sake of completeness..

A. P. Schoellig (✉)
Institute for Aerospace Studies, University of Toronto, Toronto, Canada
e-mail: schoellig@utias.utoronto.ca

H. Siegel · F. Augugliaro · R. D'Andrea
Institute for Dynamic Systems and Control, ETH Zurich, Zurich, Switzerland
e-mail: siegelh@ethz.ch

F. Augugliaro
e-mail: faugugliaro@ethz.ch

R. D'Andrea
e-mail: rdandrea@ethz.ch

A. LaViers and M. Egerstedt (eds.), *Controls and Art*,
DOI: 10.1007/978-3-319-03904-6_4,

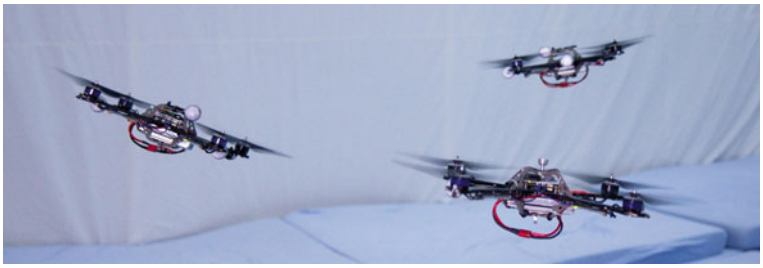


Fig. 4.1 A flight performance of multiple quadrocopters timed to music. (*Photo* Federico Augugliaro)

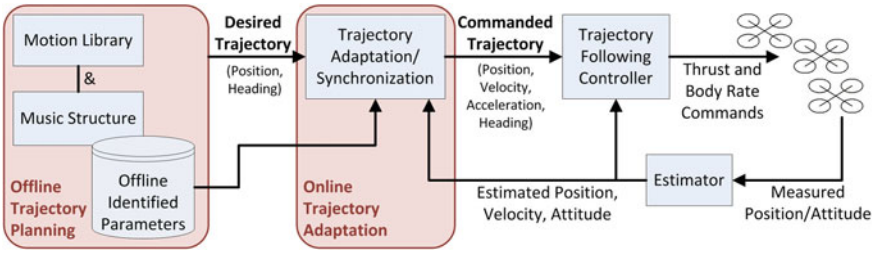


Fig. 4.2 High-level control architecture used for implementing rhythmic flight performances. Key components are the offline trajectory planning and online trajectory adaptation. (Position, velocity, and acceleration refer to the translational coordinates and heading corresponds to the vehicle yaw)

flight maneuvers that closely follow the desired periodic motion pattern ('desired trajectory' in Fig. 4.2) and align with the beat of the music.

This work can be viewed as a proof-of-concept result that shows the feasibility of rhythmic flight and represents an important step toward our vision of creating multivehicle aerial 'dance' performances.

4.1.1 Vision of a Quadrocopter Dance Performance

It takes an athlete to dance. But it takes an artist to be a dancer.
—Shanna LaFleur

Quadrocopters are exceptionally agile and "athletic" vehicles, but it takes more than agility to create a musical flight performance that is both viable and convincing. We envision a troupe of quadrocopters flying together across a big open stage—their movement choreographed to the rhythm of the music, their performance coordinated and skilled, and their choreography well-suited to their abilities and to the character of the music. A quadrocopter "dance".

A preliminary framework for designing and executing coordinated flight choreography to music has been implemented at the ETH Flying Machine Arena (Fig. 4.3).

Fig. 4.3 The ETH flying machine arena. *Left* Schematic drawing showing the motion capture camera system that provides accurate measurements of the vehicle's six degrees of freedom, position, and attitude. *Right* Photo of the installation at ETH Zurich. (*Photo* Raymond Oung)

In this framework, the underlying vehicle control is done automatically, while the high-level motion design is left to a human "choreographer". This work-in-progress currently enables the human operator to generate choreographies by assigning motion elements to individual music segments that correspond to the music's character. In addition to the algorithms presented herein, support is provided by, for example, a library of predefined, parameterized motion elements and a collision-free trajectory generator, which can be used for smoothly connecting single motion elements. Video sources of various quadrocopter flight performances are found at www.tiny.cc/MusicInMotionSite.

4.1.2 Artistic Motivation

> *As robots have grown more advanced, they have become our mirrors, as we watch the way they perform activities that we do as well. And as we watch, secrets are unlocked—secrets about how we, housed in our own biological frameworks, operate.*
>
> —Rodney Brooks, roboticist and entrepreneur

The embodied mind thesis [6], which straddles such diverse fields as philosophy, psychology, cognitive theory, neurobiology, robotics, and artificial intelligence argues that all aspects of cognition are shaped by the experiences of the body; that how we perceive the world around us (through our sensory system) and how we move through and interact with this world (through our motor system) intrinsically determines the ways in which we think and experience. Proponents of "embodied AI", such as Rodney Brooks [7] and Rolf Pfeifer [8], argue that for machines to be truly intelligent, they must have sensory and motor skills, and be connected to the world through a body.

It is interesting to consider the idea of "embodiment" also from the perspective of professional dancers, choreographers, and athletes—people for whom the ability to sense and move in the world forms a critical part of their work. In a paper entitled

"The Dance: Essence of Embodiment" [9] the philosopher/dancer duo Betty Block and Judith Lee Kissell describe dance as an "embodied way of being-in the-world," and that "an analysis of dance is a profoundly enriching way to better understand embodiment itself." In other words, to dance is to be an expert in embodiment.

It is no wonder, then, that robotics researchers have turned to dance as a means of understanding gesture and movement. Examples are provided in the subsequent Sect. 4.1.3.2. Note that many of these robotic/dance experiments involve humanoids and/or robotic arms that mimic human limbs. Indeed, mimicry is a proven means of generating understanding: much can be learned by reverse-engineering human movements and gestures.

But what happens when the "body" is not human? When the body is no longer constrained by the limits of arms, legs, torso, and head? In this research project, where quadrocopters learn and perform "dance", mere mimicry of human movement is no longer sufficient. A whole new meaning of "embodiment" begins to emerge.

It is obvious that the quadrotor body is mechanically different from the human body. It does not have arms, legs, or a head, but instead has rotating blades. Because it flies, it occupies three-dimensional space in a way that we humans cannot. Its movements are fundamentally different from ours: while we generate movement by pushing off a hard surface (such as the ground), a quadrocopter creates movement by "pushing" on air. These fundamental differences make it a challenge to design motions for quadrocopters that can be recognized as dance by humans, and that can been interpreted by human eyes as being "expressive".

Yet for all these differences, when it comes to dance performance, quadrocopters and humans share much in common as well. First and foremost, "dance"—whether performed by humans or by quadrocopters—is an exploration of three-dimensional space that must respect the boundaries of both the performance space and the body of the performer. Both humans and quadrocopters have limits to their abilities, and not every sequence of movements is feasible. In human dance, during a ballet barre exercise, for example, a Développé movement does not follow logically from a Plié (see Chap. 9); for quadrocopters, subsequent movements require smooth transitions without jumps in the vehicle position or attitude. Rhythmic ability is another feature shared by both humans and quadrocopters: when music is present, human motion is easily adapted to its meter, and with beat extraction software, this feat is accomplished by quadrocopters, too. Another commonality is the ability to dance in groups: human dance performances often feature troupes of dancers interacting with each other in a shared space; advances in trajectory planning allow quadrocopters to also share a space in a coordinated fashion without fear of collision. Humans also practice to perfect their skills—something we can enable in quadrocopters as well using parameter learning schemes [4]. And finally, humans teach and learn from each other; while cooperative machine learning remains to be explored in-depth, current research in this area is promising and suggests that shared learning could greatly enhance the learning process.

For robotics researchers, it is these commonalities that make an experiment in quadrotor dance so interesting. If the mechanical differences between humans and quadrocopters make it challenging for us to see them as "dance objects" or

"dancers", these differences are also what make quadrocopters capable of exploring and experiencing three-dimensional space in a way that humans physically cannot. For example, quadrocopters can engage with the three-dimensional space of the stage, including its full height, and can leverage air to generate movement—feats no human can do. In other words, what is challenging about quadrotor dance is also potentially liberating: when humans interface with quadrocopters by composing and executing quadrotor choreography, it opens up a new means of extending our own bodies into new physical and technological worlds.

Seen in this light, quadrocopters could become our dance partners, and the human–machine interface could become the cybernetic means through which we extend ourselves into new ranges of space and motion. This project is a first step toward that vision.

4.1.3 The Interplay of Dance and Technology

> *Dance and technology can shake hands but not at the expense of forgetting the essence of dance.*
> —Tero Saarinen, dancer and choreographer

The interplay of dance and technology has long been a space for experimentation, and a source for inspiration, innovation, and new developments. While technology has provided new means for dance expression and challenged dancers to rethink their art, dance has often challenged the state of the art of technology and motivated new technological developments. An early example is the theatrical lighting pioneered by the dancer Loie Fuller in the 1890s. Loie Fuller incorporated multicolored light in her performances and established stage lighting as a new dimension for dance expression. In addition, Fuller's work pushed the boundaries of current technology and resulted in several patents related to stage lighting technology.

4.1.3.1 Information Technology and Dance

In the past 50 years, computer and information technology have influenced and transformed dance. The term "dance technology" has become a synonym for the relationship between dance and information technology [10, 11]. Attracted by the potential of this new field, dance performers, teachers, choreographers, and computer scientists have explored the partnering of two disciplines that are, as stated in [10], quite different: "Dance and technology make seemingly odd partners. Dance is the most ethereal of art forms and computer technology perhaps the most concrete of sciences. Whereas technologists deal with the logical, the scientifically verifiable, dancers, as artists, deal with the illogical, i.e. inspiration and finding truth in that which cannot be spoken."

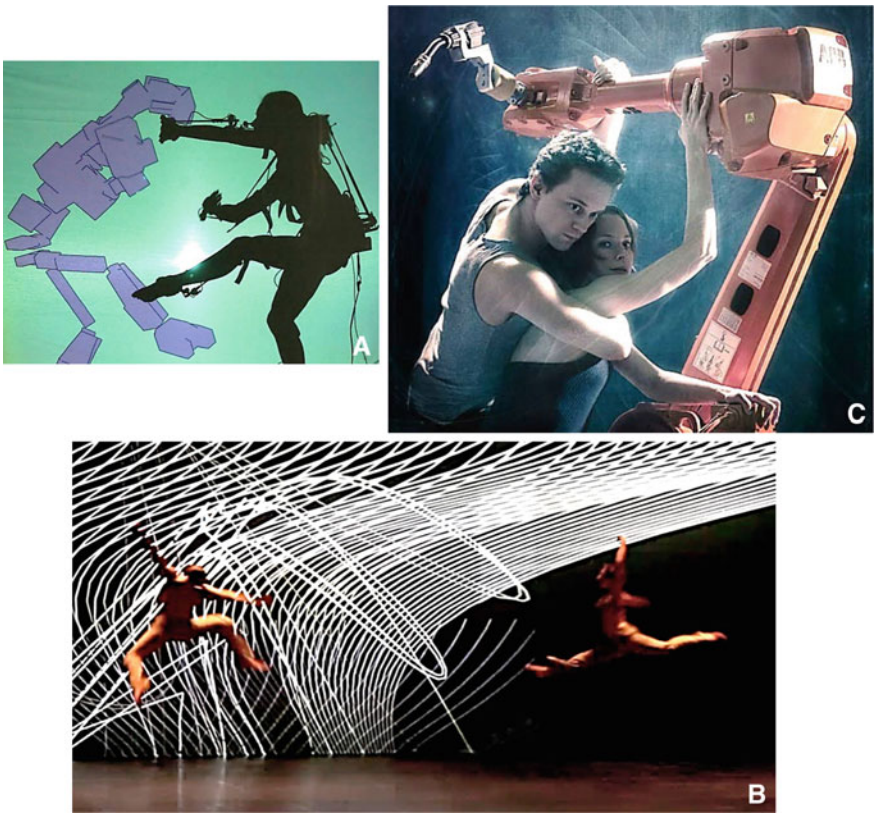


Fig. 4.4 The interplay of dance and technology. **a** Performance *100d11A0N1C00E1* of Carol Cunningham, April 2003: real-time animated projection of dancers' movement onto three large screens using motion capture (*Photo* David Umberger); **b** Performance *Apparition* of Klaus Obermaier and Ars Electronica Futurelab (www.exile.at), Ars Electronica 2004: interaction of dance and multimedia with real-time visual content generation; **c** Performance *Human interface* of Thomas Freundlich, May 2012: two dancers and two industrial robots perform together (*Photo* Johanna Tirronen)

Work at the interface of information technology and dance has advanced both disciplines by (i) integrating dance, emotions, and human character into computer technology and animations, and (ii) establishing new analysis tools and means of expression for dance performances.

Work in (i) has focused on computer graphics applications and aimed to create human-like virtual characters that are able to dance. Human motion capture data has been used to understand, model, and *imitate* human dance behavior [12, 13].

In (ii), information technology has led to new methods for expressing, creating, assessing, and instructing dance (cf. [14, 15]). Results include interactive performances of human dancers with computer-generated sound and images (e.g., computer-animated virtual dancers projected on a wall) [15–17], responsive

environments where the dancer's movement controls video, audio, and/or lightning [18, 19], computer-assisted choreography design based on a language for human movement representation and on virtual dance animation [11, 20], motion tracking and capture to record or teach a piece [21], and multimedia in dance education [22, 23]. Figure 4.4, photos a and b show two different stage performances that explored the technological possibilities for new forms of dance expression. Obermaier says [24], "The goal was to create an interactive system that is much more than simply an extension of the performer, but is a potential performing *partner*". Carol Cunningham summarizes her work as follows, "Motion capture is another tool for expression. The image may be on screen and generated by technology, but it's an *extension of the body*". Resembling human movement (Fig. 4.4a) and extending human motion into new spaces (Fig. 4.4b) were goals of the dance and technology partnering with the following outcome [18]: "The new convergences between dance and technology reflect back on the nature of dance, its physical-sensory relationship to space and the world, its immediate, phenomenological embodiedness, its lived experience in one place".

4.1.3.2 Robotics and Dance

As technology has advanced in the last 10 years and robots have become more approachable, they have found their way into dance just as information technology has done before. The physical embodiment of robots and their abilities to interact provide a new means for dance expression as well as for studying human–robot interaction and human dance.

Research on robotics and dance has come a long way: from building robots that are capable of executing human-like motions and enabling them to imitate human dance, to enabling robot–human interactions in dance performances, adapting robot dance to human preferences, and understanding human dance through robots. As dance has previously been a human-centered discipline typically designed, performed, and evaluated by humans, research into "dancing robots" has primarily dealt with humanoid robots and aimed for human-like behavior in robots. In this work, we consider a new embodiment—a group of flying robots—but still face similar questions such as: What is dance? What do humans recognize as dance? Which algorithms enable dance-like behavior in robots?

First approaches toward robotic dance of humanoid robots tried to imitate human dance motion. In [10] basic dance motions for a robotic arm were designed using choreographic elements from human dance such as shape, space, time, and force. For humanoid robots, data from human demonstrations (obtained using a motion capture system) was used to define basic robot dance motions, which—when concatenated—create full robot choreographies [25–28]. A perfect example for human imitation is a female android created by Japanese roboticists, which sings and dances along with

a troupe of humans.[1] The android's motion was created by a professional choreographer using a tool proposed in [29].

Recent work aims to understand the rules of human dance, which may ultimately lead to a larger robot autonomy when executing dance. One approach is based on the concept of human dance styles and detailed in the Chap. 9 of this book. Instead of robots that follow preprogrammed motions, various styles of human movement are defined, which in turn can be reproduced on a humanoid robot by generating sequences of motions autonomously based on underlying rules. Other approaches presented in this book try to understand human flocking in order to derive multiagent behavior (see Chap. 2) and the human communication through movements (Chap. 3). Other concepts that could explain what humans recognize as dance are skill-based approaches [30] (defining fundamental joint relationships such as Opposite, Symmetry and Formation and learning likable sequences from human feedback), effects related to motion synchrony and timed repertoire changes [31], and automatic motion selection based on musical mood [32] or musical emotions [33, 34].

Moreover, researchers currently investigate the interaction between humans and their robotic counterparts, and potential adaptation schemes for robots. The adaptation of a robot's dance behavior to human preferences is described in [35]. In [36–39] the rhythmic behavior of the robot adapts to the human based on appropriate estimation techniques that predict the human motion. Stage performances focusing on the human–robot interaction include Thomas Freundlich's performance in Fig. 4.4c and also the work in Chap. 9 of this book. Moreover, recently two artistic performances have featured quadrocopters on stage with human actors/dancers [40, 41]; these focused on the interplay between humans and machines, and had skilled human operators for controlling the quadrocopters.

4.1.3.3 Relationship to Our Work

The history of technology and dance provides a great context for our experiment, where the performers of the dance are a swarm of quadrotor vehicles. Their flight capabilities may offer—similarly to how humanoid robots have done before—new means of dance expression, including motions in the full three-dimensional space. New challenges result from the nonhuman-like body shape and motion characteristics. While work on humanoid robots has largely imitated human dance behavior, choreographies for quadrocopters must rethink the question, "What do humans recognize as dance?," and define quadrotor motions accordingly. Nevertheless, ideas for human dance choreography (such as shape, space, time and force) and concepts developed for humanoid robots may partially apply and/or may be a great source for inspiration. Overall, by studying concepts and algorithms for creating "dance-like" performances (including human–robot interaction, adaptation to the human behavior or motion planning) not in the context of the human body may enable us to understand more generally what makes robots move in a way that humans can relate to.

[1] Video found at http://youtu.be/3JOzuTUCq6s.

4.1.3.4 A Final Note

Human dance has proven to be an inspiration for technology developments. Moreover, technology has proven to extend the vocabulary of dance creation and performance to an extent that we are often not aware of. An example of the tangible connection between robots and humans is a (human) dance style called "robot dance" that became popular in the 1980s and that attempts to imitate a dancing robot. The style is characterized by jerky mechanical movements. Inspiration inevitably goes both ways: from human dance to technology and from technology to human dance. Just as our project is a robotics research experiment, it is also an experiment in dance and choreography.

4.1.4 First Steps Toward a Rhythmic Flight Performance

Art challenges technology, and technology inspires art.
—John Lasseter, chief creative officer at Pixar and Walt Disney Animation Studios

John Lasseter's quote reflects the character of many past contributions at the interface of dance and technology (cf. Sect. 4.1.3). It also provides the context for our work toward a rhythmic flight performance of multiple quadrocopters. While the technological capabilities available today (such as small-sized off-the-shelf flying robots) inspired us to think about "dancing quadrocopters" in the first place, implementing an aerial choreography challenged the current knowledge in multivehicle autonomous flight and led to novel research results, cf. [1–4, 42].

In this chapter, we focus on the research questions that are at the core of the proposed project. We show how control theory can be used to approach these questions analytically, and offer an intuitive explanation of our findings.

In particular, the topics investigated in this book chapter are:

1. Quadrocopter Dynamics: How do quadrocopters move? Which motions are possible with quadrocopters?
2. Motion Design: How to generate "dance-like" quadrocopter motions?
3. Motion Feasibility: Which motions are feasible given the actuator and sensor constraints of the vehicle?
4. Quadrocopter Control: How do quadrocopters execute their movements?
5. Motion Synchronization: Can quadrocopters move in the rhythm of the music? How well can they perform a rhythmic motion?
6. Rhythmic Performances: What has been accomplished to date?

The above questions are driven by the goal of creating a rhythmic flight performance. The answers to these questions are obtained from control theoretic analysis and design.

It is also interesting to make the connection to Chap. 5 here, where similar questions are considered for a different system, namely robotic marionettes, and tools

from controls are used to address the issue of feasibility, motion planning, and timing. An opposite approach is taken in Chap. 7, where music is generated from motion, where synchronization of motion and music plays an equally important role.

4.2 Quadrocopter Dynamics: How do Quadrocopters Move?

> *Dance is the language of movement. It is the realization of the body's potential as an instrument of expression.*
> —Victorian Board of Studies Dance study design, 1994

Human dance expression is fundamentally tied to the human body and its physical capabilities. As an "instrument of expression", the human body seems to enable an endless range of different movements and different movement qualities. Just imagine how many poses there are for a human (without even considering movement): we can stand with two feet on the ground and various hand, arm, finger, and head positions, and can make an almost infinite number of facial expressions. Moreover, skilled dancers can stand still on just one leg... The number of degrees of freedom of a human body (that is, the number of independent joints and possible directions of rotation in those joints) is large but nevertheless motions are constrained by the limits of arms, legs, torso, and head.

In comparison, for a quadrocopter (see Fig. 4.5) there is only one position that allows it to stand still; namely, being horizontal in the air and producing an upward force with its propellers that is equivalent to the gravitational force acting on the vehicle. Moreover, a quadrocopter has only six degrees of freedom: three translational (its three-dimensional position) and three rotational (its attitude), see Fig. 4.5b. However, with only four independent motors (Fig. 4.5a), quadrocopters are underactuated; that is, rotational and translational motion cannot be controlled independently but are coupled [43]. More insight into the coupling will be provided below, where we derive a model for the quadrocopter dynamics from first principles and also specify the constraints of the vehicle. The dynamics model and constraints define the dynamic capabilities of the vehicle in mathematical terms. We provide an interpretation of the findings with respect to our goal of generating rhythmic flight performances.

4.2.1 Dynamics Model of the Quadrocopter

The quadrocopter is described by six degrees of freedom: the translational position $s = (x, y, andz)$ measured in the inertial coordinate system **O** and the rotational position (also called 'attitude') represented by the rotation matrix $\mathbf{R}(t)$ from the body frame **V** to the inertial frame **O** as shown in Fig. 4.5b.

The translational acceleration of the vehicle is dictated by the attitude of the vehicle and the total thrust produced by the four propellers. The translational motion

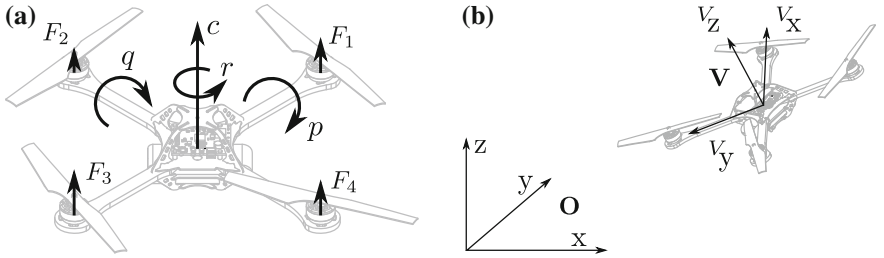


Fig. 4.5 **a** Control inputs of the quadrocopter. **b** Quadrotor position and attitude. Schematic of the quadrocopter with: (**a**) the control signals sent to the vehicle being the body rates p, q, and r and the collective thrust c, and (**b**) the quadrotor position and attitude $\mathbf{V}$ defined with respect to the inertial coordinate system $\mathbf{O}$. These control signals are converted by an onboard controller into motor forces F_i, $i \in \{1, 2, 3, 4\}$

of a quadrocopter in the inertial frame $\mathbf{O}$ is described by

$$\begin{bmatrix} \ddot{x}(t) \\ \ddot{y}(t) \\ \ddot{z}(t) \end{bmatrix} = \mathbf{R}(t) \begin{bmatrix} 0 \\ 0 \\ c(t) \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \Leftrightarrow \begin{array}{l} \ddot{x} = c\, b^x \\ \ddot{y} = c\, b^y \\ \ddot{z} = c\, b^z - g \end{array}, \tag{4.1}$$

where g is the acceleration due to gravity and $c(t)$ is the collective thrust; that is, the sum of the rotor forces F_i normalized by the vehicle mass m,

$$c = \tfrac{1}{m} \textstyle\sum_{i=1}^{4} F_i. \tag{4.2}$$

The motor forces F_i, $i \in \{1, 2, 3, 4\}$, represent the inputs to the quadrocopter (see Fig. 4.5). The values (b^x, b^y, b^z) correspond to the third column of the rotation matrix, namely $(\mathbf{R}_{13}, \mathbf{R}_{23}, \mathbf{R}_{33})$, and represent the direction of the collective thrust in the inertial frame $\mathbf{O}$.

Each rotor produces not only a force F_i, $i \in \mathcal{I} = \{1, 2, 3, 4\}$, in the positive ${}^{V}z$ direction, but also a reaction torque M_i perpendicular to the plane of rotation of the blade, see Fig. 4.5a, where

$$M_i = kF_i, \quad k = \text{const}, \tag{4.3}$$

describes the relationship between the motor force F_i and the associated reaction torque M_i. The parameter k is given by the motor characteristics, see [43] for details. Rotors 1 and 3 rotate in the negative ${}^{V}z$ direction, producing a moment that acts in the positive ${}^{V}z$ direction; while rotors 2 and 4 rotate in the opposite direction resulting in reaction torques in the negative ${}^{V}z$ direction. Given the inertia matrix I with respect to the center of mass and the vehicle frame $\mathbf{V}$, the rotational dynamics of the body-fixed frame are given by

Table 4.1 Quadrocopter parameters

	Definition	Value
m	Mass of vehicle	0.468 kg
L	Vehicle arm length	0.17 m
I_x	Inertia around vehicle ${}^{V}x$-axis	0.0023 kg m^2
I_y	Inertia around vehicle ${}^{V}y$-axis	0.0023 kg m^2
I_z	Inertia around vehicle ${}^{V}z$-axis	0.0046 kg m^2
k	Motor constant	0.016 m
$F_{\min}$	Minimum rotor force	0.08 kg m/s^2
$F_{\max}$	Maximum rotor force	2.8 kg m/s^2

$$I\dot{\Omega} = \begin{bmatrix} L(F_2 - F_4) \\ L(F_3 - F_1) \\ k(F_1 - F_2 + F_3 - F_4) \end{bmatrix} - \Omega \times I\Omega, \tag{4.4}$$

where $\Omega = (p, q, r)$ represent the quadrocopter angular body velocities around the body $({}^{V}x, {}^{V}y, {}^{V}z)$ axes and L is the distance from each motor to the center of the quadrocopter. The vehicle's principal axes coincide with the vehicle frame axes resulting in a diagonal inertia matrix with entries (I_x, I_y, I_z), where $I_x = I_y$ because of symmetry.

The rotation matrix $\mathbf{R}$ evolves according to (cf. [44])

$$\dot{\mathbf{R}}(t) = \mathbf{R}(t) \begin{bmatrix} 0 & -r(t) & q(t) \\ r(t) & 0 & -p(t) \\ -q(t) & p(t) & 0 \end{bmatrix}, \tag{4.5}$$

In our setup, an onboard controller closes the loop on the angular body velocities Ω using onboard gyroscope measurements. As a result, the control signals sent to the the quadrocopter are the collective thrust command c_c and the commanded angular body velocities $\Omega_c = (p_c, q_c, r_c)$, see Figs. 4.2 and 4.9. Based on the commanded values (Ω_c, c_c) and the gyroscope measurements, the onboard controller calculates the required motor forces F_i, $i \in \mathcal{I}$.

The specific vehicle parameters for the quadrocopters used in this work (see Fig. 4.1) are given in Table 4.1.

4.2.2 Vehicle Constraints

The agility of the quadrocopter is constrained by the minimum and maximum force of a single motor, $F_{\min} \leq F_i \leq F_{\max}$, $i \in \{1, 2, 3, 4\}$, with $F_{\min} > 0$, since the motors cannot reverse their direction. The collective thrust is bounded by

$$c_{\min} \leq c \leq c_{\max} \quad \text{with} \quad c_{\min} = 4\,F_{\min}/m, \; c_{\max} = 4\,F_{\max}/m. \tag{4.6}$$

In addition, due to the motor dynamics the rate of change of the motor forces is bounded in reality and the turn rates must be bounded because of the limited measurement range of the gyroscopes used for onboard vehicle control. We neglect both limitations in the following sections to simplify the presentation. The bounds of the thrust rate $\dot{F}_i$ and the turn rates Ω are high (23.9 kg m/s^3 and 25 rad/s, respectively) and do not significantly affect the results in Sect. 4.4.

4.2.3 Implications for a Rhythmic Flight Performance

The above equations describe the motion capabilities of a quadrotor vehicle. From (4.1) we see that the vehicle acceleration is always perpendicular to the plane of the rotors; that is, for a motion in the x, y-direction the quadrocopter must tilt. The translational and rotational degrees of freedom are, therefore, coupled and cannot be specified independently. A rotation of the quadrocopter is achieved by sending appropriate turn rates Ω_c, see (4.5). The rotational dynamics around the ${}^V x$- and ${}^V y$-axes are symmetric, see (4.4), and fast due to the low rotational inertia terms (Table 4.1).

One set of independent motion parameters for a quadrocopter is its *three-dimensional position over time* and the *evolution of the heading angle*, cf. [43] and Fig. 4.7. Compared to the human body, the "body's potential" of a quadrocopter for expressive movements is therefore limited to the position and heading in space over time. Finding motion patterns that are convincingly expressive to the human eye is not trivial and is discussed in Sect. 4.3.

4.3 Motion Design: What is a Dance Step for a Quadrocopter?

Dance is a poem of which each movement is a word —Mata Hari, dancer

As human dance choreography is typically described by sequences of basic movements, we expect a flight performance of quadrocopters to be composed of basic motion elements that—when combined into sequences—allow for a multifaceted, meaningful quadrocopter choreography. As a first step, our goal is to develop basic, rhythmic motion elements that can be executed by quadrocopters and timed to the music beat. These basic rhythmic flight motions represent the "words" that may later tell a "poem".

Periodic motions are a natural human response to hearing a recurring music beat: we often clap, sway, or tap our feet when we hear music. In our research, we want the flying vehicle to mimic this behavior. Periodic motion elements thus represent the basic building blocks of our choreography. As highlighted above, the degrees of freedom of a quadrocopter motion are restricted to the three-dimensional position and its heading. We therefore develop motion elements that show a periodicity in the

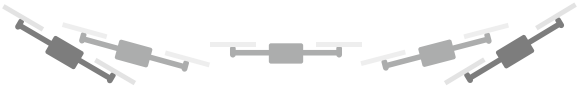


Fig. 4.6 A periodic side-to-side motion with the music beats occurring at the outermost points of the movement

vehicle position and use the vehicle's agility to achieve temporal variety. Below we present a parameterized motion description that enables motion variations that are indispensable when aiming for an expressive choreography.

4.3.1 Music Analysis

The more you understand the music, the easier you can dance.
—Orlando Gutinez

Fundamental to our goal of creating rhythmic flight movements is a tight connection of motion and music. We therefore analyze the music first and then assign appropriate motions to the vehicle. The goal of the music analysis is to extract music features and their respective time signatures. The result is a vocabulary that describes the song's temporal development. We use this time information to assign suitable quadrocopter motions to different sections of the song.

In order to achieve rhythmic behavior, we are particularly interested in the music beat, which represents the basic rhythmic unit of a song and plays a prominent role in the motion design. Currently we use the *BeatRoot* software tool [45] to extract beat times from a song. We store music beats and their respective start times in a text file. This information is then used to create matching flight trajectories; for example, movements that reflect the music tempo.

A simple example that highlights the key idea is shown in Fig. 4.6: the quadrocopter performs a planar side-to-side motion where, at beat times, the vehicle reaches the outermost points of the motion, either on the left or right.

4.3.2 Periodic Motions

When you dance, your purpose is not to get to a certain place on the floor. It's to enjoy each step along the way. —Wayne Dyer. author

We specify basic, rhythmic motion elements as the *evolution* of the quadrocopter's translational position in three dimensions $s_d(t) = (x_d(t),\ y_d(t),\ z_d(t))$ and its heading $\psi_d(t)$ over time. We introduce parameterized motion primitives

$$s_d(p,t), \quad \psi_d(p,t), \tag{4.7}$$

which depend on a set of adjustable motion parameters p and are defined over a finite time interval $t \in [t_0, t_f] \subset \mathbb{R},\ t_f < \infty$. Parameterized motion primitives allow for variety and expressiveness in the choreography design. Consider a horizontal circle, for example, where the radius, speed of rotation, and center point can be adapted depending on the use case. Note that the vehicle heading ψ_d can be designed independently of the position and is not explicitly considered in the following.

Our objective is to offer a similar range of motions as is used in human dance composition. In this context, we ask: Which choices does a professional dance choreographer have when creating a performance? How can we provide the tools and degrees of freedom necessary for implementing an expressive performance on the quadrocopter?

Four fundamental choreographic elements—time, space, energy, and structure—are commonly used by professional dancers, choreographers, and dance teachers to build choreography with interest, dynamics, and estethic appeal, cf. [46, 47]. These parameters provide a framework for meaningful quadrocopter choreography, and are described as follows:

Space Space refers to the area the dancer is performing in. It also relates to how the dancer moves through the area, as characterized by the direction and path of a movement, as well as its size, level, and shape.

Time Time encompasses rhythm, tempo, duration, and phrasing of movements. Using time in different combinations can create intricate visual effects. Ideas such as quick-quick, slow, or stop movements are examples.

Energy Energy relates to the quality of movement. This concept is recognizable when comparing ballet and tap dance. Some types of choreography are soft and smooth, while others are sharp and energetic.

Structure Structure represents the organization of movement sequences into larger concepts: the combination and variation of movements using recurring elements, contrast, and repetition. Movements can even follow a specific story line to convey certain information through a dance.

Examples illustrating the four elements of dance are found in [46, 47].

One way of introducing parameterized, rhythmic motion primitives that capture a wide range of different movements is as a Fourier series [48],

$$s_d(t) = a_0 + \sum_{k=1}^{N} a_k \cos(k\,\omega_d t) + b_k \sin(k\,\omega_d t)\,, \tag{4.8}$$

where $\omega_d = 2\pi/T$ represents the fundamental frequency corresponding to a music beat of frequency $1/T$, where beats are T seconds apart. Additional design parameters are the constant vectors $a_0, a_k, b_k \in \mathbb{R}^3,\ k \in \mathcal{K} = \{1, 2, \ldots, N\}$, and $N \geq 1$; that is, $p = \{\omega_d, N, a_0, a_k, b_k \,|\, k \in \mathcal{K}\}$. The parameters characterize the desired translational position $s_d(t)$ of the quadrocopter and allow us to express the key choreographic elements:

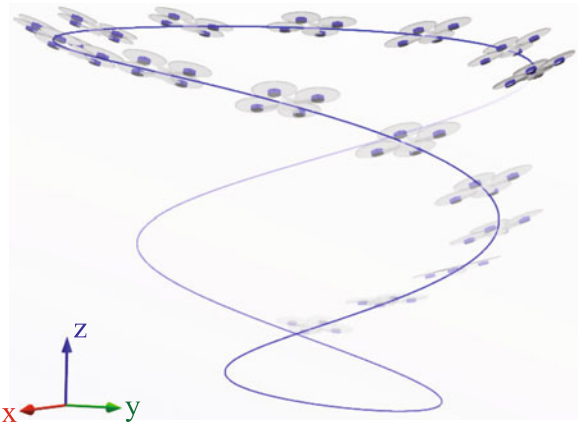


Fig. 4.7 An example of a periodic motion primitive studied in this chapter

Space The parameters a_0 and $a_k, b_k,\ k \in \mathcal{K}$ define the amplitudes of the periodic motion and, thus, the spacial dimension of the movement. These vectors also specify the direction of the motion and the overall three-dimensional shape of the curve.

Time The underlying rhythm is given by the frequency ω_d. When the choreography is set to music, the frequency ω_d is related to the music's tempo. Different tempos can be combined when choosing $N > 1$. The overall duration of the motion can be adjusted via t_f.

Energy The higher the value of N, the more energetic and sharp are the possible motions, cf. [48].

Structure The motion primitives described in (4.8) can be combined into sequences, which can in turn be combined to create an overall choreographic performance. Endless permutations are possible, much the way individual words can be combined into a variety of sophisticated stories, or a series of gestures can be combined to reveal a performer's mood or emotion to an audience.

In short, the general motion description (4.8) reflects the fundamental choreographic elements and allows for a multidimensional choreography. Out of the variety of motions captured by (4.8), Fig. 4.7 illustrates the one with $N = 3$, $T = 10$, $a_0 = (0, 0, 3)$, $a_1 = (0, 0, 1)$, $a_2 = (1, 0, 0)$, and $b_3 = (0, 1, 0)$, and a_3, b_1, b_2 being zero. A Matlab file for generating arbitrary motion primitives of the proposed type is available online at www.idsc.ethz.ch/Downloads/QuadDance.

In order to make (4.7) and (4.8) a useful tool for choreographers, we need to specify which motion primitives can be realized on the vehicle. The dynamics and physical limits of the quadrocopter define the feasible sets of parameters p. This is done in the next section.

4.4 Motion Feasibility: What are the Physical Limits of a Quadrocopter?

The dancer is restricted by self-limits, the limits of being in this body with these abilities and not others.

—Sondra Horton Fraleigh in "Dance and the Lived Body: A Descriptive Aesthetics" [49]

Though dancers and athletes are trained to push the physical limits of their bodies to extremes, they nonetheless remain constrained by the rules of physics. Quadrocopters, too, are limited by their body's dynamics (Sect. 4.2). For example, our quadrocopters cannot keep a constant height when flying sideways with angles larger than 66° (cf. Fig. 4.6). To create a "choreography" for quadrocopters, we must be aware of and account for these physical limitations.

Below we describe a method for checking the feasibility of quadrocopter motions. The approach, meant as a validation tool for preprogrammed quadrocopter performances, is based on the first principles models in Sect. 4.2 and ensures that a desired trajectory respects both vehicle dynamics and motor thrust limits. The goal is to determine sets of motion parameters p, cf. (4.7), (4.8), that represent rhythmic motions that can be realized with a quadrocopter. The result of this analysis is a library of feasible motion primitives that can be used to create multifaceted performances.

4.4.1 Motor Thrust Limits

For the subsequent feasibility analysis, we assume that motion primitives, cf. (4.7), are twice-differentiable in time. This assumption is satisfied for the periodic motions primitives (4.8) introduced in the previous section. Feasibility is formulated in terms of the collective thrust limits $(c_{\min}, c_{\max})$ and the motion parameters p. The objective is to derive a set of inequalities that specify feasible parameter sets p given the limits $(c_{\min}, c_{\max})$.

For a desired motion primitive s_d, we rewrite (4.1),

$$\mathbf{R}\, n\, c_d = \ddot{s}_d + n\, g, \tag{4.9}$$

where $n = (0, 0, 1)$ and c_d is the nominal thrust input required to achieve s_d. Taking the 2-norm, we can solve for c_d, $c_d \geq 0$,

$$\|\mathbf{R}\, n\, c_d\| = \|\ddot{s}_d + n\, g\| \quad \Leftrightarrow \quad c_d = \|\ddot{s}_d + n\, g\| \ . \tag{4.10}$$

Recalling that $s_d = s_d(p, t)$ and (4.6), the inequalities guaranteeing the maximum and minimum bounds of the collective thrust are

$$c_{\min} \leq \|\ddot{s}_d(p, t) + n\, g\| \leq c_{\max}, \quad t \in \left[t_0, t_f\right]. \tag{4.11}$$

This feasibility requirement can be checked for any given desired motion primitive $s_d(p, t)$ by calculating its second time derivative. No further calculations are necessary. In particular, the nominal quadrocopter inputs associated with $s_d(p, t)$ need not be determined in advance. The inequalities (4.11) exclude the majority of infeasible parameters p and help to build an intuition as to what is feasible for a quadrotor vehicle.

In order to be more precise, single motor constraints and turn rate constraints must be considered, cf. Sect. 4.2.2. For those constraints explicit parameter-dependent inequalities are generally difficult to derive (see [3] for details). Instead, in our current software framework, we numerically assess the feasibility of a created motion sequence before actual flight, see [50].

4.4.2 Example: Side-to-Side Motion

To demonstrate the above feasibility test, we consider a simple periodic motion that falls into the framework introduced in (4.8): a horizontal side-to-side motion as illustrated in Fig. 4.6. In fact, the side-to-side motion was the first rhythmic motion that we implemented on a quadrocopter and executed to music [1].

The planar side-to-side movement is given by

$$x_d(t) = A\cos(\omega_d t), \quad y_d(t) = z_d(t) = \psi_d = 0. \tag{4.12}$$

The side-to-side motion is a special case of the general motion primitive description (4.8), where $N = 1$, $a_1 = (A, 0, 0)$ and $a_0, b_1 = (0, 0, 0)$.

To determine feasible combinations of amplitudes A and frequencies ω_d, we calculate the second derivative of (4.12) and insert it into (4.11):

$$c_{\min} \leq \sqrt{A^2\omega_d^4\cos^2\omega_d t + g^2} \leq c_{\max}. \tag{4.13}$$

For a given pair (A, ω_d), these inequalities must be satisfied for all $t \in [0, T]$. Therefore, it is enough to consider the maximum and minimum values over T. We obtain

$$A\omega_d^2 \leq \sqrt{c_{\max}^2 - g^2} \quad \text{and} \quad c_{\min} \leq g. \tag{4.14}$$

The second inequality must be satisfied in order for a quadrocopter to be able to land. In brief, all parameter pairs (A, ω_d) satisfying the inequality (4.14) represent side-to-side motions that stay within the collective thrust limits (4.6).

For the vehicle parameters in Table 4.1, Fig. 4.8 illustrates the feasible set of side-to-side trajectories (A, ω_d). The dark gray region contains parameter sets that are infeasible due to the collective thrust limit, cf. (4.14). We also depict (light gray area) the parameter sets that become infeasible when taking into account the minimum and maximum force limits of each single motor (see Sect. 4.2.2); the corresponding

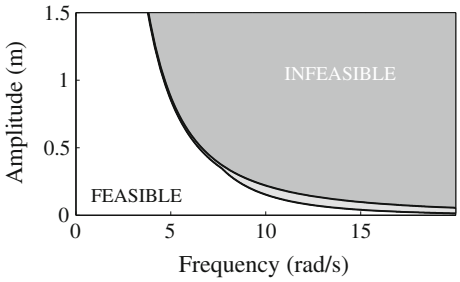


Fig. 4.8 Feasible parameter sets for the side-to-side motion primitive. The *dark gray* region denotes parameter sets that are infeasible due to collective thrust limits; *light gray* denotes additional parameter sets that are infeasible due to the minimum and maximum force limits of each single motor

derivations are presented in [3]. From Fig. 4.8 we see that if we want to perform two side-to-side motions per second ($\omega_d \approx 12.6$ rad/s), a motion amplitude of 0.5 m is clearly infeasible. We also see that for the side-to-side motion the single motor force limits exclude only a small additional number of parameter sets. The inequalities (4.11) represent a simple means to understand which motions are feasible.

4.5 Quadrocopter Control: How do Quadrocopters Execute Their Movements?

Technique—bodily control—must be mastered only because the body must not stand in the way of the soul's expression.
—La Meri, dancer and choreographer

In Sect. 4.3 we introduced rhythmic motion elements with the goal of enabling expressive choreography, where the movements were defined by the desired evolution of the quadrocopter position over time. However, similarly to human dancers who constantly work on perfecting their body control, quadrocopters require sophisticated control algorithms to guide their "bodies" along the desired trajectories. Just recall that the smallest mistake may lead to the vehicle falling out of the sky. In this section, we derive a motion controller that maintains the quadrocopter on the specified trajectory during actual flight.

4.5.1 Trajectory-Following Controller

The trajectory-following controller (TFC) accepts as input commanded positions, velocities, and accelerations, as well as, a yaw angle trajectory (cf. Fig. 4.2):

$$(s_c(t), \dot{s}_c(t), \ddot{s}_c(t), \psi_c(t)) \,. \tag{4.15}$$

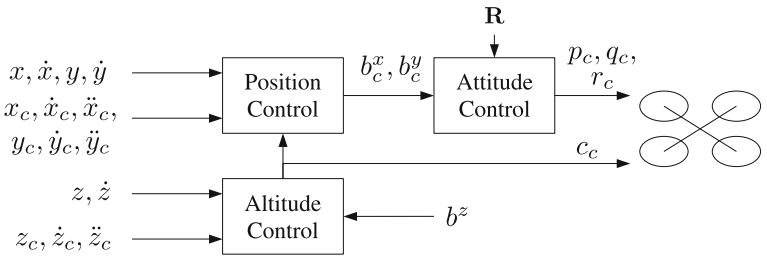


Fig. 4.9 Cascaded control loops of the trajectory-following controller (TFC)

We usually obtain appropriate input commands directly from the desired sequence of motion primitives (see Sects. 4.3 and (4.7)) by setting: $s_c(t) := s_d(t)$ and $\psi_c(t) := \psi_d(t)$. The respective time derivatives $\dot{s}_c(t), \ddot{s}_c(t)$ are also computed from the desired trajectory $s_d(t)$, which is preplanned and known in its full length prior to flight.

The TFC is a standard component of our experimental testbed. Control is based on precise measurements of the vehicle position and attitude (in our case, provided by a motion capture system). The TFC receives the quadrocopter's position $s = (x, y, z)$, velocity $\dot{s}$ and attitude $\mathbf{R}$ from an estimator and, in turn outputs the body rate and collective thrust commands (Ω_c, c_c) to the vehicle, see Fig. 4.9. The TFC consists of three separate loops for altitude, horizontal position, and attitude. While the TFC operates in discrete time, the controller design is based on the continuous-time system dynamics representation.

The altitude control is designed such that it responds to altitude errors $(z - z_c)$ like a second-order system with time constant τ_z and damping ratio ζ_z,

$$\ddot{z} = -\frac{2\zeta_z}{\tau_z}(\dot{z} - \dot{z}_c) - \frac{1}{\tau_z^2}(z - z_c) + \ddot{z}_c. \tag{4.16}$$

It uses the collective thrust to achieve this. With (4.1) and (4.16), we obtain

$$c_c = (\ddot{z} + g)/b^z. \tag{4.17}$$

Similarly, the two horizontal position control loops are shaped based on (4.1) with c_c from (4.17). Commanded rotation matrix entries b_c^x, b_c^y result. The attitude control is shaped such that the two rotation matrix entries b^x, b^y react in the manner of a first-order system; that is, for x: $\dot{b}_c^x = (b^x - b_c^x)/\tau_{RP}$. This is directly mapped to the commanded angular body velocities (p_c, q_c) using (4.5) and the estimated attitude $\mathbf{R}$,

$$\begin{bmatrix} p_c \\ q_c \end{bmatrix} = \frac{1}{\mathbf{R}_{33}} \begin{bmatrix} \mathbf{R}_{21} & -\mathbf{R}_{11} \\ \mathbf{R}_{22} & -\mathbf{R}_{12} \end{bmatrix} \begin{bmatrix} \dot{b}_c^x \\ \dot{b}_c^y \end{bmatrix}. \tag{4.18}$$

Vehicle yaw control can be considered separately, since rotations around the body ${}^V z$-axis do not affect the above dynamics. The yaw controller is a proportional controller and the resulting yaw angle rate is mapped to r_c using the kinematic relations of

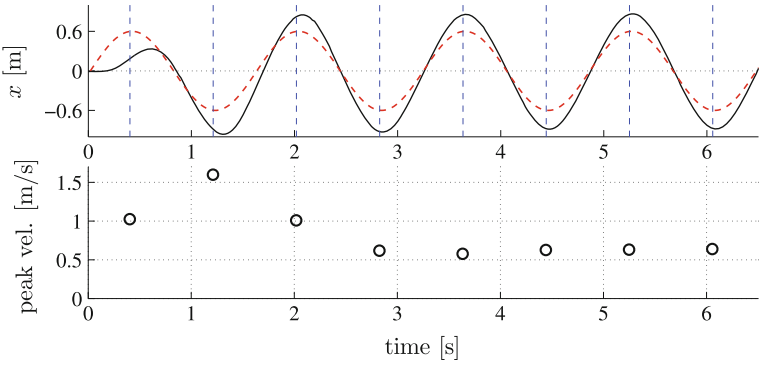


Fig. 4.10 Side-to-side motion, *no motion parameter adaptation*. *Top* quadrocopter response (*solid*) for a desired oscillation in the x-direction (*dashed*). *Bottom* corresponding peak velocities, i.e., absolute value of vehicle velocity at the peaks of the desired trajectory. High peak velocities imply a large phase error

Euler angles. The innermost loop, on board the quadrocopter, controls the angle rates (p, q, r) to the calculated set points (p_c, q_c, r_c).

In the ideal case, where the quadrocopter dynamics correspond to the model (4.1) and some other mild assumptions are made (see [4] for details), the derived controller yields perfect trajectory tracking. In summary, we have presented a control framework that enables an autonomous quadrocopter flight along a desired trajectory defining the vehicle position and the heading of the vehicle over time.

4.5.2 Tracking Performance of Periodic Motions

When using the derived TFC to track the side-to-side motion (4.12), we considered before with $s_c(t) := s_d(t)$ and $\psi_c(t) := \psi_d(t)$ (that is, the desired periodic trajectory is directly sent to the vehicle controller), we observe, at steady state, a sinusoidal motion of the same frequency with a constant error in amplitude and phase, resulting in asynchrony and spatial inaccuracies, as shown in Fig. 4.10 (top figure). The amplitude error of the quadrocopter response (black solid line) is obvious; the phase error between the reference trajectory and the actual quadrocopter response is hardly noticeable. However, small phase errors are very visible and audible in actual experiments as humans are especially sensitive to nonzero vehicle velocity at beat times (see [5] for more details). Correspondingly, the bottom plot of Fig. 4.10 illustrates the velocity of the quadrocopter at beat times; that is, when the reference trajectory reaches its maximum or minimum value.

For periodic motions in three dimensions, a similar behavior is observed: phase shift and amplitude error are observed in each translational direction and are not necessarily equal in size. In this case, the shape of the resulting motion can change.

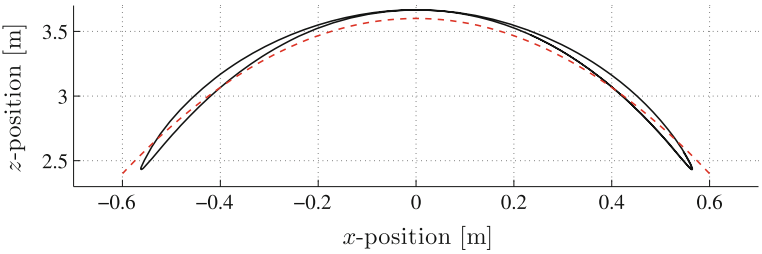


Fig. 4.11 Vertical bounce motion, *no motion parameter adaptation*. The vehicle's response (*solid*) can differ in shape from the desired trajectory (*dashed*)

For example, a desired bounce motion (Fig. 4.11) results in a bent eight-shaped vehicle motion.

In order to achieve precise temporal and spatial tracking, we adapt the parameters of the commanded trajectory (4.15) sent to the TFC in the next section. Later we see that these parameters can be identified/learned prior to the flight performance in order to effectively reduce initial transients.

4.6 Motion Synchronization: Can a Quadrocopter Move in the Rhythm of the Music?

I like to see you move with the rhythm; I like to see when you're dancing from within.
—Bob Marley, singer and composer

"Moving with the rhythm" is the ultimate goal of this work, where we aim to control the motion of quadrocopters to an external music signal. As highlighted in the previous section, pure feedback controlled to insufficient quadrocopter tracking with a noticeable phase and amplitude error.

The goal of this section is to prove the feasibility of a precise synchronization between quadrocopter motion and music, where we use the term "synchronization" loosely, inasmuch as it encompasses both spatial and temporal tracking accuracy. Our strategy for coping with the aforementioned constant phase shift and amplitude error is to adjust the motion parameters of the trajectory commanded to the underlying trajectory-following controller (see Fig. 4.2). This means, for example, that if the amplitude of the quadrocopter motion is larger than the desired one, we reduce the commanded amplitude. Similarly, if the vehicle motion is lagging, we shift the commanded trajectory by increasing the phase.

4.6.1 Synchronization: The Basic Idea

To illustrate the basic idea of the "feed-forward" strategy, we consider the side-to-side motion in (4.12) and Fig. 4.6, where we adapt the commanded amplitude A_c and phase θ_c of the commanded trajectory $s_c(t)$ (Fig. 4.2),

$$x_c(t) = A_c \cos(\omega_d t + \theta_c), \quad y_c(t) = z_c(t) = \psi_c = 0. \tag{4.19}$$

to achieve synchronization. Our original results on this topic were presented in [1].

4.6.1.1 Online Correction

The motion parameters of the commanded trajectory are set to

$$\theta_c(t) = \theta_{\text{on}}(t), \quad A_c(t) = A_d + A_{\text{on}}(t), \tag{4.20}$$

where the subscript "on" indicates the online correction terms. They are updated in real time, during the flight.

As illustrated in Sect. 4.5.2, the response of the controlled quadrocopter system to a side-to-side reference signal (4.12) when choosing $s_c(t) := s_d(t)$ and $\psi_c(t) := \psi_d(t)$ is a sinusoidal signal with the same frequency but shifted phase and different amplitude,

$$x(t) = (A(t) + A_d)\ \cos\left(\omega_d t + \theta(t)\right). \tag{4.21}$$

To determine the additive errors in amplitude $A(t)$ and phase $\theta(t)$, the two reference signals, $r_{\cos}(t) = \cos\omega_d t$ and $r_{\sin}(t) = \sin\omega_d t$, are multiplied by the position estimate $x(t)$ and integrated over N periods, that is $T = (2\pi N)/\omega_d$. Assuming a constant phase shift and an amplitude error during that time interval

$$\theta(v) = \theta_t = \text{constant}, \quad A(v) = A_t = \text{constant}, \qquad t - T \le v \le t, \tag{4.22}$$

we obtain

$$\begin{aligned} \eta_1(t) &= \frac{1}{T}\int_{t-T}^{t} x(t) r_{\cos}(t)dt = \frac{A_t + A_d}{2}\cos(\theta_t), \\ \eta_2(t) &= \frac{1}{T}\int_{t-T}^{t} x(t) r_{\sin}(t)dt = -\frac{A_t + A_d}{2}\sin(\theta_t), \end{aligned} \tag{4.23}$$

and

$$\begin{aligned} A_t &= 2\sqrt{\eta_1(t)^2 + \eta_2(t)^2} - A_d, \\ \theta_t &= -\arctan\left(\eta_2(t)/\eta_1(t)\right). \end{aligned} \tag{4.24}$$

The values θ_t, A_t can be interpreted as the mean value of the phase and amplitude errors during the last period, and when considering Fig. 4.10, the phase and

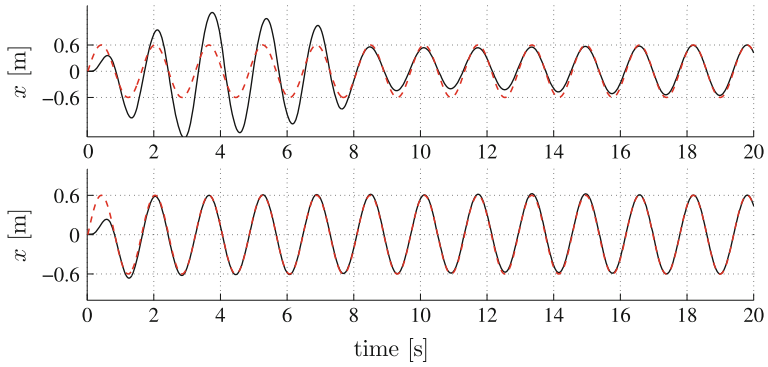


Fig. 4.12 Side-to-side motion. *Top online motion parameter adaptation only.* Quadrocopter response (*solid*) for a desired oscillation in the x direction (*dashed*) *Bottom offline motion parameter adaptation,* with online motion parameter adaptation turned on after two periods

amplitude errors are in fact constant (after a transient phase). Therefore, (4.22) is a valid assumption in steady state.

The online correction terms are calculated by integrating the errors according to

$$A_{\text{on}}(t) = k_A \int_0^t A_\tau \, d\tau, \quad \theta_{\text{on}}(t) = k_\theta \int_0^t \theta_\tau \, d\tau, \tag{4.25}$$

where the gains k_θ, k_A are chosen to ensure convergence of the online correction terms to the steady-state values $\theta_{\text{on},\infty}$ and $A_{\text{on},\infty}$, respectively.

Using the proposed online parameter adaptation strategy (4.20), (4.25), the errors in amplitude and phase are effectively regulated to zero, see Fig. 4.12 (top figure) and compare to Fig. 4.10. We observe a substantial transient phase before the online correction terms attain steady state, see Fig. 4.13. This is mainly due to the fact that the error identification scheme (4.23), (4.24) only provides reliable values after several periods.

4.6.1.2 Offline Identification

The steady-state values $\theta_{\text{on},\infty}$, $A_{\text{on},\infty}$ obtained from the online correction are repeatable (that is, different runs of the same experiment produce the same result). Consequently, the correction values can be extracted once, and later applied to improve the transient performance; that is, the tracking during the initial period of a motion. For the phase, we use

$$\theta_c(t) = \theta_{\text{off}} + \theta_{\text{on}}(t) \quad \text{with} \quad \theta_{\text{off}} = \theta_{\text{on},\infty}. \tag{4.26}$$

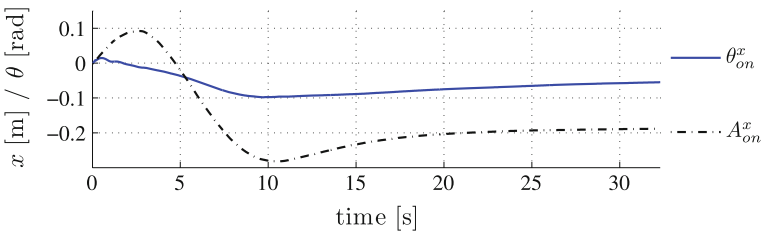


Fig. 4.13 Side-to-side motion, convergence of online correction terms

The equation for the amplitude is similar. The subscript 'off' indicates the offline motion parameters identified prior to the experiment. Figure 4.12 (bottom figure) shows the corresponding result for the side-to-side motion. The transient time is substantially decreased.

4.6.1.3 Reduced Offline Identification

Thus far, offline parameters must be identified for each side-to-side motion (A_d, ω_d) individually. To draw further conclusions, we consider the steady-state values in the following form: the amplitude-normalized amplification factor,

$$\alpha_{\text{on},\infty} = (A_d + A_{\text{on},\infty})/A_d \,, \tag{4.27}$$

and the steady-state phase $\theta_{\text{on},\infty}$ as before. Experiments in [1] have shown that the steady-state values $(\alpha_{\text{on},\infty}, \theta_{\text{on},\infty})$ depend only on the motion's frequency ω_d. That is, a single identification run must be completed for each frequency, the vehicle should perform at and the resulting parameters can be used for any side-to-side motion at this frequency with varying amplitudes.

4.6.2 Synchronization in Three Dimensions

We extend the previous results into three-dimensional (3-D) motion, which is composed of sinusoidal side-to-side motions in each translational direction:

$$\begin{bmatrix} x_d(t) \\ y_d(t) \\ z_d(t) \end{bmatrix} = \begin{bmatrix} A^x_d \cos(\omega^x_d t + \theta^x_d) \\ A^y_d \cos(\omega^y_d t + \theta^y_d) \\ A^z_d \cos(\omega^z_d t + \theta^z_d) \end{bmatrix}, \quad \psi_d(t) = 0, \tag{4.28}$$

where $\theta_d^{(x,y,z)}$ represents a potential phase shift between the sinusoidal motions in each direction. Bounces, ellipses, eights, and spirals can be obtained by appropriate parameter choices.

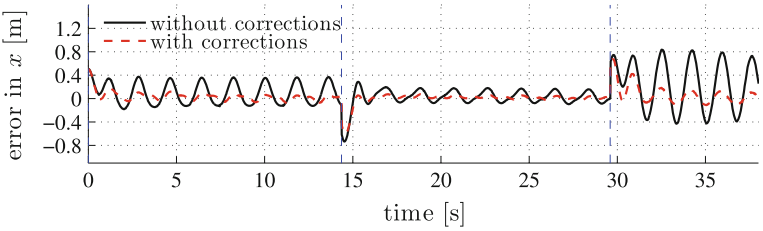


Fig. 4.14 Sequence of motions with (*dashed*) and without (*solid*) feed-forward corrections. Offline correction terms were obtained from a *reduced identification.* The errors of the desired trajectory to the response of the vehicle ($s_d(t) - s(t)$) are plotted. The motion sequence comprises: a circular motion in 3-D, a swing motion in 3-D, and a horizontal circle

As shown in [4], a key assumption can be made for the given 3-D motion: each translational direction can be treated separately. To this end, the motion parameters in the commanded trajectory $s_c(t)$ are adjusted independently for each direction according to the online strategy presented above. In addition, the offline identification benefits from the directional decoupling and the quadrocopter symmetry in that the x- and y- directions exhibit the same behavior and identification in one of the two directions is sufficient.

Consequently, it is possible to develop an identification scheme that efficiently identifies the offline correction terms for all periodic motions that can be expressed in our framework (4.28): *a single* identification run over the relevant frequency range with a 2-D motion primitive in x or y, and z is sufficient to completely identify all necessary feed-forward parameters. The offline values are stored in a look-up table ready to be used when performing new motions of (4.28).

In order to show the effectiveness of the reduced identification scheme, we perform a sequence of periodic 3-D motions with offline parameters obtained from an oscillatory motion in 2-D ($A_d^x = A_d^z = 0.4\,\text{m}$, $\omega_d^x = \omega_d^z = \omega$ and all other parameters zero). Figure 4.14 shows that the quadrocopter's deviation from the desired trajectory is clearly reduced when using the offline parameter adaptation strategy. Note that the performance can be further improved by designing smooth transitions between the motion of the sequence.

To conclude, we studied a feed-forward parameter tuning strategy that improves the tracking performance of periodic motion primitives especially during transients using preidentified correction terms and online parameter adaptation. The translational directions are independent, allowing for the efficient identification of a table stored offline. In brief, we enable quadrocopters to fly to the rhythm of the music with correctly scaled motions. This fulfills the requirements for a rhythmic motion.

4.7 Rhythmic Performances

People in the audience, when they've watched the dance, should feel like they've accomplished something, that they've gone on a journey.

—Paul Mercurio, actor and dancer

Opening night at the theater: patrons of the arts and critics take their seats while dancers do last last-minute warm-ups and take their places backstage. Tension is high: performance success is part practice, part sweat, and part luck. We have already discussed in Sect. 4.6 how quadrocopters can practice and improve their performance over time, but do sweat and luck have a role in a quadrotor performance? In control theory, the term "robustness" refers to the ability of a system to control for uncertainty; that is, for unknown effects such as wind or reduced propeller efficiencies. Given the feedback from sensors (the overhead camera system or onboard sensors), quadrocopters can react to uncertainty quickly and effectively—putting more effort into the motions if propeller efficiencies are low or executing corrective movements if unexpected external disturbances corrupt their motions. And consequently, the resulting quadrocopter performance is mostly predictable and has been demonstrated during several hundred demonstrations to visitors in and outside the lab.

4.7.1 Experimental Testbed

We demonstrate our algorithms on small, custom quadrocopters operated in the ETH Zurich Flying Machine Arena, a $10 \times 10 \times 10\,\mathrm{m}^3$ mobile testbed for quadrocopter research. The setup is similar to [51]: The system consists of a motion capture camera system that provides precise vehicle position and attitude measurements. The localization data is sent to a personal computer, which runs the control algorithms, and which in turn sends commands to the quadrocopters. More details about the test environment can be found in [52] and at: www.FlyingMachineArena.org.

4.7.2 Implementation and Robustness

Based on the rhythmic motion elements discussed in this chapter, full performances are designed for a given soundtrack. Additional motion elements not discussed in this chapter are used to smoothly concatenate the periodic motions (see [42] for more details). Moreover, acrobatic motions such as flips, loops, and bang-bang-type transitions can be incorporated in the performances to add variety; those motions are not strictly related to the music beat. In [5], the choreography design procedure is described from a practical point of view.

The resulting performance is completely preprogrammed. However, to allow for a robust and reliable execution, the preprogrammed feed-forward signals are complemented by several feedback and adaptation schemes.

We use an adaptation scheme for online synchronization of the motion to the music (Sect. 4.6). Residual phase and amplitude errors in the quadrocopter response are compensated for during the performance by adapting the commanded trajectory online (see Fig. 4.2). The online adaptation allows us to synchronize the motion of vehicles with slightly different dynamic properties (e.g., shifted center of mass

and degraded propellers). The underlying trajectory-following controller (Sect. 4.5) compensates for unexpected disturbances such as wind in a reactive manner based on the measured vehicle following errors. The trajectory-following controller in turn relies on the vehicle's onboard controller to quickly compensate for local model uncertainties such as degraded propeller efficiencies by, for example, increasing the turn rate of the propellers to obtain the required thrust in the case of reduced propeller efficiencies.

4.7.3 Choreographies

Since the start of the project, several choreographies have been designed based on the rhythmic motion elements discussed in this chapter. The following list presents the choreographies that are featured in the Flying Machine Arena with the song name, the singer or composer of the song, the number of quadrocopters, and their respective design year:

- *Please don't stop the music*, Rihanna, one vehicles, 2009
- *Pirates of the Caribbean*, Hans Zimmer, two vehicles, 2009
- *Rise Up*, Yves Larock, three vehicles, 2010
- *From the Clouds*, Jack Johnson, four vehicles, 2011
- *Armageddon*, Prism, five vehicles, 2011
- *Dance of the Flying Machines*, Victor Hugo Fumagalli, six vehicles, 2013.

These choreographies have not only been regularly demonstrated at ETH Zurich, where we conduct our research, but also at exhibitions such as the Hannover Messe (April 2012, Fig. 4.15), Google I/O (June 2012), and TEDGlobal (June 2013). Figure 4.16 shows the vehicle flight trajectories that compose the first part of the *From the Clouds* performance. Associated videos are found on the project web page, www.tiny.cc/MusicInMotionSite.

4.8 Conclusions and Outlook

> *Humans do not communicate by words alone. Non-verbal behavior, including dance, is a part of the calculus of meaning.*
> —Judith Lynne Hanna in "To Dance is Human: A Theory of Nonverbal Communication" [53]

The evolution of robotics into human-centered applications poses important research questions, especially with respect to human–machine interaction. As long as robots remained the domain of industry, precision, speed, and repeatability were of primary importance; however, as robots increasingly enter our homes, offices, and communities, there is a corresponding need for them to be able to correctly interpret and appropriately respond to human action and behavior. In this chapter, we presented a novel visual musical experience: multiple flying vehicles coordinate their flight to the

Fig. 4.15 The ETH Zurich Flying Machine Arena at the Hannover Messe in Germany, the world's biggest industrial fair, April 2012 (*Photo* Markus Hehn)

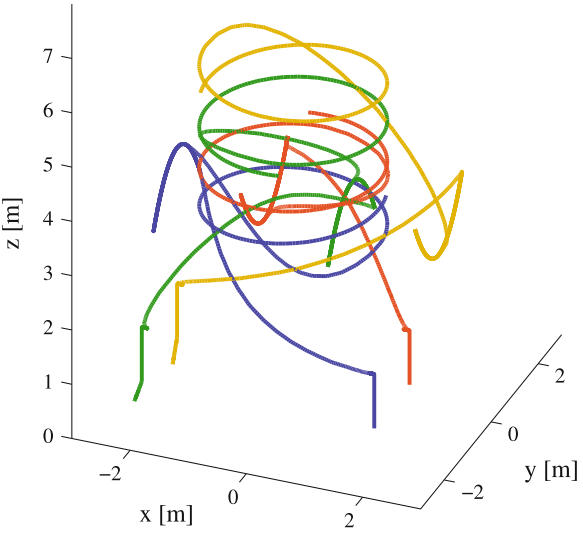


Fig. 4.16 Experimental flight data from the *From the Clouds* performance featuring four quadrocopters

rhythm of the music and perform an aerial show; a cubic indoor flight space forms the stage, and small autonomous quadrocopters are the actors of this performance. While this vision was the motivation for our work, in the process of implementing this idea, fundamental problems in trajectory planning and control were solved, including the *a priori* evaluation of a trajectory's feasibility, and a combined offline and online identification/adaptation scheme for precise tracking of periodic motions. All these components have been integrated into a software tool that facilitates the choreography design, and the range of different choreographies and the number of public demonstrations prove the feasibility and reliability of the designed algorithms. And

yet the periodic motion and patterned behaviors presented here, though they may be characteristic of dance, do not in and of themselves constitute it. According to most dancers, choreographers, and audience members, it is the emotional, social, and spiritual aspects of dance that are the more essential characteristics: a robot moving rhythmically to a beat is no more a dancer than a metronome is a musical instrument. In other words, when it comes to interpreting music into a series of motions that is recognizable as dance, what is important is the *human* element.

Critical next steps for this project will thus be to explore how dance can be used as a form of shared experience with which to build an understanding of intuitive human–machine interaction. Chapter 3 explores motion-based communication in human salsa dance. As John Baillieul says, "The ultimate goal is to understand human reaction to gestures and how machines may react to gestures." Rich Barlow describes it as follows,[2] "Good dancers move seamlessly together, responding to each other's touch and motions; amateurs without experience reading each other's cues often come off looking stilted". By investigating the nonverbal cues dance partners use to communicate, the researchers hope to gain insight into new intuitive means of robot–human interaction, which could enable robots to team with, and perhaps take over from, humans in the future. Our work similarly envisions a dance "partnership" through which human–machine interaction can be studied and enhanced; however, our use of quadrocopters poses an additional challenge as there is little shared body experience between a quadrocopter body and the human body. The relationship must be founded on a shared understanding of movement alone.

Machines and humans each have their strengths and weaknesses: machines are better at rule-based, rational tasks—like synchronization and determining feasible motions sequences— whereas humans are better at things that are hard to describe using rules—like conveying and understanding emotion. Performing together as dance partners, humans and quadrocopters have the potential to engage in complimentary ways. Parallel work in our research laboratory already includes human-in-the-loop experimentation with quadrocopter control, and suggests that this kind of human–quadrocopter partnering is feasible: the TED talk www.tiny.cc/TED_DAndrea demonstrates gesture control of quadrocopters using a Kinect system, and shows simple physical interaction between a human and a quadrocopter (see Fig. 4.17 and [54]). By incorporating this research into our aerial dance system, we can enable dancers and choreographers to directly communicate with the quadrocopters using physical interaction, or simply their body language, their gestures. Imagine an experienced dancer/choreographer guiding a suite of quadrocopters as they dash through the air. What kinds of performances would we see then? Could his or her subtle touch not convey all kinds of emotion? Indeed, as described in Sect. 4.1.3, technological props have long been used to augment dance performance. However, the proposed human–quadrocopter dance "partnership" could go beyond extending a

[2] www.bu.edu/today/2013/dances-with-robots

Fig. 4.17 Demonstration of physical interaction between a human and quadrocopters: Raffaello D'Andrea at TEDGlobal, June 2013 (*Photo* James Duncan Davidson)

human's dance performance. It could ultimately help us to better understand how humans and machines can communicate intuitively with each other, and enable new forms of human–machine interaction.

Acknowledgments The authors would like to acknowledge the contributions of the current and former Flying Machine Arena team members, in particular Markus Hehn, Sergei Lupashin, Mark W. Mueller, and Michael Sherback. The authors also thank Marc-Andre Corzillius, Carolina Flores, Hans Ulrich Honegger, and Igor Thommen for the technical support. This research was supported in part by the Swiss National Science Foundation.

References

1. Schoellig AP, Augugliaro F, D'Andrea R (2010) Synchronizing the motion of a quadrocopter to music. In: Proceedings of the IEEE international conference on robotics and automation (ICRA), 2010, pp 3355–3360
2. Schoellig AP, Augugliaro F, D'Andrea R (2010) A platform for dance performances with multiple quadrocopters. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS) - workshop on robots and musical expressions, 2010, pp 1–8
3. Schoellig AP, Hehn M, Lupashin S, D'Andrea R (2011) Feasibility of motion primitives for choreographed quadrocopter flight. In: Proceedings of the American control conference (ACC), 2011, pp 3843–3849
4. Schoellig AP, Wiltsche C, D'Andrea R (2012) Feed-forward parameter identification for precise periodic quadrocopter motions. In: Proceedings of the American control conference (ACC), 2012, pp 4313–4318
5. Augugliaro F, Schoellig AP, D'Andrea R (2013) Dance of the flying machines. IEEE Robotics and Automation Magazine
6. Varela FJ, Thompson ET, Rosch E (1991) Dance of the flying machines: Methods for Designing and Executing an Aerial Dance Choreography. The embodied mind: cognitive science and human experience. The MIT Press, Cambridge
7. Brooks RA (2000) Cambrian intelligence: the early history of the new AI. MIT Press, Cambridge

8. Pfeifer R, Bongard J (2007) How the body shapes the way we think: a new view of intelligence. MIT press, Cambridge
9. Block B, Kissell JL (2001) The dance: essence of embodiment. Theoretical medicine and bioethics 22(1):5–15
10. Gray JA (1989) Dance technology: current applications and future trends. ERIC
11. Gray JA (1984) Dance in computer technology: a survey of applications and capabilities. Interchange 15(4):15–25
12. Kim G, Wang Y, Seo H (2007) Motion control of a dancing character with music. In: Proceedings of the 6th IEEE/ACIS international conference on computer and information science (ICIS), 2007, pp 930–936
13. Kim T-H, Park SI, Shin SY (2003) Rhythmic-motion synthesis based on motion-beat analysis. ACM Trans. Graph. (TOG) 22(3):392–401
14. Bary J, Leaping into dance technology. Connect: information technology at NYU, 2002. http://www.nyu.edu/its/pubs/connect/archives/fall02/bary_dance.pdf
15. Macel E (2007) iDance. Dance Magazine
16. Latulipe C, Wilson D, Huskey S, Word M, Carroll A, Carroll E, Gonzalez B, Singh V, Wirth M, Lottridge D (2010) Exploring the design space in technology-augmented dance. In: Extended abstracts on human factors in computing systems. ACM, New York, pp 2995–3000
17. Meador WS, Rogers TJ, O'Neal K, Kurt E, Cunningham C (2004) Mixing dance realities: collaborative development of live-motion capture in a performing arts environment. Computers in Entertainment (CIE) 2(2):12–12
18. Birringer JH (2002) Dance and media technologies. PAJ J Perform Art 24(1):84–93
19. Lynch A, Majeed B, O'Flynn B, Barton J, Murphy F, Delaney K, O'Mathuna S (2005) A wireless inertial measurement system (WIMS) for an interactive dance environment. J Phys Conf Series 15(1):95
20. Calvert T, Wilke W, Ryman R, Fox I (2005) Applications of computers to dance. IEEE Comput Graph Appl 25(2):6–12
21. Chan JC, Leung H, Tang JK, Komura T (2011) A virtual reality dance training system using motion capture technology. IEEE Trans Learn Technol 4(2):187–195
22. Parrish M (2007) Technology in dance education. In: International handbook of research in arts education. Springer, Dordrecht, pp 1381–1397
23. Smith-Autard J (2003) The essential relationship between pedagogy and technology in enhancing the teaching of dance form. Res Dance Educ 4(2):151–169
24. Obermaier K, Ars Electronica Futurelab Apparition. http://www.exile.at/apparition/project.html
25. Nakazawa A, Nakaoka S, Ikeuchi K, Yokoi K (2002) Imitating human dance motions through motion structure analysis. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS), vol 3, 2002, pp 2539–2544
26. Sousa P, Oliveira JL, Reis LP, Gouyon F (2011) Humanized robot dancing: humanoid motion retargeting based in a metrical representation of human dance styles. Prog Artif Intell 7026:392–406
27. Shinozaki K, Iwatani A, Nakatsu R (2008) Construction and evaluation of a robot dance system. In: Proceedings of the IEEE international symposium on robot and human interactive communication (ROMAN), 2008, pp 366–370
28. Aucouturier J, Ogai Y, Ikegami T (2008) Making a robot dance to music using chaotic itinerancy in a network of fitzhugh-nagumo neurons. Neural information processing, pp 647–656, 2008. http://link.springer.com/chapter/10.1007/978-3-540-69162-4_67
29. Nakaoka S, Kajita S, Yokoi K (2010) Intuitive and flexible user interface for creating whole body motions of biped humanoid robots. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems, 2010, pp 1675–1682
30. Tholley IS, Meng QG, Chung PW (2012) Robot dancing: what makes a dance? Adv Mater Res 403:4901–4909
31. Avrunin E, Hart J, Douglas A, Scassellati B (2011) Effects related to synchrony and repertoire in perceptions of robot dance. In: Proceedings of the 6th international conference on Human-robot, interaction, 2011, pp 93–100

32. Grunberg DK, Batula AM, Schmidt EM, Kim YE (2012) Affective gesturing with music mood recognition. In: Proceedings of the 12th IEEE-RAS international conference on humanoid robots, 2012, pp 343–348
33. Ekman P (1992) Are there basic emotions? Psychol Rev 99:550–553
34. Xia G, Dannenberg R, Tay J, Veloso M (2012) Autonomous robot dancing driven by beats and emotions of music. In: Proceedings of the 11th international conference on autonomous agents and multiagent systems-volume 1. International foundation for autonomous agents and multiagent systems, 2012, pp 205–212
35. Meng Q, Tholley I, Chung PW (2012) Robot dancing: adapting robot dance to human preferences. pp. 557–565, 2012
36. Takeda T, Hirata Y, Kosuge K (2007) Dance step estimation method based on hmm for dance partner robot. IEEE Trans Ind Electron 54(2):699–706
37. Kosuge K, Takeda T, Hirata Y, Endo M, Nomura M, Sakai K, Koizumi M, Oconogi T (2008) Partner ballroom dance robot -PBDR-. SICE J Control Measur Syst Integr 1(1):74–80
38. Michalowski MP, Simmons R, Kozima H (2009) Rhythmic attention in child-robot dance play. In: Proceedings of the 18th IEEE international symposium on robot and human interactive, communication, 2009, pp 816–821
39. Baillieul J, Ozcimder K (2012) The control theory of motion-based communication: Problems in teaching robots to dance. In: Proceedings of the American control conference (ACC), 2012, pp 4319–4326
40. Murphy R, Shell D, Guerin A, Duncan B, Fine B, Pratt K, Zourntos T (2011) A midsummer night's dream (with flying robots). Auton Robot 30(2):143–156
41. Catton P (2011) Culture city online: dances with robots. The Wall Street Journal - Metropolis Blog. http://blogs.wsj.com/metropolis/2011/07/15/culture-city-online-dances-with-robots/
42. Augugliaro F, Schoellig AP, D'Andrea R (2012) Generation of collision-free trajectories for a quadrocopter fleet: a sequential convex programming approach. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS), 2012, pp 1917–1922
43. Mellinger D, Kumar V (2011) Minimum snap trajectory generation and control for quadrotors. In: Proceedings of the IEEE international conference on robotics and automation (ICRA), 2011, pp 2520–2525
44. Hughes PC (1986) Spacecraft attitude dynamics. Wiley, New York
45. Dixon S (2006) BeatRoot: an interactive beat tracking and visualisation system (software tool). http://www.eecs.qmul.ac.uk/simond/beatroot
46. Sofras P (2006) Dance composition basics: capturing the choreographer's craft. Human Kinetics, Champaign
47. Minton SC (2007) Choreography: a basic approach using improvisation, 3rd edn. Human Kinetics, Champaign
48. Tolstov GP, Silverman RA (1962) Fourier series. Courier Dover Publications, New York
49. Fraleigh SH (1987) Dance and the lived body: a descriptive aesthetics. University of Pittsburgh Press, Pittsburgh
50. Augugliaro F (2011) Dancing quadrocopters - trajectory generation, feasibility and user interface. Master's thesis, ETH Zurich, Switzerland, 2011. http://dx.doi.org/10.3929/ethz-a-007328864
51. How J, Behihke B, Frank A, Dale D, Vian J (2008) Real-time indoor autonomous vehicle test environment. IEEE Contr Syst Mag 28(2):51–64
52. Lupashin S, Schoellig AP, Sherback M, D'Andrea R(2010) A simple learning strategy for high-speed quadrocopter multi-flips. In: Proceedings of the IEEE international conference on robotics and automation (ICRA), 2010, pp 1642–1648
53. Hanna JL (1987) To dance is human: a theory of nonverbal communication. University of Chicago Press, Chicago
54. Augugliaro F, D'Andrea R (2013) Admittance control for physical human-quadrocopter interaction. In: Proceedings of the European control conference (ECC), 2013, pp 1805–1810

Chapter 5
Robotic Puppets and the Engineering of Autonomous Theater

Elizabeth Jochum, Jarvis Schultz, Elliot Johnson and T. D. Murphey

5.1 Puppets Manipulated by Machines Manipulated by Engineers

Creating autonomous machines that have artistic or esthetic functions has been a subject of inquiry since antiquity. The merging of control theory with artistic practice has shifted the approach of artists and engineers away from creating systems that merely imitate artful or expressive gestures toward those in which the actions or behaviors are directly related to the environment and the system dynamics. Ideally, this research will contribute to a deeper understanding of the relationship between art and the behavior of dynamical systems. However, as the experiments in this book demonstrate, it can be difficult to differentiate those features which distinguish artistic function from utilitarian function. Rather than precisely defining what makes a work of art *artful*—itself an elusive task that has occupied art historians and esthetic philosophers for centuries [1–3]—it can be advantageous to use an established art form or practice as the basis for criteria and evaluation of experiments. The esthetic frameworks of established art forms such as dance (as in Chap. 4), musical composition (as in Chap. 7), or puppetry (as in Chap. 1) provide external referents that enable researchers to measure the degree to which systems are able to generate artistic behaviors or artifacts.

Our project uses marionette puppetry as a testbed for exploring the automated synthesis of control strategies for complex, highly dynamic, underactuated systems. Marionette puppetry has a unique approach to creating expressive, mimetic

This material is based upon work supported by the National Science Foundation under award IIS-0917837. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

E. Jochum · J. Schultz · E. Johnson · T. D. Murphey (✉)
Northwestern University, Evanston, IL, USA
e-mail: t-murphey@northwestern.edu

A. LaViers and M. Egerstedt (eds.), *Controls and Art*,
DOI: 10.1007/978-3-319-03904-6_5, © Springer International Publishing Switzerland 2014

behaviors that approximate human gestures and behaviors using a range of abstracted motions that indicate—but do not replicate—recognizable human motions. In performance, puppets acquire a grace and agility not often seen in animatronics themselves automated systems that aim at mimicry but whose movements are typically heavy, slow, and perfunctory. The goal of our project is to emulate the control technique of human puppeteers and to develop automated puppets while maintaining the natural dynamics of marionettes. Controls strategies that preserve the marionette's dynamics are important as puppets create the illusion of life through the art of indication rather than precise mechanical reproduction. We anticipate that our robotic marionette platform will allow for a wider, more artistic range of automated motions for entertainment robots.

Artists and engineers have long experimented with developing efficient methods for simulating highly articulated rigid body systems [4–9]. However, these efforts have typically focused on designing stable physical motions rather than on *control* calculations. This chapter focuses on the question of how to both simulate and control an arbitrarily complex rigid body system while maintaining scalability and convergence of the resulting numerical routines. Recognizing the potential of this research for entertainment, industrial, and medical applications, we use a robotic marionette system (seen in Fig. 5.1) as an example of a complex system that requires carefully embedded control that can handle many degrees of freedom. String marionettes are interesting because they partly resist the puppeteers' attempts to direct them: puppeteers are forced to compromise with the dynamics of the underactuated puppet to create recognizable representations of human motion. A control-based analysis of string puppetry prompts the question of whether or not puppets can be programmed to perform autonomously—a question that has been considered with great vigor by theater artists [10–13] and is of increasing relevance given contemporary theatrical productions that combine animatronic technology with marionette control systems, such as Global Creatures' *How To Train Your Dragon* (2012) and *King Kong* (2013).

This project is a collaboration with Georgia Tech, the Atlanta Center for Puppetry Arts, and Walt Disney Imagineering/Disney Research. Disney Imagineering has played a central role in developing the hardware platform. While the use of animatronics is widespread in film, museums, and theme parks its influence on live performance has been largely negligible. The absence of animatronics from theater stages can be attributed to the tremendous technical difficulties and safety risks posed by combining massive robots alongside human performers. However the more likely explanation for their absence is that animatronics lacks the human feeling and artistry found in the live performance of direct-contact puppetry. When compared to marionettes, traditional animatronics are heavy, slow, and expensive. Robotic marionettes promise to be both more agile and less costly, and could potentially expand the possibilities for automated theatrical performances. For example, Global Creatures 2013 production of the stage musical *King Kong* combined traditional puppetry techniques with automated marionette control to manipulate a 6 m tall silverback gorilla alongside human performers. While the partially automated performance signaled a new paradigm for robotic puppets, the refined gestures and expressive behaviors still relied heavily on real-time operation by human puppeteers [14]. In this case,

Fig. 5.1 The robotic marionette system in **a** is actuated by small-wheeled robots that run on the underside of a tarp. The goal is to use motion captured from the dancers in **b** as reference data for the marionettes. Software must transform the dancers' motion into dynamically admissible motion for the marionettes and combine these motions together using choreography

the reliance on human puppeteers indicates the tremendous technical challenges of automating a process that, through training, happens intuitively for humans.

Controlling marionettes is a very challenging technical problem: string marionettes have many degrees of freedom, have mechanical degeneracy due to the strings, are very lowly damped, and are highly constrained. However, human puppeteers have demonstrated a reliable ability for controlling string marionettes and solving these high-dimensional motion planning problems—puppeteers convincingly imitate human motion using marionettes—so we know the problems are solvable.

Part of this project involved the authors working with professional puppeteers to formally understand how they prepare for performance and what decisions they make during performances. Working alongside puppeteers at the Center for Puppetry Arts in Atlanta, Disney Research, and the Denver Puppet Theater provided an opportunity to discover, in practical terms, how puppeteers see their jobs as a combination of algorithmic concerns (e.g., how does one organize motion) and design concerns (e.g., how does one produce mechanical objects that are easily manipulated to produce desired motions). Puppets are complex mechanisms—controlling them involves constant trade-off between mechanical capacity and sophistication of expression. Puppeteers emphasize three phases of motion imitation—Imitate,

Simplify, Exaggerate—that represent the need to capture a motion, to then simplify it for reliable mechanical reproduction, and then exaggerate it for performance. Puppeteers also make strategic decisions about how to build a puppet that will be amenable to certain motions and how to best coordinate the physical relationship between the puppeteer and the performing object. Puppeteers coordinate the timing of a motion so they can interact with other puppeteers, sometimes collaborating to control a single marionette or groups of puppets, ensuring that the marionettes remain animated throughout the performance. Scripts of puppet plays describe the action using four parameters: temporal duration, agent, space, and motion (i.e., when, who, where, and what). These motions are grouped and executed according to counts that specify when each motion begins and ends. During rehearsals and performance, the puppeteer makes decisions about the use of force, dynamics, and movement qualities that determine the expressive characteristics and the overall visual effect, handling complex choreographic sequences and solving problems of uncertainty, often before they arise. These are the processes that we set out to understand.

Using a control-based analysis, we aim to understand how puppeteers manage complexity and uncertainty and apply these insights to autonomous theater productions and optimization problems in other critical areas. Our current hypothesis is that choreography plays a critical role in how puppeteers manage complexity, and that the study of marionette choreography will further our understanding of other complex applications, such as embedded control of prosthetics (briefly discussed at the end of this chapter). Our goals are to synthesize motion control for complex systems. To that end, choreography provides a way of categorically identifying useful and recognizable motions (e.g., walking, running, waving, reaching) that can be combined together in phrases. How one combines and orders these motions determines the stability and smoothness of transitions between motions.

This chapter is organized as follows: Section 5.2 introduces marionette puppetry and articulates the distinct technical challenges of generating recognizable and artistic motions using marionettes. Section 5.3 describes typical analytical approaches in dynamic simulation and optimal control and the specific software requirements these approaches create. Section 5.4 discusses which special considerations should be taken into account when working in discrete time. Section 5.5 illustrates our software approach and briefly discusses examples of systems that the software automatically optimizes successfully. Section 5.6 discusses the relevant features of the current framework and the broader impacts for automated puppetry and the engineering of autonomous theater.

5.2 Puppets: Esthetic and Mechanical Considerations

From antiquity to the present, artists and engineers have sought to create mechanical figures that generate expressive and lifelike behaviors [10, 15]. Puppets are part of this lineage, and are representatives *par excellence* of humanity's insatiable thirst for bringing ordinary object to life through motion. As art and technology scholar

Chris Salter has observed, the histories of performing machines and robotic art have often involved a continual mingling between mimetic aspects (that is, objects that are imitative of lifelike behavior in appearance) and machinic aspects (electromechanical behavior that, though animate, is not anthropomorphic) rather than a disambiguation [16]. That is, human artists are interested in designing machines that emulate aspects of human creativity. Before the advent of computing and electronics, the approach was to design objects that appeared lifelike and imitated human actions, such as the eighteenth century humanoid automata built by Jacques de Vaucanson and Henri-Louis and Pierre Jaquet-Droz [17]. Puppets are the progenitors to these and other attempts to create mechanical life, and are part of a lineage that extends to present-day animatronics.

Puppets can take the shape of realistic or abstract figures, and are designed for use in theatrical settings. A puppet is generally defined as a material object that makes temporal use of sources of power that exist outside of itself, and that are not its own attributes [10, 18]. Animatronics are puppets that are electronically controlled through various actuators (electrical, hydraulic, pneumatic); however, the absence of a human performer and repetitive or open-loop performance means that they are not typically regarded in the same way that human-powered puppets are. Setting aside for a moment the question of human agency, we recognize that the main purpose of a puppet is movement: to establish a meaningful presence the puppet relies on motions to create a character or presence that is both recognizable to the spectator and conveys a certain artistic truth. The puppet's power of expression is therefore not determined by how well it precisely mimics human behavior, but rather by its ability to abstract human motions and offer an artistic projection of those motions and behaviors. In other words, the goal of puppetry motion is not to copy but to *create*. As Kingston et al. have also observed, a puppets primary purpose is to communicate through motion (as in Chap. 1).

Puppeteers are extremely inventive, and have developed numerous methods for controlling (often called "manipulating") and constructing expressive, moving objects. String marionettes are dynamically unique among puppets, and more than other types of puppets have the potential to teach us about optimal control. Unlike glove puppets or rod puppets which are controlled through direct, corporeal contact (hands-on, hands-in), string marionettes are operated from above by a varying number of strings. For humanoid puppets, these strings are usually attached to the puppet's head, torso, shoulders, arms, and legs, and can be strung to generate specific actions based on the specifications of the choreography. The strings can be manipulated with a variety of controls—the most common is the multistringed wooden control or "airplane" (crossbar) mechanism. This technique is used in Asian and European puppetry forms, and puppeteers can opt to vary the string lengths and attachment points according to the needs of each production or scene. The precision of the puppet's motions is relative to the control of the figures—the longer the strings, the weaker the impetus, which results in softer, less precise movements [10]. Puppeteers can control marionettes at ground level working alongside the puppet and in full view of the audience, or from a position high above the stage from a bridge and out of the audience's sight lines. In either arrangement, the puppeteer must learn to balance

the dynamics of the puppet against the need to execute expressive choreography that convincingly imitates—but does not replicate—human motions. Because string marionettes resist the puppeteer's attempts to direct them, puppeteers have developed highly evolved strategies for generating sequences of complex motions—this is what is known as marionette choreography. Because of their dynamics, string marionettes are arguably the most difficult type of puppet to operate. This also makes them a good testbed for a control-based analysis.

5.3 Typical Approach

In this section, we discuss typical analytical approaches to optimal control and the types of software infrastructure these approaches assume. We start with a discussion of how one might describe the dynamics of a mechanical system and then discuss computing optimal controllers for that system.

5.3.1 Dynamics

Marionettes are subject to the physics of the world: they swing and sway according to the forces of gravity and the interplay between the different bodies—individual units such as the torso, forearms, and legs—in the marionette. This interplay between the bodies falls within the realm of dynamics and simulation, where the dynamic description comes from a physics-based understanding of puppet motion. When computing dynamics, we are typically trying to compute equations of the form

$$\dot{x} = f(x, u) \tag{5.1}$$

where $x = (q, \dot{q})$ and $q \in Q$ describes the configuration of the system. For rigid body systems, it has historically been convenient to write down the rigid body system in Newton-Euler coordinates (i.e., $Q = SE(3)^n$, where n is the number of rigid bodies in the system. This yields a state space of dimension $12n$ that is subject to constraints. For a typical humanoid marionette, for instance, the marionette alone (no actuators) has 10 rigid bodies, so the state space would be more than 120 dimensions. If one includes string actuation in the degree count, that adds one degree of freedom for each string's length and an additional element of $SE(3)$ for each string endpoint (treating the inputs as "kinematic inputs" [19]). For a typical marionette with six strings of variable lengths, this brings the total nominal dimension of the state space up to $12 \cdot 10 + 2 \cdot 6 + 12 \cdot 6 = 204$. Naturally, we do not want to be solving for feedback controllers in a 204 dimensional space if it can be avoided. To avoid such high dimensions, we do not want to represent Eq. (5.1) as Newton-Euler equations and instead insist on working in generalized coordinates. In the case of the marionette, this reduces the dimension of the state to $2m$, where m is the number

of generalized coordinates. This yields 22 configuration variables for the marionette itself and another 18 degrees of freedom for the string actuators (six strings with endpoints moving in $\mathbb{R}^2 \times \mathbb{R}$), yielding 80 states. By utilizing a kinematic reduction [21, 22], we can reduce the state of the actuators down to 18 because they are fully actuated. This gives us equations of motion of the following form:

$$\begin{aligned} \dot{x}_a &= u \\ \dot{x}_p &= f(x_p, x_a) \end{aligned} \tag{5.2}$$

where x_a is the kinematic configuration of the actuators and $x_p = (q_p, \dot{q}_p)$ is the dynamic configuration *and* velocity of the marionette itself. (For details on this, see [19].) This leaves us with a much smaller, more manageable system to work with that only has a total of 62 dimensions in its state space. Crucially, the strings are modeled as holonomic constraints, relating the lengths of the strings to the dynamic configurations, of the form

$$h(x_a, q_p) = 0 \tag{5.3}$$

which must be maintained during simulation and control. Assuming for the moment that Eqs. (5.2) and (5.3) can be stably simulated in an efficient manner, how do we then construct the differential equation and constraints in a systematic manner? The standard way to do this is based on Featherstone's early work [9] on articulated body dynamics. This work was largely used in the context of animation and digital puppetry, where the requirements are substantially different than embedded control. For example, animation requires a simulation to "look right" only once, while controlled physical systems must be repeatable. Recursive approaches to calculating dynamics [9, 23] take advantage of special representations of mechanical systems that allow the values needed for simulation to be calculated quickly and avoid redundant calculations. The work in this chapter is based on the methods presented in [20] (based on [9]). Here, systems are represented as graphs where each node is a coordinate frame in the mechanical system and the nodes are connected by simple rigid body transformations (typically translations along and rotations about the X, Y, and Z axes, though any rigid body screw motion can be used). Transformations are either constant or parameterized by real-valued variables. The set of all variables establishes the generalized coordinates for the system. Figure 5.2 is an example of a simulated marionette. The graph description can include closed kinematic chains, but in practice the graph is converted to an acyclic directed graph (i.e., a tree) and augmented with holonomic constraints to close the kinematic chains. This approach leads to fast calculations of $f(\cdot, \cdot)$ in Eq. (5.2) and $h(\cdot)$ in Eq. (5.3). Moreover, one can use the same structure to efficiently calculate the linearization [24], which is critical to nonlinear optimal control calculations, discussed in Sect. 5.3.2.

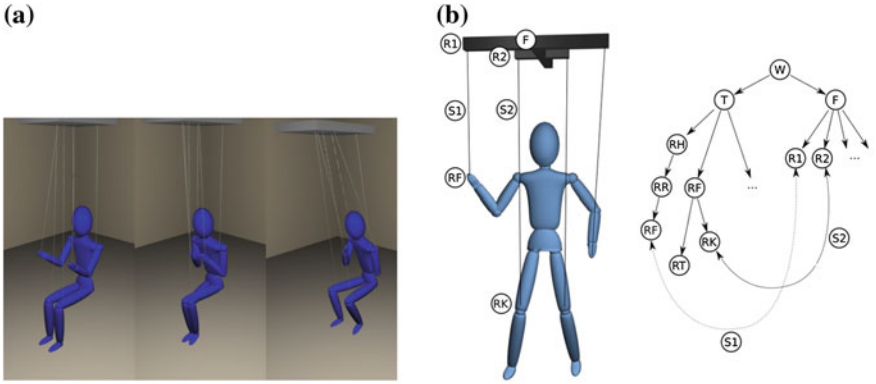


Fig. 5.2 Simulation of complex rigid bodies can take advantage of the mechanical topology of the system. For instance, a marionette is being simulated in **a** using a tree structure representation of the humanoid form in **b** and representing the constraints by cycles in the graph (see [20])

5.3.2 Nonlinear Optimal Control

Controlling marionettes involves choosing how the strings must be operated in order to achieve some desired trajectory. For instance, if we wish a marionette to generate a walking motion, the strings must pull the limbs in such a way that the body of the marionette indicates walking. The choice of string motions will, of course, depend on the physics-based description of the marionettes. For instance, if the masses of the legs are very high, we might have to pull on the strings differently than if the masses were very low. Optimal control plays a significant role in determining the outcome because it provides an algorithmic means of choosing string lengths and endpoint positions in a manner that takes the physics-based description of the marionettes into account. Optimal control typically starts out with a cost function of some sort, often of the form

$$J = \int_{t_0}^{t_f} L(x(t), x_{\text{ref}}(t), u(t))dt + m(x(t_f), x_{\text{ref}}(t_f)) \tag{5.4}$$

where $L(\cdot)$ represents a weighted estimate of the error between the state and the reference state (which is potentially not a feasible trajectory for the system) such as in the imitation problem mentioned above. We can minimize this cost function subject to the dynamics in Eqs. (5.2) and (5.3) by using iterative descent methods. In particular, one uses the equivalence between the constrained minimization and the unconstrained minimization of the objective function composed with a differentiable projection $\mathscr{P}(\cdot)$ onto the constrained subspace. That is, the two minimizations

$$\min_{v \in W \subseteq V} g(v) = \min_{v \in V} g(\mathscr{P}(v))$$

(where V is the vector space and W is the differentiable submanifold of admissible vectors) are equivalent [25]. The projection operator $\mathscr{P}(\cdot)$ comes from computing a feedback law (discussed in more detail in Sect. 5.3.3). In particular, if one interprets the "gradient" descent algorithm as starting at some nominal trajectory $\xi = (x(t), u(t))$ and solving for a descent direction $\zeta = (z, v)$ that optimizes the local quadratic model

$$\zeta = \arg\min_{\zeta} Dg(\xi) \cdot D\mathscr{P}(\xi) \cdot \zeta + \|\zeta\|^2,$$

then one must only solve a standard time-varying LQR problem. This means that one must be able to compute the time-varying linearization

$$\dot{z} = A(t)z + B(t)v \tag{5.5}$$

where $A(t) = \frac{\partial f}{dx}(x(t), u(t)) = D_1 f(x(t), u(t))$ and $B(t) = \frac{\partial f}{du}(x(t), u(t)) = D_2 f(x(t), u(t))$. One has to be able to do so for arbitrary trajectories in the state space, potentially including infeasible trajectories (in the case of linearizing about the desired trajectory). Solving for the descent direction involves computing the Riccati equations

$$\dot{P} + A(t)^T P + PA(t) + Q - PB(t)R^{-1}B(t)^T P = 0. \tag{5.6}$$

If we additionally want to guarantee quadratic convergence, then we can find a descent direction by solving a different LQR problem

$$\zeta = \arg\min_{\zeta} Dg(\xi) \cdot D\mathscr{P}(\xi) \cdot \zeta + \|\zeta\|^2_{D^2 J}$$

where

$$\begin{aligned} D^2 J(\xi) \cdot (\zeta^1, \zeta^2) = {} & D^2 g(\xi) \cdot (D\mathscr{P}(\xi) \cdot \zeta^1, D\mathscr{P}(\xi) \cdot \zeta^2) \\ & + Dg(\xi) \cdot D^2 \mathscr{P}(\xi) \cdot (\zeta^1, \zeta^2). \end{aligned} \tag{5.7}$$

The second derivative $D^2\mathscr{P}(\cdot)$ requires that we be able to also calculate $\frac{\partial^2 f}{dx^2}$, $\frac{\partial^2 f}{du^2}$, and $\frac{\partial^2 f}{dxdu}$. The details of this approach can be found in [25] and elsewhere, but for our purposes we should be able to compute Eqs. (5.2)–(5.7) in software. The difficulty of this approach is that we are representing the optimal control problem in continuous time while the actual computations are in discrete time. We will discuss this further in Sect. 5.4 and we see that it is only when we perform optimal control calculations in discrete time that we arrive at convergence of the algorithms that provide motion imitation.

5.3.3 Choreography and Hybrid Optimal Control

In [26–28], we developed an optimal control interpretation of puppet choreography. In particular, we formalized choreography as a sequence of *modes* that can be pieced together to form a script of motions. Each mode has its own dynamics, creating a system with dynamics

$$\dot{x} = f(x(t), u(t)) = f_i(x(t), u(t)) \quad t \in (\tau_i, \tau_{i+1})$$

where each i corresponds to a different mode of the system. To optimize such a system, one needs to be able to minimize an objective function J with respect to the switching times τ_i of the system. For a gradient descent algorithm, one must be able to compute the derivative $\frac{\partial J}{\partial \tau_i}$ —the derivative of the cost function with respect to the switching times—which depends on the switching time adjoint equation

$$\dot{\rho} + A(t)^T \rho + \frac{\partial L}{\partial x} = 0 \tag{5.8}$$

along with a boundary condition at $\rho(t_f)$ (see [29–32]). This adjoint equation only needs to be computed once to compute all the derivatives of J. If one wants to compute the second derivative of J, e.g., to utilize Newton's method, then the second-order switching time adjoint equation

$$\dot{P} + A(t)^T P + P A(t) + \frac{\partial^2 L}{\partial x^2} + \sum_k \rho_k \frac{\partial^2 f^k}{\partial x^2} = 0 \tag{5.9}$$

along with its boundary condition $P(t_f)$ must be solved [30]. This adjoint equation, along with a solution to Eq. (5.8), needs to be computed only once to obtain all the derivatives of J. Note that the second-order switching time adjoint equation is the same as the Riccati equation in Eq. (5.6) except that the Riccati equation has a different final term. Indeed, both Eqs. (5.8) and (5.9) only require first and second derivatives of f_i with respect to the state, so those are all that are needed for software; hence, the choreographic optimization requires the same software capabilities as the smooth optimization described in Sect. 5.3.2.

5.4 Discrete Time with Scalability

As previously mentioned, the continuous representation of dynamics found in Eq. (5.1) is not what we actually use to do computations. Moreover, when there are constraints, such as those seen in Eqs. (5.2) and (5.3), standard methods such as Runge–Kutta methods fail to preserve the constraints. Typically, one would use solvers designed for Differential Algebraic Equations (DAEs) that project the

Fig. 5.3 Software using variational integrators as the basis for simulation can accurately predict marionette motion using comparatively large time steps. In this case, the time step is $dt = 0.01\,\mathrm{s}$

numerical prediction onto the set of constrained solutions defined by the constraint in Eq. (5.3). We have found, however, that for high-index DAEs such as the marionette a tremendous amount of "artificial stabilization" is required to make the simulation of the DAE stable. This artificial stabilization—which typically takes the constraint $h(q)$ as a reference and introduces a feedback law that "stabilizes" the constraint—changes the dynamics of the system, and if the feedback gain is high, this often creates a multiscale simulation problem that is incompatible with real-time operation. As an alternative, we consider variational integrators [33–39]. Variational integration methods use the stationary action principle as a foundation for numerical integration that does not involve differential equations. This approach has several advantages such as guarantees about conservation of momenta, the Hamiltonian, and the constraints, as well as guaranteed convergence to the correct trajectory as the time step converges to zero. More importantly, variational integration techniques exactly simulate a *modified Lagrangian* system where the modified Lagrangian is a perturbation of the original Lagrangian. The Discrete Euler-Lagrange (DEL) equations are (Fig. 5.3)

$$D_1 L_d(q_k, q_{k+1}, k) + D_2 L_d(q_{k-1}, q_k, k) = F_k \tag{5.10}$$

$$h(q_{k+1}) = 0 \tag{5.11}$$

where L_d is a discretized form of the Lagrangian and F_k is an external force integrated over the k time step. This forms a root solving problem in which, given q_{k-1} and q_k, one solves for q_{k+1}. Alternatively, one can utilize the discrete Legendre transform to define the discrete generalized momentum p_k, and then convert the root solving problem of Eq. 5.11 into a one-step root solving problem where, given the pair (q_k, p_k), one implicitly solves for the next time step pair (q_{k+1}, p_{k+1}). Repeating this root solving procedure forms the basis of simulation. Using this method, we can (using a recursive tree description similar to the one described in Sect. 5.3.1),

simulate the marionette in real time using time steps of 0.01 s without adding any sort of numerical heuristics, such as artificial stabilization. Let us say we start from the DEL equations and assume, by application of the implicit function theorem, that the solution exists and is locally unique [40]. Once we have made a choice of state (we choose $x_k = (q_k, p_k)$), we have an update equation of the form

$$x_{k+1} = f_k(x_k, u_k)$$

just as we would if we had started from a differential equation. That is, the general form of the discrete time equation we wish to optimize is in principle no different in the variational integrator case than it is in the standard ODE case. More importantly, the fact that f_k is implicitly defined by the DEL equations does not affect whether the linearization is implicitly defined. In fact, one can calculate an *exact* linearization of the DEL equations, including constraints and closed kinematic chains [24]. So we may wish to know what the discrete time version of Eqs. (5.6) and (5.7) are. (These can be found in [41].) The difficulty is that one cannot linearize Eq. (5.1) directly because that is the infinitesimal linearization; to get a discrete time linearization one would nominally have to solve the state transition matrix (STM) locally over the time step. Approximating the STM leads to a linearization that does not respect the constraints, leading to a local optimal controller that essentially fights the constraints. In contrast, taking variations directly with respect to x_k yields an algebraic calculation for the linearization and higher order derivatives with respect to the state. As with variational integration, the key to linearization is to take variations with respect to the discrete state rather than the continuous state. For nonlinear optimal control in the discrete time setting, we need to know if the resulting projection operator (as discussed in Sect. 5.3.2) is in fact a projection and whether it is differentiable.To see that such a projection is valuable, consider Fig. 5.4 [42], where a planar double pendulum trajectory is being optimized. The initial guess for the optimal solution is the "zero" solution, the optimal solution is the solid black line, and the *first* iteration of Newton's method using the projection operation is the dotted line. Hence, one step of Newton's method almost solves the global optimization in this case. Naturally, that will not always be the case, but this is an indication of how much generating a differential projection operation can help. Let $\overline{\xi} = (\overline{x}, \overline{u})$ be a desired, potentially infeasible, curve in the space the trajectories reside in and let $\xi = (x, u)$ be feasible trajectories. The continuous time projection operator is defined by $\xi = \mathscr{P}(\overline{\xi})$ such that

$$\begin{aligned} x(t_0) &= \overline{x}(t_0) \\ \dot{x} &= f(x, u) \\ u &= \overline{u} - K(t)(x - \overline{x}) \end{aligned}$$

where the feedback gain $K(t)$ comes from solving the Riccati equation in Eq. (5.6). One can verify that $\mathscr{P}(\cdot)$ is a projection and that it is C^∞ if f is C^∞. What do we do if we are using Eqs. (5.10) and (5.11) instead of Eqs. (5.2) and (5.3)? Then the

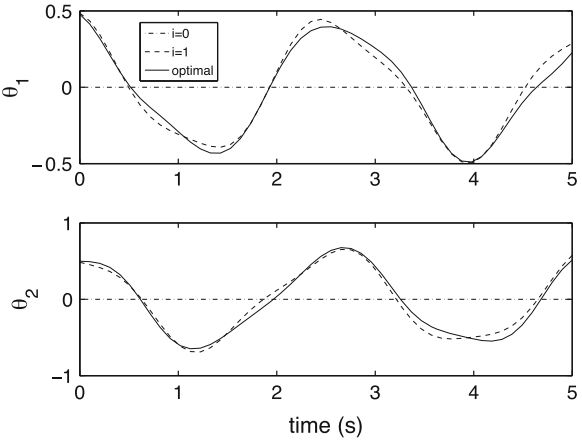


Fig. 5.4 Projection-based variational optimization of a planar double pendulum [42]

discrete projection operator $\mathscr{P}_d(\cdot)$ is $\xi_d = \mathscr{P}_d(\bar{\xi}_d)$ such that

$$\begin{aligned} x_0 &= \bar{x}_0 \\ x_{k+1} &= f_k(x_k, u_k) \\ u_k &= \bar{u}_k - K_k(x_k - \bar{x}_k) \end{aligned} \tag{5.12}$$

where the discrete time feedback gain K_k comes from solving a discrete time Riccati equation. To see that it is a differentiable projection, we introduce the following Lemma.

Lemma 1 *$\mathscr{P}_d(\cdot)$ is a projection.*

Proof We need to show that the projection satisfies the property $\mathscr{P}_d(\bar{\xi}_d) = \mathscr{P}_d(\mathscr{P}_d(\bar{\xi}_d))$. First we calculate several terms of $(a, b) = \mathscr{P}_d(\alpha, \mu)$ and get $a_0 = \alpha_0$, $b_0 = \mu_0 - K_0(a_0 - \alpha_0) = \mu_0$, $a_1 = f_0(a_0, b_0) = f_0(\alpha_0, \mu_0)$, and $b_1 = \mu_1 - K_1(a_1 - \alpha_1)$. Now calculate several terms of $(x, u) = \mathscr{P}_d(a, b)$ and find that $x_0 = a_0 = \alpha_0$, $u_0 = b_0 - K_0(x_0 - a_0) = b_0 = \mu_0$, $x_1 = f_0(x_0, u_0) = f_0(\alpha_0, \mu_0) = a_1$, and $u_1 = b_1 - K_1(x_1 - a_1) = b_1 - K_1(a_1 - a_1) = b_1$. By induction, we find $\mathscr{P}_d \circ (\alpha, \mu) = \mathscr{P}_d \circ \mathscr{P}_d \circ (\alpha, \mu)$. Hence, $\mathscr{P}_d(\cdot)$ is a projection operation from discrete time representations of curves $\bar{\xi}_d$ to discrete time representations of trajectories ξ_d.

Next we need to calculate the derivative of $\mathscr{P}_d(\cdot)$, starting with $\xi_d = \mathscr{P}_d(\bar{\xi}_d)$ (we are going to drop the d from ξ_d for notational convenience).

$$\delta\xi = D\mathscr{P}_d(\bar{\xi}) \circ \delta\bar{\xi}$$

So, by Eq. 5.12, we get

$$\begin{aligned}\delta x_0 &= \delta \bar{x}_0 \\ \delta x_{k+1} &= \frac{\partial f_k}{\partial x_k}\delta x_k + \frac{\partial f_k}{\partial u_k}\delta u_k = Df_k \circ \delta\xi_k \\ \delta u_k &= \delta \bar{u}_k - K_k(\delta x_k - \delta \bar{x}_k).\end{aligned}$$

where $\frac{\partial f_k}{\partial x_k}$ is shorthand for $\frac{\partial f_k}{\partial x}(x_k, u_k, k)$. (The same applies to $\frac{\partial f_k}{\partial u_k}$ and Df_k.) As in the continuous case, the derivative of the discrete projection is a discrete linear system. The second derivative is also straightforward (here we let $\delta\bar{\xi}^1$ and $\delta\bar{\xi}^2$ be two independent perturbations to $\bar{\xi}$).

$$\delta^2\xi = D^2\mathscr{P}(\bar{\xi}) \circ (\delta\bar{\xi}^1, \delta\bar{\xi}^2)$$

which implies, again by Eq. (5.12), that

$$\begin{aligned}\delta^2 x_0 &= 0 \\ \delta^2 x_{k+1} &= D^2 f_k \circ (\delta\xi_k^1, \delta\xi_k^2) + Df_k \circ \delta^2\xi_k \\ &= \frac{\partial f_k}{\partial x_k}\delta^2 x_k + \frac{\partial f_k}{\partial u_k}\delta^2 u_k + D^2 f_k \circ (\delta\xi_k^1, \delta\xi_k^2) \\ \delta^2 u_k &= -K_k\delta^2 x_k.\end{aligned}$$

Rewriting the second derivative, we get:

$$\begin{aligned}\delta^2 x_{k+1} &= \frac{\partial f_k}{\partial x_k}\delta^2 x_k + \frac{\partial f_k}{\partial u_k}\delta^2 u_k + D^2 f_k \circ (\delta\xi_k^1, \delta\xi_k^2) \\ &= \left[\frac{\partial f_k}{\partial x_k} - \frac{\partial f_k}{\partial u_k}K_k\right]\delta^2 x_k + D^2 f_k \circ (\delta\xi_k^1, \delta\xi_k^2).\end{aligned}$$

This is a discrete affine system, equivalent to a discrete linear system with an input:

$$\begin{aligned}\delta^2 x_{k+1} &= \hat{A}_k\delta^2 x_k + \hat{B}_k \\ \hat{A}_k &= \left[\frac{\partial f_k}{\partial x_k} - \frac{\partial f_k}{\partial u_k}K_k\right] \qquad \hat{B}_k = D^2 f_k \circ (\delta\xi_k^1, \delta\xi_k^2).\end{aligned}$$

Hence, the projection operation $\mathscr{P}_d$ is twice differentiable with derivatives that are represented by discrete time linear difference equations, allowing us to apply Newton's method to optimal control problems.

5.5 Examples

We now consider the model of a humanoid marionette shown in Fig. 5.5. The marionette has 40 configuration variables and is actuated by six strings. The strings are modeled as holonomic constraints. Kinematic configuration variables—those that

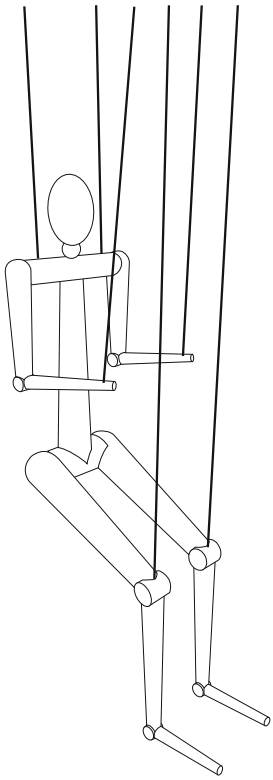


Fig. 5.5 The 3D Marionette is actuated by six strings

we assume satisfy first-order dynamics where we control the velocity of the variable directly—control the two-dimensional position of the endpoint of each string as well as the string length. There are no joint torques and only slight damping applied to each dynamic configuration variable equally. In order to indicate the real-time feasibility of the algorithms discussed in the previous sections, we now provide some timing data for the marionette simulation in Fig. 5.5 we have a system that has 22 dynamic degrees of freedom, 18 kinematic degrees of freedom (corresponding to the actuation of the strings), and six total holonomic constraints. To evaluate the continuous dynamics that would be used in a standard integrator, one evaluation of $f(x, u)$ requires 2.7 ms, while the first derivative with respect to the state (i.e., the linearization) requires 24 ms and the second derivative with respect to the state requires 400 ms. Note that this does not indicate how long it will take to simulate a particular length of time because the time step is not included here as we are not working in discrete time. With the variational integrator from Eqs. (5.10) and (5.11) with a time step of 0.01 (no other parameters are needed when using variational integrators, even with the degeneracies and constraints the strings introduce), the update step requires 5.53 ms while the first derivative with respect to the state (i.e.,

the exact discrete linearization) takes 2.4 ms and the second derivative with respect to the state takes 130 ms. This means that, at minimum, we can simulate and evaluate controllability in real time. Two optimizations are discussed in the following subsections. The optimization in Sect. 5.5.1 uses a desired trajectory that was generated separately by simulating the system. Section 5.5.2 discusses an optimization with a reference generated from motion capture data.

5.5.1 Desired Motion: Simulated Trajectory

A desired trajectory was generated by simulating the system forward in time. The lengths of the arm and leg strings were varied sinusoidally to create a "walking" motion. The configurations of the trajectory were saved. The rest of the state (e.g, configuration velocity or discrete momentum) and the simulation inputs were discarded and replaced with uniformly zero trajectories. This results in a smooth desired trajectory that we expect the puppet to be able to track, but remains an infeasible trajectory. The marionette was optimized to the desired trajectory in both continuous and discrete time. Both optimizations successfully converged to solutions that track the desired configuration very well. Convergence plots for both optimizations are shown in Fig. 5.6. The source code for the discrete optimization is distributed with `trep` at http://trep.googlecode.com in the file `/examples/puppet-optimization.py`. Figure 5.6 shows that the continuous optimization initially converges drastically faster than the discrete one: it tracks the desired trajectory almost perfectly after a single step. The discrete optimization makes slow progress initially but converges quickly after about five iterations. The discrete optimization takes 5–50 s to finish. Each iteration takes between 15 and 60 s depending on the descent direction type and number of Armijo steps. Although the convergence plot is flattering for the continuous optimization, there were numerous problems. Unlike the discrete optimization, the continuous optimization was highly sensitive to the optimization parameters. Large ratios between the maximum and minimum state cost cause the optimization to fail. Additionally, for a successful continuous optimization, the terminal conditions must be significantly relaxed. The discrete time optimization suffers from neither of these problems.

5.5.2 Desired Motion: Motion Capture Data

A more practical application of the trajectory optimization is finding trajectories to track data acquired from a motion capture system. In this example, a desired trajectory was generated using a Microsoft Kinect® to record a student walking in place. This process is illustrated in Fig. 5.7. In this case, the continuous optimization was unable to converge. The discrete optimization converged successfully and found a trajectory that closely approximates the student's movement. Figure 5.8 plots the

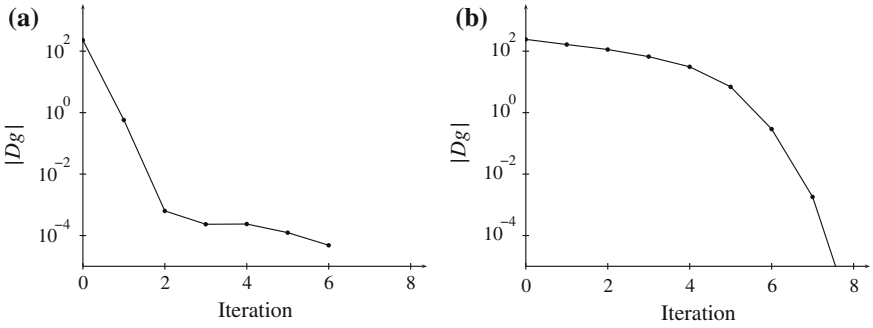


Fig. 5.6 Convergence plots for continuous and discrete time optimizations from a simulated desired trajectory. **a** Continuous time. **b** Discrete time

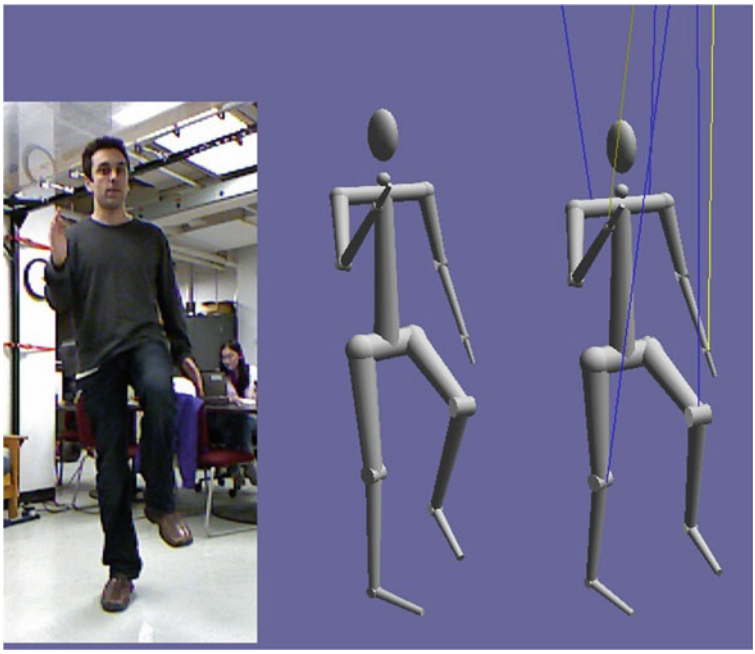


Fig. 5.7 These three images show a single frame from the motion capture optimization. The *left-most picture* shows an RGB image recorded by a Microsoft Kinect®. The *middle figure* is the motion capture data found from the Kinect's depth map. The *right-most figure* is the optimized trajectory

desired trajectory and optimization result for the angle of the right elbow as an example. The trajectory found by the optimization tracks the desired trajectory very well. However, a large amount of noise was introduced. This is most likely caused by too large of a ratio between the weight of the configuration portion of the state compared the discrete momentum portions and the cost of the inputs.

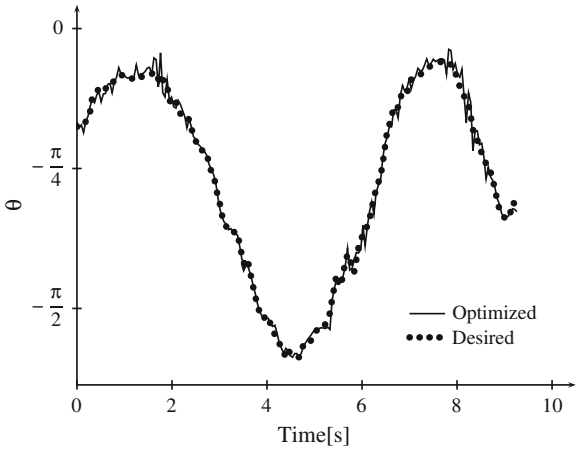


Fig. 5.8 The optimal trajectory for the *right arm* elbow tracks the desired trajectory well

5.6 Conclusions

The robotic marionette project is an example system that forces us to create software that can both simulate and control complex mechanical systems. The marionettes play a vital role in driving the system development—they have mechanical degeneracies, closed kinematic chains, and are high dimensional, but despite these features puppeteers are able to successfully and reliably control them. Therefore, marionettes make a good testbed for understanding whether or not our software is producing reasonable results. Evaluating the efficacy of the control system can be determined immediately by observing whether or not the motions and choreographic phrases are recognizable, smooth, and approximate the reference data. Viewed this way, we see how a control-based analysis of an existing art form allows us to conceptualize new approaches in optimal control, and also increases the likelihood that such a system will be used by artists to develop choreography for marionettes or other artificial, articulated bodies. The techniques we use provide both optimal trajectories and control laws that help stabilize those trajectories. Moreover, because we formulate the optimal control problem using a differentiable projection, we can analytically guarantee quadratic convergence locally around the optimal trajectory. It is also important to note that the techniques we have applied to the marionettes are also applicable to many other fields. For example, we have used these software techniques for the tendon-articulated hand in Fig. 5.9 and can compute linearizations and local LQR controllers for the hand. (This simulation capability is now being used with prosthetic control in a collaboration with the Rehabilitation Institute of Chicago.)

Puppeteers use marionettes in dynamic, expressive ways, so we presume that extremely conservative motions based on a "quasi-static" approach or an inverse-kinematics approach are unlikely to produce interesting, artistic motions. Puppetry privileges imitation over precise replication, but to be considered "artistic" the

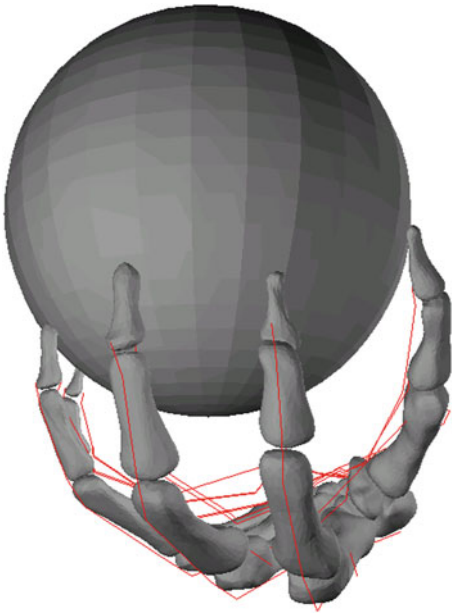


Fig. 5.9 The graph-based approach to calculate linearizations scales to complex mechanical systems like this dynamic model of a tendon-articulated human hand holding an object. The linearization at this configuration shows that the system is locally controllable

imitation must rise above perfunctory or routine motions. Typically, the artistic quality of puppetry has been assumed to be the result of the interpretation and execution of the human puppeteer, but our automated marionette platform provokes the question of whether or not human operation is essential to a marionette's expressiveness and theatrical value. Although we can produce an optimal "imitation" of a human motion in the case of a simple walking motion captured by the Kinect sensor, we must first solve the calculations for the full marionette for a variety of motions before we can claim that marionette imitation can be defined as an optimization problem. We are currently working on solving these problems. It remains to be seen whether or not a fully automated marionette can execute choreography in such a way that transcends mere imitation and achieves artful or esthetic resonance.

One potential application of the automated marionettes is live entertainment and theater productions that utilize large-scale animatronics and automated performers. While Disney pioneered the technologies of animatronics, enabling artists and engineers to partially realize their shared vision of creating an autonomous theater, Creature Technology Company and Global Creatures are developing sophisticated animatronics for use in live performance. The 2012 production based on Dreamworks' animated film *How To Train Your Dragon* and the 2013 *King Kong* are two recent productions that combine large-scale animatronics with human performers in a live production. Unlike traditional animatronics, these shows are purposefully

designed for international tours. The demands of an international touring production present considerable challenges: the animatronics must be able to perform reliably in a wide range of venues, the choreography of these productions brings live actors into close contact with heavy, dangerous machines, and audiences expect the machines to be as interactive and agile as live performers. Perhaps not surprisingly, Global Creatures has opted to work with expert puppeteers to develop choreography for large-scale animatronics in the shape of flying dinosaurs, dragons, and a six-meter tall silverback gorilla. The machines are controlled through a combination of marionette-automation and teleoperated controls (known at Global Creatures as "voodoo puppeteering") [43]. A team of puppeteers work together to control a single puppet (sometimes with a puppeteer seated in a chassis inside of the puppet), while the marionette-automation enables these large puppets to execute expressive choreography such as flying and aerial stunts—even scaling the side of a replica of the Empire State building. This unique hybrid control system expands the range and quality of motions that are available to the puppet, allowing the puppeteer to develop choreography on a larger scale than has previously been imaginable: animatronic choreography can now utilize the entire stage space. The level of automation and sensing technologies used to control the puppets further distance the human operator from the physical act of puppeteering, resulting in increasingly automated performances that are not rote or perfunctory but are rather perceived as lively and entertaining. This arrangement challenges the notion that autonomous theater can only be preprogrammed or repetitive, and reignites the debate of whether or not human operators are essential for live theater performances.

In his famous essay *What is Art?* the Russian novelist Leo Tolstoy wrote "To evoke in oneself a feeling one has once experienced and having evoked it in oneself then by means of movements, lines, colours, sounds, or forms expressed in words, so to transmit that feeling that others may experience the same feeling—this is the activity of art." As artists and engineers focus their attention on emulating artistic process through the generation of performances and artifacts based on system dynamics and control theory, established art forms such as music, dance, and puppetry provide important frameworks for modeling and evaluating the esthetic or artistic outcomes. The research projects discussed in this volume are illustrative of the urge to understand and emulate the intangible aspects of esthetics while evaluating the tangible outcomes of these investigations. Whether it is through the creation of esthetic or communicative gestures (as in Chap. 1), generating appealing or interesting musical compositions (as in Chap. 7), or expressive and engaging choreography (as in Chap. 4), the goal is to generate esthetic behaviors that are emergent and are evocative of some human feeling. Whether the application of control theory can engender similar esthetic responses as works of art generated by human artists remains to be seen, but it is a question worth considering.

References

1. Goodman N (1978) Ways of worldmaking. Hackett Publishing, Indianapolis
2. Kant I (1911) Critique of aesthetic judgement (1790). Oxford University Press, Oxford
3. Schiller F (1794) On the aesthetic education of man. Oxford University Press, Oxford
4. Smith R (2008) Open dynamics engine. http://www.ode.org
5. Smith R (2004) Dynamics simulation: a whirlwind tour (current state, and new frontiers). http://ode.org/slides/parc/dynamics.pdf
6. Baraff D (1996) Linear-time dynamics using Lagrange multipliers. In: SIGGRAPH, pp 137–146
7. Baraff D (1994) Fast contact force computation for nonpenetrating rigid bodies. In: SIGGRAPH
8. Baraff D (1993) Non-penetrating rigid body simulation. In: State of the Art Reports
9. Featherstone R (1987) Robot dynamics algorithms. Kluwer Academic Publishers, Boston. The Springer International Series in Engineering and Computer Science Vol 22, Springer US New York
10. Francis P (2012) Puppetry. Palgrave Macmillan, London
11. Kleist H (1972) On the marionette theatre. Drama Rev 16(3):22–26
12. Craig EG (2009) On the art of the theatre. Routledge, London
13. Kaplin S (1999) A puppet tree: a model for the field of puppet theatre. TDR 43(3):28–35
14. Jochum E (2013) King kong by Craig Lucas and Marius de Vries (review). Theatre J 65(4):580–582
15. Burnham J (1968) Beyond modern sculpture: the effects of science and technology on the sculpture of this century. George Braziller, New York
16. Salter C (2010) Entangled: technology and the transformation of performance. MIT Press, Cambridge
17. Wood G (2002) Edison's Eve: a magical history of the quest for mechanical life. A.A. Knopf, New York
18. Jurkowski H (1988) Aspects of puppet theatre. Puppet Centre Trust, London
19. Johnson E, Murphey T (2007) Dynamic modeling and motion planning for marionettes: Rigid bodies articulated by massless strings. In: International conference on robotics and automation, Vol 2007. Rome
20. Johnson ER, Murphey TD (2010) Scalable variational integrators for constrained mechanical systems in generalized coordinates. IEEE Trans Rob. 25(6):1249–1261
21. Bullo F, Lewis A (2005) Low-order controllability and kinematic reductions for affine connection control systems. SIAM J Control Optim 44(3):885–908
22. Bullo F, Lewis A (2004) Geometric control of mechanical systems, ser. number 49 in texts in applied mathematics. Springer, London
23. Nakamura Y, Yamane K (2000) Dynamics computation of structure-varying kinematic chains and its application to human figures. IEEE Trans Robot Autom 16(2):124–134
24. Johnson E, Murphey TD (2010) Linearizations for mechanical systems in generalized coordinates. In: American controls conference (ACC), pp 629–633
25. Hauser J (2002) A projection operator approach to optimization of trajectory functionals. In: IFAC world congress, Barcelona, 2002
26. Martin P, Johnson E, Murphey TD, Egerstedt M (2010) Constructing and implementing motion programs for robotic marionettes. IEEE Trans Autom Control 56(4):902–907
27. Egerstedt M, Murphey TD, Ludwig J (2007) Hybrid systems: computation and control. In: Bemporad A, Bicchi A, Buttazzo GC (eds) Motion programs for puppet choreography and control. Lecture notes in computer science, vol TBD. Springer, pp 190–202
28. Murphey TD, Egerstedt ME (2007) Choreography for marionettes: imitation, planning, and control. In: IEEE international conference on intelligent robots and systems workshop on art and robotics, p 6
29. Johnson E, Murphey TD (2010) Second-order switching time optimization for nonlinear time-varying dynamic systems. IEEE Trans Autom Control. 56(8):1953–1957

30. Caldwell T, Murphey TD (2011) Switching mode generation and optimal estimation with application to skid-steering. Automatica 47(1):50–64
31. Egerstedt M, Wardi Y, Delmotte F (2003) Optimal control of switching times in switched dynamical systems. In: IEEE conference on decision and control, Maui, Dec 2003
32. Egerstedt M, Wardi Y, Axelsson H (2003) Optimal control of switching times in hybrid systems. In: IEEE methods and models in automation and robotics, Miedzyzdroje
33. Johnson E, Murphey TD (2009) Scalable variational integrators for constrained mechanical systems in generalized coordinates. IEEE Trans Robot 25(6):1249–1261
34. Nichols K, Murphey TD (2008) Variational integrators for constrained cables. In: IEEE international conference on automation science and engineering (CASE), pp 802–807
35. Kharevych L, Yang W, Tong Y, Kanso E, Marsden JE, Schroder P, Desbrun M (2006) Geometric, variational integrators for computer animation. Eurographics/ACM SIGGRAPH symposium on computer, animation
36. Lew A, Marsden JE, Ortiz M, West M (2004) Variational time integrators. Int J Numer Meth Eng 60:153–212
37. Lew A, Marsden JE, Ortiz M, West M (2004) An overview of variational integrators. In: Finite element methods: 1970's and beyond, pp 98–115
38. West M (2004) Variational integrators. California Institute of Technology Thesis
39. Lew A, Marsden JE, Ortiz M, West M (2003) Asynchronous variational integrators. Arch Ration Mech Anal 167:85–146
40. Marsen JE, West M (2001) Discrete mechanics and variational integrators. Acta Numerica 10:357–514
41. Anderson B, Moore J (1971) Linear optimal control. Prentice Hall Inc, Englewood Cliffs. Upper Saddle River, New Jersey
42. Snyder K, Murphey TD (2010) Second-order DMOC using projections. In IEEE international conference on decision and control (CDC), 2010
43. Paynter B (2009) Robodinos: what could possibly go wrong? Wired magazine. Issue 17.08

Chapter 6
The Artistic Geometry of Consensus Protocols

Panagiotis Tsiotras and Luis Ignacio Reyes Castro

Mighty is geometry; joined with art, resistless.
Euripides (480–406 BC)

6.1 The Role of Geometric Patterns in the History of Art

The use of geometric patterns in art has a long history. The geometric period of ancient Greek art (ca. 900–700 BC) is characterized by the extensive use of geometric motifs, mainly on vase and amphorae painting. These decorative motifs (meanders, triangles, circles, etc.) extend horizontally in multiple bands about the vase circumference, and they exhibit central and translational symmetry [7, 18, 45]. Elaborate symmetric geometric patterns also appear extensively in Islamic art, largely due to their aniconic quality [10, 11]. Influenced by previous classical Greek, Roman, and Sasasian works, and fueled by the intellectual contributions of Islamic mathematicians, astronomers, and scientists of the time, Islamic artists created this unique new style, which is characterized by repeated combinations and duplications of simple geometric forms (such as circles and the squares), arranged in intricate, interlaced geometric ornamentations whose complexity is ever increasing, offering the possibility of infinite growth [15]. The exploration of infinity and symmetric growth has also been explored by many subsequent artists. The Dutch graphic artist M. C. Escher (1898–1972) is most famous for his exploration of infinity and of his creations of

P. Tsiotras (✉)
Daniel Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0150, USA
e-mail: tsiotras@gatech.edu

L. I. R. Castro
Currently a graduate student at the Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

A. LaViers and M. Egerstedt (eds.), *Controls and Art*,
DOI: 10.1007/978-3-319-03904-6_6,

Fig. 6.1 The Irish Trinity Knot (triquetra) consisting of the intersection of three trochoidal-like curves, appears often in medieval Celtic art [38]

impossible images in graphic art [43]. His work on infinite tessellations involving repetitive symmetric patterns on the plane has apparently been heavily influenced by the work of mathematician George Polya (1887–1985) on plane symmetry groups. Escher studied Polya's 17 plane symmetry groups, which led him to develop a mathematical approach to expressions of symmetry, which he later incorporated in his art works. Escher's works primarily exploit planar symmetry groups, otherwise known as "wallpaper designs."

In this chapter we will be dealing with similar geometric planar patterns, which, however, exhibit circular/point symmetry as opposed to regular horizontal/vertical plane symmetry. The interested reader is referred to the seminal work of H. Weyl (1885–1955) [54] for an in-depth discussion on symmetry and symmetry groups. Reference [9] also provides a nice classification of all symmetry groups in one, two, and three dimensions.

We will investigate the generation of highly stylized geometric patterns morphed by repeated repetitions of trochoidal curves on flat or curved surfaces, such as those shown in Fig. 6.2. Trochoidal curves have been in the center of study by several scientists, and are the main focus of our work. They include cycloids, ellipses (and circles), epitrochoids, hypotrochoids, as well as cardioids, astroids, limaçons, and all polar coordinate roses [16]. Although ancient Greeks had discovered trochoids (for instance, the epicycloid had been used by ancient Greeks to describing the movement of the planets long before N. Copernicus (1473–1543) and J. Kepler (1571–1630) established the correct view of heliocentric planet movement in the heavens), the use of trochoids in art seems to have been limited, except in architecture. For example, the Persian astronomer and mathematician Nasir Al-Din al-Tusi (1201–1274) studied the two-cusped hypocycloid [34].

Trochoidal-like curves also appear in Western medieval art. The triquetra symbol (also known as the Irish Trinity Knot), which is usually illustrated as the intersection of three vesicae piscis shapes, appears often in Celtic artwork (Fig. 6.1). This symbol is also prominently depicted in the U-937 runestone, which is one of the four Funbo Runestones, and has been attributed to the 11th century runemaster Fot [41].

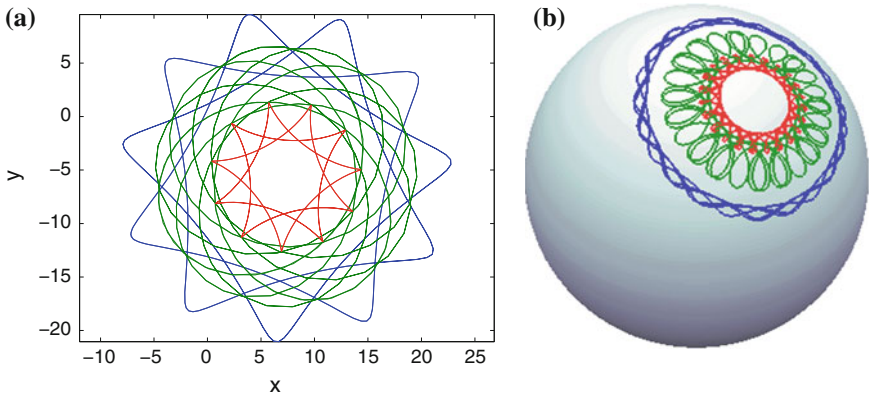


Fig. 6.2 The trochoidal family of curves, which includes cycloids, epitrochoids, hypotrochoids, cardioids, astroids, and limaçons, is the main focus of our work. Trochoids have been studied by several scientists and artists. **a** Interlace of three *trochoidal curves* on the *plane*. **b** Interlace of three *trochoidal curves* on the *sphere*

It is the German artist of the sixteenth century Albrecht Dürer (1471–1528), however, who is credited to be the first to have incorporated trochoids (specifically, hypotrochoidal curves) in his art [47]. He introduced the hypocycloid curve along with the more general family of trochoid curves, in his 1,525 four-volume geometry treatise *The Art of Measurement with Compass and Straightedge*. After Dürer, purely trochoidal curves seem to have been absent from the artistic world up until the invention of the *spirograph* by English engineer Denys Fisher (1918–2002), introduced during the 1965 Nuremberg International Toy Fair. The introduction of the spirograph has created a momentum in the use of trochoidal curves in the Pop Art, Op Art, and Psychedelic Art movements since the 1960s, with several American and European artists incorporating them into their works. Perhaps the first artist in this new line of trochoid admirers is Seattle-based painter Jeffrey Simmons (1968–), who conceived a seven-painting collection titled *Trochoid* [46]. These paintings rely on large hypotrochoids as central features, and were produced with the use of a special-purpose device constructed by the artist himself. English artists Ian Dawson (1969–) and Lesley Halliwell (1965–), also make use of trochoids, although using a very different technique [8, 17]. In their work, they create large colorful shapeless compositions by putting together large numbers of hypotrochoids generated using spirographs. The same technique has also been utilized by Pittsburgh-based illustrator David Pohl, who has used the spirograph as a tool to explore the recurring theme of repetition as a means to illustrate the cyclical nature of life [37]. The American architect Louis Kahn (1901–1974) has also used cycloids in his design of the Kimbell Art Museum [35].

6.2 A Brief of Consensus Protocols

In this chapter, we show how elaborate patterns that are closely related to trochoidal curves can be generated as the paths followed by a team of interacting agents moving on the plane. In the literature of multi-agent control systems, problems involving the coordination of a team of agents such as flocking, swarming, etc., are referred to as consensus problems, and the underlying control strategies enforcing team coordination are commonly referred to as consensus protocols.

Consensus problems have been extensively used for many years in the area of distributed computing and management science. Their recent popularity in the controls community stems from their utilization in formulating and solving a variety of multi-agent, mobile network problems [32, 40]. In this chapter, we propose a generalization of the standard consensus algorithm used widely in the literature [12, 27, 31], and we show how this algorithm can be utilized to generate intricate geometrical patterns for the ensuing agent paths. Using minimal assumptions, the proposed feedback control is able to generate geometric patterns for the agent trajectories that go beyond formation-type geometric models, which deal mainly with identical agents in cycle pursuit [21, 25, 36, 49].

Our inspiration comes from gyroscopic control strategies used in the wheeled robotics community [52] for obstacle avoidance. Since the proposed control law introduces circulation in the underlying vector field, it cannot be derived from a scalar potential, and hence it does not belong to the family of consensus control laws that are gradient-based. As an added benefit of the proposed extension, it is shown that this control law results in consensus points that lie outside the convex hull of the initial positions of the agents. This may be useful for obstacle avoidance and/or consensus with deception, for instance.

As a direct consequence of the proposed extended consensus protocol, in the second part of this chapter we particularize this control law to the case of periodic and quasi-periodic pattern generation, and show how it can be used to generate elaborate, esthetically beautiful patterns.

6.3 Motivating Example

In order to demonstrate the main idea, we start with the simplest of cases, namely, two agents ($N = 2$) in the plane. The extension to the case of an arbitrary number of agents follows readily from this case and it is given in the next section, along with the stability analysis of the overall system with all interacting agents. To this end, assume a given global coordinate frame $\mathcal{E}$ with origin O and two agents at locations $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$, respectively. The kinematic equation for each agent is given by

$$\dot{\boldsymbol{r}}_i = \boldsymbol{u}_i, \qquad i = 1, 2. \tag{6.1}$$

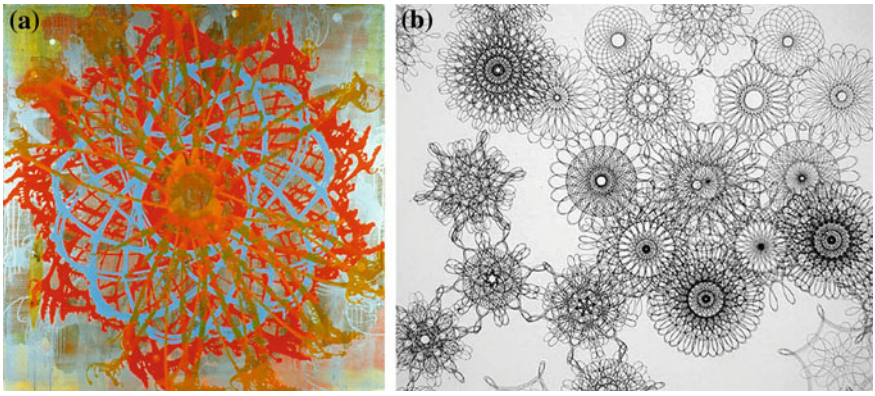


Fig. 6.3 Several modern artists have experimented with the use of trochoidal curves in their work. **a** Warmth of the Sun (1999). Oil and alkyd on canvas by J. Simmons. (Reprinted with permission [46]). **b** A80 (2000). Ink on gesso on primed board by I. Dawson. (Reprinted with permission [8])

We assume that only the *relative* distance $\boldsymbol{r}_{12} = \boldsymbol{r}_1 - \boldsymbol{r}_2$ is known to agent no. 1 and, similarly, only the relative distance $\boldsymbol{r}_{21} = -\boldsymbol{r}_{12}$ is available to agent no. 2. It can be easily shown [27] that the control law

$$\boldsymbol{u}_1 = -\gamma_1 \boldsymbol{r}_{12}, \qquad \boldsymbol{u}_2 = -\gamma_2 \boldsymbol{r}_{21}, \qquad \gamma_1 + \gamma_2 > 0 \tag{6.2}$$

achieves consensus. That is, the distance between the two agents will tend to be zero as the time progresses. Furthermore, with the control law in (6.2), the two agents will meet somewhere along the line segment initially connecting $\boldsymbol{r}_1(0)$ and $\boldsymbol{r}_2(0)$. Our first objective is to modify (6.2) in order to allow convergence of the agents to points that do not necessarily belong to the line segment (in general, the convex hull) defined by the initial position vectors.

The main observation here is that the control law (6.2) does not make use of all available geometric information to each agent. For instance, agent no. 1 knows not only the vector $\boldsymbol{r}_{12}$ but also all vectors (directions) *perpendicular* to $\boldsymbol{r}_{12}$, which can then be used in a feedback strategy. Similarly, for agent no. 2, this additional information in the control law, inferred from—but distinct than—the relative position vector between the agents, can lead to more flexibility for trajectory design. To this end, let $\boldsymbol{q}_{12}$ and $\boldsymbol{q}_{21}$ be such that $\boldsymbol{q}_{12} \cdot \boldsymbol{r}_{12} = \boldsymbol{q}_{21} \cdot \boldsymbol{r}_{21} = 0$, and assume the following control laws[1]

$$\boldsymbol{u}_1 = -\gamma_1 \boldsymbol{r}_{12} + \beta_1 \boldsymbol{q}_{12}, \qquad \boldsymbol{u}_2 = -\gamma_2 \boldsymbol{r}_{21} + \beta_2 \boldsymbol{q}_{21} \tag{6.3}$$

Later, it will be shown that this control law also achieves consensus for $\gamma_1 + \gamma_2 > 0$ and $\beta_1, \beta_2 \in \mathbb{R}$.

[1] Owing to the freedom in choosing $\boldsymbol{q}_{12}$ and $\boldsymbol{q}_{21}$, we define a "position orientation" such that $\boldsymbol{r}_{12} \times \boldsymbol{q}_{12} = \boldsymbol{r}_{21} \times \boldsymbol{q}_{21}$.

In preparation for the general case, let us now introduce coordinates, with respect to a global frame $\mathcal{E}$, leading to $[\boldsymbol{r}_i]_{\mathcal{E}} \triangleq x_i \in \mathbb{R}^2$, $(i = 1, 2)$ and $[\boldsymbol{r}_{12}]_{\mathcal{E}} = [\boldsymbol{r}_1]_{\mathcal{E}} - [\boldsymbol{r}_2]_{\mathcal{E}} = x_1 - x_2$. Let the error vector $z \in \mathbb{R}^2$ of the relative distance between the two agents be

$$z \triangleq x_1 - x_2 = d_{11}x_1 + d_{21}x_2 = (D^{\mathsf{T}} \otimes I_2)x, \tag{6.4}$$

where $D = \begin{bmatrix} 1 & -1 \end{bmatrix}^{\mathsf{T}}$ and where $x = [x_1^{\mathsf{T}},\ x_2^{\mathsf{T}}]^{\mathsf{T}} \in \mathbb{R}^4$. Furthermore, let $[\boldsymbol{q}_{12}]_{\mathcal{E}} \triangleq p = Sz$ where S is the skew symmetric matrix

$$S = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}. \tag{6.5}$$

It is clear that $p^{\mathsf{T}}z = z^{\mathsf{T}}p = 0$. It can then be easily seen that the control law (6.3) can be written compactly, as follows

$$\begin{aligned} u &= -(\Gamma \otimes I_2)(D \otimes I_2)z + (B \otimes I_2)(D \otimes I_2)Sz \\ &= -(\Gamma D \otimes I_2)z + (BD \otimes S)z, \end{aligned} \tag{6.6}$$

where $u = [u_1^{\mathsf{T}},\ u_2^{\mathsf{T}}]^{\mathsf{T}} \in \mathbb{R}^4$ and $\Gamma = \operatorname{diag}(\gamma_1, \gamma_2)$ and $B = \operatorname{diag}(\beta_1, \beta_2)$. From (6.4) it follows that the error equation is given by

$$\begin{aligned} \dot{z} &= (D^{\mathsf{T}} \otimes I_2)\dot{x} = (D^{\mathsf{T}} \otimes I_2)u \\ &= -(D^{\mathsf{T}} \otimes I_2)(\Gamma \otimes I_2)(D \otimes I_2)z + (D^{\mathsf{T}} \otimes I_2)(B \otimes I_2)(D \otimes I_2)Sz \\ &= -\Big(\big(D^{\mathsf{T}}\Gamma D\big) \otimes I_2\Big)z + \Big(\big(D^{\mathsf{T}}BD\big) \otimes S\Big)z. \end{aligned}$$

Stability is determined by the eigenvalues of the matrix $A_{\mathrm{CL}} = -\big((D^{\mathsf{T}}\Gamma D) \otimes I_2\big) + \big((D^{\mathsf{T}}BD) \otimes S\big)$. A simple calculation shows that $\operatorname{spec}(A_{\mathrm{CL}}) = \{-(\gamma_1 + \gamma_2) \pm \mathrm{i}(\beta_1 + \beta_2)\}$. Hence consensus is achieved asymptotically as long as $\gamma_1 + \gamma_2 > 0$. The "classical" consensus control law (6.2) corresponds to the case when $\beta_1 = \beta_2 = 0$. When $B \neq 0$ stability is still maintained, however, the transient response is different. Furthermore, the point where consensus is achieved can be selected to lie outside the line segment connecting $x_1(0)$ and $x_2(0)$ by a proper choice of the gains β_1 and β_2. This is demonstrated in Fig. 6.4 where the result of a simulation with the data $x_1(0) = (-1, 1)^{\mathsf{T}}$, $x_2(0) = (2, 3)^{\mathsf{T}}$, $\Gamma = \operatorname{diag}(0.1, 1)$, $B = \operatorname{diag}(-0.5, 2)$ is shown. For this example the two agents meet at the point with coordinates $(-2, 1)$.

6.4 Extension to N Agents in the Plane

For the general case, consider N agents in the plane. Assume that their location is given by the state variables $x_i \in \mathbb{R}^2$ for $i = 1, \ldots, N$, expressed in the same, common global frame $\mathcal{E}$, satisfying the differential equations

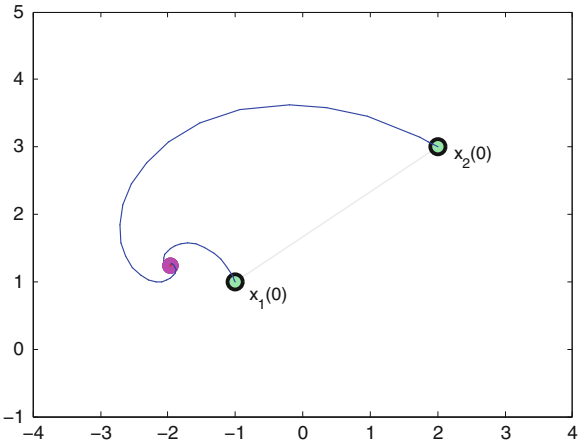


Fig. 6.4 Numerical example with "skew-symmetric" feedback. The skew-symmetric term creates a vector field with circulation

$$\dot{x}_i = u_i, \qquad i = 1, \ldots, N. \tag{6.7}$$

To the N agents we associate a graph $\mathcal{G}$ that describes the communication topology between the agents. That is, $\mathcal{G}$ has N nodes and M edges (links), with each edge denoting knowledge of the relative position between the corresponding agents. We can define the incidence matrix $D \in \mathbb{R}^{N\times M}$ with elements as follows [1]. We assign $d_{ij} = +1$ (-1) if the ith node is the head (tail) of jth edge, and $d_{ij} = 0$ otherwise. If the ith agent is a neighbor with the jth agent, then they are connected by an edge, and we have the difference (error) variable

$$z_k = \sum_{\ell=1}^{N} d_{\ell k} x_\ell = \begin{cases} x_i - x_j, & \text{if } i \text{ is the head,} \\ x_j - x_i, & \text{if } j \text{ is the head,} \end{cases} \tag{6.8}$$

where $z_k \in \mathbb{R}^2$ for $k = 1, \ldots, M$. If the columns of D are linearly independent, that is, if the graph does not contain cycles, then the vectors z_k are linearly independent [1]. Note also that the graph is connected if and only if rank $D = N - 1$ [14, 31]. Introducing the stack vector $x = \left[x_1^\mathsf{T} \cdots x_N^\mathsf{T}\right]^\mathsf{T} \in \mathbb{R}^{2N}$, the state equations (6.7) can be written compactly as

$$\dot{x} = u, \tag{6.9}$$

where $u = \left[u_1^\mathsf{T} \cdots u_N^\mathsf{T}\right]^\mathsf{T} \in \mathbb{R}^{2N}$. Following (6.6), we propose the control law

$$u = -(\Gamma D \otimes I_2)z + (BD \otimes S)z, \tag{6.10}$$

where $z = \left[z_1^\mathsf{T} \cdots z_M^\mathsf{T}\right]^\mathsf{T} \in \mathbb{R}^{2M}$, and where $\Gamma = \text{diag}(\gamma_1, \ldots, \gamma_N)$ and $B = \text{diag}(\beta_1, \ldots, \beta_N)$. The standard consensus algorithm results as a special case of (6.10) where $B = 0$.

Convergence Analysis

From (6.8) it can be easily shown that the error vector z can be written compactly as follows

$$z = (D^\mathsf{T} \otimes I_2)x. \tag{6.11}$$

From (6.10) the differential equation for x is then given by

$$\begin{aligned} \dot{x} &= -(\Gamma D \otimes I_2)(D^\mathsf{T} \otimes I_2)x + (BD \otimes S)(D^\mathsf{T} \otimes I_2)x \\ &= -\big((\Gamma DD^\mathsf{T}) \otimes I_2 - (BDD^\mathsf{T}) \otimes S\big)x \\ &= -\big((\Gamma L) \otimes I_2 - (BL) \otimes S\big)x, \end{aligned} \tag{6.12}$$

where $L \triangleq DD^\mathsf{T} \in \mathbb{R}^{N\times N}$ is the *graph Laplacian* [27]. Let $\mathbf{1}_N \triangleq (1, 1, \ldots, 1)^\mathsf{T} \in \mathbb{R}^N$ denote the N-dimensional column vector of ones, and recall that $L\mathbf{1}_N = 0$ [14, 27]. For any $v \in \mathbb{R}^2$ we have that $\big((\Gamma L)\otimes I_2 - (BL)\otimes S\big)(\mathbf{1}_N \otimes v) = (\Gamma L\mathbf{1}_N)\otimes v - (BL\mathbf{1}_N)\otimes(Sv) = 0$. It follows that the vector $\mathbf{1}_N \otimes v$ spans the null space of the matrix in (6.12). The equilibrium point $\bar{x}_\infty$ of the linear differential equation (6.12) therefore satisfies the condition $\bar{x}_\infty \triangleq \lim_{t\to\infty} x(t) = \mathbf{1}_N \otimes x_\infty$ for some $x_\infty \in \mathbb{R}^2$, from which it follows that $\lim_{t\to\infty} x_1(t) = \lim_{t\to\infty} x_2(t) = \cdots = \lim_{t\to\infty} x_N(t) = x_\infty$, thus achieving consensus.

Let the coordinates of the final consensus point be $x_\infty = [\mathsf{x}_\infty\ \mathsf{y}_\infty]^\mathsf{T} \in \mathbb{R}^2$. We have the following proposition.

Proposition 1 [51]. *Let* $v_1, v_2 \in \mathbb{R}^{2N}$ *be such that* $\text{span}\{v_1, v_2\} = \mathcal{R}^\perp\big((\Gamma L) \otimes I_2 - (BL) \otimes S\big)$. *The final rendezvous point is given by*

$$x_\infty = \begin{bmatrix} \mathsf{x}_\infty \\ \mathsf{y}_\infty \end{bmatrix} = \begin{bmatrix} v_1^\mathsf{T}(\mathbf{1}_N \otimes I_2) \\ v_2^\mathsf{T}(\mathbf{1}_N \otimes I_2) \end{bmatrix}^{-1} \begin{bmatrix} v_1^\mathsf{T} x(0) \\ v_2^\mathsf{T} x(0) \end{bmatrix}. \tag{6.13}$$

If $\Gamma = 0$ there is not "rendezvous" point. Instead, the agents follow closed trajectories centered around the point given by equation (6.13).

6.5 Periodic and Quasi-Periodic Trajectories

Given an interconnection topology, the particular choices of the gain matrices Γ and B can be used to generate specific trajectory patterns for the ensuing agent paths. Since we are mainly interested in periodic or quasi-periodic trajectories, next we restrict the discussion to the case $\Gamma = 0$. By letting $\Gamma = 0$ in (6.10) the control law becomes

$$u = (BD \otimes S)z, \tag{6.14}$$

and the closed-loop system reduces to

$$\dot{x} = ((BL) \otimes S)\, x. \tag{6.15}$$

The shape and frequencies of the resulting paths are therefore determined by the eigenvalues and eigenvectors of the matrix $(BL) \otimes S$. Recall from the properties of the Kronecker product [6] that the eigenvalues of the matrix $(BL) \otimes S$ are of the form $\lambda\mu$, where $\lambda \in \text{spec}(BL)$ and $\mu \in \text{spec}\, S$. Additionally, the corresponding eigenvectors are of the form $\mathsf{v} \otimes \mathsf{u}$ where $\mathsf{v} \in \mathbb{C}^3$ is the eigenvector of the matrix BL associated with λ and $\mathsf{u} \in \mathbb{C}^2$ is the eigenvector of the matrix S associated with μ. Since $\det(\lambda I_N - BL) = \det(\lambda I_N - BDD^{\mathsf{T}}) = \det(\lambda I_M - D^{\mathsf{T}}BD)$ it follows that the nonzero eigenvalues of the matrix BL coincide with the nonzero eigenvalues of $D^{\mathsf{T}}BD$. Because the latter matrix is symmetric, all eigenvalues of BL are real. Consequently, all eigenvalues of $(BL)\otimes S$ lie on the imaginary axis. It follows that the solutions of (6.15) consist, in general, of a superposition of sine and cosine functions, perhaps multiplied by polynomials in t (in the case of multiple eigenvalues).

Let $BL = VJV^{-1}$ be the spectral decomposition of the matrix BL. It can be easily shown that

$$\mathrm{e}^{((BL)\otimes S)t} = (V \otimes I_2)\, \mathrm{e}^{(J\otimes S)t}\, (V^{-1} \otimes I_2). \tag{6.16}$$

The spectral decomposition of the matrix BL thus provides all information needed to investigate the nature of the solutions of (6.15). In fact, additional information can be gathered owing to the special structure of the state matrix in (6.15).

Lemma 1. *Let A be an $n \times n$ square matrix and let S be the 2×2 skew-symmetric matrix given in (6.5). Then*

$$\mathrm{e}^{A\otimes S} = \cos A \otimes I_2 + \sin A \otimes S. \tag{6.17}$$

Proof. Notice that $S^{2k} = (-1)^k I_2$ and $S^{2k+1} = (-1)^k S$, $k = 0, 1, 2, \ldots$ and recall that

$$\mathrm{e}^{A\otimes S} = \sum_{k=0}^{\infty} \frac{1}{k!} (A \otimes S)^k.$$

The right-hand side of the previous equation can be expanded as follows

$$\begin{aligned}
&\sum_{k=0}^{\infty} \frac{1}{(2k)!}\big(A \otimes S\big)^{2k} + \sum_{k=0}^{\infty} \frac{1}{(2k+1)!}\big(A \otimes S\big)^{2k+1} \\
&= \sum_{k=0}^{\infty} \frac{1}{(2k)!}\big(A^{2k} \otimes S^{2k}\big) + \sum_{k=0}^{\infty} \frac{1}{(2k+1)!}\big(A^{2k+1} \otimes S^{2k+1}\big) \\
&= \Big(\sum_{k=0}^{\infty} \frac{(-1)^k}{(2k)!} A^{2k}\Big) \otimes I_2 + \Big(\sum_{k=0}^{\infty} \frac{(-1)^k}{(2k+1)!} A^{2k+1}\Big) \otimes S.
\end{aligned}$$

Making use of the fact that for a square matrix A,

$$\cos A = \sum_{k=0}^{\infty} \frac{(-1)^k}{(2k)!} A^{2k}, \quad \sin A = \sum_{k=0}^{\infty} \frac{(-1)^k}{(2k+1)!} A^{2k+1},$$

the result of the lemma follows immediately.

We therefore have the following proposition.

Proposition 2 *The solution of* (6.15) *is given by*

$$\begin{aligned}
x(t) &= \big(\cos(BLt) \otimes I_2 + \sin(BLt) \otimes S\big)x(0), \\
&= (V \otimes I_2)\big(\cos(Jt) \otimes I_2 + \sin(Jt) \otimes S\big)(V^{-1} \otimes I_2)x(0),
\end{aligned} \tag{6.18}$$

for all $t \geq 0$ *and all* $x(0) \in \mathbb{R}^{2N}$.

The structure of the state matrix in (6.15) (e.g., its eigenvalues and eigenvectors) thus can provide a great deal of information regarding the paths followed by the agents in the Cartesian coordinate frame, as well as the relative location of the agents on these paths (i.e., their relative phasing). For instance, one can ensure that the agent trajectories either form closed paths with given phasing, or they form a dense set of trajectories, ensuring that almost every point in a given region will be visited at least once by one or more agents. Such orbits could be desirable, for instance, in surveillance or area coverage applications (see Fig. 6.8). Moreover, as shown in the next section, these orbits are also esthetically appealing. In that respect, geometric beauty serves as a functional element for the solution of meaningful engineering problems.

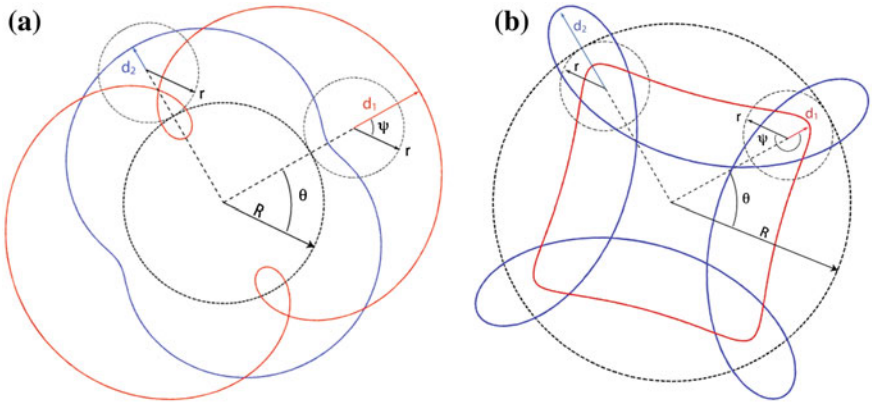


Fig. 6.5 Representative examples of epitrochoids and hypotrochoid curves. **a** *Epitrochoid curves* The *blue curve* has $d < r$, while the second one has $d > r$. Both epitrochoids have $R = 4$, $r = 2$ (hence $k = 2$). **b** *Hypotrochoid curves* The *blue curve* has $d > r$, while the second one has $d < r$. Both epitrochoids have $R = 6$, $r = 1.5$ (hence $k = 4$)

6.6 Orbit Pattern Generation

In this section, we show that the solutions of (6.15) result in elaborate geometric trochoidal patterns. As is seen by (6.18), the solutions depend on the gain matrix B, the Laplacian matrix L that encodes the connectivity, as well as the initial conditions $x(0)$.

6.6.1 A Family of Achievable Paths

The solutions in (6.18) fall in the general class of *trochoidal* curves. An *epitrochoid* curve is generated by a point P attached at a radial distance d from the center of a circle of radius r, which is rolling without slipping around a circular track of radius R; see Fig. 6.5a. The distance d can be smaller, equal, or greater than the radius r of the rolling circle. The ratio of the circular two tracks $k = R/r$ indicates the number of points at which the agent is closest to the center of the circular track. These are referred to as *crests*. In the special case when $r = d$, the curve becomes an *epicycloid* with k *cusps*; at these points, the curve is not differentiable. Note that ellipsoidal paths correspond to the case when $k = 0$. A *hypotrochoid* is generated by a point P attached at a distance d from the center of a circle of radius r, which rolls inside a circle of radius R; see Fig. 6.5b. Again, the distance d can be smaller, equal, or greater than the radius r of the rolling circle; this radius, however, cannot exceed that of the circle R.

As mentioned in the introduction, the trochoidal family of curves is very rich and includes many of the well-known curves such as ellipses and circles, (epi/hypo) cycloids, cardioids, limaçons, etc. The most well-known example of a trochoid curve

is the cycloid—the "courbe merveilleuse" of M. Chasles (1793–1880). The cycloid often appears as the solution to many problems in mathematics, physics. For instance, Bernoulli's brachystochrone problem in the Calculus of Variations, and the paths followed by charged particles in crossed electric and magnetic fields turn out to be cycloids. Because of its recurring appearance as the solution of many problems the cycloid has been the center of investigation by several mathematicians, and its study has not escaped controversy. It fact, owing to the many disputes it provoked between mathematicians over the centuries it has been called the "Helen of geometers" [5], in reference to the beautiful Helen of Troy who caused many quarrels among men, and whose abduction by Paris, the son of Priam King of Troy, caused the Trojan War.

In the next section, we investigate a few interesting cases of trochoids resulting from the solution of (6.18).

6.6.2 Illustrative Example: Three Agents

In this section, we investigate in greater detail the simple nontrivial case, namely, three agents in the plane ($N = 3$), connected either in a path graph ($M = 2$) or a complete graph ($M = 3$). For a path graph interconnection the incidence matrix is given by

$$D = \begin{bmatrix} -1 & 0 \\ 1 & -1 \\ 0 & 1 \end{bmatrix}. \tag{6.19}$$

A straightforward calculation shows that the two nonzero eigenvalues of the matrix BL for this case are given by

$$\frac{\beta_1}{2} + \beta_2 + \frac{\beta_3}{2} \pm \frac{\sqrt{\beta_1^2 - 2\beta_1\beta_3 + 4\beta_2^2 + \beta_3^2}}{2}.$$

For the complete graph the incidence matrix is given by

$$D = \begin{bmatrix} -1 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix}. \tag{6.20}$$

The nonzero eigenvalues of the matrix BL for this case are given by

$$\beta_1 + \beta_2 + \beta_3 \pm \sqrt{\beta_1^2 + \beta_2^2 + \beta_3^2 - \beta_1\beta_2 - \beta_2\beta_3 - \beta_3\beta_1}.$$

The ratio of the two nonzero eigenvalues is equal to $k + 1$ for an epitrochoid or $k - 1$ for a hypotrochoid. Note that if k turns out to be an irrational number, then the

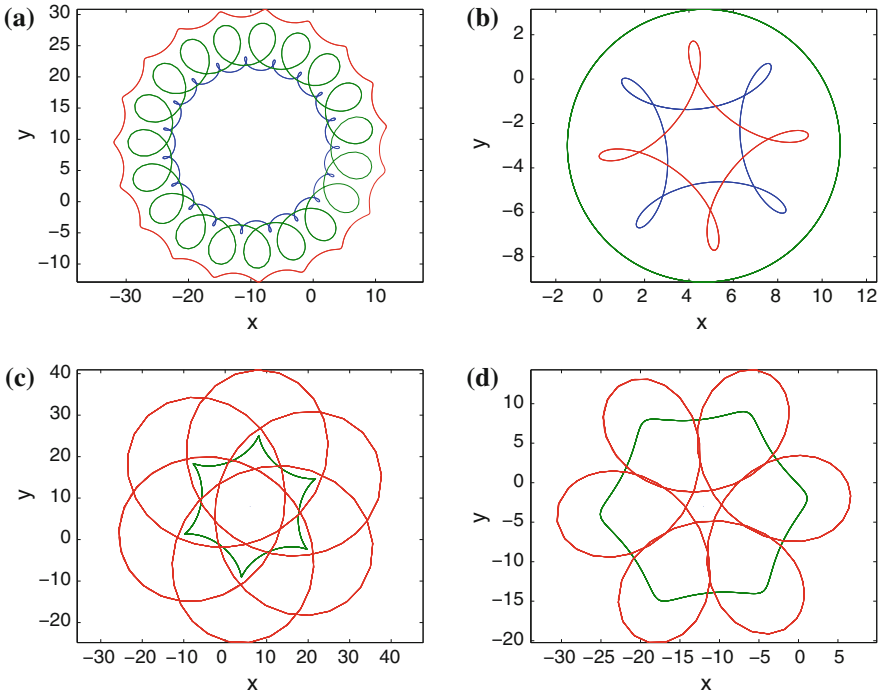


Fig. 6.6 Orbits with three agents using the extended consensus protocol; path graph. **a** $B = \text{diag}(1, 2.0034, -1)$. **b** $B = \text{diag}(-1, 2, -1)$. **c** and **d** $B = \text{diag}(0, 1, -6.5933)$

number of crests is infinite, which means that the curve is not closed; instead, the trajectories form a dense subset of the space [19]. See Figs. 6.6 and 6.7.

An orbit redesign can yield periodic orbits of a particular shape that can be used for coordinated, distributed surveillance, and perimeter monitoring applications; see, for instance, Fig. 6.8. Such an orbit redesign may require a complete interconnection topology [51].

An interesting case occurs when the closed-loop system has two zero eigenvalues at the origin. In this case the trajectories exhibit secular motion. Figure 6.9a shows the trajectories when $B = \text{diag}(0.5, -1, -1)$. It can be easily verified that in this case the relative orbits for the three agents are all circles; see Fig. 6.9b.

There is of course a plethora of possibilities to explore, and one can only imagine the different ways to use the flexibility offered by this serendipitous marriage of art, geometry, and multi-agent control system for solving meaningful, real-world engineering problems.

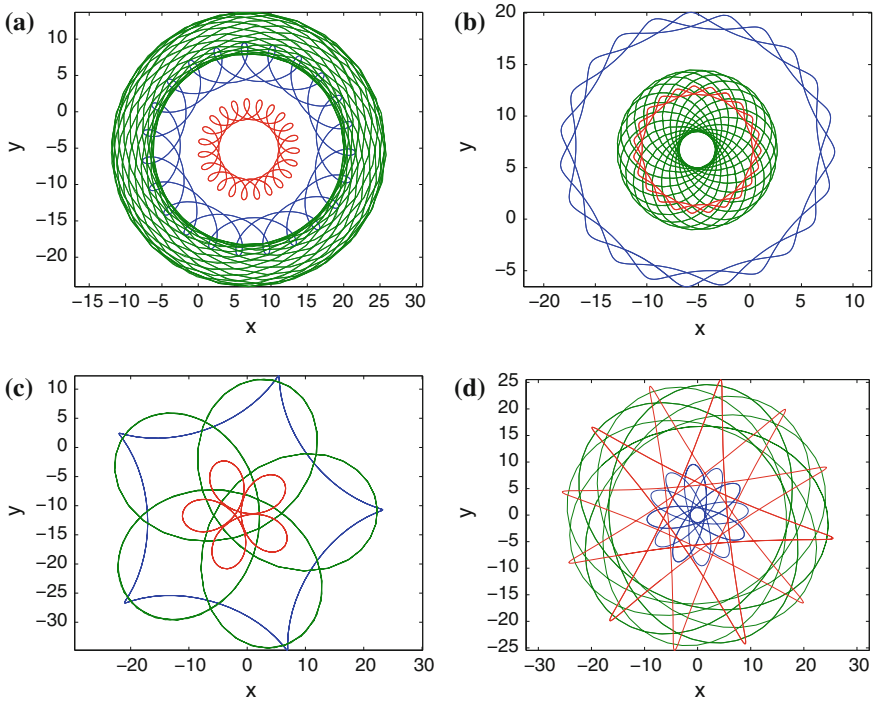


Fig. 6.7 Orbits with three agents using the extended consensus protocol; path graph. **a** and **b** $B = \mathrm{diag}(1, -2.9054, 0.5)$. **c** $B = \mathrm{diag}(1, -1, -0.5)$. **d** $B = \mathrm{diag}(-0.75, 1.67259, -2)$

6.7 A Gallery of Orbits

Using the extended consensus protocols one can clearly generate a myriad of beautiful geometric patterns, by changing the gain matrix B and by choosing a suitable graph Laplacian L in (6.15). Figures 6.6, 6.7, 6.8, 6.9, 6.10, 6.11 and 6.12 provide a glimpse on the plethora and variety of geometric patterns generated using the consensus control law in (6.15) for the case of three and four agents on the plane. We urge the reader to try his/her own skills at generating visually pleasing curves using Eq. (6.15).

In addition to monitoring and surveillance applications already mentioned, these and similar geometric patterns, can also be used in all cases where the resulting motion of a group of agents is to be determined distributively, solely by inter-agent interactions. For instance, they could serve as periodic motion primitives for executing elaborate choreographic patterns for human dancers or small autonomous robotic vehicles, as it is done, for example in the work of Schoelling et al. [44] and Leonard et al. [22] elsewhere in this book. Indeed, one can envision situations where swarm dance patterns—accompanied perhaps by music—can evolve to agents paths resembling those shown in Figs. 6.6, 6.7, 6.8, 6.9, 6.10, 6.11, and 6.12. One needs only to impose the correct communication topology in the underlying graph and the

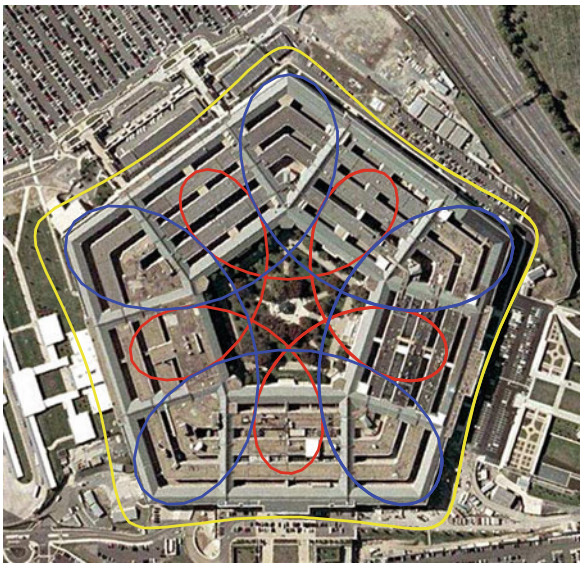


Fig. 6.8 Trochoidal paths that could be used by three agents to patrol a *pentagon-shaped* area. Satellite image courtesy of USGS [39]

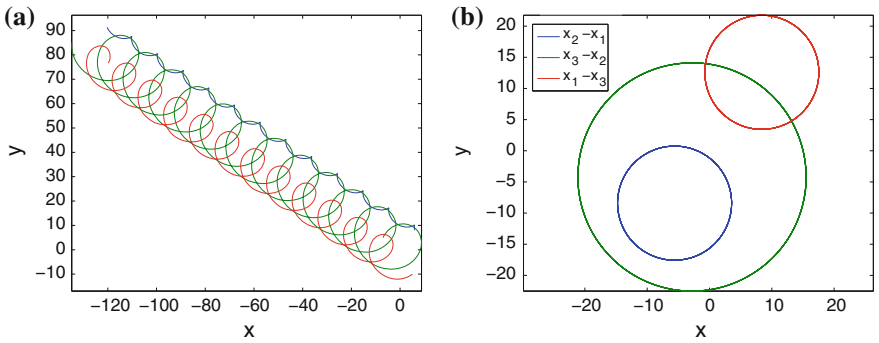


Fig. 6.9 $B = \text{diag}(0.5, -1, -1)$ and initial conditions $x_1(0) = (6, 8)$, $x_2(0) = (-7, 5)$, $x_3(0) = (5, -10)$ (path graph interconnection). The figure on the *right* shows the relative orbits

correct gain weights. Although such an idea may seem intuitive and appealing, we should offer a word of caution: in all our developments so far we have not taken into consideration the case when two agents happen to be at the same location at the same time. That is, collision avoidance is not built-in a priori into (6.15) and, depending on the size of the agents with respect to the size of these orbits, may indeed be a problem during implementation. Along these lines, a much needed future research direction is the design of such geometric patterns for dance or flock formation (or other more

engineering oriented applications such as patrolling or surveillance) while—at the same time—incorporating collision avoidance guarantees.

6.8 Extensions to Pattern Generation on Curved Surfaces

The previous methodology can be easily extended to generate similar intricate trochoidal paths on any two-dimensional manifold. Although the classical consensus protocol has been extended to the case of agents moving on a sphere [30] or a general manifold [42, 50], in this work we will follow an alternative—more direct approach—to generate trochoidal curves on a sphere, by taking advantage of the fact that a two-dimensional manifold is a surface that "locally" looks like a two-dimensional plane. Given therefore a two-dimensional manifold $\mathcal{M}$ (in this case, a sphere, $\mathcal{M} = \mathbb{S}^2$) we can define local coordinates $y_1 = \phi_1(q)$ and $y_2 = \phi_2(q)$, where $q \in \mathcal{M}$, and where $\phi \triangleq (\phi_1, \phi_2) : \mathcal{M} \mapsto \mathbb{R}^2$ is a homeomorphism between an open subset of $\mathcal{M}$ and an open subset of $\mathbb{R}^2$.

Using the spherical coordinates $\phi \in [-\pi, \pi]$ (azimuth) and $\theta \in [-\pi/2, \pi/2]$ (elevation), the equations of any point of the orbit on the unit sphere is given by

$$\mathsf{x} = \cos\theta \cos\phi, \qquad \mathsf{y} = \cos\theta \sin\phi, \qquad \mathsf{z} = \sin\theta. \tag{6.21}$$

Assume now that the equations of motion of each agent on the sphere obey the equations (6.7), where $x_i = (\phi_i, \theta_i)$ are the coordinates for each agent, for $i = 1, \ldots, N$. We again assume that the agents implement the control law

$$u = (BD \otimes I_2)p = (BD \otimes S)z, \tag{6.22}$$

where $p = (I_M \otimes S)z$. Results from implementing this control law for various values of the matrices B and D are shown in Fig. 6.13. Figure 6.13 shows examples of trochoidal patterns on a two-dimensional unit sphere using the formulas (6.21).

Still another alternative approach to generate paths on the sphere is to use the (inverse) stereographic or geodesic projections onto the two-dimensional sphere of a pattern generated by the proposed extended consensus protocol on the plane. Recall that the stereographic projection $\pi_s : \mathbb{S}^2 \backslash \{(0, 0, 1)\} \mapsto \mathbb{R}^2$ is defined via the expressions

$$x = \frac{\mathsf{x}}{1 - \mathsf{z}}, \qquad y = \frac{\mathsf{y}}{1 - \mathsf{z}}, \tag{6.23}$$

for $(\mathsf{x}, \mathsf{y}, \mathsf{z}) \in \mathbb{S}^2$, that is, $\mathsf{x}^2 + \mathsf{y}^2 + \mathsf{z}^2 = 1$. Alternatively, the geodesic projection $\pi_g : \mathbb{S}^2 \backslash \mathbb{S}^1 \mapsto \mathbb{R}^2$ is defined via the expressions

$$x = \frac{\mathsf{x}}{\mathsf{z}}, \qquad y = \frac{\mathsf{y}}{\mathsf{z}}, \tag{6.24}$$

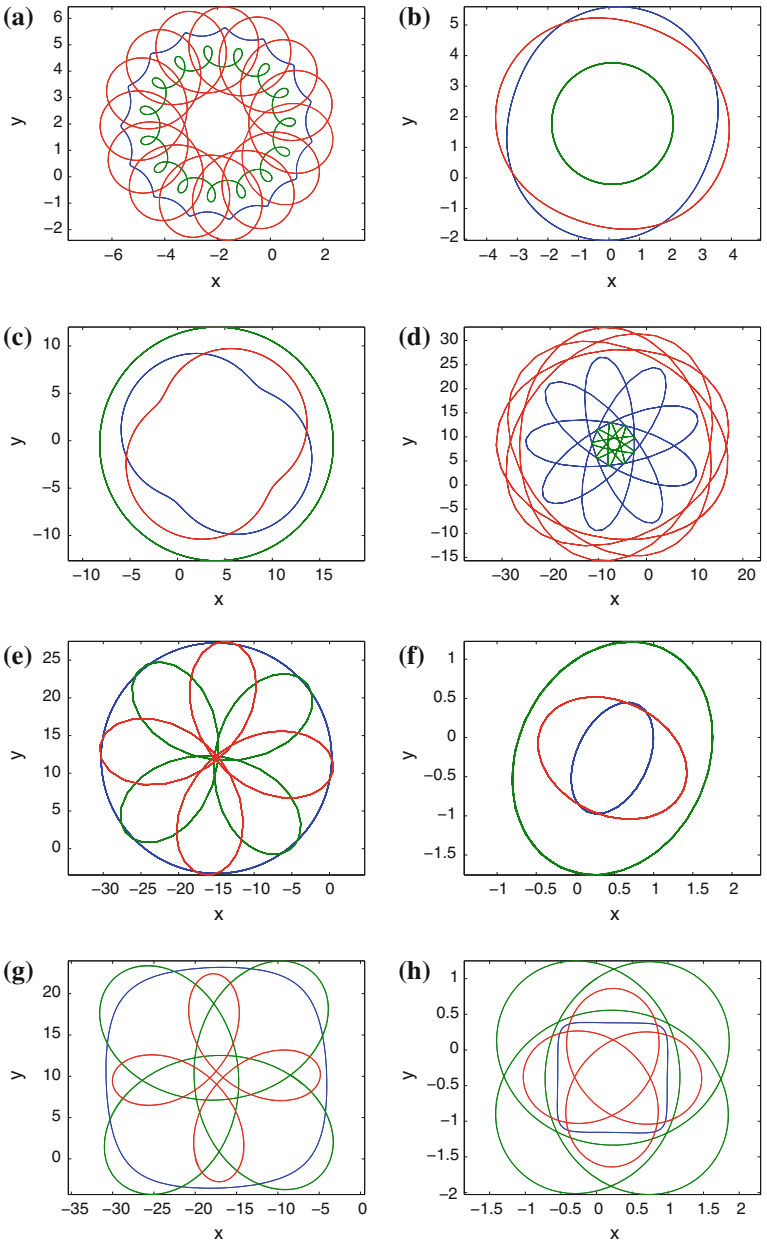


Fig. 6.10 A menagerie of orbits with three agents using the extended consensus protocol; complete graph. **a** $B = \text{diag}(-1, 1, 3)$, **b** and **c** $B = \text{diag}(0.5, 2, 0.5)$, **d** $B = \text{diag}(-1, -0.1936, 1)$, **e** $B = \text{diag}(4, -4, -4)$, **f** $B = \text{diag}(-5, 3, 2)$, **g** and **h** $B = \text{diag}(-2.4736, 3, 2)$

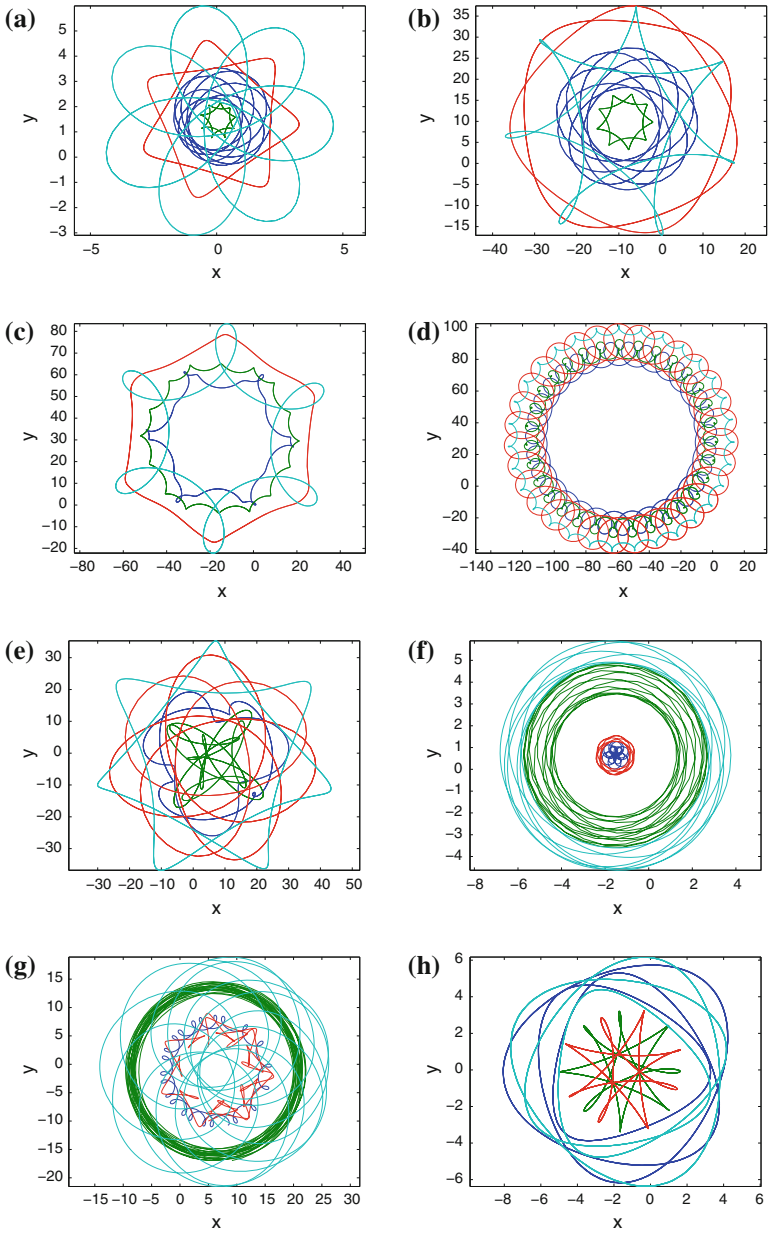


Fig. 6.11 Sample orbits with four agents using the extended consensus protocol; path graph. **a** and **b** $B = \text{diag}(2, 0.1826, -0.6126, 2)$, **c** $B = \text{diag}(2, 1.7141, -0.8257, 2)$, **d** $B = \text{diag}(2, 3.622, 2.336, -1)$, **e** $B = \text{diag}(-1, -1.145, 1.297, -1)$, **f** $B = \text{diag}(0.15, -1, 0.15, -1)$, **g** $B = \text{diag}(0.15, -1, 0.15, 1)$, **h** $B = \text{diag}(5, -2, -2, 5)$

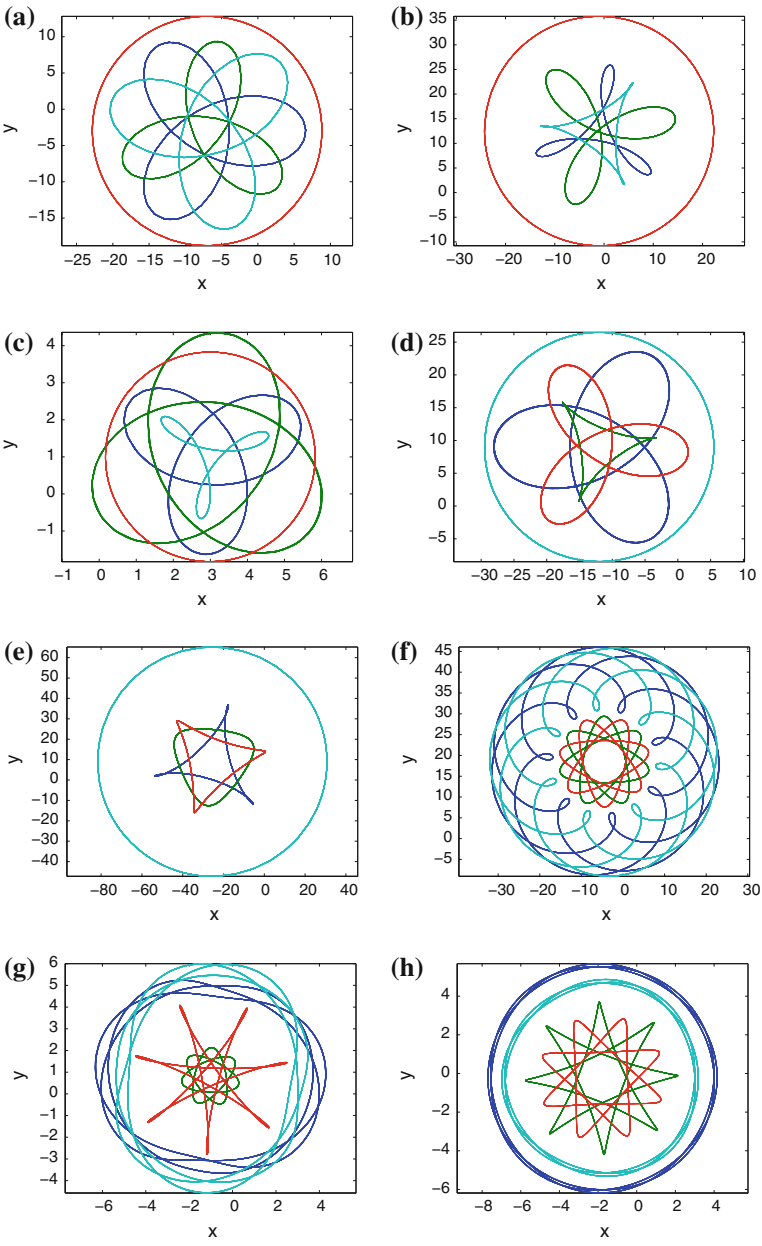


Fig. 6.12 Sample orbits with four agents using the extended consensus protocol; complete graph. **a**–**c** $B = \text{diag}(-1, -1, +1, -1)$, **d** and **e** $B = \text{diag}(-2, -2, -2, 2)$, **f** $B = \text{diag}(5, -2, -2, 5)$, **g** $B = \text{diag}(5, -0.866, -3.208, 5)$, **h** $B = \text{diag}(3.229, -2, -1.515, 5)$

where $\mathbb{S}^1 = \{(x, y) \in \mathbb{R}^2 : x^2 + y^2 = 1\}$ is the equator. The geodesic projection creates larger distortions of distances near the equator.

Figure 6.14 shows the results by applying the geodesic projection on a surface of revolution. The "vase" shown in this figure was generated using the parametric equations

$$x = (2 + \cos u) \sin v, \qquad y = (2 + \cos u) \cos v, \qquad z = u, \tag{6.25}$$

where u and v are the local coordinates on the two-dimensional vase manifold. Although not as elaborate or elegant as the vases of the Greek geometric period shown in Fig. 6.14, one nonetheless cannot help but admire the richness of the geometric patterns shown in Fig. 6.14, generated by the simple control law (6.10).

6.9 Discussion: The Mathematics of Aesthetics

An often repeated dictum of common wisdom is that "beauty is in the eye of the beholder." But is it? Although it is clear that perception of appeal or beauty is mainly an objective process (no ontological esthetic feeling is known), it is also equally clear that there exist esthetically attractive visual stimuli that transcend personal taste and seem to invoke the same feelings of acceptance or pleasure (equivalently, distastefulness or displeasure) among the majority of human observers. Several psychological studies seem to indicate that the human visual perception system is wired to be drawn to (overt or covert) symmetric, orderly patterns [4, 23, 24]. Plato (427–347 BC) was the first to state that "balanced things are always beautiful." Besides, isn't true that the creative artistic process, in general, produces order from disorder? If symmetry and order is the embodiment of harmony and beauty, and their lack is the manifestation of the opposite[2] (the constant struggle between χάος and κόσμος in ancient Greek culture) shouldn't perhaps be possible to describe certain esthetics using formal methods?

In his influential essay *Inquiry into the Origin of Our Ideas of Beauty and Virtues* the British philosopher Francis Hutcheson (1694–1746) attempted to answer this question, by showing how beauty depends on formal qualities. He suggested that beauty is "uniformity amidst variety." Thus, according to Hutcheson, richly varied compositions that are organized in accordance with some underlying unifying principle are beautiful [28]. The contemporary theory of the psychology of esthetics actually replaces the prominence of classical symmetry with the somewhat similar, albeit vague, notion of "organic unity" [33].

[2] Not everyone is in agreement, of course, with the classical notion of beauty and symmetry. One can easily argue that nonsymmetric patterns may also be esthetically pleasing as long as they do not result in chaos [26]. Recent psychological studies of works of modern art actually claim that it is the cognitive processes themselves that are involved in understanding, classifying, and evaluating a work of art which determine positive, self-rewarding esthetic experiences [20]. Even so, good gestalts tend to give preference to symmetry over nonsymmetry [13].

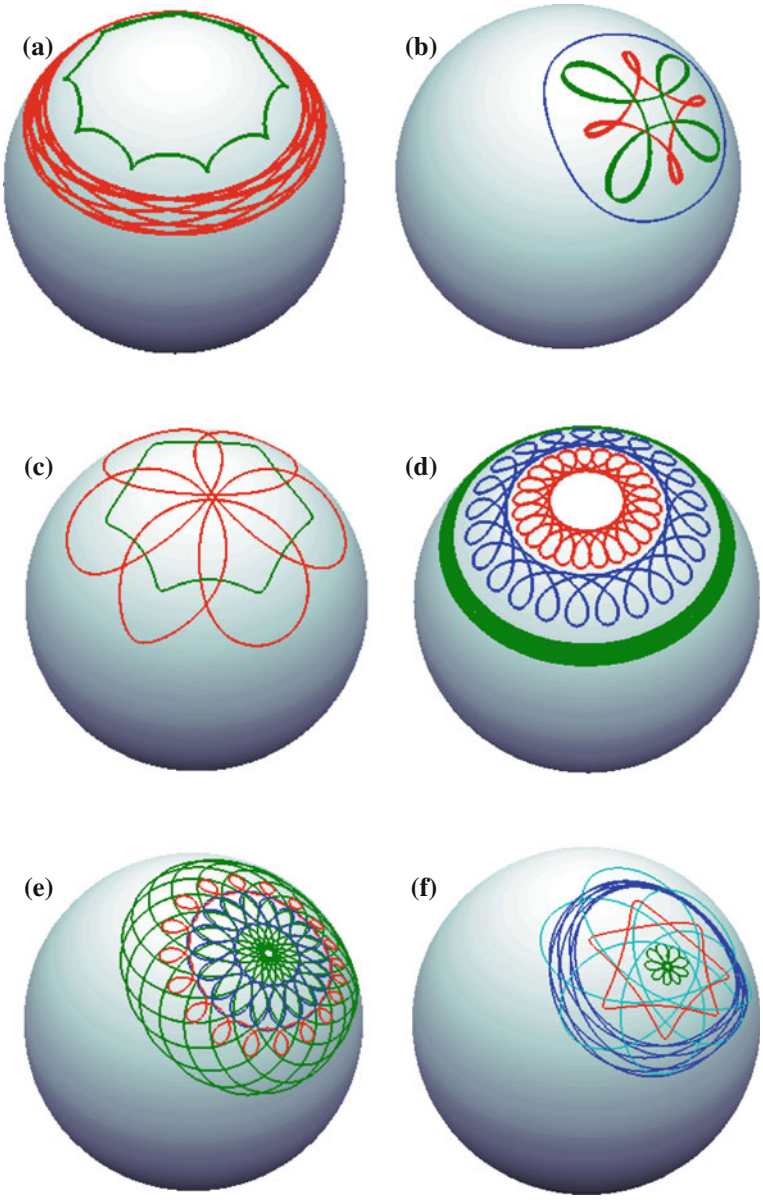


Fig. 6.13 *Sample orbits* on a *sphere* with three [(**a**)–(**d**)] and four [(**e**)–(**f**)] agents using the extended consensus protocol; path graph. **a** and **c** $B = \text{diag}(0, 1, -6.5933)$, **b** $B = \text{diag}(-2.4736, 3, 2)$, **d** $B = \text{diag}(1, -2.9054, 0.5)$, **e** $B = \text{diag}(1, 2, 2.0034, -1)$, **f** $B = \text{diag}(2, 0.1826, 0.6126, 2)$

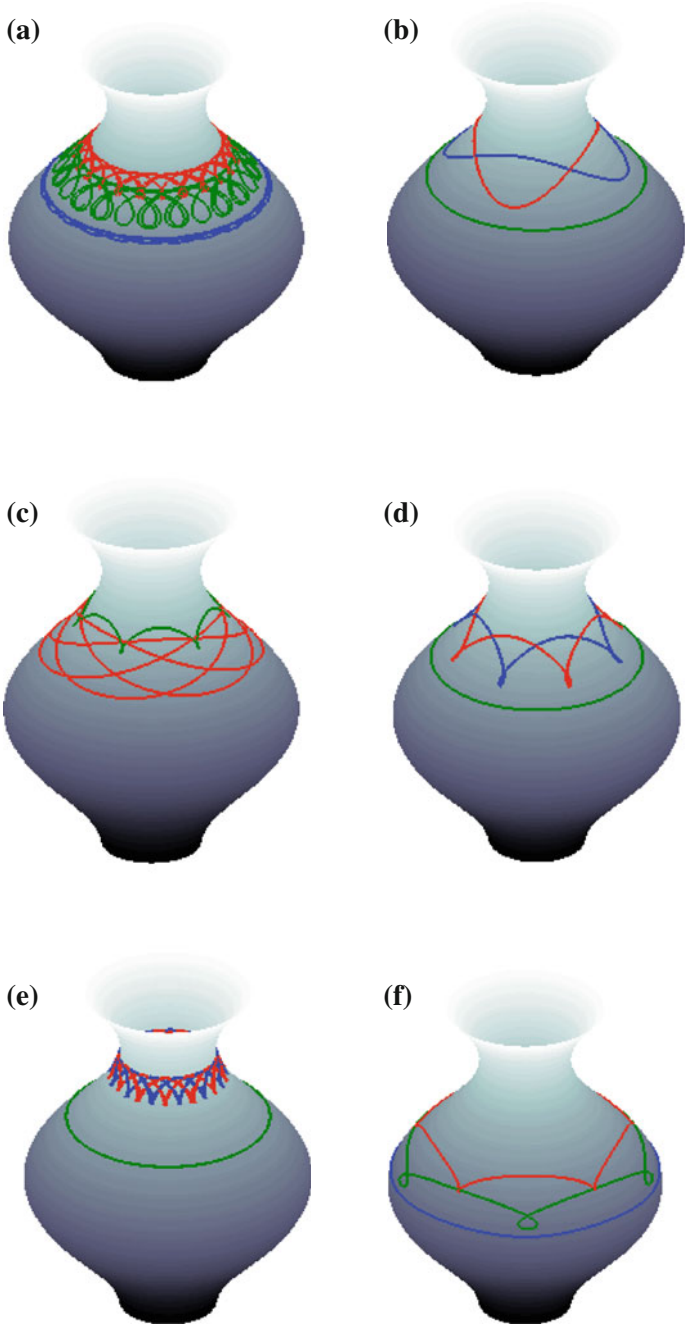


Fig. 6.14 *Sample orbits* on a *vase* with three agents using the extended consensus protocol; path graph. **a** $B = \text{diag}(-2.4736, 3, 2)$, **b** $B = \text{diag}(4, -4, 4)$, **c** $B = \text{diag}(0, 1, -6.5933)$, **d** $B = \text{diag}(-1, 2, -1)$, **e** $B = \text{diag}(1, -8.7289, 1)$, **f** $B = \text{diag}(1, -1, -0.5)$,

Motivated by such observations, the American mathematician G. Birkhoff (1884–1944) introduced the concept of an *esthetic measure*, a single number that describes the esthetic appeal of an object [3]. Since then, several versions and modifications of Birkhoff's original esthetic measure have been proposed, all of which defining the esthetic measure—in one form or another—as the ratio of order over disorder, equivalently, symmetry over complexity.

More formally, Birkhoff's esthetic measure M is defined by

$$M = \frac{O}{C}, \tag{6.26}$$

where O is the order or harmony of the observed work, and C is the object's complexity. As also Birkhoff himself recognized, formalizing these concepts, which depend on the context, biases of the observer, and so on, is difficult. Whereas there is a consensus among researchers that O is essentially a measure of the object's symmetries, there seems to be less of an agreement on how one measures its complexity. One proposal that has come forth is that C is related to the redundant information conveyed [2, 29]. This has given rise to several information-theoretic measures ("informational aesthetics measures") based on Kolmogorov complexity, Shannon's information theory and physical entropy [41]. The field of computational esthetics has evolved around these ideas in an effort to define and quantify artistic creativity using mathematical algorithms [48].

What does the field of computational esthetics have to do with control theory? In this work we have shown that control algorithms can be used to generate geometric patterns in a natural manner. Is not difficult for one to envision generalizations leading to more complicated art forms. Most importantly, recall that a large part of control theory deals with the maximization of a given payoff. In this context, it would be intriguing to investigate control algorithms that attempt to maximize the esthetic measure within a given class.

6.10 Conclusions

We have presented an extension of the classical consensus algorithm for multi-agent systems to achieve consensus outside the convex hull of the initial conditions of the agents. As a by-product of this idea, we have shown how to generate agent trajectories leading to intricate geometric patterns in the plane using only relative, local information. Future work will concentrate on developing a general theory for orbit design for an arbitrary number of agents in two, and three dimensions. Apart from their inherent esthetical appeal, these orbits can have immediate applications in the area of coordinated, persistent surveillance and monitoring using a team of agents interacting using local information. An interesting lingering question that still remains to be answered, is whether one can classify these curves according to their esthetic appeal using a properly defined esthetic measure.

Acknowledgments The authors would like to thank Magnus Egerstedt and Amy LaViers for organizing the session on "Controls and Art" during the 2011 American Control Conference and for inviting the authors of this paper to contribute to the session. The authors would also like to thank Prof. Athanassios Economou from the College of Architecture at Georgia Tech for several informative discussions over the use of group symmetries in art and architecture.

References

1. Arcak M (2007) Passivity as a design tool for group coordination. IEEE Trans Autom Control 52(8):1380–1390
2. Bense M (1969) Einführung in die informationstheoretische Asthetik. Grundlegung und Anwendung in der Texttheorie. Rowohlt Taschenbuch Verlag, Hamburg
3. Birkhoff GD (1933) Aesthetic measure. Harvard University Press, Cambridge
4. Bouleau C (1980) The painter's secret geometry: a study of composition in art. Hacker Art Books, New York
5. Boyer CB (1968) A history of mathematics. Wiley, New York
6. Brewer J (1978) Kronecker products and matrix calculus in system theory. IEEE Trans Circ Syst 25(9):772–781
7. Coldstream JN (2003) Geometric Greece: 900–700 BC. Psychology Press, Abingdon
8. Dawson I (2013) Spirographs. http://www.iandawson.net/spiro.php. Accessed 20 May 2013
9. Economou A (2011) The symmetry lessons from Froebel building gifts. Environ Plann B: Plann Des 26(1):75–90
10. El-Said I, El-Bouri T, Critchlow K, Damlūjī SS (1993) Islamic art and architecture: the system of geometric design. Garnet Publishing, London
11. Ettinghausen R, Grabar O, Jenkins M (2001) Islamic art and architecture: 650–1250, vol 51. Yale University Press, London
12. Fax JA, Murray RM (2004) Information flow and cooperative control of vehicle formations. IEEE Trans Autom Control 49(9):1465–1476
13. Frith CD, Nias DKB (1974) What determines aesthetic preferences? J Gen Psychol 91:163–173
14. Godsil CD, Royle G (2001) Algebraic graph theory. Springer, New York
15. Grünbaum B, Grünbaum Z, Shepard GC (1986) Symmetry in moorish and other ornaments. Comput Math Appl 12(3):641–653
16. Hall LM (1992) Trochoids, roses, and thorns-beyond the spirograph. Coll Math J 23(1):20–35
17. Halliwell L (2013) http://www.lesleyhalliwell.co.uk/. Accessed 20 May 2013
18. Knight TW (1995) Transformations in design: a formal approach to stylistic change and innovation in the visual arts. Cambridge University Press, Cambridge
19. Lawrence JD (1972) A catalog of special plane curves. Dover Publications, New York
20. Leder H, Belke B, Oeberst A, Augustin D (2004) A model of aesthetic appreciation and aesthetic judgments. Br J Psychol 95(4):489–508
21. Leonard NE, Fiorelli E (2001) Virtual leaders, artificial potentials, and coordinated control of groups. 40th IEEE Conference on Decision and Control, pp 2968–2973, 4–7 Dec 2001
22. Leonard NE, Young GF, Hochgraf K, Swain DT, Trippe A, Chen W, Fitch K, Marshall S (2014) Controls and art. In the dance studio: an art and engineering exploration of human flocking. In: Lecture Notes in Computer Sciences. Springer-Verlag, New York, p xxx
23. Locher P, Nodine C (1989) The perceptual value of symmetry. Comput Math Appl 17(4):475–484
24. Locher P, Nodine C (1987) Eye movements: from physiology to cognition. Chapter Symmetry Catches the Eye. Elsevier, Holland, pp 353–361
25. Marshall JA, Broucke ME, Francis BA (2004) Formations of vehicles in cyclic pursuit. IEEE Trans Autom Control 49(11):1963–1974
26. McManus C (2005) Symmetry and asymmetry in aesthetics and the arts. Eur Rev 13:157–180

27. Mesbahi M, Egerstedt M (2010) Graph theoretic methods in multiagent networks. Princeton University Press, Princeton
28. Mitchell WJ (1990) The logic of architecture: design, computation, and cognition. MIT Press, Cambridge
29. Moles A (1968) Information theory and esthetic perception. University of Illinois Press, Champaign
30. Olfati-Saber R (2006) Swarms on sphere: a programmable swarm with synchronous behaviors like oscillator networks. In: 45th IEEE Conference on Decision and Control, pp 5060–5066
31. Olfati-Saber R, Murray RM (2004) Consensus problems in networks of agents with switching topology and time-delays. IEEE Trans Autom Control 49(9):1520–1533
32. Olfati-Saber R, Fax JA, Murray RM (2006) Consensus and cooperation in multi-agent networked systems. Proc IEEE 97:215–233
33. Osborne H (1986) Symmetry as an aesthetic factor. Comput Math Appl 12(1):77–82
34. Parchos EA, Sotiroudis P (1999) The schemata of the stars: Byzantine astronomy from A.D. 1300. World Scientific, Singapore
35. Park J-H, Joo Y, Yang J-G (2007) Cycloids in Louis I. Kahns Kimbell Art Museum at Fort Worth, Texas. Math Intell 29:42–48
36. Pavone M, Frazzoli E (2007) Decentralized policies for geometric pattern formation and path coverage. J Dyn Syst Meas Contr 129:633–643
37. Pohl D (2013) The loop yoga drawing project. http://spirographart.blogspot.com/. Accessed 20 May 2013
38. Public domain image "File:Triquetra_on_book_cover.jpg" from user Chameleon on the Wikimedia Commons. http://commons.wikimedia.org/wiki/File:Triquetra_on_book_cover.jpg;. Accessed 10 Sept 2013
39. Public domain image ID 289442_WDC25C21 (acquisition date 01 Sept 2005). United States Geological Survey. http://earthexplorer.usgs.gov/;. Accessed 21 Nov 2013
40. Ren W, Beard RW (2005) Consensus seeking in multi-agent systems using dynamically changing interaction topologies. IEEE Trans Autom Control 50(5):655–661
41. Rigau J, Feixas M, Sbert M (2008) Informational aesthetics measures. IEEE Comput Graph Appl 28(2):24–34
42. Sarlette A, Sepulchre R (2009) Consensus optimization on manifolds. SIAM J Control Optim 48(1):56–76
43. Schattschneider D, Hofstadter D (2004) In: Escher MC (ed) Visions of symmetry. Thames & Hudson, London
44. Schoelling AP, Siegel H, Augugliaro F, D'Andrea R (2014) Controls and art. So you think you can dance? Rhythmic flight performances with quadrocopters. In: Lecture Notes in Computer Sciences. Springer-Verlag, New York, p xxx
45. Schweitzer B (1971) Greek geometric art. Phaidon, London
46. Simmons Jeffrey. Paintings: 1999–2000: Trochoid. http://jeffreysimmonsstudio.com/project/trochoids-paintings-1999/. Accessed 20 May 2013
47. Simoson AJ (2008) Albrecht Dürer's trochoidal woodcuts. Probl Resour Issues Math Undergraduate Stud (PRIMUS) 18(6):489–499
48. Stiny G, Gips J (1978) Algorithmic aesthetics: computer models for criticism and design in the arts. University of California Press, Berkeley
49. Suzuki I, Yamashita M (1999) Distributed anonymous mobile robots: formation of geometric patterns. SIAM J Comput 28(4):1347–1363
50. Tron R, Afsari B, René Vidal (2012) Riemannian consensus for manifolds with bounded curvature. IEEE Trans Autom Control
51. Tsiotras P, Reyes Castro LI. A note on the consensus protocol with some applications to agent orbit pattern generation. In: 10th Symposium on Distributed Autonomous Robotic Systems (DARS). Lausanne, Switzerland, 1–3 Nov 2010
52. Wang L-S, Krishnaprasad PS (1992) Gyroscopic control and stabilization. J Nonlinear Sci 2:367–415
53. Wessén E (1952) Det svenska runverket. http://fornvannen.se/pdf/1950talet/1952_193.pdf (in Swedish)
54. Weyl H (1952) Symmetry. Princeton University Press, London

Chapter 7
Generating Music from Flocking Dynamics

Cristián Huepe, Marco Colasso and Rodrigo F. Cádiz

7.1 Order, Disorder, Flocks, and Music

While it may be impossible to determine precisely what defines appealing and interesting music, it can be argued that, in order to have these qualities, sounds should contain enough elements of order and disorder, coherence and decoherence, or structure and variation throughout a musical piece. These can appear in different ways and at various levels; in the interplay between expected and surprising note sequences found in engaging melodies, in the superposition of perfectly and imperfectly coinciding harmonics required for a rich chord, or in the mixture of regular and syncopated rhythms that build a driving percussion layer, to name a few. Another important component is the inclusion of multiple elements that can both combine into a coherent musical structure or disband into independently acting parts. Music is often based entirely on fragile harmonic or rhythmic structures that fluctuate between states of order and disorder, constantly consolidating and collapsing. Composers explicitly explore these extremes, for example, in the interplay between instruments in an orchestra, between melodic lines in counterpoint [1], between collective and solo parts in a Jazz performance [2], or in the dense chromatic melodies of some contemporary music [3].

C. Huepe (✉)
CHuepe Labs, 954 W 18th Place, Chicago, IL 60608, USA
e-mail: cristian@labolabs.com

C. Huepe
Northwestern Institute on Complex Systems, Northwestern University, Evanston, IL, USA

M. Colasso
Center for Research in Audio Technologies, Music Institute, Santiago, Chile

M. Colasso · R. F. Cádiz
Computer Science Department, School of Engineering Pontificia Universidad Católica de Chile, Santiago, Chile

A. LaViers and M. Egerstedt (eds.), *Controls and Art*,
DOI: 10.1007/978-3-319-03904-6_7,

A similar interplay between coherence and decoherence can be observed in the dynamics of groups of agents following decentralized control algorithms that can lead to consensus or to solutions where they act independently [4–6]. These can be described as either displaying ordered collective dynamics or remaining in a disordered state. An example of such systems that has received much attention in recent years is given by autonomous agents that move according to "flocking" algorithms [7–9]. These algorithms are inspired by the study of the coordinated collective motion observed in bird flocks, fish schools, or insect swarms [10–13]. They attempt to mimic such behavior by implementing motion control rules followed by each individual based only on its local information, but that can lead the group to achieve consensus and move as a whole.

The analysis above leads us to postulate that parallels can be drawn between the dynamical structures found in flocks and in music [14]. We will explore here these connections by studying different approaches for mapping one into the other. By comparing the resulting sounds, we will examine which approaches can best capture the main features of the flocking dynamics, as understood intuitively by our hearing, and which can produce interesting musical results. While the mapping of mathematical algorithms to music has a long history as a creative tool, our work takes a somewhat different approach, inspired by the relationship between emergent phenomena in complex systems and music.

Our guiding principle is that the components of a musical piece combine to produce collective effects that are more than the sum of the parts, much like the individual dynamics of physical or biological components often combine to generate emergent collective results. Conversely, music is simply a one-dimensional function of time that describes the sound signal, but it can somehow contain various emergent properties of the system that generated it. In this context, our exploration of different mappings between flocks and sound allows us to ask fundamental questions about the possibility of generating interesting and appealing music from complex systems and about our ability to perceive emergent phenomena (here an order–disorder state transition) through sonic perception. Our study could lead to various applications, such as science-inspired art or the development of new sonification techniques for mapping datasets into music that take advantage of the brain's ability to extract structures from sound [15–18].

We find in this book other examples of mathematical algorithms playing an active role in live performances. In Chap. 9 "Style-based Robotic Motion in Contemporary Dance Performance" by LaViers et al., a framework for generating robotic motion was employed in a performance with a humanoid robot and real dancers. In Chap. 8 by Godbehere and Goldberg, "Algorithms for Visual Tracking of Visitors Under Variable-Lighting Conditions for a Responsive Audio Art Installation", the motion of visitors in a gallery space was captured by a camera and processed by tracking software to trigger sonic responses that were part of an audio art installation. In addition to these works demonstrating human–machine performances, other chapters also relate to our study by considering interesting artistic expressions that result from multi-agent control theory. In Chap. 6 by Tsiotras and Reyes Castro, "The Artistic Geometry of Consensus Protocols", intricate periodic, and quasi-periodic geometric

patterns are generated by using a multi-agent consensus motion control protocol to draw lines on a surface. Finally, Chap. 2 "In the Dance Studio: An Art and Engineering Exploration of Human Flocking" by Leonard et al applies multi-agent flocking algorithms similar to the model considered below to decide the movements of dancers in real-time during the performance. A combination of this approach with ours opens the interesting possibility of a live piece where dancers follow flocking-like dynamics in their motion, which is then captured by cameras and tracking software to generate the corresponding music (also in real-time) using one of the algorithm that we explore here.

Other studies have also developed algorithms for generating music based on flocking or swarming dynamics (two terms that, in this context, are often considered as equal). Their focuses, however, have been quite different from ours. In [19], for example, a *Boids* algorithm [20] was implemented where each agent emits its own sound wave (a filtered noise impulse) that is spatialized according to position. These sound waves then interact constructively or destructively based on their phases and amplitudes, producing interesting effects that depend on the parameters of the flocking dynamics. In [21], a Boids algorithm was used to generate music based on the dynamics of goal-directed swarms. These consisted of three different agent species that gain energy when colliding with randomly distributed energy sources and lose energy in all other interactions. A different instrument is assigned to each species, one with a clear tonal musical timbre, one with a chirping sound, and one with a more complex timbre. At each collision a musical note is produced, with its pitch determined by each agent's energy level. In [22], a flocking simulation was implemented in a three-dimensional space that is divided into different acoustic regions. Whenever an agent enters a particular region, a predefined musical pattern is made audible. In *Swarm Music* [23], a swarm of interacting particles is also moving in a three-dimensional space, but here MIDI events are produced based on the coordinates of each particle, which control their pitch, their amplitude, and the time interval between them. In *Swarm Lab* [24] the goal was to design a swarming system that is well suited for generating music. A user controllable swarm of sound-generating particles was implemented to allow some of the relevant properties of the swarm to be heard, such as its spatial position, heading direction, or the tightness of its collective motion. Finally, a different objective was pursued in [25], where human motion is captured by a video camera and used to influence a flock's behavior, with the state of each agent then used to control a MIDI instrument. The user thus assumes the role of a conductor, influencing the flock's musical activity.

In sum, flocking and swarming dynamics present a number of exciting research avenues that can be explored from a complex systems and artistic perspective. While they are often based in very simple dynamical rules, these algorithms can produce results that are sophisticated enough to outperform the designer and create new structures that cannot be predicted by the elements used in their construction [26]. They are, therefore, a natural choice for exploring the potential of what could be called *performing machines* [23].

We develop below our exploration on generating music from flocking dynamics. In Sect. 7.2, we present the flocking algorithm used throughout our analysis. Section 7.3

describes the software developed for our study. In Sect. 7.4, we detail the different approaches that were considered for mapping flocking dynamics into sound and the musical results obtained for the algorithms and parameters tested. In Sect. 7.5, we describe the composition of *Ritmos Circadianos*, a piece based entirely on flocking dynamics that was created for a robot orchestra by one of the co-authors of this work (Rodrigo F. Cádiz). Finally, Sect. 7.6 includes our discussion and conclusions, where we elaborate on the natural connections that can be established between flocking dynamics and music, and suggest other possible approaches for mapping one into the other.

7.2 A Minimal Flocking Algorithm

We consider as a starting point a minimal algorithm that generates flocking dynamics and achieves collective motion; the Vicsek model [27], which has become a referent in the field. This model has several features that make it well suited for our study. It is simple, relatively easy to implement, and does not require excessive computational resources, while still capturing the essential aspects of more sophisticated flocking algorithms. It has been extensively studied since its inception in 1995, which allows us to start our exploration on solid grounds. It displays many of the characteristic features of complex statistical systems, such as nonequilibrium dynamics, a noise-driven order–disorder transition, and self-organization. An additional advantage of considering a flocking algorithm is that its results are directly observable as motion in physical space. This lack of hidden degrees of freedom facilitates the simultaneous examination of the model dynamics and its corresponding generated sounds. Finally, the natural origin of flocking motion is appealing for creative musical work, since natural themes are often artistically inspiring due to their ubiquity in our common human experience.

The Vicsek model is defined as follows. A group of agents advancing at a fixed common speed v_0 are coupled only through alignment interactions that steer them toward the mean heading direction of all agents within a given radius. Specifically, a set of N point-like agents are placed in a two-dimensional square box with side L and periodic boundary conditions. At every time-step t, the system is fully defined by the set of agent positions $\boldsymbol{x}_i(t)_{i=1,\cdots,N}$ and their heading angles $\theta_i(t)_{i=1,\cdots,N}$. For the next time-step, the state of the system is computed through:

$$\theta_i(t+\Delta t) = \mathrm{Ang}\left\{ \sum_{Z_i(t)} \mathrm{Vel}\left[\theta_j(t)\right] \right\} + \eta\,\xi_i(t) \tag{7.1}$$

$$\boldsymbol{x}_i(t+\Delta t) = \boldsymbol{x}_i(t) + \Delta t\; v_0 \mathrm{Vel}\left[\theta_i(t)\right]. \tag{7.2}$$

Here, the Ang{.} function outputs the angle of its vector argument and the Vel[.] function produces a unit vector that points in the direction of its angle argument. Noise

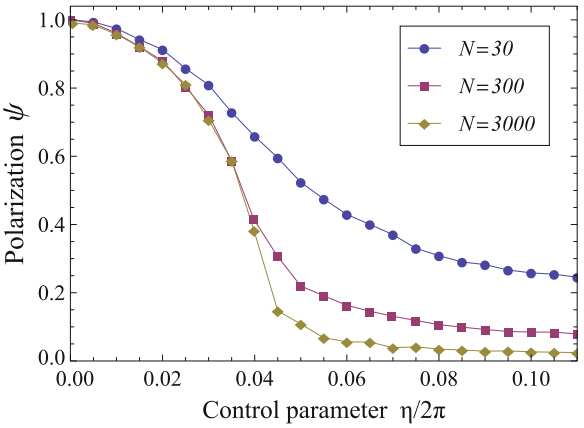


Fig. 7.1 Polarization order parameter ψ as a function of noise η for the standard Vicsek algorithm used in this paper. As the noise level is decreased, the agents start aligning their heading directions and go from a disordered state to an ordered state where they advance in a common direction. The three curves result from simulations with equal mean density but different number of agents. We observe a sharp transition in large systems and a smooth transition in smaller systems. However, even for the $N = 30$ case that we use in our sonifying exploration, the ordered and disordered states can be clearly distinguished

of intensity η is introduced by adding to each agent's heading angle a delta-correlated random variable $\xi_i(t)$ that is uniformly distributed in the interval $[-1/2, 1/2]$. The set $Z_i(t)$ includes the j indexes of all agents that are within a distance R of agent i.

The equations above define a standard Vicsek algorithm. For low enough noise η, they converge to a consensus state where all agents are heading in a similar direction. For high noise values, the system disorganizes and agents point in random directions. In order to quantify the degree of order in the system, we define the polarization order parameter as

$$\psi = \frac{1}{v_0 N} \sum_{i=1}^{N} v_i(t), \tag{7.3}$$

which is proportional to the total momentum of the group. It will be equal to 0 in the disordered state and to 1 in the polarized flocking state.

We chose to use the Vicsek model dynamics as the starting point of this investigation because it is a minimal algorithm and one of the best studied models for collective motion. For large N values, it displays a sharp transition between its ordered and disordered phases at a critical noise level (see Fig. 7.1). In both phases, agents tend to group into clusters of a variety of sizes (despite the lack of attractive interactions) that are continuously splitting and merging, with a higher proportion of large clusters occurring at low noise levels. Some clusters head in the same approximate direction while others drift in different directions until they intersect the trajectories of others. Due to the finite interaction range, clusters have no influence over each other until

they come closer than distance R. Within a cluster, agents must move in a common direction. It has also been shown that, even in the ordered state, no fully stationary solutions exist in finite systems with nonzero noise. Instead, random fluctuations will always produce intermittent bursts where all order is lost before a new heading direction is selected [28]. Finally, note that the lack of explicit attraction terms implies that, for any nonzero noise level and in the absence of boundaries, agents will spread over larger and larger areas and all clusters will end up disintegrating by splitting or losing agents. The presence in our simulation of periodic boundary conditions fixes instead the mean density of agents in the system.

The dynamics of clusters and single agents described above produce a rich combination of ordered and disordered, individual and collective, and coherent and decoherent motion that could potentially translate into interesting musical pieces. In order to explore this possibility, we must first implement specific algorithms that map the spatial dynamics to sound, a task that we address in the following sections.

7.3 Software Tools

We developed software tools to explore the sounds generated by our flocking simulations in real-time, which allowed us to interactively test different approaches and parameters. The Vicsek algorithm was programmed first in Java, together with the computation of the quantities required for the mapping approaches described in Sect. 7.4. This code was then imported into Jitter, a visual processor and editor embedded inside MaxMSP. All sound synthesis algorithms were implemented in the MaxMSP audio programming environment [29, 30], a visual programming language for music, and multimedia that is widely used in the computer music community.

Figure 7.2 displays a snapshot of the control and output panels of the MaxMSP patch developed for our analysis. The top panel contains faders that allow the user to interactively control the simulation and mapping parameters. It also displays the polarization ψ and other order parameters describing the state of the simulation. The middle panel includes a high-pass filter curve with adjustable roll-off, used to avoid low-frequency rumble, and a scrolling real-time spectrogram of the generated sounds. The tabs on the bottom panel allow the activation and control of the different algorithms that map the flocking dynamics to sound. Note that the amplitudes $A_i(t)$ and wave frequencies $\omega_i(t)$ computed in Sect. 7.4 are not directly translated into sounds. Instead, this panel controls a final processing stage, carried out within MaxMSP, that allows us to increase audibility by setting the range of sound frequencies covered (choosing a linear or exponential mapping between $\omega_i(t)$ and the output sound frequencies) and by selecting if $A_i(t) = 0$ will correspond to complete silence or to low volume.

Finally, notice that the bottom panel also includes tabs for AM sound generation and granular synthesis. While these modules were implemented in our program and can produce richer musical results, we will not discuss them in this work. We

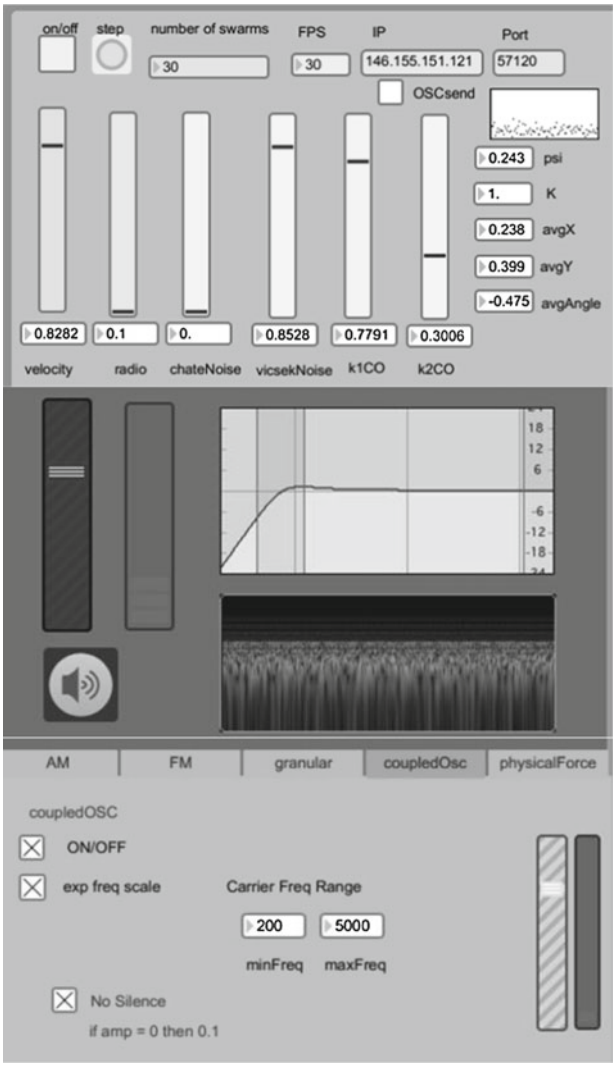


Fig. 7.2 Control and output panels of the MaxMSP patch developed for mapping the flocking dynamics to sound. The *top panel* has faders that allow the user to interactively control the parameters of the flocking simulation and of the mapping algorithm. It also displays averaged quantities like the order parameter ψ for the current state of the simulation. The *middle panel* includes an adjustable high-pass filter and displays a spectrogram of the generated sound in real-time. The tabs on the *bottom panel* control general features, such as the accessible frequency range, of the different mappings that can be implemented. A separate window (not shown) displays the spatial dynamics of the flocking simulation, with graphic properties controlled by an additional panel

focus instead on simple sinusoidal sound generators based mainly on frequency modulation (FM), since our purpose here is to explore the basic features of the different approaches presented in Sect. 7.4.

7.4 From Emergent Dynamics to Emerging Sounds

We now consider three different approaches for mapping flocking dynamics into sound. First, a simple *direct approach*. Then, one based on the synchronization of oscillators; the *coupled oscillators approach*. And finally, one built on a friction-like process; the *physical friction approach*.

7.4.1 The Direct Approach

The most straightforward procedure to map the results from our flocking simulations into sounds is to directly relate the state of each agent (given by its position and velocity) to the corresponding amplitude $A_i(t)$ and frequency $\omega_i(t)$ of a sinusoidal function $A_i(t)\sin[\omega_i(t)t]$. Of the three approaches explored in our study, this is the only one that has been investigated in the literature. It has been tried with oscillators [31], granular synthesis [32], and pre-recorded sounds [31]. In these works, well-known swarm intelligence simulations (such as ant foraging [33] or flocking [34–37] simulations) are typically used as the basic model, with the expectation that they will generate sounds that at first seem to disperse randomly but eventually organize. The state of the individual agents is used to create the microscopic sonic structure, whereas the emergent behavior results in the macroscopic structure of the musical composition [31].

In our study, we identified the x coordinate of agent i directly with $\omega_i(t)$ and its y coordinate with the frequency $\alpha_i(t)$ of an amplitude modulation (AM) function $A_i(t) = A_{\max}\sin[\alpha_i(t)]$. In the example below, however, we do not use this AM component, keeping instead the amplitude constant in order to simplify our analysis.

When the direct approach is used, the resulting sounds mimic the individual dynamics of each agent, but they do not convey the strong feeling of cohesiveness, structure, and emergence that is apparent to the eye when viewing the simulations. Panels a and b on Fig. 7.3 show snapshots of the simulations in the disordered ($\eta = 0.87$) and ordered ($\eta = 0.07$) states, respectively. Panels c and d display the corresponding spectrograms of the sounds obtained using the direct approach. Panel e shows a case where the noise level was changed dynamically during the simulation, to transition from the disordered to the ordered regime. All spectrograms presented correspond to approximately 15 s of sound. In the sounds and spectrograms resulting from this approach, the dynamics of each agent can often be distinguished, but other collective structures, such as the clusters, do not seem to be reflected in the musical mapping. The resulting sounds are quite simple. In the disordered state, they mainly

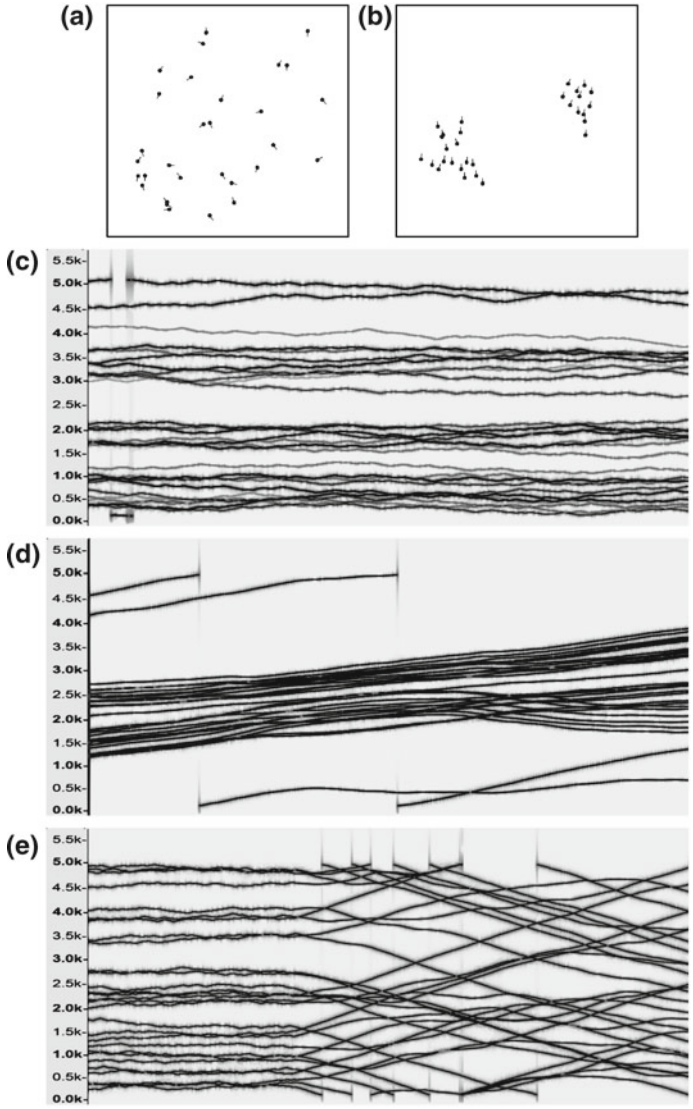


Fig. 7.3 Mapping from flocking dynamics to sounds using the direct approach. Top: Snapshots of the positions and headings of the agents in the disordered (**a**) and ordered (**b**) states. Note that various cluster sizes can be observed, including single-particle ones. **c** and **d** Spectrograms of the sounds generated in the disordered and ordered states, respectively. **e** Spectrogram of the transition from a disordered to an ordered state; the resulting change of dynamics can be clearly seen and heard

consist of several independently fluctuating frequencies. When the noise is reduced, as the system organizes, these sounds typically start rising or decreasing in pitch as a function of time. For very low noise, in the fully organized state, the frequencies of all oscillators will increase or decrease continuously at a similar rate. Although the sonic impression of the direct approach does allow the listener to distinguish between the ordered and disordered phases, the lack of additional structure yields a musical result that quickly becomes monotonous. In addition, the periodicity of the box implies that when the frequency associated to a given particle reaches its edge, it must instantaneously jump to its opposite extreme value.

7.4.2 The Coupled Oscillators Approach

In this approach, the idea is to go beyond the direct mapping of individual positions and velocities to sounds. Instead, we search for a way to capture the coherence of the trajectories of different agents over time. This can be achieved by associating to each agent an additional variable that controls its sonic output, and evolving it based on agent interactions. The value of this variable is thus decoupled from the current state of the system.

We implemented an algorithm where the frequency of the sound-generating sine wave associated to each agent i is defined as a new variable $\omega_i(t)$ that couples to the frequencies of its neighboring agents through a simple continuous-time consensus protocol. Each agent is given a randomly selected preferred frequency ω_i^0 in the interval $[0, 1]$, which remains fixed throughout the run. The value of the $\omega_i(t)$ of each agent is evolved in parallel to the flocking algorithm, following

$$\frac{d\omega_i(t)}{dt} = \beta_1 \left[\Omega_i(t) - \omega_i(t)\right], \tag{7.4}$$

$$\Omega_i(t) = \frac{1}{\beta_2 + \tilde{n}_i}\left[\beta_2\omega_i^0 + \sum_{j\in S_i}\omega_j(t)\right]. \tag{7.5}$$

Here, S_i contains the j-indexes of all $\tilde{n}_i$ agents within the sonic interaction range R_S of agent i, that is, with $\|\boldsymbol{x}_j - \boldsymbol{x}_i\| < R_S$. We define $R_S > R$ in order to have the interaction between sound generators reflect a group cohesiveness that goes beyond the coordinated motion of neighbors within the Vicsek interaction range R, which is trivially guaranteed by this flocking algorithm. The dynamics resulting from Eq. (7.4) drives $\omega_i(t)$ towards $\Omega_i(t)$. When agent i is within a distance R_S of other agents, the $\Omega_i(t)$ that results from Eq. (7.5) is a weighted average between its current neighbors' frequencies and its own preferred frequency ω_i^0. When agent i moves again in isolation, $\Omega_i(t) = \omega_i^0$ and $\omega_i(t)$ thus converges back to its own preferred frequency. Note that all $\omega_i(t)$ will remain in the interval $[0, 1]$ if initialized within this range. Parameter β_1 controls the rate at which each agent's frequency converges toward its current

target frequency $\Omega_i(t)$, whereas β_2 controls the weight of its preferred frequency ω_i^0 with respect to the mean frequency of its current neighbors when computing $\Omega_i(t)$. For large β_2, the preferred frequency of each agent will remain almost unperturbed when interacting with other agents. For small β_2, the frequencies of all agents within a cluster will become almost identical if they flock together long enough.

Regarding the amplitude, we are interested here in implementing dynamics that will enhance the audibility of sounds associated to interacting agents. We thus define the amplitude $A_i(t)$ of the oscillator associated to each agent i as a function of the number of neighbors $\tilde{n}_i$ within its sonic interaction range R_S, through the expression

$$A_i(t) = \frac{\tilde{n}_i A_{\max} + A_{\min}}{\tilde{n}_i + 1}. \tag{7.6}$$

With this definition, $A_i(t)$ is equal to $A_{\min}$ when agent i has no neighbors and grows with $\tilde{n}_i$, saturating at $A_{\max}$ when it has many neighbors.

The coupled oscillators approach produces sounds that are harder to identify with specific spatial dynamics, but that still clearly reflect, indirectly, the emergence of polarization order. In the disordered phase, neighbors are coming together and separating so often that no consensus frequency appears. The result is a cacophony of voices (as long as $A_{\min}$ is large enough to make these audible), each with its own $\omega_i(t) \approx \omega_i^0$, which also display rapid frequency fluctuations driven by random encounters of small groups of agents. In the ordered phase, by contrast, agents typically spend longer time within a cluster, and therefore with the same neighbors. The frequencies of their corresponding oscillators will have more time to converge, even if clusters are still constantly splitting and merging, which can produce substantial sporadic variations of $\omega_i(t)$ over time. The interplay of all these effects yields more interesting and appealing musical results than in the direct approach, both in the ordered and disordered states.

Our exploration of the coupled oscillators approach is summarized in Fig. 7.4. It presents runs with $R_S = 2R$, $\beta_1 = 0.78$, $\beta_2 = 0.30$, and $\eta = 0.85$ or $\eta = 0.03$. As in Fig. 7.2, panels a and b display snapshots of the agent dynamics and panels c and d, spectrograms of the resulting sound in the disordered and ordered phases, respectively. In the disordered state (a and c), each agent spends most of its time in small clusters or isolated. The spectrogram shows several horizontal lines corresponding to their randomly selected preferred frequencies ω_i^0. These are interrupted by quick frequency variations due to brief agent encounters. The resulting sounds are perceived as a layer of quiet, broadly distributed dissonant tones with several louder superimposed trickling-like pitch fluctuations. When the noise is decreased and agents form more coherent clusters (b and d), we observe and hear in this run a loud stable note containing frequencies around 1.5 kHz, which corresponds to a persistent isolated cluster. In addition to this, there are a range of fluctuating sounds of lower amplitude, at frequencies below 1.0 kHz, which result from other less persistent clusters and some isolated agents. These produce interesting dynamics as the clusters split and merge. When a splitting event occurs, for example, the smaller component or single particle that left the main cluster will jump to its preferred

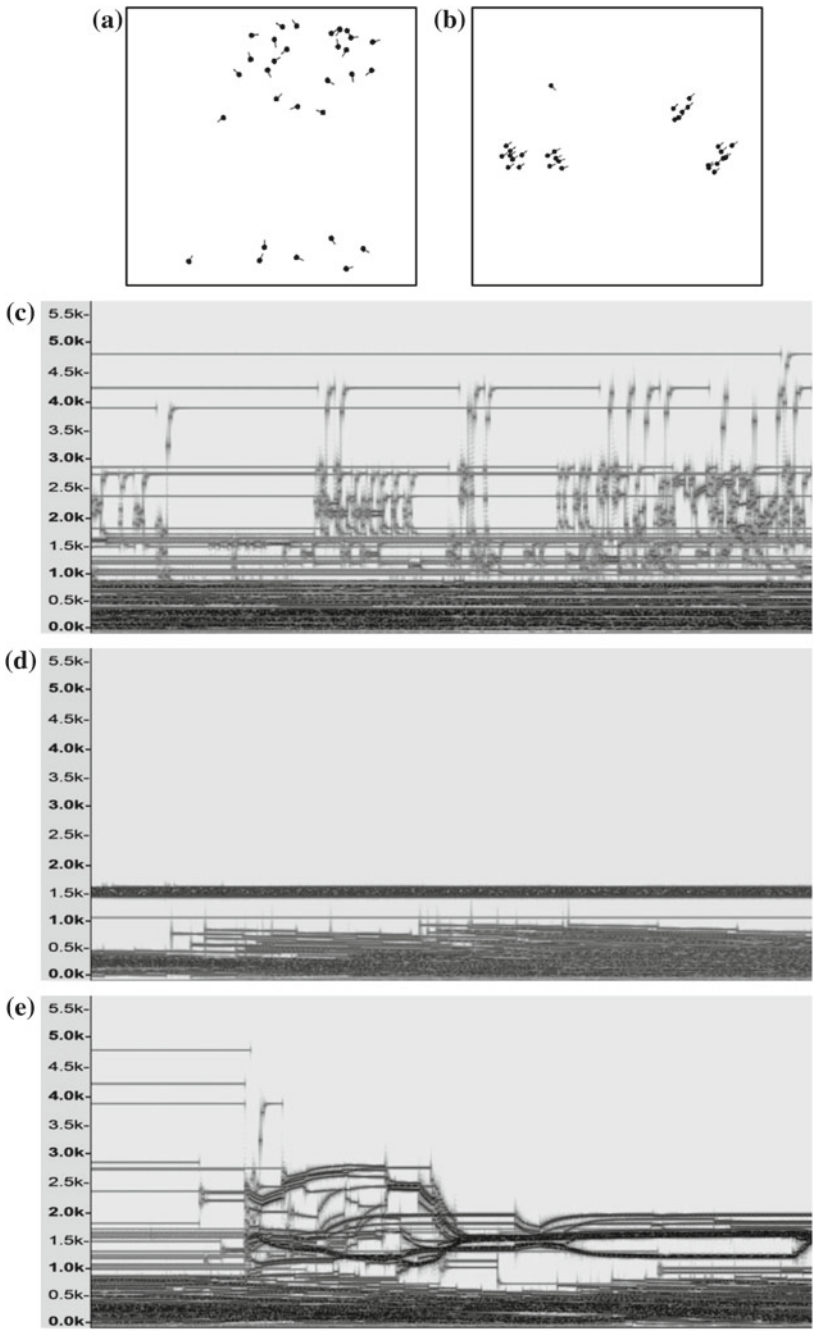


Fig. 7.4 Mapping from flocking dynamics to sound using the coupled oscillators approach. We present the same corresponding snapshots and spectrograms as in Fig. 7.3. **a** and **c** Disordered state. **b** and **d** Ordered state. **e** Transition from disordered to ordered state

frequency very rapidly, while the larger cluster will take much longer to reach its new mean cluster frequency. Finally, panel e displays a transition in which the amount of noise in the system is reduced during the run. We observe and hear that, as the lower noise level shifts the system from a disordered to an ordered state, particles can form larger and more persistent clusters which drive their frequencies closer. However, clusters still split and merge randomly, thus producing successive, musically interesting convergences, and divergences of frequencies, which can be seen later on the same spectrogram.

7.4.3 The Physical Friction Approach

In this approach, the aim is to implement an algorithm that captures elements of what we could intuitively associate to the physical response of a group of interacting moving particles. The purpose is to take advantage of our evolved ability to connect sounds to their physical sources, that is, to the objects and processes that produce them.

In order to construct a minimal algorithm that yields a physics-inspired sonic response, we consider an artificial, friction-like process that produces sounds as agents "rub" against each other. While in some physical processes (such as the drawing of a bow against violin strings) the frequencies of the sounds produced depend only on the fundamental oscillation modes of the participating objects, in others (such as the friction within a rotating motor), they depend on the relative speed of the moving parts. Here, we base ourselves on the latter type of processes and make the frequency of the sound generated by each agent proportional to its relative speed with respect to the moving average position of its neighbors. We thus define the frequency produced by agent i as

$$\omega_i(t) = \frac{1}{2v_0} \left\| \left[\frac{1}{\tilde{n}_i} \sum_{j \in S_i} v_j(t) \right] - v_i(t) \right\| \tag{7.7}$$

Here, v_0 is the constant particle speed imposed in the Vicsek dynamics, $v_i(t)$ and $v_j(t)$ are the corresponding velocities, and S_i is the set of $\tilde{n}_i$ indexes of all agents within the sonic interaction range R_S of agent i at a given time. With this definition, $\omega_t(t)$ takes values between 0 and 1. It can then be mapped linearly or exponentially to a range of sound frequencies, as in the previous algorithms. We complete the definition of the physical friction approach by using for the sinusoidal wave associated to each particle i the same amplitude $A_i(t)$ that was defined in Eq. (7.6) and used in the coupled oscillators approach. This amplitude increases with the number of neighbors, starting from $A_{\min}$ for $\tilde{n} = 0$ and reaching the saturation value $A_{\max}$ for large $\tilde{n}$.

Figure 7.5 displays examples of the results obtained with the physical friction approach for $\eta = 0.61$ and $\eta = 0.09$. Here, the disordered state (panels a and c) produces a random sequence of short sounds of all possible frequencies. This results

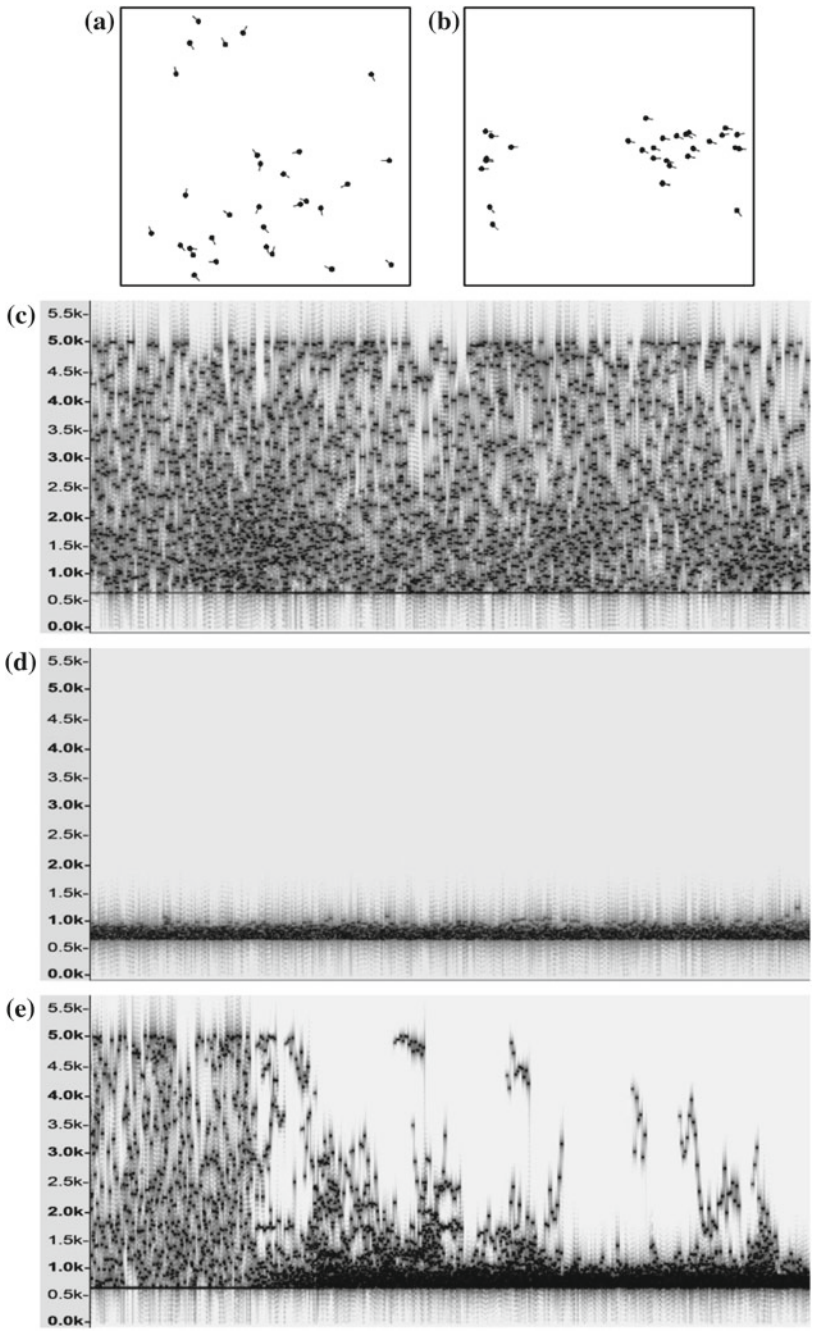


Fig. 7.5 Mapping from flocking dynamics to sound using the physical friction approach. We present the same corresponding snapshots and spectrograms as in Fig. 7.3. **a** and **c** Disordered state. **b** and **d** Ordered state. **e** Transition from disordered to ordered state

from the many random encounters between particles and gives an intuitive sonic impression of the dynamics, especially if one initially listens to it while viewing the simulations. For runs in the ordered state (panels b and d), we observe (and hear) very different dynamics. Almost all oscillators approach the $\omega_i(t) = 0$ state, here set to around 700 Hz, which corresponds to zero relative velocity. In other runs, the oscillator frequencies organized into different groups of low ω, corresponding to clusters of different sizes. This is due to the fact that more closely packed regions (usually related to specific clusters) will have lower mean relative velocity between its components, because the higher number of interactions will better suppress fluctuations. This can be seen on panel e, which displays a transition from the disordered state to the ordered state, obtained by changing the noise level during the run. Starting from a disordered state, we can observe (and hear) the transition as the suppression of the initial random short notes while they shift toward the $\omega_i(t) = 0$ state. As the system gets ordered, there are often "sparkles" of groups of notes that appear on different parts of the spectrum. This corresponds to cluster fragmentations, which can produce significant relative velocities, and to smaller groups that display larger internal fluctuations. The combination of all these dynamical effects often generates sounds with rich textures that can be musically appealing.

7.5 From Emergent Dynamics to Emerging Music

We now present an example of the artistic possibilities of our approach by describing a musical piece generated through flocking dynamics. *Ritmos Circadianos* is a piece for robot orchestra, composed in 2012 by Cádiz (one of the co-authors of the current study), based on the concept of circadian rhythms and generated entirely using an extension of one of the flocking-based music algorithms described above. This work was specifically composed for the Logos Foundation's Man and Machine Robot Orchestra [38], on the occasion of the *World Music Days Festival of Contemporary Music* that took place in several cities in Belgium on October of 2012.

The Man and Machine Robot Orchestra, based in Ghent, features over 50 robotic musical instruments, including organ-like instruments, monophonic wind instruments, string instruments, percussion instruments, and noise generators. All were designed and built by Godfried-Willem Raes, a Belgium composer and inventor, who has been dedicated to this task for over 25 years. Several of these robots are shown on Fig. 7.6. The Logos Foundation started in 1968 as a collective of experimental composers and musicians. According to [38], the motivation for Logos' interest and involvement in robotics stems from the view that loudspeakers as sound sources are mere virtualizations of an acoustic reality. Therefore, they undermine concerts as social rituals. The dissociation between the musicians' gestures and sonic results takes away the rhetoric that characterizes live performances. In contrast, by using automated acoustic instruments, music flows from concrete physical sound sources under precise computer control, making explicit the coupling between gestures and produced sounds.

Fig. 7.6 Composer Rodrigo F. Cádiz working with the logos man and machine robot Orchestra in the tetrahedron concert hall at the logos foundation in Ghent, Belgium

In Ritmos Circadianos, the robots go through several cyclical rhythmic patterns similar to the circadian rhythms experienced by humans and other living systems. The term circadian comes from the Latin words *circa* and *diem*, meaning "approximately" and "day", respectively. A circadian rhythm is any biological process that displays an endogenous, entrainable cycle of approximately 24 h. These rhythms are driven by an internal circadian clock. They have been widely observed in plants, animals, fungi, cyanobacteria, and humans. Although circadian rhythms are endogenous, they adjust to the cycles of the local environment through external cues, the most common of which is daylight. Well-known examples of human circadian rhythms are the sleep-wake cycle and the body-temperature cycle, which can be modeled using networks of coupled oscillators [39]. The flocking algorithm and coupled oscillators approach described in previous sections are used in Ritmos Circadianos to generate and control in real-time the dynamical behavior of the musical material performed by the robot orchestra. The state of each flocking agent is coupled to the states of other agents in its neighborhood, which corresponds to having each one adjust to the circadian rhythms of others. Making the robots experience a human-like clock and rhythm constitutes the gist of the composition.

Ten pitched and two percussion instruments of the robot orchestra were chosen for this composition. These were: *Heli*, an automated helicon; *Llor*, consisting of 12 large automated stainless steel shell-bells; *Sax*, an automated sax; *Klar*, an automated clarinet; *So*, an automated tuba; *Vibi*, an automated vibraphone; *Qt*, a quartertone organ; *Vox*, an automated pipe organ; *Fa*, an automated bassoon; *Vacca*, consisting of 48 automated cow bells; *Snar*, an automated snare drum; and *Troms*, a set of

automated drums. Each instrument was linked to one agent of a Vicsek flocking simulation with twelve agents. The variables that dictated the performance of each instrument were directly derived from the outputs of the simulations, utilizing the coupled oscillators approach described in Sect. 7.4.2. The performance is generated in real-time by running the flocking model on a MaxMSP patch that controls the orchestra using the MIDI protocol, through a 64-channel MIDI interface, as can be seen in Fig. 7.6. The piece lasts for 10 min and 24 s and is structured around 26 sections of 24 s each, producing a direct analogy between the length of each section and a 24 h circadian cycle. Each section lasts exactly ten bars, with a tempo of 100 beats per minute. For each section a different combination of model parameters and instruments was used, according to the information listed in Table 7.1.

The pitch material of the composition is derived from the correspondence between the ω_i wave frequency of each agent in the swarming model and the pitch range of each robot instrument. The ω_i wave frequencies are converted from a continuous range in Hertz to quantized MIDI notes, according to each instrument's range. This implies that whenever all agents in the flock are coupled (and, therefore, all their oscillator frequencies are constant and almost the same), each instrument of the orchestra will produce a fixed tone, but these could be very different in pitch for each instrument, depending on their ranges. The sound intensity of each melodic sequence is derived from the spatial coordinates x_i, y_i of each agent. When an agent approaches the center of the navigation space, the intensity of the tones produced by its corresponding robot increase. As the agents move away from the center, their corresponding intensities decrease.

In terms of the rhythmic material, very complex rhythmic patters can be produced by each agent's behavior, which is probably the most interesting aspect of this piece from a musical viewpoint. Each instrument of the orchestra experiences a particular rhythmic pattern derived from the state of its corresponding agent. The basic rhythmic pattern of each instrument is given by the superposition of up to three different regular pulses that turn on the MIDI notes (and then let the instrument follow its natural decay curve). The number of pulses superimposed for each instrument and the range of their temporal periods are specified by the composer. These temporal periods are rescaled independently within these ranges by a factor that depends on the current state of its corresponding agent. More specifically, on its $\omega_i(t)$ and on: the absolute value of the difference between its heading angle and the mean heading angle of all agents for the first regular pulse, its x_i position for the second one, and its y_i position for the third one. A master clock runs through the entire piece to give it an overall structural coherence, producing a synchronizing event that resets all pulses to the same starting point every 24 s cycle, at the beginning of each section. The combination of all these rhythmic patterns results in each instrument displaying different degrees of regularity. If the angle of an agent is not very different from the mean angle, its pattern will regularize. But in the opposite case (that typically occurs for high angular noise levels) its pattern becomes very irregular. Note that the relationship between the rhythmic pattern, pitch, and $\omega_i(t)$ of each agent can produce very interesting poly-rhythmic dynamics and pitch textures when several instruments play at the same time. In contrast, when only a few instruments are

Table 7.1 Parameters and instruments used in the 26 sections of *Ritmos Circadianos*

Section	Model parameters					Instrumentation
	r	η_p	η_a	β_1	β_2	
1	8	0	0	0.2	0.3	Heli, Llor
2	8	0.1	0.01	0.2	0.3	Heli, Sax, Llor, Klar, So
3	3.1	0.1	0.2	0.2	0.3	Heli, Sax, Llor, Klar, So
4	3.1	0.1	0.2	0.2	0.3	Heli, Vibi, Sax, Qt, Vox, Klar, So
5	2.1	0.5	0.6	0.6	0.2	Heli, Vibi, Sax, Qt, Vox, Llor, Klar, So
6	2.1	0.5	0.6	0.6	0.2	Fa, Qt, Klar, So
7	2.1	0.5	0.6	0.6	0.2	Sax, Klar
8	2.1	0.8	0.8	0.5	0.5	Vibi, Vox, So, PERC
9	7.1	0.1	0.2	0.5	0.5	Sax, Fa, Klar, So
10	7.1	0.05	0	0.5	0.5	Vibi, So
11	8.1	0.05	0.3	0.5	0.5	Vibi, Qt
12	8.1	0.45	0.3	0.5	0.5	Vacca, Vibi
13	3.1	0.85	0.7	0.5	0.5	Vacca, Vibi
14	8.1	0.45	0.3	0.5	0.5	Vacca
15	8.1	0.45	0.3	0.5	0.5	Vacca, Vibi, PERC
16	3.1	0.45	0.3	0.5	0.5	Heli
17	8.1	0.01	0.01	0.5	0.5	Heli, Fa
18	7.1	0.05	0.03	0.9	0.1	Heli, Vibi, So
19	8.1	0.01	0.01	0.5	0.5	Heli, Vacca, Vibi, Sax, Fa, Vox, Klar, So
20	8.1	0.01	0.01	0.5	0.5	Vox
21	8.1	0.01	0.01	0.5	0.5	Heli, Vacca, Vibi, Sax, Fa, Vox, Klar, So, PERC
22	1.1	0.81	0.81	0.3	0.4	Vacca, PERC
23	1.1	0.81	0.81	0.3	0.4	Vacca, Sax, Fa, Vox, Klar, PERC
24	1.1	0.51	0.51	0.3	0.4	Heli, Vacca, Vibi, Sax, Fa, Qt, Vox, Llor, Klar, PERC
25	4.1	0.15	0.23	0.4	0.5	Heli, Vibi, Llor, So
26	8	0	0	0.2	0.3	Heli, Llor

Several parameters were changed from section to section in order to generate different musical results: the radius of interaction r, the position noise η_p, the angle noise η_a, the convergence rate β_1 and the preferred frequency weight β_2. The instrumentation was also varied for each section as listed. PERC denotes the usage of the percussion instruments *Troms* and *Snar*

overlapped, more clear and stable rhythmic patterns tend to emerge. All the elements of the composition process described above are summarized as a block diagram on Fig. 7.7.

Figure 7.8 shows a musical score representation of the last two measures of section 24 and the first three of section 25. In section 24, several layers of rhythmic complexity and fast changing melodies can be observed. This is due to the relatively high levels of position and angular noise that the model has for that section, as shown in Table 7.1. In contrast, section 25 presents more stable melodies with low rhythmic activities, mainly due to the low levels of noise. In this section, the radius of influence is considerably higher than in the previous one, resulting in a more homogeneous overall texture. In this example, both sections contain similar amounts of convergence rate and preferred frequency weight factors, which implies that each robot tends to reach its natural frequency in a similar fashion and timescale.

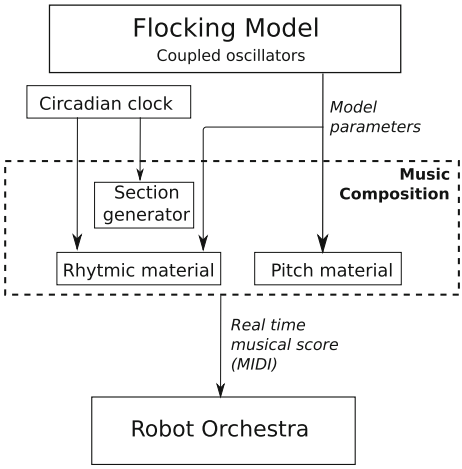


Fig. 7.7 Block diagram of the composition process for *Ritmos Circadianos*. The parameters of the flocking model determine the pitch and rhythmic material. A circadian clock with a cycle of 24 s is used to generate the different sections of the composition and the rhythmic patters performed by the robot instruments

As reviewed by the press [40], the premiere of this work was very successful. The behavior of the flocking model resulted in engaging sonic performances by the robot orchestra. Its phase transitions generated clearly audible contrasts in the musical flow. This, combined with the inclusion of different rhythmic patterns based on circadian rhythms, produced an innovative and appealing piece of music. Since the flocking model contains a controllable amount of noise, each performance was slightly different, while exhibiting similar parts and transitions.

7.6 Music and Complex Systems

Flocking algorithms seem to be a natural starting point for linking complex systems and music. The individual or collective motion of independent interacting agents moving in space resembles in many ways the dynamics of performers navigating a large set of musical possibilities as they play together, or of the various components of a musical piece combining to achieve a coherent musical result. By exploring different mappings between motion in physical space and dynamics in these abstract musical spaces, we can generate a broad variety of sounds, with different frequencies, rhythms, intensities, and textures. We have shown above that even the first steps of this exploration can yield interesting results. These include: establishing an intuitive connection between flocking dynamics, mapping algorithms, and their emergent sounds; finding features of complex dynamical systems that can translate into interesting musical results; and understanding how characteristic emergent properties, such as the order–disorder transition in the Vicsek model explored here, will be reflected in the resulting sounds.

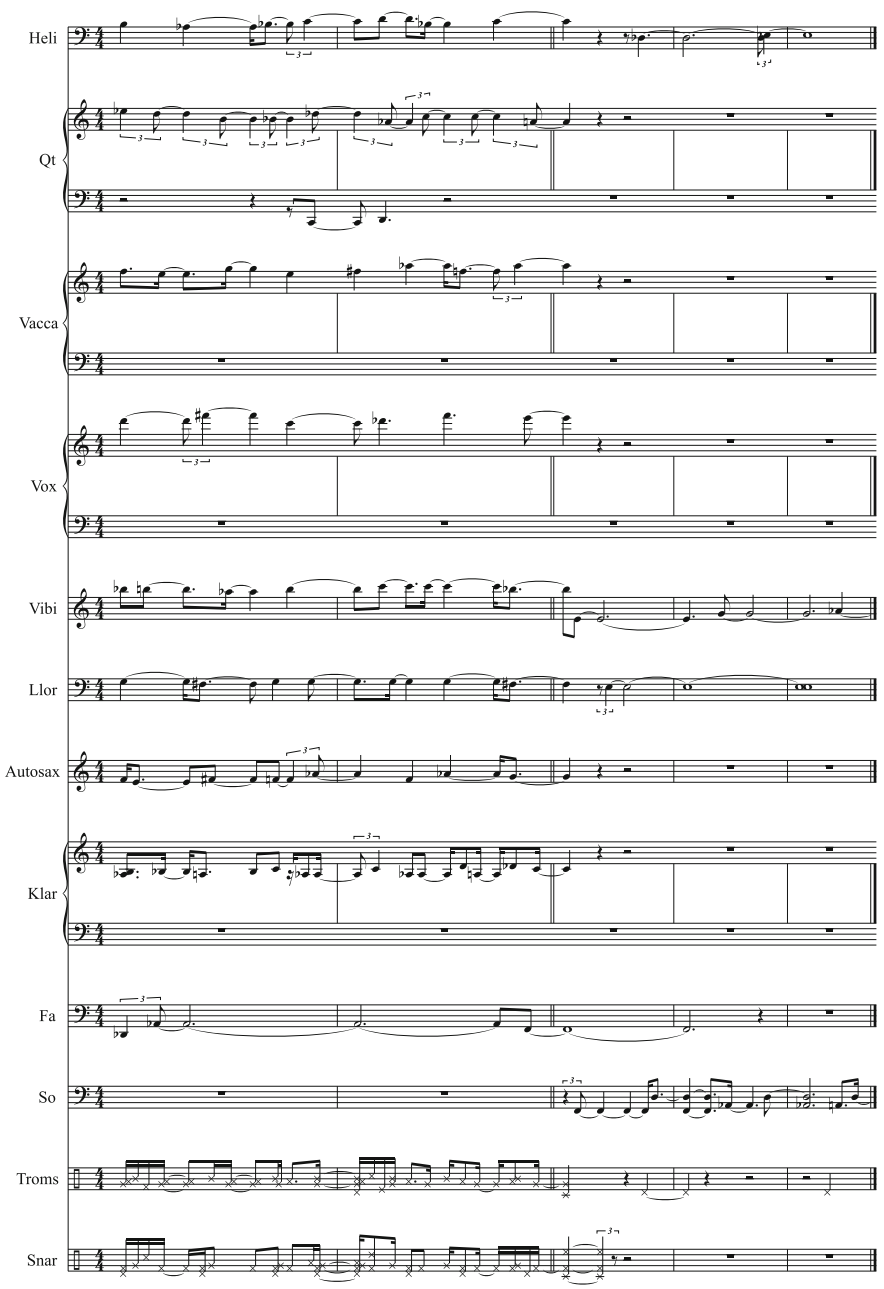


Fig. 7.8 Musical score of *Ritmos Circadianos*, showing the transition from section 24 to section 25. The figure shows the last two measures of section 24 and the first three of section 25

An interesting question that follows from this work is if our exploration can bring, in turn, new insights to the scientific and engineering communities that study collective motion in biological and robotic systems. We argue that this is indeed the case. At a practical level, the use of our brain's evolved ability to extract underlying structures from sounds has been underutilized as a tool for examining data. Our work could lead to new sonification approaches either by providing specific mapping alternatives or by helping to develop intuition on how to translate complex dynamics into meaningful sonic information. At a conceptual level, our work could also help inspire novel approaches for characterizing flocks and collective motion. For example, the results obtained using the coupled oscillators and physical friction approaches show that sounds based only on the dynamics of encounters between agents allow us to clearly distinguish the ordered and disordered phases, thus suggesting the use of statistics based only on these encounters as novel order parameters. Furthermore, the search for dynamics that produce appealing musical results forces us to consider a new set of factors in the design and control of flocking systems in general.

Beyond the possibility of establishing specific connections between music, flocking dynamics, and control theory, a broader fascinating question is the relationship between music and complex systems in general. Although music has often been related to science through acoustics, mathematical rhythmic or harmonic structures, sound synthesis, and music-generating algorithms, the emergence of the science of complexity (supported by powerful and ever-growing computational and data analysis tools) allows us to develop a somewhat new perspective in which music can be viewed in the context of self-organization and emergence. The connection between these fields appears at multiple levels. At the individual perception scale, it is reflected in the neurological processes that are involved in producing specific effects (emotional, psychological, physiological, etc.) in a given listener. At the music creation level, in the combination of multiple sonic components required to produce a desired result. At the appreciation level, in the complex network of social interactions that underlies the collective evaluation of a musical piece and the evolution of musical styles. From this perspective, different aspects of the musical experience could be naturally described through the language of complex systems, self-organization, and emergence.

A further connection between music and complex living systems is suggested by the concept of criticality. There has been a long-standing argument proposing that organisms must evolve to be close to a critical state [41, 42]. In recent years, various experimental analyses have shown critical signatures in the dynamics of different biological systems [43, 44], including the brain [45, 46]. The reasoning that leads to this argument can be summarized as follows. For a living system to survive evolutionary pressures it must be able to quickly adapt to changes in the environment by exploring new alternatives. On the other hand, it must also be robust enough to stop perturbations from destroying its essential biological features. These conflicting constraints suggest that living systems must evolve toward dynamics that are "at the edge of chaos" [47], which are able to remain close to strong attractors when needed, but also to rapidly switch to chaotic trajectories that allow them to explore different adaptations to address a broad range of challenges. Given these general arguments,

it is reasonable to hypothesize that music must follow similar constraints in order to be attuned to living beings. This may be the reason why appealing music must have the combination of coherence and decoherence or predictability and surprise that we described in our Introduction.

In this context, the analysis that we carried out here can be viewed as a first step that explores how music and complex systems come together using a specific flocking algorithm. Even at this initial stage, we observed that a diversity of mapping strategies can allow the listener to easily detect the order or disorder in the system through its sonic impression. We have also shown that indirect measures of the flocking dynamics (as used in the coupled oscillators approach) or those that appeal to our real-world intuition (as in the physical friction approach) often produce more interesting and appealing musical results. Furthermore, the successful creation of the musical composition Ritmos Circadianos demonstrates the artistic possibilities of our perspective.

The work presented here is by no means exhaustive; a number of alternative ways of mapping flocking to sound could be considered. A natural extension of the coupled oscillators approach would be to implement a Kuramoto model that couples the oscillators of neighboring agents [48]. This model has been used to describe the synchronization of clapping [49] and for human–machine music performance interactions [50]. However, to fully implement its dynamics, the phase of each agent would have to be computed as fast as the audio sampling rate for all agents. This process is time-consuming and would not allow a real-time interactive exploration of the parameter space with the current software tools. Another possible extension would be to consider a different physics-based approach. An interesting possibility is to compute the oscillations that a cluster would produce as a whole if interpreted as an elastic membrane. This is similar to the approach in [37], where it is found that the relationship between swarm simulations and sound generation is characterized by a certain conclusiveness, which stems from the fact that they are both based on a model of the same physical phenomenon: the dynamics of an elastic object. This approach would also require the development of new software tools to be able to produce sound in real-time.

As suggested by the discussion above, our work could lead to a broad range of applications. Given the connections between biology, complex systems, and music, these could include: searching for critical features or other signatures in music that can be correlated to biological dynamics, using these connections to inspire musical compositions based on complex dynamical systems, or taking advantage of our intuitive ability to extract features from music to develop sonification approaches for data analysis.

Acknowledgments The work of C. H. was supported by the U.S. National Science Foundation under Grant No. PHY-0848755. The work of R. F. C. and M. C. was supported by Fondecyt under Grant No. 11090193 and by the Fondo de Fomento de la Música, Consejo Nacional de la Cultura y las Artes, under grant No. 15872-0, Government of Chile.

References

1. Mazzola G, Park J, Thalmann F (2011) Musical Creativity: strategies and tools in composition and improvisation. Springer, Heidelberg
2. Panish J (1997) The color of jazz: race and representation in postwar American culture. University Press of Mississippi, Jackson
3. Barskii V (1996) Chromaticism. Harwood Academic Publishers, Amsterdam
4. Olfati-Saber R, Murray R (2004) Consensus problems in networks of agents with switching topology and time-delays. IEEE Trans Autom Control 49(9):1520–1533
5. Blondel VD, Hendricks JM, Olshevsky A, Tsitsiklis JN (2005) Convergence in multiagent coordination, consensus, and flocking. In: Proceedings of joint 44th IEEE conference on decision control Eurupean Control Conference, Seville, Spain, pp 2996–3000
6. Scardovi L, Leonard N, Sepulchre R (2007) Stabilization of collective motion in three dimensions: a consensus approach. In: Proceedings of the 46th IEEE conference on decision and control, New Orleans, LA, pp 2931–2936
7. Jadbabaie A, Lin J, Morse AS (2003) Coordination of groups of mobile autonomous agents using nearest neighbor rules. IEEE Trans Autom Control 48(6):988–1001
8. Freeman RA, Yang P, Lynch KM (2006) Distributed estimation and control of swarm formation statistics. In: Proceedings of the American control Conference, Minneapolis, MN, pp 749–755
9. Sepulchre R, Paley D, Leonard N (2008) Stabilization of planar collective motion with limited communication. IEEE Trans Autom Control 53(3):706–719
10. Gueron S, Levin SA, Rubenstein DI (1996) The dynamics of herds: From individuals to aggregations. J Theoret Biol 182:85–98
11. Couzin ID, Krause J, James R, Ruxton GD, Franks NR (2002) Collective memory and spatial sorting in animal groups. J Theoret Biol 218(1):1–11
12. Couzin ID, Krause J, Franks NR, Levin SA (2005) Effective leadership and decision-making in animal groups on the move. Nature 433(7025):513–516
13. Buhl J, Sumpter DJT, Couzin ID, Hale JJ, Despland E, Miller ER, Simpson SJ (2006) From disorder to order in marching locusts. Science 312(5778):1402–1406
14. Codognet P, Pasquet O (2009) Swarm intelligence for generative music. In: 11th IEEE international symposium on multimedia, IEEE Computer Society Washington, DC, pp 1–8
15. Brady R, Bargar R, Choi I, Reitzer J (1996) Auditory bread crumbs for navigating volumetric data. In: Proceedings of the late breaking hot topics of IEEE visualization '96, San Francisco, CA, pp 25–27
16. Kaper HG, Tipei S, Wiebel E (1999) Data sonification and sound visualization. Comput Sci Eng 1(4):48–58
17. Cullen C, Coyle E (2005) TrioSon: a graphical user interface for pattern sonification. 11th meeting of the international conference on auditory display., ICAD 05Limerick, Ireland, pp 6–9
18. Yeo WS, Berger J, Lee Z (2005) SonART: a framework for data sonification, visualization and networked multimedia applications. In: Proceedings of the 2004 international computer music conference, ICMC 2004, Miami, FL
19. Davis T, Rebelo P (2005) Hearing emergence: towards sound-based self-organisation. In: Proceedings of international computer music conference, Barcelona, Spain
20. Reynolds C (1987) Flocks, herds, and schools: a distributed behavioural model. SIGGRAPH' 87 21(4):25–34
21. Spector L, Klein J (2002) Complex adaptive music systems in the Breve simulation environment. In: Proceedings of the 8th international conference on the simulation and synthesis of living systems, Artificial Life VIII
22. Tang E, Shiffman D (2003) Musical flocking box, http://www.antiexperience.com/edtang/works/flockingbox.html

23. Blackwell T (2007) Swarming and music. In: Reck Miranda E, Biles JA (eds) Evolutionary computer music. Springer, Berlin, pp 194–217
24. Davis T (2011) Complexity as practice: a reflection on the creative outcomes of a sustained engagement with complexity. Leonardo 45(2):106–112
25. Unemi T, Bisig D (2005) Music by interaction among two flocking species and human. In: Proceedings of the third international conference on generative systems in electronic arts, Melbourne, Australia, pp 171–79
26. Davis T (2010) Complexity as process: complexity inspired approaches to composition. Organised Sound 15(2):137–146
27. Vicsek T, Czirók A, Ben-Jacob E, Cohen I, Shochet O (1995) Novel type of phase transition in a system of self-driven particles. Phys Rev Lett 75(6):1226–1229
28. Huepe C, Aldana M (2004) Intermittency and clustering in a system of self-driven particles. Phys Rev Lett 92(16):168701
29. Puckette M (1988) The patcher. In: Proceedings of the 14th international computer music conference, Koln, Germany, pp 420–429
30. Zicarelli D (1998) An extensible real-time signal processing environment for max. In: Proceedings of the international computer music conference, Ann Arbor, Michigan, pp 463–466
31. Davis T, Karamanlis O (2007) Gestural control of sonic swarms: composing with grouped sound objects. In: Proceedings of the SMC'07., 4th sound and music computing conference Lefkada, Greece, pp 192–195
32. Blackwell T, Young M (2004) Swarm granulator. In: EvoWorkshops 2004, Coimbra, Portugal, pp 399–408
33. Blackwell TM, Bentley P (2002) Improvised music with swarms. In: Proceedings of congress on evolutionary computation, Piscataway, NJ, pp 1462–1468
34. Kim-Boyle D (2005) Sound spatialization with particle systems. In: Proceedings of the 8th international conference on digital audio effects (DAFX-05), Madrid, Spain, pp 65–68
35. Bisig D, Neukom M, Flury J (2007) Interactive swarm orchestra. In: Proceedings of the generative art conference, Milano, Italy
36. Wilson S (2008) Spatial swarm granulation. In: Proceedings of the international computer music conference, Belfast, Northern Ireland
37. Schacher JC, Bisig D, Neukom M (2011) Composing with swarm algorithms—creating interactive audio—visual pieces using flocking behaviour. In: Proceedings of the international computer music conference, University of Huddersfield, UK
38. Laura M, Godfried-Willem R, Troy R (2011) The man and machine robot orchestra at logos. Comput Music J 35(4):28–48
39. Strogatz SH (1987) Human sleep and circadian rhythms: a simple model based on two coupled oscillators. J Math Biol 25:327–347
40. Ramírez JCF (2013) Chileno lanza sorprendente concierto para orquesta de robots, La Segunda, Santiago, Chile. 15 Jan 2013 http://www.lasegunda.com/Noticias/Impreso/2013/01/814245/chileno-lanza-sorprendente-concierto-para-orquesta-de-robots Accessed 25 Sep 2013
41. Kauffman S (1993) The origins of order: self-organization and selection in evolution. Oxford University Press, Oxford
42. Bak P (1999) How nature works: the science of self-organized criticality. Copernicus, New York
43. Balleza E, Alvarez-Buylla ER, Chaos A, Kauffman S, Shmulevich I, Aldana M (2008) Critical dynamics in genetic regulatory networks: examples from four kingdoms. PLoS ONE 3(6):e2456
44. Mora T, Bialek W (2011) Are biological systems poised at criticality? J Stat Phys 144(2):268–302
45. Ribeiro TL, Copelli M, Caixeta F, Belchior H, Chialvo DR, Nicolelis MAL, Ribeiro S (2010) Spike avalanches exhibit universal dynamics across the sleep–wake cycle. PLoS ONE 5(11):e14129
46. Plenz D (2013) The critical brain. Physics 6:47

47. Lewin R (2000) Complexity: life at the edge of chaos. The University of Chicago Press, Chicago
48. Strogatz S (2000) From Kuramoto to Crawford: exploring the onset of synchronization in populations of coupled oscillators. Physica D 143(1–4):1–20
49. Peltola L (2004) Analysis, parametric synthesis, and control of hand clapping sounds. Master of Science Thesis, Helsinki University of Technology
50. Mizumoto T, Otsuka T, Nakadai K, Takahashi T, Komatani K, Ogata T, Okuno HG (2010) Human-robot ensemble between robot thereminist and human percussionist using coupled oscillator model. In: Proceedings of the 2010 IEEE/RSJ international conference on intelligent robots and systems, Taipei, Taiwan, pp 1957–1963

Chapter 8
Algorithms for Visual Tracking of Visitors Under Variable-Lighting Conditions for a Responsive Audio Art Installation

Andrew B. Godbehere and Ken Goldberg

8.1 Installation Concept and Visitor Experience

The responsive audio art installation "Are We There Yet? : 5,000 Years of Answering Questions with Questions" was on exhibit at the Yud gallery of the Contemporary Jewish Museum in San Francisco, CA from March 31–July 31, 2011. Video and documentation of the project was archived at http://are-we-there-yet.org. The image in Fig. 8.1 depicts a view of the Yud Gallery from its entrance and the illustration in Fig. 8.2 depicts a schematic of the installation design.

Conceived by Ken Goldberg and Gil Gershoni, visitors enter the gallery and encounter soaring walls and windows carving the space with sunlight. A voice asks: "Can we talk?" As visitors move, they discover that the sound moves with them. "Are you experienced?" Deeper in the gallery, questions become more abstract, "Who is a Jew?", "Is patience a virtue?", "Is truth a matter of perspective?" Each visitor creates their own unique experience as questions take on new contexts and meanings.

The installation evoked an early stage in Jewish history when the open desert created opportunity for revelation; it presented new perspectives on the fundamental search and questioning that lies at the heart of Jewish identity. The Yud Gallery, designed by Daniel Libeskind and depicted in Figs. 8.1 and 8.2, acknowledges the second Commandment, emphasizing the auditory over the visual. Localized speakers were strategically positioned to preserve the open gallery space. Visitors experienced the work as a responsive sequence of questions that encouraged movement throughout the space to explore, discover, and consider.

The project was also inspired by the Talmudic representation of multilayered Jewish intellectual discourse. The Talmud is a surprisingly contemporary model for communal conversation in the digital age. Rather than resolving each issue with an authoritative unified "answer," each page of the Talmud reflects the spiraling layers

A. B. Godbehere (✉) · K. Goldberg
Department of EECS and IEOR, University of California at Berkeley, Berkeley, CA, USA
e-mail: abg34@eecs.berkeley.edu

A. LaViers and M. Egerstedt (eds.), *Controls and Art*,
DOI: 10.1007/978-3-319-03904-6_8,

Fig. 8.1 The skylit atrium that was the site for this installation at the Contemporary Jewish museum in San Francisco, CA from April–July 2011

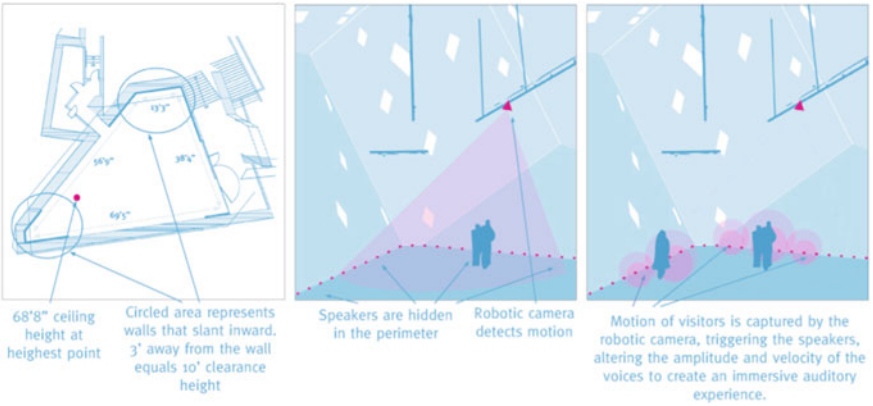


Fig. 8.2 Concept design for installation showing local speaker arrangement in gallery

of debate and celebrates the dissent at the heart of Jewish thought and tradition. Open inquiry is fundamental to electronic connectivity and social networking: the culture of new media encourages participation and a natural skepticism about the authenticity and authority of information.

Launched in advance of the exhibition, the "Are We There Yet?" companion website (http://www.are-we-there-yet.org) gave viewers the chance to learn more

about the exhibit, suggest their own questions for inclusion, and visually explore the suggestions of others. In the gallery space, a kiosk with a custom iPad interface and live, streaming interactive projection system gave visitors an opportunity to view and submit their own questions.

This chapter describes the computer vision system and a coupled interactive sound system we developed for the installation. A single fixed camera mounted on the ceiling of the gallery and custom algorithms were used to dynamically monitor the position of visitors in the space and to trigger hundreds of prerecorded audio files containing questions.

To address the challenges of constantly changing lighting conditions of the skylit gallery and background noise produced by the camera, we developed an adaptive statistical background subtraction algorithm. Outliers (foreground pixels) in each frame are defined statistically, and grouped together into connected components. Our system rejected small components while considering larger components as candidates for labeling as visitors. Dynamic information, based on interframe consistency (and the notion that visitors do not move very rapidly) is used to further reject noise.

8.1.1 Technical Overview

In the remainder of this chapter, we present details on the design of a computer vision system that separates video into "foreground" and "background", and subsequently segments and tracks people in the foreground while being robust to variable lighting conditions. Using video collected during the operation of the installation, under variable illumination created by myriad skylights, we demonstrate a marked performance improvement over existing methods in OpenCV 2.1. The system runs in real-time (15 frames per second), requires no training datasets or calibration (unlike feature-based machine learning approaches [50]), and uses only 2–5 s of video to initialize.

Our system consists of two stages: first is a probabilistic foreground segmentation algorithm that identifies possible foreground objects using Bayesian inference with an estimated time-varying background model and an inferred foreground model, described in Sect. 8.2. The background model consists of nonparametric distributions on RGB color-space for every pixel in the image. The estimates are adaptive; newer observations are more heavily weighted than old observations to accommodate variable illumination. The second portion is a multi-visitor tracking system, described in Sect. 8.3, which refines and selectively filters the proposed foreground objects. Selective filtering is achieved with a heuristic confidence model, which incorporates error covariances calculated by the multi-visitor tracking algorithm. We describe the tracking subsystem in Sect. 8.3. We apply a bank of Kalman filters [24] and match tracks and observations with the Gale-Shapley algorithm [19], with preferences related to the Mahalanobis distance under the estimated error covariance. Finally, a feedback loop from the tracking subsystem to the segmentation subsystem is introduced: the results of the tracking system selectively update the background image model,

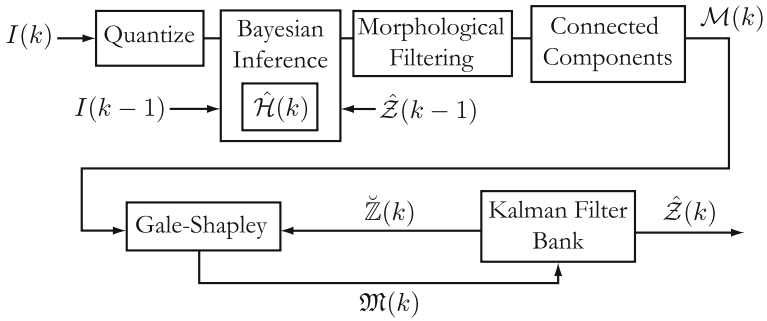


Fig. 8.3 Algorithm Block Diagram. An image $I(k)$ is quantized in color-space, and compared against the statistical background image model, $\hat{\mathcal{H}}(k)$, to generate a posterior probability image. This image is filtered with morphological operations and then segmented into a set of bounding boxes, $\mathcal{M}(k)$, by the connected components algorithm. The Kalman filter bank maintains a set of tracked visitors $\hat{\mathbb{Z}}(k)$, and has predicted bounding boxes for time k, $\breve{\mathbb{Z}}(k)$. The Gale-Shapley matching algorithm pairs elements of $\mathcal{M}(k)$ with $\breve{\mathbb{Z}}(k)$; these pairs are then used to update the Kalman Filter bank. The result is $\hat{\mathcal{Z}}(k)$, the collection of pixels identified as foreground. This, along with image $I(k)$, is used to update the background image model to $\hat{\mathcal{H}}(k+1)$. This step selectively updates only the pixels identified as background

avoiding regions identified as foreground. Figure 8.3 illustrates a system-level block diagram. Figure 8.4 offers an example view from our camera and some visual results of our algorithm.

The operating features of our system are derived from the unique requirements of an interactive audio installation. False negatives, or people the system has not detected, are particularly problematic because the visitors expect a response from the system and become frustrated or disillusioned when the response doesn't come. Some tolerance is allowed for false positives, which add audio tracks to the installation; a few add texture and atmosphere. However, too many false positives create cacophony. Performance of vision segmentation algorithms is often presented in terms of precision and recall [37]; many false negatives corresponds to a system with low recall. Many false positives lowers precision. We discuss precision, recall, and the F_2-score in Sect. 8.1.5.

Section 8.4 contains an experimental evaluation of the algorithm on video collected during the 4 months the system operated in the gallery. We evaluate performance with recall and the F_2-score [22, 30]. Our results on three distinct tracking scenarios indicate a significant performance gain over the algorithms in OpenCV 2.1, when used with the recommended parameters. Further, we demonstrate that the feedback loop between the segmentation and tracking subsystems improves performance by further increasing recall and the F_2-score.

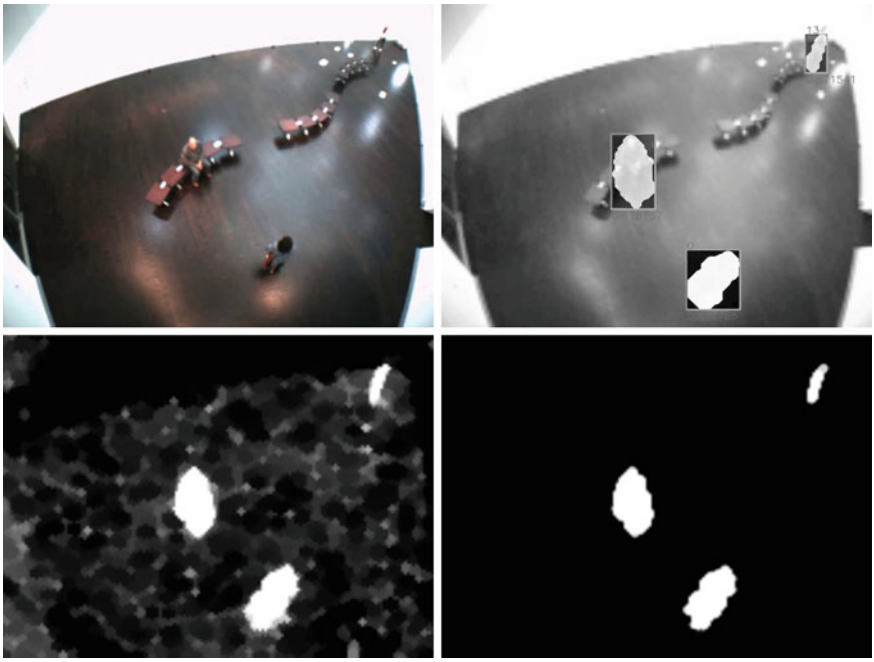


Fig. 8.4 Probabilistic Foreground Segmentation and Tracking Pipeline. *Upper Left* Raw image. *Lower Left* Posterior probability image. *Lower Right* Filtered and thresholded posterior image. *Upper right* Bounding boxes of tracked foreground objects and annotated confidence levels

8.1.2 Related Work

The OpenCV computer vision library [8, 10, 23, 28], offers a variety of probabilistic foreground detectors, including both parametric and nonparametric approaches, along with several multi-target tracking algorithms, utilizing, for example, the mean-shift algorithm [13] and particle filters [35]. Another approach applies the Kalman Filter on any detected-connected component, and does not attempt collision resolution. We evaluated these algorithms for possible use in the installation, although they exhibited low recall: visitors in the field of view of the camera were too easily lost, even while moving. This problem is due to the background model update method. Every pixel of every image is used to update the histogram, so pixels identified as foreground pixels are used to update the background model. The benefit is that a sudden change in the appearance of the background in a region is correctly identified as background; the cost is the frequent misidentification of pedestrians as background. To mitigate this problem, our approach uses dynamic information from the tracking subsystem to filter results of the segmentation algorithm, so only the probabilistic models associated with background pixels are updated.

The class of algorithm we employ is not the only class available for the problem of detecting and tracking pedestrians in video. A good overview of the various approaches is provided by Yilmaz et al. [54]. Our foreground segmentation algorithm is derived from a family of algorithms which model every pixel of the background image with probability distributions, and use these models to classify pixels as foreground or background. Many of these algorithms are parametric [12, 20], leading to efficient storage and computation. In outdoor scenes, mixture-of-gaussian models capture complexity in the underlying distribution that single Gaussian distribution models miss [23, 40, 46, 55]. Ours is nonparametric: it estimates the distribution itself rather than its parameters. For nonparametric approaches, kernel density estimators are typically used, as distributions on color-space are very high-dimensional constructs [16]. To efficiently store distributions for every pixel, we make a sparsity assumption on the distribution similar to [29], i.e., the random variables are assumed to be restricted to a small subset of the sample space.

Other algorithms use foreground object appearance models, leaving the background unmodeled. These approaches use support-vector machines, AdaBoost [18], or other machine learning approaches in conjunction with a training dataset to develop classifiers that are used to detect objects of interest in images or videos. For tracking problems, pedestrian detection may take place in each frame independently [2, 49]. In [36], these detections are fed into a particle-filter multi-target tracking algorithm. These single-frame detection approaches have been extended to detecting patterns of motion, and Viola et al. [50] show that incorporation of dynamical information into the segmentation algorithm improves performance. Our algorithm is based on different operating assumptions, notably requiring very little training data; initialization uses only a couple seconds of video.

A third, relatively new approach, is Robust-PCA [11], which neither models the foreground nor the background, but assumes that the video sequence I may be decomposed as $I = L + S$, where L is low-rank and S is sparse. The relatively constant background image generates a "low-rank" video sequence, and foreground objects passing through the image plane introduce sparse errors into the low-rank video sequence. Candès et al. [11] demonstrate the efficacy of this approach for pedestrian segmentation, although the algorithm requires the entire video sequence to generate the segmentation, so it is not suitable for our real-time application.

Generally, multi-target tracking approaches attempt to find the precise tracks that each object follows, to maintain identification of each object [7]. For our purposes, this is unnecessary, and we avoid computationally intensive approaches like particle filters [35, 36, 53]. Our suboptimal approximation of the true maximum likelihood multi-target tracking algorithm allows our system to avoid exponential complexity [7] and to run in real-time. Similar object-to-track matching utilizing the Gale-Shapley matching algorithm is explored in [5].

There is an emerging interest in applications of control algorithms to art [6, 21, 25–27, 44], as evidenced by this book.

Our work on analyzing human motion for esthetic ends is reminiscent of the flocking control analysis of Leonard et al "In the Dance Studio: An Art and Engineering Exploration of Human Flocking," see Chap. 2 in this book. In that work,

computer vision approaches are employed in tracking and analyzing flocking behaviors of dancers in a live performance. We employ similar tools to track motions of individuals to craft a unique acoustic experience for each visitor on the fly.

Flocking is also explored by Huepe et al in "Generating Music from Flocking Dynamics," Chap. 7 in this book. Aggregate motion of (simulated) groups are used to generate a musical soundtrack in real time. Using flocking and ad hoc computational mappings, spatial movement is transformed naturally into music.

Work by LaViers et al, "Style-based Robotic Motion in Contemporary Dance Performance," in Chap. 9 goes further in analyzing the style of movement of individual dancers and applies the results to live dance performance. Their work draws from former work by Egerstedt, Murphey, and Ludwig on the control of marionettes [15], where a hybrid systems analysis of marionettes is used in conjunction with choreography to create a live puppet-show performance.

The growing body of literature on artistic applications suggest a spectrum of opportunities and open problems in control theory.

8.1.3 Notation

We consider an image sequence of length N, denoted $\{I\}_{k=0}^{N-1}$. The kth image in the sequence is denoted $I(k) \in \mathcal{C}^{w \times h}$, where w and h are the image width and height in pixels, respectively, and $\mathcal{C} = \{(c_1, c_2, c_3) : 0 \leq c_i \leq q-1\}$ is the color-space for a 3-channel video. For our 8-bit video, $q = 256$, but quantization described in Sect. 8.2.1 will alter q. We downsample the image by a factor of 4 and use linear interpolation before processing, so w and h are assumed to refer to the size of the downsampled image.

Denote the pixel in column j and row i of the kth image of the sequence as $I_{ij}(k) \in \mathcal{C}$. Denote the set of possible subscripts as $\mathfrak{I} \equiv \{(i, j) : 0 \leq i < h, 0 \leq j < w\}$, referred to as the "index set", and $(0, 0)$ is the upper left corner of the image plane. For this paper, if $A \subset \mathfrak{I}$, let $A^c \subset \mathfrak{I}$ and $A \bigcup A^c = \mathfrak{I}$. Define an inequality relationship for tuples (x, y) as $(x, y) \leq (u, v)$ if and only if $x \leq u$ and $y \leq v$.

The color of each pixel is represented by a random variable, $\mathbf{I}_{ij}(k) \sim H_{ij}(k)$, where $H_{ij}(k) : \mathcal{C} \to [0, 1]$ is a probability mass function. Using a "lifting" operation L, map each element $c \in \mathcal{C}$ to unique axes of $\mathbb{R}^{q^3}$ with value $[H_{ij}(k)](c)$ to represent probability mass functions as vectors (or normalized histograms), a convenient representation for the rest of the paper. Note that $\mathbf{1}^T H_{ij}(k) = 1$, when conceived of as a vector; $\mathbf{1} \in \mathbb{R}^{q^3}$. Denote an estimated distribution as $\hat{H}_{ij}(k)$. Let $\hat{\mathcal{H}}(k) = \{\hat{H}_{ij}(k) : (i, j) \in \mathfrak{I}\}$ represent the background image model, as in Fig. 8.3.

A foreground object is defined as an 8-connected collection of pixels in the image plane corresponding to a visitor. Define the set of foreground objects at time k as $\mathbb{X}(k) = \{\chi_n \subset \mathfrak{I} : n < R(k)\}$, where χ_n represents an 8-connected collection of pixels in the image plane, and $R(k)$ represents the number of foreground objects at time k. Let $F(k) = \bigcup_{\chi \in \mathbb{X}(k)} \chi$ be the set of all pixels in the image

associated with the foreground. We define the minimum bounding box around each contiguous region of pixels with the upper left and lower right corners: let $x_n^+ = \arg\min_{(i,j)\in\mathfrak{I}}(i,j)$ s.t. $(i,j) \geq (u,v)\ \forall (u,v) \in \chi_n$, and $x_n^- = \arg\max_{(i,j)\in\mathfrak{I}}(i,j)$ s.t. $(i,j) \leq (u,v)\ \forall (u,v) \in \chi_n$. The set of pixels within the minimum bounding box of χ_n is $\bar{\chi}_n = \{(i,j) : x_n^- \leq (i,j) \leq x_n^+\}$. Then, let $\overline{F}(k) = \bigcup_{n<R(k)} \overline{\chi}_n$, the set of all pixels within the minimum bounding boxes around each foreground object. $\overline{F}(k) \subset \mathfrak{I}$ is referred to as the foreground bounding box support of the image $I(k)$.

The tracking algorithm returns a set $\hat{\mathcal{Z}}(k) \subset \mathfrak{I}$, indicating the pixels identified as foreground, described in more detail in Sect. 8.3. Throughout, variants of the symbol $\mathbb{Z}$ will refer to collections of tracks, not to the set of integers.

8.1.4 Assumptions

We make the following assumptions:

1. Foreground regions of images are small.
   In general, there are relatively few visitors and this assumption holds. In some anomalous circumstances, this assumption may be violated, during galas and special events. We implemented a failsafe in these circumstances to allow the system to reinitialize and recover.
   Let $B(k) \equiv F(k)^c$ represent the set of pixels associated with the background. Assume that $|B(k)| \gg |F(k)|$.
2. The color distribution of a given pixel changes slowly relative to the frame rate.
   The appearance is allowed to change rapidly, as with a flickering light, but the distribution of colors at a given pixel must remain essentially constant between frames. In practice, this condition is only violated in extreme situations, as when lights are turned on or off. High-level logic helps the algorithm recover from a violation of this assumption.
   Interpreting $H_{ij}(k)$ as a vector, $\exists \epsilon > 0$ such that for all i, j, k,
   $\left|\left|H_{ij}(k) - H_{ij}(k+1)\right|\right| < \epsilon$, where ϵ is small.
3. To limit memory requirements, we store only a small number of the total possible histogram bins. To avoid a loss of accuracy, we make an assumption that most elements of $H_{ij}(k)$ are 0. In other words, each pixel can only take on a few colors relative to the total number of possible colors.
   The support of the probability mass function $H_{ij}(k)$ is sparse over $\mathcal{C}$.
4. By starting the algorithm before visitors enter the gallery, we assume that the image sequence contains no visitors for the first few seconds. $\exists K > 0$ such that $R(k) = 0\ \forall k < K$.
5. Pixels corresponding to visitors have a color distribution distinct from the background distribution.
   Consider a foreground pixel $I_{ij}(k)$ such that $(i,j) \in F(k)$, has probability mass function $\mathcal{F}_{ij}(k)$. The background distribution at the same pixel is $H_{ij}(k)$. Interpreting distributions as vectors, $||\mathcal{F}_{ij}(k) - H_{ij}(k)|| > \delta$ for some $\delta > 0$. While

this property is necessary in order to detect a visitor, it is not sufficient, and we use additional information for classification.

6. Visitors move slowly in the image plane relative to the camera's frame rate.
   Formally, assuming $\chi_i(k)$ and $\chi_i(k+1)$ refer to the same foreground object at different times, there is a significant overlap between $\chi_i(k)$ and $\chi_i(k+1)$: $\frac{|\chi_i(k) \cap \chi_i(k+1)|}{|\chi_i(k) \cup \chi_i(k+1)|} > O$, $O \in (0, 1)$, where O is close to 1.
7. Visitors move according to a straight-line motion model with Gaussian process noise in the image plane.
   Such a model is used in pedestrian tracking [31] and is used in tracking the location of mobile wireless devices [33]. Further, the model can be interpreted as a rigid body traveling according to Newton's laws of motion. We also assume that the time between each frame is approximately constant, so the Kalman filter system matrices of Sect. 8.3 are constant.

8.1.5 Problem Statement

Performance of each algorithm is measured as a function of the number of pixels correctly or incorrectly identified as belonging to the foreground bounding box support, $\overline{F}(k)$. First, tp refers to the number of pixels the algorithm correctly identifies as foreground pixels: $tp(k) = |\overline{F}(k) \bigcap \hat{\mathcal{Z}}(k)|$. fp is the number of pixels incorrectly identified as foreground pixels: $fp(k) = |\overline{F}(k)^c \bigcap \hat{\mathcal{Z}}(k)|$. Finally, fn is the number of pixels identified as background that are actually foreground pixels: $fn(k) = |\overline{F}(k) \bigcap \hat{\mathcal{Z}}(k)^c|$. As in [37], define "precision" as $\mathfrak{p} = \frac{tp}{tp+fp}$ and "recall" as $\mathfrak{r} = \frac{tp}{tp+fn}$. For our interactive installation, recall is more important than precision, so we use the F_2-score [22, 30], a weighted harmonic mean that puts more emphasis on recall than precision:

$$\mathfrak{F}_2 = \frac{5\mathfrak{p}\mathfrak{r}}{4\mathfrak{p} + \mathfrak{r}} \tag{8.1}$$

The problem is then: for each image $I(k)$ in sequence $\{I\}_{k=0}^{N-1}$, find a collection of foreground pixels $\hat{\mathcal{Z}}(k)$ such that $\mathfrak{F}_2(k)$ is maximized. The optimal value at each time is 1, which corresponds to an algorithm returning precisely the bounding boxes of the true foreground objects: $\hat{\mathcal{Z}}(k) = \overline{F}(k)$. We use Eq. 8.1 to evaluate our algorithm in Sect. 8.4.

8.2 Probabilistic Foreground Segmentation

In this section, we focus on the top row of Fig. 8.3, which takes an image $I(k)$ and generates a set of bounding boxes of possible foreground objects, denoted $\mathcal{M}(k)$. $\hat{\mathcal{Z}}(k)$, the final estimated collection of foreground pixels, is used with $I(k)$ to update the probabilistic background model for time $k+1$.

8.2.1 Quantization

We store a histogram $\hat{H}_{ij}(k)$ on RGB color-space for every pixel. $\hat{H}_{ij}(k)$ must be sparse by Assumption 3, so the number of exhibited colors is limited to F_{max}, a system parameter. Noise in the camera's electronics, however, spreads the support of the underlying distribution, threatening the sparsity assumption. To mitigate this effect, we quantize the color-space. We perform a linear quantization, given parameter $q < 256$, and interpreting $I_{ij}(k) \in \mathcal{C}$ as a vector, $\hat{I}_{ij}(k) = \lfloor \frac{q}{256} I_{ij}(k) \rfloor$. The floor operation reflects the typecast to integer in software in each color channel. Note that this changes the color-space $\mathcal{C}$ by altering q as indicated in Sect. 8.1.3.

8.2.2 Histogram Initialization

We use the first T frames of video as training data to initialize each pixel's estimated probability mass function, or background model. Interpret the probability mass function $\hat{H}_{ij}(k)$ as a vector in $\mathbb{R}^{q^3}$, where each axis represents a unique color. We define a lifting operation $L : \mathcal{C} \to \mathcal{F} \subset \mathbb{R}^{q^3}$ by generating a unit vector on the axis corresponding to the input color. The set $\mathcal{F}$ is the "feature set," representing all unit vectors in $\mathbb{R}^{q^3}$. Let $f_{ij}(k) = L(\hat{I}_{ij}(k)) \in \mathcal{F}$ be a feature (pixel color) observed at time k. Of the T observed features, select the $F_{tot} \leq F_{max}$ most recently observed unique features; let $\mathbb{I} \subset \{1, 2, \ldots T\}$, where $|\mathbb{I}| = F_{tot}$, be the corresponding time index set. (If $T > F_{max}$, it is possible that F_{tot}, the number of distinct features observed, exceeds the limit F_{max}. In that case, we throw away the oldest observations so $F_{tot} \leq F_{max}$.) Then, we calculate an average to generate the initial histogram: $\hat{H}_{ij}(T) = \frac{1}{F_{tot}} \sum_{r \in \mathbb{I}} f_{ij}(r)$. This puts equal weight, $1/F_{tot}$, in F_{tot} unique bins of the histogram.

8.2.3 Bayesian Inference

We use Bayes' Rule to calculate the likelihood of a pixel being classified as foreground (F) or background (B) given the observed feature, $f_{ij}(k)$. To simplify notation, let $p(F|f)$ represent the probability that pixel (i, j) is classified as foreground at time k given feature $f_{ij}(k)$. Using Bayes' rule and the law of total probability,

$$p(B|f) = \frac{p(f|B)p(B)}{p(f|B)p(B) + p(f|F)p(F)} \tag{8.2}$$

We calculate $p(f|B) = f_{ij}(k)^T \hat{H}_{ij}(k)$, as $\hat{H}_{ij}(k)$ represents the background model. The prior probability that a pixel is foreground is a constant parameter, $p(F)$, a design parameter that affects the sensitivity of the segmentation algorithm. As there

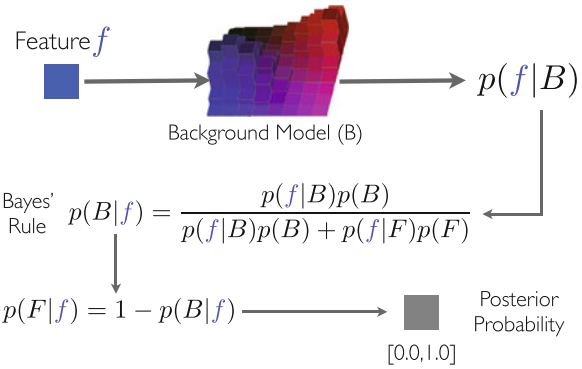


Fig. 8.5 Probabilistic Foreground Segmentation System Block Diagram for a single pixel. Feature is the observed RGB color. Its observed likelihood is referenced from the existing empirical probability distribution on color-space for the pixel. Bayes' rule enables us to calculate the probability that the pixel is part of the foreground

are only two labels, $p(B) = 1 - p(F)$. Without a statistical model for the foreground, however, we cannot calculate Bayes' rule explicitly. Making use of Assumption 5, we let $p(f|F) = 1 - p(f|B)$, which has the nice property that if $p(f|B) = 1$, then the pixel is certainly identified as background, and if $p(f|B) = 0$, the pixel is certainly identified as foreground. After calculating posterior probabilities for every pixel, the posterior image is $P(k) \in [0, 1]^{w \times h}$ where $P_{ij}(k) = p(F|f_{ij}(k)) = 1 - p(B|f_{ij}(k))$.

8.2.4 Filtering and Connected Components

Given the posterior image, $P(k)$, we perform several filtering operations to prepare a binary image for input to the connected components algorithm. We perform a morphological open followed by a morphological close on the posterior image with a circular kernel of radius r, a design parameter, using the notion of morphological operations on grayscale images discussed in [47, 48]. Such morphological operations have been used previously in segmentation tasks [32]. Intuitively, the morphological open operation will reduce the estimated probability of pixels that are not surrounded by a region of high-probability pixels, smoothing out anomalies. The close operation increases the probability of pixels that are close to regions of high-probability pixels. The two filters together form a sort of smoothing operation, yielding a modified probability image $\breve{P}(k)$ (Fig. 8.5).

We apply a threshold with level $\gamma \in (0, 1)$ to $\breve{P}(k)$ to generate a binary image $\mathfrak{P}(k)$. This threshold acts as a decision rule: if $\breve{P}_{ij}(k) \geq \gamma, \mathfrak{P}_{ij}(k) = 1$, and otherwise, $\mathfrak{P}_{ij}(k) = 0$, where 1 corresponds to "foreground" and 0 to "background." Then, we perform morphological open and close operations on $\mathfrak{P}_{ij}(k)$; operating on a

binary image, these morphological operations have their standard definition. The morphological open operation will remove any foreground region smaller than the circular kernel of radius r', a design parameter. The morphological close operation fills in any region too small for the kernel to fit without overlapping an existing foreground region, connecting adjacent regions.

On the resulting image, the connected components algorithm detects 8-connected regions of pixels labeled as foreground. For this calculation, we make use of OpenCV's `findContours()` function [9] which returns both contours of connected components, used in Sect. 8.3.2, and the set of bounding boxes around the connected components, denoted $\mathcal{M}(k)$. These bounding boxes are used by the tracking system in Sect. 8.3, so we represent them as vectors: for $m \in \mathcal{M}(k)$, $m \in \mathbb{R}^4$ with axes representing the x, y coordinates of the center, along with the width and height of the box.

8.2.5 Updating the Histogram

The tracking algorithm takes $\mathcal{M}(k)$, the list of detected foreground objects, as input and returns $\hat{\mathcal{Z}}(k)$, the set of pixels identified as foreground. To update the histogram, we make use of feature $f_{ij}(k)$, defined in Sect. 8.2.2.

First, the histogram $H_{ij}(k)$ is not updated if it corresponds to a foreground pixel: if $(i, j) \in \hat{\mathcal{Z}}(k)$, then $H_{ij}(k+1) = H_{ij}(k)$.

Otherwise, let $\mathcal{S}$ represent the support of the histogram $H_{ij}(k)$, or the set of nonzero bins: $\mathcal{S} = \{x \in \mathcal{F} : x^T H_{ij}(k) \neq 0\} \subset \mathcal{F}$. By the sparsity constraint, $|\mathcal{S}| \leq F_{max}$. If feature $f_{ij}(k)$ has no weight in the histogram ($f_{ij}(k)^T H_{ij}(k) = 0$) and there are too many features in the histogram ($|\mathcal{S}| = F_{max}$), a feature must be removed from the histogram before updating to maintain the sparsity constraint. The feature with minimum weight (one arbitrarily selected in event of a tie) is removed and the histogram is renormalized. Selecting the minimum: $\mathfrak{f} \in \arg\min_{x \in \mathcal{S}} x^T H_{ij}(k)$. Removing $\mathfrak{f}$ and re-normalizing:

$$\hat{H}_{ij}(k) = \frac{H_{ij}(k) - \mathfrak{f}^T H_{ij}(k)\mathfrak{f}}{1 - \mathfrak{f}^T H_{ij}(k)} \tag{8.3}$$

Finally, we update the histogram with the new feature:

$$H_{ij}(k+1) = (1-\alpha)\hat{H}_{ij}(k) + \alpha f_{ij}(k) \tag{8.4}$$

The parameter α affects the adaptation rate of the histogram. Given that a particular feature $f \in \mathcal{F}$ was last observed τ frames in the past and had weight ω, the feature will have weight $\omega(1-\alpha)^\tau$. As α gets larger, the past observations are "forgotten" more quickly. This is useful for scenes in which the background may change slowly, as with natural lighting through the course of a day.

8.3 Multiple Visitor Tracking

Lacking camera calibration, we track foreground visitors in the image plane rather than the ground plane. Once the foreground/background segmentation algorithm returns a set of detected visitors, the challenge is to track the visitors to gather useful state information: their position, velocity, and size in the image plane.

Using Assumption 7, we approximate the stochastic dynamical model of a visitor as follows: $z_i(k+1) = Az_i(k) + q_i(k)$, $m_i(k) = Cz_i(k) + r_i(k)$, $q_i(k) \sim \mathcal{N}(0, Q)$, $r_i(k) \sim \mathcal{N}(0, R)$, $R = \sigma I$,

$$A = \begin{bmatrix} A' & 0 & 0 \\ 0 & A' & 0 \\ 0 & 0 & I_2 \end{bmatrix}, \; A' = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1\,0\,0\,0\,0\,0 \\ 0\,0\,1\,0\,0\,0 \\ 0\,0\,0\,0\,1\,0 \\ 0\,0\,0\,0\,0\,1 \end{bmatrix}, \; Q = \begin{bmatrix} Q_x & 0 & 0 \\ 0 & Q_y & 0 \\ 0 & 0 & Q_s \end{bmatrix}$$

where I_2 is a two-dimensional identity matrix. State vector $z_i(k) \in \mathbb{R}^6$ encodes the x-position, x-velocity, y-position, y-velocity, width, and height of the bounding box, respectively, relative to the center of the box. $m_i(k) \in \mathbb{R}^4$ represents the observed bounding box of the object. $Q, R \succ 0$ are the covariances, parameters for the algorithm. Let $\mathbb{Z}(k) = \{z_i(k) : i < Z(k)\}$ be the true states of the $Z(k)$ visitors. Let $\hat{\mathbb{Z}}(k) = \{\hat{z}_i(k) : i < \hat{Z}(k)\}$ be the set of $\hat{Z}(k)$ estimated states. Let $\check{\mathbb{Z}}(k) = \{\check{z}_i(k) : i < \check{Z}(k)\}$ be the set of $\check{Z}(k)$ predicted states. $\mathcal{M}(k)$ is the set of observed bounding boxes at time k, and $\check{\mathcal{M}}(k) = \{\check{m}_i : \check{m}_i = C\check{z}_i(k), i < \check{Z}(k)\}$ is the set of predicted observations.

Given this linear model, and given that observations are correctly matched to the tracks, a Kalman filter bank solves the multiple target tracking problem. In Sect. 8.3.1, we discuss the matching problem. When observations are not matched with an existing track, a new track must be created in the Kalman filter bank. Given an observation $m \in \mathbb{R}^4$, representing a bounding box, we initialize a new Kalman filter with state $z = (C^T C)^{-1} C^T m$, the pseudo-inverse of $m = Cz$, and initial error covariance $P = C^T RC + Q$. In Sect. 8.3.2, we discuss criteria for tracks to be deleted. After matching and deleting low confidence tracks, the tracking algorithm has a set of estimated bounding boxes, $\hat{\mathcal{M}}(k) = \{\hat{m}_n = C\hat{z}_n(k) : n < \hat{Z}(k)\}$. The final result must be a set of pixels identified as foreground, $\hat{\mathcal{Z}}(k) \subset \mathfrak{I}$, and we need to convert m_i from vector form to coordinates of the corners of the bounding box to generate $\hat{\mathcal{Z}}(k)$, which is used to evaluate performance at time k in Sect. 8.4. Using superscripts to denote elements of a vector, m_n^1 and m_n^2 are the x and y coordinates of the center of the box. m_n^3 and m_n^4 are the width and height. To convert the vector back to a subset of $\mathfrak{I}$, let $m_n^- = (m_n^1 - \frac{m_n^3}{2}, m_n^2 - \frac{m_n^4}{2}) \in \mathfrak{I}$ and $m_n^+ = (m_n^1 + \frac{m_n^3}{2}, m_n^2 + \frac{m_n^4}{2}) \in \mathfrak{I}$. If any coordinate lies outside the limits of $\mathfrak{I}$, we set that coordinate to the closest value

within $\mathfrak{I}$, to clip to the image plane. Let $\nu_n = \{(i, j) : m_n^- \leq (i, j) \leq m_n^+\}$. Finally, $\hat{\mathcal{Z}}(k) = \bigcup_{n<\hat{Z}(k)} \nu_n \subset \mathfrak{I}$, the set of pixels within the estimated bounding boxes.

8.3.1 Gale-Shapley Matching

Matching observations to tracks make multiple-target tracking a difficult problem: in its full generality, the problem requires recomputation of the Kalman filter over the entire time history as previously decided matchings may be rejected with the additional information, preventing recursive solutions. To avoid this complexity, suboptimal solutions are sought. In this section, we describe a greedy, recursive approach that, for a single frame, matches observations to tracks to update the Kalman filter bank.

While some algorithms, e.g., mean-shift [13], use information gathered about the appearance of the foreground object to aid in track matching, our algorithm does not: we assume that individuals are indistinguishable. Here, observation-to-track matching is performed entirely within the context of the probability distribution induced by the Kalman filters. We make use of the Gale-Shapley matching algorithm [19], the solution to the "stable-marriage" problem.

In what follows, we describe the matching problem at time k. Formally, we are given $\mathcal{M}$, the set of detected foreground object bounding boxes, and $\breve{\mathbb{Z}}$, the set of predicted states. Let $|\mathcal{M}| = M$ and $|\breve{\mathbb{Z}}| = Z$. Introduce placeholder sets $\mathcal{M}_\emptyset$ and $\mathbb{Z}_\emptyset$ such that $|\mathcal{M}_\emptyset| = Z$ and $|\mathbb{Z}_\emptyset| = M$. Further, $\mathcal{M} \bigcap \mathcal{M}_\emptyset = \emptyset$ and $\breve{\mathbb{Z}} \bigcap \mathbb{Z}_\emptyset = \emptyset$. These placeholder sets will allow tracks and observations to be unpaired, implying a continuation of a track with a missed observation [45], or the creation of a new track. Define extended sets as $\mathcal{M}^+ = \mathcal{M} \bigcup \mathcal{M}_\emptyset$ and $\mathbb{Z}^+ = \breve{\mathbb{Z}} \bigcup \mathbb{Z}_\emptyset$. Note that $|\mathcal{M}^+| = |\mathbb{Z}^+|$, a prerequisite for applying the Gale-Shapley algorithm [19]. Let $G \equiv |\mathcal{M}^+|$.

We now describe the preference relation necessary for the Gale-Shapley algorithm. Let $m_i \in \mathcal{M}$ and $\breve{z}_j \in \breve{\mathbb{Z}}.\breve{z}_j$ is the predicted state of track j. The Kalman filter estimates an error covariance for the predicted state: $P_j \succ 0$. We are interested in comparing observations, not states, so the estimated error covariance of the predicted observation, $\breve{m}_j = C\breve{z}_j$, is $CP_jC^T + R$, from the linear system described at the start of Sect. 8.3. The Mahalanobis distance between two observations under this error covariance matrix is

$$d(m_i, \breve{m}_j) = \sqrt{(m_i - \breve{m}_j)^T (CP_jC^T + R)^{-1}(m_i - \breve{m}_j)} \tag{8.5}$$

To make a preference relation, we exponentially weight the distance: $\omega_{ij} = \exp(-d(m_i, \breve{m}_j))$, $\omega_{ij} \in (0, 1)$. As the distance approaches $0, \omega_{ij} \to 1$. Making use of Assumption 6, we place constraints on the distance: for some threshold $\gamma_{min} \in (0, 1)$, if $\omega_{ij} < \gamma_{min}$ (equiv. the distance is too great), then we deem the matching impossible, by Assumption 6. The symmetric preference relation $\phi : \mathcal{M}^+ \times \mathbb{Z}^+ \to \mathbb{R}$ is as follows:

$$\phi(m_i, \breve{z}_j) = \begin{cases} 0 & m_i \in \mathcal{M}_\emptyset \text{ or } \breve{z}_j \in \mathbb{Z}_\emptyset \\ \omega_{ij} & \omega_{ij} \geq \gamma_{min} \\ -1 & \omega_{ij} < \gamma_{min} \end{cases} \tag{8.6}$$

Equation 8.6 indicates that if a track $\breve{z}_j$ or observation m_i is to be unpaired, the preference relation between $\breve{z}_j$ and m_i is 0. If the Mahalanobis distance is too large, the preference relation is -1, so not pairing the two is preferred. Otherwise, the preference is precisely the exponentially weighted Mahalanobis distance between the predicted observation $\breve{m}_j$ and m_i.

Then, the Gale-Shapley algorithm with $\mathbb{Z}^+$ as the proposing set pairs each $z \in \mathbb{Z}^+$ with exactly one $m \in \mathcal{M}^+$, resulting in a stable matching. That is, if observation i is paired with track j, and another observation n is paired with track k, if $\omega_{ij} < \omega_{ik}$, then $\omega_{ik} < \omega_{nk}$, so while observation i would benefit from matching with track k, track k would lose, so no rematching is accepted. Gale and Shapley prove that their algorithm generates a stable matching, and that it is optimal for $\mathbb{Z}^+$ in the sense that, if w_j is the final score associated with $z_j \in \mathbb{Z}^+$ after matching, then $\sum_j \omega_j$ is maximized over the set of all possible stable matchings [19]. Thus, tracks are paired with the best possible candidate observations.

We refer to the final matching as the set $\mathfrak{M} \subset \mathbb{Z}^+ \times \mathcal{M}^+$, where $|\mathfrak{M}| = G.\mathfrak{M}$ is the input to the Kalman Filter bank as in Fig. 8.3. Then, each pair $(z, m) \in \mathfrak{M}$ is used to update the Kalman filter bank: depending on the pairing, this creates a new track, or updates an existing track with or without an observation. The Kalman update step generates $\hat{\mathbb{Z}}(k)$ and $\breve{\mathbb{Z}}(k+1)$.$\hat{\mathbb{Z}}(k)$ is used to generate $\hat{\mathcal{M}}(k)$ and $\hat{\mathcal{Z}}(k)$ as described at the beginning of Sect. 8.3, and $\breve{\mathbb{Z}}(k+1)$ is used as input for the next iteration of the Gale-Shapley Matching algorithm.

8.3.2 Heuristic Confidence Model

We employ a heuristic confidence model to discern people from spurious detections such as reflections from skylights. We maintain a confidence level $c_i \in [0, 1]$ for each tracked object $z_i \in \hat{\mathbb{Z}}(k)$, which is a weighted mix of information from the error covariance of the Kalman filter, the size of the object, and the amount of shape deformation of the contour of the object (provided by OpenCV). Typically, undesirable objects are small, move slowly, and have a nearly constant contour.

In the following, we drop the dependence on time k for simplicity and denote time $k+1$, with a superscript $+$.

Consider an estimated state $\hat{z} \in \hat{\mathbb{Z}}$, with error covariance P. Let $c^{dyn} = \exp(-\det(P)/\gamma_{det})$, with parameter γ_{det}. Intuitively, as the determinant of P increases, the region around $\hat{z}$ which is likely to contain the true state expands, implying lower confidence in the estimate. Let $c^{sz} = 1$ if the bounding box width and height are both large enough, $c^{sz} = 0.5$ if one dimension is too small, and

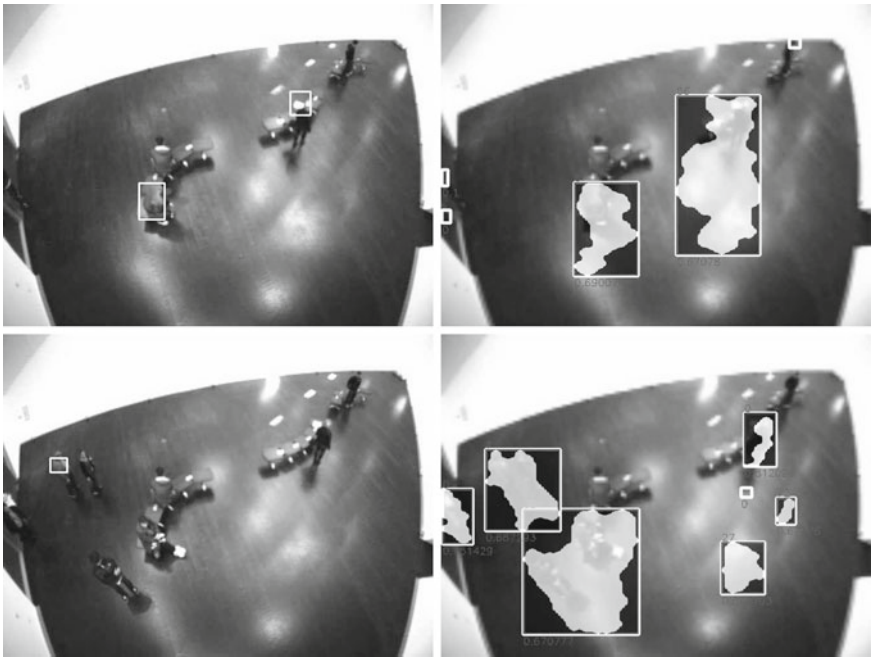


Fig. 8.6 Foreground annotations on two frames of the `ManyVisitors` sequence. *Left* OpenCV. *Right* Our approach. This image conveys that the results from OpenCV have higher precision but significantly lower recall, resulting in the F_2-scores of Fig. 8.7

$c^{sz} = 0$ if both are too small, relative to parameters $\underline{w}$ and $\underline{h}$ representing the minimum width and height. The third component, c^{sh}, is derived from the Hu moment (using OpenCV functionality), measuring the difference between the contour of the object at time $k-1$ and time k. Let $\nu_{dyn}, \nu_{sz}, \nu_{sh}$ be parameters in $[0, 1]$ such that $\nu_{dyn} + \nu_{sz} + \nu_{sh} = 1$; these are weighting parameters for different components of the confidence model. Then, given a parameter β,

$$c^{+} = (1-\beta)c + \beta(\nu_{dyn}c^{dyn} + \nu_{sz}c^{sz} + \nu_{sh}c^{sh})$$

When a track is first created at time k, $c(k) = 0$. After the first update, if at time $r > k$, $c(r) < \varphi$, another parameter, the track is discarded.

8.4 Results

We evaluate the performance of our proposed algorithm in comparison with three methods in OpenCV 2.1. Performance is measured according to precision, $\mathfrak{p}$, recall, $\mathfrak{r}$, and the F_2 measure $\mathfrak{F}_2$, introduced in Sect. 8.1.5. These are evaluated with respect

to manually labeled ground-truth sequences, which determine $\overline{F}(k)$. We compare our algorithm against tracking algorithms in OpenCV using a nonparametric statistical background model similar to what we propose, `CV_BG_MODEL_FGD` [28]. We compare against three "blob tracking" algorithms, which are tasked with segmentation and tracking: `CCMSPF` (connected component and mean-shift tracking particle-filter collision resolution), `CC` (simple connected components with Kalman Filter tracking), and `MS` (mean-shift). These comparisons, in Fig. 8.7, indicate a significant performance improvement over OpenCV across the board. A visual comparison illustrating the tradeoff between precision and recall in its effect on the F_2 score is in Fig. 8.6. We also explore the effect of the additional feedback loop we propose, by comparing our "dynamic" segmentation and tracking algorithm with a "static" version, which utilizes only the top row of the block diagram in Fig. 8.3. In the "static" version, the background model is not updated selectively, and no dynamical information is used. Figure 8.8 illustrates a precision/recall tradeoff. In both comparisons, we see an $\mathfrak{F}_2$ gain similar to the recall gain, so recall is not shown in the former and $\mathfrak{F}_2$ in the latter comparisons, due to space limitations. These comparisons, along with annotated videos of algorithm output, are available at http://automation.berkeley.edu/ACC2012Data/.

In each experiment, the first 120 frames of the given video sequence are used to initialize the background models. Results are filtered with a gaussian window, using 8 points on either side of the datapoint in question. We evaluate performance on three videos. The first is a video sequence called `StationaryVisitors` where three visitors enter the gallery and then stand still for the remainder of the video. Situations where visitors remain still are difficult for all the algorithms. Second is a video sequence called `ThreeVisitors` with three visitors moving about the gallery independently, a typical situation for our installation. Figure 8.8 illustrates that this task is accomplished well by a statistical segmentation algorithm without any tracking. Third is a video with 13 visitors, some moving about and some standing still, a particularly difficult segmentation task; this is called the `ManyVisitors` sequence.

8.5 Conclusions

This chapter presents a single-camera statistical tracking algorithm and results from our implementation at the Contemporary Jewish Museum installation entitled "Are We There Yet?." This system worked reliably during museum hours (5–8 h a day) over the 4 month duration of the exhibition under highly variable lighting conditions. We would also like to explore how the system can be extended with higher level logic. For example, we added a module to check the size of the estimated foreground region; when the lights were turned on or off, and too many pixels were identified as foreground, we would refresh the histograms of the background image probability model, allowing the system to recover quickly. In future versions, we would also like to explore automatic parameter adaptation, for example, to determine the prior

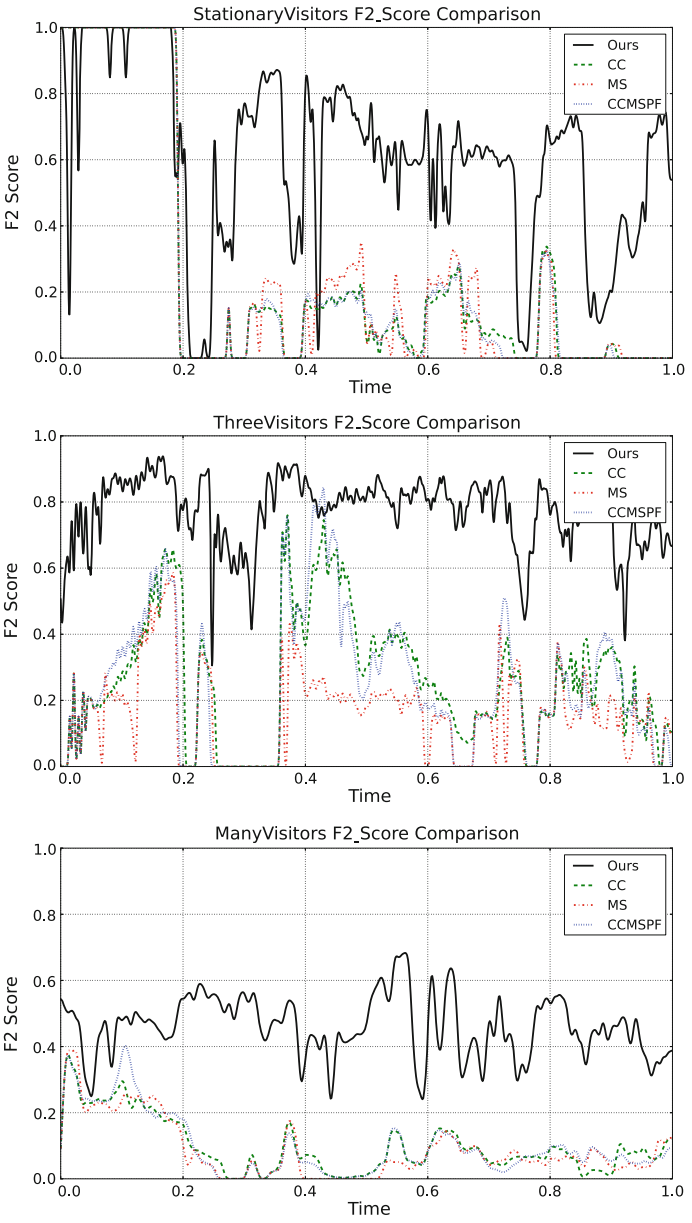


Fig. 8.7 Comparisons with OpenCV. Results indicate a significant improvement in $\mathfrak{F}_2$ score in each case. The recall plots have very similar characteristics and demonstrate our claim of improved recall over other approaches; these plots and more are available on our website. Situations when visitors stand still are a challenge for all algorithms, indicated by drastic drops. When the $\mathfrak{F}_2$ score approaches 0 for OpenCV's algorithms, our algorithm's performance is significantly reduced, although in general, it remains above 0, indicating a better ability to keep track of museum visitors, even when standing still

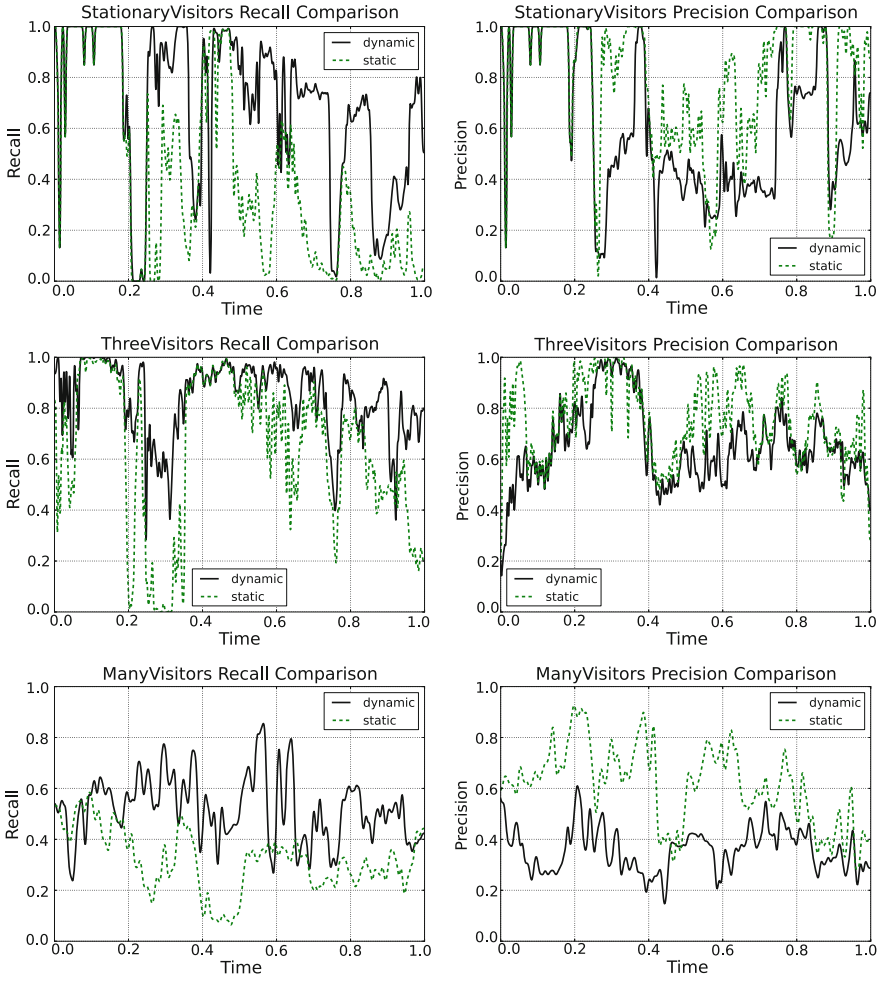


Fig. 8.8 Comparisons between "dynamic" and "static" versions of our algorithm. While the dynamic feedback loop improves the overall $\mathcal{F}_2$ score, illustrated on our website, we illustrate here that the approach improves recall at the price of precision. The `StationaryVisitors` sequence illustrates the high gains in recall with the dynamic algorithm when visitors stand still. In more extreme cases, as in `ManyVisitors`, this difference is exaggerated. The `ThreeVisitors` sequence shows very similar performance for both algorithms, indicating selectively updated background models are less useful when visitors are continuously moving

probabilities in high-traffic zones such as doorways. We welcome others to experiment with our data and use the software under a Creative Commons License. Source code and benchmark datasets are freely available in OpenCV. For details, visit: http://automation.berkeley.edu/ACC2012Data/.

8.5.1 Reviews

A 2-minute video describing the installation is available at http://j.mp/awty-video-hd. Installation reviews and documentation are available at http://are-we-there-yet.org. Below is a sample of published reviews the installation received.

- **Molly and Seth Samuel, KALW NPR** [43] : "...The exhibit is designed so every visitor hears a different combination of questions. the room seems to know exactly where I am. It feels like the questions are following me as I walk around."
- **Jonathan Curiel, SF Weekly** [14]: "The contemplative walk is rightfully celebrated as a ritual of high importance, and here, at last, is the perfect museum hybrid: an audio-visual exhibition that asks visitors thought-provoking questions...getting the words out in midstroll—in a skylit room designed by Daniel Libeskind, no less—is an exceptional blessing."
- **David Pescovitz, BoingBoing** [39]: "The project pulls a thread dating back thousands of years through Jewish culture and weaves it with innovative digital technology to create a unique, playful, poetic, and perhaps even spiritual experience as you wander the room."
- **Emily Savage, J-Weekly** [17]: "...The walls are white and bare, and the only sensation is crisp, clear sound... the sensation isn't like one you've felt before."
- **Glenn Rosenkrantz, Covenant Foundation** [41]: "Take the ages-old Jewish impulse to question and challenge. Add 21st century technology. And mix it up a bit with educators emerging embrace of new media. The result is potent."
- **Molly Samuel, CNN** [42]: "A new exhibit at the Contemporary Jewish Museum in San Francisco doesn't just challenge visitors. It questions them... Cameras track each visitor, then a computer uses statistical models to understand who is where, where they've been, and where they're heading."
- **Huffington Post** [4]: "...Gershoni and Goldberg have ingeniously made the scope as open as possible, anyone with internet access can paticipate, question, challenge, and create."
- **Sarah Adler, SF Chronicle** [1]: "...Everyone who attends is encouraged to submit questions to be added to the ever-evolving exhibit. The project is not about finding answers, but rather about learning how many questions remain undiscovered."
- **Stephanie Orma, SF Weekly** [38]: "Bay Area artists Ken Goldberg and Gil Gershoni challenge us to slow down, ask questions, and embrace contemplation. For they believe that it's questions—not answers—that help us understand the past and propel us forward in society and in our lives."

8.6 Acknowledgements

Major support for this project was provided by the Jim Joseph Foundation and the Creative Work Fund, a program of the Walter and Elise Haas Fund supported by The

William and Flora Hewlett Foundation, and The James Irvine Foundation. Additional support and hardware was generously provided by Meyer Sound Laboratories.

Special thanks to Gil Gershoni and those who helped with design and development:

– Gilad Gershoni, *Sound Designer & Senior System Architect*
– Perrin Meyer, *Audio Engineering, Meyer Sound Labs*
– Amy Gershoni, *Founder and Director of Strategy, Gershoni*
– Dave Johnson, *Senior Technologist*
– John Fox, *App Developer*
– Kate Wendell, *Project Manager, Gershoni*
– Carmi Katsir, *Project Manager, Gershoni*
– Shauna Steinbach, *Designer, Gershoni*
– Nicholas Owens, *Production Artist, Gershoni*
– Vanessa Santiago, *Programmer, Gershoni*
– Ashley Ferro-Murray, *PhD Candidate, Dept of Theater, Dance and Performance Studies*.

The authors appreciate the OpenCV open-source compute vision toolkit, including founder Gary Bradski and all developers who contributed. We also appreciate insights and suggestions from Jon Barron and Dmitry Berenson, the hard work of Akihiro Matsukawa, Hui Peng Hu, Kevin Horowitz, and Tuan Le, and the generous support of Caitlin Marshall and Jerry and Nancy Godbehere.

We thank Advisory Board members including Dr. Natasha Boas (Independent Curator), Catharine Clark (Gallerist), Dr. Nathaniel Deutsch (Professor of Religion at UC Santa Cruz), Dr. Ari Kelman (Professor, Stanford), Rabbi Noa Kushner, Amichai Lau-Lavie (Founder, Executive and Artistic director of Storahtelling), Dr. Perrin Meyer (Audio Engineer and Designer, Meyer Sound Labs), Dan Schifrin (Director of Public Programs and Writer-in-Residence at the Contemporary Jewish Museum), and Tiffany Shlain (filmmaker, artist, founder of the Webby Awards and co-founder of the International Academy of Digital Arts and Sciences).

Special thanks to the Contemporary Jewish Museum Staff:

– Connie Wolf, *Director*
– Dan Schifrin, *Director of Public Programs/Writer-in-Residence*
– James Leventhal, *Deputy Director for Development*
– Leah Tarlen, *Institutional Gifts Manager*
– Jodi Waynberg, *Curatorial Associate*
– Mark Edwards, *AV Coordinator & IT Assistant*
– Josh Pieper, *Preparator*
– Fraidy Aber, *Director of Education*.

Special thanks as well to the editors of this book and the organizers of the American Control Conference Special Session on Controls and Art: Amy LaViers, John Baillieul, Naomi Leonard, D'Andrea Raffaello, Cristian Huepe, Magnus Egerstedt, Rodrigo Cadiz, and Marco Colasso.

References

1. Adler S (2011) Q&A and Q at Jewish Contemporary Museum. SFChronicle, 26 April 2011. http://www.sfgate.com/entertainment/article/Q-A-and-Q-at-Jewish-Contemporary-Museum-2373817.php
2. Alonso IP, Llorca DF, Sotelo MÁ, Bergasa LM, de Toro PR, Nuevo J, Ocaña M, Garrido MAG (2007) Combination of feature extraction methods for svm pedestrian detection. IEEE Trans Intell Transp Syst 8(2):292–307
3. Are we there yet? at the Contemporary Jewish Museum. DesignBoom Magazine, 28 June 2011. http://www.designboom.com/art/are-we-there-yet-at-the-contemporary-jewish-museum/
4. 'Are we there yet?': Gil Gershoni, Ken Goldberg At San Francisco's Contemporary Jewish Museum. Huffington Post, 10 August 2011. http://www.huffingtonpost.com/2011/06/10/are-we-there-yet-gil-gers_n_872899.html#s288913
5. Argyros AA, Lourakis MIA (2006) Binocular hand tracking and reconstruction based on 2D shape matching. In: IEEE 8th international conference on pattern recognition, vol 1. pp 207–210
6. Baillieul J, Ozcimder K (2012) The control theory of motion-based communication: problems in teaching robots to dance. In: IEEE American control conference (ACC' 2012), pp 4319–4326
7. Blackman SS (1986) Multiple-target tracking with radar applications. Artech House Inc, Dedham, 463 p 1
8. Bradski GR, Pisarevsky V (2000) Intel's computer vision library: applications in calibration, stereo segmentation, tracking, gesture, face and object recognition. In IEEE computer vision and pattern recognition 2000, vol 2, pp 796–797
9. Bradski G, Kaehler A (2008) Learning OpenCV: computer vision with the OpenCV library. O'Reilly Media Inc, Sebastopol, USA
10. Chen T, Haussecker H, Bovyrin A, Belenov R, Rodyushkin K, Eruhimov V (2005) Computer vision workload analysis: case study of video surveillance systems. Intel Technol J 9(2):109–118
11. Chen Y, Yang J (2013) Robust principal component analysis for recognition. In: Sun C, Fang F, Zhou ZH, Yang W, Liu ZY (eds) IScIDE 2013, vol 8261. LNCS Springer, Heidelberg, pp 223–229
12. Coifman B, Beymer D, McLauchlan P, Malik J (1998) A real-time computer vision system for vehicle tracking and traffic surveillance. Transp Res Part C: Emerg Technol 6(4):271–288
13. Comaniciu D, Meer P (2002) Mean shift: a robust approach toward feature space analysis. IEEE Trans Pattern Anal Mach Intell 24(5):603–619
14. Curiel J (2011) 'Are we there yet? 5,000 years of answering questions with questions': art review. SFWeekly, 13 April 2011. http://bit.ly/awty-sf-weekly
15. Egerstedt M, Murphey T, Ludwig J (2007) Motion programs for puppet choreography and control. In: Bemporad A, Bicchi A, Buttazzo G (eds) HSCC 2007, vol 4416. LNCS Springer, Heidelberg, pp 190–202
16. Elgammal A, Duraiswami R, Harwood D, Davis LS (2002) Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. Proc IEEE 90(7):1151–1163
17. Emily S (2011) The art of the question: robotic algorithms put visitors in an auditory sea of queries. JWeekly, 7 April 2011. http://www.jweekly.com/article/full/61372/the-art-of-the-question-robotic-algorithms-put-visitors-in-an-auditory-sea-/
18. Friedman J, Hastie T, Tibshirani R (2000) Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). Ann Stat 28(2):337–407
19. Gale D, Shapley LS (1962) College admissions and the stability of marriage. Am Math Mon 69(1):9–15
20. Horprasert T, Harwood D, Davis LS (1999) A statistical approach for real-time robust background subtraction and shadow detection. In IEEE ICCV, vol 99, pp 1–19
21. Huepe C, Cádiz RF, Colasso M (2012) Generating music from flocking dynamics. In: IEEE American control conference (ACC' 2012), pp 4339–4344

22. Jeon J, Lavrenko V, Manmatha R (2003) Automatic image annotation and retrieval using cross-media relevance models. In: Proceedings of the 26th annual international ACM SIGIR conference on research and development in informaion retrieval, pp 119–126
23. KaewTraKulPong P, Bowden R (2002) An improved adaptive background mixture model for real-time tracking with shadow detection. In: Video-based surveillance systems. Springer, pp 135–144
24. Kalman RE et al (1960) A new approach to linear filtering and prediction problems J Basic Eng 82(1):35–45
25. LaViers A, Egerstedt M (2011) The ballet automaton: a formal model for human motion. In: IEEE American control conference (ACC' 2011), pp 3837–3842
26. LaViers A, Egerstedt M (2012) Style based robotic motion. In: IEEE American control conference (ACC' 2012), pp 4327–4332
27. Leonard NE, Young G, Hochgraf K, Swain D, Trippe A, Chen W, Marshall S (2012) The dance studio: analysis of human flocking. In: IEEE American control conference (ACC' 2012), pp 4333–4338
28. Li L, Huang W, Gu IYH, Tian Q (2003) Foreground object detection from videos containing complex background. In: Proceedings of the 11th ACM international conference on multimedia, pp 2–10
29. Li L, Huang W, Gu IYH, Tian Q (2004) Statistical modeling of complex backgrounds for foreground object detection. IEEE Trans Image Process 13(11):1459–1472
30. Martin DR, Fowlkes CC, Malik J (2004) Learning to detect natural image boundaries using local brightness, color, and texture cues. IEEE Trans Pattern Anal Mach Intell 26(5):530–549
31. Masoud O, Papanikolopoulos NP (2001) A novel method for tracking and counting pedestrians in real-time using a single camera. IEEE Trans Veh Technol 50(5):1267–1278
32. Meyer F, Beucher S (1990) Morphological segmentation. J Vis Commun Image Represent 1(1):21–46
33. Najar M, Vidal J (2003) Kalman tracking for mobile location in nlos situations. In: 14th IEEE proceedings on personal, indoor and mobile radio communications, 2003 (PIMRC 2003), vol 3. IEEE, pp 2203–2207
34. Nataraj N (2011) 'Are we there yet?' at the Jewish Museum. SFChronicle, 31 March 2011. http://www.sfgate.com/art/article/Are-We-There-Yet-at-the-Jewish-Museum-2377224.php
35. Nummiaro K, Koller-Meier E, Van Gool L (2003) An adaptive color-based particle filter. Image Vis Comput 21(1):99–110
36. Okuma K, Taleghani A, Freitas ND, Little JJ, Lowe DG (2004) A boosted particle filter: multitarget detection and tracking. In: Proceedings of the computer vision-ECCV 2004. Springer, Heidelberg, pp 28–39
37. Olson DL, Delen D (2008) Advanced data mining techniques. Springer, Heidelberg
38. Orma S (2011) 'Are we there yet?' leaves many unanswered questions—but that's the point. SFWeekly, 31 March 2011. http://blogs.sfweekly.com/exhibitionist/2011/03/are_we_there_yet_leaves_visito.php
39. Pescovitz D (2011) Are we there yet?: new art installation at SF Contemporary Jewish Museum. BoingBoing, 22 March 2011. http://boingboing.net/2011/03/22/are-we-there-yet-new.html
40. Power PW, Schoonees JA (2002) Understanding background mixture models for foreground segmentation. In: Proceedings image and vision computing, New Zealand, vol 2002
41. Rosenkrantz G (2011) The art of questioning in the 21st century. The Covenant Foundation, 16 June 2011. http://www.covenantfn.org/news/79/251/The-Art-Of-Questioning-In-The-21St-Century.
42. Samuel M (2011) Have you ever questioned art? This art questions you. CNN, 19 April 2011. http://www.cnn.com/2011/LIVING/04/19/questions.as.art/index.html
43. Samuel M, Samuel S (2011) Are you listening? The sound installation that gets inside your head. KALW News, 30 June 2011. http://www.kalwnews.org/audio/2011/04/13/are-you-listening-the-sound-installation-gets-inside-your-head_934965.html
44. Schoellig AP, Wiltsche C, D'Andrea R (2012) Feed-forward parameter identification for precise periodic quadrocopter motions. In: IEEE American control conference (ACC' 2012), pp 4313–4318

45. Sinopoli B, Schenato L, Franceschetti M, Poolla K, Jordan MI, Sastry S (2004) Kalman filtering with intermittent observations. IEEE Trans Autom Control 49(9):1453–1464
46. Stauffer C, Grimson WEL (2000) Learning patterns of activity using real-time tracking. IEEE Trans Pattern Anal Mach Intell 22(8):747–757
47. Vincent L (1994) Morphological area openings and closings for grey-scale images. NATO ASI Series F Comput Syst Sci 126:197–208
48. Vincent Luc (1993) Morphological grayscale reconstruction in image analysis: applications and efficient algorithms. IEEE Trans Image Process 2(2):176–201
49. Viola P, Jones M (2001) Rapid object detection using a boosted cascade of simple features. In: Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition (CVPR' 2001), vol 1
50. Viola P, Jones MJ, Snow D (2005) Detecting pedestrians using patterns of motion and appearance. Int J Comput Vis 63(2):153–161
51. White R (2011) M.V. artist would like to ask you a few questions. Mill Valley Herald, 13 April 2011. http://www.marinscope.com/mill_valley_herald/news/article_f3a7b988-7f35-5ef0-8617-cbe95ec14867.html
52. Winant C (2011) Are we Jewish yet? KQED, 12 April 2011. http://www.kqed.org/arts/visualarts/article.jsp?essid=50353
53. Yang C, Duraiswami R, Davis L (2005) Fast multiple object tracking via a hierarchical particle filter. In: IEEE computer society: 10th IEEE international conference on computer vision (ICCV' 2005), vol 1, pp 212–219
54. Yilmaz A, Javed O, Shah M (2006) Object tracking: a survey. ACM Comput Surv (CSUR) 38(4):13
55. Zivkovic Z, van der Heijden F (2006) Efficient adaptive density estimation per image pixel for the task of background subtraction. Pattern Recogn lett 27(7):773–780

Chapter 9
Style-Based Robotic Motion in Contemporary Dance Performance

Amy LaViers, Lori Teague and Magnus Egerstedt

The topic of this chapter might, most simply, be considered "dancing robots." Indeed, to provide an artistic outlet for the concepts presented in [18–22], an original work of contemporary dance was choreographed and performed April 6, 2013 and April 13, 2013 in Clough Undergraduate Learning Commons at Georgia Institute of Technology. The four professional dancers and NAO Aldebaran humanoid robot pictured in Fig. 9.1 were members of the cast. What follows is a discussion of how principles from technical publications were included in the choreography of a contemporary modern dance showing entitled *Automaton*.

Thus, a method for taking human movement and animating a version of it on a robot will be provided. This is accomplished not from a direct joint angle mapping from one highly articulated skeleton (the human) to a deficient one (the robot). Instead, the method provides a way to extract key stylistic features of human movement (recovered via motion capture) and use them to instantiate a general movement model. With this approach, novel sequences, which are in the same *style*, can be produced rather than simple imitation of the original sequence. However, such a narrative, casting the work as simply a method for mimicry, short changes the diverse applications this work can have. The aim is to use engineering and art to tease out an understanding of human behavior, which is key to applications in robotics, human-centered technology, and the performing arts.

A. LaViers (✉)
Systems and Information Engineering Department, University of Virginia,
Charlottesville, VA 22904, USA
e-mail: alaviers@virginia.edu

L. Teague
Dance and Movement Studies Program, Emory University, Atlanta, GA 30322, USA
e-mail: lteague@emory.edu

M. Egerstedt
School of Electrical and Computer Engineering, Georgia Institute of Technology,
Atlanta, GA 30332, USA
e-mail: magnus@gatech.edu

A. LaViers and M. Egerstedt (eds.), *Controls and Art*,
DOI: 10.1007/978-3-319-03904-6_9,

Fig. 9.1 The four dancers in *Automaton* along with the humanoid robot used in the show. Photo by Christian Moreno

9.1 The Robot as an Onstage Character

The robot used in *Automaton* is immediately engaging to audiences. The robotic figure calls to mind experiences of humanoids in pop culture, film, and fantasy. This presents unique challenges and opportunities from a choreographic perspective. Because the robot conjures up the audience member's impressions about robots, one of the opportunities is to challenge these preconceived ideas. People can feel quite differently about a piece of technology that is human-shaped (for example, see Fig. 9.2). While few people would personify their ice maker or iPhone, visitors to our lab immediately ask what the robot's name is (and sometimes give it one themselves).

Thus, one of the choreographic goals of *Automaton* is to blur this boundary between human-shaped and non-human-shaped technology as well as the boundary between humans and programmed devices. At the interface where humans interact with technology, their own movements are induced by the automated aspects associated with a given piece of technology [12]. In this sense, humans often mimic robotic qualities in their movements. The confluence of the discrete description of movement presented in following sections, a robot to implement it on, and trained dancers who could also execute that description with a high level of accuracy made *Automaton* a perfect setting to explore the interaction of natural human movement with the movement induced by a programmed device.

In the third movement of the piece, two dancers perform with the NAO humanoid robot. This trio explores the idea that there is some level of expected behavior and

Fig. 9.2 Two dancers depict the robot being aggressively confronted. The extent to which this image conjures up empathy for a plastic container for motors, circuit boards, and a processor is surprising until its human shape is taken into account. Photo by Christian Moreno

reaction from any being or device with which a person interacts. For example, the dancers engage in a game of patty cake. This is a set of movements, a series of alternating hand claps between two people, that are essentially pre-programmed in many children (at least in Western culture). In the trio one dancer breaks the pattern of the movement—in a sense malfunctioning—and causes his partner to have a moment of extreme confusion. The confused partner then turns to the robot and makes it his new patty cake partner. But the robot also breaks the traditional pattern of patty cake. Here both the human and the robot "malfunctioned"—seemingly on purpose—or started playing a new game, causing the human partner, in a twist of irony, to experience a situation he seemed not to know how to navigate, a plight that typically affects robotic systems—particularly in dynamic environments.

This vignette displays a reversal of roles: instead of humans evolving under unknown patterns where next movements are not predictable, they engaged in a child's game with preset movements. Because the robot can easily mimic such pre-programmed movements, it can play along too. But what happens to a human when the rules of this game are slightly different? In this case (and in many others), the human malfunctions just like a robot presented with an unpredicted situation. This is analogous to an autonomous car that was designed for operation in the U.S., where

cars drive on the right side of the road, that has just been dropped in the U.K., where cars drive on the left side of the road—it would crash instantly. This metaphor mirrors aspects of human behavior as well.

Beyond the theatrical episode described here, how can we make a robot move *automatically* in a given style of movement? This is the subject of this chapter, which will proceed as follows: Sect. 9.2 presents a discussion of movement and movement style from the point of view of a certified Laban movement analyst (CMA), Sect. 9.3 gives an overview of how stylistic features might be extracted from human movement, Sect. 9.4 describes how we build style "knobs" in order to capture key aspects of movement style from real data, Sect. 9.5 describes how this method for stylistic movement generation was applied during the performance of *Automaton*, Sect. 9.6 presents a human study conducted at the showings of *Automaton*, and finally conclusions are presented in Sect. 9.7.

9.2 A Description of Movement Style

Style becomes recognizable through a gestalt of integrated elements. An aspect of style is static, determined by a core group of features that repeat, like the compositional pattering of a quilt or a dialect. However, we are continually moving from this recognizable source, while we invite new combinations into our repertoire. How does our core influence our creativity, and vice versa? We are instinctual. We adapt. We invent. We control. We play. The possibilities that unfold expand our capacity to communicate, inviting us to be bold, focused, flexible, whimsical, fierce, generous, etc... The psyche does have some control, and it gives us insight into aspects of our culture and our identity.

At our core, in our most natural state, movement is authentic–expressing the relationship between the kinesthetic, creative and psychological dimensions of the body. Our movement vocabulary emerges, including the attitudes and emotions that are embedded in our psyche. There is a continuous interplay between the space inside and outside of the body that embraces our survival mechanisms and of course, our need to connect. As we are becoming fully embodied human beings, our personal movement can already be noted in our unique walk, the tone of our personality, conversational gestures, the body's carriage, and the space we inhabit. This is what defines our personal movement style. Whether we invest in this natural state, or transform, control, and manipulate the body's language to express something specific, we are still synchronizing, sometimes balancing, the layers of movement itself.

More broadly, if a viewer recognizes the style of dance as a waltz, it is not just because it is in $\frac{3}{4}$ time. It is a particular dance within a prescribed genre–composed of its vocabulary, qualitative use of energy, composition in space, and predominant shapes. A waltz maintains a formal carriage in the upper torso in partnership with another dancer; the whole body sweeps and arcs through the space, creating under curves and sustaining peaks with a patterned timing of steps. And break dancing moves are architectural puzzles, like a Rubik's cube. The improvisational structure allows the mover to play with resiliency, momentum and balance. Body surfaces

leverage and spin off the floor; weight is shifted quickly from the hands to the feet, to the head, inverting the center of gravity and discovering tripods. Personal movement styles can potentially live inside a dance genre or speak loudly on their own.

Choreographers and movement artists develop distinctive movement vocabularies and sequences of movements that become phrase material or dances. Their movement language is an autobiographical manifestation of skeletal and muscular range, creativity, cultural values, and geographic resources. Ultimately the way in which any mover webs the layers, and invests in each of them, is what their style becomes. The dance is both a patterning of movement and a representation of aesthetic choices. A choreographer's style will become more imprinted as they discover new relationships and shape the nexus.

The movement style of Martha Graham, a modern dance pioneer, is iconic. Her shape, standing on one leg with the other leg arcing out into space, foot flexed, is supported by a contraction in the torso. Her arms balance with seriousness and emotion. The shape of her body, one arm reaching up in the vertical and one reaching forward in the saggital, as her back pulls backwards, molds the space. Graham's direct, bold, architectural movement vocabulary defined her womanhood in the 1940s. Her choreography swept powerfully left and right, pierced the diagonal, and dropped to the earth, like a shaft of light.

The description in the previous paragraph can be credited to the work of movement theorist Rudoph Laban who provided a framework for movement analysis. When observing movement, Laban defined four areas: *Body*, *Effort*, *Shape*, and *Space*. Each area contains numerous layers. For example, when observing movement with a body lens, you can see what part of the body is initiating; how the movement is sequencing throughout the body; how the body develops connections through the spine and out to the limbs; what part of the body is grounded to allow for mobility; how breath supports movement range; and how the senses and intellect inform the qualitative nature of the movement.

Flow, *weight*, *space* and *time* are the motion factors of Effort, Laban's term to describe our inner attitude towards movement. These qualitative choices, often in combination with each other, are an essential component of a personal movement style. While conscious of flow, a dancer's movement fluctuates between bound and free. The dancer can use their center of gravity or levity to produce a quality of lightness, limpness, force, or resiliency. The dancer can approach the environment with directness or interact with its volume. And lastly, the timing of the dancer's movement can be impulsive, impactful, sustained, or increase and decrease in response to body rhythms or the mover's surroundings.

9.3 Interpreting Human Motion for Robotics

How can we organize the multidimensional life of movement, including its emotional and cultural context? This section reviews the work of others and outlines our approach for directly interpreting measurement of human movement, which is not

the subject of the remainder of the chapter but is an important context within which to consider it.

In [3] Bregler coins the term *movemes*—in analogy to the linguistic concept of a phoneme—which is cited in many following publications, often offering an alternate technical definition to this same general concept. A moveme is a motion primitive, and some combination of several movemes can desribe real human movement. The term may be viewed as synonomous with the goal to develop an appropriate mathematical definition that can segment these primitives from real data—in either two (i.e., video) or three (i.e., motion capture) dimensions. The concept comes up in many arenas: general pattern recognition [6, 8, 9, 31, 33], robotics [7, 14, 17, 27, 28], and gait analysis [23, 32]. The basic search that pervades even between these academic arenas is for atomic nuggets of movements which can recombine, in the same way phonemes are the sounds that create words, to successfully recreate and form novel full-fledged movement sequences.

A large body of research has also aimed at understanding motion patterns for application to vision and animation with the hopes of implementing character to on screen avatars. Some work [2, 26, 29] has a more statistical bent while others [1, 11, 13, 15, 16, 25] aim more directly at generation of novel motion sequences according to various constraints and character-driven rules. Similarly, attempts at "stylizing" robotic motion such as [10], aim to facilitate better human-robot interaction through a similar, character-driven philosophy.

The work in the "Controls and Art" community, as loosely defined in this book, has embraced some of the same artistic questions as those we tackle here. Consider in Chaps. 2, 7, and 8 how the authors' technical work enables specific artistic expression in dance performance, museum installations, and musical composition, respectively. In Chap. 2 the authors put forward a formal model for salsa, a pair dance. The work in Chaps. 1 and 4 deals with questions of sequencing and movement arrangement for quadrotors.

The remainder of this section describes our approach for interpreting high-level aspects of human movement—such as style. Subsequent describe the implementation of the extractions outlined here on a robot—particularly as in *Automaton*. Imagine the scenario depicted in Fig. 9.3. On the left side, envision a robot endowed with style "knobs." As the values set by these parameters are changed, the style of the robot's motion changes accordingly; this motion may be viewed as a trajectory of joint angles over time. On the right side, consider human movement recorded by motion capture; also a quantity in the form of joint angles over time. Optimizing a measure of similarity between these two signals—which can be improved by adjusting the setting of the style "knobs"—would provide the setting for the style parameters that best describes the human motion. In this way, a measure of style can be achieved.

For example, to solve for parameters which best describe the shape of a trajectory between given start and end points over a specified interval, consider the optimal control problem

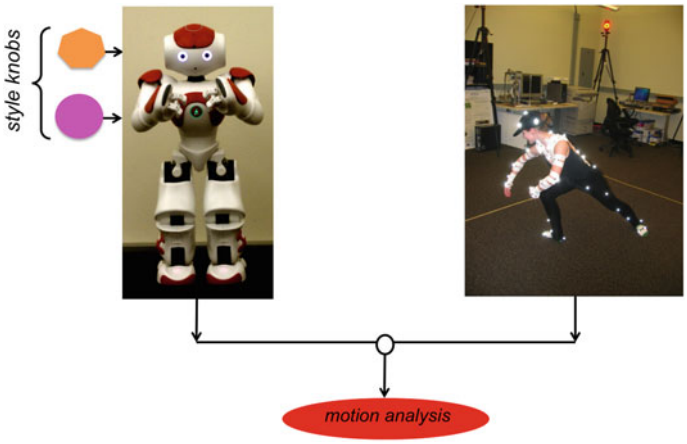


Fig. 9.3 The figure shows a conceptual overview of how the stylistic parameters might be used to measure high level aspects of human motion. On the *left* imagine a signal (perhaps implementable on a robot) that we are able to generate by varying the value of different stylistic parameters; on the *right* real data of human movement, such as a motion capture recording. The *circle* indicates we are comparing these two signals in order to measure an aspect of the real data

$$\min_u J_u = \int_0^T F(x, \sigma, \pi)dt + \psi(x(T), \sigma(T), \pi) \tag{9.1}$$

$$s.t. \begin{cases} \dot{x} = f(x, u)\ x(0) = x_0 \\ y = h(x) \end{cases} \tag{9.2}$$

where $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^m$ is the input, $y \in \mathbb{R}^l$ is the output, σ is a reference, and π is a vector of so-called "stylistic" parameters (which will be enumerated in subsequent sections).

The maximum principle states that the optimizer, u^*, can be expressed as a function of x, ξ, σ, and π, where ξ is the costate satisfying

$$\begin{aligned} \dot{\xi} &= -\frac{\partial F}{\partial x}^T - \frac{\partial f}{\partial x}^T \xi \\ \xi(T) &= \frac{\partial \psi}{\partial x}(x(T), \sigma(T), \pi). \end{aligned} \tag{9.3}$$

Plugging in the optimal u^* into the equations for x and ξ gives the expression for the Hamiltonian dynamics:

$$\dot{x} = f_x(x, u^*(x, \xi, \sigma, \pi)) \tag{9.4}$$

$$\dot{\xi} = f_\xi(\xi, u^*(x, \xi, \sigma, \pi)), \tag{9.5}$$

which we denote with

$$\dot{x} = f_x(x, \xi, \sigma, \pi) \tag{9.6}$$

$$\dot{\xi} = f_\xi(x, \xi, \sigma, \pi). \tag{9.7}$$

Now define a second cost function which we wish to minimize with respect to the weighting parameters π under the constraints imposed by the problem just outlined. That is,

$$\min_{\pi} J_\pi = \int_0^T L(x, \rho)dt + \Psi(x(T), \rho(T)) \tag{9.8}$$

$$s.t. \begin{cases} \dot{x} = f_x(x, \xi, \sigma, \pi) \; x(0) = x_0 \\ \dot{\xi} = f_\xi(x, \xi, \sigma, \pi) \; \xi(T) = \frac{\partial \psi}{\partial x}(x(T), \sigma(T), \pi) \end{cases} \tag{9.9}$$

where ρ is the empirical data we wish to mimic and classify.

The analytical solutions to this problem, and a similar segmentation problem, are presented in [22][1] and give the optimal value of π needed to best recreate ρ according to an L_2-norm. The next section (specifically Sect. 9.4.3) describes our formulation for what the stylistic parameters in π should be in order to capture a relevant aspect of style.

9.4 Generating Stylized Motion

This section[2] presents a quantitative definition of style—a choice for the style "knobs" proposed in the previous section. The definition is comprised of three distinct mathematical objects, or *stylistic* parameters, which drive a two part generation scheme. The first, an automaton, lists basic, primary movements, which are represented abstractly by the transitions of the automaton; the states in the automaton are preselected body poses. These primary movements are then recombined and sequenced according to knobs on a supervisory controller, the second stylistic parameter, which are scripted either as sets or LTL specifications. The primary movements are, in parallel, modulated via an optimal control problem where weights in a cost function, the third stylistic parameter, determine the *quality* of each movement; this scheme generates specific trajectories which are not explicitly part of the automaton. The overall scheme is illustrated in Fig. 9.4.

The definition of "style" of movement presented in this section is precise such that it can be implemented on robots, which is what Sect. 9.5 will describe—in the setting of a contemporary dance performance.

[1] The dissertation [22] also provides an implementation of these solutions to extract the quality from real motion capture data.

[2] Sections 9.4.1, 9.4.2, and 9.4.3 previously appeared in [20, 21].

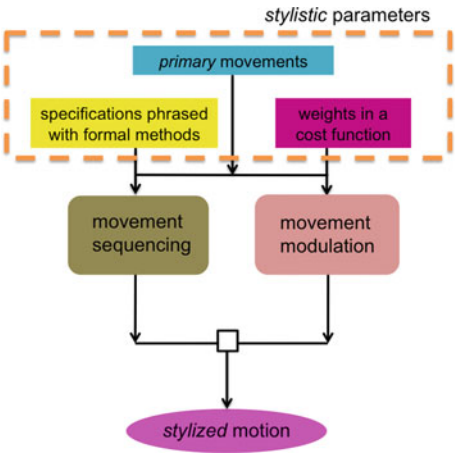


Fig. 9.4 The figure shows a conceptual overview of how the three stylistic parameters and the two-part movement scheme fit together

9.4.1 A Quantitative Movement Model

We define basic movements by enumerating them in the first element of our model: a discrete event description of *primary* motions. These pre-defined motions are initially abstracted as events in a transition system—defined only by their starting and ending poses. Later, we will use these motions as reference signals for an optimal control problem that can modulate exactly how each motion is executed. Thus, from the simple system first enumerated, we can produce much more complex movement phrases that are structured combinations (see Sect. 9.4.2) and modulations (see Sect. 9.4.3) of these primary movements.

This stance is directly inspired by the method of training used in classical ballet, the barre, in which dancers repeatedly perform small movements over and over. This canonical combination is cerntral to the practice of ballet and serves as a warm-up for dancers. These basic movement snippets are then recombined later in class and during performances to create more complex, full-fledged movements and movement phrases—thus, we think of them as *primary*. For example, the skill required to execute a grand jeté, a majestic split-legged leap common in ballet choreography, is honed through grand battements, straight-legged high kicks, at the barre. Performing repeated kicks forward, backward, and to the side with one leg, while the other, standing leg provides stability allows the dancer to develop these muscles. Then, away from the barre, the dancer kicks one leg forward and one leg backward simultaneously to perform the grand jeté, a combination of these primary motions that invokes both practiced muscle groups at the same time (Fig. 9.5).

Thus, a rational choice for the primary motions, and state transitions, in a model describing free flowing motion in the style of classical ballet are the movements from the barre exercises as described in [34]. Additionally, we distinguish two transitions

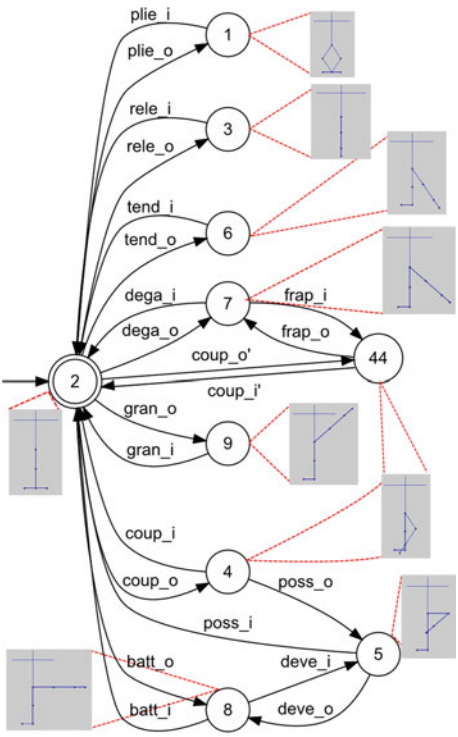


Fig. 9.5 A transition system which models the working leg (*right leg*) of a dancer during a ballet barre exercise. This figure illustrates the basic element of our movement model: a discrete event description of primary motions

for each movement listed in the table using a subscript to indicate an *in* and *out* variant. The variants stem from the fact that each primary movement should be unique. Thus, a movement is either going towards or away from each of the two states (body positions) it connects, and the next state depends on which of these is the case.

In order to demonstrate how a sample path through the system works, consider, for example, a développé; this movement is found both in barre exercises and more complex ballet movement phrases. A développé is the action when the working leg's foot is moved from the knee and then extends from the body. At the barre this motion is practiced by incremental flexion of the leg; beginning flat on the floor next to the standing leg, it passes the ankle and knee and extends out into space to full extension. Lifting the foot to the ankle or knee (without, for example, any extension to follow) are allowable movements called coupé and passé, respectively. Thus, our uniquely defined transitions (and trajectories) are three separate events for the working leg: coup_o, pass_o, deve_o. Next, the dancer performs a closing movement where the foot remains extended from the body and the leg is lowered till the foot is returned to the starting stance. This is modeled as the event batt_i—the transition from pose 8 directly

to pose 2 with a label that corresponds to the *in*-trajectory of a battement (a simpler movement that looks like a high straight-legged kick). The events coup_i, pass_i, and deve_i are also defined, that is, the reverse pose transitions are allowed and used for more complex movements. And, we will enumerate a special event to describe the relative inaction of the standing leg.

Formally, we will model this system as a finite state machine

$$\mathcal{G} = (X, E, O, f, \Gamma, o, x_0, X_m, \epsilon, \omega), \tag{9.10}$$

where X is the finite state space, E is the event set, O is an output set, $f : X \times E \to X$ is the state transition function, $\Gamma : X \to 2^E$ is the set of feasible events (at a given state), $o : X \times E \to O$ is the output map, $x_0 \in X$ is the initial condition, and $X_m \subseteq X$ is a set of marked states. In order to allow for both synchronous and asynchronous transitions once we take the Cartesian composition of two such systems, we use "empty" transitions, which are defined by the symbol ϵ. The interpretation is that for our finite state machine, we insist on $\epsilon \in E$, with the result that $\epsilon \in \Gamma(x)$ as well as $f(x, \epsilon) = x, \ \forall\, x \in X$. Moreover, we associate $\omega \in O$ with the outputs from "empty" events, i.e., $o(x, \epsilon) = \omega, \ \forall\, x \in X$.

We will use the output map as a means of preventing physically infeasible and stylistically inappropriate motions from happening when composing two automata in order to emulate more complex routines. For this, we consider the continuous set $\mathcal{X}$ which spans the entire physical space of configurations—namely, the ones describing the primary motions that are abstracted away by the state machine. We may define this additional state space as

$$\mathcal{X} = \left\{(\theta_1, ..., \theta_n) \mid \theta_i \in [\theta_{i_{\min}}, \theta_{i_{\max}}]\right\}. \tag{9.11}$$

For example, this set may describe body shapes, defined in terms of the joint angles which are limited by general kinematic constraints of humanoid geometry.

We can now think of transitions between the discrete states of the automaton as tracing paths[3] through $\mathcal{X}$. And, as our formulation associates every event with a unique state, we can associate the output map with the path that the corresponding pose transition sweeps in $\mathcal{X}$.

Sometimes a physically infeasible (i.e. trajectories which intersect) or aesthetically undesirable trajectory pair will exist. These trajectories are kept track of in O as follows: For $e \in E, x \in X$, we let

$$o(x, e) \in 2^{\mathcal{X}} = O. \tag{9.12}$$

Thus, in the next section when we consider $\mathcal{G} \times \mathcal{G}$, the set O will correspond to trajectory pairs. These will be trimmed systematically according to our stylistic parameters.

[3] A relaxed version of this may take into account that the angles corresponding to the states in the automaton are, in practice, approximate. In this case we may think of these paths as "tubes."

9.4.2 Movement Sequencing

In this section we will outline a general method for combining the elemental unit of our movement model presented in the previous section such as to impart rules—which may be style based—to sequences in the combined state space. In order to capture the fact that these primary motions may be performed at the same time and thus produce more complex movement sequences, we want to be able to compose two state machines, of the form in Eq. 9.10, while transitions that are deemed impossible or incorrect are removed from the composed system. For example, we may aspire to go from a one-legged automaton describing the primary movements of that single leg to a system involving two legs without violating the laws of physics or rules that ensure we stay faithful to a given style of movement.

To accommodate this, we introduce some compositional operations.

Given two finite automata

$$\mathcal{G}_i = (X_i, E_i, O_i, f_i, \Gamma_i, o_i, x_{i,0}, X_{i,m}, \epsilon_i, \omega_i), \ i = 1, 2,$$

we let the Cartesian composition of these two systems be given by

$$\mathcal{G} = \mathcal{G}_1 \times \mathcal{G}_2 = (X, E, O, f, \Gamma, o, x_0, X_m, \epsilon, \omega),$$

where

$$\begin{aligned}
X &= X_1 \times X_2 \\
E &= E_1 \times E_2 \\
O &= O_1 \times O_2 \\
f((x_1, x_2), (e_1, e_2)) &= (f_1(x_1, e_1), f_2(x_2, e_2)) \\
\Gamma((x_1, x_2)) &= \Gamma_1(x_1) \times \Gamma_2(x_2) \\
o((x_1, x_2), (e_1, e_2)) &= (o_1(x_1, e_1), o_2(x_2, e_2)) \\
x_0 &= (x_{1,0}, x_{2,0}) \\
X_m &= X_{1,m} \times X_{2,m} \\
\epsilon &= (\epsilon_1, \epsilon_2) \\
\omega &= (\omega_1, \omega_2).
\end{aligned} \tag{9.13}$$

Note that this is a synchronous composition in the sense that events have to happen to both of the two systems in order for a transition to happen. However, through the introduction of the empty word, we can produce effectively asynchronous transitions directly through the use of the events (e_1, ϵ_2) or (ϵ_1, e_2) in a straightforward manner.

If we perform this operation on two automata that describe the primary movements at the ballet barre, one for the right leg and one for the left, as $\mathcal{G} = \mathcal{G}_{\text{barre}_1} \times \mathcal{G}_{\text{barre}_2}$, we obtain a system that no longer is a one-legged warm-up routine, but rather a

two-legged dance-like model. For example, *pas de chat*[4] is a jump in which the dancer draws both feet up quickly, flexing at the knee, in a quick canon sequence. It is modeled for either leg in this automaton as the event sequence

$$\text{coup}_o, \text{pass}_o, \text{pass}_i, \text{plie}_o, \text{plie}_i \in E^{\star}_{\text{barre}},$$

where $\star$ denotes the Kleene closure. The corresponding event string for the composite system would be

$$(\epsilon_1, \text{coup}_o), (\text{coup}_o, \text{pass}_o), (\text{pass}_o, \text{pass}_i), (\text{pass}_i, \text{plie}_o), (\text{plie}_o, \text{plie}_i), (\text{plie}_i, \epsilon_2).$$

Note that both of these joint event strings belong to the composite set $(E_{\text{barre}} \times E_{\text{barre}})^{\star}$ and that these strings correspond to sequences of our primary movements.

We now introduce two operations, trans$(\cdot)$ and pose$(\cdot)$, which will whittle away undesirable sequences and narrow our style to something more specific than strings in $(E_{\text{barre}} \times E_{\text{barre}})^{\star}$. Let

$$\mathcal{G} = (X, E, O, f, \Gamma, o, x_0, X_m, \epsilon, \omega)$$

be a finite state machine as per Eq. 9.10 and let $O_{\text{infeas}} \subset O$ and $O_{\text{unaesth}} \subset O$ be subsets of the output set that do not contain ω. The trans$(\cdot)$ operation will take $\mathcal{G}$ and $O_{\text{infeas}} \cup O_{\text{unaesth}}$ as arguments and return a new finite state machine,

$$\text{trans}(\mathcal{G}, O_{\text{infeas}} \cup O_{\text{unaesth}}) = (X, E, O, f, \hat{\Gamma}, o, x_0, X_m, \epsilon, \omega),$$

where we only have changed the definition of Γ i.e., the set of events that are allowed to happen at a given state. The new such set is given by

$$e \in \hat{\Gamma}(x) \;\Leftrightarrow\; e \in \Gamma(x) \text{ and } o(x, e) \notin O_{\text{infeas}} \cup O_{\text{unaesth}}. \tag{9.14}$$

Thus, trans$(\cdot)$ removes transitions from our state machine that represent joint motions that are physically impossible or not of our defined aesthetic.

Secondly, certain poses when composed with one another may not be physically feasible or aesthetically desired. Thus we limit the system from entering such poses through the use of a supervisory controller and two sets of states X_{infeas} and X_{unaesth} which restrict each type of joint state, respectively. Formally speaking, given a finite state machine together with the sets $X_{\text{infeas}} \subset X$ and $X_{\text{unaesth}} \subset X$, with $x_0 \notin X_{\text{infeas}} \cup X_{\text{unaesth}}$, we let the operation

$$\text{pose}(\mathcal{G}, X_{\text{infeas}} \cup X_{\text{unaesth}})$$

[4] Literal translation from French: "step of the cat."

be the supervised system $G \backslash S$, where S is the maximally permissive, non-blocking supervisor that ensures that the set $X_{\text{infeas}} \cup X_{\text{unaesth}}$ is never reached. Note that such supervisors can be automatically generated, e.g., [4].

The final system is then given by

$$\text{pose}(\text{trans}(\mathcal{G}_1 \times \mathcal{G}_2, O_{\text{infeas}} \cup O_{\text{unaesth}}), X_{\text{infeas}} \cup X_{\text{unaesth}}). \tag{9.15}$$

Returning to our example, the set $O^{\text{ballet}}_{\text{infeas}}$, defined for the ballet model, allows us to whittle away excess transitions in order to produce system behaviors consistent with the physical capabilities of a bipedal geometry. For example, lifting a flat-footed leg off the ground, in a manner which indicates a jump (or, equivalently, raising the leg when the other is already in the air), without a bend in the knees to provide spring for the jump, is physically impossible. This corresponds to state 2 (foot flat on the ground with a straightened knee), and the infeasible event is an extension of the leg from the ground to state 8 (extended away from the body, parallel to the floor). Similar such rules used to govern a leg providing critical support (a leg in state 1, 2, or 3 when the other is in any of the others, states 44 and 4–8) such as "no coronal extension of the supporting leg from states 1 and 2" can easily be translated into regions of $2^{\mathcal{X}} \times 2^{\mathcal{X}}$.

Thus, pairs of regions of continuous space define $O^{\text{ballet}}_{\text{infeas}}$ and correspond to disallowed synchronous leg paths. Likewise, we may enumerate $O^{\text{ballet}}_{\text{infeas}}$ to describe any undesired synchronous motions that, while possible to execute, are not in the desired style of movement. Since we began this example from such a narrow set of primary motions which are, by design, intended to be performed synchronously, we leave this set empty.

Each joint state in the two-legged barre automaton is physically feasible; thus, $X^{\text{ballet}}_{\text{infeas}}$ is an empty set. The definition of $X^{\text{ballet}}_{\text{unaesth}}$ in the ballet model is a straightforward list of two-legged states which are perhaps considered ugly as judged by the metric of ballet; often, these are asymmetrical poses or poses which cannot be seen from the audience's distant perspective.

Finally, with both of these components in place, Eq. 9.15 reduces to

$$\text{pose}(\text{trans}(\mathcal{G}_{\text{barre}_1} \times \mathcal{G}_{\text{barre}_2}, O^{\text{ballet}}_{\text{infeas}}), X^{\text{ballet}}_{\text{unaesth}}).$$

This describes an automaton that accepts strings of body poses that follow our ballet-inspired rule set.

It is worth asking the question of how easy this method is to implement—for example, is naming the sets O_{infeas}, O_{unaesth}, X_{infeas}, and X_{unaesth} in situations where the desired motion style is less formalized a manageable task? A possible resolution to this question is offered in the notable and exciting extension presented in [18] and [19] where the discrete sequencing rules are assembled through statements scripted in linear temporal logic (LTL), which resembles natural language as it uses Boolean and temporal operators to operate on basic statements about the system.

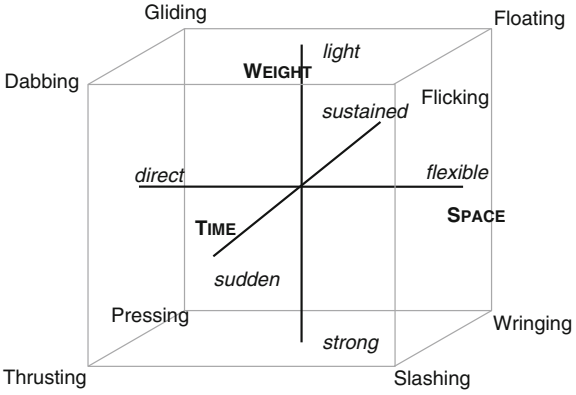


Fig. 9.6 The dynamosphere. Laban's arrangement of eight basic Efforts according to the axes of space, weight, and time. In *bold font* are the three Laban *motion factors* which deal with single movements; in *italics* are the two qualities Laban associates with each factor; and in *plain font* are the eight basic Efforts which result from the pairwise combination of each quality [30]

9.4.3 Movement Modulation

As outlined in Sect. 9.2, motions may be performed with different dynamic quality. Thus, the basic primary motions—which are abstracted away in the discrete motion sequencing framework from the previous section—tell a limiting story of stylized motion. In this section, we use a linear quadratic optimal control framework to find time-varying trajectories between static poses through a mapping between Rudolf Laban's Effort system and weights in a cost function that was first proposed in [20].

Laban names four categories of Effort or *motion factors*: *space*, *weight*, *time*, and *flow*. Space, weight, and time deal with individual movements while flow describes the quality of a succession of movements; each are described in detail in [24, 30].

The relationship of space, weight, and time can be seen in Laban's *dynamosphere* as in Fig. 9.6. The extremes of these three motion factors combine pairwise to form the eight basic Efforts: dabbing, gliding, floating, flicking, thrusting, pressing, wringing, and slashing. These movements embody the extreme notions of each motion factor, and our framework will generalize this binary scale to one of continuously variable weights. Each of the basic Efforts corresponds to a familiar pedestrian action, highlighting, even to a lay audience, the nature of the dynamosphere arrangement: changing the quality of one motion factor moves around the cube to a different basic Effort. These three motion factors, and the fourth factor, flow, which describes the quality of the connection between movements, are described—along with some intuition behind our mathematical interpretation—in the next four paragraphs.

The space axis describes how the dancer's attitude toward space is perceived. *Flexible* movements seem more carefree, meandering, and indirect; *direct* motions appear more matter of fact and judicious with their use of space. We pair this concept with a system's notion of reference tracking; direct movements will track their path

more aggressively than flexible ones. Thus, we will make use of nominal trajectories away from which our solutions may deviate or adhere closely.

The axis of weight deals with the emanated sense of weight in the dancer's body during the movement. *Light* movements look as though they are less influenced by gravity—as if they are effortless—whereas *strong* movements are muscular and seem taxing on the body to perform. We interpret this as a specification for how much control effort is used to perform the movement, or, in terms of a cost function, how "cheap" it is to increase the magnitude of the control signal.

Thirdly, Laban prescribes a time axis on which the movement may either be *sudden* or *sustained*. This describes a quality which, while it adheres closely to the colloquial notion of these terms, is more subtle than just the duration of the movement; that is, a movement which lasts five seconds may be executed with a sudden or sustained quality. We interpret this as a metric over how much the state of the system is allowed to change: during a sustained movement it should change less while in a sudden movement it may deviate wildly to produce a trajectory that appears to the audience as frantic.

Finally, Laban describes the quality of a series of movements with flow; these may either be *free* or *bound*. In free flow, dancers seems to move through movements with great ease and continuity; while in bound flow the dancer feels control and rigidity in the muscles. We interpret this from a systems perspective as a lesser (or greater) desire for the dancer to hit poses between movements exactly. In the sequencing framework we will employ, this translates to varying the weights on a terminal pose.

To make these concepts mathematically rigorous, consider a linear system with an input $u \in \mathbb{R}^m$, a state $x \in \mathbb{R}^n$, and an output $y \in \mathbb{R}^l$ which tracks a reference signal $r \in \mathbb{R}^l$. We establish a quadratic cost function

$$J = \frac{1}{2}\int_0^{T_f} \left[(y-r)^T Q(y-r) + u^T R u + \dot{x}^T P \dot{x}\right] dt + \frac{1}{2}(y-r)^T S(y-r)\Big|_{T_f} \tag{9.16}$$

in order to find an input u principled on the weight matrices $Q \in \mathbb{R}^{l\times l}$, $R \in \mathbb{R}^{m\times m}$, $P \in \mathbb{R}^{n\times n}$, and $S \in \mathbb{R}^{l\times l}$. By construction, each of these matrices are positive definite and symmetric. Furthermore, their entries create a continuous, quantitative version of Laban's Effort system and will determine which movement qualities are exhibited by the optimal trajectory, i.e. the trajectory may be bound, direct, sudden, and strong.

Based on the previous discussion, we associate the weight Q to the Laban's motion factor space, R with the factor weight, P with time, and S with flow. These weights correlate with the quality of each factor as follows:

$$Q \sim \text{direct} \tag{9.17}$$

$$R \sim \text{light} \tag{9.18}$$

$$P \sim \text{sustained} \tag{9.19}$$

$$S \sim \text{bound} \tag{9.20}$$

where the opposite of the qualities listed, flexible, strong, sudden, and free, are achieved when these weights are relatively small, respectively.

Using these weights as the style-based parameters for varying the resulting trajectory, we solve the optimal control problem

$$\min_u J \quad s.t. \begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases} \tag{9.21}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, and $C \in \mathbb{R}^{l\times n}$.

Differentiating the Hamiltonian

$$H = \frac{1}{2}[(y-r)^T Q(y-r) + u^T Ru + \dot{x}^T P\dot{x}] + \lambda f$$

with respect to u and x gives the first order necessary condition for optimality and the dynamics of the costate $\lambda = [\lambda_1, \lambda_2, ...\lambda_n]$:

$$\frac{\partial H}{\partial u} = u^T(R + B^T PB) + x^T A^T PB + \lambda B = 0$$

from which it follows that

$$u = -(R + B^T PB)^{-1}(B^T PAx - B^T \lambda^T) \tag{9.22}$$

and

$$\frac{\partial H}{\partial x} = -\dot{\lambda} = x^T(C^T QC + A^T PA) + u^T B^T PA + \lambda A - r^T QC$$

from which it follows that

$$\begin{aligned} \dot{\xi} = \dot{\lambda}^T &= (A^T PB(R + B^T PB)^{-1}B^T PA - C^T QC - A^T PA)x \\ &+ (A^T PB(R + B^T PB)^{-1}B^T - A^T)\xi + C^T Qr. \end{aligned} \tag{9.23}$$

Applying the transversality condition we obtain:

$$\xi(T_f) = C^T SCx(T_f) - C^T Sr(T_f). \tag{9.24}$$

To solve this system for an optimal $x(t)$ we need to find ξ_0. Thus, we assemble a new state $z = [x, \xi]^T$. Now

$$\dot{z} = Mz + Nr \tag{9.25}$$

where the entries of M and N are determined from Eqs. 9.21 to 9.23 and are given below:

$$M_{11} = A - B(R + B^T P B)^{-1} B^T P A \tag{9.26}$$

$$M_{12} = -B(R + B^T P B)^{-1} B^T \tag{9.27}$$

$$M_{21} = A^T P B(R + B^T P B)^{-1} B^T P A - C^T Q C - A^T P A \tag{9.28}$$

$$M_{22} = A^T P B(R + B^T P B)^{-1} B^T - A^T \tag{9.29}$$

$$N_1 = [0]_{n \times l} \tag{9.30}$$

$$N_2 = C^T Q. \tag{9.31}$$

We know that in general

$$z(T_f) = e^{MT_f} z_0 + \int_0^{T_f} e^{M(T_f - t)} N r(t) dt, \tag{9.32}$$

which we denote by

$$z(T_f) = \Phi z_0 + q, \tag{9.33}$$

where

$$\Phi = \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{21} & \Phi_{22} \end{bmatrix} \text{and} \quad q = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix}. \tag{9.34}$$

Combining Eq. 9.24 with Eqs. 9.33–9.35 we get

$$\begin{aligned} \xi_0 = (C^T S C \Phi_{12} - \Phi_{22})^{-1} [(\Phi_{21} - C^T S C \Phi_{11}) x_0 \\ + C^T S C q_1 + q_2 + C^T S r(T_f)]. \end{aligned} \tag{9.35}$$

Thus, we have found our initial condition $z_0 = [x_0, \xi_0]^T$. When combined with Eq. 9.25, this gives the optimal $x(t)$—and thus our output $y(t)$—as nominated by the weights in Eq. 9.16.

9.5 Rule-Based Movement Generation

Choreographers are constantly seeking sources of inspiration and techniques to generate new movement. Often this involves using measures external to their bodies to generate new constraints for their creative problems—à la Merce Cunningham and his dice. Cunningham [5] famously used dice to introduce chance and randomness into his work; a roll of the dice would determine the next movement in a sequence or the order of a show on a given night. These techniques produced new and exciting movements. Later in his career, he also developed technology, the software suite LifeForms, which allowed for movement generation on a computer, to facilitate his choreographic process. The tools in this chapter afford a similar opportunity: style

Fig. 9.7 The underlying poses for the five styles of movement performed during *Automaton*. Photos by Jessica Portugal

of motion may be *programmed* through the automata-theoretic description of style presented in this chapter.

Developing movement "styles" using automata inherits the same advantage that automata afford to the technological objects which they are traditionally used to describe. Namely, they allow for discrete changes in state. In the case of a remote control, this means that when the highest channel is reached, the TV interprets the "up" signal from the remote as a command to go *down* to the first or lowest channel. This is to some degree inconsistent and would be extremely inconvenient to capture this desired behavior using continuous mathematics. In the case of dancing, the programmatic advantage is that moving between body poses does not have to be governed by the natural movement patterns of the choreographer, but instead offers an opportunity to produce unusual sequences as drawing arrows between states is a completely disembodied task—just as Cunningham's technique of rolling the dice.

Figures 9.7, 9.8, 9.9, 9.10 and 9.11 show the rule sets for each dancer. Just as the movement patterns *look* different to watch, these automata have different structures.[5] For example, one automaton is quite linear and represents the same sequence of movement repeated over and over; whereas others present more options and allow for repeated movement. Moreover, these descriptions are amenable to robotics, and the styles of movement may be *automated*. Thus, the rule-set pictured in Fig. 9.8, is also animated on the NAO humanoid robot. By employing different rule sets on each

[5] Audience members were given an experiential tutorial in automata via a printed automaton that instructed them on how to fold their programs

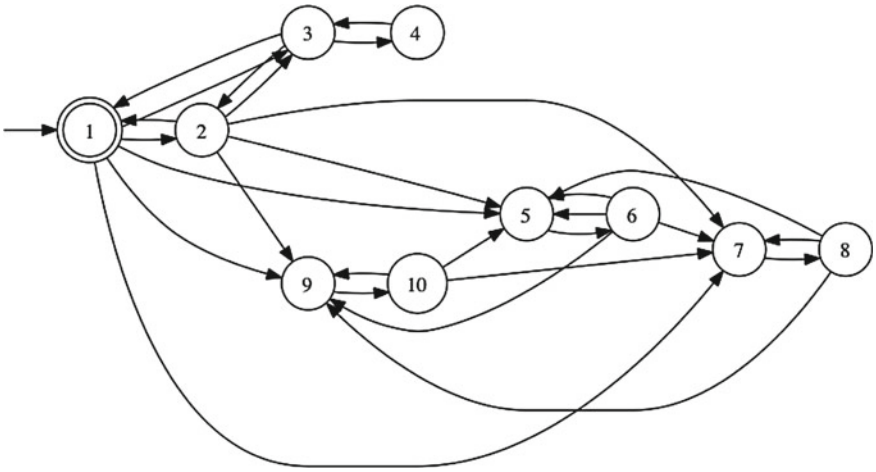


Fig. 9.8 The rule set for the dancer Helen and the humanoid robot

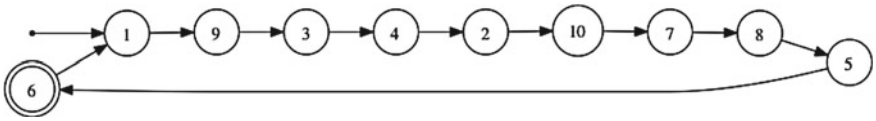


Fig. 9.9 The rule set for the dancer Camille

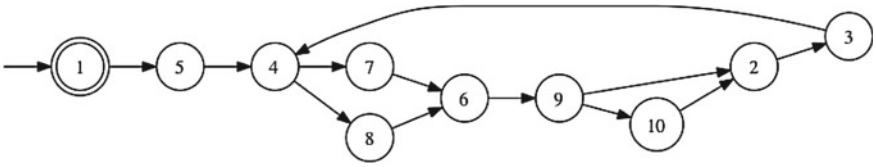


Fig. 9.10 The rule set for the dancer Alex

dancer and the robot, the audience is able to see different automata animated, live. A snapshot from this section of the piece is shown in Fig. 9.12.

9.6 What did the Audience See?: A Human Study

This section will provide an initial step toward validating the goal of endowing a robot with a consistent style of movement. The sequencing framework described in this chapter was used in *Automaton* as described in the previous section. Audience members were polled after the performances (held on April 6 and 13, 2013) for their impressions of the robot's movement.

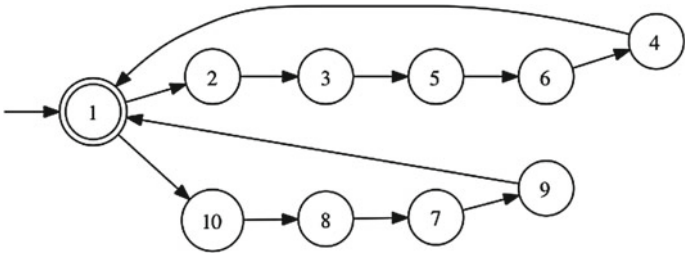


Fig. 9.11 The rule set for the dancer Erik

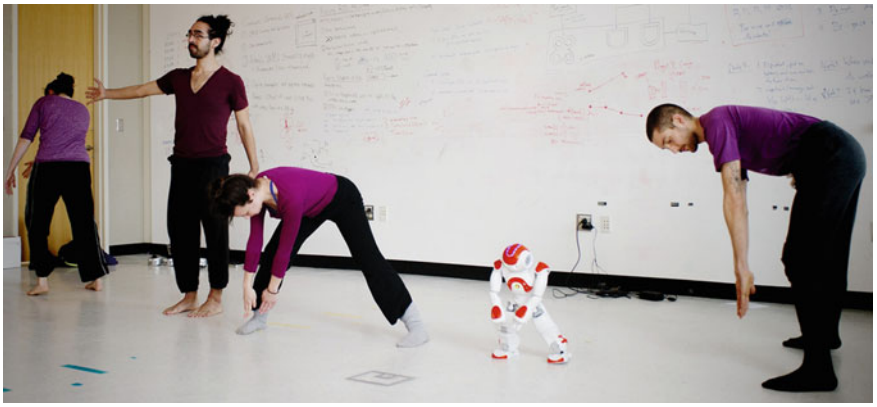


Fig. 9.12 Five styles of movement are performed during *Automaton*. Photo by Rob Felt

Between the two showings, about 100 people saw *Automaton*, and 28 audience members volunteered to fill out a survey collecting their impressions of the robot's movement. Figures 9.13 and 9.14 show the results of the responses and the averaged values for each question. (The complete responses can be found in [22].) The questions aimed to capture the viewer's impression of a colloquial notion of "style" as applied to the overall movement of the robot in the show. The study subjects were asked if their impressions of the movement were that it was "specific," "stylized," "virtuosic," "enjoyable," and whether they "felt" anything at all about the movement. These questions were phrased broadly and were not meant to define a subjective or psychological notion of "style." Instead, the questions simply probed whether the audience believed that something consistent was underlying the robot's movement.

The responses indicate that the viewers polled did experience the robot's movement as something other than random motions set to music. In particular, the responses to the first two questions support this claim. Question 1 polled the viewer's sense that the robot's movement was very "specific." The average of almost 8 on a scale of 1 to 10, with 1 corresponding to random motion and 10 corresponding to very specific motion, indicates the audience did feel the robot's movement was "specific." Question 2 was similar except phrased using the term "style"; the average

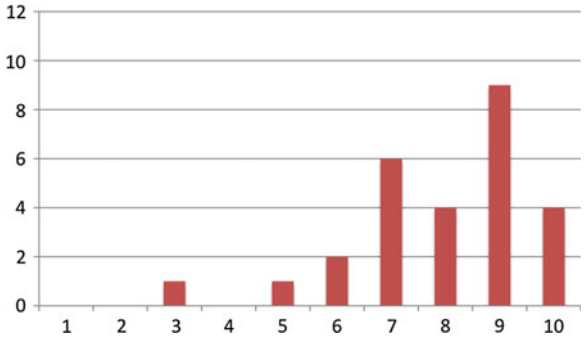


Fig. 9.13 A histogram of survey responses collected between both showings of *Automaton* completed by volunteers from the audience. The question shown here was stated as follows: *Please rate your impression of the overall consistency of the robots movement. A rating of 1 indicates that it seemed to be moving completely randomly. A rating of 10 indicates that it was moving in a very specific way*. Each column counts how many subjects selected 1, 2, 3, ..., 10, respectively, for this question

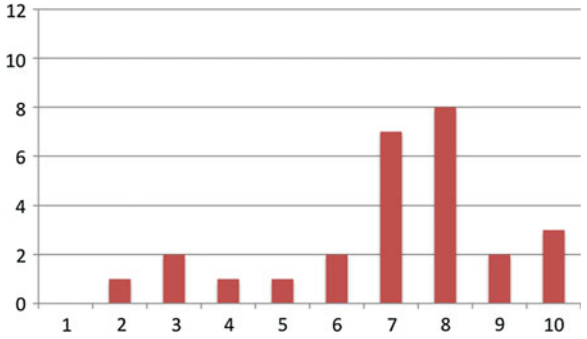


Fig. 9.14 A histogram of survey responses collected between both showings of *Automaton* completed by volunteers from the audience. The question shown here was stated as follows: *Please rate your impression of the style of the robots movement. A rating of 1 indicates that it seemed to be moving without style. A rating of 10 indicates that it was moving in a very stylized way*. Each column counts how many subjects selected 1, 2, 3, ..., 10, respectively, for this question

rating of about 7 out of 10 indicates that the viewers polled found the robot's motion to be more "stylized" than not. The responses to the other questions also indicated the audience sensed some high-level design of the robot's movement. Thus, this initial study indicates we have achieved robotic movement that is *perceived* to be of a certain style.

9.7 Conclusions

Functional tasks have a straightforward interpretation for metrics of success, efficiency, and completion and are, thus, implementable on robots in a variety of shapes. However, robots that are shaped like humans have a greater vocabulary of movement to aspire to as their movements are compared to the highly capable and varied natural movements of human beings. To cite two popular examples of how a fully autonomous humanoid robot might first appear in complex roles: in-home hospice robots should move as though they are gentle and caring and museum tour guide robots should have a brisk air of authority about their movements. Thus, a quantitative parameterization of high-level aspects of human movement is a necessary component for human-shaped robots that interact with humans.

Futher, technology of any shape relies more and more on an interpretation of human movement. Human-centered technology has moved from vending machine style interfaces where users need to figure out the function of various buttons to operate a machine to iPhones where much of the interaction takes place based on the machine's interpretation of the motion of the human fingers. Following this trend, Microsoft's Kinect extracts a human skeleton using computer vision techniques and aims to incorporate full-body motions into the Xbox entertainment system. Being able to interpret the style and other high-level aspects of a user's movement can enhance these efforts.

Finally, the work presented in this chapter can interface with art. The analysis and creation of art can benefit from quantitative tools. The focus of this chapter has been on the creation of art, but on the part of analysis, being able to generate—and even *measure*—stylized movement from three parameters allows for movement from different dancers, choreographers, genres, and sections of the same piece to be compared quantitatively. Such analysis can bolster the body of academic work that *qualitatively* discusses the evolution of and connection between various genres of dance and the differences between the work of two choreographers and provide insight into the magical effect stylized movement can have on an audience.

Acknowledgments This work was supported by the US National Science Foundation through grant number 0757317. The performance of "Automaton" was funded by a generous grant from GA Tech's School of Electrical and Computer Engineering.

References

1. Bradley E, Stuart J (1998) Using chaos to generate variations on movement sequences. Chaos: an interdisciplinary. J Nonlinear Sci 8(4):800–807
2. Brand M, Hertzmann A (2000) Style machines. In: Proceedings of the 27th annual conference on computer graphics and interactive techniques. ACM Press/Addison-Wesley Publishing Company, New York, pp 183–192
3. Bregler C (1997) Learning and recognizing human dynamics in video sequences. In: IEEE computer conference on computer vision and pattern recognition. IEEE, pp 568–574

4. Cassandras CG, Lafortune S (2008) Introduction to discrete event systems. Springer, New York
5. Copeland R (2003) Merce Cunningham: the modernizing of modern dance. Routledge, London
6. Del Vecchio D, Murray RM, Perona P (2003) Decomposition of human motion into dynamics-based primitives with application to drawing tasks. Automatica 39(12):2085–2098
7. Do M, Romero J, Kjellstrom H, Azad P, Asfour T, Kragic D, Dillmann R (2009) Grasp recognition and mapping on humanoid robots. In: 9th IEEE-RAS international conference on humanoid robots, humanoids 2009. IEEE, pp 465–471
8. Egerstedt M, Balch T, Dellaert F, Delmotte F, Khan Z (2005) What are the ants doing? vision-based tracking and reconstruction of control programs. In: Proceedings of the IEEE international conference on robotics and automation (ICRA 2005). pp 18–22
9. Fanti C, Zelnik-Manor L, Perona P (2005) Hybrid models for human motion recognition. In: IEEE computer society conference on computer vision and pattern recognition, CVPR 2005, vol 1. IEEE, pp 1166–1173
10. Gielniak MJ, Liu CK, Thomaz AL (2010) Stylized motion generalization through adaptation of velocity profiles. In: RO-MAN, IEEE. IEEE, Atlanta, pp 304–309
11. Gillies M (2009) Learning finite-state machine controllers from motion capture data. IEEE Trans Comput Intell AI in Games 1(1):63–72
12. Haraway DJ (1991) A cyborg manifesto: science, technology, and socialist-feminism in the late twentieth century. Simians, cyborgs and women: the reinvention of nature. Routledge, New York, p 149–181
13. Hsu E, Pulli K, Popović J (2005) Style translation for human motion. ACM Trans Graphics 24(3):1082–1089
14. Jenkins OC, Mataric MJ (2003) Automated derivation of behavior vocabularies for autonomous humanoid motion. In: Proceedings of the second international joint conference on autonomous agents and multiagent systems. ACM, Australia, pp 225–232
15. Kingston P, Egerstedt M (2011) Time and output warping of control systems: comparing and imitating motions. Automatica 47(8):1580–1588
16. Kovar L, Gleicher M, Pighin F (2002) Motion graphs. ACM Trans Graphics 21(3):473–482
17. Kulic D, Ott C, Lee D, Ishikawa J, Nakamura Y (2012) Incremental learning of full body motion primitives and their sequencing through human motion observation. Int J Robot Res 31(3):330–345
18. LaViers A, Chen Y, Belta C, Egerstedt M (2011) Automatic generation of balletic motions. In: ACM/IEEE second international conference on cyber-physical systems, pp 13–21
19. LaViers A, Chen Y, Belta C, Egerstedt M (2011) Automatic sequencing of ballet poses. IEEE Robot Autom Mag 18(3):87–95
20. LaViers A, Egerstedt M (2011) The ballet automaton: a formal model for human motion. In: Proceedings of the American control conference 2011, pp 3837–3842
21. LaViers A, Egerstedt M (2012) Style based robotic motion. In: Proceedings of the American control conference 2012, pp 4339–4344
22. LaViers A, Egerstedt M (2012) Style based robotic motion. Ph.D. thesis
23. Lee L, Grimson WEL (2002) Gait analysis for recognition and classification. In: Fifth IEEE international conference on automatic face and gesture recognition. IEEE, pp 148–155
24. Maletic V (1987) Body, space, expression. Walter de Gruyter & Company, Berlin
25. Martin P, Egerstedt MB (2010) Timing control of switched systems with applications to robotic marionettes. Discrete Event Dyn Syst 20(2):233–248
26. Matsubara T, Hyon S, Morimoto J (2010) Learning stylistic dynamic movement primitives from multiple demonstrations. In: IEEE/RSJ international conference on intelligent robots and systems (IROS) 2010. IEEE, pp 1277–1283
27. Miller S, Van Den Berg J, Fritz M, Darrell T, Goldberg K, Abbeel P (2012) A geometric approach to robotic laundry folding. Int J Robot Res 31(2):249–267
28. Nakaoka S, Nakazawa A, Yokoi K, Ikeuchi K (2004) Leg motion primitives for a dancing humanoid robot. In: IEEE international conference on robotics and automation (ICRA'04), vol 1. IEEE, pp 610–615

29. Nakazawa A, Nakaoka S, Ikeuchi K (2003) Synthesize stylistic human motion from examples. In: IEEE international conference on robotics and automation ICRA'03, vol 3. IEEE, pp 3899–3904
30. Newlove J, Dalby J (2004) Laban for All. Nick Hern Books, London
31. Ren X (2008) Finding people in archive films through tracking. In: IEEE conference on computer vision and pattern recognition, CVPR 2008. IEEE, pp 1–8
32. Surer E, Kose A (2011) Methods and technologies for gait analysis. Computer analysis of human behaviour. Springer, pp 105–123
33. Wang L, Hu W, Tan T (2003) Recent developments in human motion analysis. Pattern Recogn 36(3):585–601
34. Warren GW, Cook S (1989) Classical ballet technique. University of South Florida Press, Gainesville

Author Index

A. LaViers and M. Egerstedt (eds.), *Controls and Art*,
DOI: 10.1007/978-3-319-03904-6,

Subject Index

A. LaViers and M. Egerstedt (eds.), *Controls and Art*,
DOI: 10.1007/978-3-319-03904-6, © Springer International Publishing Switzerland 2014