Proceedings of the 2000 IEEE
International Conference on Robotics & Automation
San Francisco, CA • April 2000

Using Haptic Vector Fields for Animation Motion Control*

Bruce Randall Donald† and Frederick Henle‡
Dartmouth Computer Science Department
Hanover, NH 03755, U.S.A.

Abstract

We are developing paradigms and algorithms for browsing and editing families of animations using a haptic force-feedback device called a Phantom. These techniques may be generalized to navigation of any high degree-of-freedom system from a lower degree-of-freedom control space, with applications to telerobotics and simulation of virtual humans. We believe that modeling the animation configuration space coupled with the highly interactive nature of the haptic device provides us with useful and intuitive means of control.

We have implemented our ideas in a system for the manipulation of animation motion capture data; in particular, anthropomorphic figures with 57 degrees of freedom are controlled by the user in real time. We treat trajectories, which encode animation, as first-class objects; haptic manipulation of these trajectories results in change to the animation. We have several haptic editing modes in which these trajectories are either haptically deformed or performed by the user with expressive control subject to dynamic haptic constraints. The initial trajectories are given by sample animations (for example, motion capture data) but may be authored by other means.

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 parametric 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.

Unlike the 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 trajectories, 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 entities 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.

*This research is supported by the following grants to B.R.D. from the National Science Foundation: NSF IIS-9906790, NSF EIA-9901407, NSF 9802068, NSF CDA-9726389, NSF EIA-9818299, NSF CISE/CDA-9805548, NSF IRI-9896020, NSF IRI-9530785, and by an equipment grant from Microsoft Research.

†brd@cs.dartmouth.edu

‡henle@cs.dartmouth.edu

2 How Can Haptic Vector Fields Control Animations?

2.1 Overview

The state of an animation is encoded as a point in its configuration space. A continuous animation or animation segment is encoded as a continuous trajectory in the configuration space. Since the animation and the trajectory are equivalent, we may alter the trajectory and derive a new animation from the altered trajectory. However, it is difficult to work in such a high-dimensional configuration space directly, so we provide a mapping from a lower-dimensional control space to the configuration space, and manipulate trajectories in the control space.

The control space is defined by the degrees of freedom of our haptic device, the Phantom. The user interacts with the Phantom by manipulating a pen-like appendage (called a "stylus"). It has six degrees of freedom, three for the position of the tip of the stylus and three for its orientation. There is also a switch on the stylus which may be used like a mouse button, e.g. to click and drag.

Thus, a trajectory in the control space is represented visually (in a two-dimensional projection on the computer monitor) and haptically (through the Phantom) as a continuous path in three dimensions. We have provided several techniques for editing existing trajectories, and as this is done the user can see the effect on the animation in real time.

The construction of the configuration space, the control space, the mapping between them (and the haptic forces) makes it possible to author and edit animations by manipulating trajectories in the control space. We will discuss the haptics in Section 2.3, but first we give the mathematical model.

2.2 Mathematical Model

We call the configuration space D, with a point in D representing one "frame" of the animation. For example, in this paper we take D to represent the set of possible joint angles [13, 15] for an articulated human figure. A 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 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 3-D). We take as input a smooth trajectory[1] $\varphi_1 : I \longrightarrow C$. 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. 1.

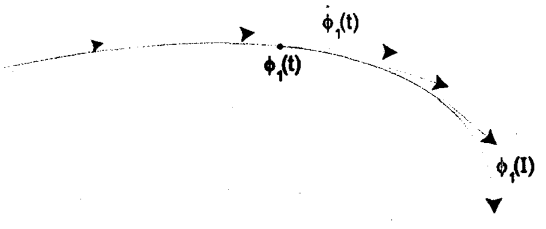

Figure 1: The trajectory φ_1 induces a vector field along its image.

2.3 Vector Fields for Haptics

Follow mode: We have developed several haptic modes. Let us begin by describing one of the simplest ones. We take as input a default trajectory, corresponding to a default animation. We ask the user to perform a similar trajectory by tracing the default one, and we use a haptic vector field to make this task easier—in fact, when the user releases the Phantom or merely relaxes her grip it will trace the default trajectory autonomously. Thus, any deviations from the default trajectory are the result of an expressive act of will on the part of the user, and not simply an inability to trace with a steady hand. In order to accomplish this, we need to embed a vector field in the control space and express the field as Phantom forces.

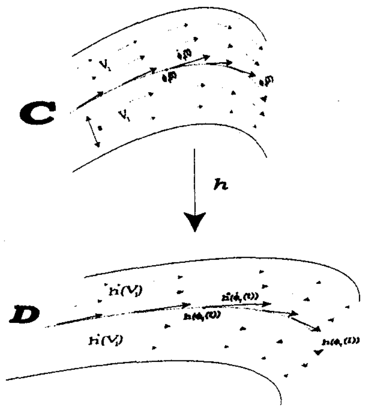

Figure 2: The haptic vector field V_1 is defined on haptic control space C. The haptic map h maps from C to the animation space D.

[1]Here I represents time, parameterized to the unit interval $[0,1]$. In general, of course, animations could take different amounts of time. 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 $\mathbb{S}^1 \longrightarrow C$.

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 has a radial and a tangential component. The radial component Y_1 points towards the center of the tube, where $\varphi_1(I)$ lies. The tangential component X_1 near $\varphi_1(t)$ lies parallel to $\dot{\varphi}_1(t)$. Both components decrease in magnitude with distance from the tube center. In fact, the radial component also vanishes at the trajectory, so that we avoid a force discontinuity. 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 field defines not only the flow of the animation, but also a force function, parameterized by the position in C of the Phantom; this field is 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. 2.

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 allowed to travel along with the vector field. 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. 3).

Stretchy Tubes mode: 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.

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 $h \circ \varphi'_i$.

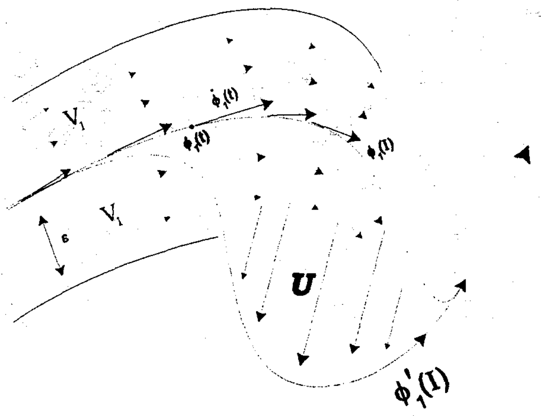

Figure 4: A sample animation is encoded as a trajectory φ_1, which induces a vector field V_1 about its image in C (see Figs. 1–2). During playback in FOLLOW mode, the Phantom's manipulandum by default follows the flow of V_1, therefore tracing out φ_1. Here, the user alters the trajectory by exerting physical forces (the force field U) on the Phantom. This results in an edited trajectory φ'_1, which is an integral curve of the 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.

3 Examples

So far we have described 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 animation. 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

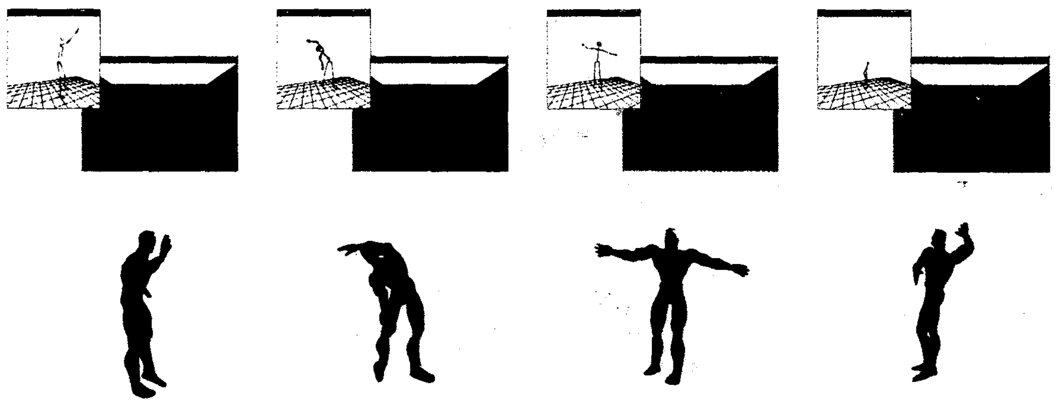

Figure 3: Haptic control of a high-dimensional space constructed from four example motions: an angry gesture, a spinning dance, a Russian dance, and Capoeira. The FOLLOW phase and 3DStudio output of a four-example animation are shown. The haptic map uses cylindrical coordinates (r, θ, z) to specify time (which frame of the animation) and to interpolate between the four 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 four examples. The HOVF pulls the Phantom along the trajectory. To see detailed figures and animations for this paper, visit `http://www.cs.dartmouth.edu/~henle/ICRA2000/`

call STRETCHY TUBES, the animation is defined solely by the trajectory, but that 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 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. 3, 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 four-example STRETCHY TUBES editing phase and a FOLLOW performance phase. The resulting animations can then be rendered using 3DStudio; see Fig. 3.

4 Haptic Control

The forces commanded in STRETCHY TUBES mode are simple. Initially, the Phantom forces help the user to find

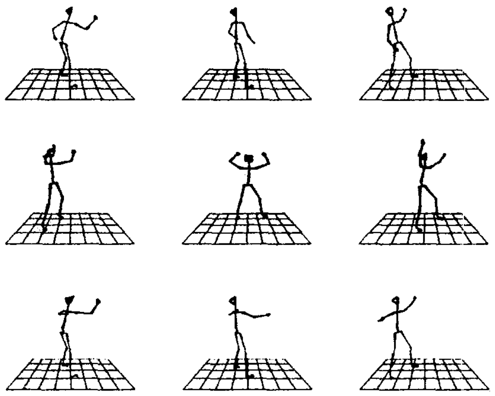

Figure 5: Our system takes as input several motion capture files. Here one is red, and depicts an angry figure making a few derisive one-armed gestures. The second is blue, and depicts a figure cheering with both arms. Next, we define a composite figure whose lower half is taken entirely from the red example but whose upper half ranges between red and blue examples according to the parameter of interpolation.

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

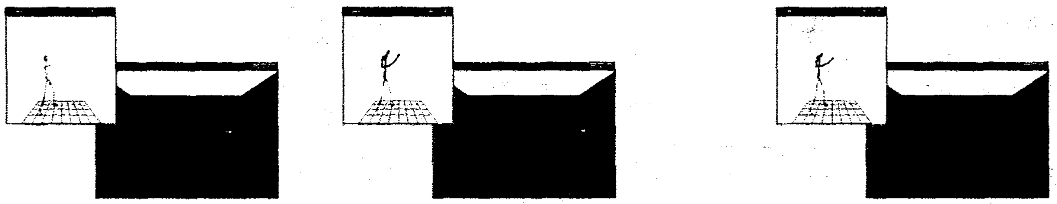

Figure 6: The STRETCHY TUBES and FOLLOW paradigms. The animation is in approximately the same frame in each screenshot. Note the green arrow indicating the force vector from the Phantom cursor in the stretch and follow pictures. **Left:** before stretch. **Center:** after stretch. **Right:** FOLLOW phase using result of stretch. The Left and Center frames show STRETCHY TUBES phase of editing the trajectory, using the inputs in Fig. 5.

the trajectory by providing a mild spring force, given by $k_f(c(p) - p)$ where k_f is the spring constant, p is the current Phantom position, and $c(p)$ is the point on the trajectory closest to p. Once the trajectory has been found and as it is being deformed, the restoring force is computed as a somewhat stronger spring force, given by $k_s(p_0 - p)$, where k_s is the spring constant, and p_0 is the position of the Phantom when the trajectory was grabbed, i.e. before it was deformed.

The FOLLOW mode forces are more complex. We stated earlier that there is both a tangential component and a radial component to the vector field lying in an ε-tube about the trajectory. Note that if we only define forces within the ε-tube, then the Phantom is inert and unhelpful everywhere outside it. So, we must make ε quite large, effectively ensuring that every point in the control space is within ε of at least some section of the trajectory.

To compute the force for FOLLOW mode we identify the closest point on the trajectory, and command a spring force drawing the Phantom towards it. This is the radial component of the vector field. To this we add the tangential component, a force damped to the tangent velocity of the trajectory at that closest point. That is, if the Phantom is at p and the closest point is $c(p) = \varphi_1(t)$, then $\dot{\varphi}_1(t)$ is the tangent velocity vector at that closest point. If we denote the velocity of the Phantom as $\dot{p}$ then the radial plus tangential components of the vector field is computed by $k_r(\varphi_1(t) - p) + b_d(\dot{\varphi}_1(t) - \dot{p})$ where k_r is a spring constant and b_d is a damping constant.

5 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 [17, 4, 6, 22] 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. Control systems and abstractions in this work have been important in building our haptic system. Vector fields have been widely used in robot control [11, 12, 19, 18, 3], and these mathematical foundations were influential in our system design. Nonholonomic control and HOVFs 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, 23] multi-target motion interpolation [20], real-time control of virtual humans [10], retargeting of motion [7], motion transitions [21], 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 animation's dynamical systems using vector force fields. Our paper represents a first step towards realizing that goal.

6 Materials and Methods

A key element of our system is the PHANToM (Personal haptic interface mechanism) from SensAble Technologies[2]. It is currently hooked up to a Pentium III workstation running Windows NT but may alternatively be used with a Silicon Graphics O2 running Irix. We also wrote a Phantom driver for 3D Studio MAX, (a commercial animation authoring package from Kinetix[3]) which runs on the NT workstation. We use 3DStudio as an animation back end, and in particular we use the Character Studio plug-in for importing and animating motion capture files.

[2]http://www.sensable.com/
[3]http://www.ktx.com/

7 Modeling and Algorithms

7.1 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 in Sec. 2.3. When two rivers R_1 and R_2 intersect, a "combined" vector field must be defined.

The naïve approach would be simply to add the two vector fields together, i.e. $V_1 + V_2$. However, if we are following one trajectory, we don't necessarily want to feel a pull towards another trajectory. For example, suppose one trajectory represents a character walking across a room, and another trajectory represents the same character performing a standing jump. The two trajectories might cross, but we don't want the walker to give a little involuntary hop at that point due to the inappropriate influence of the jumping trajectory.

Our solution is to use the velocity of the Phantom to determine the relative influence of the vector fields, and not simply its position. This makes it easy to cross from one trajectory to another only when they intersect more or less at a tangent, i.e. when they are proximate in phase space.

For intuition, let us ignore the radial forces Y_i until Sec. 7.2. 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. 8. 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)*.

7.2 Higher-Order Vector Fields

A HOVF is like a standard vector field, in that it defines a *constraint* that the integral curves must follow. HOVFs are related to nonholonomic constraints.[4]

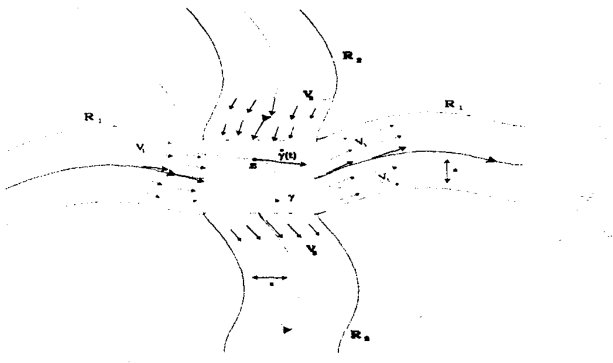

Figure 8: Two rivers R_1 and R_2 cross in the purple square. Suppose the user is guiding the Phantom along some trajectory γ in C, and at time t, the point $z = \gamma(t)$ lies within the square. The force experienced by the user should depend on the direction of motion (i.e. on the velocity $\dot\gamma(t)$). 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 $V_2(z)$.

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. 7.1, we define $f_p(v) = V_1(p)$ when $v \approx X_1(p)$, and $f_p(v) = V_2(p)$ when $v \approx X_2(p)$, and 0 otherwise. This construction is "discrete"; we have also experimented with a smooth version.

7.3 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 HOVFs have the form[5] $L : TC \times I \longrightarrow TC$, 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.

[4]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.

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

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 7.2. 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.[6] This HOVF paradigm was employed in all the FOLLOW examples in this paper.

8 Extensions and Future Work

In LEADING mode, our system can take a sample motion, e.g. a walking or dancing figure, and alter the motion by specifying a new path for the walker. The Phantom leads the figure around, but the haptics prevent the user from changing direction too suddenly, using a variant of simulated inertia. We may want the system to behave differently when the figure has one foot on the ground, or two, or none at all.

We can think of the Phantom as specifying a moving objective for the walking figure. As before, in the absence of user-supplied forces the Phantom and the figure will describe a default trajectory. Move the Phantom to one side and the figure will attempt to turn and follow, and the force-feedback will discourage further exploration while the figure is catching up. Move the Phantom directly back and the figure will step backwards, retracing its earlier motion in reverse. The system remembers the path taken so far, so the haptics can guide the Phantom while backtracking.

We would also like to be able to handle transitions between several input motions, in a manner consistent with the scheme described in Sec. 7.1.

8.1 The Lifting Problem

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:

$$\begin{tikzcd} C \arrow[r, "h"] & D \\ I \arrow[u, "\widetilde{\alpha_i}"] \arrow[ur, "\alpha_i"'] & \end{tikzcd} \qquad (1)$$

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).

9 Summary

We have designed and implemented a system in which haptic vector fields enable a user to manipulate high-dimensional streams of motion capture data naturally and in real time. Our system deals with animation motion capture data but the techniques we have developed may easily be applied to guided teleoperation of machinery, or simulation of robots or virtual humans with many degrees of freedom. Our use of the Phantom haptic device is novel in that it applies to motion data and is not limited to surface interaction with 3-D models. The force-feedback allows the user to navigate the environment with confidence, since "good" motions are favored by the system and "bad" regions of the configuration space are discouraged by haptic vector fields which steer through dynamically good channels for the parametric animation.

References

[1] J. Baillieul, R. Brockett, B. R. Donald et al. *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.

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

[3] K.-F. Böhringer, B. R. Donald, and N. C. MacDonald. Programmable Vector Fields for Distributed Manipulation, with Applications to MEMS Actuator Arrays and Vibratory Parts Feeders. *Int. Journal of Robotics Research,* vol. 18(2), Feb., (1999) pp. 168–200.

[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] T. Lozano-Pérez. Spacial planning: A configuration space approach. *IEEE Transactions on Computers*, C-32(2):108–120, Feb. 1983.

[16] T. Lozano-Pérez, M. T. Mason, and R. H. Taylor. Automatic synthesis of fine-motion strategies for robots. *International Journal of Robotics Research*, 3(1), 1984.

[17] 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.

[18] 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.

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

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

[21] 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.

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

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