Sliding Viewport for Interactive Virtual Environments

Andrei Sherstyuk
University of Hawaii
andreis@hawaii.edu

Dale Vincent
University of Hawaii
dvincent@hawaii.edu

Caroline Jay
University of Manchester
caroline.jay@manchester.ac.uk

Abstract

The ability to manipulate objects is fundamental to most virtual reality (VR) applications. A multitude of metaphors have been developed to facilitate object selection and manipulation [1], including virtual hand [2], which remains by far the most popular technique in this category.

The utility of the virtual hand depends on the user's ability to see it. Unfortunately, ensuring this is not always easy, especially in immersive systems which employ head mounted displays (HMD) with a limited field of view. Building on [3], we show how the virtual hand metaphor may be used effectively both to manipulate virtual objects, and to guide the user's view in order to ensure continuous visibility of the hand. We provide implementation detail and report experimental results from the user studies. This paper is intended for an audience interested in building HMD-based VR systems, where users are required to actively interact with the environment by using their virtual hand.

1. Introduction

The width of human field of view (FOV), for each eye, extends approximately $150°$ horizontally ($60°$ overlapping, $90°$ to side) and $135°$ vertically ($60°$ up, $75°$ down). Commercially available head mounted displays under $\$\,25\,K$ have a FOV ranging from 40 to 60 degrees diagonally. Examples include: *nVisorSX* and *V8* HMDs with $60°$, *ProViewXL* with $50°$, *eMagine* and *5DT* HMD with $40°$ [4]. It is a common knowledge that the loss of panoramic vision is one of the most noticeable differences between real and HMD-based simulated environments.

Restricted vision inevitably limits users' ability to interact with the environment [5]. In most situations, such limits are perceived as the system's deficiency, with a few exceptions. One example is the Biosimmer VR system, developed for training first responders in simulated hazardous conditions [6]. According to the authors, a $60°$ FOV V8 HMD adequately reproduces the experience that trainees have in real-life training, wearing a gas-mask and protective suits. A driving simulator [7], equipped with a $40°$ 5DT-800 HMD, provides another example of how limited FOV can be turned into an advantage. In this application, users' hands are intentionally kept out of view, so the students can learn how to operate car controls without looking. Since in both cases the developers explicitly explained how they used the low FOV values to their advantage, these two exceptions prove the rule that panoramic viewing generally benefits the application.

As VR systems penetrate new areas and new markets, the variety of configurations increases and the gamut of user tasks grows. In this work, we focus on a traditional VR architecture where users are immersed in the environment with a stereoscopic HMD; both the head and at least one hand are tracked with 6 degrees of freedom. The head tracking directly controls the location and direction of view of the virtual camera; the hand tracking allows to manipulate the virtual hand. Figure 1 illustrates a user immersed in such system. In this particular example, the head and both hands are tracked. Users don't have own avatars – their hands are the only visible parts of their virtual bodies.

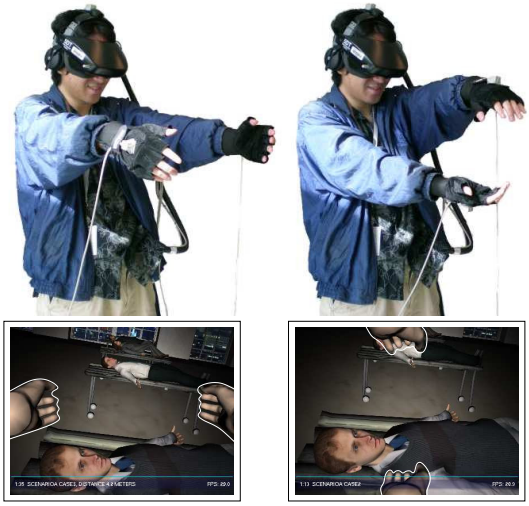

Figure 1. Immersive patient simulator. Equipment used: $40°$ FOV 5DT HMD, Flock-Of-Birds motion tracking system by Ascension. The lower images show the first person view, with the virtual hands at the borders of the screen space, contoured in white.

ICAT 2008
Dec. 1-3, Yokohama, Japan
ISSN: 1345-1278

1.1. The problem: where are my hands?

Figure 1 demonstrates how severe the limitations imposed by a narrow FOV on the use of virtual hands can be. The user holds his hands at the borders of the visible frame, showing the working volume where he can manipulate his hands without losing sight of them. Here, the user must examine and treat fifteen virtual patients using hand-held medical instruments.

After observing over a hundred users in this and similar VR settings we can confidently say that the most frequent question asked by novice users is, "where are my hands?" (Very few remarked on missing the rest of the body.) People seemed to be very dependent on being able to maintain constant visual contact with their hands, thus, most users made a conscious effort to coordinate head and hand movements to ensure that their hands were always in view.

As a result, users significantly restricted their hand movements by keeping them inside the narrow viewing cone, as shown in Figure 1, top. When reaching for objects located on far left or far right, they rotated the whole body, which looked very unnatural.

1.2. Hardware solution: a better HMD

One obvious way to extend the workspace is to use an HMD with a wider field of view. In recent years, the advances in microdisplay technologies spawned a number of new HMDs with panoramic characteristics: 150° *Wide5* by Fakespace Labs [8]; a *piSight* series with 82° to 180° FOV by Sensics. Long-standing products, such as *V8* from Visual Research Systems were also improved. Completely new display solutions were introduced [9]. Overall, after being nearly extinct in the last decade, panoramic HMDs are going through a renaissance phase.

However, upgrading to a panoramic HMD may not always be practical. A major reason is the cost. All the models mentioned above start at prices far above $\$\,25\,K$, and some go over into 6 figures. A second reason is the increased mass. As of this writing, all consumer-level panoramic HMDs weigh approximately 1 kg. This may not be a problem in applications where users mostly observe the scene, but if active body movements are expected from users, the inertia of the HMD may become noticeable and degrade user performance. Finally, in some HMD models, the multidisplay architecture requires precise positioning of the user's eyes in the focal point of the array of micro displays. Even small displacements of the HMD on the head, which are to be expected in fast-action VR applications, may cause the composite view to break into a set of disconnected sub-images. Thus, upgrading to a panoramic HMD may not always be a practical or cost-effective solution.

1.3. Software solution: a better interface

In this paper, we advocate a software solution to the limited visibility of the virtual hand. We base our approach on facts, known from neurobiology and the cognitive sciences, that the human eye is a highly sensitive perception device, with its movements being proactive, anticipating actions [10]. The studies show that the eyes are positioned at only task-relevant objects, which in our case are, (a) the virtual hand, (b) hand-held tools, (c) locations of tool applications. Thus, our goal is to ensure that the moving virtual hand remains in continuous view. Here, we achieve this by using a non-linear mapping between the relative positions and orientation of the user's head and hand and the virtual camera controls.

2. Related work

Several researchers explored non-isomorphic mapping between the orientation of the user's head and/or body and the virtual camera, by amplifying the original head rotation. Early experiments with amplification of head rotation were presented by Poupyrev *et al.* [11], where the authors augmented the video stream of a user's face with a superimposed model of a Kabuki mask. When amplified, the mask was turning faster than the face. Later, the authors developed a general non-isomorphic rotation mapping technique [12], with a detailed analysis of possible implementations for different UI purposes. In the original experiments with the rotating mask, a simple desktop monitor was used as a display device.

LaViola *et al.* [13] investigated how indirect camera mapping may be used in CAVE systems. Their goal was to turn a three wall based CAVE with 270° horizontal field of view into a completely closed environment with 360° viewing. Several mapping schemes were tried with mixed results, such as continuous direct scaling of head rotation, head and torso rotation, and a 2D-extended Gaussian surface as a mapping function of both the orientation and location of the user within the environment. The last approach yielded good results, whereas the direct amplification caused significant user discomfort. In a recent study, LaViola and Katzourin [14] showed how non-isomorphic rotation improved user performance by 15% in surround-screen virtual environments.

For HMD-based systems, encouraging results were reported by Jaekl *et al.* [15, 16], who investigated the human tolerance to errors between head motions and the visual display. Their findings show that changes in the orientation of head rotation did not influence the user's performance. Also, there was no apparent effect of the direction of gravity either [16]. The authors reported that given the choice of adjusting the HMD rotation response manually, participants tended to amplify the rotation, as it felt "more natu-

ral". These findings were supported by experimental studies by Jay and Hubbold [17], who developed an immersive environment with amplified head rotation and compared user performance with and without amplification. Amplifying head movements lead to a 21% improvement in a visual search task. This condition was also preferred by participants, who believed it to be the control condition. Dislike was expressed for the real control condition, where movements felt, "slowed down".

The results described above suggest that amplifying user head rotation is a promising technique for controlling cameras in immersive environments with limited display capabilities. However, in the general case, a rotation-based approach leads to problems.

Firstly, mappings that involve head-only rotation have limited use in systems where users are required to operate their hands. The location of the virtual hands also needs adjustment, otherwise users will not be able to see them [17]. In a general case of amplifying 3D head rotations, hands can not be adjusted with the same mapping, as the source rotations happen about different pivots and have different ranges, for the head and hands.

Secondly, as discussed in [12], amplifying an arbitrary rotation in 3D does not satisfy two important requirements, called the directional and nulling compliances. The directional compliance demands that the amplified rotation happens around the same axis as the source rotation. The nulling compliance means that returning the interface object (i.e., the head) into the initial position, must also cancel the amplified response. Non-conformance to these requirements severely violates generally accepted recommendations for user-interface design [18]. As of this writing, there are no published solutions describing how to circumvent this fundamental property of rotation, in the context of building camera controls in immersive VR applications.

3. Sliding viewport

We propose to improve the usability of existing HMD-based immersive VR systems and increase the *perceived* size of the field of view, by dynamically shifting all four corners of the viewport in the camera space. The amount and the direction of the shift is derived from the position of the virtual hand, moving freely in the environment.

We base our approach on an observation that the user's eyes always focus on task-relevant objects: a hammer hitting a nail, a virtual hand touching a patient's wrist for pulse readings, and so on [19, 20]. Thus, we set it as our goal ensuring that the moving hand does not reach the end of the viewable field.

3.1. The algorithm outline

The algorithm is illustrated in Figure 2. The virtual hand, initially located in the north-east corner of the viewport (drawn as a solid), is moving away from the viewable area (drawn as a contour). As shown on the diagram, the hand new location (x, y), projected onto the image plane, resides inside two slabs $[X_{start}, X_{stop}]$ and $[Y_{start}, Y_{stop}]$. Normalized values of x and y determine the amounts of horizontal and vertical increments, as returned by a mapping function $W(x, y)$. These increments are added to all four corners of the viewport. As a result, the hand becomes visible again (Figure 2, right side). Figuratively speaking, the viewport slides on a (x, y) plane within the fixed box of maximal values, as set by the mapping function. The view-sliding mechanism is applied at each cycle of the graphics loop, which ensures continuous adjustment of the viewport location in camera space.

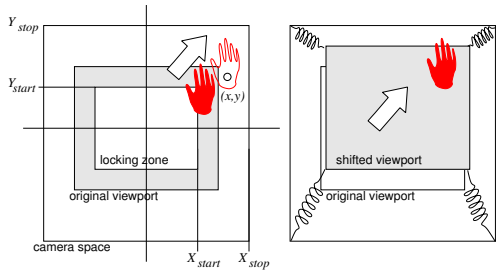

Figure 2. The sliding viewport algorithm computes new positions of the viewport corners, based upon the current values of azimuthal and elevation angles of the hand in camera space. As a result, the view slides towards the hand.

Note that the actual FOV does not change its horizontal nor vertical sizes during the process. We only help uses to peek "outside the frame" of the HMD by temporarily shifting that frame in the right direction at the right time.

It is also important to note that the viewport sliding is performed *in addition* to normal view controls due to head tracking. Our technique is intended to complement, rather than replace, normal tracking of the user's head.

3.2. Implementation

The viewport sliding mechanism does not need to be engaged permanently, but can be turned on and off in real time, as the application needs dictate. These needs are specific for each application and each context. However, certain situations are fairly common. For example, when the user is simply observing the environment, without active interaction, the viewport shifting should be switched off. Unless instructed otherwise, the UI system can make a good guess that the user is in a "sight-seeing" mode if his or her hands are staying out of view for a long time and are hanging down at full arm's length. When the user is actively interacting

with the environment, the virtual hands usually remain in view for a relatively long time (a few seconds), which can be used as a signal to activate the viewport sliding mechanism, or "lock" the view on the hand.

After experimenting with different heuristics for viewport locking, the following rules yielded useful results, with a good balance of effectiveness (the "just-in-time" feature) and unobtrusiveness.

- View lock is engaged when the hand stays within the area where $|x| \leq X_{start}$ and $|y| \leq Y_{start}$. longer than 2 seconds.
- Lock is engaged when a hand-held tool is being used, for example, a stethoscope, a roll of bandage, etc.
- Lock is broken when the hand stays outside of adjustment zone ($|x| \geq X_{stop}$ and $|y| \geq Y_{stop}$) longer than 1 second.
- Lock is broken when the user selects objects that have been pre-arranged to fit the original view: 2D menus, a tray with instruments. In our implementation, such objects are parented to the head object for easier access.
- Lock can only be applied to the dominant hand; that simple rule prevents multiple conflicts when both hands may start competing for the direction of the shift, for example by moving out of view in the opposite directions. Presently, none of the user tasks in our patient simulator requires bimanual manipulations: both hands may only be used in sequence. Thus, this rule fits our system quite naturally.

These rules are applied in each cycle of the simulation loop. At first, the lock is set to false and, while it remains false, all lock-enabling rules are executed. If, at some stage, the lock is enabled, all lock-breaking rules are checked. If the lock survives all the tests, the view adjusting values are calculated and applied.

We intentionally avoided using velocity-based rules for engaging and disengaging the view lock. For example, the lock could have been allowed only if the hand's speed does not exceed a certain point. The rationale for this rule is that users should be allowed to gesticulate quickly without tracking their hands visually. It seems unlikely to find "good" values for such speed limits' for hands that would feel natural for all users. On the contrary, rules solely based on position and visibility of the virtual hand are more general, easier to explain and less susceptible to user temperament and hand-manners.

The last two lock-breaking rules are specific to our application, as we have employed some 2D elements in the user interface, such as a multiple choice survey, operated by pointing the virtual hand. In a similar manner, virtual tools are accessed from a tool tray by pointing. In both cases, the locking is turned off as the objects of interest have been designed to stay in the center of unmodified view.

The new coordinates of the viewport in the image plane X and Y are computed as follows (for clarity, we show solutions only for the first quadrant of the screen space, where all angles are positive):

$$\begin{aligned} X &= X_0 + \Delta X(\mathbf{p}), \\ Y &= Y_0 + \Delta Y(\mathbf{p}), \\ \Delta X(\mathbf{p}) &= k\, W_x(a_x), \\ \Delta Y(\mathbf{p}) &= k\, W_y(a_y) \end{aligned} \tag{1}$$

X_0, Y_0	initial location of the upper right corner of the viewport
X, Y	new values, in screen 2D space
$\mathbf{p} = (x, y, z)$	hand position in camera 3D space
$a_x = \arctan(x/z)$	is the hand's azimuthal angle
$a_y = \arctan(y/z)$	is the hand's elevation angle
H_{start}	start angle for engaging view lock
H_{stop}	end angle after which tracking stops
k	a multiplier, useful range $1.5 - 1.7$

For the mapping function, we used a windowed and displaced W-shaped quartic $(1-s^2)^2$. For horizontal mapping,

$$W_x(s) = \begin{cases} 0, & a_x < H_{start}; \\ (1-s^2)^2, & H_{start} \leq a_x \leq H_{stop}; \\ 1, & a_x > H_{stop}; \end{cases} \tag{2}$$

with its argument s computed as the normalized hand's position between the start and end values: $s = (a_x - H_{start})/(H_{stop} - H_{start})$. Vertical mapping W_y is obtained by replacing azimuthal angle a_x and its bounds H_{start}, H_{stop} in Eq. 2 by elevation angle a_y and its bounds V_{start}, V_{stop}. In our system, the upper right corner (X_0, Y_0, Z_0) is set as $(1.33, 1, 3)$, with other points being symmetrical around the X and Y axes. For these settings, the following values were found to be useful:

$$\begin{aligned} &H_{start} = 17^\circ, H_{stop} = 50^\circ, \\ &V_{start} = 12^\circ, V_{stop} = 40^\circ \text{ up, } 60^\circ \text{ down} \end{aligned} \tag{3}$$

The viewport begins to follow the hand when the hand azimuthal angle reaches 17° and continues until 50°, on either side. Vertically, the tracking is set to be stronger in the lower part of the screen space, because in our application users mostly operate their hands while holding them at waist level or lower. When the tracking lock is engaged, the hand remains in the same position on the screen (typically, near the edge) and the view moves in the direction of the hand movement. From the user's perspective, it appears as if the eyes are following the moving hand, which is an illusion – the eyes remain fixed on the same screen location where the hand is displayed.

3.3. Summary of our contribution

The main difference between our approach and the previous work is that we use a 2D solution for an intrinsically 2D problem (narrow FOV), while other researchers explored 3D algorithms for the same problem. By switching from the 3D scene-graph space to a 2D screen space, we gain the following advantages:

- *Compatibility with other techniques.*
  The view sliding technique will work in the presence of other UI enhancing algorithms, such as Go-Go arm stretching [21], because view sliding does not alter the content of the scene-graph: coordinates, transformation matrices, etc.
- *Head/hand coordination.*
  This problem was reported by Jay and Hubbold [17] and is naturally addressed by the design of the view sliding mechanism, based on the location of the virtual hand.
- *Nulling and directional compliances.*
  Substituting translations in 2D for rotations in 3D preserves both compliances in the system responses, which makes this approach attractive for UI design purposes.
- *Convenient parameter space.*
  Using the starting and ending values of horizontal and vertical angles H and V is a convenient way of specifying the extents of the hand-tracking zones, when configuring the system for a new HMD or tuning up the existing configuration. These angles are usually listed in the HMD reference charts and can be trivially converted to screen coordinates X and Y, shown in Figure 2. Also, this parameter space closely matches the conventional description of the human visual field. Therefore, the view sliding can easily accommodate the asymmetric nature of human vision, by setting H and V values separately for each viewing quadrant.

4. User evaluation

The viewport sliding technique was tested in pilot trials illustrated in Figure 3 and appeared very promising. To check whether it will work in real applications, we conducted an experimental study within the context of an existing virtual patient simulator.

The experiments were organized as a between-subjects study: half of the group had the view sliding mechanism enabled, while the rest used traditional tracking-only view controls. The goal of the experiments was to collect and compare objective and subjective evidence on how the view sliding technique affected participant performance in operating their virtual hands.

Figure 3. Pilot trials. Two composite images show the original (left) and enhanced (right) zones of hand visibility. Here, the user was asked to point at the edges and corners of his view, without turning his head.

4.1. The participants

The total of 28 volunteers participated in the study, recruited among medical students and staff members. Most participants were in their twenties, all had normal or corrected-to-normal vision, none had previous experiences with immersive VR. All participants successfully completed the exercise, with the exception of one person, who complained about feeling claustrophobic early in the session.

Figure 4. The experimental study. Top: the user is picking the watch tool from the tray. Bottom: checking the pulse at the neck (left) and wrist (right). Note that only one of the two locations can be seen at any given time, forcing the user to either turn physically, slide the view by moving his hand or do combination of both.

4.2. The mission

The user mission was based upon scenarios, developed for teaching mass casualty triage in VR. Each participant was asked to perform a medical examination of 10 virtual patients, one person at a time, followed by optional treatment for visible injuries. The whole exercise lasted between 10 and 15 minutes, performed in a single session.

During the examination, participants had to check patients' vital signs using medical instruments, as shown on Figure 4. The instruments were selected by pointing and operated by placing them at touch-sensitive zones on the patients' bodies: the neck and both wrists for checking pulse rate, upper arms for measuring blood pressure, left and right chest sides for examining patients breathing, etc. All 10 patients were placed at arm length from the participants, about 1 meter above the floor level (Figure 4). To prevent the participants from developing patterns in their movements, the virtual patients were oriented in random directions and put in different poses. Thus, to check the pulse at the neck and both wrists, required different movements for each new patient.

4.3. The VR system

During the experiments, the subjects were fully immersed into the environment, using a 40° FOV HMD. The head and both hands were tracked in 6 DOF by Flock of Birds magnetic motion tracking system by Ascension Corp, running in extended range mode with 9 foot tracking radius. The scene was rendered at fixed rate of 25 fps by custom-made system derived from the Flatland project [22].

As soon as the participant picked up an instrument from the tray, the view locking mechanism was engaged. At that moment, the virtual hand was still inside the normal view, because the tray was designed to fit the original 40° FOV frustrum. This guaranteed that mapping function 2 returned 0 values for shift increments, upon engaging the lock. The hand tracking continued until two conditions remained true: (1) an instrument was still in use and (2) the hand remained within the tracked range, given by equations 3.

4.4. The procedure

After putting on an HMD and gloves with motion sensors, each participant went through a quick calibration sequence, adjusting his or her hand positions and body height in VR. Then, the participant spent a few minutes with one virtual patient, practicing tool access and vital signs checking. Participants with the sliding view enabled did not receive any special training on hand/view coordination – they were allowed to learn it themselves.

4.5. The outcomes: subjective user evaluation

After completing the mission, participants were asked to fill out a short survey, which included the following questions.

Question 1. For this exercise, my ability to keep my hand in view was very important:

1. strongly disagree
2. disagree
3. neutral
4. agree
5. strongly agree

Question 2. In this experiment, the visible area for hand manipulations was:

1. much too small
2. somewhat too small
3. just right
4. somewhat more than needed
5. much more than needed

Question 3. The way the view followed my hand felt natural and helpful (offered to participants with view sliding enabled):

1. strongly disagree
2. disagree
3. neutral
4. agree
5. strongly agree

Their answers are summarized in the table below.

User group/ question	Median score	Mean score	SD	95% confidence interval
Normal view				
Q 1	4 (agree)	3.84	1.40	3.00 to 4.69
Q 2	3 (just right)	2.46	0.88	1.93 to 2.99 ($\star$)
Sliding view				
Q 1	4 (agree)	3.93	1.14	3.27 to 4.59
Q 2	3 (just right)	2.79	0.58	2.45 to 3.12 ($\star\star$)
Q 3	4 (agree)	4.00	1.13	3.28 to 4.72

Table 1. Summary of participants evaluation reports.

Both groups agreed on importance of having the hand in view. This result validates our assumptions, discussed in section 3.

For the second question on the size of the visible area for hand manipulation, both groups came up with median 3 answer (just right). Interestingly, the confident interval given by normal viewers dipped below 2 (much too small), marked with ($\star$), while sliding viewers started at 2.45 value ($\star\star$). However, the difference between the two groups was not statistically significant. Apparently, the view sliding technique alone could not turn a 40° HMD into a display that feels "just right" for all participants.

The answers to the third questions, indicating the sliding view was helpful and natural, are most encouraging. Evidently, participants were able to notice and make good use of the new technique intuitively, without special training.

4.6. The outcomes: objective evidence

In this study, VR software kept detailed logs of user activities and system responses. For each user, we captured aggregate times when the view was shifted. It appeared that, on average, participants spent 25% of their total time in VR with the view shifted towards their hands. More details are given in the table.

View shift	Mean time, %	SD
all directions	25.42	1.44
down	17.99	2.03
up	2.444	0.85
left	2.329	1.20
right	2.648	0.59

Table 2. Time spent in shifted view, as percentage of total time.

As the table shows, the view was most often sliding downwards, which was expected, because all virtual patients were placed low (see Figure 4). Similar times for all the other directions confirm that the areas of tool application were sufficiently randomized.

The most interesting results came from the analysis of how virtual tools were used over time. During each session, the system logged all user actions, time-stamping them with the frequency of the graphics loop. This made it possible to capture the time intervals elapsed from the moment a tool was picked up until the moment it was applied, by touching the areas on the virtual patient. We call these intervals "tool seek time" and associate them with the user ability to locate and access random places in the working area. In particular, we were interested in the seek time for the watch, which was used for checking pulse. As was explained before, this task involved active search and was not easy.

Figure 5 shows how the seek time for the watch tool changed over time, for both groups of participants. The seek times were collected and averaged for each participant, with the time step of 60 seconds. Then, linear models

$$y_i = a + bx_i + \epsilon$$

were fit to the resulting sample points, using the least squares regression. Significance probabilities P, for hypothesis $b \neq 1$, were found to be 0.886 for the control group and 0.00173 for our method. In other words, the control group didn't improve their seek time at all (Figure 5, top). The participants with the sliding viewport mechanism enabled showed a steady decrease from 11.3 seconds to 5.56 seconds. That result clearly demonstrates that our technique did help the participants to operate their virtual hands.

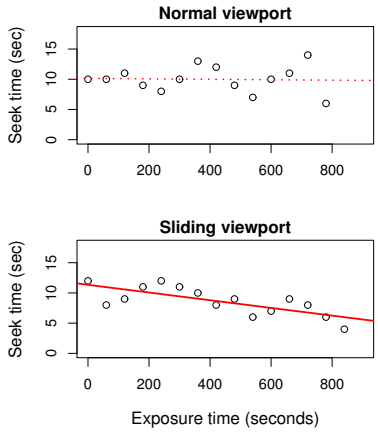

Figure 5. Changes in seek time for the watch tool. Top: the control group showed no improvement. Bottom: seek times for the participant with the sliding viewport significantly reduced. Slope P-values for linear models are 0.886 (top) and 0.00173 (bottom).

5. Extensions and future work

Initially intended for immersive VR systems, the view sliding technique works in desktop configurations as well, and allows control of both the hand and the camera with a single input device. Figure 6 shows a mosaic of views taken from a laptop-based system, where both the camera and the virtual hand are operated with a mouse. In this configuration, hand-assisted viewing significantly reduces clutching, a forced redirection of the mouse input from the camera to the virtual hand and back.

So far, we have explored only one of many possible applications of the viewpoint sliding technique, namely, direct handling of virtual objects that are in close proximity of the user. One important task that may benefit from our technique, is travel.

In many VR systems, travel is implemented by steering, i.e., continuously specifying the direction of movement with a pointing device [1]. In cases when the direction is set by pointing with a virtual hand, assisted hand visibility will give travelers additional advantages. Firstly, users will be able to navigate more confidently, with their steering hands not confined into a small cone of visible directions, as shown in Figure 3, left. Secondly, visibility of the steering object may affect the cruising speed. For example, if the steering hand remains in view long enough, the speed increases, up to a certain limit. Conversely, when the hand moves out of view, the speed drops to a lower value. If this happens accidentally (which is easy to imagine in systems with a narrow FOV), such unwanted interventions from the navigation speed control may be very frustrating. Thus, the sliding viewport technique may significantly improve the utility of hand-based steering travel systems.

Target-based travel could also benefit from the assisted visibility of the hand. An example of such a travel-intensive scene is shown in Figure 6, where users are required to move around large areas. Every object on the scene is a travel target, including umbrellas that appear only in the right-shifted view. Although we have not yet conducted formal evaluations into whether sliding viewports help users navigate better, early results are very encouraging.

Figure 6. A mosaic of sliding viewports in a desktop system. The central tile shows the original view, the hand resting in the inactive area of the screen. All other tiles show shifted views, driven by the virtual hand. The camera remains static in all tiles.

6. Conclusion

We presented a new technique for the automatic sliding of the viewport, which effectively quadruples the working area of the virtual hand, as illustrated in Figure 3. The techniques works both in immersive and desktop systems.

The view sliding mechanism is easy to implement and tune up for any HMD model, using their native FOV values. The amount of the maximal shift can be conveniently specified in terms of desired angles of vision, that can be set up separately for each eye and each direction of view.

We have demonstrated experimentally that our technique works well in immersive VR applications, where users actively interact with the environment with their virtual hands.

References

[1] D. Bowman, E. Kruijff, J. LaViola, I. Poupyrev. *3D user interfaces: Theory and practice*, Addison-Wesley, 2004. 1, 7

[2] R. Jacoby, M. Ferneau, J. Humphries. Gestural Interaction in a Virtual Environment. *Stereoscopic Displays and Virtual Reality Systems*, SPIE 2177, pp. 355-364, 1994. 1

[3] A. Sherstyuk, D. Vincent, C. Jay. Sliding Viewport for Head Mounted Displays in Interactive Environments. *Proceedings of IEEE Symposium on 3D User Interfaces*, pp. 135-136, March 2008. 1

[4] Christoph Bungert. HMD Comparison Chart, last updated on Dec. 27, 2006. http://www.stereo3d.com/hmd.htm. 1

[5] D. Neale. Head-Mounted Displays: Product Reviews and Related Design Considerations, *Hypermedia Technical Report HCIL-98-02*, Human-Computer Interaction Laboratory, Virginia Tech 1998. 1

[6] S. Stansfield , D. Shawver, A. Sobel, M. Prasad and L. Tapia. Design and Implementation of a Virtual Reality System and Its Application to Training Medical First Responders, *Presence: Teleoperators and Virtual Environments*, MIT Press Journals, Vol. 9, No. 6, Dec. 2000. 1

[7] The 5DT Driving Simulator Series, last updated on Jul 19, 2007. http://www.5dt.com/products/pdrivertraining.html 1

[8] M. Bolas, J. Pair, K. Haynes and I. McDowall. Display Research at the University of Southern California. *IEEE Emerging Displays Workshop*, Alexandria, VA, March 2006. 2

[9] K. Kiyokawa. A Wide Field-of-view Head Mounted Projective Display using Hyperbolic Half-silvered Mirrors. *Proceedings of ISMAR'07*, Nara, Japan, November 2007. 2

[10] M. Land. Eye movements in daily life. In L. Calupa, & J. Werner (Eds.), *The Visual Neurosciences* vol. 2, pp. 1357-1368, New York: MIT Press, 2004. 2

[11] I. Poupyrev, T. Otsuka, S. Weghorst, T. Ichikawa. Amplifying rotations in 3D interfaces. *Proceedings of CHI 1999*, pp. 256-257, 1999. 2

[12] I. Poupyrev, S. Weghorst, S. Fels. Non-isomorphic 3D Rotational Techniques. *Proceedings of CHI'2000*, pp. 546-547, 2000. 2, 3

[13] J. LaViola Jr., D. Feliz, D. Keefe, R. Zeleznik. Hands-Free Multi-Scale Navigation in Virtual Environments. *In proceedings of ACM SIGGRAPH I3D 2001 Symposium on Interactive 3D Graphics*. North Carolina, March 2001. 2

[14] J. LaViola and M. Katzourin. An Exploration of Non-Isomorphic 3D Rotation in Surround Screen Virtual Environments, *Proceedings of the IEEE Symposium on 3D User Interfaces*, pp. 49-54, March 2007. 2

[15] P. Jaekl, R. Allison, L. Harris, U. Jasiobedzka, H. Jenkin, M. Jenkin, J. Zacher, D. Zikovitz. Perceptual stability during head movement in virtual reality. *Proceedings IEEE VR'02*, pp. 149-155, 2002. 2

[16] P. Jaekl, M. Jenkin, and L. R. Harrisa. Perceptual stability during active head movements orthogonal and parallel to gravity. *Journal of Vestibular Research*, 13, pp. 265271, 2003. 2

[17] C. Jay and R. J. Hubbold. Amplifying head movements with head-mounted displays. Presence, MIT Press, 12:268-276, June 2003. 3, 5

[18] E. Britton, J. Lipscomb, M. Pique. Making nested rotations convenient for the user. *Proceedings of SIGGRAPH'78*, pp. 222-227, 1978. 3

[19] B. Biguer, M. Jeannerod, P. Prablanc. The coordination of eye, head and arm movements during reaching at a single visual target. *Experimental Brain Research*, 46, pp. 301-304, 1982. 3

[20] D. Rosenbaum. *Human Motor Control*. New York: Academic. Chapter 8, pp. 253-264, 1991. 3

[21] I. Poupyrev, M. Billinghurst, S. Weghorst and T. Ichikawa. The Go-Go Interaction Technique: Non-Linear Mapping for Direct Manipulation in VR, *ACM Symposium on User Interface Software and Technology*, pp. 79-80, 1996. 5

[22] Flatland Project, http://www.hpc.unm.edu/homunculus/ 6