Vis Comput (2011) 27: 173–185
DOI 10.1007/s00371-010-0516-0

ORIGINAL ARTICLE

Impact of hand-assisted viewing on user performance and learning patterns in virtual environments

Andrei Sherstyuk · Caroline Jay · Anton Treskunov

Published online: 22 July 2010

Abstract The ability to locate, select and interact with objects is fundamental to most Virtual Reality (VR) applications. Recently, it was demonstrated that the virtual hand metaphor, a technique commonly used for these tasks, can also be employed to control the virtual camera, resulting in improved performance and user evaluation in visual search tasks.

In this work, we further investigate the effects of hand-assisted viewing on user behavior in immersive virtual environments. We demonstrate that hand-assisted camera control significantly changes the way how people operate their virtual hands, on motor, cognitive, and behavioral levels.

Keywords View sliding · Non-isomorphic camera control · User study · VR · Learning patterns · Variance in learning curves

1 Introduction

In immersive VR systems, a direct capture of user's hands with motion tracking equipment is the most common technique for controlling the virtual hand in VR; the same is true for the virtual camera [4]. In conventional VR implementations, the hands and eyes are assumed to act independently.

A view-sliding technique, introduced by Sherstyuk, Vincent and Jay [22], binds these two input streams together, providing dynamic, task-oriented camera controls. The view-sliding mechanism proactively modifies the location of the user viewport on the image plane, shifting it towards the current position of the virtual hand. As a result, the hand remains inside the viewable area during active use, even for display devices with a narrow field of view. This creates an illusion that the display device, for example, a head-mounted display (HMD), has a larger field of view than it has in reality. The new technique received positive user evaluations in experimental studies. There were indications that users actually perceived a wider field of view, induced by view sliding. Also, it has been shown that view sliding allows users to achieve better scores in operating their virtual hands, using a task-dependent performance metric.

In this work, we continue to explore the effects of the view-sliding technique on user performance in VR. We introduce several metrics related to the virtual hand metaphor, in order to evaluate user performance on a motor and cognitive levels. We present experimental evidence that there exists a correlation between the use of the new technique and the shape and even the direction of the learning curve for tasks that involve active use of virtual hand.

In the first sections, we present the essential material on the view-sliding technique, including motivation, literature review of the related work and the technical implementation, from [22]. The new method of analysis of the experimental data, new evaluation metrics and the results of the analysis are presented in Sects. 7–8. This material constitutes our main contribution to the subject.

A. Sherstyuk (✉)
Avatar-Reality, Inc., 55 Merchant Street, Ste 1700, Honolulu, HI 96813, USA
e-mail: andrei@avatar-reality.com

C. Jay
Human Centered Web Lab School of Computer Science, LF 1, Kilburn Building, University of Manchester, Oxford Road, Manchester, M13 9PL, UK
e-mail: caroline.jay@manchester.ac.uk

A. Treskunov
Samsung Information Systems America, 75 West Plumeria Drive, San Jose, CA 95134, USA
e-mail: anton.t@sisa.samsung.com

2 Motivation: need for a wider view

The utility of the virtual hand often depends on the user's ability to see it. Ensuring this is not an easy task in immersive systems which employ HMDs with limited field of view (FOV), typically ranging from 40 to 60 degrees diagonally. By contrast, the width of human field of view, for each eye, extends approximately 150° horizontally (60° overlapping, 90° to side) and 135° vertically (60° up, 75° down).

Figure 1 demonstrates how severe the limitations imposed by a narrow FOV on the use of virtual hands can be. The two users hold their hands at the borders of the visible frame, showing the working volume where they can manipulate their hands without losing sight of them. For curiosity's sake, the readers are welcome to try this simple test with their own hands, and then compare their poses against the photographs in Fig. 1. The results will be nothing less of surprising.

2.1 The problem: where are my hands?

Direct observations in previous experimental studies [21, 22, 24] suggest that most people feel uncomfortable when they lose sight of their hands in VR. After seeing over a hundred people in various VR settings, we assert that most users make a conscious effort to coordinate head and hand movements to ensure that their hands remain in view. As a result, they significantly restrict their hand movements by keeping them inside the narrow viewing zone. When reaching for objects located on far left or far right, users tend to rotate the whole body, which looks very unnatural. When reaching down, they often tilt the head forward in a very inconvenient manner, as shown in Fig. 1, bottom right.

In a survey conducted by Boger [2], it is reported that the commonplace horizontal field of view (50° or lower) and the commonplace vertical field of view (30° or lower) are considered "good enough" by fewer than 10% of surveyed population. For a 4:3 aspect ratio display configura-

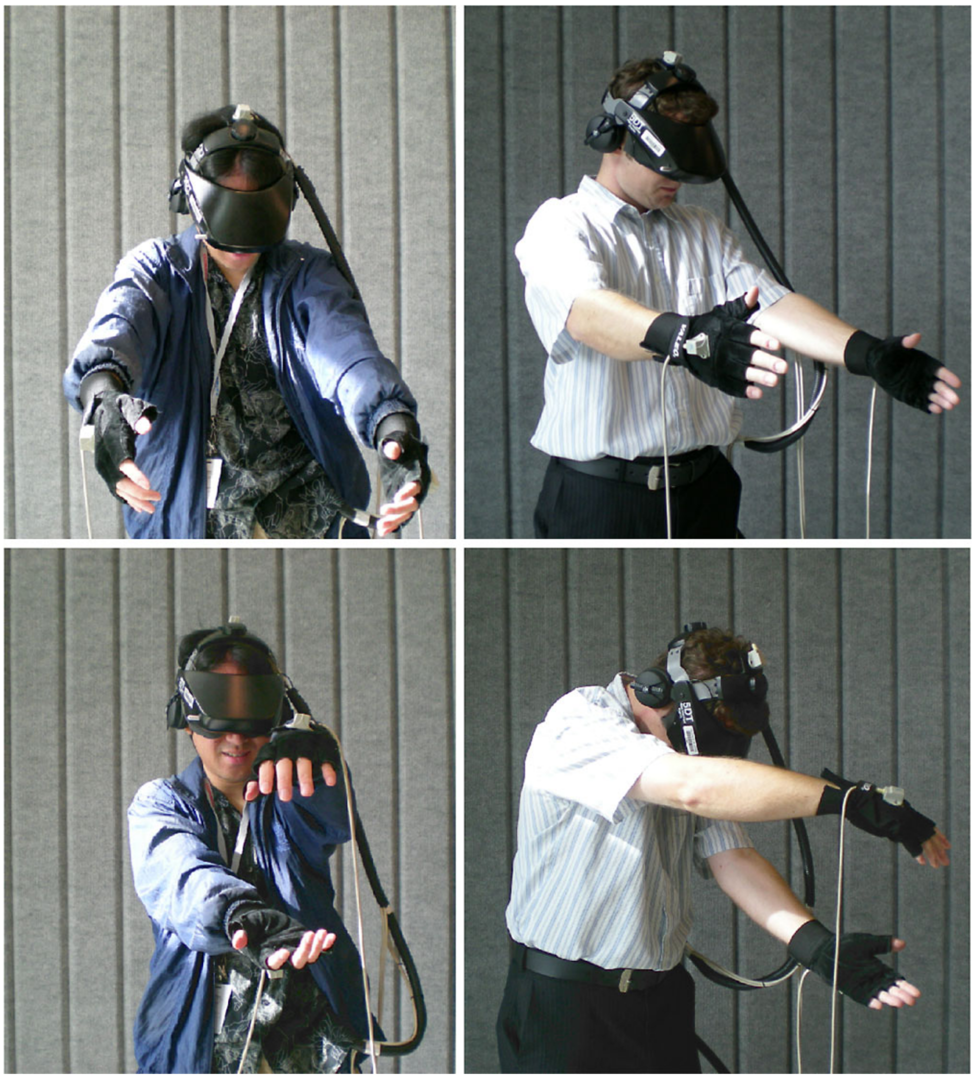
Fig. 1 VR users demonstrate areas of hand visibility for a 40° HMD: horizontal (*top*) and vertical (*bottom*). Both hands and head are tracked in 6 degrees-of-freedom. In order to see their hands, users sometimes have to assume very uncomfortable poses (*bottom right*)

tion, these values correspond to diagonal FOV of approximately 60 and 50 degrees, typical for many commercially available HMDs [6]. It has also been shown that, in the general case, restricted vision inevitably limits users' ability to interact with the environment [17]. This brings up the question: How can one expand the hardware-limited field of view in immersive VR applications? In our specific case: How can we ensure that the virtual hand does not leave the viewing area of the HMD?

2.2 Hardware solution: a better HMD

One obvious way to extend the workspace for hand manipulation in VR is to use an HMD with a wider field of view. In recent years, the advances in microdisplay technologies spawned a number of models with panoramic or nearly-panoramic viewing fields: 150° *Wide5* HMD by Fakespace Labs [3], a *piSight* HMD series with 82° to 180° FOV by Sensics Inc. Completely new display architectures were introduced, such as hyperbolic mirror by Kiyokawa [11]. 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 six figures. A second reason is the increased mass. As of writing, all consumer-level panoramic HMDs weigh approximately 1 kg or more. This may not be a problem in applications where users mostly observe the scene, but if active body movements are expected from the users, the inertia of the HMD may become noticeable and degrade the performance. Thus, at present, upgrading to a panoramic HMD may not always be a practical or cost-effective solution.

2.3 Software solution: a better interface

We applied a software solution to the problem of limited visibility of the virtual hand, by improving the control–response loop between the user's head motions and the virtual camera parameters. Instead of a direct transfer of user's head rotation to the virtual camera, we employ a non-isomorphic mapping, which also accounts for the position of user's hand. First, we briefly describe the previous work in that field.

3 Previous work on nonlinear camera control

Several researchers have explored non-isomorphic mappings between the orientation of the user's head and/or body and the virtual camera. The basic method used for this purpose is an amplified transfer function from the user's rotational head motion to the virtual camera orientation. The core idea behind this technique is to minimize user efforts in rotating the virtual camera, by setting its angular velocity to be larger than that of the user's actual head motion. Poupyrev, Weghorst and Fels [19] presented a complete framework for processing non-isomorphic rotation mapping, with a detailed analysis of its possible implementations for general 3D UI purposes, including camera controls. Hinckley et al. [13] compared usability of various input devices in orientation-matching tasks.

LaViola et al. [14] investigated how modified camera rotation may be used in CAVE systems. Their goal was to turn a three wall CAVE with 270° horizontal field of view into a completely closed environment with 360° viewing. Several mapping schemes were tried and compared, such as direct scaling of head rotation, head and torso rotation, and more elaborate approaches. In a more recent study, LaViola and Katzourin showed that non-isomorphic rotation improved user performance by 15% in surround-screen virtual environments [16]. In the follow-up experimental study, LaViola et al. [15] explored how viewing conditions and display technology affect the user performance in rotating tasks.

For HMD-based systems, encouraging results were reported by Jaekl et al. [8] 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 [9]. The authors reported that given the choice of adjusting the HMD rotation response manually, participants tended to amplify the rotation, as it felt "more natural." These findings were supported by experimental studies by Jay and Hubbold [10] who developed an immersive environment with amplified head rotation and compared user performance with and without amplification. Amplifying head movements led 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 without amplification, where movements felt "slowed down."

The results described above suggest that amplifying user head rotation is a promising technique for controlling cameras in VR environments with limited display capabilities. This technique appeared effective for certain types of tasks and environments, such as visual search in scenes where interaction with objects is limited. However, for other types of tasks and conditions, a rotation-based approach to camera controls may lead 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. For one-dimensional rotations, e.g. about the vertical Y-axis only, the hands and head rotation may be amplified by the same mapping scheme as discussed by Jay and Hubbold [10]. However, in a general case of arbitrary 3D rotations, the hands cannot be adjusted in the same way simply because they rotate and move independently from the head.

Secondly, as discussed by Poupyrev et al. [19], 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 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, discussed by Britton, Lipscomb and Pique [5]. As of this writing, there are no published solutions describing how to circumvent this fundamental property of 3D rotation, in the context of building usable camera controls for immersive VR applications.

4 Sliding viewport

We proposed to improve the usability of existing HMD-based 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 are derived from the position of the virtual hand, moving freely in the environment.

We base our approach on the fact that the human eye is a highly sensitive perception device, with its movements being proactive, anticipating actions [12]. Studies show that the eyes are generally positioned at only task-relevant objects, such as the virtual hand, hand-held tools and locations of tool applications (see, for example, Biguer, Jeannerod and Prablanc [1] and Rosenbaum [20]). Thus, our goal is to ensure that the moving virtual hand remains in continuous view.

4.1 The algorithm implementation

The algorithm is illustrated in Fig. 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). The hand's new location (x, y), projected onto the image plane, resides inside two slabs, $[X_{\text{start}}, X_{\text{stop}}]$ and $[Y_{\text{start}}, Y_{\text{stop}}]$. Normalized values of x and y determine the amounts of horizontal and vertical view shifts, calculated by a mapping function $W(x, y)$. In a new shifted view, the hand becomes visible again (Fig. 2, right). Figuratively speaking, the viewport slides on a (x, y) plane within the fixed box of maximal values X_{stop} and Y_{stop}.

The view sliding is performed at the frequency of the graphics loop, computing horizontal and vertical view displacements ΔX and ΔY as described below. For clarity, we show solutions only for the first quadrant of the screen space, where all angles are positive.

$$\begin{aligned} X &= X_0 + \Delta X(p), \\ Y &= Y_0 + \Delta Y(p), \\ \Delta X(p) &= k\ W_h(a_h), \\ \Delta Y(p) &= k\ W_v(a_v), \end{aligned} \tag{1}$$

X_0, Y_0	initial location of the upper right corner of the viewport;
X, Y	new shifted values;
$p = (x, y, z)$	hand position in camera space;
$a_x = \arctan(x/z)$	the hand's azimuthal angle;
$a_y = \arctan(y/z)$	the hand's elevation angle;
$H_{\text{start}}, H_{\text{stop}}$	horizontal hand tracking range (e.g., 17°/50°);
$V_{\text{start}}, V_{\text{stop}}$	vertical hand tracking range (e.g., 12°/40°);
k	a multiplier, useful range 1.5–1.7.

The k-factor explicitly controls the sliding speed. The range of useful values was obtained experimentally during system development. In the user study, described in Sect. 5, we set

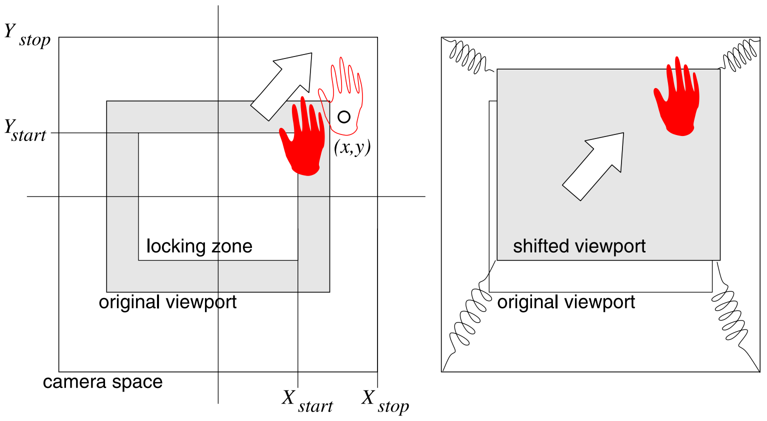

Fig. 2 The sliding viewport algorithm computes new positions of the viewport, based upon the current position of the hand in camera space. As a result, the view slides towards the hand, ensuring its continuous visibility

$k = 1.5$ for all participants. In more elaborate settings, this parameter can be adjusted for each user individually, during system calibration.

A windowed W-shaped quartic $(1 - a^2)^2$ was used as a mapping function. The shape of this curve, which is basically a smooth step-function, ensures smooth viewport adjustment over distance, as it has zero derivatives at 0 and 1 locations. For horizontal mapping, we have

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

The argument s for horizontal mapping function is computed as the normalized horizontal hand's positions between the start and end values:

$$s = (a_x - H_{\text{start}})/(H_{\text{stop}} - H_{\text{start}}). \tag{3}$$

Note that the actual FOV of the display device (i.e., the HMD) does not change its horizontal nor vertical sizes during the process. We only help users 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 conventional view controls due to head tracking. Our technique complements, rather than replaces, the normal tracking of the user's head.

4.2 View locking rules

The viewport sliding mechanism does not need to be engaged permanently. Instead, it should 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 scene, without active interaction, the viewport shifting should be switched off. The VR system can make a good guess that the user is in a "sightseeing" 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. In a similar fashion, the four-wheel drive in many vehicles is engaged only when needed in real time, without direct intervention from the driver.

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 $|x| \le X_{\text{start}}$ and $|y| \le Y_{\text{start}}$ longer than 2 seconds ("locking zone" in Fig. 2, left, shaded in light-gray).
- Lock is engaged when a hand-held tool is being used.
- Lock is broken when the hand stays outside of adjustment zone ($|x| \ge X_{\text{stop}}$ and $|y| \ge Y_{\text{stop}}$) longer than 1 second.
- Lock is broken when the user works with UI elements that have been prearranged to fit the view, such as 2D menus operated by pointing or touching.
- Lock can only be applied to the dominant hand; this 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.

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, using (1), (2) and (3).

During preliminary experiments with the system, we tested other conditions for engaging and disengaging the view lock. Specifically, we tried using speed-based rules. For example, the view was allowed to be locked only if the user hand's speed did not exceed a certain limit at this moment. We rejected this rule because it seemed unlikely to find "good" values for such speed limit that would feel natural to all users. On the contrary, rules solely based on position and visibility of the virtual hand are more general, easier to understand and less susceptible to variations in user temperament and hand-manners.

4.3 Summary of the viewport sliding algorithm

The main difference between our approach and the previous work is that we use a 2D solution for an intrinsically 2D problem (narrow horizontal and vertical values of FOV on the image plane), 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 3D UI enhancing algorithms, such as Go-Go arm stretching [18], because view sliding does not alter the content of the scene-graph: coordinates, transformation matrices, etc.
- *Hand–eye coordination.*
  This problem was reported by Jay and Hubbold [10] 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 3D UI design.

- *Convenient parameter space*.
  Using the starting and ending values of horizontal and vertical angles H and V provides a convenient way of specifying the extents of the hand-tracking zones. These angles are usually listed in the reference charts for most HMD models. Also, this parameter space closely matches the conventional description of the human visual field. Thus, the view sliding can easily accommodate the asymmetric nature of human vision, by setting H and V values separately for each viewing quadrant.

5 Experimental study

The viewport sliding technique was tested in pilot trials illustrated in Fig. 3 and appeared very promising. To check whether it will work in real applications, we conducted an experimental study using our VR medical 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 participants' performance in operating their virtual hands.

5.1 The participants

A 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, and none had previous experiences with VR. All participants successfully completed the exercise, with the exception of one person, who complained about feeling claustrophobic early in the session.

5.2 The mission

The user mission was based upon scenarios developed for teaching mass casualty triage in immersive VR by Vincent et al. [24]. Each participant was asked to perform a medical examination of 10 life-size virtual patients, one patient at a time. The whole exercise lasted between 10 and 15 minutes, performed in a single session.

During the examination, participants had to check the patients' vital signs using medical instruments, as shown in Fig. 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, upper arms for measuring blood pressure, etc. All 10 patients were placed at arm's length from the participants, about 1 meter above the floor level (Fig. 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.

5.3 The equipment

During the mission, the participants were fully immersed into the environment, using a stereoscopic 40° FOV HMD.

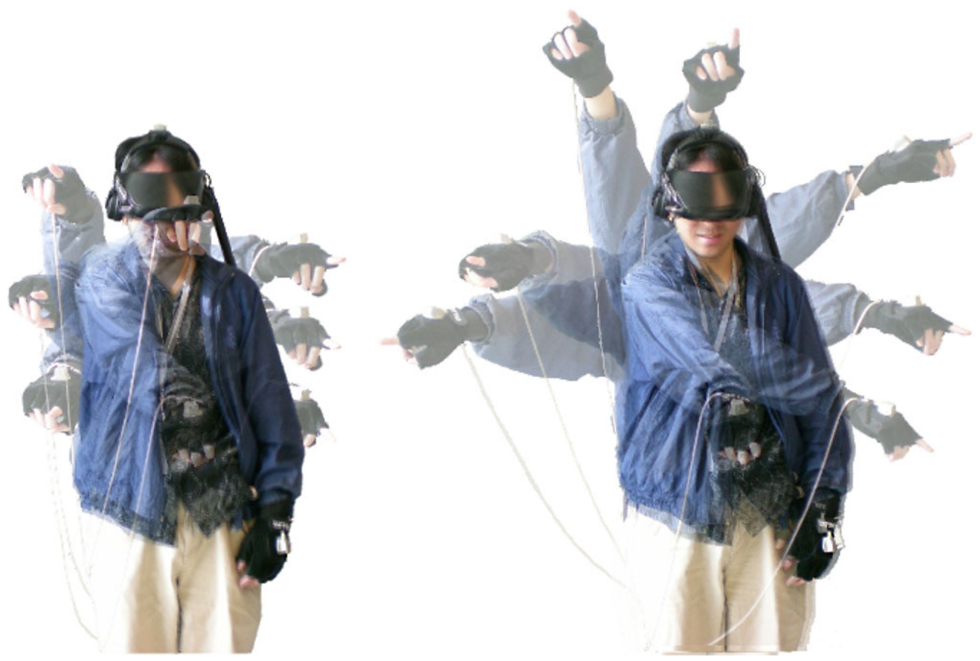
Fig. 3 Two composite images showing the extents of the static (*left*) and sliding views (*right*). For this test, the user was asked to draw the contours of his visible areas, without turning his head. The view-sliding technique more than quadruples the area

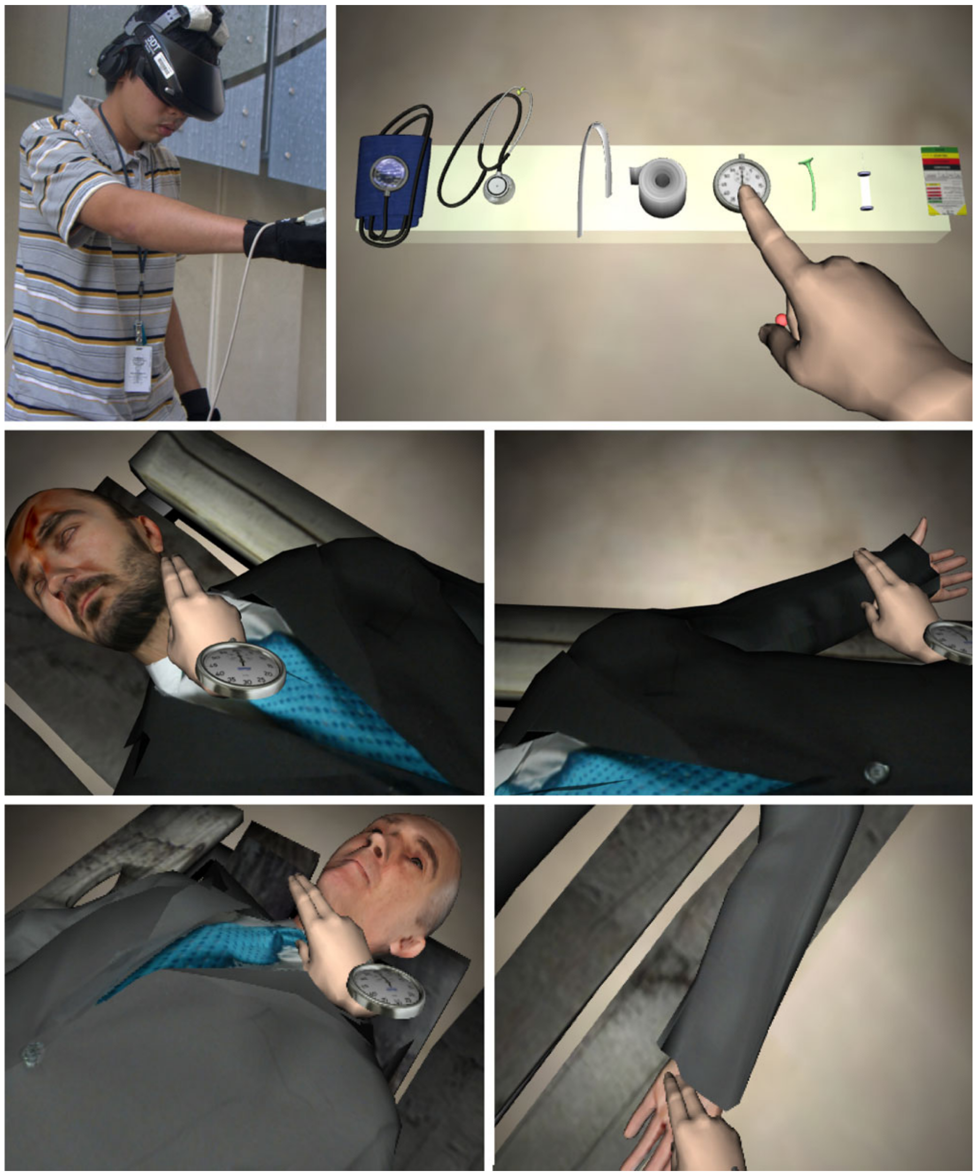
Fig. 4 The user (*top left*) is picking the watch tool from the instrument tray in VR (*top right*). The watch is then used to check pulse rates on virtual patients, at their neck and wrists. Note that only one of the two locations can be seen at any given time, forcing the user to either turn physically, slide the view by moving his hand or do a combination of both

The head and both hands were tracked in 6 degrees-of-freedom by a *Flock of Birds* magnetic motion tracking system by Ascension Corporation, running in extended range mode with a 9-foot tracking radius. The scene was rendered at 25 frames per second by a custom-made VR system based on open source *Flatland* engine [23]. The rendering rate was deliberately locked at a rather low value, to ensure consistent response from the system at all times. Most of the time, the participants were expected to have a single virtual patient in view, of complexity between 10,000 and 15,000 polygons each, as shown in Fig. 4, middle and bottom rows. However, due to 6 DOF camera tracking, it was possible for users to see all ten virtual patients at once, which put significant load on the renderer. Because the view-sliding mechanism is executed at the frequency of the graphics loop, it was essential to keep the update rate constant.

5.4 The procedure

After putting on an HMD and gloves with magnetic 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 were briefly informed about this feature, in simple terms ("You may notice that when you move your virtual hand, the view will follow the hand's direction, to accommodate"). However, these users were not offered any additional time to practice view sliding. Instead, they were expected to learn it as they progressed with the mission.

6 The results: general outcomes

As we mentioned before, the goal of this study was to assess and analyze how the hand-assisted viewing affected user performance and experience in virtual environment. For that purpose, we collected both subjective and objective evidence from each participant.

6.1 User evaluation

After completing the mission, participants were asked to fill out a short survey, which included the following questions. The answers are summarized in Table 1.

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

Table 1 Summary of the participants evaluation reports

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 (⋆)
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 (⋆⋆)
Q 3	4 (agree)	4.00	1.13	3.28 to 4.72

Answering the first question, both groups agreed on importance of having the hand in view. This result is important, as it supports our arguments, presented in Sect. 2.1.

For the second question on the size of the visible area for hand manipulation, both groups came up with median 3 answer (just right). Notably, the 95%-confidence interval for normal viewers dipped below 2 (much too small), marked with (⋆) in the table, while sliding viewers started at 2.45 value (⋆⋆). 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 that the sliding view was helpful and natural, are most encouraging. Evidently, participants were able to make good use of the new technique intuitively, without special training.

We should note that the subjective evaluation of user experience with the hand-sliding viewport controls, discussed above, has mostly illustrative purpose. Thus, it is rather simple and far from being complete. The main contribution comes from the analysis of objective data, collected during the experiments, that captured user activities and performance on a minute-by-minute basis. The data structure and detailed analysis are presented in detail in Sect. 7.

6.2 A special note on motion sickness

We heard concerns the view sliding may cause user discomfort and even motion sickness. Apparently, this was not an issue in our study, perhaps due to a short exposure time and the fact that the participants did not have to travel in the scene. The virtual patients were brought to them instead, on the instructor's command. However, it seems reasonable to expect that under other circumstances, motion sickness may become noticeable, especially if users are required to travel.

6.3 From the session logs

In this study, the VR software kept detailed logs of all user actions and system responses. Among other data, we recorded aggregate times when the view was shifted, for each user. The view lock was engaged and disengaged, according to the rules described in Sect. 4.2. On average, the participants spent 25% of their total time in VR with the view shifted towards their hands. Details are given in Table 2.

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

For both groups of participants, we compared the mission completion times; the summary is given in Table 3. Although the users with the sliding viewport enabled showed

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

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 3 Mission completion times for both groups of participants. Values of p_1 and p_2 show results of the Wilcoxon rank sum test and Welsh two-sample t-test, respectively

	Normal view	Sliding view	Difference
Completion time (sec)			
Median	800.0	710.0	$p_1 = 0.6430$
Mean	738.62	725.27	$p_2 = 0.8417$
SD	166.68	156.28	

slightly better results than the control group, the difference between the two groups is not statistically significant.

The most interesting results came from the time-based analysis of user activities. During each session, the VR system recorded all user actions, time-stamping them with the 1/25 second precision (the frequency of the system internal update cycle). The logging system gave us a valuable tool for measuring and monitoring user progress over time, using various performance metrics. These results are discussed in the next section.

7 Time-based analysis: effects of sliding viewport technique on user performance and the learning process

In our VR system, all UI commands are implemented using a gesture-based language (Sherstyuk et al. [21]) that employs a pairwise *"intention + confirmation"* command structure. In order to issue a complete command, the user must first indicate an intention to perform such command. Upon receiving the intention part, the UI module starts a timer, waiting for the second signal that may either confirm or cancel the pending command. All intention, confirmation and cancellation events are recorded by the UI processing module as text messages, time-stamped and logged into a mission log file.

7.1 Evaluation metrics

Next, we define the three metrics used to evaluate user performance in both groups. These metrics are: command execution rate, hand jitter and tool seek time.

7.1.1 Command execution rate

By examining the ratio of the number of executed commands to the number of intended commands, we can estimate how effectively users communicate with the UI system. For example, a user might initiate a tool-application command by placing the tool on the patient's body (*intention* signal) and then withdraw (*cancellation* signal). In this case, the command is not complete and the system records a "false start". High rates of command executions indicate that the user is confident in how he or she proceeds with the task. Also, that means that the user is comfortable with the UI controls: the number of false starts is low. Low values of execution rate suggest that the user is either uncomfortable with the UI or simply does not know what to do. Having a reasonably large number of participants (28 people total) allowed us to collect a large number of intention and cancellation events at any given time interval. Thus, we define the collective execution rate as a function of time spent on the mission, as follows:

$$\text{Execution rate,} \qquad R(t, \Delta t) = \frac{1}{N}\sum_{k=1}^{N}\frac{E_k(t, \Delta t)}{I_k(t, \Delta t)},$$

where

N	number of users who were issuing commands during $[t, t + \Delta t]$ time interval;
I_k	number of intended commands issued by user k;
E_k	number of executed commands issued by user k.

We have already employed the execution rate metric for evaluating user proficiency in traveling in VR [21]. In that study, the values were based upon the numbers of intentions and confirmations collected over the whole duration of the VR session. In the present work, we examine the trends of the execution rate, as a function of time elapsed from the beginning of the mission, t. The time step Δt used in all calculations was chosen 60 seconds. In this study, we associate the execution rate with the users' proficiency in operating their virtual hands.

7.1.2 Hand stability

In order to select and pick up virtual objects reliably, users must point at them with a steady hand. To execute a *tool pick up* command, the user first indicates an intention to do so, by pointing at the desired tool on the instrument tray (see Fig. 4, top). The confirmation is issued upon a 0.5 second time-out, during which the user must continue pointing at the selected object. During that time, the system captures the length of the path traveled by the tip of the index finger (recall that user's hands are tracked with 6 DOF). The average hand jitter is defined as follows:

$$\text{Hand jitter,} \quad J(t,\Delta t) = \frac{1}{N}\sum_{k=1}^{N}\frac{\sum_{j=1}^{I_k} w_{kj}}{T_k(t,\Delta t)},$$

where

I_k number of intended commands, that triggered the timer for user k, on $[t, t+\Delta t]$ time interval;
T_k total elapsed time when the jitter values were collected for user k, on $[t, t+\Delta t]$ interval;
w_{kj} length of the path traversed by hand of user k, captured for every pending commands j, $j = 1,\dots,I_k$.

Effectively, the hand jitter is the speed of the user's hand, measured and averaged over time while the system was waiting for command confirmation. At these times, users were expected to hold their hands steady.

7.1.3 Tool seek time

The logging system made it possible to capture the exact time intervals elapsed from the moment a tool was picked up until the moment it was applied, by touching dedicated areas on the virtual patient's body (see Fig. 4). 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 seek time values for the watch tool, which was used for checking the pulse. The seek time values were collected as:

$$\text{Seek time,} \quad S(t,\Delta t) = \frac{1}{N}\sum_{k=1}^{N}\frac{\sum_{j=1}^{E_k} s_j}{E_k(t,\Delta t)},$$

where

E_k number of executed commands, issued by user k during $[t, t+\Delta t]$ interval. In this case, the command is "apply watch" by touching the correct place on the patient's body;
s_j seek time values, in seconds, captured from the moment when the watch was picked up and until it was applied, $j = 1,\dots,E_k$.

7.2 Different views—different behaviors

Execution rates, hand jitter values and tool seek times are plotted in Figs. 5 and 6, for participants with normal and sliding viewport, respectively. Apparently, the two groups demonstrated very different trends in how they were using their virtual hands. To investigate further, for all six data sets shown in Figs. 5 and 6, linear models were fit

$y_i = a + bx_i + \epsilon$

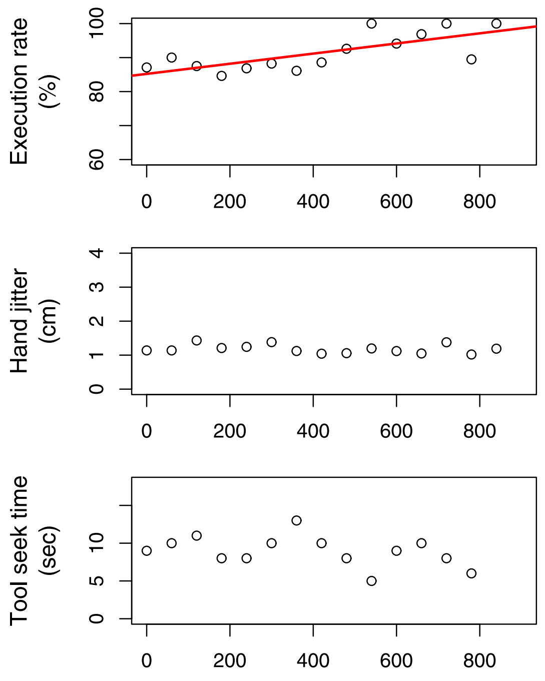

Fig. 5 Progress of the control group of participants with normal view over time: conventional learning curve, climbing up to nearly 100% (*top*); hand jitter remained constant (*middle*) and tool seek times did not change significantly (*bottom*). p_{slope} values: 0.00172, 0.3590, 0.1605

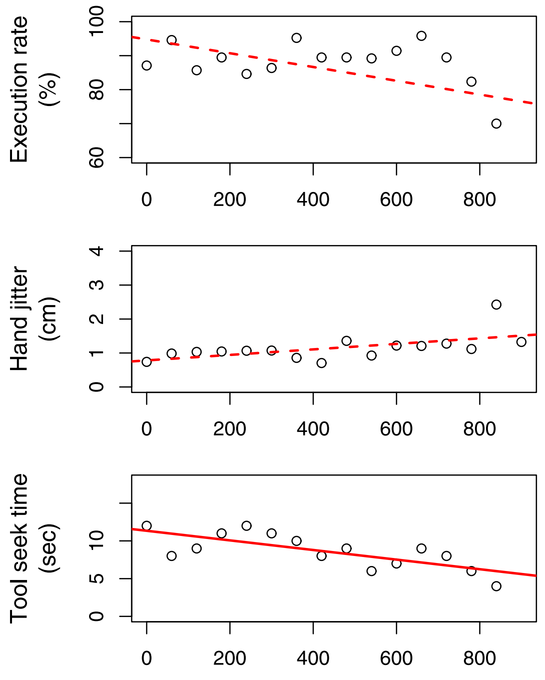

Fig. 6 Participants with sliding viewport demonstrated declining learning pattern, hand stability degraded, tool seek times improved from 11.3 to 5.56 seconds. p_{slope} values: 0.0420, 0.0155, 0.00173

using the least squared regressions, where y_i are values for execution rates, hand jitter and tool seek time and x_i is independent time variable. In all 6 cases, we tested the hypoth-

Table 4 Trends in virtual hand usage for both groups

	Normal view Fig. 5	Sliding view Fig. 6
Execution rate	improves	degrades
p_{slope}	0.00172**	0.0420*
Hand stability	no change	degrades
p_{slope}	0.3590	0.015543*
Tool seek time	no change	improves
p_{slope}	0.1605	0.00173**

esis $b \neq 1$, i.e., that the y_i values are changing with time consistently. Significance probabilities p_{slope} for all models are summarized in Table 4. There, a double asterisk (⋆⋆) marks p-values with a strong significance ($p < 0.01$) and a single asterisk (⋆) marks values with standard significance ($p < 0.05$). In Figs. 5 and 6, the corresponding linear models are plotted as solid and dashed lines, respectively. For data sets with $p_{\text{slope}} > 0.05$, linear models are not plotted, as this is an indication that the scattered samples do not show any consistent trend in their value change.

7.3 Use of sliding viewport and reversed shape of the classical learning curve

Perhaps the most surprising result of this study is that the participants with the sliding viewport appeared to be exempt from the "practice makes perfect" general rule, as the top chart in Fig. 6 suggests. The participants in the control group demonstrated the classical learning pattern, when people start at low values and improve over time. In our case, shown in Fig. 5 top chart, users with normal view started at execution rates of 85.20% (standard error 1.8) and reached 100% towards the end of the trials. Participants with the sliding viewport, on the contrary, showed reverse behavior, with the start value of 94.77% and steady decline over time to 78% (Fig. 6, top). Their hand jitter also increased by the end of the mission (Fig. 6, middle).

7.4 Aggregate metric values

For completeness' sake, we compared the values of execution rate, hand jitter and tool seek time, averaged over the whole duration of the mission, for each participant, and then averaged for each group. The results are presented in Table 5. As the tests indicate, there was no significant difference between users with the normal view and with the sliding view in any of the aggregate values of their performance. This result emphasizes the utility of the time-based analysis, used in this study.

Table 5 Aggregate characteristics for both groups of participants. Values of p_1 and p_2 show results of the Wilcoxon rank sum test and Welsh two-sample t-test, respectively

	Normal view	Sliding view	Difference
Execution rate (%)			
Median	89.80	91.74	$p_1 = 0.6664$
Mean	89.99	89.31	$p_2 = 0.7651$
SD	5.38	5.46	
Hand jitter (cm)			
Median	7.99	5.11	$p_1 = 0.2066$
Mean	7.95	5.39	$p_2 = 0.1445$
SD	4.89	3.30	
Tool seek time (sec)			
Median	9.556	8.880	$p_1 = 0.3107$
Mean	11.260	9.459	$p_2 = 0.1914$
SD	4.15	2.36	

7.5 Discussion

In this study, we used three characteristics of user performance: the execution rate, hand stability and tool seek time. The first two describe the motor component of how users handle the UI controls; they are completely task-independent. The last one, tool seek time, has a strong cognitive element in it, because users were required to apply the tools not just anywhere, but in well defined locations. That required an active search which is a task-dependent procedure. Thus, the three metrics provide both micro- and macro-descriptions of user actions.

By observing degrading values of execution rate and hand stability for participants with the sliding viewport camera control, one might expect that the task-dependent metric, tool seek time, should also degrade. Yet, the reverse is true: by the end of the mission, participants significantly improved their tool-placing skills, reducing the search time from 11 to 5 seconds (Fig. 6, bottom plot). How it can be explained?

We believe, an analogy with learning how to drive a car may be helpful. Novice drivers tend to do everything strictly "by the book": they indicate turns, observe road signs, take multiple precautions diligently. However, these perfectly executed low-level actions do not help them in high-level navigation tasks, such as getting to their destination place faster or not getting lost on the scene. Experienced drivers, on the contrary, are more loose with their movements and actions, while operating the car controls. They drive without thinking consciously about *how* to drive; instead, their mind is preoccupied with *where-to-go* problems. By devoting all their attention to the application tasks, they are able to navigate better.

Moreover, experienced drivers also learn to sidestep recommended procedures in operating car controls. For example, after reaching certain level of confidence, some people prefer to steer using one hand only. Similarly, VR users apparently learned to ignore the recommendation to keep the pointing hand as steady as possible during command execution. Users found out that small amounts of jitter are still allowed by the UI system, and learned to use this fact to avoid overstraining themselves.

We argue that the sliding viewport technique allowed the users to direct their attention from managing motor components of the UI controls more towards the application tasks. Freed from the need to keep the pointing hand inside the narrow viewing cone, people were able to use their hands in a more relaxed way. As a result, the hand jitter values increased and the general style of interaction with the UI controls became less strict and more "noisy." However, the task-related scores improved.

8 Conclusions and future work

We presented new results of the user study on the viewport sliding technique. This semi-automatic real-time camera control mechanism effectively quadruples the working volume of the virtual hand, as illustrated in Fig. 3.

We introduced several performance metrics, both for motor and cognitive skills, and studied how they change under control and experimental conditions. We showed that traditional implementation of the virtual camera results in learning curves of classical style, when users start at low values and show steady progress over time. Conversely, the view-sliding technique prompts radically different trends in user progress. In particular, the time-based analysis of user performance helped us observe an unusual phenomenon of "regressing learners," or people with declining learning curves.

This result alone merits additional experimental studies, in the context of various applications of the virtual hand metaphor. Some results have been obtained already. Using the time-based analysis of user travel activities in VR, we were able to categorize all travelers into "progressing," "regressing" and "neutral" learners, basing on their progress in mastering travel controls over time. There, we used two travel metaphors: steering-by-pointing and teleportation towards a target. Both techniques require an active use and precise control over the virtual hand. We believe that the observed polarization of the virtual travelers into three categories was due to the fact that the execution rate metric reflects certain personal qualities. Hesitant and risk-averse users are likely to operate the travel UI controls with lower values of execution rate than self-confident and risk-seeking users. Another factor that may have influenced the values and trends in execution rates is that some people are what is called "natural" players, who almost immediately grasped the idea behind the travel UI, and started using it confidently, showing very high execution rates. These people demonstrated the "regressing" pattern in their travel skills. Interestingly, regressing users also showed best task completion times, which is consistent with the findings presented in this work. The results of the study on virtual travel are presently under review. We expect that adding view-sliding extension to hand-based travel techniques, such as steering and teleportation, may help to learn more about hand–eye coordination during travel in VR.

9 Closing remarks

The experimental results show that people seemed to like view sliding, which is clearly an argument for using it in VR interface design. Tool seek time also decreased, indicating that as people get used to using it, they become better at locating the tools they need. However, people also become less precise in their movements, possibly because the non-isomorphic mapping provided by view sliding is less precise than their usual experience of how their field of view changes. If this is correct (it requires further study) this is potentially important for VR design in general. Use non-isomorphic mappings because people, in general, feel positive about them and they help them to get the task done, but be prepared for a decrease in the accuracy of their movements.

References

1. Biguer, B., Jeannerod, M., Prablanc, P.: The coordination of eye, head and arm movements during reaching at a single visual target. Exp. Brain Res. **46**, 301–304 (1982)
2. Boger, Y.: Are existing head-mounted displays "good enough"? A summary of a worldwide HMD survey on past experience and future expectations from HMDs (2007). http://www.sensics.com/downloads/2007HMDSurveyResults.pdf, accessed on June 19, 2010
3. Bolas, M., Pair, J., Haynes, K., McDowall, I.: Display research at the University of Southern California. IEEE Emerging Displays Workshop, Alexandria, VA (2006)
4. Bowman, D., Kruijff, E., LaViola, J., Poupyrev, I.: 3D User Interfaces: Theory and Practice. Addison-Wesley, Reading (2004)
5. Britton, E., Lipscomb, J., Pique, M.: Making nested rotations convenient for the user. In: Proceedings of SIGGRAPH'78, pp. 222–227 (1978)
6. Bungert, C.: HMD/Headset/VR-Helmet Comparison Chart. http://www.stereo3d.com/hmd.htm, accessed on June 19, 2010
7. Jacoby, R., Ferneau, M., Humphries, J.: Gestural interaction in a virtual environment. Stereoscopic displays and virtual reality systems. SPIE **2177**, 355–364 (1994)
8. Jaekl, P., Allison, R., Harris, L., Jasiobedzka, U., Jenkin, H., Jenkin, M., Zacher, J., Zikovitz, D.: Perceptual stability during head movement in virtual reality. In: Proc. IEEE VR, pp. 149–155 (2002)

9. Jaekl, P., Jenkin, M., Harrisa, L.: Perceptual stability during active head movements orthogonal and parallel to gravity. J. Vestib. Res. **13**, 265–271 (2003)
10. Jay, C., Hubbold, J.: Amplifying head movements with head-mounted displays. Presence **12**, 268–276 (2003)
11. Kiyokawa, K.: A wide field-of-view head mounted projective display using hyperbolic half-silvered mirrors. In: Proceedings of ISMAR'07, Nara, Japan (2007)
12. Land, M.: Eye movements in daily life. In: Calupa, L., Werner, J. (eds.) The Visual Neurosciences, vol. 2, pp. 1357–1368. MIT Press, New York (2004)
13. Hinckley, K., Tullio, J., Pausch, R., Proffitt, D., Kassell, N.: Usability analysis of 3D rotation techniques. In: UIST '97: Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology, pp. 1–10. ACM Press, New York (1997)
14. LaViola Jr., J., Feliz, D., Keefe, D., Zeleznik, R.: Hands-free multi-scale navigation in virtual environments. In: Proceedings of ACM SIGGRAPH I3D Symposium on Interactive 3D Graphics, North Carolina (2001)
15. LaViola, J., Forsberg, A., Huffman, J., Bragdon, A.: The influence of head tracking and stereo on user performance with non-isomorphic 3D rotation. In: Proceedings of the 14th Eurographics Symposium on Virtual Environments, pp. 111–118 (2008)
16. LaViola, J., Katzourin, M.: An exploration of non-isomorphic 3D rotation in surround screen virtual environments. In: Proceedings of the IEEE Symposium on 3D UI, pp. 49–54 (2007)
17. Neale, D.: Head-mounted displays: product reviews and related design considerations. Hypermedia Technical Report HCIL-98-02, Human–Computer Interaction Laboratory, Department of Industrial and Systems Engineering, Virginia Tech, Blacksburg, VA 24061-0118 (1998)
18. Poupyrev, I., Billinghurst, M., Weghorst, S., Ichikawa, T.: The go-go interaction technique: non-linear mapping for direct manipulation in VR. In: Proceedings of ACM UIST, pp. 79–80 (1996)
19. Poupyrev, I., Weghorst, S., Fels, S.: Non-isomorphic 3D rotational techniques. In: Proceedings of CHI'2000, pp. 540–547 (2000)
20. Rosenbaum, D.: Human Motor Control, pp. 253–264. Academic Press, New York (1991)
21. Sherstyuk, A., Vincent, D., Hwa Lui, J., Connolly, K., Wang, K., Saiki, S., Caudell, T.: Design and development of a pose-based command language for triage training in virtual reality. In: Proceedings of IEEE 3DUI (2007)
22. Sherstyuk, A., Vincent, D., Jay, C.: Sliding viewport for interactive virtual environments. In: Proceedings of ICAT, Yokohama, Dec. 1–3 (2008)
23. The Homunculus Project. http://www.hpc.unm.edu/homunculus, accessed on June 19, 2010
24. Vincent, D., Sherstyuk, A., Burgess, L., Connoly, K.: Teaching mass casualty triage skills using immersive 3D virtual reality. Acad. Emerg. Medicine J. **15**(11), 1160–1165 (2008)

Andrei Sherstyuk earned his first higher education degree in physics from Novosibirsk State University, Soviet Union, in 1989. After working several years as a consultant for AT&T, he started an M.S. program in Computer Science at Caltech and graduated in 1994. He defended his Ph.D. dissertation "Convolution surfaces in computer graphics" at Monash University, Australia.
In 1999, Andrei Sherstyuk moved to Hawaii to work on the world's first photo-realistic CG film Final Fantasy, "The Spirits Within."
In 2002, Andrei Sherstyuk joined Telehealth Research Institute, where he directed research and development on Virtual Reality systems for medical education and training. Now he works on the next-generation online virtual world Blue Mars, at Avatar Reality Inc. His recent publications are in the area of Virtual/Augmented/Mixed realities, navigation and human-computer interactions.

Caroline Jay completed a B.A. in Psychology and M.Sc. in Computer Science at the University of Manchester, UK, before moving to engineering firm ALSTOM to work as a software developer. She returned to the University of Manchester to study for a Ph.D., "Modeling the effects of network delay on human performance in collaborative virtual environments", which she completed in 2006, and has remained there since working as a researcher in Human Factors.
Jay's current research focuses on modeling how people perceive and use digital information in both Virtual Environments and on the Web. In addition she works as a consultant at the Web Ergonomics Lab, in the areas of Web accessibility and usability.

Anton Treskunov graduated from the Mathematics Department at Moscow University. Dr. Treskunov completed his Ph.D. in Computer Science at the Keldysh Institute of the Russian Academy of Science. His Ph.D. work focused upon issues related to the application of machine vision techniques for automated industrial inspection.
In 1999, he moved to United States, where he worked as a multimedia software developer at Phantom Reality. In 2003–2009, Dr. Treskunov worked for Institute for Creative Technologies (University of Southern California) as the lead researcher. Among his many responsibilities, Dr. Treskunov oversaw software engineering decisions related to the development of various mixed reality projects. His research at ICT has been focused upon computer vision based techniques for enhancing tracking and rendering in mixed reality systems. In 2009, Treskunov joined Silicon Valley based Computer Science Lab of Samsung Electronics; his research here targets Human Computer Interaction and presenting 3D information in novel ways.