5-6-1999

Using Haptic Vector Fields for Animation Motion Control

Bruce Randall Donald
Dartmouth College

Frederick Henle
Dartmouth College

Using Haptic Vector Fields for Animation Motion Control

Technical Report PCS-TR99-353

Bruce Randall Donald and Frederick Henle
Dartmouth Computer Science Department
{brd, henle}@cs.dartmouth.edu

May 6, 1999

Abstract

We are exploring techniques for animation authoring and editing using a haptic force-feedback device. In our system, a family of animations is encoded by a bundle of trajectories. This bundle in turn defines a time-varying, higher-order vector field on a configuration space for the animation. A haptic input device provides a low-dimensional parameterization of the resulting dynamical system, and the haptic force feedback permits browsing and editing of the space of animations, by allowing the user to experience the vector field as physical forces.

1 Introduction

It is inevitable that computers someday use touch as a medium both for input and output. Haptic interfaces in the form of computer peripherals are rapidly becoming less expensive and more widely available. While some applications for haptics in computer graphics may be immediately useful (such as "touchable" virtual reality) we believe a less obvious yet fruitful paradigm is to use the haptic device as a sophisticated input device for exploring and driving complex dynamical systems such as computer models for animation. In our system, a family of animations is encoded by a *bundle* of trajectories. This bundle in turn defines a time-varying, higher-order vector field (HOVF) on a configuration space for the animation. A haptic input device provides a low-dimensional parameterization of the resulting dynamical system, and the haptic force feedback permits browsing and editing of the space of animations, by allowing the user to experience the vector field as physical forces. Informally, the HOVF may be thought of as a vector field on haptic control space that varies with position, velocity, and time. A *haptic map* from control space to animation space enables the HOVF to operate as follows: (a) it defines a control system for the haptic device, (b) it encodes the control system for an animation, and (c) it implements a bidirectional *connection* between (a) and (b).

One goal of our work is to test the following hypotheses:

- The vector field representation is useful for encoding a family of animations.
- Haptics is a good technique for sensing, browsing, modifying, editing, storing, and interacting with this representation.

To do this, we prototyped a system for expressive control of animations, in the hopes that a real-time system based on the vector field principles will be useful for giving feedback on content, and shorten the animation authoring time. We explored the use of a Phantom force feedback device as a haptic user interface (HUI) scheme for a class of animations. Our first efforts have concentrated on using the Phantom as a control device to edit motions, to browse a family of animations, and to drive animations. The availability of 3-D force feedback differentiates the Phantom (and haptic devices in general) from other pointing, guiding, and motion capture systems, in that the animator may drive the animation while simultaneously receiving force feedback that encodes information about the state of the animation.

Unlike more commonly used sensory channels (video and audio) where input and output are decoupled, haptic force-feedback provides a unique opportunity for the computer and user to work in collaboration to author motions and trajectories which may then be interpreted as computer animations. In particular, the computer can use force-feedback to guide the user along certain paths, or away from "bad" regions of the control space.

There has been a great deal of work on virtual reality applications of haptic force feedback, in which, for example, a virtual character represented as a 3-D model can be felt or posed with force feedback. Our work can also be viewed as a way of feeling or browsing virtual objects—the main difference is that these virtual objects are (respectively) trajectories, bundles of trajectories, vector fields, dynamical systems, and other en-

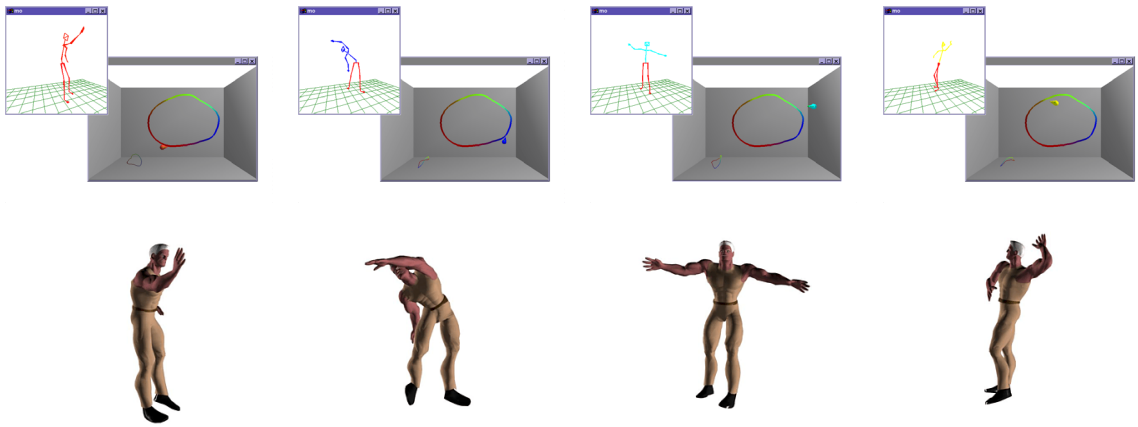

Figure 1: *Haptic control of a high-dimensional space constructed from 4 example motions: an angry, gesturing man, a spinning dancer, a Russian dancer, and capoeira. The* FOLLOW *phase and 3DStudio output of a 4-example animation are shown. The haptic map uses cylindrical coordinates* (r, θ, z) *both to control time/frame, and to interpolate between the 4 motion capture inputs. As in the red-blue example (Figs. 5, 6, 7), the angle* θ *controls the time/frame. Parameters* r *(radius) and* z *(depth) are used to interpolate between the 4 examples. The HOVF along the tube pulls the Phantom along the trajectory. The bottom of the "centaur" is the angry man, and the tops are, in chronological order: angry(derisive gesture):red(L,I), Capoeira(side stretch):blue(L,O), spinning(arms wide):cyan(H,I), and Russian(dramatic dance pose):yellow(H,O). In parentheses after the color are the performed deviations from the HOVF tube during* FOLLOW *to create the animation: L=low, H=high, I=inside, O=outside.*

tities that encode the visual variation of an animation over time and space. Since these objects are (a) often high dimensional, and (b) not as familiar as the solid 3-D objects surrounding us in everyday life, we have developed some new techniques for visualizing, browsing, and "feeling" them. Of particular interest may be methods for direct manipulation of trajectory bundles, which permit haptic editing of an animation.

In order to encode a family of animations in this manner, a number of representational problems must be solved. The mathematical and computational underpinnings of this work devolve to the theory of vector fields and dynamical systems, developed in robotics and control theory. However, their use in the context of animation authoring is novel and requires some extension. Of particular utility is the concept of higher-order vector fields, which we exploit in our representational and control framework.

2 How Can Haptic Vector Fields Control Animations?

2.1 Basic Concept

To illustrate our approach, consider the following example. A configuration space D is established, such that a point in D represents one "frame" of the animation.[1] For example, in this paper, we take D to represent the set of possible joint angles [13] for an articulated figure. A haptic control map is established so that the Phantom's degrees of freedom control the animation. This is done by constructing a mapping $h : C \longrightarrow D$ where C is the haptic control space representing the six input degrees of freedom of the Phantom (in our case, $C = SE(3)$, the Special Euclidean group of rigid body motions in 3D). We take as input a smooth trajectory[2] $\varphi_1 : I \longrightarrow C$. Here φ_1 represents an entire animation "clip," because the mapping $h \circ \varphi_1$ defines an animation "frame" for each point t in I. Note that φ_1 trivially defines a vector field along its image $\varphi_1(I)$, namely the field of tangent velocity vectors $(\varphi_1(t), \dot{\varphi}_1(t))$; see Fig. 2-L.

We define a small tube of radius ε about the image of φ_1, and extend the vector field to this tube in the following manner. The field will have a radial and a tangential component. The radial component Y_1 will point towards the center of the tube, where $\varphi_1(I)$ lies (Fig. 2-R). The tangential component X_1 near $\varphi_1(t)$ will lie parallel to $\dot{\varphi}_1(t)$. Both components decrease in magnitude with the distance from the tube center. The sum $V_1 = X_1 + Y_1$ of the radial and tangential components defines a dynamical system on C that may be viewed as a "river," pulling configurations into and along a central attracting flow defined by the animation. This vector

[1] In practice, the time domain will be discretized or sampled. We follow [5] in our terminology for sampling: "An animation system should have a sampling rate that is decoupled from the nominal 'frame rate' of the final product. We will speak of 'frames' at the sample rate without intending any loss of generality."

[2] Here I represents time, parameterized to the unit interval $[0, 1]$. In general, of course, animations could take different amounts of time.

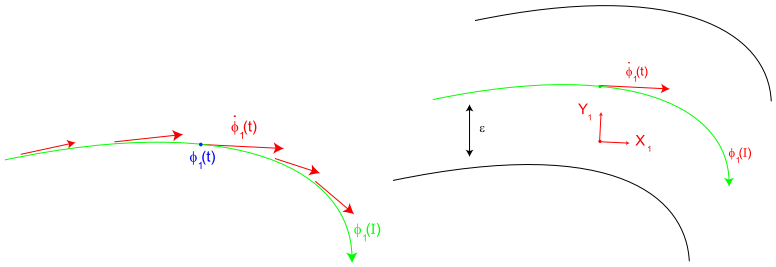

Figure 2: *(L) The trajectory φ_1 induces a vector field along its image. (R) An ε-tube about the image $\varphi_1(I)$ of the trajectory φ_1 is shown, with the tangential and radial fields X_1 and Y_1. X_1 is parallel to $\dot{\varphi}_1$.*

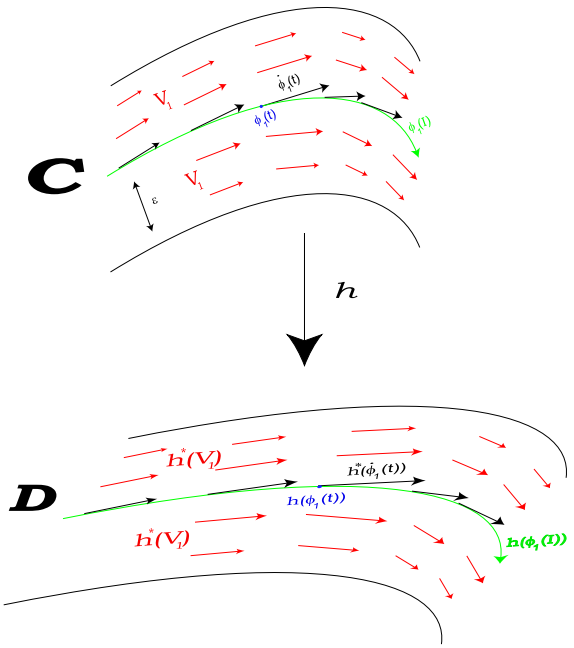

Figure 3: *The haptic vector field V_1 is defined on haptic control space C. The haptic map h maps from C to the animation space D.*

field not only defines the flow of the animation, but a force function, parameterized by the position in C of the Phantom; this field may be experienced by the user as haptic forces. Finally, when h is injective, the vector field on C may be "pushed forward" using h^*, the derivative (or *Jacobian*) of h, to the configuration space D. See Fig. 3.

Now, the vector field in the ε-tube about $\varphi_1(I)$ defines a dynamical system on the haptic control space C, linked via the haptic control map h to the animation configuration space D. To play back an animation, the Phantom is positioned in space, and travels along with the vector field. More specifically: a point z in C represents a configuration of the Phantom (that is, when we say "put the Phantom at $z \in C$," we mean place the Phantom's manipulandum in pose z). As above, we denote the vector field (induced by φ_1) on C by V_1, so that for a configuration z in C, $V_1(z)$ represents the vector force at z. As described above, when the user places the Phantom at z, she experiences the force $V_1(z)$ through haptic force feedback. However, this same force causes the Phantom to move, through a physical dynamics equation (that is, we command force $F = V_1(z)$, and, by Newton's equation $F = MA$, the trajectory of the Phantom then evolves over time through integration).[3] Hence, in the absence of a user-supplied force, the Phantom's manipulandum will first converge to, and then traverse the trajectory φ_1 autonomously. Mathematically, the resulting trajectory is obtained by ordinary integration of the vector field from a starting configuration. During this traversal, the haptic control map h defines an animation "frame" for every configuration in the resulting trajectory; sequential display of these frames results in an animation. Hence as the Phantom moves in the vector field, an animation plays (Fig. 1).

During playback, interactive modification of the Phantom's position results in a run-time modification of the "river" V_1, and hence in a new animation. Perturbation of the manipulandum during playback results in a displacement in C, which, when "played" through the haptic control map h, results in a slightly different animation. By this means, the user can interactively modify the animation by tugging and pushing the Phantom slightly off its course. For cyclic animations (e.g. walking, running, hopping), time is viewed as circular (parameterized by the unit circle $\mathbb{S}^1$) and cyclic animations are represented by mappings $\mathbb{S}^1 \longrightarrow C$ In this case, the Phantom autonomously follows a limit cycle in C, thereby driving the cyclic animation. Perturbation of this limit cycle results in an edited or morphed animation; in the absence of perturbation the vector field will converge to and restore the underlying anima-

[3] In fact the dynamics is considerably more complicated: for example there is damping.

tion cycle. During playback, the user-supplied forces define another vector field, U. During interactive modification, the new family of animations can be represented by the sum of V_1 and the user-supplied force field U. We can record the combined vector field $U + V_1$ as a stored representation for the new animation system. See Fig. 4.

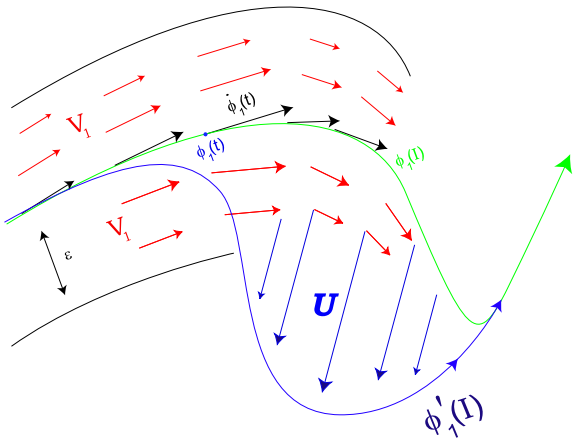

Figure 4: *A sample animation is encoded as a trajectory φ_1, which induces a vector field V_1 about its image in C (see Figs. 2–3). During playback in* FOLLOW *mode, the Phantom's manipulandum by default follows the flow of V_1, therefore tracing out φ_1. Here, the user alters the trajectory by exerting physical forces (the force field U) on the Phantom. This results in an edited trajectory φ_1', and an edited dynamical system $V_1 + U$. φ_1' represents a new path for the Phantom; given a haptic map $h : C \longrightarrow D$, $h \circ \varphi_1'$ encodes the edited animation.*

We regard the input trajectory φ_1 as an example animation that defines a dynamical system. This scheme becomes more interesting when we have multiple trajectories $\varphi_1, \varphi_2, \varphi_3, \ldots$, each defining an animation (of the same character), and we wish to combine these examples into a parametric family of animations, encoded as a dynamical system. One way to do this would be to construct a vector field V_i for each trajectory φ_i, and then to employ a dynamical system defined by their sum $\mathcal{V} = \sum_i V_i$. Thus, in principle, $\mathcal{V}$ defines a new flow, representing a family of animations that can be browsed and edited using haptic force feedback. In our experience, this works reasonably well when the trajectories are disjoint in C. However, when their images (tubes) intersect, the behavior of the system can be unsatisfactory. For this reason, we have explored more sophisticated ways of combining example trajectories to define a dynamical system (Sec. 5). Our method may be viewed as a simple example-based technique for defining dynamical systems. The key modeling insight to defining the combination operator is to employ time-varying, higher-order vector fields. Once the combination operator is correctly defined, the user can resist, modify the flow, etc., thereby changing the animation, yielding real-time authoring of new motions. As before, the forces the user exerts are recorded and encoded as a user field U; the modified family of animations is $\mathcal{V}+U$.

We believe our work shows how force feedback in animation authoring is a useful enabling technology. We are also trying to explore the capabilities and limits of our approach. For example, once we represent a family of animations by a dynamical system, a number of different parameterizations are possible. We have experimented with a few different techniques for direct manipulation of such systems, using haptic browsing and force fields. For example, suppose we are given a set of trajectories $\varphi_1, \varphi_2, \ldots$ defining example animations. It is possible to build virtual tubes around the images of these trajectories in haptic control space, and to directly manipulate the tubes. This may be done by constructing the Minkowski sum of a small ε-ball in $\mathbb{R}^3$ with the projection (into $\mathbb{R}^3$) of the image of a trajectory. These tubes may be treated as a set of springy fibers in a virtual 3-D space. We can manifest these tubes both visually and haptically as virtual objects. The Phantom can then be used to push, pull, or manipulate a folded trajectory, and thereby change the animation. During the direct manipulation, the tube haptically appears rubbery and resistant to motion ("stretchy"). See Fig. 6. For example, the manipulandum can virtually approach a trajectory tube, grab it, stretch it, and move it to a new position. Simultaneously, the user views the corresponding animation playing, while the point $\varphi_i(t)$ in configuration space (representing the animation) is seen to move along the virtual tube. Deformation of the tube changes the trajectory from φ_i to φ_i' and therefore the animation changes from $h \circ \varphi_i$ to $h \circ \varphi_i'$.

2.2 Examples

So far, we have defined two paradigms for animation control using a haptic device. Both rely on the construction of *a priori* sample trajectories, which are either authored in advance (when creating the haptic map h), on the fly by the end user, or are implicit in the haptic map itself. The sample trajectory dictates a sample animation, and creative control over the animation devolves from alteration to or variation from the sample trajectory.

In the first paradigm, which we call FOLLOW, the manipulandum follows the sample trajectory unless deflected by forces exerted by the user. These deflection forces represent perturbations to the sample trajectory, and allow expressive control of the resulting anima-

tion. The perturbations combine with the "default" HOVF (induced by the sample trajectory, as described in Sec. 2.1) to form a new dynamical system on the fly, which represents a performance of the animation. In the second paradigm, which we call STRETCHY TUBES, the manipulandum is always constrained to lie on a sample trajectory, but the trajectory may be dynamically modified using the haptic "stretchy tubes" effect described in Sec. 2.1.

In fact, these modes may be applied in sequence. First, the sample trajectory is altered (edited) using the STRETCHY TUBES paradigm, during which time we see the evolving animation dictated by a point traveling along the trajectory. Second, for finer-grained control, the user (in the FOLLOW paradigm) performs an inexact and one hopes inspired traversal of the trajectory to create a fresh new animation which is only loosely based on the given sample. If there is more than one sample trajectory the rules for traversal and interpretation are more complex, but the basic idea is the same.

Figs. 5, 6, and 7 and the video illustrate this methodology, forming an extended example. While this example is based on interpolating motion capture data, it is applicable to any parametric animation; in addition, multi-target interpolation is possible after defining an appropriate haptic map. In Fig. 1, a haptic control space with cylindrical coordinates was defined, by orthogonally extruding the red-blue haptic map in Fig. 6. Using our haptic paradigms, we developed novel animations by conducting a 4-example STRETCHY TUBES editing phase and a FOLLOW performance phase. The resulting animations can then be rendered using 3DStudio; see Fig. 1 and the video.

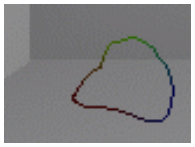

Figure 8: *The 4-colored control tube in Fig. 1 is a twisted loop embedded in 3D, and was haptically authored in a* STRETCHY TUBES *phase (not shown).*

3 Previous Work

Few techniques use haptics to browse and edit the dynamical system of an animation through direct manipulation. The encoding and editing of such systems as palpable vector fields appears to be novel. Previous research falls into a few broad categories. Fundamental work in haptics and force-feedback [15, 4, 6, 20] has allowed devices such as the Phantom to be integrated with computer graphics. Most of this work is targeted for scientific visualization, or for the combined visual-haptic display of complex virtual-reality environments. The control systems and abstractions in this work have been important in building our haptic

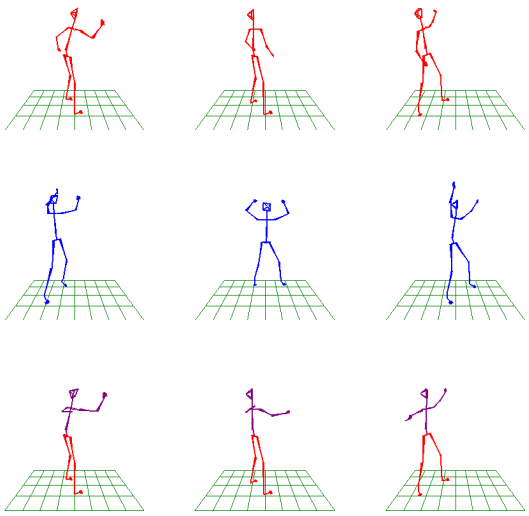

Figure 5: *As input to our system, we take several motion capture files. Here, one is red, and depicts an angry figure walking about and making a few derisive gestures with one arm. The second is blue, and depicts a figure jumping around and waving both arms happily. For convenience, we use a red/blue color gradient to illustrate which animation or interpolant we're using. Next, we define a* centaur *which interpolates between the two as follows: the centaur had a red bottom; the top of the centaur depends on an interpolation parameter r between 0 (red) and (1) blue. For $r \in (0, 1)$, the top is defined by interpolating the joint angles between the red and blue examples. Thus, $r = 0.5$ represents a centaur with a red bottom, and whose top is a purple intermediate, interpolated halfway between the red and blue tops. This input data is then modified in Fig. 6.*

system. Vector fields have been widely used in robot control [11, 12, 17, 16, 3], and these mathematical foundations were influential in our system design. Nonholonomic control and HOVF's were developed in the context of control for non-linear geometric dynamics, and have a wide range of applications [1, 13, 2, 9, 14]. There have been a number of elegant papers on processing motion data [5, 21] multi-target motion interpolation [18], real-time control of virtual humans [10], retargeting of motion [7], motion transitions [19], and constraint-based motion adaptation [8]. Inspired by this work, we employ very simple forms of interpolation and motion processing in order to demonstrate the power of haptic vector fields for animation motion control. We believe that in the future, sophisticated motion processing, interpolation, and retargeting algorithms will be integrated with haptics for direct manipulation of trajectory bundles, and for haptic browsing of an an-

Figure 7: *The animation edited and performed in Fig. 6 is rendered using 3DStudio.*

imation's dynamical systems using vector force fields. Our paper represents a first step towards realizing that goal.

4 Materials and Methods

A key element of our system is the PHANToM (Personal haptic interface mechanism) from SensAble Technologies[4]. It is currently hooked up to a dual Pentium II workstation running Windows NT but may alternatively be used with a Silicon Graphics O2 running Irix. We wrote a Phantom driver for 3D Studio MAX, (a commercial animation authoring package from Kinetix[5]) which runs on the NT workstation. We use 3DSMAX as an animation back end, and in particular we use the Character Studio plug-in for importing and animating motion capture files. The code we have written falls into two categories: (1) a library for browsing and editing 3-D trajectories using the Phantom, and (2) an application for viewing and editing motion capture files using the Phantom plus a GUI. Both are written in C^{++} using OpenGL for all graphics, and should run under either Windows NT or Irix; so far the libraries have only been tested under NT.

5 Modeling and Algorithms

5.1 Crossing Rivers of Force

Define a *river* R_1 in C to be an ε-tube about a trajectory $\varphi_1(I)$, together with a vector field V_1 defined on the tube, as described above. When two rivers R_1 and R_2 intersect, a "combined" vector field must be defined.

For intuition, let us ignore the radial forces Y_i until Sec. 5.2. Suppose the user is guiding the Phantom along some trajectory γ in C, and at time t, the point $z = \gamma(t)$ lies within the intersection of R_1 and R_2. In this case, we propose that the force experienced in FOLLOW mode by the user (through the Phantom force feedback) should depend on the direction of motion (i.e. on the velocity $\dot\gamma(t)$. That is, if $\dot\gamma(t)$ is parallel to $V_1(z)$, then the force should be $V_1(z)$. On the other hand, if $\dot\gamma(t)$ is parallel to $V_2(z)$, then the force should be $V_2(z)$. See Fig. 9. Speed as well as direction are used for selection, because speed may be a useful criterion for example in enforcing ballistic constraints. For example, to initiate a ballistic backflip, one must move fast enough. To make a sharp turn one must move slowly enough. Defining a force field as a function of both position and velocity results in an interesting control system, called a *Higher-Order Vector Field (HOVF).*

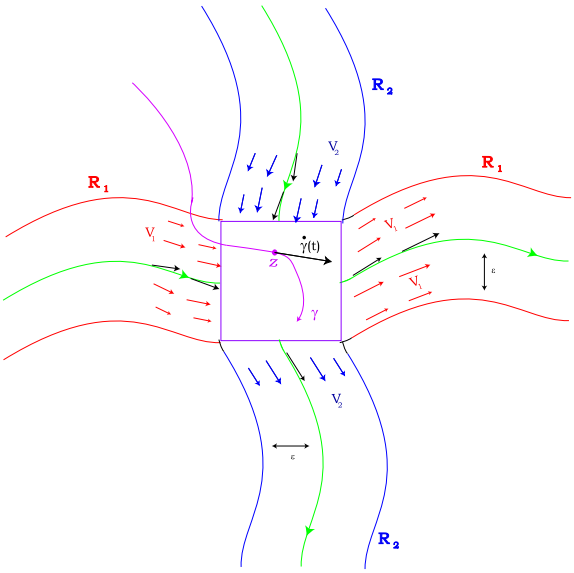

Figure 9: *Two rivers R_1 and R_2 cross in the purple square. Suppose the user is guiding the Phantom along some trajectory γ in C, and at time t, the point $z = \gamma(t)$ lies within the square. The force experienced by the user should depend on the direction of motion (i.e. on the velocity $\dot\gamma(t)$). That is, if $\dot\gamma(t)$ is parallel to $V_1(z)$, then the force should be $V_1(z)$. On the other hand, if $\dot\gamma(t)$ is parallel to $V_2(z)$, then the force should be $V_2(z)$.*

5.2 Higher-Order Vector Fields

A HOVF is like a standard vector field, in that it defines a *constraint* that the integral curves must follow. HOVF's are related to nonholonomic constraints.[6]

[4] http://www.sensable.com/
[5] http://www.ktx.com/

[6] In mechanics, systems may be *holonomic* or *non-holonomic*. In general, a *holonomic constraint* is a wholly integrable sub-bundle E of the tangent bundle. The system outcome for a nonholonomic system is path-dependent.Non-holonomic systems have been studied in robotics [1, 13, 2, 9, 14]. Examples include: Car-like robots, tractor-trailers, bicycles, roller-blades, airplanes, submarines, satellites, and spherical fingertips rolling on a manipulandum. In robotics, a non-holonomic system is usually defined by a series of non-integrable constraints of the form $\Xi_i(p, v) = 0$ on the tangent bundle. For example, whereas holonomic kinematics can be expressed in terms of algebraic equations which constrain the internal, rotational coordinates of a robot to the absolute position/orientation of the body of interest, nonholonomic kinematics are expressible with differential relationships only. This distinction has important implications for the implementation of a control system.

A HOVF is a map $F : TC \longrightarrow TC$, with $F(p, v) = (p, f_p(v))$, where TC is the tangent bundle (phase space) of C, and (p, v) is a tangent vector (position and velocity). Observe that since C is a manifold, so is the tangent bundle TC. Then F is a vector field on the manifold $M = TC$, with values in $TM = T^2C$. Now, we wish to construct F in a well-defined manner on the intersection of the two rivers R_1 and R_2. As in Sec. 2.1, we decompose an induced vector field V_i into its tangential and radial components X_i and Y_i, respectively, so that $V_i = X_i + Y_i$. X_i and Y_i are also vector fields. To formalize the construction in Sec. 5.1, we define $f_p(v)$ to be $V_1(p)$ when $v \approx X_1(p)$, $V_2(p)$ when $v \approx X_2(p)$, and 0 otherwise. This construction is "discrete"; we have also experimented with a smooth version.

5.3 Time-Varying Higher-Order Vector Fields

All the vector fields we have seen so far are *static*, in that they do not change over time. The most effective HOVFs for haptic manipulation of animations are often time-varying HOVFs.

Time-varying HOVF's have the form[7] $L : TC \times I \longrightarrow TC$, with $L(p, v, t) = (p, f_p(v, t))$. A useful time-varying HOVF may be defined as follows. Consider river R_1 again. Given an example trajectory φ_1 and a time t, a position of the Phantom (called the *configuration point*) is given by $\varphi_1(t)$, and the corresponding frame of the animation is $h(\varphi_1(t))$. In FOLLOW mode, as time evolves, the user will see the animation change, and feel (and see) the configuration point change, through force feedback. We implement this force feedback by placing a virtual "bunny" at the moving configuration point; the bunny exerts an attractive force on the manipulandum, which follows it along the "track" of $\varphi_1(I)$ like a "greyhound." The attractive force is centered on the (moving) bunny, and decreases smoothly to zero with distance. This results in a complex behavior, which can be succinctly modeled as a time-varying HOVF. Let F represent a (static) HOVF induced by river R_1, as described in Section 5.2. To implement the bunny model, we define a time-varying HOVF L as follows: $L(p, v, t) = F(p, v) * G_\delta(\varphi_1(t))$, where $G_\delta(\varphi_1(t))$ is a multi-dimensional Gaussian of width δ about $\varphi_1(t)$, and "$*$" denotes convolution.[8] This HOVF paradigm was employed in all the FOLLOW examples in this paper.

[7] As before, the time domain can also be cyclic, represented by substituting $\mathbb{S}^1$ for I.

[8] Convolution of a vector function with a scalar function is performed component-wise, yielding a new vector function.

5.4 Inertia and Viscosity

The force feedback effects of inertia and viscosity can be modeled using a HOVF. If the user is executing a Phantom trajectory γ, viscosity may be modeled using the HOVF $F(p, v) = (p, -\mu(p, v)v)$. Such effects can be used in the following animation experiment. Initially, a circular example trajectory is defined, which controls a cyclic animation (e.g. a walking character). Deviations from the circle perturb the animation (through h) to control expressiveness. For example, a larger or smaller radius can control the height of the steps (shuffling vs. skipping) and the depth (orthogonal to the plane of the circle) can control the mood of the animation (happy vs. sad). In the absence of perturbation, the Phantom converges to and follows the circular limit cycle, and the animation cycles through its default course.

Now, a radial inertia and viscosity field is defined as an HOVF, enabling the Phantom—hence the animation—to habituate to a new "orbit" at a different altitude and depth. The haptic effect is as if one were digging a groove in a viscous 3D medium. When the Phantom learns the groove—corresponding to a new limit cycle—then the user releases the Phantom. The Phantom remains orbiting in the newly defined groove, thereby driving a new default animation. We have used this technique to author animations under the 4-example cylindrical-coordinates interpolating haptic map in Fig. 1.

6 Conclusions

Our system could be viewed as a learning environment based exclusively on positive reinforcement: all haptic forces are attractive. Using repulsive forces would allow haptics to author and enforce constraints in C and D. These constraints could force the user away from undesirable poses, motions, or transitions.

A series of interesting problems arise in using haptics to author the haptic map $h : C \longrightarrow D$. Suppose we are given a series of motion capture files corresponding to different behaviors or motions of the same character. For example, these files could represent different walks, runs, or dance moves. Each file encodes a trajectory $\alpha_i : I \longrightarrow D$, for $i = 1, 2, 3, \ldots$ The problem we confront is how to author a *set* of maps $\widetilde{\alpha_1}, \widetilde{\alpha_2}, \ldots$, together with a *single* haptic map $h : C \longrightarrow D$, such that the following diagram commutes for every α_i:

$$\begin{array}{ccc} C & \xrightarrow{h} & D \\ {\scriptstyle \widetilde{\alpha}_i}\uparrow & \nearrow_{\alpha_i} & \\ I & & \end{array} \qquad (1)$$

This is called a *lifting* problem, since α_i is "lifted" up to C. In this paper we have solved the lifting problem

by constructing the map h "by hand." This permitted automatic construction of haptic force vector fields induced by the examples. Using these fields, a haptic device can browse and edit a family of animations. This allows us to directly mediate and interpolate between the motion capture examples using the haptic force vector fields in a dynamical system representing the entire family of animations. A fruitful direction for future research is an automated solution of the lifting problem in Eq. (1).

References

[1] J. Baillieul and R. Brockett. *Robotics*, volume 41 of *Symposia in Applied Mathematics*. American Mathematical Society Press, Providence, RI, 1990.

[2] J. Barraquand and J.-C. Latombe. Nonholonomic multibody mobile robots: Controllability and motion planning the the presence of obstacles. *Algorithmica*, 10(2):121–155, August 1993. Special Issue on Computational Robotics.

[3] K.-F. Böhringer, B. R. Donald, and N. C. MacDonald. Upper and lower bounds for programmable vector fields with applications to MEMS and vibratory plate parts feeders. In J.-P. Laumond and M. Overmars, editors, *Algorithms for Robotic Motion and Manipulation*, pages 255–276. A. K. Peters, Wellesley, MA 02181, 1997. .

[4] F. P. Brooks, Jr., M. Ouh-Young, J. J. Batter, and P. J. Kilpatrick. Project GROPE — haptic displays for scientific visualization. *Computer Graphics*, 24(4):177–185, Aug. 1990.

[5] A. Bruderlin and L. Williams. Motion signal processing. *Computer Graphics*, 29(Annual Conference Series):97–104, 1995.

[6] T. A. Galyean and J. F. Hughes. Sculpting: An interactive volumetric modeling technique. *Computer Graphics*, 25(4):267–274, July 1991.

[7] M. Gleicher. Retargeting motion to new characters. *Proc. SIGGRAPH (Computer Graphics)*, Aug. 1998.

[8] M. Gleicher and P. Litwinowicz. Constraint-based motion adaptation. *Jour. Visualization and Comp. Graphics*, 9:65–94, 1998.

[9] K. Goldberg, J.C.-Latombe, D. Halperin, and R. Wilson, editors. *The Algorithmic Foundations of Robotics: First Workshop*. A. K. Peters, Boston, MA, 1995.

[10] P. Kalra, N. Magnenat-Thalmann, L. Moccozet, G. Sannier, A. Amaury, and D. Thalmann. Real-time animation of realistic virtual humans. *IEEE Computer Graphics and Applications*, pages 42–56, Sept. 1998.

[11] O. Khatib. Real time obstacle avoidance for manipulators and mobile robots. *Int. Journal of Robotics Research*, 5(1):90–99, Spring 1986.

[12] D. E. Koditschek and E. Rimon. Robot navigation functions on manifolds with boundary. *Advances in Applied Mathematics*, 1988.

[13] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Press, 1991.

[14] J.-C. Latombe. Robot algorithms. In T. Kanade and R. Paul, editors, *Robotics Research: The Sixth International Symposium*, 1993.

[15] M. Minsky, M. Ouh-young, O. Steele, F. P. Brooks, Jr., and M. Behensky. Feeling and seeing: Issues in force display. *Computer Graphics*, 24(2):235–243, Mar. 1990.

[16] J. Reif and H. Wang. Social potential fields: A distributed behavioral control for autonoomous robots. In K. Goldberg, D. Halperin, J.-C. Latombe, and R. Wilson, editors, *International Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages 431–459. A. K. Peters, Wellesley, MA, 1995.

[17] E. Rimon and D. Koditschek. Exact robot navigation using artificial potential functions. *IEEE Transactions on Robotics and Automation*, 8(5), October 1992.

[18] C. Rose, M. Cohen, and B. Bodenheimer. Verbs and adverbs: Multidimensional motion interpolation. *IEEE Computer Graphics and Applications*, 18(5):32–30, Sept. 1998.

[19] C. F. Rose, B. Guenter, B. Bodenheimer, Cohen, and M. F. Efficient generation of motion transitions using spacetime constraints. *Computer Graphics*, 30(Annual Conference Series):147–154, 1996.

[20] D. C. Ruspini, K. Kolarov, and O. Khatib. The haptic display of complex graphical environments. *Computer Graphics*, 31(3A):345–352, Aug. 1997.

[21] A. Witkin and Z. Popovi'c. Motion warping. *Computer Graphics*, 29(Annual Conference Series):105–108, 1995.

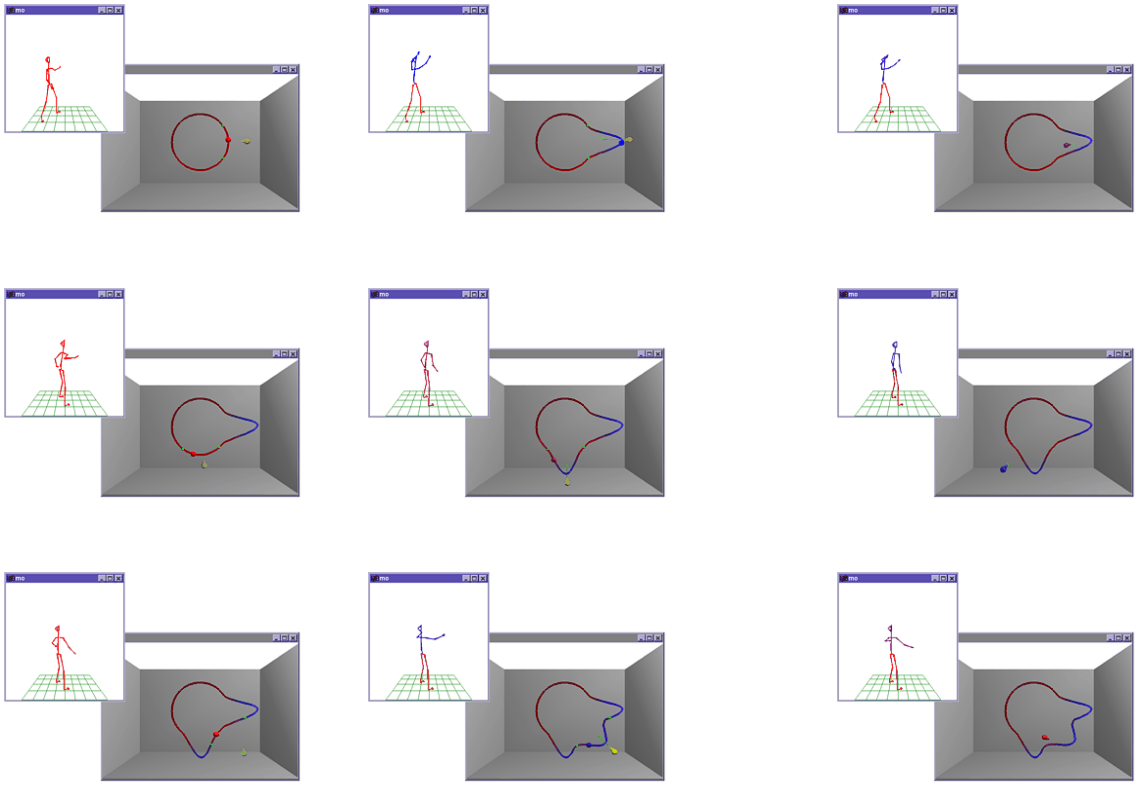

Figure 6: *The* Stretchy Tubes *and* Follow *paradigms. The animation is in approximately the same frame throughout each row. Note the green arrow indicating the force vector from the Phantom cursor in the stretch and follow pictures.* **Left:** *before stretch.* **Center:** *after stretch.* **Right:** Follow *phase using result of stretch. The Left and Center frames show* Stretchy Tubes *phase of editing the trajectory, using the inputs in Fig. 5. A new window is added to the display, in which is shown a circular red planar trajectory in 3D, which a red ball traverses. The ball drives the animation—its position on the trajectory determines the time frame, and the distance from the center determines parameter r. Thus the haptic control space has a radial gradient from red (near the center), through purple, to blue (at the periphery). Initially, the circle is small, and red. The user controlling the Phantom (whose cursor is seen as a pointy yellow bulb) stretches the trajectory out into the blue region, and the ball follows the new path, turning purple then blue when entering the outer regions. As it does this, the top half of the figure also turns purple then blue, and resembles the top half of the blue figure as it does so. In this manner the sample trajectory is modified and stretched as the user desires. During this process, the user (through the Phantom) experiences haptic forces: our philosophy is to use the haptic forces to encode as much information as possible about the animation's dynamical system. The force field for* Stretchy Tubes *is particularly simple. When the user is positioning the Phantom, she feels an attractive force towards the trajectory. This force field guides the user to a position in which the tube can be easily grabbed. To grab the trajectory, the user presses a button on the the manipulandum. At that point, the user begins to experience a springy-stretchy restoring force when tugging on the tube. This force operates under a simple spring control law. The haptic forces are illustrated by a green arrow. The force fields for* Stretchy Tubes *are static vector fields that do not depend on velocity or time. The* **Right** *frames show the* Follow *phase. We next see a different haptic window. It contains the trajectory we just created, but the ball is gone and haptic cursor (the pointy bulb) is no longer yellow but takes on the color of the region it inhabits—for it is now the cursor which determines the state of the animation, subject to the same haptic map as before. Unlike the ball in the prior* Stretchy Tubes *example, the cursor is not restricted to the trajectory, and is under the direct control of the user. It may speed up, slow down, cut corners, or wander farther afield. Left to its own devices (i.e. in the absence of user-supplied forces) it follows the given trajectory. The default* Follow *behavior of the dynamical system is implemented using the HOVF induced by the trajectory we created in the editing (*Stretchy Tubes*) phase above. Occasionally (as in* Stretchy Tubes*) you can see a green arrow protruding from the tip of the bulb. This is a visual representation of the actual force that the user feels when holding the haptic device. During the* Follow *phase, the user experiences a HOVF force field induced by the trajectory, as described in Sec. 2.1.*