Viewpoint Adaptation during Navigation based on Stimuli from the Virtual Environment

Szilárd Kiss*
University of Twente

Anton Nijholt†
University of Twente

Abstract

We consider the possibility of automatically modifying the user's viewpoint orientation for the purpose of enhancing the navigation experience. We concentrate on outdoor virtual environments where the terrain is uneven as well as certain cases of indoor flat environment floors, respectively on environments where focus-drawing objects and different types of obstacles exist. Most natural environments present these properties and virtual environments that are based on natural environments are in abundance. The navigational adjustments are introduced as response to environment stimuli such as slope changes, the emergence of visibility rays to focus objects or as navigational difficulty events occur after encounters with non-oversteppable obstacles. A scheme is presented that facilitates the viewpoint orientation feature. Conceptually, the viewpoint orientation is a method of partially and temporarily decoupling the head movements from the navigation direction. Virtual environments in general and their encoding in VRML, X3D, Java3D as well as other scenegraphs are analyzed to reveal bottlenecks in current scenegraphs that make the implementation of such automatic adjustments from difficult to impossible, needing custom navigational code where possible. Different scenarios are presented, where these adjustments can prove useful by speeding up the navigation and providing a higher quality, richer experience. Additional to relevant literature, further experiments are necessary to analyze the effect of automatic view adjustments on the user's immersiveness experience and direction sense (path finding), experiments which could be carried out using custom navigational interfaces incorporating head movement extensions separated from the navigational data, extensions proposed in this paper.

CR Categories: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities; H.5.2 [Information Interfaces and Presentation]: User Interfaces—Interaction styles; I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction techniques; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Virtual reality; I.4.8 [Image Processing and Computer Vision]: Scene Analysis—Motion, Sensor fusion, Tracking.

Keywords: Navigation, attention, automatic viewpoint adaptation, virtual environments.

1 Introduction

Many virtual environments recreate the characteristics of most natural environments: the terrain is sloped, there are obstacles, and there are interesting objects that catch the eye of observers. Navigation in these environments with a rigid viewpoint orientation, linked to the navigation interface, takes away from the experience that the environment should convey to the user. The pre-created viewpoints showing the most interesting views and manual view adjustments can reveal more of the environment, but they are limited in number and convey little or no wayfinding cues in the case of viewpoints, while manual view adjustments could take up more time than a pleasant experience might suggest. Therefore, we propose the introduction of automatic view (orientation) adjustment methods as responses to environment stimuli. In our scheme, we extract terrain height information to visually follow the terrain slope, automatically focus on virtual objects popping into the view radius and to point out the navigational difficulty when encountering (bumping into) obstacles.

Flat environments could also benefit from our method. These environments could contain floor, ceiling, wall paintings or other decorations eventually even objects that are not easily observed with the normal horizontal view when there are more of them: that would require periodical tilting operations to see each target object or "blind" navigation, when target view is acquired but normal (navigation) view is not restored, resulting in looking up or down to some degree during navigation. Imagine that in this case the terrain slope yields for navigation a null or close to null deviation from the horizontal plane, but the head is already tilted, so it gets an adjustment that puts it back temporarily (until the user continues to move) to a horizontal value for a normal navigation.

Our aim is to provide a richer exploring experience by simulating real-life exploration and focusing behavior. Automatic view adjustment is the key to ease of use, where no double control is needed for head (orientation) movements and to the introduction of environment knowledge into the navigation and exploration process. The navigational scheme responds to the properties of the environment, making the user more aware of the surroundings and more effective and successful in exploration.

We present a responsive method that provides automatic focus (orientation) adjustment for navigating variable height terrains and focus-objects enabled environments respectively the implementation problems of this method in current scenegraphs (virtual envi-

*e-mail: S.Kiss@cs.utwente.nl
†e-mail:A.Nijholt@cs.utwente.nl

ronment types). The novelty of the paper consists in the presentation of a new navigational approach which separates movement direction and view orientation parameters as well as methods to introduce environment knowledge into the navigation by using focus objects in combination with a number of existing navigational or graphics techniques applied in new, navigational context.

The next section provides a review of prior research and basic ideas, which are then elaborated in the following sections. Section three presents the general scenario and human exploring behavior, while the following section outlines our navigational aid scheme. We then analyze different scenarios and scene graphs, presenting our test cases and conclusions at the end.

2 Background

In the literature and different textbooks, navigation in desktop virtual environments is mainly regarded as static with respect to orientation [Bowman et al. 1997]. Wherever the user moves, the view orientation follows. This is slightly different in immersive virtual environments such as a CAVE, where the orientation of the avatar is controlled by sensors on the user's head or on shutter glasses [Pape et al. n. d.]. This makes possible the decoupling of head and navigational movements. Since in desktop environments the use of double input devices is rare, for purposes of separated, more natural navigation and exploration modalities, automatic orientation adjustment methods could be used.

Another experiment which provides additional control possibilities for user interaction in virtual environments is [LaViola et al. 2001]. They eliminate hand constraints using special foot hardware and use hand movements for other type of interactions. Although they do not separate orientation and navigation, they do recognize the need for additional control mechanisms. [Bowman and Hodges 1997] analyzes several techniques for manipulating target (sensor equipped) objects. Some of these techniques are automatically created pointing extensions based on environment stimuli. We use a similar approach to respond to the virtual environment stimuli for the purpose of view orientation adjustment for a better navigational experience and enhanced immersion.

The first person perspective used in virtual environments is also widely used in the games industry, along with other types of perspectives and camera handling inspired by cinematographic camera concepts [Halper et al. 2001]. The camera theory aims at capturing from advantageous positions the events that are happening, theory which is applicable to games and virtual environments containing elements that are predefined or automatically viewed, such as replays in games, dialog following [Cassell et al. 1994], etc. However, when the goal is exploration, the main perspective to be used is the first person perspective, while automatic camera positioning breaks away from this view. This makes the deduction or integration of camera theory in virtual environments not always feasible, contrary to the path of camera theory integration in games observed in [Halper et al. 2001].

In this paper, we concentrate on navigational aid techniques for walk-type desktop virtual environments. Unless providing precalculated slope information [Wernert and Hanson 1999] or the methods to calculate height information in the virtual environment like [Steed 1997], we simply use the height information provided by the virtual environment and we calculate elevation changes and act upon this information. Together with real-life observations of subjective human exploring behavior, we tackle simulation problems that are largely disregarded by both navigational and behavioral literature, since exploration is a human-directed cognitive process, and not a biological, brain process to be mapped. We rely on common knowledge about human exploring behavior since at the time of writing this paper we are unaware of studies in this direction. Cognitive literature textbooks analyze eye movement patterns for the purpose of defining the human information recognizing and processing procedure, and not because of why and how explores a human its environment. Human psychology considerations are used generally for directing autonomous agents rather than as navigational aid for user interactions [Gillies and Dodgson 1999].

We also look at the physiological feasibility of our approach. Literature shows that for eliminating confusion and lostness, any viewpoint (scenery) change has to be transitional and must have a minimum duration of 0.1 second to be followable [Nielsen 1999]. This has to be combined with our necessity of quick responses to environment stimuli (based on a movement speed and effectuated distance formula) so that the navigational aid is still useful and with the need of allowing some response time for the user in the case of temporarily focused objects. Thus one of the key concepts we use is gradual adjustment of the view orientation. We also take advantage of servo-ing [Bowman and Hodges 1997] and weighting techniques, applying them to viewpoint orientation respectively focus objects. Snap-to direction control is also useful for visual confirmation of selected targets or orientations. Although the technique is primarily used for positioning in graphics design packages [Bier 1986], it provides an excellent extension to servo-ing.

The research that is most related to our work is presented in [Wernert and Hanson 1999]. They use virtual guides and leaders that orient themselves in interest directions, signaling this way to the user possible exploration paths, and the user can choose to follow the suggestions or not. For this, the user needs to stop, it is automatically oriented, and then depending on the application, continues its navigation in the original direction or in the new, interest direction. This particular construction builds on the VRML format particularities (user is automatically oriented only after stopping) and does not give to the user the decision capability for selecting it's own path, contrary to our approach, where the user does not have to stop at interest directions (orientation adjustment is effectuated while traveling) and the user has the possibility to show interest, disinterest or passivity towards the interest object(s). All this can happen in fractions of a second, comparable to taking a glimpse out on the side windows while driving, and then maneuvering in the desired direction when some promising item shows up. We also eliminate the navigational constraints (user is within a certain range of guide) and the view obstruction (guide is in the center of the view) of following the guide of [Wernert and Hanson 1999], by incorporating the attention drawer mechanism into the interest objects themselves.

Another method that is related to this field is the method of attentive navigation [Hughes et al. 2002], where instead of user orientation adjustment towards objects of interest, special marks (highlight techiques) are used to pinpoint interesting objects from a number of objects in the view frustum.

A quick overview of view orientation properties of different scenegraps like OpenInventor [SGI, Inc. n. d.], OpenSG [Open Source Initiative n. d.b] or Open Scene Graph (OSG) [Open Source Initiative n. d.a] shows a window Viewport and Camera object construction that build upon OpenGL where the interaction is not specified, and has to be programmed on a case to case basis, separate for each application. This makes possible the implementation of interfaces based on separated and automated head orientations.

The method we use for producing stimuli in the virtual environment which are then used to adjust the user's viewing properties, is similar to the virtual sensors concept used by [Thalmann et al. 1997], [Noser and Thalmann 1998] and [Tu 1999]. Their work covers visual, auditory, olfactory and tactile virtual sensors, used for autonomous actor and animal directing respectively simulation. While in those cases the sensor data is extracted using depth buffers, color comparison, scene queries and other methods and transferred to autonomous entities, our method is based on a simpler framework, where the environment data is gathered "passively" (from

custom environment events). The virtual sensory data is used generally to map (to discover) the environment where an autonomous character is located. In our case, we use these environment stimuli to provide navigation enhancements to the target entity, which is the active user-manipulated avatar.

3 General scenario

The hypothetical scenery could be either an outdoor, uneven terrain or a flat terrain with stairs, eventually with target tilted views (like ceiling, floor or wall view) that represent an exploration goal. This scenery would be populated with focus objects that are created as exploration targets and different types and sizes of obstacles that represent navigational diversity, introducing challenges in exploration. We apply our scheme only in case of walk mode, when the user's avatar is pulled down by gravitation (ground). On a side note, the focus drawing method could be applied also to fly mode where gravitation is disregarded, but in that case the main question would be whether the resulting adjustments would provide dizziness or an usable navigational approach since the human navigation is accustomed to gravitation and a more limited number of degrees of freedom that would be possible when exploring by flying, especially in environments with true 3D spatial distribution of objects where the terrain is missing.

3.1 Human focusing behavior

In the most natural resembling scenario, there is an uneven terrain which a user navigates. If it would be the real world, a person's orientation would follow the terrain properties: if the person would go upwards or downwards, the person's orientation would tilt (pitch) upwards, respectively downwards. This is used for enhancing the awareness and for better navigational overview. Immediate (close to person) obstacle checking is not necessary, which is particularly important in virtual environments, where the tactile sensors are left out. This provides also a better support for medium-sized obstacle avoidance, that can not be stepped over and are not in view when they are close, as depicted in figure 1, support that is particularly important in descending cases.

In case of stairs, humans have different behavior. Stair climbing is a repetitive, almost automatically executed task, and as such, the different climbing styles can represent emotional or mental states of the subject. We could even divide the different styles by intuition: staring close to foot could mean uncertainty or shyness, staring at eye level could mean distractedness, while looking up could mean certainty, decidedness. In virtual environments the last style is the more natural, since users are active, inquisitive and won't trip/stumble in the stairs and fall, respectively they will find the way better if taking an overview look over the slope. Furthermore, personal emotional context has little importance since the user is interested in the environment if it's inside and exploring it. Walking in virtual environments requires no attention for foot positioning even in the case when the body (including the foot) of the avatar would be also visualized for greater awareness [Usoh et al. 1999], and visible in downward movements and downward view orientation. However, this raises the problem of foot contact and positioning on variable height terrain, which is not the concern of this paper. The avatar geometry would also need a separation of navigation and view orientation control, since the head (view orientation) would need to tilt separately from the body geometry [Usoh et al. 1999]. Flat environments would only need in such a case the simulation of a walking cycle controlled by the navigational input.

If we look at human exploration within real environments, we could also observe that exploration is not always successful. For instance, in a situation where there are more rooms and not all rooms turn out to be interesting means some failures resulting in

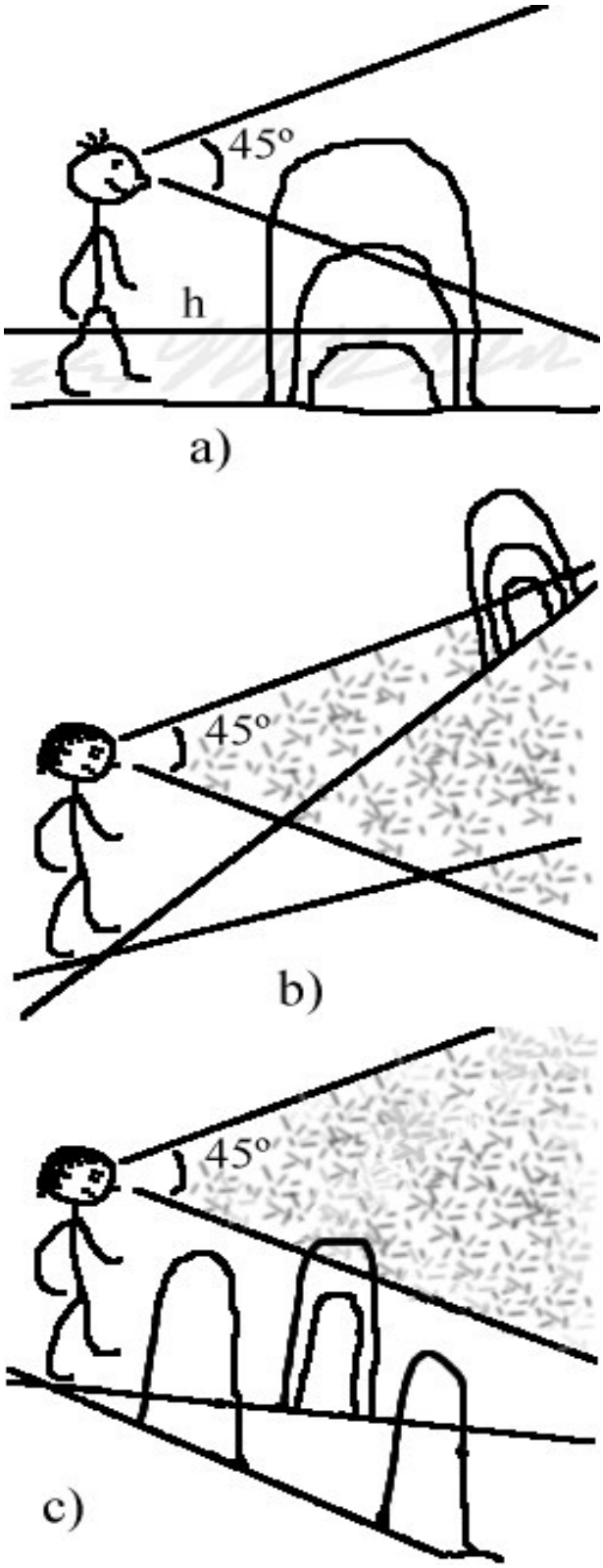

Figure 1: Different cases of view orientation: a). Normal view with the stepover height 'h', medium and big obstacles are easily observed; b) Upwards navigation with normal view reveals size information about obstacle only when obstacle is close; c) Downwards navigation with normal view is like a walk in the dark, if the slope angle is high enough, even the big obstacles aren't in view.

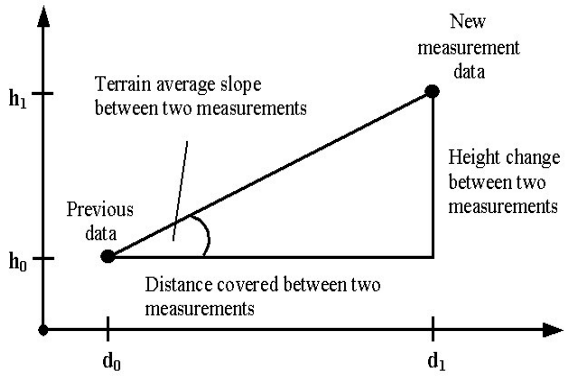

Figure 2: Two successive terrain height measurements.

wasted time. This is similar to exploring a virtual environment, but with live humans, focusing in different directions while navigating is easy and it's not time-consuming. Generally humans show an inquisitive behavior, looking around and sweeping, scanning the environment. This is an important deficiency and thus a negative aspect in virtual environment exploration that we try to facilitate with our scheme. Human exploration is also pretty exhaustive thus simulating automatic focus attention (focus drawing) helps in the exhaustive exploration of the environment in a shorter time than without automatic focusing.

3.2 Obstacles

Obstacle avoidance with large, visible objects is easy even for the non-experienced user. Small, yet non-oversteppable objects can restrain the navigation possibilities, many times without visual cues. For increased navigational effectiveness, in case of unsuccessful movement attempts, view adjustment in the direction of collision (down, since collision with high situated objects is visible from normal view) could easily reveal the navigational bottleneck. In this case also, awareness could be enhanced if the avatar body is also visible [Usoh et al. 1999], but this is still a very expensive process with uneven terrain for desktop computers and for on-line virtual environments.

4 Framework

We present a scheme for the navigation adjustment using two components that are conceptually separated from each other. The first one uses terrain data measurements for automatic pitch adjustments (head tilting), while the second one uses environmental knowledge (focus objects) to direct qualitative discovery by yaw adjustments (head turning).

4.1 Terrain measurements

One of the navigational aid methods is based on height and length distance measurements of the user's position and movement data within a virtual environment. Knowing the distance covered by the user's avatar and the height information, as it is obvious from figure 2, an average terrain slope can be calculated with simple geometric equations. Making the measurements frequent ensures that this average terrain slope is a good approximation for the actual terrain slope.

The data used in our method and the relationships between them can be demonstrated using the graph in figure 3. We make frequent

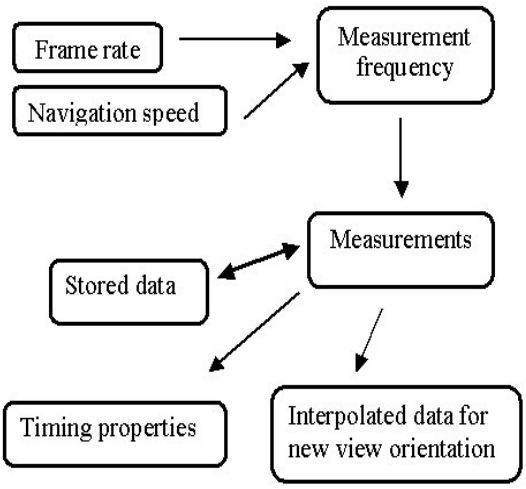

Figure 3: Measurements, data types and their relationships.

measurements, where the frequency depends on the user's navigation speed and the computer's visualization speed (frame rate), but taking in account the 0.1 second necessary for humans to follow the adjustments. In this light, the time between two measurements is at least 0.1 second to be able to accommodate successive adjustment operations. If the navigation speed is slow or the frame rate is low, a less frequent measurement policy is applied.

The data entities and their roles are as follows:

- User set (original) tilting (pitch) angle. This is the value it is returned to when the user stops and there is no change anymore in the positional data.
- The avatar's position at the time of the previous measurement and the current measurement. From this data the avatar covered distance and the change in height can be calculated, and from these data the average slope angle of the terrain covered by the user can be deducted.
- Current orientation is also stored and it is compared to the angle of the slope, which would be the value of the adjusted view orientation needed for the terrain the avatar is on. The difference between the two values is the adjustment value that has to be introduced into the avatar's orientation.
- An angle interpolator gets its start and end values for the orientation adjustment depending on the properties of the orientation setting mechanism. If this mechanism is an addition-based mechanism, then the angular difference is interpolated, or if it is an overwrite-based mechanism, then the previous orientation measurement value is interpolated into the new orientation based on the terrain slope.
- A time sensor controls the orientation adjustment rate. Its duration is at least 0.1 seconds, and varies in function of the frame rate and the user's navigational speed. After a new measurement, the starting time for the time sensor is set to a very little ahead of the current time, so that the adjustment can take place after all the adjustment values are calculated and old measurements values are updated so that new measurements can take place without loosing data.

Unsuccessful attempts to movement in virtual environments also carry information. The most often cause is bumping into a smaller

obstacle that is out of the field of view. In this case, a view orientation is necessary that pinpoints the source of trouble. An adjustment of 90 degree downwards from the standard view (thus a constant view orientation) is enough in such cases. Further movement will redress the optimal orientation for the particular terrain that is being explored.

We differentiate between positive and negative orientation changes, when going uphill respectively downhill. Since in upward movement an orientation parallel with the terrain slope is not as important as with downward navigation and sometimes even not necessary, in this case we opt to use only some percentage of the adjustment suggested from the slope angle, calculated from the horizontal (default) orientation.

Eventually, a dead-reckoning algorithm may be used [Çapin et al. 1997] to approximate beforehand the slope that the user will most probably follow. In this case, an extra step of error correction should be introduced to eliminate the accumulation of small deviances between the approximations and actual measured navigational data. Terrain information is also necessary in this case, since the calculations are not based on the avatar's navigation measurements.

4.2 Focus objects

The focus objects are meant to convey the interesting aspects of an environment to the user in an indirect way, based on the positional context in which the user's avatar is situated. Just like the slope angle influences the pitch orientation, the focus objects are meant to temporarily attract the user's orientation (adjusting it's yaw orientation values), suggesting interesting objects or places to visit, without altering the user's original navigational direction. This way it is possible to make suggestions to the user, and the user is free to investigate, disregard or even repel these suggestions.

Figure 4 is an overview of the focus scheme. Around the avatar's position there is a focus radius specified, which is a limitation for the activation of the focus objects. If a focus object is inside the focus radius, it is an active focus object. When the avatars enters in an area wherefrom the focus object has a visibility line (object can be partially occluded) even when the object is not in direct view, the focus object attracts the view orientation to itself. This attention drawing mechanism has properties that are based on the weight values of focus objects. The actual orientation updating mechanism is essentially the same as in case of the terrain information, only the pitch adjustment is substituted with yaw adjustment.

The characteristics of the focus object weights are:

- The higher the weighting, the more important the object is and more focusing time is allowed.
- Weight values range from 0 to 1, with values below 0.5 indicating already focused objects while 0.5 and above values indicating objects that did not yet exercised their focus drawing rights.
- Weight changes after viewing an object, after focusing an object without viewing it (disinterest) or after demonstrating repulse (turning away from possible interesting object before, during or even after it is focused). The changes are proportional to the action: 0.5 is deducted in case of disinterest, the whole weight but a percentage depending on time spent is deducted when the object was examined respectively a percentage is deducted when focusing is met with resistance (with a minimum of 0.5, so an object that was repelled for instance because of temporal constraints may reappear as focusable object.
- Focus object conflicts regarding focusing order (since focusing is an exclusive operation) are resolved based on relative distances in the direction of navigation as well as based on weight values when the distance differences are not decisive enough.

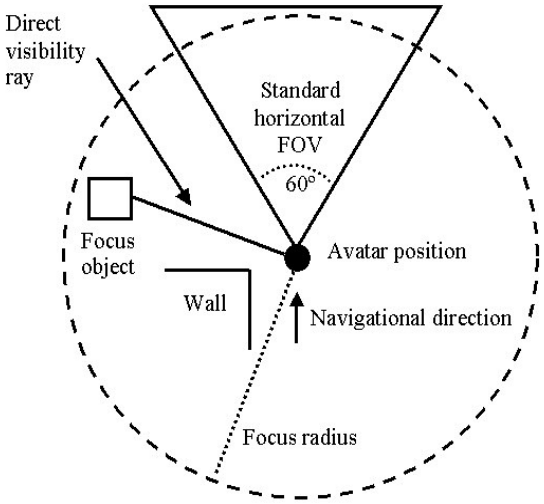

Figure 4: Focusing scheme.

Additionally to the weight based orientation adjustment, servoing [Bowman and Hodges 1997] can be used to increase the responsiveness rate of the system. This would be most useful in cases of repulse, when a focus object attracts the attention of the avatar, but the user has other goals in mind or is under time limitations and decides not to follow the suggestion and effectuates manually a contrary yaw rotation movement. In this case, a greater rotation could be applied than the actual user action indicates to return to the desired (original) orientation that coincides with the movement direction at a faster rate. Another case, although not so important, is servo-ing in case of interest in a focus object, but without decreasing the focusing time duration below the limit of human observation.

Servo-ing may be used in combination with the snap-to technique used in graphics modeling systems [Bier 1986]. Similarly to object positioning using target coordinates and similarly to the normal pitch level used as snap-to level in CosmoPlayer's "tilt" mode, the original orientation of our system (before focusing takes effect) and the focus orientation could act as the snap-to targets such that in case of user actions, when the target orientation is reached a visual feedback is obtained, eventually also a sound feedback.

4.3 Combining terrain and focus adjustments

Theoretically, the terrain-based and focus objects-based navigational adjustment schemes have little in common, being based on different data sets and environment stimuli. They act upon the same target, the avatar's orientation. Since they simulate different behaviours, pitch and yaw orientation, technically they can also be combined. The orientation adjustment schemes can both update the orientation interpolator values which will be the actual data that is used as orientation values, since the updating is done in a prior step, and a combination of orientation angle values can easily be calculated. It is a viable alternative to separately, manually adjusting tilting angle and effectuating orientations respectively paning for a number of times while exploring a virtual environment. The approach of navigating ahead and looking sideways it is at the same time more natural than the pan approach when the user looks in

front and navigates sideways, which in real-life is a necessity only when the neck is stiff for some reason.

When the focus objects are situated at a different altitude than the user (objects above the user or objects in a hollow), simultaneous stimuli could cause different behavior, which could sometimes cancel each other. For instance, if the user hits an obstacle while a bird appears in the visible vicinity above the user, the two adjustments suggested by these stimuli would conflict for the short duration of the focusing adjustment. After the focusing behavior ends the focusing adjustment is removed from the user's orientation, and the more lasting obstacle pinpointing orientation remains. This results in missing a glimpse towards a possible interest object (which is already missed with normal navigation) and a minor drawback through the delay in the obstacle pinpointing orientation adjustment.

In cases where two focus objects might suggest different, conflicting orientation behavior (for instance within passage crossings, when the side passages have both focus objects), there might be necessary to set up an execution order for the focusing orientation adjustments or even choose between them. This could be done based on the weights of the focus objects, but also there is the possibility to base the focusing mechanism on priority concepts, all depending on the target application. For instance, in a zoo environment the weighting can be done on an individual preference basis (which animal is interesting to the user), while in a driving simulation a truck coming from a side-road would have a much higher priority than a parking truck on the opposite side of the road.

4.4 User interaction possibilities

The following is a list of cases of what happens in response to the user navigation:

- The user moves (navigates). In this case, the vertical field of view (pitch) is adjusted to be close to parallel with the terrain slope. The yaw (horizontal field of view) is adjusted only when a focus object becomes visible from the avatar's position, and it is adjusted only for providing a "glimpse" of the focus object. The weight of the focus object is also adjusted. The user selected (original) orientation remains unchanged.

- The user turns while navigating. The terrain slope-based adjustment mechanism remains the same. If a focus object-based adjustment is also effectuated, then the turning suggests interest or repulse, depending on the turning direction. In case of interest the original (user selected) orientation is adjusted towards the focus object synchronized with the user action, while in case of repulse the original orientation remains unchanged and the return is effectuated in an accelerated pace. In both cases, the new respectively unchanged orientations behave as snap-to orientations. The weight of the focus object is adjusted in each case.

- The user stops. In this case, the adjustments made as result of terrain and focus objects stimuli are restored to the original, user selected values.

The adjustment scheme is designed to provide to the user control over the navigation in spite of the automatic view orientations. At the end, the user has to make the navigating actions even if he wants to follow a suggestion done by our scheme. We provide only quick suggestion glimpses and navigational overview for the explored virtual environment, which would be otherwise cumbersome with separate, manual controls for tilting, paning, etc.

5 Scenario in VRML/X3D

Applying the described scenario to VRML [Web3D Consortium 1997] and its successor X3D is almost impossible due to the properties of these standards. There may be some complicated rotational tricks that could satisfy the scheme requirements, but a clean method that we could use is missing. In the following, common user tracking and user orientation updating methods are presented and their shortcomings.

5.1 Tracking and updating the avatar's position and orientation

The user position and orientation can be set and it is generally bound to a Viewpoint node, with the following structure:

```
Viewpoint {
  description ""
  position 0 0 10     # Most important properties,
  orientation 0 0 1 0 # they specify only the local
                      # coordinate system values
  jump TRUE
  fieldOfView 0.785398 # 45deg. vert. 60deg. horiz.
}
```

Tracking in VRML or X3D is possible using a ProximitySensor. This sensor is capable of determining (signaling) the user's global position and orientation throughout the virtual environment, if the sensor's boundaries are big enough to comprise the whole environment. Other elements, such as the bound (effective) Viewpoint node does not contain the user's actual position and orientation. It contains a position and orientation value for the user's starting properties (set by the content creator), and disregards the relative movement that the user effectuates after the Viewpoint was activated (bound).

The VRML/X3D standard document suggests as animation possibility for Viewpoints the update of values inside the Viewpoint (the Viewpoint's local coordinate system) or using an enclosing Transform node to animate the viewpoint. We can observe the following properties of these constructions:

- Internal changes in the Viewpoint (relative movement) cannot be accessed, the Viewpoint does not transmit movement change events after keyboard or mouse manipulation.

- ProximitySensor sees the movement changes, but if we do adjustments inside a bound Viewpoint, the ProximitySensor will notice and propagate the induced changes just like in a chain reaction. To eliminate this reaction, parent Transform nodes could be used for the bound Viewpoint.

- A parent Transform node can host the changes (automatic navigational controls), but because of the internal relative changes in the Viewpoint, the center of the automatic control adjustments won't be aligned with the Viewpoint's local coordinate system. If the global avatar position data is used as the center for the parent Transform orientation adjustment, that will be a previously measured value thus small differences will exist in the actual orientation value, which yields in small position changes, which then will be propagated again as a chain reaction by the ProximitySensor.

- Another possibility would be the unbinding and rebinding of the Viewpoint with global position and orientation values to eliminate the user's accumulated relative position. But since an unbinding and rebinding causes the ProximitySensor to transmit again new values (the position and orientation is

changed) to the Viewpoint, causing a new unbind and rebind action, thus an initial pitch data addition becomes repetitive and the Viewpoint starts spinning in space.

As it can be seen, the standard suggested animations aim to provide viewpoint adjustments without user movements. Unfortunately, none of these possibilities offer simple pitch animation, due to the user's relative position and orientation from the Viewpoint's origin. To overcome these difficulties and provide a scheme that can be used for automatic focusing control it is necessary to gain access to the relative position/orientation of the user's avatar within the bound viewpoint. To achieve this, a new subnode or new (exposed)fields are necessary in the viewpoint node (or a new, separate node to retain these properties).

5.2 Enhancing the navigation

An ideal construction would be a construction that would allow simulating head movements inside the bound Viewpoint. The current Viewpoint node could be extended with two additional exposedFields: an SFVec3f **relativePosition** and a SFRotation **relativeOrientation** to contain, as the names suggest, the relative position and orientation after user navigation within the local coordinate of the bound Viewpoint node. This way, global position tracking and the possibility of simulating separated head and navigation orientations thus also terrain-based and focus-objects induced navigational aid would be possible in VRML/X3D. Since we want to simulate head orientation, the center field for that would be the avatar's relative position from the bound Viewpoint, thus for our purposes a new SFVec3f **relativeCenter** exposedField is not mandatory, but may provide additional manipulation possibilities if introduced.

Another possibility would be to include the relative position and orientation (eventually center) fields in an Avatar node for instance, as a subnode of the Viewpoint node or as a separate node, representing data and manipulating possibilities for the relative positioning and orientation of the avatar inside the coordinates of a bound Viewpoint node.

6 Scenario in Java3D and OpenGL based scenegraphs

Java3D [Sun Microsystems, Inc. n. d.] is regarded generally as a programming API, contrary to VRML/X3D which is considered as a file format, although it has scripting and programming extensions. As a programming API, it is more broad and general, more usable for application programmers than for content creators. It contains interaction (behavior) classes whose functioning have to be specified in customized ways if someone wants to handle the viewing properties (contained in the classes View, ViewPlatform, ViewingPlatform) with separated navigational direction and view orientation controls.

Similar is the case with OpenGL based scenegraphs like OpenInventor [SGI, Inc. n. d.], Open Scene Graph (OSG) [Open Source Initiative n. d.a] or OpenSG [Open Source Initiative n. d.b] where the navigation relays on a Camera class, but it has to be custom programmed for each application based on these scene graphs.

7 Assessment

We presented a scheme to produce automatic adjustments in the horizontal and vertical fields of view (the avatar's gaze direction) that is separated from the navigational controls. The adjustments are created as consequences of environment stimuli, namely terrain height and focus-drawing objects. The scheme permits the additional integration of methods such as orientation servo-ing and snap-to directions. All of this is meant to enhance awareness and provide a faster, more exhaustive exploration paradigm.

The following are the cases where our scheme can provide aid in navigation and environment exploration:

- Cases of variable height terrain that require frequent manual pitch adjustment for a better overview of the navigational direction, especially with fast descend. Yields a consistent navigation with smoother wayfinding.

- Interesting objects can be enhanced with focusing information for drawing the respective objects to the user's attention for exhaustive environment exploration.

- In case of non-straight stairs or steep stairs, the right pitch adjustment can provide information about the shape of the stairs (turns, for instance) or information over objects close to the stairs. Imagine a medium-sized or a little bit broader stairs case, with splendid and diverse flowers on both of the balustrades. Navigating in the middle of the stairs with the standard orientation would result in a view of a number of stairs and little or no glimpse of the neighboring balustrade contents. A navigation adjustment based on height information would provide a view close to the terrain slope, broadening the user's view to include all the flowers ahead of the user. Additionally, the measurements in case of stair climbing yield slightly different values for the slope at a constant speed (depending on the number of stairs covered during a measurement cycle) which results in slightly different view orientation adjustments, eliminating stiffness in the view orientation behavior.

- Obstacles on high-positioned objects (for instance a balustrade on a mountain top) represent a way of automatically looking down and observing the scenery after collision and sustaining the collision.

- Floor, ceiling or wall examination with a fixed view orientation to which the avatar returns to after automatic navigation adjustment (use of height technique in flat environments), in this case the user doesn't have to change the tilting angle again and again for examination of multiple objects, textures or other artefacts.

Further extensions of our scheme could include other visual focus elements like change in lighting, movement/shadows, but also auditory cues. These would be of great importance and would represent a qualitative improvement in time-based, fast-paced interactive environments like 1st person games.

Technologically, our method is not (yet) feasible in VRML or X3D but achievable with custom code in other scenegraphs and also in game engines. However, it requires a qualitative change in navigation concepts and interfaces, and may require a bit of practice and accustoming time. Although [Hughes and Lewis 2000] suggests both enhanced experience with easier pathfinding and discomfort as the result of collision while orientation was adjusted, our method eliminates the negative experience cases of collision with view adjustment pointing out the collision object. A very simple, but limited method of testing our concepts is to simulate walkthroughs in virtual environments with separated position and orientation values, since in that case the navigational input from the user is disregarded. It is limited in the sense of pre-scripted paths, but it gives a taste of how such a navigational aid system would be received by humans.

References

BIER, E. A. 1986. Skitters and Jacks: Interactive 3D Positioning Tools. In *Proceedings 1986 ACM Workshop on Interactive 3D Graphics*, 183–196.

BOWMAN, D. A., AND HODGES, L. F. 1997. An Evaluation of Techniques for Grabbing and Manipulating Remote Objects in Immersive Virtual Environments. In *Proceedings of the 1997 Symposium on Interactive Computer Graphics*.

BOWMAN, D. A., KOLLER, D., AND HODGES, L. F. 1997. Travel in Immersive Virtual Environments: An Evaluation of Viewpoint Motion Control techniques. In *IEEE Proceedings of VRAIS'97*.

CASSELL, J., PELACHAUD, C., BADLER, N., STEEDMAN, M., ACHORN, B., BECKET, T., DOUVILLE, B., PREVOST, S., AND STONE, M. 1994. ANIMATED CONVERSATION: Rule-based Generation of Facial Expression, Gesture & Spoken Intonation for Multiple Conversational Agents. In *Computer Graphics (Siggraph '94 Proceedings)*, ACM Press, vol. 28, 413–420.

ÇAPIN, T. K., PANDZIC, I. S., MAGNENAT-THALMANN, N., AND THALMANN, D. 1997. A Dead-Reckoning Algorithm for Virtual Human Figures. In *VRAIS'97*, IEEE Press, 161–168.

GILLIES, M. F. P., AND DODGSON, N. A. 1999. Psychologically-based Walking in a Cluttered Environment. In *Eurographics Short Papers*.

HALPER, N., HELBING, R., AND STROTHOTTE, T. 2001. A Camera Engine for Computer Games: Managing the Trade-Off Between Constraint Satisfaction and Frame Coherence. In *CG Forum/Eurographics Proceedings*, A. Chalmers and T.-M. Rhyne, Eds., vol. 3 of *20*.

HUGHES, S., AND LEWIS, M. 2000. Attentive Camera Navigation in Virtual Environments. In *IEEE International Conference on Systems, Man & Cybernetics*.

HUGHES, S., BRUSILOVSKY, P., AND LEWIS, M. 2002. Adaptive Navigation Support in 3D E-Commerce activities. In *Workshop on Recommendation and Personalization in eCommerce at the 2nd International Conference on Adaptive Hypermedia and Adaptive Web Based Systems*.

LAVIOLA, J. J., FELIZ, D. A., KEEFE, D. F., AND ZELEZNIK, R. C. 2001. Hands-Free Multi-Scale Navigation in Virtual Environments. In *Proceedings of the 2001 Symposium on Interactive Computer Graphics*, ACM Press, 9–15.

NIELSEN, J. 1999. *Designing Web Usability*. New Riders.

NOSER, H., AND THALMANN, D. 1998. Towards Autonomous Synthetic Actors. In *Chapter 9 in Cyberworlds*, Springer, A. L. T. L. Kunii, Ed., 143–158.

OPEN SOURCE INITIATIVE. Open Scene Graph (OSG). http://www.openscenegraph.org.

OPEN SOURCE INITIATIVE. OpenSG scenegraph. http://www.opensg.org.

PAPE, D., CRUZ-NEIRA, C., AND CZERNUSZENKO, M. CAVE User's Guide. http://www.evl.uic.edu/pape/CAVE/prog/CAVEGuide.html.

SGI, INC. OpenInventor. http://www.sgi.com/software/inventor.

STEED, A. 1997. Efficient Navigation Around Complex Virtual Environments. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST)*, ACM Press, 173–180.

SUN MICROSYSTEMS, INC. Java3D(tm) API. http://java.sun.com/products/java-media/3D.

THALMANN, D., NOSER, H., AND HUANG, Z. 1997. Autonomous Virtual Actors based on Virtual Sensors. In *Creating Personalities for Synthetic Actors: Towards Autonomous Personality Agents*, Springer, vol. 1195 of *LNAI*, 25–42.

TU, X. 1999. *Artificial Animals for Computer Animation: Biomechanics, Locomotion, Perception, and Behavior*, vol. 1635 of *LNCS*. Springer.

USOH, M., ARTHUR, K., WHITTON, M. C., BASTOS, R., STEED, A., SLATER, M., AND FREDERICK P. BROOKS, J. 1999. Walking → Walking-in-Place → Flying, In Virtual Environments. In *Computer Graphics (Siggraph '99 Proceedings)*, ACM Press, 359–364.

WEB3D CONSORTIUM, 1997. VRML97: The Virtual Reality Modeling Language. http://www.web3d.org/technicalinfo/specifications/vrml97/.

WERNERT, E. A., AND HANSON, A. J. 1999. A Framework for Assisted Exploration with Collaboration. In *IEEE Visualization '99*, D. Ebert, M. Gross, and B. Hamann, Eds., 241–248.