Learning Force Fields for Limb Control in Character Animation

by

Matthew Koichi Grimes

ScB Engineering-Physics, Brown University (2000)

Submitted to the Department of Media Arts and Sciences,
School of Architecture & Planning
in partial fulfillment of the requirements for the degree of

Master of Science in Media Arts and Sciences

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2003

Author . . . .
Department of Media Arts and Sciences,
School of Architecture & Planning
May 9, 2003

Certified by . . . .
Bruce M. Blumberg
Asahi Broadcasting Corporation Career Development Professor of
Media Arts and Sciences
Thesis Supervisor

Accepted by . . . .
Andrew B. Lippman
Chairperson, Department Committee on Graduate Students

Learning Force Fields for Limb Control in Character Animation

by

Matthew Koichi Grimes

Submitted to the Department of Media Arts and Sciences,
School of Architecture & Planning
on May 9, 2003, in partial fulfillment of the
requirements for the degree of
Master of Science in Media Arts and Sciences

Abstract

This work applies vector-field-based limb control to physically simulated character animation. Vector fields of joint torques, defined over joint angle space, have been shown in the neuroscience literature to be able to generate simple reaching motions in two dimensions when linearly combined with time-varying weights. In this work, three-dimensional versions of such "basis" fields generate a similar diversity of realistic motions, while maintaining tractable control through a concise parameter space. A decomposition algorithm is presented that takes a set of limb motions and generates a set of torque fields, whose various linear combinations can generate each member of the input set, and other plausible motions. Unlike other motion extrapolators, the system extrapolates torque control programs instead of the motion paths themselves, retaining Newtonian physical correctness.

Thesis Supervisor: Bruce M. Blumberg
Title: Asahi Broadcasting Corporation Career Development Professor of Media Arts and Sciences

Acknowledgments

I am deeply grateful for the professors who have held my hand through a four year transition from physics to computer animation. In particular, many thanks to Nancy Pollard for converting me by example, Greg Turk for raising the bar on all future advisors, Aaron Bobick for introducing me to vision, computer and otherwise, Jovan Popović for demonstrating the diversity and depth of animation research, and Bruce Blumberg for making me work for it.

Many thanks to Emilio Bizzi, who agreed to be a reader after inspiring this thesis into existence.

I am indebted to the amazing people whom I blush to call my peers. Josh Lifton, Ari Benbasat, Matt Hancher and Marc Downie make me question my acceptance to MIT every day. Many parts of this thesis would've been impossible were it not for the rich pools of expertise that were never further than a few doors away.

Thanks also to the lively group that populated the Synthetic Characters' masters student office. Matt Berlin, Derek Lyons, Daphna Buchsbaum and Jesse Gray are clearly poised to rule the world in a tyranny of intelligence and laughter.

– mkg

To my parents,

Kevin and Sachiko Grimes

Learning Force Fields for Limb Control in Character Animation

by

Matthew Koichi Grimes

The following people served as readers for this thesis:

Thesis Reader ....

Jovan Popović
Assistant Professor
Department of Electrical Engineering and Computer Science

Thesis Reader ....

Emilio Bizzi
Eugene McDermott Professor in the Brain Sciences
and Human Behavior
Department of Brain and Cognitive Sciences

Contents

List of Figures

Chapter 1

Introduction

1.1 What are torque fields?

In the context of this work, torque fields are functions that map a limb's configuration to a vector of joint torques (fig 1-1). The hypothesis of this thesis is that a small set of torque fields can be linearly combined to actuate various realistic limb movements, and that simple limb movements can be decomposed into linear combinations of a shared set of fields.

A special case of such a definition is the linear angular spring, which linearly maps a single joint's configuration to its torque, relative to a neutral position. In the general case, a torque field may define joint torques as nonlinear functions of multiple joint configurations.

1.2 Motivation

The idea of modular torque generators is motivated by parallel efforts in both neuroscience and computer animation towards composable controllers of limb motion. While a detailed review of these efforts is deferred until chapter 2, an overview of torque fields is provided below.

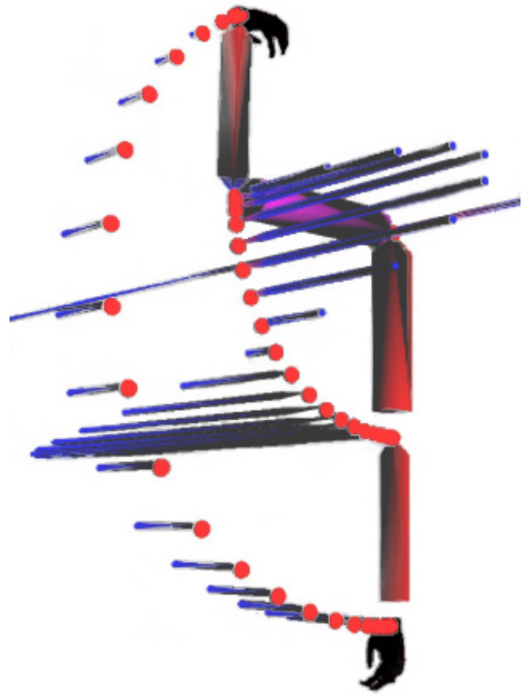


Figure 1-1: A torque field's torques, sampled at various limb configurations.

1.2.1 Biological Precedent

Discoveries in the neuroscience literature suggest that the motor cortex drives limb motions not by specifying individual muscle tensions, but through the temporally coherent coactivation of muscle sets. The effect of such a set of muscle tensions on the end effector can be modeled as a convergent spatial force field acting on it, driving the limb to the neutral position dictated by those tensions. Co-activating multiple tension sets or force fields drives the limb to an intermediate neutral position. Blending from one force field to another results in a smooth movement of this neutral position from the first field to the second. Blending force fields with temporally changing weights is a low-dimensional and potentially more tractable method for controlling a desired hand trajectory. This thesis explores how a set of such "basis" force fields can be learned from a set of limb motions, so that the input motions and other plausible motions can be recreated in simulation by various combinations of those force fields.

1.2.2 Modular Control

In computer animation, physical simulation has been widely applied to modeling minimally controlled phenomena, such as the motion of clothing [2], fluids [15], and flame [34]. Using simulation to animate controlled movements is often difficult, as tuning physical parameters such as friction and stiffness can be a highly unintuitive and unpredictable means of choreographing motion. Nonetheless, there have been great successes in executing acrobatic motions such as leaps, flips, and vaults in simulation [18]. The motor programs for physically simulated characters are typically written for a specific task, and require precise initial conditions. Modularity in these control programs has been difficult to achieve. By contrast, torque fields generate different motions by changing the weights of a shared set of torque generators.

1.2.3 Low-Dimensional Control

In the biological domain, a single force field encodes the balanced co-activations of multiple muscle groups involved in driving a limb to the field's particular neutral position. This relieves the controller, be it the brain or a spine-stimulating electrode, from individually specifying each muscle's stiffness and neutral length. Natural looking whole-limb movements can be orchestrated by simply blending from one torque field to another using a smooth step function of time. This is in contrast to the individual setting of stiffnesses and impedances in single-joint PD controllers such as damped springs. This concise parameter space gives torque fields an advantage over lower-level controllers when applying learning algorithms to determine control parameters for a particular motion.

1.3 Objectives

In this thesis, the goal is to decompose motions into a set of torque fields that can recreate those motions under simulation. Chapter 3 presents a simple parameterization of convergent torque fields, and demonstrates a method for searching the parameter space for fields that will actuate a limb to approximate a given animation. The algorithm is simply extended to decompose multiple animations into a minimal "basis set" of torque fields whose various linear combinations recreate the input animations.

1.4 Road map

- Chapter 2 situates torque fields in related work in neuroscience, robotics, and computer animation.
- Chapter 3 describes the mathematical representation of torque fields used in this work.
- Chapter 4 presents the results of distilling limb animations into torque field representations, and of recreating those animations from the torque fields through a physical simulator. Conclusions and future work are suggested.

Chapter 2

Related Work

2.1 Introduction

Torque field motor control can be contextualized among its origins in neuroscience, among other control primitives from robotics, or among other approaches to similar applications in computer animation. The following sections introduce related work from each of these three fields. The neuroscience section traces the emergence and evolution of force field control theory from its origins in equilibrium-point control to the present. The robotics section discusses field-based control among other popular robot motor learning primitives. Finally, the animation section compares force field primitives to other efforts at low-dimensional, physics-assisted animation.

2.2 Neuroscience

While some have proposed that limb control is performed by direct computation of the inverse dynamics problem [21], it is somehow unappealing to believe that anything as "wet and nonlinear" as the nervous system constantly churns through pages of second-order differential equations in real time. The ease with which humans and other animals seemingly solve the inverse dynamics problem has often inspired neuroscience research to look for evidence of hierarchical or otherwise simplifying representations.

One proposal is that the nervous system has no model of dynamics, generating joint torques instead as a function of the error between perceived and desired dynamic states. Merton once even suggested [28] that movement was initiated not by direct neural impulses to the muscle, but by first contracting the muscle's stretch sensors, creating a discrepancy between sensor tension and muscle tension that the muscles then worked to correct.

While solutions based solely on feedback are attractive in their simplicity, they fail to account for observed stability in the face of significant feedback delay [20], and the surprisingly robust capacity for coordinated motion by deafferented[1] subjects. Cats, for instance, will continue to walk after being decerebrated[2], despinalized[3], and/or deafferented. Polit and Bizzi even showed that a deafferented monkey can continue to execute coordinated pointing movements [37]. This is particularly surprising in light of the fact that the reaches were not conducted from a standard starting position, and the monkey was prevented from seeing its own arm. With neither visual nor proprioceptive feedback to let the monkey know where its arm started the reach from, this strongly discourages the theory that motions are made by pre-computed solutions to bounded-value problems. Polit suggested that the monkey controlled its arm by inducing a set of muscle tensions such that the arm's equilibrium position was the desired end position, thus driving the arm towards the target independent of perturbation or initial position. This echoed earlier suggestions of an "equilibrium point theory" [13]. Supporting evidnece came by the fact that applying a constant torque to the forearm caused the deafferented subject's final forearm direction to miss the target in the direction of the torque, but when the torque was released, the arm corrected its direction immediately.

One problem with this theory is that reaching by simply turning on an equilibrium point does not allow for trajectory control. The logical follow-up hypothesis, then, is that the equilibrium position of the arm smoothly moves from the start position to

[1]Preventing all in-bound nervous signals from reaching the spinal cord by disconnecting the afferent nerve roots on its dorsal side.

[2]Cutting through the midbrain.

[3]Cutting through the spinal cord at the thoracic level.

the goal position over the course of the movement, rather than being set there from the start. Bizzi et al. tested this [3] using the same experimental apparatus as above. This time, at the onset of muscle activity, an external actuator quickly moved the deafferented monkey's arm to the goal position. If the equilibrium position had been immediately placed at the goal position, the arm would have stayed there. Instead, the forearm partially backtracked toward the initial position before moving again to the goal. This suggested that when the arm was moved to the goal position, the equilibrium position was still near the initial position, pulling the arm back. Eventually, the equilibrium position reached the goal, bringing the arm with it. The trajectory of the equilibrium position was termed a "virtual trajectory."

Planning a virtual trajectory is an easier task than planning an actual trajectory, as the virtual trajectory need not follow Newtonian mechanics. The flip side is of course that the arm's Newtonian trajectory will differ from the planned virtual one. One way to reduce the discrepancy is by increasing the stiffness of the elastic force around the moving equilibrium position. Flash [14] simulated the actual trajectory of an arm executing a straight reach of moderate speed using a straight virtual trajectory. The resulting actual motion showed inflections and curves that were encouragingly similar to those observed in straight reaches by human subjects. However, the stiffness required to keep the hand close to its equilibrium position scales with the square of the speed, and later work by Bizzi et al showed that for fast forearm movements, the required stiffness would far exceed the recorded range of the human arm's dynamic mechanical impedance at those speeds [4]. Given such upper limits on stiffness, an alternative way to plan quick motions is to make the virtual trajectory itself curvy [19], outrunning the actual limb position near the beginning of a reaching motion and lagging behind it near the end. This is analogous to dragging a mass around by a rubber band.

The virtual trajectory formulation had seemingly provided a coherent perspective on the mechanics of muscles as biasable springs, the stability of animal movement, and the search for a computation-light solution to the inverse dynamics problem. If the virtual trajectory must assume a complex shape for quick, straight motions, the computational simplicity previously promised by virtual motion planning is lost.

Bizzi et al [5] introduced force fields to the biological motor control literature under the banner of equilibrium point control. Spinalized frogs had already been known to be capable of coordinated reflex movements. Bizzi found that stimulating certain points in the spinal cord of such a spinalized frog caused a specific coactivation of leg muscles, causing the leg to move to an equilibrium position with anisotropic stiffness. By sampling the restorative force exerted by the leg at various positions around this equilibrium position, Bizzi mapped what he called a convergent force field associated with that spinal module. When multiple spinal modules were simultaneously stimulated, the resulting force field was a simple vector sum of the modules' individual fields. This, the authors note, is "highly unexpected because of the complex nonlinearities that characterize the interactions both among neurons and between neurons and muscles." To emphasize this unexpectedness, they reported that random activation of available motor neuron pools produced stable convergent force fields only 5% of the time. Loeb et al [27] searched the space of binary (on or off) muscle activations of 16 frog leg muscles in simulation. Out of 65536 possible combinations of active muscles, only 23 yielded force fields that were convergent, stable, and robust to fluctuations in the activation levels of the muscles involved.

A year after the original force field paper, Bizzi et al reported results of recording force fields as functions of time, starting from the onset of spinal stimulus. Figures from the paper show how gradually adding one force field to another smoothly moves the equilibrium point while maintaining a convergent field around the equilibrium point at all times. This fulfills the criteria of a virtual trajectory. Subsequent studies on reflex actions by spinalized animals such as rats and cats demonstrate that it is likely that such coordinated reflex motions are parameterized and adjusted using force field primitives [24, 23, 31, 43]. These studies also demonstrate that force field motor control is not a fluke of amphibian evolution.

The discouragingly complex virtual trajectories that quick motions require might now be interpreted as the outcome of controlling weights to built-in force field representations. Kargo and Giszter [24] show that weights for different torque fields change in time according to a common temporal pulse, scaled by different magnitudes. This stereotyping of the temporal behavior of the weights further limits and simplifies the control space. While their experiments were performed on spinalized frogs, other work has suggested that humans also learn motor programs in a time-invariant fashion, adapting to time-varying external forces using purely position and velocity-dependent torque functions [8].

2.3 Robotics

Construction of motor control programs from independent primitives has been a recurring theme in robot control as well. Atkeson et al train locally windowed linear functions of state variables to quickly learn various tasks such as balancing a pole on end [1], and devil-sticking [41]. The local scope of these primitives helps avoid the destructive interference common to motor learning paradigms with global primitives such as sigmoidal neural networks [40].

Closer to this thesis is later work by Ijspeert, Nakanishi, and Schaal, who encode desired trajectories using learned ratios of fixed dynamical primitives [22]. They represent the velocity curves of individual degrees of freedom as linear combinations of simple pulse functions, and show a method to evolve the weights of such pulses to follow an example motion. By explicitly setting the velocity, however, their approach does away with Newtonian mechanics, relying on a high-frequency PD controller to enforce the velocities dictated by their algorithm. As such, their method relies on a full specification of a natural-looking path in order to itself produce natural-looking motion. The technique acts as an autonomous trajectory machine that plays back animations in a perturbation-resistant manner.

By applying inverse dynamics to a given arm motion, one can find the torques necessary to perfectly recreate that motion in simulation. An arm motion traces a path through joint space, and inverse dynamics associates torque vectors to each point along that path. Therefore, the problem of learning torque fields from arm motions is one of finding a vector field that smoothly interpolates between these configuration-torque pairs. Furthermore, this torque vector field must be a sum of a minimal set of basis vector fields. To this end, Mussa-Ivaldi devised a method for interpolating arbitrary sample vectors with linear combinations of a fixed set of simple irrotational[4] and solenoidal[5] fields. In a companion paper, Mussa-Ivaldi and Giszter apply this interpolation to the specific domain of a two-joint limb operating in a plane. Citing Colgate and Hogan [6, 7], they argue that fields used to control a linked limb must be irrotational in joint space as a precondition to stability. In the paper that most directly inspired this thesis, Mussa-Ivaldi solves for the coefficients of a linear combination of a fixed set of irrotational fields to actuate straight reaching motions in a plane [30]. This thesis extends that work to three-dimensional movements, and learns not just the torque field coefficients but also the parameters of the fields themselves.

2.4 Animation

One of the first cross-pollinations from robotics to computer animation was by Raibert and Hodgins [39], who extended robotic gait controllers for telescoping legs to the hopping of an animated planar kangaroo, actuating its angular joints with damped angular springs. This damped-spring actuator model was successfully applied to increasingly complex gymnastics in later work by Hodgins and Wooten [18, 46], generating animations of jogging, vaulting, and diving.

One drawback to their approach was that the control programs involved in actuating such coordinated movements are highly specialized, each requiring specific initial conditions. Faloutsos et al [11] devised controllers that allowed a range of initial conditions, in the service of a “virtual stuntman” that reacted to impacts and recovered from falls using such flexible control. Another problem with control through actuators is that controlling motion through low-level physical parameters such as spring stiffness and damping is highly unintuitive, as parameter adjustments more often lead

[4]Irrotational fields have zero curl and nonzero divergence.
[5]Solenoidal fields have zero divergence and nonzero curl.

to catastrophic failure than to an interesting change in style.

Less brittle control can be enacted through constrained optimization of a user-specified penalty function that encodes the desired motion style. Witkin and Kass [45], for example, solve for the joint torques of a "sluggish" leap by first constraining the takeoff and landing positions on the floor, then minimizing the power exerted by the actuators during the leap. A significant limitation of this approach is that optimizing a function entails repeated evaluation of its gradient, and that this operation is typically $O(D^2)$, where D is the number of degrees of freedom. Witkin and Kass tested their method on a Luxo lamp character with four degrees of freedom, but the method was prohibitively slow for fully articulated humanoid characters.

Difficulties notwithstanding, the notion of animating by "fitting" physics to sparse and intuitive user constraints remains appealing. Z. Popović and Witkin [38] edited the motions of complex humanoid figures by first "projecting" the motion to a simplified model, then applying constrained optimization on the fewer degrees of freedom, then "projecting" the modified motion back to the original character, filling in the lost data using heuristics on the original animation. The resulting motion is not strictly physically correct, but retains the perceptually salient characteristics of the simplified model's physically correct motion.

While constrained optimization is far more practical for animation than directly manipulating physical parameters, optimization is still an iterative design tool. One chooses a penalty function, optimizes with respect to it, observes the results, and repeats as needed. Preferably, one would specify a rigid body's trajectory directly, letting a computer decide on the necessary simulation parameters. This, however, is made difficult by a highly nonlinear and one-to-many mapping from trajectory to passive dynamic parameters. J. Popović solves this problem for the case of colliding rigid bodies, by locally linearizing the "simulation function," or the function that maps the simulation parameters to a body's state at a given time. A user's differential changes to the body's state can then be mapped to corresponding differential changes in the simulation parameters. Degeneracy is resolved by user-set preferences and stiffnesses on the parameter values.

A recent paper by Fang and Pollard [12] demonstrates that commonly used physical constraints can be formulated so that calculating their gradient is linear in the number of degrees of freedom, thus removing the previously prohibitive cost of optimizing for any reasonable number of character DOFs.

Chapter 3

Torque Fields

3.1 Introduction

A torque field is a vector field of joint torques, defined over the domain of limb configurations. The torques vary smoothly over the domain, and converge to a single point, the neutral configuration. As discussed in chapter 2, this force generator model is motivated by the fact that vertebrates seem to move limbs with predefined muscle co-contractions. Each of these co-contractions drives the limb to a neutral position at which the net torques vanish. The aggregate effect of the activated muscles is modeled as a convergent torque field around the neutral position, with anisotropic stiffness.

Section 3.2 extends Mussa-Ivaldi's concise parameterization [30] of such fields to 3-D limbs. Section 3.3 shows how the linear combinations are changed smoothly in time. Section 3.4 defines a cost function over the parameter space that can be minimized to solve for a set of torque fields that will approximately generate a given limb motion.

3.2 Parameterizing torque fields

3.2.1 Representing the neutral configuration

For an linked limb of n joints, the limb configuration can be represented as a stacked vector η of axis-angle vectors v_i:

$$\nu = \begin{bmatrix} v_{1x} & v_{1y} & v_{1z} & \dots & v_{nx} & v_{ny} & v_{nz} \end{bmatrix}^T \tag{3.1}$$

Torque fields apply torques as a function of the distance from the limb's current configuration ν to the neutral configuration η, thus requiring a distance metric in limb configuration space. Conveniently enough, taking the difference between axis-angle orientations yields the smallest-angle rotation ν_d between them[1]:

$$\nu_d = \nu - \eta; \tag{3.2}$$

One could make the torque field a linear spring in limb space by defining the restorative force as $-K\nu_d$ for some scalar $K < 0$. However, previous attempts at force field reconstruction have favored localized basis fields to facilitate the learning process. Poggio and Girosi [35, 36] note that using basis sets of Green's functions (localized radial wave-like functions that decay over distance), one can approximate arbitrary continuous scalar fields arbitrarily well. Mussa-Ivaldi [32, 33, 30] chooses the standard bulwark of localized bases, the radial Gaussian. Gaussians are used in this thesis as well.

3.2.2 Representing the stiffness

A Gaussian basis scalar field $G(\nu)$ is defined by a neutral limb configuration η and an inverse covariance matrix K:

$$G(\nu) = \exp\left(-\frac{1}{2}{\nu_d}^T K \nu_d\right) \tag{3.3}$$

[1] The reader is referred to section 3.3.5 in [9] for a proof.

The simplest Gaussians require few parameters, and increasing the complexity of K adds details in an intuitive manner. A radially symmetric Gaussian requires just a n-dimensional neutral configuration and a scalar K, corresponding to equal, isometric, and uncorrelated joint stiffnesses. Using diagonal K matrices allows different weights to the different dimensions of $\nu - \eta$, corresponding to different stiffnesses for rotations around the x,y, and z axes. Making K block-diagonal in 3 by 3 blocks allows different principal axes of stiffness, and a symmetric K allows the joint stiffnesses to be correlated to each other. To be a valid distance metric, K must be positive definite. One way to achieve this is to generate K as the product of a matrix with its transpose, $K = M^T M$. The experiments done in this thesis restrict K to be diagonal with positive diagonal elements, which also enforces positive-definiteness. See figure 3-1 for planar examples of torque fields with scalar, diagonal, and symmetric K matrices.

To generate torque vector fields from these Gaussian potentials, we take their gradient:

$$\chi(\nu) = \nabla G(x) = -K\nu_d \exp\left(-\frac{1}{2}{\nu_d}^T K \nu_d\right) \tag{3.4}$$

This defines an region of convergent torque around the equilibrium configuration η, with an ellipsoidal shape defined by K.

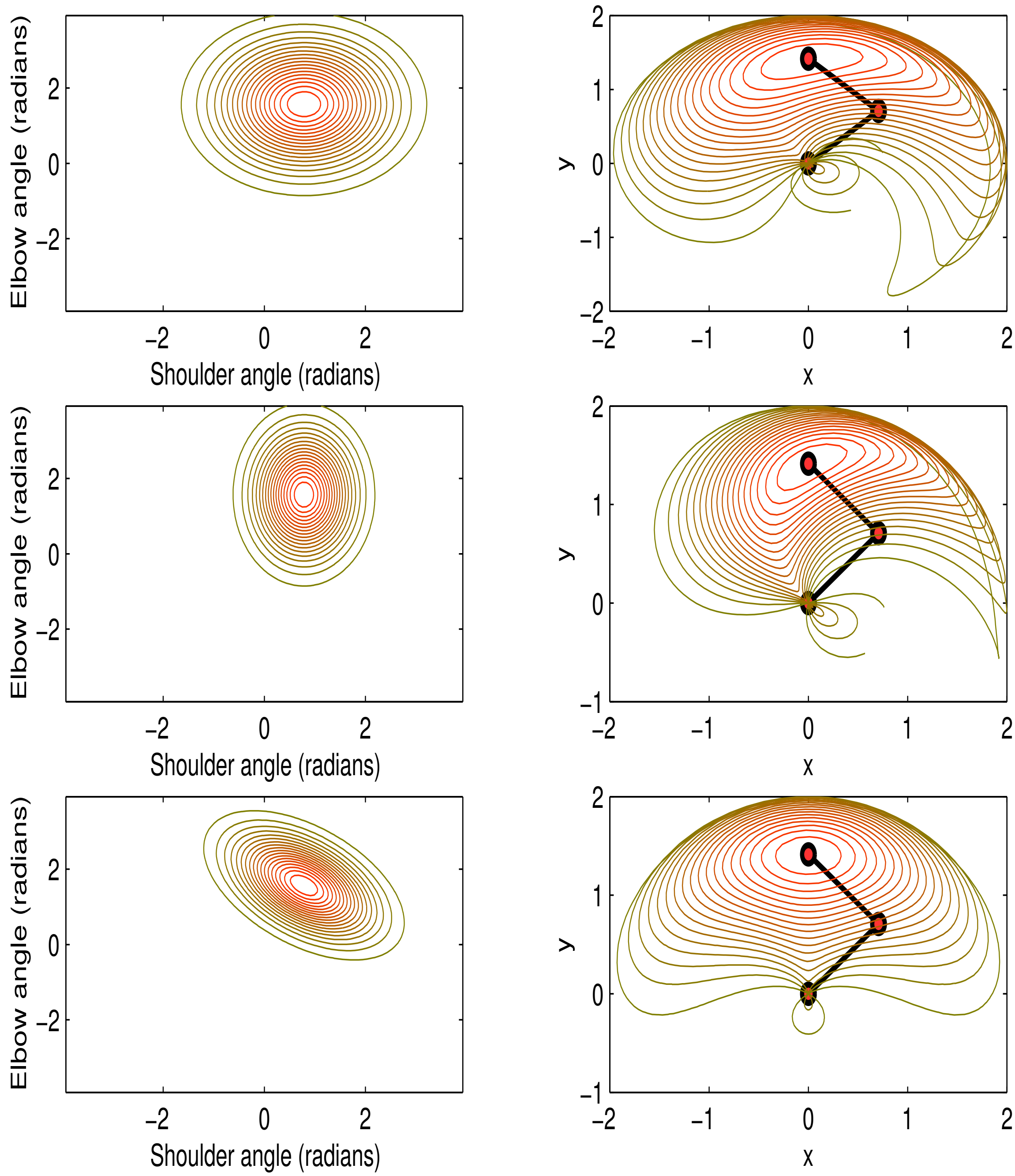
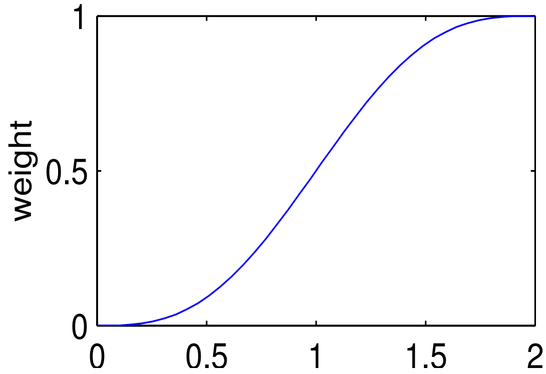


Figure 3-1: Torque fields with different K matrices. Left column: in joint angle space. Lines show isocontours of torque. Torque vectors are perpendicular to contours, pointing in. Right column: in Cartesian space. Lines show isocontours of the force at the end effector. Forces perpendicular to contours, pointing in. Arm shown, with shoulder joint at bottom. First row: scalar K. Second row: diagonal K. Third row: symmetric K.



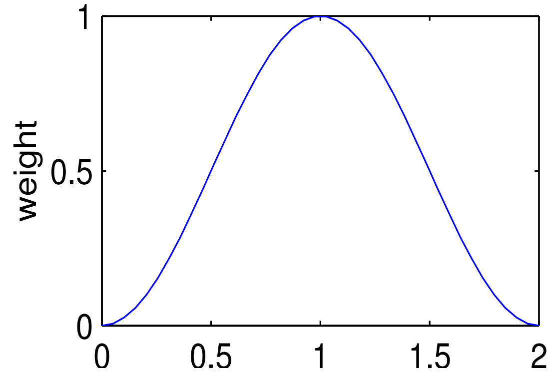


Figure 3-2: Step function $\psi(t)$ (equation 3.5) and pulse function$\phi(t)$ (equation 3.6).

3.3 Combining torque fields

Limb movement is generated by a smooth change over time of the basis fields' linear combination weights. Experimental evidence by Giszter et al. [17] shows that these changes are driven by a common function of time, resembling "a fixed half-cycle oscillation, or a fixed impulse response, such as a cosine packet." We use a smooth step function and a smooth pulse function to guide weight changes during a movement over a time duration λ:

$$\psi(t) = \begin{cases} 0 & (t < 0) \\ \frac{1}{\lambda}\left(t - \frac{\lambda}{2\pi}\sin\left(\frac{2\pi t}{\lambda}\right)\right) & (0 \leq t \leq \lambda) \\ 1 & (t > \lambda) \end{cases} \tag{3.5}$$

$$\phi(t) = \begin{cases} 0 & (t < 0) \\ \frac{1}{2}\left(1 - \cos\left(\frac{2\pi t}{\lambda}\right)\right) & (0 \leq t \leq \lambda) \\ 0 & (t > 0) \end{cases} \tag{3.6}$$

The basis set of M torque fields χ_i are combined using scaled versions of equations 3.5 and 3.6, to form the joint torques $F(t)$ as follows:

$$F(t) = \sum_{i=1}^{M} a_i\psi(t)\chi_i(\nu(t)) + b_i\phi(t)\chi_i(\nu(t)) \tag{3.7}$$

a_i and b_i are constant scalars, which we constrain to be nonnegative in order to ensure that the resulting field is locally convergent. We term those torque fields χ_i with nonzero a_i to be "step fields", and those with nonzero b_i "pulse fields". The step fields define the final equilibrium point, and the pulse fields modulate the equilibrium point trajectory in transit.

3.4 Solving for torque field parameters

In this section we write $F(t)$ from equation 3.7 as $F(t, x)$, where x is a parameter vector, containing the step weights a, pulse weights b, inverse covariance matrices K and neutral configurations ν. Given a motion $\nu(t)$, we wish to solve for the parameter set x_ν such that the torques $F(t, x_\nu)$ generate a similar motion. A straightforward approach adopted here is to minimize a cost function $C_\nu(x)$ that has a global minimum at $x = x_\nu$. We must first parameterize path $\nu(t)$, and define cost function C.

We represent motion $\nu(t)$ by sampling it at equal time intervals between $t = 0$ and $t = \lambda$, and stacking the sampled limb configurations into a column vector:

$$p = \begin{bmatrix} \nu(0) \\ \nu(\Delta t) \\ \nu(2\Delta t) \\ \vdots \\ \nu(\lambda) \end{bmatrix} \tag{3.8}$$

We can calculate the exact torque $T(t)$ needed to perfectly recreate motion $\nu(t)$ using inverse dynamics (see appendix A). An obvious cost function then is the integral of the squared difference between the "actual" torques $T(t)$ and those generated by the torque field model of equation 3.7, $F(t)$:

$$C(x, \nu(t)) = \int_{t=0}^{\lambda} \|T(t) - F(t)\|^2 \, dt \tag{3.9}$$

In practice, we use a discrete approximation of the above integral:

$$C(x, \nu(t)) = \sum_{i=0}^{\lambda/\Delta t+1} \|T(i\Delta t) - F(i\Delta t)\|^2 \Delta t \tag{3.10}$$

We use Sequential Quadratic Programming with numerical derivatives to minimize cost function C, subject to the following constraints:

- The torque field weights a and b must be nonnegative, to ensure that the torque fields are still convergent after being scaled.
- The inverse covariance matrices K are forced to be diagonal with positive elements, to preserve positive-definiteness, and keep the parameter space small.

The solution x_ν returned by the SQP run is tested by simulating a limb actuated by torque function $F(t, x_\nu)$, and seeing if the resulting motion is similar to input motion $\nu(t)$.

SQP is a form of gradient descent, and cannot be applied to discrete variables. One such variable in equation 3.7 is M, the number of basis fields χ. This is dealt with in an ad-hoc manner by repeating the search for decreasing values of M starting from some maximum, then choosing those value(s) of M whose solution generates a motion similar to the input motion.

3.4.1 Solving for multiple motions

We can decompose multiple motions into a common basis set of fields by straightforwardly extending the cost function to optimize:

$$C(x, \nu_1(t), \nu_2(t), \cdots, \nu_N) = \sum_{j=0}^{j=N-1} \sum_{i=0}^{\lambda/\Delta t+1} \|T_j(i\Delta t) - F(i\Delta t)\|^2 \Delta t \tag{3.11}$$

For M torque fields and N input motions, the parameter vector x will consist of one K and one η for each torque field, and MN as and MN bs.

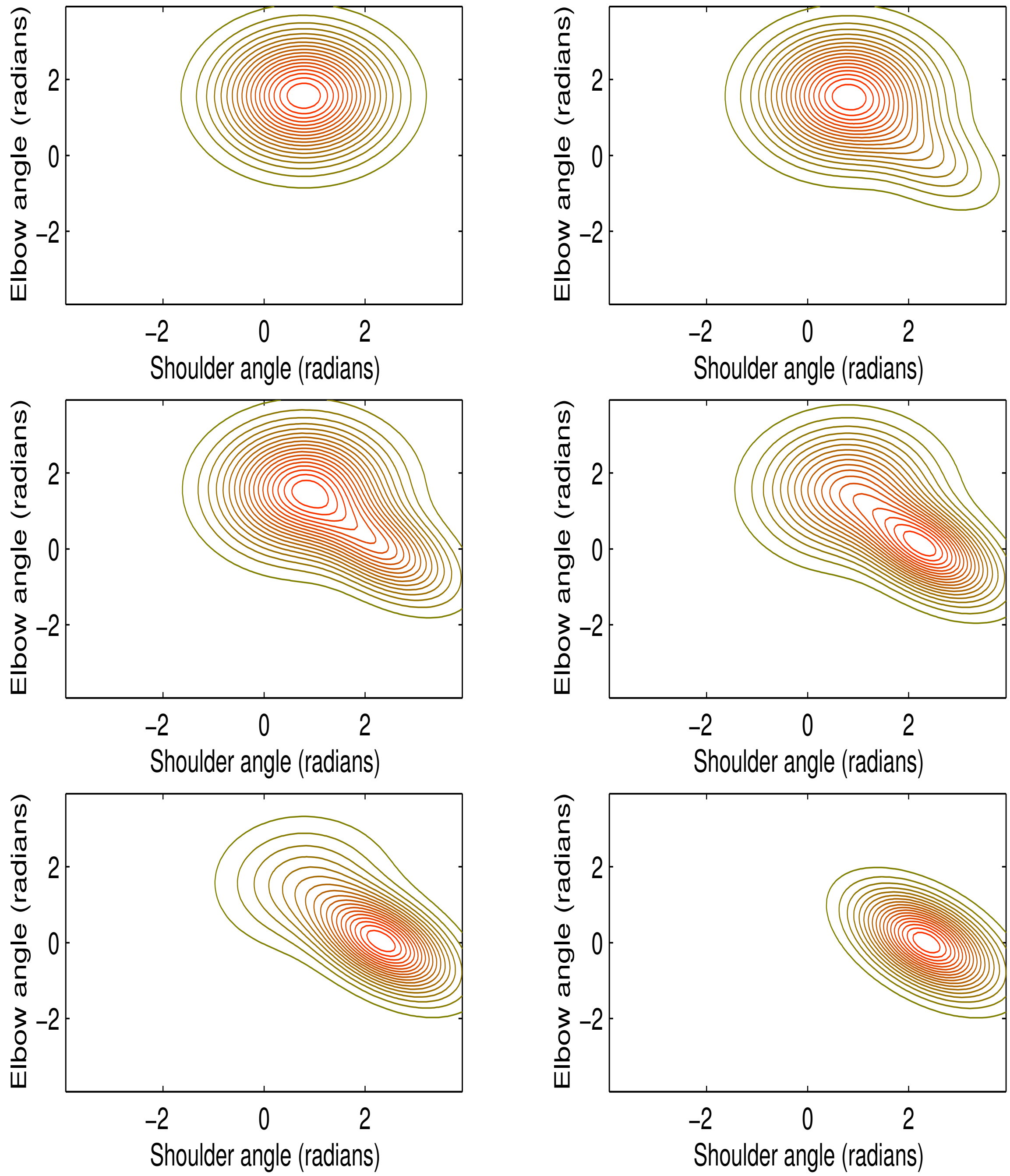


Figure 3-3: Shifting from the upper left torque field to the lower right torque field, using linear interpolation of the field weights. See 3-4 for resulting end effector force fields.

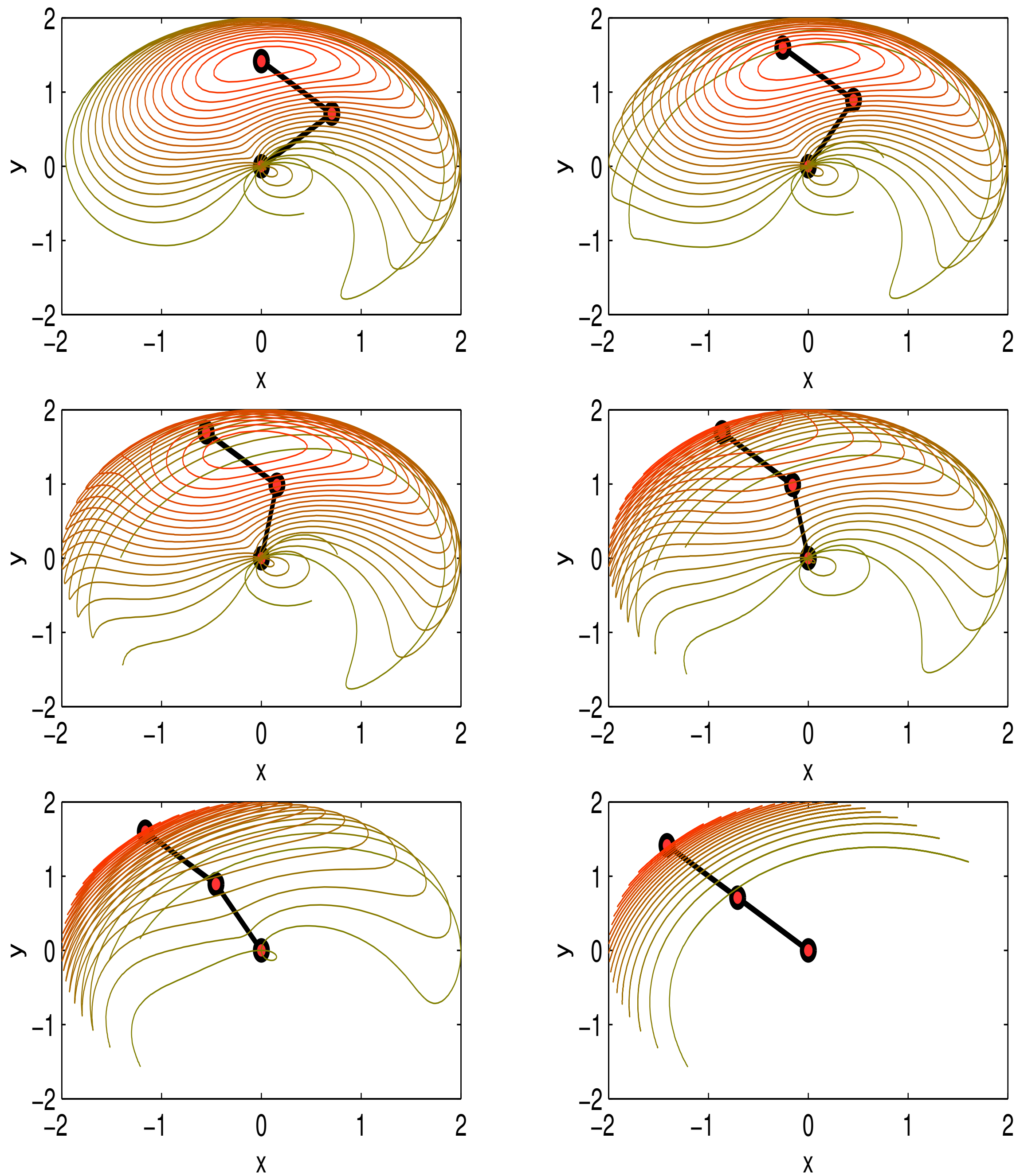


Figure 3-4: Force fields resulting from the torque fields of figure 3-3. Neutral arm configuration displayed for each composite field.

Chapter 4

Results

Testing the search method of chapter 3 involves two questions:

1. How robust is the searching algorithm against large parameter spaces?

2. How well does the searching algorithm match torque fields to torque functions of arbitrary input motions?

We tested each question against synthetic input motions, generated either from simulation or keyframed spline animations.

4.1 Determining the roughness of the search space

As Poggio and Girosi [35, 36] point out, the search for a Gaussian approximation of a multivariate function entails a non-convex search space. Gradient descent must therefore be augmented with stochastic search heuristics such as simulated annealing or multiple initial guesses. To this end, it helps to get a first-order sense of how far from the true solution one can start a search without getting trapped in a local minimum of the cost function (equation 3.10).

One way of gauging this distance is to generate a limb motion using a set of torque field parameters x, adding noise to x to make x_{guess}, then using the x_{guess} as an initial guess to the optimization of equation 3.10. This experiment was performed for three single-torque-field motions, with increasingly bad x_{guess} values, to find how much noise could be added to each motion's x before the optimizer failed to recover the original motion. x was "noised" by multiplying each parameter in it by a random number between 1 and r, where r varied between trials.

4.1.1 Experiment 1: Single torque field

Three single-field parameter sets were hand-crafted to generate a planar "come here" lifting gesture (figure 4-1), a planar straight-elbowed lifting gesture (figure 4-2), and a non-planar tucking motion (figure 4-3). The left image in each of the figures shows the original motion. The other two images show the motion after x has been noised and optimized, with different noise parameters r. Specifically, the center image shows the motion for the greatest r value tried that still generated motion similar to the original motion. The image on the right shows the motion from the next greatest r value tried. Note that for all three motions, this failure case resulted in no motion at all, not just incorrect motion. This was because the neutral configuration had moved far away from the arm's initial position, thus putting the arm out of the torque field's domain of influence. The torque field parameters for each motion in this set can be seen in appendix B.

It is interesting to note that the torque field parameters of many of the noised and successfully recovered motions are more than marginally different from those of the original motion. It is common that pulse fields are introduced where there were none, or that the values in the stiffness matrix K increase several times over. Furthermore, the robustness of the optimization process to noise parameter r seems to vary greatly with the particular motion being optimized. The non-planar tuck, for example, survives scaling by random numbers between 1 and 4.2, while the stiff-elbowed lift fails after being noised by $r = 1.8$.

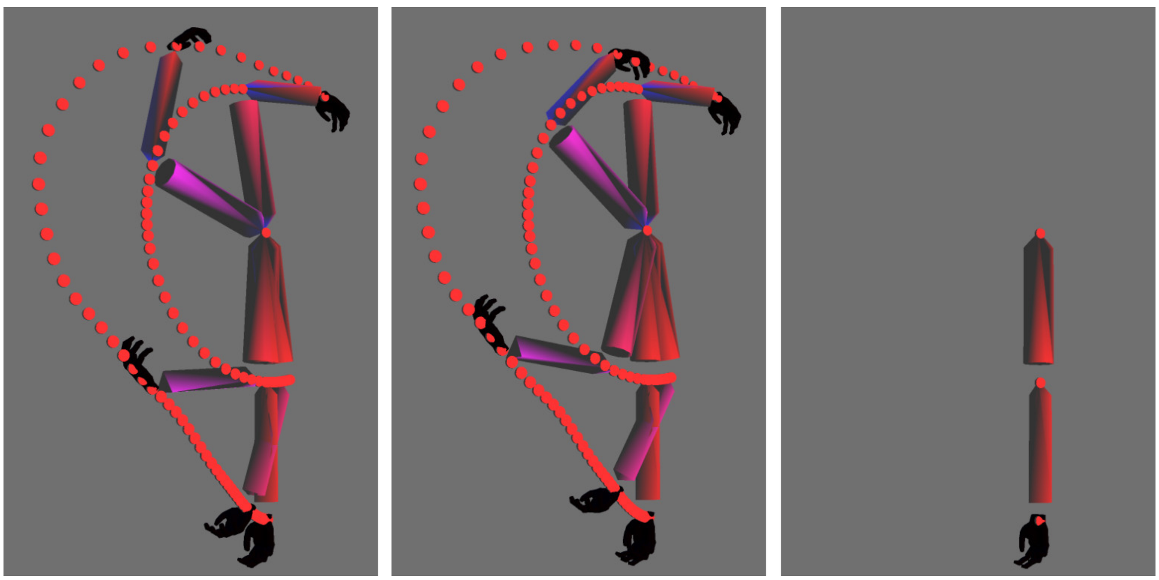


Figure 4-1: Left: A "come here" gesture. Center: The original gesture's torque field parameters were multiplied by random numbers between **1** and **2.6**, then optimized to fit the original trajectory. The motion resulting from these parameters is shown. Right: When the original gesture's parameters were scaled by random numbers between **1** and **3**, then optimized, the new parameters produced no motion. The torque field parameters for each figure are available in Appendix B.

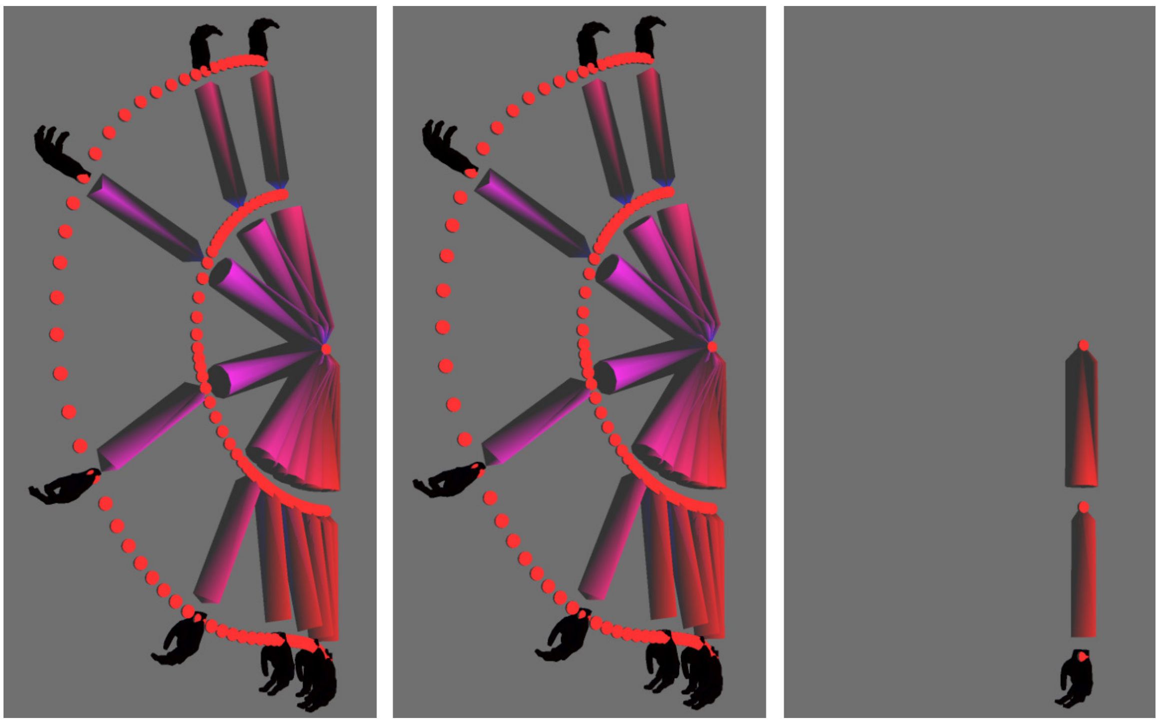


Figure 4-2: Left: A stiff-elbowed lift gesture. Center: The original gesture's torque field parameters were multiplied by random numbers between **1** and **1**.**4**, then optimized to fit the original trajectory. Right: When the original gesture's parameters were scaled by random numbers between **1** and **1**.**8**, then optimized, the new parameters produced no motion. The torque field parameters for each figure are available in Appendix B.

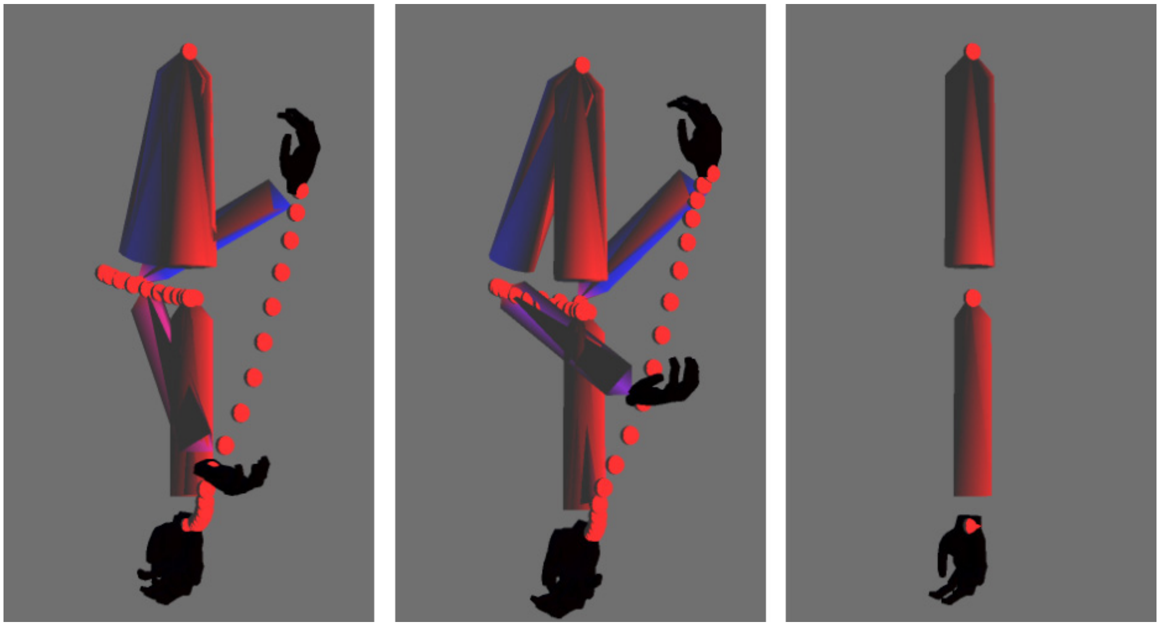


Figure 4-3: Left: A non-planar "tuck". Center: The original gesture's torque field parameters were multiplied by random numbers between **1** and **3.8**, then optimized to fit the original trajectory. Right: When the original gesture's parameters were scaled by random numbers between **1** and **4.2**, then optimized, the new parameters produced no motion. The torque field parameters for each figure are available in Appendix B.

4.1.2 Experiment 2: two torque fields

The same experiment was performed on motions of two torque fields, planar and non-planar. Figures 4-4 and 4-5 summarize the results. As in the previous examples, failure is catastrophic and not incremental, and torque field parameters may change significantly with almost no change to the resulting motion.

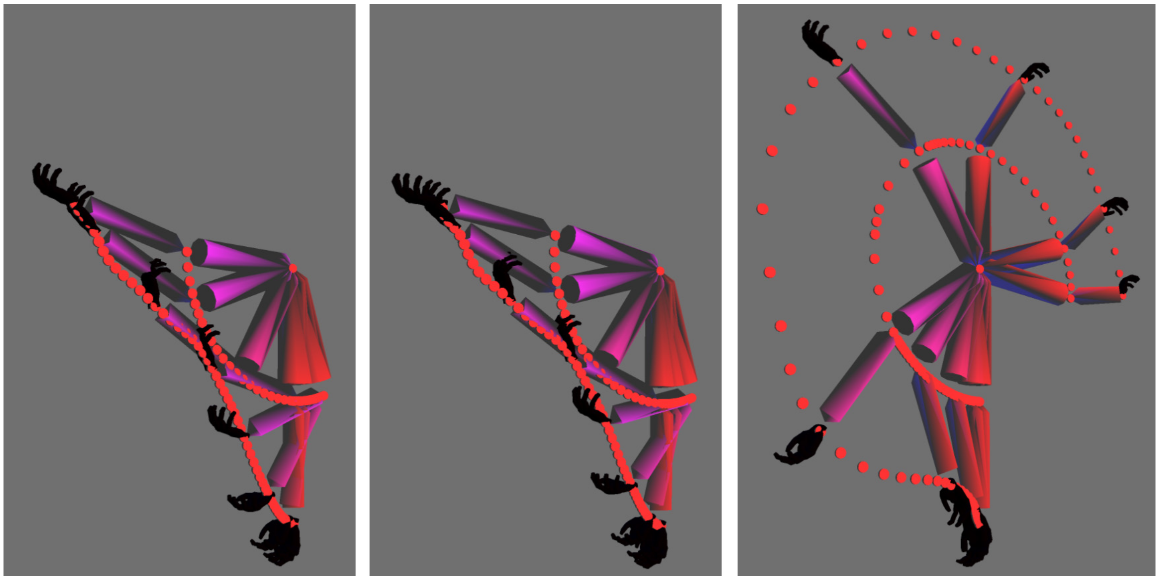


Figure 4-4: Left: A planar straight-line motion with two torque fields. Center: Gesture produced by the torque field parameters for the left figure after being multiplied by random numbers between **1** and **2**.**2**, then optimized to fit the original trajectory. Right: Gesture produced by noising the original torque field parameters by random numbers between **1** and **2**.**6**, then optimizing.

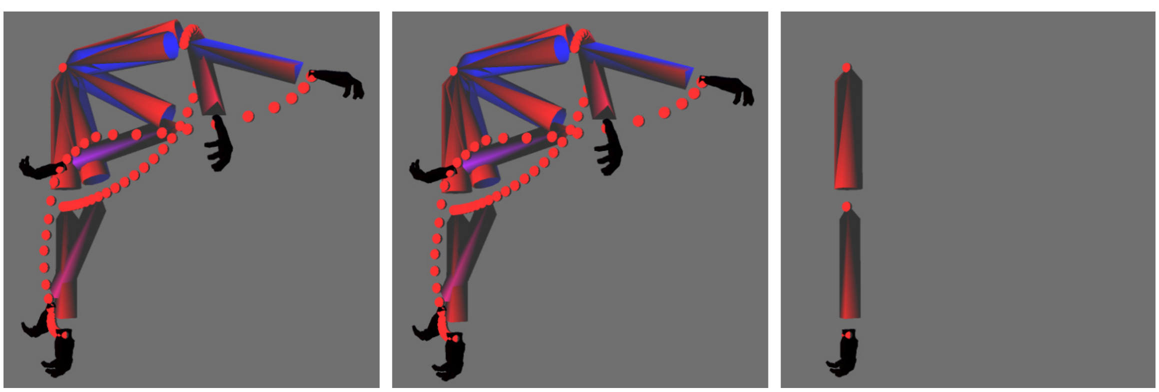


Figure 4-5: Left: A non-planar backwards reach using two torque fields. Center: Gesture produced by the torque field parameters for the left figure after being multiplied by random numbers between **1** and **2**.**2**, then optimized to fit the original trajectory. Right: Gesture produced by noising the original torque field parameters by random numbers between **1** and **2**.**6**, then optimizing (no motion).

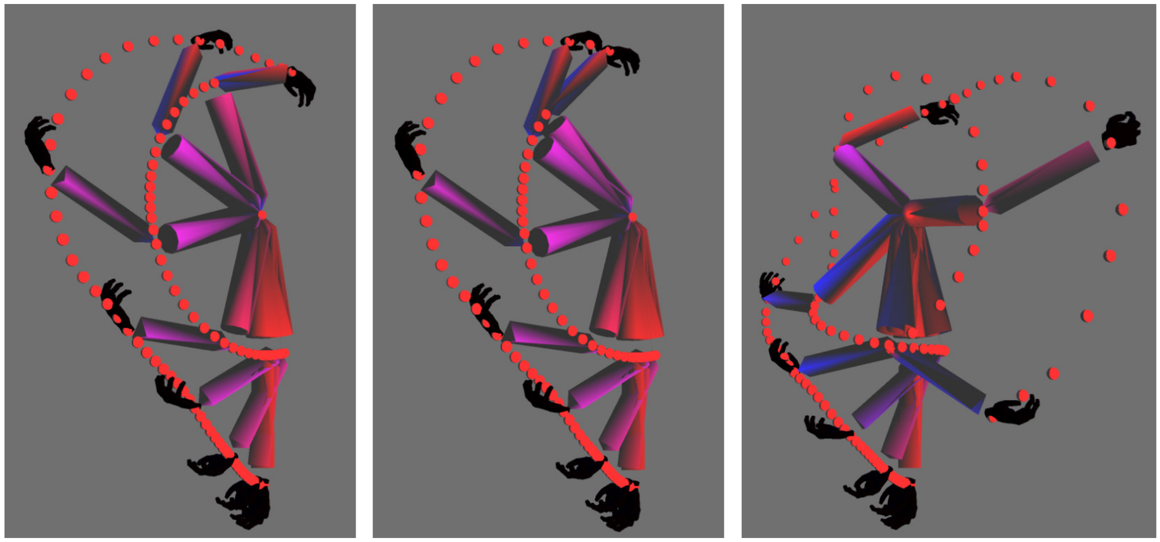


Figure 4-6: The "tuck" motion from figure 4-1, noised, then solved with four torque fields. Left: noise parameter $r = 1.8$. Center: $r = 2.2$. Right: $r = 2.6$. Note lessened resistance to noise as compared to previous experiment, where only one torque field was used.

4.2 Decomposing multiple motions into shared basis fields

Here again, we start from a known solution. The four torque fields responsible for the planar "come here" (figure 4-1), the non-planar tuck (figure 4-3), and the non-planar backwards reach (figure 4-5) were grouped as one parameter set x. Noised versions of x were used as initial guesses to finding a single torque field basis set to all four motions. The transition to failure becomes more gradual than in the previous trials, but tolerance for noise, on the average, is reduced.

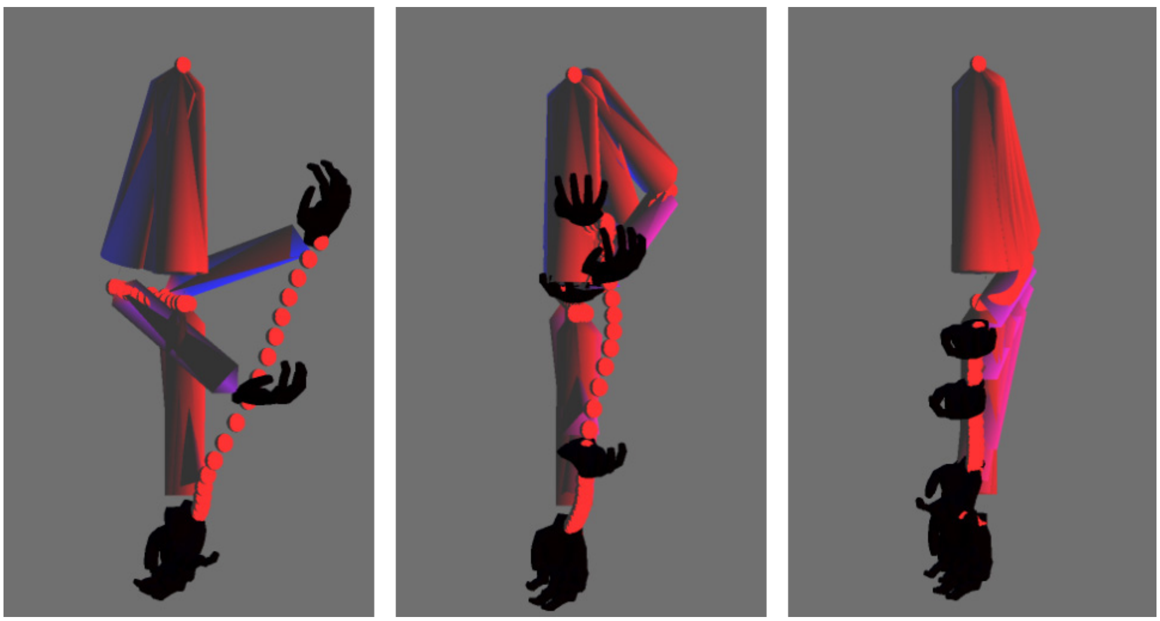


Figure 4-7: The "tuck" motion from figure 4-3, noised and solved with four torque fields. Left: noise parameter $r = 2.2$. Center: $r = 2.6$. Right: $r = 3.0$. Note lessened resistance to noise as compared to previous experiment, where only one torque field was used.

Figure 4-8: The backwards reach motion from figure 4-5, noised and solved with four torque fields. Left: noise parameter $r = 2.2$. Center: $r = 2.6$. Right: $r = 3.0$. In this case, the increase in the number of torque fields increased the motion's robustness to noise.

4.3 Decomposing motions not generated by torque fields

Mussa-Ivaldi decomposes planar motions into planar torque fields by using a fixed set of overlapping fields with evenly spaced neutral configurations, and optimizing solely on their weights, leaving the the torque fields themselves unchanged [32, 33]. While seeding the workspace with evenly spaced torque fields is a reasonable approach for a 2-DOF planar limb, in higher dimensions this becomes impractical. Even for a minimal 4-DOF limb with a shoulder and an elbow, a regular grid of neutral configurations placed at $3/4\pi$ intervals would require $3^4 = 81$ torque fields.

In the following experiment, a three-keyframe spline animation was created to resemble the "come here" animation of 4-1 and 1-1. This motion was decomposed into torque fields using the four-torque-field "come here" from figure 4-6 as an initial guess.

The search fails, perhaps explained by the fact that although the spline animation resembles the "come here" animation, its torques differ significantly, as can be seen by comparing figures 1-1 and 4-9.

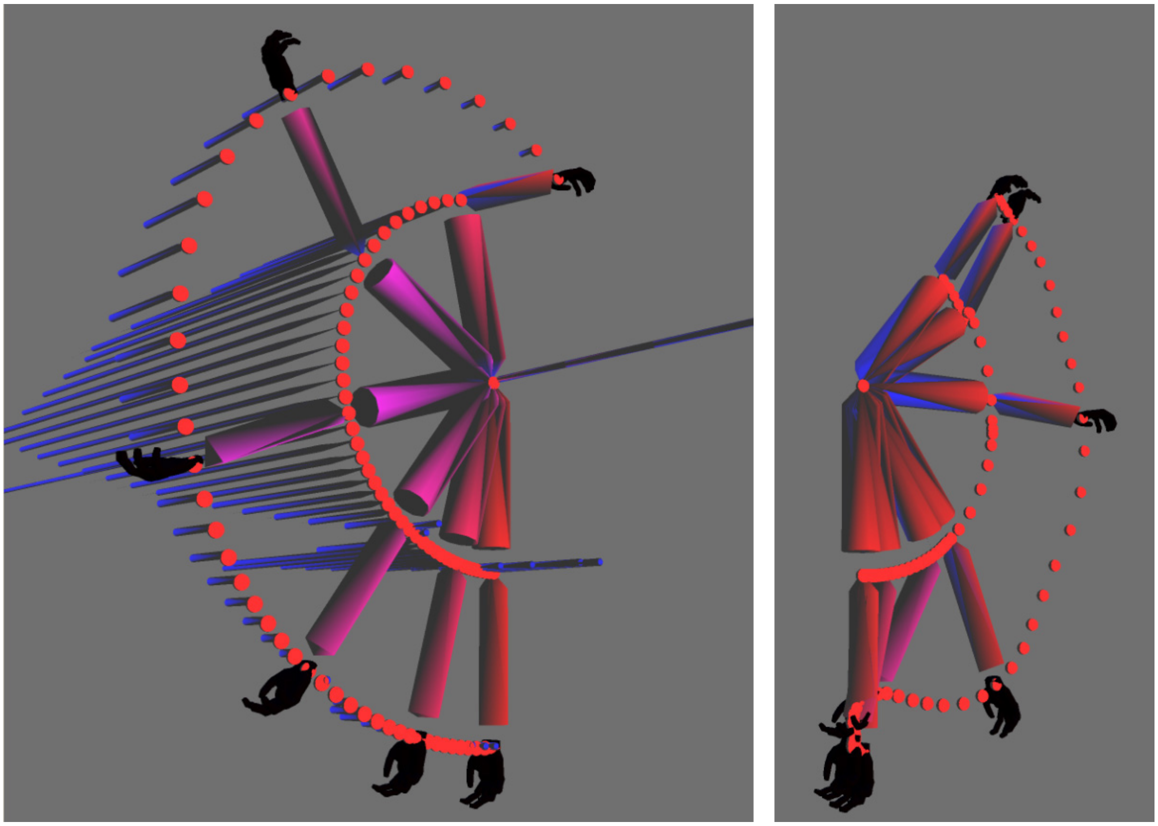


Figure 4-9: Left: a three-keyframe spline animation resembling the "come here" animation from figure 4-6. Right: the four torque field basis set from figures 4-6, 4-7, 4-8 was optimized to match the torques of the spline animation, with catastrophic results (torques not shown).

4.4 Conclusions

The optimizing searches in the preceding sections started from known solutions and worked outwards in parameter space, testing the extent of the well around the global minimum of cost function C. While some single-torque-field motions (e.g. figure 4-3) proved robust to very wrong initial guesses, others (figure 4-2) did not, and motions generated from more torque fields tended on the whole to be less robust to noise. It is clear that pure gradient descent is ill-suited to deducing multiple torque fields from a set of motions, and the high dimensionality of the search space will thwart random sprays of starting points. Some intelligent stochastic search method, such as simulated annealing or gradient descent with momentum, is required for a more reliably successful solver.

Given an animation, there is no clear way to make initial guesses to the appropriate torque fields. With regular, close-packed seeding of the parameter space ruled out by the curse of dimensionality, the animator is forced to make loose guesses. This combination of physical parameters with dependence on human expertise and labor makes this method of motion design a baroque affair.

This is not to say that torque fields will never be a good idea for computer animation. The search method used in the experiments above is but the simplest, and as discussed in the following section, there are many avenues of improvement. It is the conclusion of this thesis, however, that torque fields are certainly not low-hanging animation fruit to be plucked from an outside field. There is much climbing to be done.

4.5 Future work

This work is an initial exploration, leaving many unexplored search algorithms, torque field representations, data sets, and potential applications.

4.5.1 Search algorithms

Stochastic searches

Poggio and Girosi wrote [35, 36] on a problem similar to that in this work, namely that of approximating a multivariate function f from sparse samples, using a linear combination of Gaussians of various scales. (Approximating a scalar field with Gaussians is equivalent to approximating a vector field with gradients of Gaussians, if the vector field is irrotational.) They note that functionals of f are in general non-convex, and attempts at gradient descent should include stochastic terms to avoid local minima. In addition, they suggest that other iterative methods, such as "conjugate gradient, simulated annealing, or variations of the Metropolis algorithm [16] may be better than gradient descent and should be used in practice."

EM for vector fields

Mussa-Ivaldi takes the approach of using a fixed set of torque fields and solving only for their optimal weighting to interpolate between given torque samples [33, 30]. An interesting extension to this approach would be to iterate by solving for the optimal weights, then updating the field centers and variances, in the manner of Expectation-Maximization.

Path matching

The cost function of equation 3.10 only compares torques, and not lower-order values such as velocity and position, which are the more salient qualities in the perceptual comparison of two trajectories. This was motivated by an implicit assumption that the input motion is driven by torque fields to be searched for, and the most direct cost metric for such a search was a comparison of torques. Comparing trajectories would also greatly increase the search time, as each iteration in the gradient descent would require a full simulation using that iteration's torque fields.

4.5.2 Representations

Variable phase

One admittedly arbitrary assumption made in this work is that the time-step and time-pulse functions $\psi(t)$ and $\phi(t)$ (equations 3.5 and 3.6) start at the onset of the movement, and have periods equal to the duration of the movement. The general case would be to associate with each torque field a step or pulse function with its own start time and duration. Such a general treatment was avoided in this work to keep the parameter space small, but future work that aims to accommodate more than simple, discrete movements will likely need such flexibility.

Rotational fields

This thesis limits itself to Gaussian primitives for simplicity's sake, but it may be that using multiple types of primitives leads to a flexible solution that can do more with fewer parameters. In a study of primitive-based vector field approximation [32], Mussa-Ivaldi shows that fields with a rotational component are poorly approximated by even large numbers of irrotational primitives, while a small number of irrotational and solenoidal basis fields reconstructs it accurately. As noted in Chapter 2, Mussa-Ivaldi and Giszter have argued that torque fields must be irrotational for control stability. It may be, however, that in the context of short, time-pulsed primitives, such concerns for stability are outweighed by the flexibility in control.

Impedance

Starting with Bizzi's original paper on force fields [5], measuring force fields has always been a static task. The experimenter attaches the subject's limb to an unyielding force transducer, stimulates the subject's spine, and records the force with which the limb pushes against the transducer. Any velocity sensitivity of the primitive must go unmeasured, as the technology does not exist to measure the dynamic impedance of a limb in motion. As a result of this laboratory limitation, mathematical models of torque fields are often without impedance. At best, they include an unspecified function of velocity, as in [29]. In the absence of guidance from biology, future work may use any number of stiffness models, chosen primarily on practical grounds from an animation and control standpoint.

Cartesian force fields

Discussions of Cartesian force fields far outnumber those of angle-space torque fields in the biological literature, which may lead the reader to wonder why this thesis concerns itself with the less intuitive, higher-dimensional angle-space formulation. One answer is that Cartesian force fields are the manifest result of coordinated coactivations of muscle pairs, and it would be a mistake to let the force field itself drive the limb by its end. Pushing around the end-effector with a force field underspecifies the joint paths, and a limb modeled as a limp chain dragged by its end should not be trusted to look like anything else.

That being said, one could define a force field that exerts forces on each joint, giving it control over the whole limb and not just its end. On first inspection, it would seem that searching for the parameters of such a field would be more difficult that with a torque field of equal dimensionality, as the geometry of a linked limb limits the space of joint forces to be a subset of the set of all possible external forces. For example, when applying an external force to the elbow, any force component parallel to the upper arm will have no effect. Much of the force field space will be "redundant", complicating automated searches for particular trajectories. However, from a motion design perspective, Cartesian force fields are considerably more intuitive to visualize, potentially making them amenable to manual design. Visual design tools for vector fields are increasingly visible in the computer graphics literature [42, 44], possibly enabling such an approach. Cartesian force fields have the added advantage of being able to simply counter external influences such as gravity, or accelerations of the limb's root.

4.5.3 Datasets

Spinalized frogs

Perhaps the most attractive motion dataset to which to apply this work is that of frogs, specifically the limb motion of spinalized frogs having their spinal modules stimulated. Comparing the torque field generated from analyzing such a frog's motion with the actual recorded torque field that caused the motion would be an invaluable quality feedback loop.

Non-inertial frames

One class of motions that this work leaves untouched is that of limb motions with acceleration at the root. Successfully dealing with non-inertial frames is crucial if torque field motor control is to generate limb motion for animated characters.

4.5.4 Applications

Motion filtering

Mussa-Ivaldi used a set of predetermined torque fields to generate planar reaches along straight lines. The resulting motions demonstrated the same curves and inflections characteristic of “straight” reaches as performed by humans and primates. In the context of computer animation, this could be described as a motion filter that takes a crudely specified motion (a straight line) and performs it in a more natural manner, with Newtonian dynamics preserved. In three dimensions, the curse of dimensionality may prevent us from applying a similar approach with uniformly spaced torque fields in joint angle space. However, a set of torque fields trained on a particular class of motions (for example, straight reaches,) may be primed to reconstruct crudely specified motions of that class. In such an application, the path-based cost function mentioned in section 4.5.1 may be particularly applicable.

Motion transfer

Torque fields represent the local stiffness resulting from co-contracting a particular set of muscles, thereby encoding muscle group strength profiles in summary form. If a set of torque fields trained on the motions of one person encodes enough of his or her characteristic joint strengths, motions filtered through those torque fields (see “motion filtering” above) may exhibit body language recognizable as belonging to that individual. If the input motion is captured from another individual, this instance of motion filtering may be termed motion transfer.

Appendix A

Calculating the torques of an animation

Given a series of frames $\nu[i]$ represented as axis-angle vectors, (see equation 3.1), we would like to use inverse dynamics to calculate the torques needed to cause such a motion. For this we need continuously evaluable expressions for the angular acceleration at each joint. The obvious way to do this would be to use cubic interpolation between the frames $\nu[i]$ to make a continuous, twice-differentiable path $\nu(t)$, then use $\dot{\nu}(t)$ and $\ddot{\nu}(t)$ to analytically evaluate angular acceleration. However, the mapping from derivatives of angle-axis vectors to angular acceleration is undefined when $\nu(t) = 0$, as shown below:

$$
\begin{aligned}
q &= \left[\begin{array}{cc} \cos(\theta/2), & \sin(\theta/2)\,\hat{\nu} \end{array} \right] \\
\dot{q} &= \left[\begin{array}{cc} -\frac{\dot{\theta}}{2}\sin(\theta/2), & \frac{\dot{\theta}}{2}\cos(\theta/2)\,\hat{\nu} + \sin(\theta/2)\,\dot{\hat{\nu}} \end{array} \right] \\
\theta &= \sqrt{\nu \cdot \nu} \\
\dot{\theta} &= \frac{\nu \cdot \dot{\nu}}{\sqrt{\nu \cdot \nu}} \text{ (undefined when } \nu \cdot \nu = 0)
\end{aligned}
$$

Alternatively, we could calculate the angular acceleration numerically from $\nu(t)$, but inverse dynamics techniques require the desired trajectory to be specified by its accelerations, and these accelerations must be very precise to produce the original motion when twice-integrated. Such precision requirements typically rule out numerical evaluation of the angular acceleration. Instead, we start by converting the vectors $\nu[i]$ into quaternions $q[i]$. These quaternions can be spherically interpolated with $C1$ continuity using a Hermite spline modified with the following changes:

Vector operations	replaced with	Quaternion operations
$v_a + v_b$		$q_b q_a$
$v_a - v_b$		$q_b \tilde{q}_a$

$\tilde{q}$ denotes the conjugate of unit quaternion q. For details on spherical cubic interpolation of quaternions, see [10], [25]. For an introduction to Hermite splines, see [26][1].

The Hermite spline formulation makes it easy to evaluate its first and second derivatives $\dot{q}(t)$, and $\ddot{q}(t)$ (again, see [26]). Using these, the angular velocity and angular accelerations can be calculated as follows:

$$\omega = 2\dot{q}\tilde{q} \tag{A.1}$$

$$\dot{\omega} = 2(\dot{q}\dot{\tilde{q}} + \ddot{q}\tilde{q}) \tag{A.2}$$

Inverse dynamics can then be applied to $\dot{\omega}(t)$ to yield the joint torques.

[1]Note that the original Hermite spline paper [26] contains errors regarding the accommodation of irregularly placed keypoints. See [10] for details.

Appendix B

Torque field parameters of animations in section 4.1.1

B.1 Symbol key:

See equation 3.7 to see these in context:

a	Step field weight
b	Pulse field weight
K	Stiffness matrix
I	Identity matrix
η	Neutral configuration
λ	Pulse step duration

B.2 Parameters of torque fields in figure 4-1

B.2.1 Parameters of left image (Original parameters)

$$
\begin{aligned}
a &= 3 \\
b &= 0 \\
K &= I \\
\eta &= \begin{bmatrix} \begin{bmatrix} -\pi/2 & 0 & 0 \end{bmatrix}^T \\ -\pi/2 \\ \begin{bmatrix} -\pi/4 & 0 & 0 \end{bmatrix}^T \end{bmatrix} \\
\lambda &= 3
\end{aligned}
$$

B.2.2 Noised parameters 1

These are the above parameters, multiplied by random numbers from 1 to 2.6

$$
\begin{aligned}
a &= 5.94996 \\
b &= 0 \\
K &= 1.76955I \\
\eta &= \begin{bmatrix} \begin{bmatrix} -2.08069 & 0 & 0 \end{bmatrix}^T \\ -3.0946 \\ \begin{bmatrix} -0.997603 & 0 & 0 \end{bmatrix}^T \end{bmatrix} \\
\lambda &= 5.15485
\end{aligned}
$$

B.2.3 De-noised parameters 1 (center image)

These are the above parameters, after being optimized to generate torques close to those of the original motion.

$$
\begin{aligned}
a &= 1.16717 \\
b &= 2.32159 \\
K &= 1.03443I \\
\eta &= \begin{bmatrix} \begin{bmatrix} -1.5743 & -1.55804\times 10^{-7} & -1.55804\times 10^{-7} \end{bmatrix}^T \\ -1.57132 \\ \begin{bmatrix} -0.706938 & -1.55998\times 10^{-7} & 7.82389\times 10^{-8} \end{bmatrix}^T \end{bmatrix} \\
\lambda &= 5.15485
\end{aligned}
$$

B.2.4 Noised parameters 2

These are the original parameters of section B.2.1, multiplied by random numbers between 1 and 3.

$$
\begin{aligned}
a &= 10.5898 \\
b &= 0 \\
K &= 3.54581I \\
\eta &= \begin{bmatrix} \begin{bmatrix} -3.1249 & 0 & 0 \end{bmatrix}^T \\ -2.65996 \\ \begin{bmatrix} -1.52254 & 0 & 0 \end{bmatrix}^T \end{bmatrix} \\
\lambda &= 4.60259
\end{aligned}
$$

B.2.5 De-noised parameters 2 (right image)

These are the above parameters, after optimization.

$$
\begin{aligned}
a &= 10.3837 \\
b &= 0.00109853 \\
K &= 6.42748I \\
\eta &= \begin{bmatrix} \begin{bmatrix} -3.13611 & -6.44742e-010 & -6.44742e-010 \end{bmatrix}^T \\ -3.13018 \\ \begin{bmatrix} -3.14159 & 0 & 0 \end{bmatrix}^T \end{bmatrix} \\
\lambda &= 4.60259
\end{aligned}
$$

B.3 Parameters of torque fields in figure 4-2

B.3.1 Original Parameters (left image)

$$
\begin{aligned}
a &= 2 \\
b &= 0 \\
K &= 3I \\
\eta &= \begin{bmatrix} \begin{bmatrix} -\pi/2 & 0 & 0 \end{bmatrix}^T \\ 0 \\ \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T \end{bmatrix} \\
\lambda &= 3
\end{aligned}
$$

B.3.2 Noised parameters 1

$$
\begin{aligned}
a &= 2.27977 \\
b &= 0 \\
K &= 3.53835I \\
\eta &= \begin{bmatrix} \begin{bmatrix} -1.98382 & 0 & 0 \end{bmatrix}^T \\ 0 \\ \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T \end{bmatrix} \\
\lambda &= 3.02218
\end{aligned}
$$

B.3.3 De-noised parameters 1 (center image)

$$
\begin{aligned}
a &= 2.0007 \\
b &= 0.0169349 \\
K &= 2.99971I \\
\eta &= \begin{bmatrix} \begin{bmatrix} -1.57071 & 7.34373 \times 10^{-10} & 7.34437 \times 10^{-10} \end{bmatrix}^T \\ -3.00353 \times 10^{-5} \\ \begin{bmatrix} 3.74867 \times 10^{-6} & -9.73763 \times 10^{-9} & 4.12577 \times 10^{-8} \end{bmatrix}^T \end{bmatrix} \\
\lambda &= 3.02218
\end{aligned}
$$

B.3.4 Noised parameters 2

$$
\begin{aligned}
a &= 3.56097 \\
b &= 0 \\
K &= 5.30907I \\
\eta &= \begin{bmatrix} \begin{bmatrix} -2.22229 & 0 & 0 \end{bmatrix}^T \\ 0 \\ \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T \end{bmatrix} \\
\lambda &= 3.97692
\end{aligned}
$$

B.3.5 De-noised parameters 2 (right image)

$$
\begin{aligned}
a &= 0 \\
b &= 0 \\
K &= 3.13435I \\
\eta &= \begin{bmatrix} \begin{bmatrix} 3.14159 & 2.68594 \times 10^{-9} & 2.68594 \times 10^{-9} \end{bmatrix}^T \\ -3.14159 \\ \begin{bmatrix} 0.673388 & 0.673388 \times 10^{-9} & 0.673388 \times 10^{-9} \end{bmatrix}^T \end{bmatrix} \\
\lambda &= 3.97692
\end{aligned}
$$

B.4 Parameters of torque fields in figure 4-3

B.4.1 Original Parameters (left image)

$$
\begin{aligned}
a &= 2 \\
b &= 0 \\
K &= 2I \\
\eta &= \begin{bmatrix} \begin{bmatrix} 0 & -\pi/2 & 0 \end{bmatrix}^T \\ -pi/2 \\ \begin{bmatrix} -pi/4 & 0 & 0 \end{bmatrix}^T \end{bmatrix} \\
\lambda &= 3
\end{aligned}
$$

B.4.2 Noised parameters 1

$$
\begin{aligned}
a &= 4.68243 \\
b &= 0 \\
K &= 3.17164I \\
\eta &= \begin{bmatrix} \begin{bmatrix} 0 & 4.0218 & 0 \end{bmatrix}^T \\ -4.76139 \\ \begin{bmatrix} -1.71223 & 0 & 0 \end{bmatrix}^T \end{bmatrix} \\
\lambda &= 9.46756
\end{aligned}
$$

B.4.3 De-noised parameters 1 (center image)

$$
\begin{aligned}
a &= 0.465545 \\
b &= 4.2936 \\
K &= 1.97861I \\
\eta &= \begin{bmatrix} \begin{bmatrix} 0.00274648 & 1.57128 & 8.53986e-005 \end{bmatrix}^T \\ -1.57145 \\ \begin{bmatrix} -0.707378 & 0.000205817 & -0.00100284 \end{bmatrix}^T \end{bmatrix} \\
\lambda &= 9.46756
\end{aligned}
$$

B.4.4 Noised parameters 2

$$
\begin{aligned}
a &= 9.02322 \\
b &= 0 \\
K &= 7.43094I \\
\eta &= \begin{bmatrix} \begin{bmatrix} 0 & 2.71553 & 0 \end{bmatrix}^T \\ -4.48069 \\ \begin{bmatrix} -2.60628 & 0 & 0 \end{bmatrix}^T \end{bmatrix} \\
\lambda &= 13.1937
\end{aligned}
$$

B.4.5 De-noised parameters 2 (right image)

$$
\begin{aligned}
a &= 5.97473 \\
b &= 0 \\
K &= 4.73324I \\
\eta &= \begin{bmatrix} \begin{bmatrix} 0 & 3.14159 & 0 \end{bmatrix}^T \\ -2.845 \\ \begin{bmatrix} -3.14159 & 0 & 0 \end{bmatrix}^T \end{bmatrix} \\
\lambda &= 11.0543
\end{aligned}
$$

Bibliography

[1] Chris Atkeson and Stefan Schaal. Robot learning from demonstration. In D.H. Fisher Jr., editor, *Machine Learning: Proceedings of the Fourteenth International Conference*, volume 96, pages 12–20. Morgan Kaufmann, 1997.

[2] David Baraff and Andrew Witkin. Large steps in cloth simulation. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 43–54. ACM Press, 1998.

[3] Emilio Bizzi, Neri Accornero, William Chapple, and Neville Hogan. Posture control and trajectory formation during arm movement. *The Journal of Neuroscience*, 4(11):2738–2744, 1984.

[4] Emilio Bizzi, Neville Hogan, Ferdinando Mussa-Ivaldi, and Simon Giszter. Does the nervous system use equilibrium-point control to guide single and multiple joint movements? *Behavioral and Brain Sciences*, 15:603–613, 1992.

[5] Emilio Bizzi, Ferdinando Mussa-Ivaldi, and Simon Giszter. Computations underlying the execution of movement: A biological perspective. *Science*, 253(5017):287–291, 1991.

[6] Ed Colgate. *The control of dynamically interacting systems*. PhD thesis, MIT Department of Mechanical Engineering, 1988.

[7] Ed Colgate and Neville Hogan. An analysis of contact instability in terms of passive physical equivalents. In *IEEE Proceedings of the International Conference on Robotics and Automation*, pages 404–409, 1989.

[8] Michael Conditt and Ferdinando Mussa-Ivaldi. Central representation of time during motor learning. In *Proceedings of the National Academy of Sciences*, volume 96, pages 11625–11630, 1999.

[9] Erik B. Dam, Martin Koch, and Martin Lillholm. Quaternions, interpolation, and animation. Technical Report DIKU-TR-98/5, University of Copenhagen, July 1998.

[10] Eberly. www.magic-software.com. Website.

[11] Petros Faloutsos, Michiel van de Panne, and Demetri Terzopoulos. Composable controllers for physics-based character animation. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 251–260. ACM Press, 2001.

[12] Anthony C. Fang and Nancy S Pollard. Efficient synthesis of physically valid human motion. In *Proceedings of the 30th annual conference on Computer graphics and interactive techniques*. ACM Press, 2003.

[13] A. G. Feldman. Functional tuning of the nervous system with control of movement or maintenance of a steady posture - ii. controllable parameters of the muscles. *Biofizika*, 11(3):498–508, 1966.

[14] Tamar Flash. The control of hand equilibrium trajectories in multi-joint arm movements. *Biological Cybernetics*, 57:257–274, 1987.

[15] Nick Foster and Ronald Fedkiw. Practical animation of liquids. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 23–30. ACM Press, 2001.

[16] F. Girosi, T. Poggio, and B. Caprile. Extensions of a theory of networks for approximation and learning: outliers and negative examples. In *Proceedings of the IEEE Conference on Neural Informaiton Processing Systems*, Denver, Colorado, November 1990. Morgan Kaufmann Publishers.

[17] Simon F. Giszter, Karen A. Moxon, Ilya A. Rybak, and John K. Chapin. Neurobiological and neurorobotic approaches to control architectures for a humanoid motor system. *Robotics and Autonomous Systems*, 37:219–235, 2001.

[18] Jessica K. Hodgins, Wayne L. Wooten, David C. Brogan, and James F. O'Brien. Animating human athletics. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 71–78. ACM Press, 1995.

[19] Neville Hogan. An organizing principle for a class of voluntary movements. *The Journal of Neuroscience*, 4(11):2745–2754, 1992.

[20] Neville Hogan, Emilio Bizzi, Ferdinando Mussa-Ivaldi, and Tamar Flash. Controlling multi-joint motor behavior. *Exercise and Sport Science Reviews*, 15:153–190, 1987.

[21] John M. Hollerbach and Tamar Flash. Dynamic interactions between limb segments during planar arm movements. *Biological Cybernetics*, 44:67–77, 1982.

[22] Auke Jan Ijspeert, Jun Nakanishi, and Stefan Schaal. Trajectory formation for imitation with nonlinear dynamical primitives. In *IEEE International Conference on Intelligent Robots and Systems*, 2001.

[23] William Kargo and Simon Giszter. Afferent roles in hindlimb wipe-reflex trajectories: Free-limb kinematics and motor patterns. *Journal of Neurophysiology*, 83:1480–1501, 2000.

[24] William Kargo and Simon Giszter. Rapid correction of aimed movements by summation of force-field primitives. *Journal of Neuroscience*, 20(1):409–426, 2000.

[25] Myoung-Jun Kim, Myung-Soo Kim, and Sung Yong Shin. A general construction scheme for unit quaternion curves with simple high order derivatives. *Computer Graphics*, 29(Annual Conference Series):369–376, 1995.

[26] Doris H. U. Kochanek and Richard H. Bartels. Interpolating splines with local tension, continuity, and bias control. In *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, pages 33–41, 1984.

[27] E. P. Loeb, Simon Giszter, Phillipe Saltiel, Emilio Bizzi, and Ferdinando Mussa-Ivaldi. Output units of motor behavior: An experimental and modeling study. *Journal of Cognitive Neuroscience*, 12(1):79–97, 2000.

[28] P. A. Merton. How we control the contraction of our muscles. *Scientific American*, 226:30–37, 1972.

[29] F.A. Mussa-Ivaldi and E. Bizzi. Motor learning through the combination of primitives. *Philosophical Transaction of the Royal Society of London B*, 355, 1755–1769.

[30] Ferdinando Mussa-Ivaldi. A distributed system of control primitives for representing and learning movements. In *Proceedings of the 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, page 84, Monterey, CA, June 1997. IEEE.

[31] Ferdinando Mussa-Ivaldi, Simon Giszter, and Emilio Bizzi. Linear superposition of primitives in motor control. In *Proceedings of National Academy of Sciences*, volume 91, pages 7534–7538, 1994.

[32] Ferdinando A. Mussa-Ivaldi. From basis functions to basis fields: vector field approximation from sparse data. *Biological Cybernetics*, 67:479–489, 1992.

[33] Ferdinando A. Mussa-Ivaldi and Simon F. Giszter. Vector field approximation: a computational paradigm for motor control and learning. *Biological Cybernetics*, 67:479–489, 1992.

[34] Duc Quang Nguyen, Ronald Fedkiw, and Henrik Wann Jensen. Physically based modeling and animation of fire. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 721–728. ACM Press, 2002.

[35] T Poggio and F Girosi. Networks for approximation and learning. In *IEEE*, volume 78, pages 1487–1497, 1990.

[36] T Poggio and F Girosi. A theory of networks for learning. *Science*, 247:978–982, 1990.

[37] A. Polit and Emilio Bizzi. Characteristics of motor programs underlying arm movements in monkeys. *Journal of Neurophysiology*, 42:183–194, 1979.

[38] Zoran Popović and Andrew Witkin. Physically based motion transformation. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 11–20. ACM Press/Addison-Wesley Publishing Co., 1999.

[39] Marc H. Raibert and Jessica K. Hodgins. Animation of dynamic legged locomotion. In *Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, pages 349–358. ACM Press, 1991.

[40] Stefan Schaal and Chris Atkeson. Constructive incremental learning from only local information. *Neural Computation*, 10:2047–2084, 1998.

[41] Stefan Schaal, Chris Atkeson, and Sethu Vijayakumar. Real-time robot learning with locally weighted statistical learning. In *Machine Learning: Proceedings of the Fourteenth International Conference*, 2000.

[42] Yiying Tong, Santiago Lombeyda, Anil N. Hirani, and Mathieu Desbrun. Discrete multiscale vector field decomposition. In *Proceedings of the 30th annual conference on Computer graphics and interactive techniques*. ACM Press, 2003.

[43] M.C. Tresch and Emilio Bizzi. Responses to spinal microstimulation in the chronically spinalized rat and their relationship to spinal systems activated by low threshold cutaneous stimulation. *Experimental Brian Research*, 129:401–416, 1999.

[44] Adrien Treuille, Antoine McNamara, Zoran Popović, and Jos Stam. Keyframe control of smoke simulations. In *Proceedings of the 30th annual conference on Computer graphics and interactive techniques*. ACM Press, 2003.

[45] Andrew Witkin and Michael Kass. Spacetime constraints. In *Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, pages 159–168. ACM Press, 1988.

[46] Wayne Wooten and Jessica Hodgins. Simulation of human diving. In *Graphics Interface*, 1995.