A Framework for Assisted Exploration with Collaboration

Eric A. Wernert Andrew J. Hanson *

Computer Science Department
Indiana University
Bloomington, IN 47405 USA

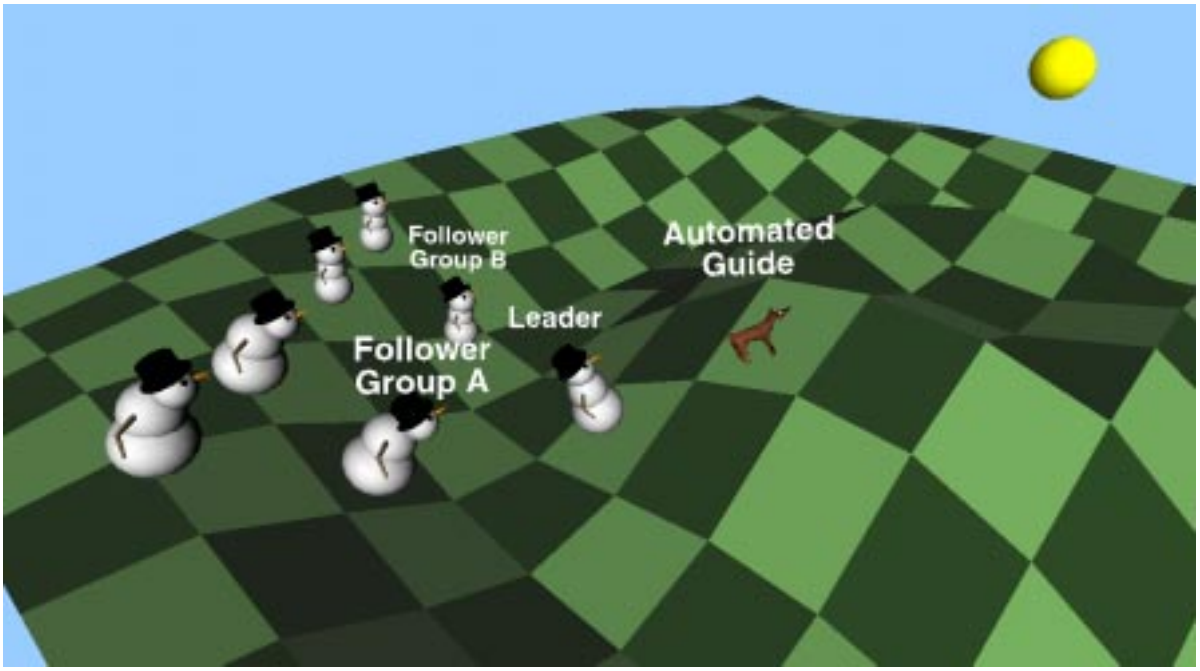


Figure 1: Assisted collaborative exploration scenario with the leader's guide avatar being a dog, and two groups of following collaborators. Observe that the guide avatar points to objects of interest even if they are not aligned with the direction of locomotion or the observer's ground plane, and that the local leader serves as the guide avatar for all attached followers.

Abstract

We approach the problem of exploring a virtual space by exploiting positional and camera-model constraints on navigation to provide extra assistance that focuses the user's explorational wanderings on the task objectives. Our specific design incorporates not only task-based constraints on the viewer's location, gaze, and viewing parameters, but also a personal "guide" that serves two important functions: keeping the user oriented in the navigation space, and "pointing" to interesting subject areas as they are approached. The guide's cues may be ignored by continuing in motion, but if the user stops, the gaze shifts automatically toward whatever the guide was interested in. This design has the serendipitous feature that it automatically incorporates a nested collaborative paradigm simply by allowing any given viewer to be seen as the "guide" of one or more viewers following behind; the leading automated guide (we tend to select a guide dog for this avatar) can remind the leading live human guide of interesting sites to point out, while each real human collaborator down the chain has some choices about whether to follow the local leader's hints. We have chosen VRML as our initial development medium primarily because of its portability, and we have implemented a variety of natural modes for leading and collaborating, including ways for collaborators to attach to and detach from a particular leader.

Keywords: wayfinding, locomotion, navigation, exploration, collaboration, virtual reality, VRML

1 Introduction

Getting around a virtual environment or data set while keeping track of one's whereabouts and task objectives is the subject of a number of techniques often referred to by such names as wayfinding, locomotion, and navigation. We advocate a family of approaches involving the incorporation of task-based constraints on the navigation parameters such as viewer position and orientation; this approach enables the environment designer to provide extra assistance to keep the user's explorational wanderings and attention focused on the task objectives [10, 11].

The critical feature of such constraints is the enablement of behaviors that are not possible with simple "movie-like" animations along one-dimensional parameter paths, and which are more goal-directed than free motion. For desktop systems, "fishtank" virtual reality displays, and fully immersive display environments, we supply at least two degrees of freedom corresponding to mouse move-

*Email: {ewernert, hanson}@cs.indiana.edu

ments or horizontal controller movements, plus a variety of options for controlling the gaze direction of the user's navigation frame. The same general theoretical structure naturally includes other layers of interpolatable state variables such as points of interest, spotlights, fog, "gravitational" attraction to critical positions, and variable controller sensitivity.

We address a number of issues relevant to collaboration, which has the dual goals of facilitating information and viewpoint sharing while promoting individual exploration and the development of conceptual models. For example, by forcing collaborating avatars to adhere to the constraint manifold while following one of several rules for attaching oneself to a leader, we avoid some frequent problems such as losing your tour group or crashing through the scenery while trying to catch up with the leader.

Previous work in this area ranges widely in its objectives. Methods that intelligently focus on particular scene points include Mackinlay et al. [15], while Phillips et al. [20] construct constraint-based camera placement, and more general control systems are treated by Ware and Osborne [27] and Drucker et al. [6]. Robinett and Holloway [22] employ constraints in view selection, while expert knowledge is utilized by Billinghurst and Savage [1]; wayfinding strategies in general are analyzed in Darken and Sibert [5], while viewpoint control and locomotion in immersive environments have been studied by Bowman, et al., [2, 3]. Other work by Ware and Fleet [26] focuses on the nature of "flying" modes of navigation, and Pierce et al. [21] observe the manner in which image plane interaction can be exploited for getting around the scene. An exhaustive analysis of usability characteristics, including navigation methods, has been undertaken in a report by Gabbard and Hix [8].

Very recent work includes, for example, user studies showing the advantage of a hierarchy of usability methods in virtual reality by Hix, et al. [14], who conclude that "*navigation* ... in a virtual world ... profoundly affects all other user tasks." They also compare exocentric viewing to egocentric viewing, and perform experiments on the tradeoffs between providing user control of many degrees of freedom versus less control but more convenience. The results support the general philosophy of our system designs, which emphasize situations where constraints enabling convenience and efficiency seem to outweigh the freedom of detailed control, although experts may choose to alternate between full freedom and assisted modes.

The behavior of groups of collaborators and the nature of leadership in shared virtual environments were studied by Steed et al. [25, 24], who found that, if only one collaborator was immersed, that person tended to lead, and that a sense of being in the same place aided collaborative performance. Related issues such as the effects of network bandwidth on collaborative tasks in virtual reality are treated, e.g., in Park and Kenyon [19], while Wartell, et al. [28] have dealt with the challenging problem of integrating navigation tasks with a large range of scales.

2 Concepts

The critical issues addressed here include the following:

- **Using constraints to implement assisted exploration.** One way to view constrained navigation is as a generalization of a conventional animation. An MPEG file, QuickTime movie, or videotape has the property that each camera position in the animation has a unique precedent and a unique antecedent, so the exploration of the environment is characterized by a 1D constraint manifold parameterized, say, by t, with a viewpoint $\mathbf{x}(t)$ and camera model preselected at each parameter value t. By playing the animation backward and forward at various speeds, one explores the space of viewpoints preordained by the director of the animation, but has no feeling of free motion or ego-determined control. We assert that by adding one or more degrees of freedom to the animation concept, the feeling of ego-determined control returns even though the designer — the director of the generalized animation — can in fact continue to impose substantial control so the viewer is invisibly assisted in exploring the space according to the designer's choices. Constrained navigation implements the designer's conception of what a viewer is supposed to see and learn while giving the exploration a feeling of freedom and self-discovery.

- **Benefits of Constraints.** When the viewer is given complete navigational freedom, he or she becomes in effect the director of a film with no script. Without constrained navigation, the best a designer can do is to make the important objects stand out and prevent the viewer from running into things; but even this simple assistance can be very expensive to implement with traditional methods. The application of constraints to assisted exploration, in contrast, can do away with such headaches as collision detection and the associated expense entirely simply by restricting the guide manifold to unoccupied subsets of the domain. The viewer is provided with a self-guided script, and need not worry about getting into or mistakenly focusing on unimportant scene areas.

- **Locomotion merged with assisted focus on interest points.** We explore methods for single users to traverse a constrained space with two principal modes of gaze control. The case most like a standard animation has the viewer's gaze predetermined at each position, so the viewer controls the position in parameter space but not the direction of viewing. One variation permits a family of prestored gaze directions that may depend on the direction of motion. To provide an additional sense of control while preserving context, we also can prestore the heads-up direction only, and use the remaining freedom to choose the direction of viewing to be the direction of motion, as though driving a vehicle over a terrain that determines heads-up. We then supply a separate indicator (typically provided by an automated guide avatar) that points towards the object of interest at each point (or the selected direction out of a family of directions keyed to the motion direction); if this hint interests the user, the user can stop moving and the gaze will be turned toward the interest point. With additional controller degrees of freedom, one can pivot the gaze at will while continuing to travel in a different direction, like looking out the side window of a car.

- **Gaze interpolation and sense of orientation.** There are a variety of choices to make when the user stops moving, and when the user resumes motion. First, the gaze can be constrained to the plane perpendicular to the heads-up direction and rotated in the general direction of the interest point, but typically not looking directly at it. A second choice is to interpolate the gaze from the current traveling direction to the actual interest point, wherever it may be; this has some advantages for off-track points, but can be more disorienting. Finally, when motion resumes, we can either swing *back* to the remembered prior vehicle orientation, *or* we can force the vehicle to start moving in the direction of the current gaze (which is towards the current interest point).

- **Guide avatars as navigational assistants.** Our experience is that individual navigators require additional help from an automated assistant to help point out where they should be looking. A simple model for this is walking a dog; the user holding the leash in principle chooses the route, where to turn, what roads to cross, but the dog may be continually pointing

its nose in various directions and tugging away at the leash "suggesting" alternative choices. While the user has absolute control of which sidewalk networks to traverse (i.e., the freedom permitted in the guide manifold), the dog's more sensitive perceptions can lead to explorations in directions the leash-holder would not have thought to go. It is thus up to the designer of this multiple-degree-of-freedom movie to suggest task-effective exploration directions and provide them to the automated guide.

- **Collaborative tools for assisted exploration.** Collaboration modes occur almost automatically as extensions of the single-user assisted exploration modalities. Just as the guide avatar provided by the designer of the prestored environment tugs at the single user, any single user can have several followers for which he or she serves the same purpose, focusing their attention by turning towards significant environmental features. This directly facilitates the goal of collaboration, i.e., the sharing of information and experiences. Additional richness derives from the fact that each person has the objective of helping the other collaborators achieve new understandings, and exactly who is teaching whom may be ambiguous and may change rapidly in a true collaboration. Thus we look at ways in which collaborators can have some freedom to wander with varying looseness of the tether to the leader, making it possible to break away from the main group and rejoin at the touch of a button, and facilitate exchange of leadership roles.

- **The VRML environment as a portable implementation.** The original prototype of the system described here was implemented on the desktop in Iris Explorer and Inventor, and then ported for further testing to the CAVE™ [4] immersive environment using Performer. The main advantage of these environments is high performance at the cost of portability and ease of exposure to a larger base of users for feedback. As new projects arose with additional needs for portability, user-modifiability of the data models, and long-distance multi-collaborator sharing of visualization experiences, it became apparent that a more portable implementation was needed. The obvious choice of a relatively portable and user-modifiable 3D graphics environment was VRML as supported by a number of plugins for Web browsers [13]. A wealth of publicly available models and avatars become available for free, and the system is potentially available for a wider variety of future projects and users. Since VRML worlds can also be displayed in the CAVE using tools such as Open Worlds™ [18] and `cave6u` [16], we can also utilize our system to some extent in immersive display environments. More significantly, VRML browsers can also run on almost any recent PC in addition to high-end SGI workstations that have been the traditional domain for such work.

  There are fundamentally two choices to make in the design decision to use VRML: whether to add private enhancements via `VRMLScript`, a close cousin of `JavaScript`, which is slow and somewhat awkward but widely supported, or via Java and the EAI (External Authoring Interface [7]). It turns out that Java and EAI are essential for any activities that must communicate among different machines fluidly for remote collaboration; however, if only a single user is involved in a given visualization, `vrmlscript`'s simple data exchange interface makes it relatively easy to use and extends its portability.

  One elegant feature of the VRML implementation is that the array of viewing parameters and the guide manifold themselves may, with some labor in the scripting language, be initially stored and even viewed as vanilla VRML objects.

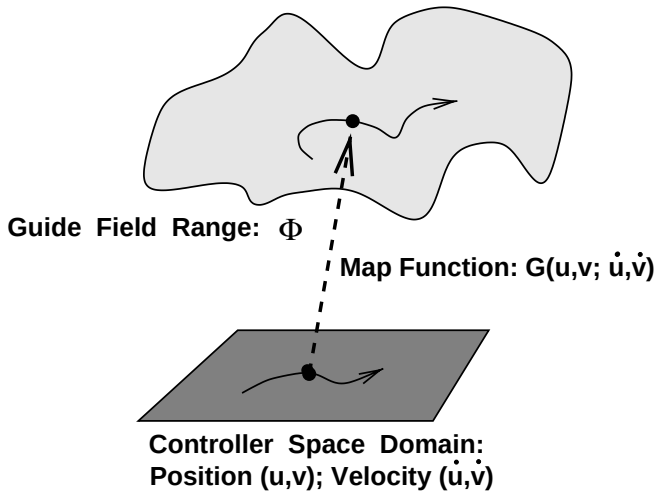


Figure 2: The constrained navigation approach to assisted exploration involves tandem interpolations among sampled parameter sets associated with each point of the controller space.

Then one has the option of inspecting the designer's navigation space before enabling it as a constraint system for its associated scene. Constraint manifolds and arrays of view parameters simply become VRML files that are loaded with the scene, but that have a special meaning for the navigation controllers implemented in Java and/or `vrmlscript`.

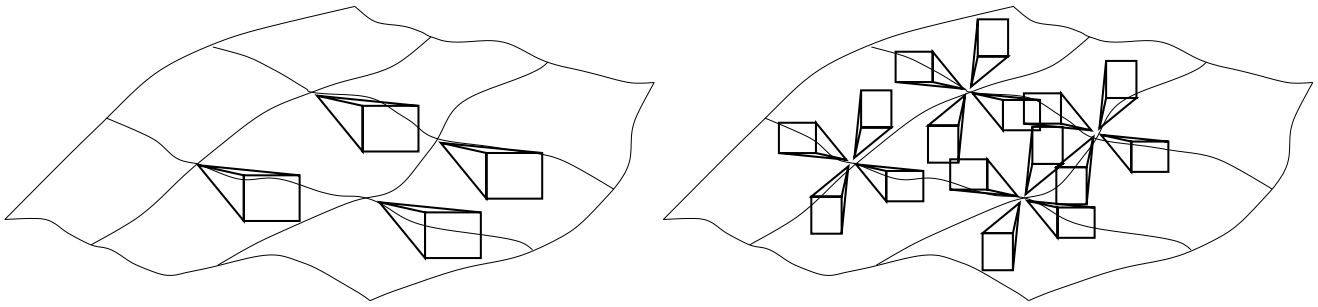
3 Constraints and Assisted Exploration

In previous work, we described the general framework of constrained navigation [10] in the context of an arbitrarily complex space of scene-viewing parameters attached to each point of a navigation manifold spanned by the motions of a chosen controller, as shown schematically in Figure 2. Typically, we map a controller space of dimension two or greater, $(u,\ v, \ldots)$ and its optional velocity information $(\dot{u},\ \dot{v}, \ldots)$, into a sampled set of viewing parameters denoted by the map $\mathbf{G}(u, v, \ldots; \dot{u},\ \dot{v}, \ldots)$ from the domain of the control device to the full space Φ of parameters.

$$\mathbf{G} : (u, v, \ldots; \dot{u},\ \dot{v}, \ldots) \mapsto \Phi\ . \tag{1}$$

The objects in the range Φ include such things as

1. Viewer position on guide manifold: the point in the universe where the virtual owner of the device appears to be standing.
2. Viewer gaze orientation: where the virtual user is actually looking.
3. Task-driven suggestion for gaze orientation: where the virtual user should be looking to perform well on the task at hand. This can be either a direction or a specific point of interest in the environment.
4. Viewing parameters: focal length (wide angle, telephoto lens), depth of field, and binocular convergence.
5. Viewing properties: fog, light attenuation, etc.
6. Control modifiers: mouse response, importance weighting, gravity attraction, etc.
7. Visualization application parameters: streamline characteristics, particle source location, pseudo-color assignments, etc.



(a) (b)

Figure 3: (a) Each point of the guide manifold permits one single viewing choice, interpolated among the sampled values selected by the designer. (b) A compass rose of view parameters is preselected at each sample point of the guide manifold to allow a viewpoint keyed to the user's direction of motion.

4 Fundamental Methods.

4.1 Single user locomotion/navigation.

We begin our presentation of the implementation methods of the current system with the single-user navigation paradigms that we have found most useful. Experience with assisted navigation procedures has led us to work with two fundamental frameworks: *Fixed Gaze* and *Motion-Weighted Gaze*.

Fixed Gaze. There are two subclasses of fixed gaze. The first, illustrated in Figure 3(a), permits the user no freedom; wherever the controller is positioned, the viewing parameters are given uniquely by multidimensional interpolation of the sampled values supplied with the guide manifold. Wherever one moves, the gaze direction is predetermined, even if it points in the opposite direction to the viewer's motion. This mode is well suited to applications such as molecular inspection that require the gaze to be continually readjusted to face towards a complex, often nonconvex, object. The second, illustrated in Figure 3(b), is somewhat more clever: by storing a compass rose of fixed directions at each sample point, we can use the user's velocity vector to select from an interpolated family of appropriate directions. Instead of having the view parameters fixed by the position alone, the parameters are determined uniquely by the position and velocity $(u, v, \ldots; \dot{u},\ \dot{v}, \ldots)$ taken together.

Motion-Weighted Gaze. In motion-weighted mode, the user is imagined to be traveling in a sort of automobile or golf cart, or perhaps pulled by a sled dog. The heads-up direction of the viewing frame is fixed at sample points of the guide manifold by the designer, but the horizontal direction is undetermined. On the desktop, the user's gaze follows the direction of motion as long as the velocity is above threshold, though the guide avatar (which we like to choose as a guide dog) may look in the designer-stored directions of interest as they pass. In an immersive display environment, the navigation frame follows the same behavior, though of course the viewer can look off to the side if desired. When the user slows down, the gaze direction shifts to look at the interest point. Two modes are provided when motion resumes: either the original velocity is remembered and the gaze and motion shift back to the previous values, or the interest-point gaze direction becomes the new direction of initial motion and the old parameters are discarded. One other option that we have used is to alter the predetermined heads-up direction temporarily when the user is stopped, and rotate the entire frame, including the heads-up direction, in the plane of the interest direction and the heads-up direction, so the point of interest is centered in the field of view; the predetermined heads-up is recovered when motion resumes.

Navigation Formulas. The basic features of the navigation environment include several variables. First, we have at each point (u, v) of the guide manifold a prior point $(u - du, v - dv)$, so the velocity direction is the difference $(\dot{u} = du, \dot{v} = dv)$. The corresponding point $\mathbf{x}(u, v)$ in physical space thus has an instantaneous velocity

$$\mathbf{V}(u, v) = \dot{\mathbf{x}}(u, v) \approx \mathbf{x}(u, v) - \mathbf{x}(u - du, v - dv)\ . \quad (2)$$

Other environment variables at each navigation parameter include an interpolated, fixed, heads-up direction $\hat{\mathbf{h}}(u, v)$, and a gaze point $\mathbf{p}(u, v)$ or a gaze direction $\hat{\mathbf{q}}(u, v)$; typically, $\mathbf{q}(u, v) = \mathbf{p}(u, v) - \mathbf{x}(u, v)$ and we use the unit normal direction $\hat{\mathbf{q}} = \mathbf{q}/\|\mathbf{q}\|$.

At each point in the journey, the velocity may be computed approximately from neighboring differences as in Eq. (2); however, since the heads-up vector may not be related to the manifold normal, we need a more specific convention. All that is really needed is to use the Gram-Schmidt procedure to find the part of each vector that is parallel to $\hat{\mathbf{h}}(u, v)$ and subtract it:

$$\mathbf{q}_\perp(u, v) = \hat{\mathbf{q}}(u, v) - \hat{\mathbf{h}}(u, v)\left(\hat{\mathbf{h}}(u, v) \cdot \hat{\mathbf{q}}(u, v)\right) \quad (3)$$

$$\mathbf{V}_\perp(u, v) = \hat{\mathbf{V}}(u, v) - \hat{\mathbf{h}}(u, v)\left(\hat{\mathbf{h}}(u, v) \cdot \hat{\mathbf{V}}(u, v)\right)\ , \quad (4)$$

with the corresponding unit vectors $\hat{\mathbf{q}}_\perp = \mathbf{q}_\perp/\|\mathbf{q}_\perp\|$ and $\hat{\mathbf{V}}_\perp = \mathbf{V}_\perp/\|\mathbf{V}_\perp\|$ as usual. $\hat{\mathbf{V}}_\perp(u, v)$ is the corrected velocity direction in the plane orthogonal to the heads-up direction, and should be our standard choice.

Rotating about heads-up. To rotate about the heads-up direction to get as close to the interest-point direction as we can without changing heads-up, we first project to the ground plane using Eq. (3) to get $\hat{\mathbf{q}}_\perp$ as shown in Figure 4(a). Defining

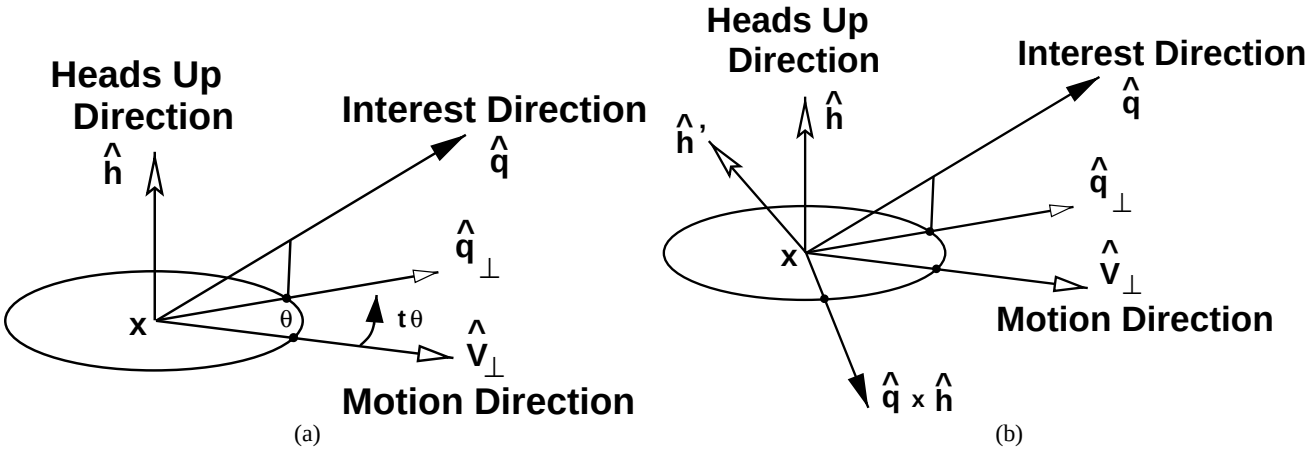
$$\cos\theta = \hat{\mathbf{V}}_\perp \cdot \hat{\mathbf{q}}_\perp$$
$$\operatorname{sign}\theta = \operatorname{sign}\hat{\mathbf{h}} \cdot (\hat{\mathbf{V}}_\perp \times \hat{\mathbf{q}}_\perp)\ ,$$

we find the basic linear interpolation in the parameter t to be

$$R(t\theta, \hat{\mathbf{h}})\ , \quad (5)$$

where we take $R(\phi, \hat{\mathbf{n}})$ to be the standard right-handed 3D rotation matrix about the fixed direction $\hat{\mathbf{n}}$ by angle ϕ.

Tilting to a new heads-up. The alternative interpolation to temporarily relinquish heads-up in favor of centering the point-of-interest $\mathbf{p}(u, v)$ in the view is schematized in Figure 4(b). We represent each frame as a standard OpenGL-like coordinate frame, with



(a) (b)

Figure 4: Motion-weighted gaze modes. The direction of user travel determines gaze direction while moving, but guide avatar may indicate points of interest as they go by. (a) In this mode, if the user chooses to slow or stop, the direction of the point of interest is projected to the plane perpendicular to the current heads-up direction and the gaze is swung around to align with that direction while leaving heads-up invariant. (b) An alternative is to determine two frames, one pointing towards the direction of motion, and the other towards the point of interest, and formulate an interpolation between them by finding the eigenvector of the matrix $F_2 \cdot F_1^{-1}$ and using that to interpolate between the initial and final frames.

the gaze direction the negative of the z direction (in the third column) and the heads-up being the positive y direction (in the second column). Thus, if F_1 is the frame matrix corresponding to looking in the direction of travel at a particular point (u, v), and F_2 is the frame formed by tilting in the plane of $(\hat{\mathbf{h}}, \hat{\mathbf{q}})$ to gaze directly in the interest-point direction $\hat{\mathbf{q}}$, we have

$$F_1 = \left| \begin{array}{ccc} \hat{\mathbf{V}}_\perp \times \hat{\mathbf{h}} & \hat{\mathbf{h}} & -\hat{\mathbf{V}}_\perp \end{array} \right| \quad (6)$$

$$F_2 = \left| \begin{array}{ccc} \hat{\mathbf{q}} \times \hat{\mathbf{h}}' & \hat{\mathbf{h}}' & -\hat{\mathbf{q}} \end{array} \right| \quad (7)$$

where $\hat{\mathbf{h}}'$ can be computed using Gram-Schmidt as usual,

$$\mathbf{h}' = \hat{\mathbf{h}} - \hat{\mathbf{q}} \left(\hat{\mathbf{q}} \cdot \hat{\mathbf{h}} \right) , \quad (8)$$

where $\hat{\mathbf{h}}' = \mathbf{h}'/\|\mathbf{h}'\|$. The key to the interpolation is now to find a parameterized rotation that starts at F_1 and follows a great circle geodesic path to F_2. Such an interpolation is computed starting with the rotation matrix

$$R(\theta, \hat{\mathbf{n}}) = F_2 \cdot (F_1)^{-1} \quad (9)$$

where θ and $\hat{\mathbf{n}}$ are easily determined by standard methods [23, 17]. Then the matrix

$$M(t) = R(t\theta, \hat{\mathbf{n}}) \cdot F_1 \quad (10)$$

is easily verified to reduce to F_1 at $t = 0$ and to F_2 at $t = 1$.

Hybrid Tilt. An intuitively pleasing variant that seems to combine the best of both these methods is to rotate first about the heads-up direction to align with $\hat{\mathbf{q}}_\perp$, and then tilt by rotating about $\hat{\mathbf{q}} \times \hat{\mathbf{h}}$ using the interpolated rotation

$$R(t \arccos(\hat{\mathbf{q}} \cdot \hat{\mathbf{q}}_\perp), \hat{\mathbf{q}} \times \hat{\mathbf{h}}/\|\hat{\mathbf{q}} \times \hat{\mathbf{h}}\|) . \quad (11)$$

to look in the $\hat{\mathbf{q}}$ direction, making overall a kind of "L"-shaped sweep.

4.2 Taxonomy of Assisted Navigation

Most situations appropriate to assisted navigation can be realized with several basic classes of user options, including where to look while *moving*, how to behave when *stopping*, and how to move when *restarting*:

Moving.

- **Fixed Gaze.**
  - **Single Direction.** At each point of the guide manifold, only one fixed set of gaze parameters is stored. Whether one moves backward, forward, or sideways, the direction of view and the family of viewing parameters is always the same, as illustrated in Figure 3(a).
  - **Rose of Directions.** A compass-rose of directions and parameter fields, typically four, is stored at each sample point of the guide manifold, as represented in in Figure 3(b). The travel direction $(\dot{u}, \dot{v})$ is used to interpolate an appropriate value given the sampled values on the compass rose.
- **Motion Weighted Gaze.** If full control of the gaze is relinquished, we can store data or algorithms determining the heads-up direction at every guide manifold sample point, while permitting any viewpoints within the remaining rotational degree of freedom. The mathematical framework for dealing with such freedom is in fact isomorphic to the problem of assigning frames to surfaces with fixed normal directions, and was treated thoroughly using a quaternion framework in recent work by one of the authors [9, 12].
  - **Watch where you are going.** Using the model of a traveling vehicle such as an automobile or golf-cart, the user will want to focus on the "road," and so the gaze will be aligned with the direction of motion consistent with the fixed heads-up direction. Note that the correct plane for the vehicle velocity is determined by interpolating the vertex normals, so the proper velocity is found from Eq. (4) by projecting the actual difference between successive vehicle (u, v) positions to that local plane.
  - **Be slightly distracted.** We assume that some, if not all, points along our route of locomotion have directions of interest that may be pointed out to us by the guide avatar. If it is task-appropriate, we may wish to allow our gaze to be distracted from the motion direction towards the interesting direction using a certain percent-

age or weight that would logically depend on our velocity.

- **Dual gaze and motion control.** In an immersive environment as well as in a desktop system with multiple controls, one can alternatively modify one's gaze direction while continuing to move independently in a different direction.

Stopping.

- **Force the gaze to preserve heads-up.** Assuming that a pleasant, scene-oriented heads-up constraint has been supplied for the guide manifold, we could assume that, wherever we are, we have only the freedom to pivot about that fixed (interpolated) direction determined by the sampled grid values. In this case, an interest-direction above or below the viewer's plane cannot be gazed at directly, but must be approximated; normally this involves projecting the gaze to the user plane and pivoting to face in the direction of this projection, as shown in Figure 4(a).

- **Gaze in the direction of interest.** If we are willing to risk the disorientation of losing the heads-up direction temporarily in the interest of having the object of attention centered in the view, we can perform a more complex view interpolation when the viewer stops to look. In this case, we determine the plane containing the desired gaze direction and the heads-up direction and force the temporary, new heads-up to lie in that plane. This determines all the new frame parameters, and we simply do a frame interpolation directly to this orientation starting from the original traveling frame, as shown in Figure 4(b).

Restarting.

- **Remember prior direction.** Using the model of a vehicle as our mode of travel, we can assume the vehicle has stopped but has not turned. Thus no matter where we turned our head to look, the vehicle will start moving again in its old direction, and we will swing our heads gradually to align with that direction as we gain velocity. The vehicle remains fixed as shown in Figure 5(a); only the view direction changes.

- **Redirect towards interest point.** Alternatively, the vehicle can turn with us as we stop, as shown in Figure 5(b). The interpolation is computed according to Eq. (5) which was used also to compute Figure 4(a). Then when we restart our journey, there is no memory of the prior state, and we trundle on in the direction of the interesting object we stopped to observe.

The utility of these modes cannot be evaluated universally; the appropriateness depends on the task and context.

5 Guided Collaboration

The basic environment that we have described embodies a natural extension to a collaborative system. The assisting features of the guide avatar may be exploited in the extension to collaborative viewing. Among the relevant properties we note the following:

- **Tracking and Watching.** The leader's guide appears in the center of the view of the leader, but the leader appears in the center of the view of the followers, so as the leader tracks the guide, the collaborators track the leader. Each human leader's avatar is seen by the attached collaborators in a central location; if the leader is immersed in a system with a headtracker, the followers see the leader's head motions much as the leader can see the pointing motions of the automated guide avatar. And each following collaborator can in turn be a leader for another group of followers, who will see him or her in the leadership position, as shown in Figure 1.

- **Sharing Points of Interest.** When the leader stops to check a point of interest, all the following collaborators either turn in place, or swing, attached to the leader, to face that direction while remaining within the legal guide manifold. This avoids many troublesome navigation problems for the collaborators, since they can pay visual attention to what is being emphasized without extra navigation effort to get in a good position. In regions where the designer knows in advance that collaborative viewing of an object will take place, extra care can be taken to construct an "arena" — a bowl-like area in the guide manifold that will let the collaborators place themselves in good viewing positions as though sitting on a hillside above the guide. Alternatively, in applications like molecule inspection that may require gaze directions essentially orthogonal to the guide surface, it may be better to use a model in which the collaborators swing out of the guide manifold behind the leader (see, e.g., Figure 7(b)).

- **Tethering Freedom.** Of course, there will always be some situations in which the collaborators will have to jockey their positions to get a point of view, but each has some degrees of control of their situation as well. Returning to the leash analogy, the collaborators following a particular leader are treated as though constrained by elastic leashes. The can wander from side to side, pull themselves up close behind the leader for a better view, or drop back. One mode permits the following collaborators to detach completely from the leader and navigate the guide manifold with their own guide avatar if they are not interested in what the leader is pointing out. A press of the button re-attaches the leash or tether, and the straggler rejoins the group with a slow-in slow-out camera motion similar to the standard VRML browser's transitions among prestored viewpoints.

- **Speed Sensitivity** When traveling at reasonable speed, the viewer will be moving in a particular direction and will be centered directly behind the local guide. Regardless of the speed, the human or automated guide avatar will typically turn its head or pointer to indicate interesting things that are being passed by. If the navigator slows below a threshold, all are coerced to look in the "preferred" direction. Upon resuming speed, travel resumes according to the chosen options from Figure 5. The redirection mode of Figure 5(b) may be preferred for continuous over-the-shoulder collaborative viewers.

- **Intelligent Dragging.** If a collaborative tethering paradigm is too rigid, the following viewers can easily bump into walls or get disoriented. For terrain-like environments, our system keeps the viewer tied to the constraint manifold at a fixed distance and horizontal angle from the guide, thus eliminating the need for expensive collision detection, and permits bringing the reins up tight, wandering loosely behind the leader, or detaching and requesting an automated return later. For molecule-like environments, or situations where the collaborators need to look from a viewpoint that is behind their guide, but cannot be exactly on the guide manifold because the guide is facing perpendicular to the guide surface, the collaborators can be constrained to keep a fixed relationship to the guide per se, rather than being constrained to the navigable manifold; the leading guide, however, retains context and keeps a position on the constraint manifold while staying in full view of the collaborators.

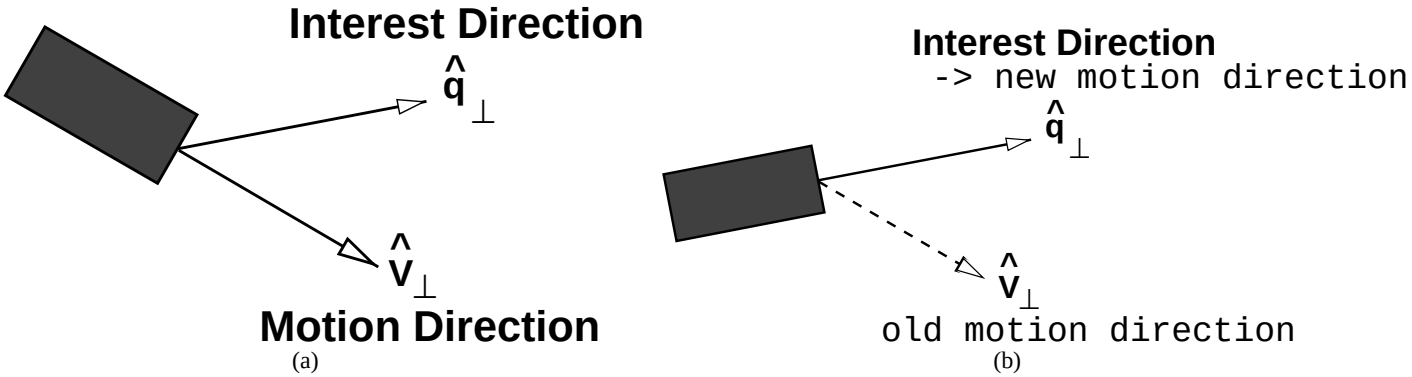


Figure 5: Restart modes after stopping to inspect an interest point. (a) Leave vehicle pointing in original direction, turning only the gaze when inspecting the interest point; continue in original direction of travel when resuming motion. (b) Rotate the vehicle in the ground plane, basically as in Figure 4(a), to align as closely as possible with the gaze direction, and then resume motion in that direction, forgetting the original direction.

6 Examples

Terrain Following. In Figure 6, we show examples of an environment represented by a 3D terrain elevation map. This environment is distiguished by the assignment of a fixed heads-up vector to each point of the plane, typically either oriented with gravity or with the pointwise terrain normal vectors. Each motion produces a new camera model constructed from the prestored normal and the shape of the object being navigated.

Molecular Examination. We first get a hint of the value of constrained navigation when we examine the cumbersome environment of a large molecule in Figure 7. Here, choosing the heads-up direction requires significant designer attention to match the particular application, which might be the examination of a protein binding site or unusual geometry.

Non-Convex Buildings. Standard 3D viewers attempt to see all aspects of an object by "rolling it over" in your virtual hand. The critical problem with this approach is that it does not account for various non-convex nooks and crannies which could keep the viewer in a standard examiner from seeing important details. In Figure 8, we show how a bubble-like navigation field can settle over a very nonconvex building or an entire set of buildings without any need for complex user actions or prior knowledge of the interest points.

7 Conclusion and Future Work

We have implemented a designer-assisted approach to the problem of exploring and visualizing a virtual space that permits a natural extension to shared collaborative experiences. The paradigm exploits positional and camera-model constraints on navigation, as well as taking advantage of additional variables such as interest-points to provide extra assistance that focuses the user's explorational wanderings on the task objectives. Our specific design incorporates not only task-based geometric constraints on the viewer's location and gaze, but also a personal "guide" that serves two important functions: keeping the user oriented in the navigation space, and "pointing" to interesting subject areas as they are approached. Collaboration in a particular style is supported by recursively attaching following collaborators to a guiding avatar, which may be either an automaton or a human. By implementing the constraint configurations as VRML objects with additional navigational interpretation, we achieve a new level of portability and availability.

Among the areas remaining for future work, we note:

- **Scaling.** Scaling up to large applications such as the entire earth requires the introduction of interesting new complexities such as multiresolution constraint grids. See [28] for an initial attempt at this problem.
- **Time Evolving Systems.** Data, points of interest, and guide manifolds that vary with time present additional challenges due to the requirements imposed by additional dimensions of interpolation and context knowledge. An example of such a problem would be modeling an arctic hiker traversing an ice flow that is in the middle of breaking up and drifting away in pieces.
- **User Testing.** Gathering user data on effectiveness of the various navigation modes is currently in progress. Specific measurement and data-recording tools need to be added to support this task. A quite advanced system for supporting such inquiries has been presented by Hix et al., [14].
- **Collaborative Testing.** Collaboration data are particularly difficult to come by because of the challenge of finding groups of users with an appropriate problem and resources. Several projects currently underway in our laboratory, including work on immersive exploration of astrophysical data, are anticipated to provide an appropriate framework. A sampling of the questions specifically related to virtual collaboration that need to be understood has been studied recently by Tromp, et al., and Steed, et al. [25, 24].

Acknowledgments

This research was made possible in part by NSF infrastructure grant CDA 93-03189 and the support of the Indiana University Advanced Information Technology Laboratory.

References

[1] M. Billinghurst and J. Savage. Adding intelligence to the interface. In *Proceedings of VRAIS '96*, pages 168–175, 1996.

[2] D. Bowman, D. Koller, and L.F. Hodges. Travel in immersive environments: An evaluation of viewpoint motion control techniques. In *Proceedings of VRAIS '97*, pages 42–52, 1997.

[3] D. Bowman, D. Koller, and L.F. Hodges. A methodology for the evaluation of travel techniques for immersive virtual environments. *Virtual Reality: Journal of the Virtual Reality Society*, 3:120–131, 1998.

[4] Carolina Cruz-Neira, Daniel J. Sandin, and Thomas A. DeFanti. Surround-screen projection-based virtual reality: The design and implementation of the CAVE. In James T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 135–142, August 1993.

[5] R. P. Darken and J. L. Sibert. Wayfinding strategies and behaviors in large virtual environments. In *Proceedings of Human Factors in Computing Systems (CHI '96)*, pages 142–149, 1996.

[6] S. M. Drucker, T. A. Galyean, and D. Zeltzer. Cinema: A system for procedural camera movements. In *Computer Graphics*, pages 67–70, 1992. Proceedings of 1992 Symposium on Interactive 3D Graphics.

[7] EAI. External authoring interface. A Java interface specification for VRML; see http://www.vrml.org/WorkingGroups/vrml-eai/Specification/ for more information.

[8] Joseph L. Gabbard and Deborah Hix. A taxonomy of usability characteristics in virtual environments. Report for ONR, November 1997. Obtainable from http://csgrad.cs.vt.edu/ jgabbard/ve/taxonomy.

[9] A. J. Hanson. Constrained optimal framings of curves and surfaces using quaternion gauss maps. In *Proceedings of Visualization '98*, pages 375–382. IEEE Computer Society Press, 1998.

[10] A. J. Hanson and E. Wernert. Constrained 3D navigation with 2D controllers. In *Proceedings of Visualization '97*, pages 175–182. IEEE Computer Society Press, 1997.

[11] A. J. Hanson, E. Wernert, and S. Hughes. Constrained navigation interfaces. In Hans Hagen and Hans-Christian Rodrian, editors, *Scientific Visualization*. Springer Verlag, 1999. To appear in collection based on proceedings of Dagstuhl '97 Workshop on Scientific Visualization.

[12] A.J. Hanson. Quaternion gauss maps and optimal framings of curves and surfaces. Indiana University Computer Science Department Technical Report 518 (October, 1998).

[13] Jed Hartman and Josie Werneke. *The VRML 2.0 Handbook*. Addison-Wesley, 1996.

[14] Deborah Hix, J. Edward Swan II, Joseph Gabbard, Mike McGee, Jim Durbin, and Tony King. User-centered design and evaluation of a real-time battlefield visualization virtual environment. In *Proceedings of IEEE VR '99*, pages 96–103, 1999.

[15] J. D. Mackinlay, S. Card, and G. Robertson. Rapid controlled movement through a virtual 3D workspace. In *Computer Graphics*, volume 24, pages 171–176, 1990. Proceedings of SIGGRAPH 1990.

[16] Swaminathan Narayanan. cave6u. A VRML browser for the CAVE. A description is available at http://www.evl.uic.edu/swami/cave6u/vrml.html.

[17] G. M. Nielson. Smooth interpolation of orientations. In N. M. Thalman and D. Thalman, editors, *Computer Animation '93*, pages 75–93, Tokyo, June 1993. Springer-Verlag.

[18] OpenWorlds. A commercial VRML browser. Information is available at http://www.openworlds.com.

[19] Kyoung Shin Park and Robert V. Kenyon. Effects of network characteristics on human performance in a collaborative virtual environment. In *Proceedings of IEEE VR '99*, pages 104–111, 1999.

[20] C. B. Phillips, N. I. Badler, and J. Granieri. Automatic viewing control for 3D direct manipulation. In *Computer Graphics*, pages 71–74, 1992. Proceedings of 1992 Symposium on Interactive 3D Graphics.

[21] J.S. Pierce, A. Forsberg, M.J. Conway, S. Hong, R. Zeleznik, and M.R. Mine. Image plane interaction techniques in 3d immersive environments. In *Computer Graphics*, pages 39–43, 1997. Proceedings of 1997 Symposium on Interactive 3D Graphics.

[22] W. Robinett and R. Holloway. Implementation of flying, scaling, and grabbing in virtual worlds. In *Computer Graphics*, pages 189–192, 1992. Proceedings of 1992 Symposium on Interactive 3D Graphics.

[23] K. Shoemake. Animating rotation with quaternion curves. In *Computer Graphics*, volume 19, pages 245–254, 1985. Proceedings of SIGGRAPH 1985.

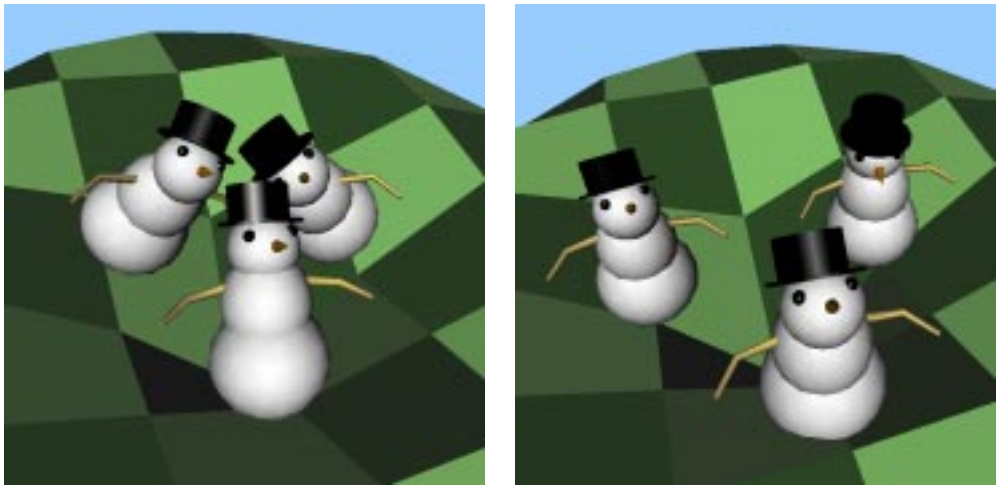
[24] A. Steed, M. Slater, A. Sadagic, A. Bullock, and J. Tromp. Leadership and collaboration in shared virtual environments. In *Proceedings of IEEE VR '99*, pages 112–115, 1999.

[25] J.G. Tromp, A. Steed, E. Frecon, A. Bullock, A. Sadagic, and M. Slater. Small group behaviour in the COVEN project. *IEEE Computer Graphics and Applications*, 18(6):53–63, 1998.

[26] C. Ware and D. Fleet. Context sensitive flying interaction. In *Computer Graphics*, pages 127–130, 1997. Proceedings of 1997 Symposium on Interactive 3D Graphics.

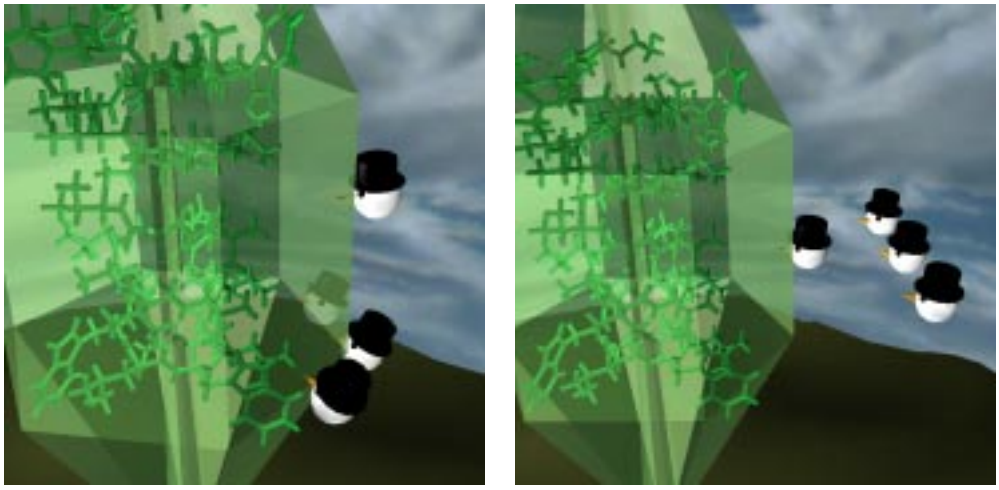
[27] C. Ware and S. Osborne. Exploration and virtual camera control in virtual three-dimensional environments. In *Computer Graphics*, volume 24, pages 175–184, 1990. Proceedings of 1990 Symposium on Interactive 3D Graphics.

[28] Z. Wartell, W. Ribarsky, and L. Hodges. Third-person navigation of whole-planet terrain in a head-tracked stereoscopic environment. In *Proceedings of IEEE VR '99*, pages 141–148, 1999.



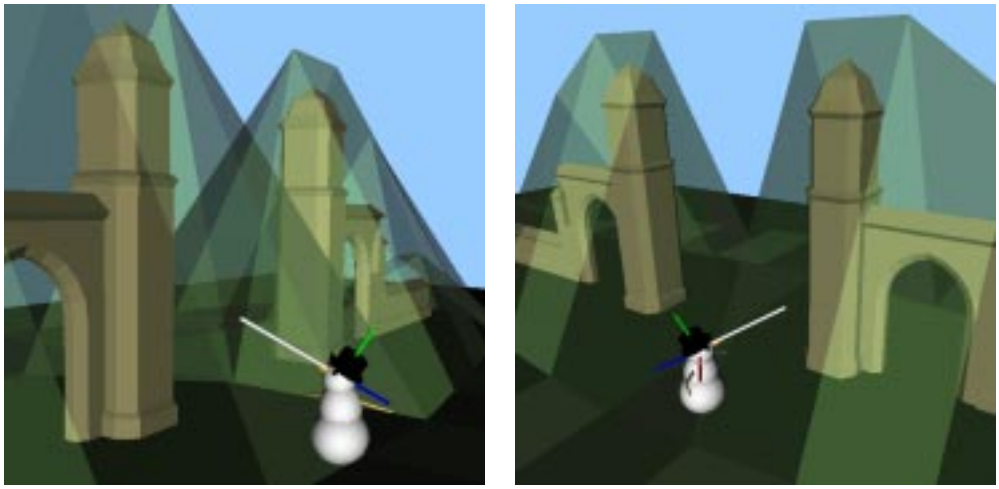
(a) (b)

Figure 6: Two modes of group terrain-traversal. (a) Constrained to the guide manifold using surface normal for heads-up. (b) Using gravity for heads-up. Viewer on right respects the heads-up constraint of Figure 4(a).



(a) (b)

Figure 7: A topologically complex guide manifold. (a) Collaborators cluster around leader constrained to surface. (b) Collaborators hover with fixed orientation to leader, who remains in guide surface.



(a) (b)

Figure 8: Nonconvex structure that confounds “examiner” browsing mode. (a) Moving to the left gate, gaze is on left gate. (b) As we move to the right gate, the navigation constraints effortlessly shift the gaze.