3D Interaction Techniques for 6 DOF Markerless Hand-Tracking

Markus Schlattmann Tanin Na Nakorn Reinhard Klein

Universität Bonn, Institut für Informatik II - Computergraphik, D-53117 Bonn, Germany
markus@cs.uni-bonn.de, tanin47@yahoo.com, rk@cs.uni-bonn.de

ABSTRACT

Recently, stable markerless 6 DOF video based hand-tracking devices became available. These devices track the position and orientation of the user's hand in different postures with at least 25 frames per second. Such hand-tracking allows for using the human hand as a natural input device. However, the absence of physical buttons for performing click actions and state changes poses severe challenges in designing an efficient and easy to use 3D interface on top of such a device. In particular, solutions have to be found for clicking menu items, selecting objects and coupling and decoupling the object's movements to the user's hand (i.e. grabbing and releasing). In this paper, we introduce a novel technique for grabbing and releasing objects, an efficient clicking operation for selection purposes and last but not least a novel visual feedback in order to support the ease of using this device. All techniques are integrated in a novel 3D interface for virtual manipulations. Several user experiments were performed, which show the superior applicability of this new 3D interface.

Keywords: 3D interaction, user interfaces, hand-tracking

1 INTRODUCTION

For interaction with a virtual environment, hand-tracking is one of the favorite approaches, because it directly exploits the ease and perfection with which humans employ their hands in everyday life. In order to support immersive user experience, markerless real-time hand-tracking without the need of special initialization procedures gained a lot of interest in recent years. Presently, methods fulfilling these properties are capable of tracking up to 6 continuous degrees of freedom (DOF) of the hand pose (global position and orientation) and recognizing several stiff postures.

While such systems enable the translation of real hand movements to virtual movements, several problems have to be solved in order to build a 3D interface on top of the basic hand-tracking technology. Exploiting the detection of position, orientation and posture the interface has to provide mechanisms for basic interaction techniques such as object selection, grabbing/releasing and 3D navigation. Thereby an easy to use realization of these techniques is crucial for usability and efficiency of the interface. These techniques have to be adapted to the users' capabilities and incapabilities (e.g. the limited range of angle movements of the human wrist), the applications' specifications as well as the requirements and drawbacks of the used hand-tracking method. This diversity of demands poses several challenges in the design of efficient interfaces as described in the following.

As suggested in [BKLP04] we strictly distinguish between the notions pose, posture and gesture. The pose is meant to be the combination of a rigid body's 3D position and orientation (e.g. the hand's global pose). A hand posture is defined as a specified configuration of finger limb positions and orientations relative to the hand pose (e.g. the fist posture). A hand gesture (or free-hand gesture) describes a predefined movement of the hand pose (e.g. writing a letter in the air).

The first challenge is grabbing and releasing of objects, which are inherent tasks during 3D interaction sessions due to the following reasons. The virtual world is typically significantly larger than the working volume of the user (the 3D region the hand is tracked in). Therefore, to enable users to move a virtual object to every position in the virtual space it has to be possible to grab it and release it in order to move it step by step. Scaling the working volume to the whole virtual world is not an option, because the accuracy would decrease too heavily. Similar, the range of angle movements of the human wrist is limited. In order to fully rotate and inspect an object from all viewing directions grabbing and releasing are indispensable.

In interfaces that employ a standard 2D mouse, grabbing and releasing is solved either by lifting the mouse (while it is lifted a mouse movement induces no object movement) or by exploiting button states (usually the object is only moved if a button is held down). But in contrast to standard controllers no direct adequate exists for markerless hand-tracking. Simple solutions for the realization of a grab and release cycle in the absence of physical buttons are disposing one degree of free-

dom (DOF) of the hand pose, e.g. only if z-coordinate is greater than a certain value the object is grabbed, exploiting the second hand or applying different postures for different button states. Unfortunately, these approaches also have drawbacks. Exploiting one DOF of the hand pose is only possible, if less than 6 DOF are needed in the current task. Using the second hand, e.g. to press some virtual button defining the current grab and release states, is much more uncomfortable due to the need of straining both hands and arms simultaneously. The use of different postures, one for grabbing and another for releasing, is significantly more demanding for the user than simply pressing a mouse button, because the physical effort as well as the complexity of coordination of changing postures is considerably higher, especially if various specified postures are concurrently needed. Moreover, a posture change always induces an unintended pose change mainly in rotation in current markerless hand-tracking systems. This is due to the problem that the tracking state is temporarily undefined during a posture change. Therefore, it would nearly be impossible to instantly stop an object's movement by switching to another posture.

The second challenge is the lack of suitable techniques for selecting objects and tools. In standard interfaces typically one or more controller buttons are exploited and a selection is performed by moving the cursor above a virtual button or object on the screen and performing a click. To generalize this to 3D hand-tracking interfaces the hand movements have to be used both for moving a cursor (or the like) and simultaneously for clicking. Fortunately, in this case normally not the all DOFs of the hand pose are needed for moving a cursor; the hand's position and/or pointing direction is sufficient such that some DOFs can be used for clicking purposes. Nevertheless, the usability and efficiency of selection is crucial.

Another challenge of using hand-tracking for interaction is providing the user information about the limited working volume of the hand-tracking device and details about the current tracking state. The user has to ensure permanently that her hand is located inside the working volume and that she is performing the correct pose and posture for solving the current task. Without suitable techniques to facilitate these needs, this can lead to extremely high demands for the user.

In this paper we introduce three novel approaches to solve the above mentioned problems: first, a new interaction technique is introduced, which allows grabbing and releasing of objects while still enabling manipulations using the full 6 DOF of the hand pose without the need of posture changes or the incorporation of the second hand. Second, in order to perform selection operations we introduce an effective technique to simulate the left and right button of a standard 2D mouse. As a third key contribution we introduce different kinds of visual feedback for supporting manipulation tasks. This visual feedback helps the user to manage the general problems of markerless video based hand-tracking (limited working volume and pose/posture verification) as well as handling the employed interaction modes.

All novel techniques are consistently integrated into a graphical user interface (GUI) in order to demonstrate how they can be applied. Moreover, this interface is described in detail and several ideas for the realization of the interface are proposed. Last but not least a short evaluation and discussion of our new interface is presented, which is based on a user study, user questioning and our observations.

2 RELATED WORK

The large amount of literature of interaction techniques makes it practically impossible to give a full review of the previously reported methods here; elaborate analysis can be found in [BKLP04], or [JS07] for multi modal interaction. We will only discuss the most related methods that are designed for or can be applied to 3D interaction interfaces based on markerless hand-tracking as an input device.

2.1 Virtual clicking techniques

If objects in a scene or menu items in a 3D menu (a survey of 3D menus can be found in [DH07]) shall be selected a suitable clicking technique is needed to trigger the selection event while some kind of cursor has to be moved to choose the desired object/item. If a hand-tracking device is employed the clicking operations have to be simulated due to the absence of physical buttons. Note that for selection purposes exploiting a DOF of the hand pose for triggering clicking events is feasible because in general not all 6 DOFs are needed for moving the cursor. In the following the different approaches for clicking simulation suitable for hand-tracking interfaces are outlined.

The first and easiest solution for performing clicking operations is extending the hand-tracking interaction interface with additional physical buttons as for example floor pedals. However, this kind of interaction is awkward and slow (according to [GFF$^+$04]).

Another approach is using a cursor dwell time threshold for triggering a click event as for example used in [WP03] and [GFF$^+$04]. Although this is simple, it introduces a constant lag in the interaction.

A further approach is to use speech to signal a selection [Bol80]. But this is especially excessive if several click down and up events have to be captured.

To perform a click by specified movements of the hand is another option. In [GFF$^+$04] clicking is performed if the user moves her hand 20 cm toward the camera. We observed this technique to lack efficiency, because it requires a quite spacious hand movement,

which is slow and inefficient. A better solution is proposed in [VB05], where clicking can be performed by a small movement with the index finger, similar to how we move when clicking a physical mouse button. But obviously, this technique can not be used when the pointing direction of the index finger is needed at the same time (e.g. for selecting by pointing).

An additional commonly used technique is exploiting different hand postures to click. In [GWB05] and [VB05] a button down or up event is triggered, when the thumb is moved in or out toward the index finger side of the hand. Unfortunately, due to unstable tracking states while a transition between two postures is performed, this often leads to unmeant changes in the pointing direction. Moreover, even if a simple posture is applied, it is significantly more complex and uncomfortable to change postures than simply press a mouse button.

2.2 Virtual grabbing and releasing techniques

Once an object is selected, different manipulations can be applied. During complex object movements all 6 DOFs of the user's hand pose are required to move the object and additionally a suitable mechanism is needed for precise releasing it in the desired pose. For this reason, the virtual clicking techniques can in general not be applied for this problem and other solutions have to be found for determining when the object shall be attached to the hand (i.e. coupling the object's movements to the hand's movements).

According to Zachmann [Zac00] grabbing an object (i.e. attaching the object to the hand) can be realized in (at least) three different ways: single-step, two-step or naturally. Single-step grabbing attaches the object at a certain event (e.g. a spoken command like "grab thing"). Two-step grabbing can be further divided into the following interaction steps:

1. Some event (e.g. a posture or spoken command) switches the grabbing mode on; only in this mode, objects can be grabbed.

2. The object is attached to the hand at another event.

To release the object usually the same event as in the first step is used. In the grabbing mode natural grabbing is typically realized by conditioning collisions of the finger tips with the object (e.g. the thumb and one forefinger must collide with the object). The object's movements will be coupled directly to the hand's, when the object is touched this way. The types of natural grabbing can be further distinguished into several different classes (for details see [Zac00]).

Using physical buttons, a dwell time threshold or speech for triggering an attach action suffer from the same drawbacks as in the case of clicking. As well exploiting one DOF of the hand pose is not possible, because for object movements normally all 6 DOFs are needed.

Therefore, most approaches adopt grabbing postures to determine whether an object is attached to the hand or not (e.g. [MF04] or [BI05]). Unfortunately, it turned out to be quite difficult to release an object at a precise position [Osa06]. The reasons for this are: first, it is demanding for people to fix a hand precisely in midair without having physical support. Second, judging the release point without tactile feedback can be difficult. Third, the finger movements of a grabbing action often cause the hand's global pose to change. To solve the third problem of using grabbing postures Osawa [Osa06] proposed an approach to automatically adjust the release pose of a virtual object based on the relative speed of the two grabbing fingers (usually the thumb and one forefinger).

Furthermore, in current markerless hand-tracking systems most of the different types of natural grabbing are not available, typically only one grabbing posture is supported. Additionally, in order to ensure a stable tracking, this posture must be performed very exactly and clearly. We observed this to be cumbersome for most users.

2.3 Object manipulation techniques

In a hand-tracking based 3D interface commonly a virtual representation of the user's hand is shown in the 3D scene. When the virtual representation intersects with an object, the object can be grabbed. Once grabbed, the movements of the virtual hand are directly applied to the object in order to move, rotate or deform it. This is called the virtual hand metaphor, which is the most common direct manipulation technique for manipulating objects [BRC05]. When the coupling between the physical world (hand or device) and the virtual representation works well, this interaction technique turns out to be very intuitive, since it is similar to every-day manipulation of objects. The main drawback of the virtual hand metaphor is the scaling problem; the limited workspace of the user's limbs or the input device, which makes distant objects unreachable.

To solve this problem various approaches were reported. For example the Go-Go technique [PBWI96] simulates an interactively non-linear growing of the user's arm. When the user's hand is close to her, the mapping of real hand pose to virtual object pose is one to one. As she extends her hand and arm beyond a certain range, the mapping becomes nonlinear and the virtual arm "grows". Thus, she is able to reach objects out of her range. Several other solutions to the scaling problem were introduced as for instance the World-in-Miniature technique [SCP95], HOMER (hand-centered object manipulation extending ray-casting) [BH97], Scaled-World Grab [MFPBS97] or Voodoo Dolls [PSP99]. However, none of these techniques can be

identified as the "best" solution; their performance depends on the task and environment.

In order to improve the accuracy of object movements, the PRISM method was introduced [FK05]. This interaction technique acts on the user's behavior in the environment to determine whether they have precise or imprecise goals in mind. When precision is desired, PRISM dynamically adjusts the control-to-display-ratio which determines the relationship between physical hand movements and the motion of the controlled virtual object. A similar approach to automatically adjust the speed of the current action is described in [Osa06].

3 HAND-TRACKING DEVICE

To markerless track the 6 continuous DOFs of the user's global hand pose in several different stiff postures we implemented the method of Schlattmann et al. [SKSK07]. We decided to use this approach due to the real-time capability and automatic initialization. In contrast to [SKSK07], we connected all three cameras to one and the same computer where as well the hand-tracking is computed. The intersection of the viewing volumes of the cameras defines the working volume of the hand-tracking. It describes the physical space the user's hand pose and posture are determined in (approximately $80cm \times 50cm \times 50cm$).

In this method the fingertip positions of two stretched fingers are identified and the hand center is computed as a third point to sufficiently determine the hand pose in each frame. Note that for this reason movements of the stretched fingers slightly influence the derived hand pose. The hand posture is determined by evaluating some heuristics based on the local surroundings of the fingertip points. In our interface we generally employ the pointing posture with the stretched thumb and index-finger.

4 INTERACTION TECHNIQUES

Each of the basic techniques described in this section can be used in different parts of an interface as will be described in Sec. 5.

4.1 Visual feedback box

The adaption to novel devices and/or interaction techniques can be very demanding particularly for an unexperienced/unsupervised user. As a support we therefore integrate specific visual feedback into our GUI, which is shown in a small 3D box that can be positioned freely (typically in the upper left or right corner). The visual feedback box provides three basic visual cues (see Fig. 1(Left)):

1. A hand model, which is moved according to the user's hand in order to indicate the recognized pose and posture.
2. A cuboid, in order to show the working volume of the hand-tracking. If the user moves her hand inside the working volume, the hand model (see Cue 1) is shown inside this box. This way the user gets a visual hint, if her hand can be seen by enough cameras and if the tracking is working correctly.
3. The shadow of the hand model on the floor of the cuboid. This helps the user to estimate the position of the hand along the z-axis more accurately.

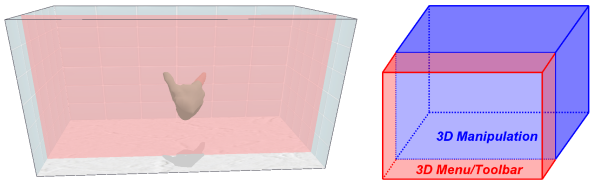

Figure 1: (Left) An example of the basic visual feedback. (Right) Partitioning of our working volume.

Furthermore, the visual feedback box can be extended with other visual cues in order to help users to learn how to interact with the GUI. Currently, there are three major extensions for the respective modes of interaction, which will be explained in the respective sections. For lack of space, we refrain from illustrating all extensions by depicting screenshots; they are shown in the accompanying video.

4.2 Roll click

The roll click is introduced for simulating button events. The simulation of button events is needed for selecting objects and menu items. Thereby, the clicking operation should be easy to learn as well as easy to perform. To this end, we decided to exploit one DOF for triggering button events. We found exploiting a specified rotation around the roll-axis (i.e. around the axis described by the forearm) serves best. This has several reasons. First, exploiting a rotational DOF for clicking is superior to using a translational DOF due to comfort issues. Using a translational DOF would force the user to move the whole fore arm in order to perform a click.

Second, a rotation around the roll-axis performs better than around the yaw or pitch-axis, because the range of rotation the user can utilize for this rotation is significantly larger. Moreover the roll-angle's value is only marginally affected by changing the hand's position or pointing direction. The yaw and pitch angles depend loosely on the position of the hand (e.g. translating the hand toward the left induces a rotation toward the left except the wrist is bended for compensation), which could lead to unmeant clicking operations.

Our approach is particularly advantageous compared to exploiting a posture or the second hand for clicking because it is significantly easier and less exhausting to

perform. We observed some users to be nearly incapable to form a specified posture while further concentrating on the current task.

For deciding if a virtual button event is triggered in frame i (the i-th time the hand pose/posture was determined), we use two sufficient conditions based on the value of the user's hand's roll-axis angle α_r^i. The first condition enables very slow clicking with a more spacious movement while the second condition also enables clicking by smaller but faster movements. Note that one condition would be sufficient, but using both conditions better accounts for the individual user preferences. The first condition employs a hysteresis thresholding (i.e. a thresholding, that employs different threshold values depending on the state that is occupied) based on α_r^i for triggering a button event. This is expressed in the first terms in Fig. 2, respectively. T_1 and T_2 denote the hysteresis thresholds. Currently we use $T_1 = \frac{\pi}{4}$, $T_2 = \frac{\pi}{16}$. In our current setting the roll-axis angle is defined to be zero, if the index finger is pointing toward the front and the thumb is pointing up. A counterclockwise rotation of the user's hand around its roll-axis increases this angle while a clockwise rotation leads to a decrease.

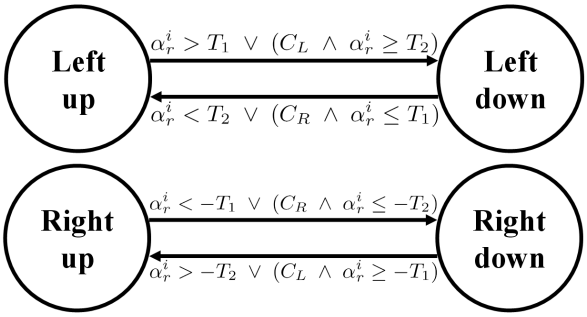

Figure 2: State machines illustrating when the virtual left and right button is pressed or released.

The second sufficient condition is expressed in the second terms in Fig. 2, respectively. These terms comprise one of the conditions C_L or C_R and an additional constraint based on the hysteresis thresholds (see previous paragraph). The additional constraints are needed to disambiguate between left and right button events; otherwise a right button down event could not be distinguished from a left button up event. The conditions C_L and C_R are based on α_r^i as well as on the signed angular velocity v_r^i of the roll-axis angle, which is defined as

$$v_r^i = \frac{\alpha_r^i - \alpha_r^{i-1}}{t^i - t^{i-1}}, \tag{1}$$

where t^i is the time of frame i. Now C_L can be defined as follows. If k is the greatest positive number with $v_r^{i-j} \geq \varepsilon$ for all $j = 1, ..., k$, then the condition C_L is defined as

$$C_L = v_r^i < \varepsilon \ \wedge \ (\alpha_r^{i-1} - \alpha_r^{i-k} > \frac{\pi}{16}). \tag{2}$$

This way, already a small counterclockwise rotation can be employed to simulate a left button down or right button up event. The threshold ε ensures the rotation to have a minimal velocity (we used $\varepsilon = \frac{1}{2}\pi \frac{rad}{s^2}$). The second term of Eq. (2) is needed to avoid unmeant button events by requesting the angular movement to exceed a minimal value (otherwise an infinitesimal movement could lead to a button event). C_R is defined analogously by substituting $-\varepsilon$ for ε and $-\frac{\pi}{16}$ for $\frac{\pi}{16}$ and inverting the relational operators.

If a clicking operation is performed, we observed the pointing direction to lack accuracy due to unmeant angular movements around the pitch or yaw-axis. Therefore, selecting a small object by employing the virtual pointer metaphor [PIWB98] can be hard to accomplish. To this end, we replace the yaw and pitch angle values α_y^i and α_p^i of the current frame i with the angles of the last frame, which fulfilled the condition $\left|\alpha_r^i - \alpha_r^{i-1}\right| < 2(\left|\alpha_p^i - \alpha_p^{i-1}\right| + \left|\alpha_y^i - \alpha_y^{i-1}\right|)$. This way, the pointing direction remain constant during a clicking operation, because in this case the hand rotation is mainly around the roll-axis.

When this technique is currently applied in the interface, the visual feedback box provides two small buttons (visualized as cylinders), positioned left and right alongside the hand model, which indicate the user how she can perform clicking.

Note that exploiting one DOF for clicking purposes leaves us only 5 DOFs for other manipulations. Therefore, this technique can only be employed in specific interaction modes such as selection of tools or objects.

4.3 Jerky release

Once an object is selected and the full amount of the 6 continuous DOFs of the global hand pose is employed for moving the object, an additional suitable mechanism is needed for determining when the object shall be attached to the hand or not. This mechanism should enable precise releasing of objects and should be manageable fast and efficiently. To this end, we used an approach based on the velocity and acceleration of the hand translation and rotation to trigger grabbing (attach action) and releasing. The idea is to move a virtual object only if the user moves her hand smoothly (i.e. no abrupt pose changes). If she instead performs a fast and jerky movement the object is released. This provides an intuitive interaction metaphor as it corresponds to real life experience (e.g. if a screw is turned downward, people typically do a relatively slow clockwise rotation while turning the screw and a relatively fast counterclockwise rotation back without turning the screw).

This behavior is implemented in the state machine depicted in Fig. 3. C_a and C_v are conditions based on

the translational and rotational velocities and accelerations v_t, v_r, a_t and a_r and are defined as

$$\begin{aligned} C_a &= a_t > A_t \vee a_r > A_r, \\ C_v &= v_t < V_t \wedge v_r < V_r, \end{aligned}$$

where V_t, V_r, A_t and A_r denote the respective thresholds. Note that by conditioning the signed accelera-

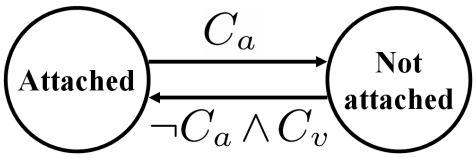

Figure 3: State machine illustrating how an object can be grabbed and released.

tions no releasing is performed, when the user jerks to a halt (leads to high negative accelerations). In our current setting the thresholds were set to be $A_t = 50\frac{cm}{s^2}$, $A_r = \frac{5}{3}\pi\frac{rad}{s^2}$, $V_t = 40\frac{cm}{s}$ and $V_r = \pi\frac{rad}{s}$. These threshold values were determined in a pilot experiment, where first an object should be moved in only one direction (we used the positive x-direction for translation and a clockwise rotation around the roll-axis for rotation) with different employed threshold values while the percentage of involuntary movements was measured. A performed hand movement is classified to be *involuntary* either if the movement is in the desired direction but the object is not moved (the acceleration threshold was exceeded without intension) or if the movement is in the opposite of the desired direction but the object is moved (the acceleration threshold was not exceeded). Note that using lower threshold values leads to more occurrences of the first kind of involuntary movements while higher threshold values abets the second kind. Therefore, we rate the chosen thresholds by the sum of the squared respective errors (the amount of involuntary movements). This way, both kinds of involuntary movements are minimized. Additionally, we let the subjects perform some simple manipulation tasks while the completion times and precision were measured (similar to the user experiments which will be described in Sec. 6). We noticed a strong coherence between good performance (completion times and precision) and good threshold rating. The above stated threshold values result from several of these tests and had the best overall performance. In our interface, the user can adjust these values by choosing a factor from the interval $[0.5;2]$ with which all threshold values are scaled in order to account for the different preferences of each individual. However, for the user experiments in Sec. 6 this factor was locked to 1.

When a fast and jerky movement is performed, typically the acceleration curve has a positive peak at the beginning, then decreases to approximately zero and has a negative peak in the end (see Fig. 4(Right)). Therefore, the condition C_v is essential for conditioning a transition back to state 'Attached'; otherwise the 'Not attached' state would be leaved directly after the positive peak at the beginning.

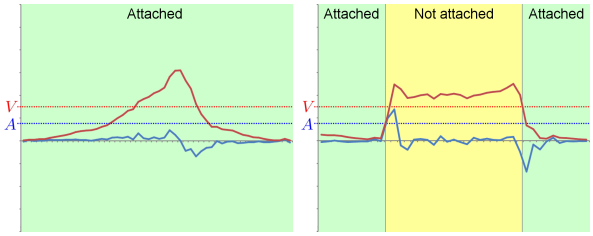

Figure 4: Diagrams of velocity (red) and acceleration (blue) across several frames (x-axis) of two different translational movements. The two horizontal dashed lines indicate our chosen thresholds A_t and V_t and the green and yellow regions the periods of having attached or released the object. (Left) A movement is performed by smoothly increasing the speed. (Right) A fast and jerky movement is performed.

With this implementation a jerky translational or rotational hand movement induces a transition from state 'Attached' to state 'Not attached' and if the translational and rotational speed as well as the acceleration of the hand movement falls below the according given thresholds a transition back to state 'Attached' is performed. As long as the speed is not abruptly increased the object will move according to the users hand.

Note that if a simple thresholding on the velocity would be used, precise releasing of the object would hardly be possible, because dependent on the acceleration it could take several frames until the velocity threshold is reached. But in these frames the object would still be moved. Using a very low velocity threshold to diminish this problem would prohibit the user from performing any fast operation. This can be seen in Fig. 4, where we depicted graphs of velocity and acceleration for two translational motion sequences: one, where the user performed a typical smooth manipulation (Left), and another, where she performed a fast and jerky movement (Right). Analogously, we recorded two rotational motion sequences, which showed the same characteristics.

In addition, using the acceleration as a single criterion for transiting to state 'Not attached' has the advantage, that this way, the velocity with which an object can be moved is not constrained. For example in the motion sequence of Fig. 4(Left) the object was attached all along although the velocity exceeded the velocity threshold.

Using the acceleration criterion for releasing operations enables fairly precise positioning of objects. However, sometimes slight unintentional movements occur in the direction the fast and jerky movement is per-

formed, because dependent on the jerkiness of the performed movement it can take a short while until the acceleration threshold is exceeded. To overcome this problem, we introduce a simple post-correction step, when the 'Not attached' state is reached in frame i. We undo the last k manipulation steps (both translation and rotation), whereby k denotes the greatest number of steps, that fulfill the following conditions for all j with $i-k \leq j < i$:

$$\left(\frac{\left\| p'^{j} \right\|}{t^{j} - t^{j-1}} < \frac{\left\| p'^{j+1} \right\|}{t^{j+1} - t^{j}} \right) \wedge \left(t^{i} - t^{j} \right) < t_0. \qquad (3)$$

t^j denotes the time and p'^j is defined depending on whether the 'Not attached' state is reached due to a high translational or high rotational acceleration. In the case the translational acceleration threshold is exceeded, p'^j is defined to be the hand position increment $p^j - p^{j-1}$ (p^j denotes the hand position in frame j). In the other case, we instead define p'^j to be the normalized rotation axis of quaternion q'^j multiplied by its angle. q'^j is defined as the rotation of the hand from frame $j-1$ to frame j. The first condition in Eq. (3) ensures the absolute acceleration value to be strictly increasing for the k steps. This avoids unintended undoing of steps, if for example the user moves an object, stops shortly and then wants to release it by performing a fast and jerky movement. The second condition prohibits undoing steps that are longer ago than t_0. This threshold describes the maximal available time the user has to exceed the acceleration threshold. In our current setting t_0 is chosen to be 100 milliseconds as several experiments showed this to be suitable. This post-correction enables very precise release operations in all manipulation tasks. Moreover, because only very slight post-corrections are needed, the distraction induced by automatic undoing is only marginal.

The visual feedback for this technique shows how an object is moved according to the position and orientation of the user's hand. A small solid cube is elastically attached to the hand model. The elastic relationship between the hand model and the object inherently indicates that the object will be released if the hand moves too fast. If the object is released in the current task, the cube is released in its current pose. If the object is grabbed in the current task, the cube jumps back to the hand model and is reattached. This indicates that an object will be released if the hand moves rapidly.

4.4 Changing interaction techniques/settings

In order to be able to switch between different manipulation tools or adjust application settings we chose a similar approach as used in standard interfaces, where the user can switch between a menu and manipulation mode. In the menu mode, the different interaction modes/settings can be selected/changed from a 3D toolbar, and in the manipulation mode the selected interaction mode is applied. To this end, the user can choose between two different techniques:

Working volume split: The 3D working volume is divided into a near region and a far region as depicted in Fig. 1(Right). If the user's hand is located in the near region, the menu mode is chosen. When the hand enters the far region, the selected interaction mode is applied. Thereby the far region is about four times larger than the near region. This way, switching between menu and manipulation mode is simple, but the available manipulation space is reduced and some distraction results from unintentional menu/manipulation transitions.

Free-hand gestures: The user switches between menu and manipulation mode by performing a certain free-hand gesture (we used a circle in the xy-plane), while her hand remains in the 'Not attached' state (see Sec. 4.3). This way, the whole working volume can be exploited for manipulation purposes, but switching between menu and manipulation is more difficult. As both solutions have their advantages and drawbacks, the user can choose between them. If for example a single long manipulation step is planned, she could select the second alternative and otherwise the first one. Additionally, we enable free-hand gestures for switching between interaction modes directly (e.g. between object manipulation and selection).

5 THE INTERFACE

When the user's hand is directly used as an input device for controlling an application it is extremely desirable that no other controller is involved during appliance to ensure immersive interaction. Therefore we designed a graphical user interface (GUI) that is fully controlled by the user's tracked hand. In our GUI a 3D scene is shown and several basic manipulation tools can be selected from a toolbar, when the menu mode is active (see Sec. 4.4). The user can choose a tool by moving a hand model such that it intersects the 3D object, which represents a tool (currently a labeled cylinder), and performing a roll click (see Sec. 4.2).

Our system is designed as a state machine, where the states are represented by the different manipulation tools. If a certain manipulation state is occupied and the manipulation mode is active, specific manipulations are applied to the selected objects (e.g. state 'Move' for translating and rotating objects) or the camera (in state 'Steer'). In the following the different available manipulation states are described.

In the 'Move' state the currently selected objects are translated and rotated according to the user's hand movements as long as they are grabbed, which is determined by the jerky release technique introduced in Sec. 4.3.

In the 'Scale' state the object is scaled up if the user moves her hand to the positive x-direction and down if she moves her hand to the negative x-direction. If the hand movement is classified to be fast and jerky according to the jerky release technique in Sec. 4.3 no scaling is applied. This way, the object can be scaled up or down arbitrarily in several cycles.

In the 'Select' state the two most common standard techniques (according to [PIWB98]), namely the virtual hand metaphor (an object can be selected when it collides with a virtual hand) and the virtual pointer metaphor (an object can be selected when it collides with a virtual ray emanating from the virtual hand) are available. Each technique is combined the roll click (see Sec. 4.2). When the virtual pointer metaphor is applied an additional ray is drawn in the visual feedback box illustrated as a simple line emanating from the hand model toward the pointing direction.

In the 'Steer' state the virtual camera can be moved. Thereby we decided to use the traditional approach of steering, which was introduced by Ware and Osborne [WO90] and named the flying vehicle control metaphor. The translational and rotational distances of the user's hand pose from a certain resting pose are used to control the translational and rotational velocities of the virtual camera to the according directions. In this state, several modifications are applied to the visual feedback box. Due to the lack of haptic feedback users have often problems to return their hands to the resting pose in order to stop moving the camera. To this end, in our visual feedback box the resting position is visualized by showing a static additional hand model. To further hint the currently applied camera translation, a 3D arrow originating from this static hand model and ending at the currently moving hand model is provided. The orientation is split into roll, pitch and yaw angle and the resting orientation is illustrated by three static lines on the back wall, side wall, and floor of the cube, respectively. Simultaneously, three rotating lines indicate the currently applied rotation. The area between each of the rotating lines and its according static line is shaded and two circular arrows are depicted, respectively. For an illustration we refer the reader to the accompanying video.

6 EXPERIMENTS

We conducted an experiment to evaluate the performance of the jerky release technique for moving and positioning virtual objects. Our hypothesis was that our method would be superior to other techniques commonly applied for hand-tracking devices. Moreover, we expected that a hand-tracking device combined with our technique had superior or at least similar performance compared to a standard 6 DOF controller.

To this end, we compared our technique to both the use of a grabbing posture (i.e. only if the subjects form the grabbing posture the virtual object moves according to her hand) and the use of a standard 3D mouse. As we are interested in complex interaction with the simultaneous manipulation of 6 DOFs we refrained from a comparison to a standard 2D mouse, because a 2D mouse would need specific interaction metaphors to enable the control over 6 DOFs. Several manipulation tasks had to be solved by using each controller, while the completion times and precisions were measured.

For the posture based grabbing and releasing the adjustments of Osawa [Osa06] for precise releasing were implemented. However, we could not use the same velocity threshold of $1\frac{cm}{s}$ as proposed in [Osa06], because depending on the velocity of the hand pose the accuracy of the determined thumb and front-fingertip positions were not sufficient (Osawa used data gloves, which have a high precision and whose global pose does not influence other hand parameters). Therefore, we had to use a higher threshold ($10\frac{cm}{s}$ in the experiments), which led to inferior releasing precisions.

6.1 Experimental setting

Two connected PC-based systems (Intel E6600, Geforce 8800) were used in the experiment, one coupled to the cameras for tracking the hand (see Sec. 3) and one for running the virtual environment application. The application was visualized on a standard 19" TFT-Display. Additionally, a 3D connexion SpaceNavigator was connected to the second PC.

Experimental tasks: In the experimental tasks a virtual object had to be approximately moved to a specified position (less than 2 units translational error) and/or orientation (less than 4 degrees rotational error) by using the different techniques/controllers. In the first task only translation had to and could be modified until the desired position was approximately reached. In the second task, the object's position was fixed and only orientation had to be modified. These two tasks were established in order to check if one of the techniques has specific advantages in either the rotational or translational DOFs. To check the performance for more complex tasks both the orientation and position had to be manipulated in the third task. These three tasks were used to measure the completion times, therefore, it was communicated to the subjects to be as fast as possible.

In the fourth and last task again orientation and position had to be modified, but the subjects could decide by themselves when the final pose was reached and then had to release the object by either pulling the hand out of the working volume (if the hand-tracking device was used) or pressing the left 3D mouse button (if the 3D mouse was used). No snapping algorithm (the object snaps to the desired pose, when it is near by) was applied. This task was used to determine the positioning errors. For this reason, the subjects were advised of being as accurate as possible for this task.

Participants: Eight participants (one female, seven males, all university students) took part in the experiment. They had little or no virtual reality experience.

Procedure: Each participant had to solve all four tasks four times by employing each of the three controllers (3D mouse, hand-tracking with grabbing posture and hand-tracking with our technique). Thereby, the sequence of the employed controllers was permuted evenly and all tasks had to be finished until the next controller was adopted. Before starting the test for each controller, its mode of action was explained and the subjects could familiarize with it in a short preparation time (two minutes).

6.2 Results

The average task completion times including standard deviations are depicted in Fig. 5. Employing the grabbing posture was clearly inferior to our technique or the 3D mouse. This is mainly because some time is needed for switching the postures. If the object's orientation is manipulated, this becomes even more relevant, because more grab and release cycles are needed due to the little space of anatomical rotational freedom.

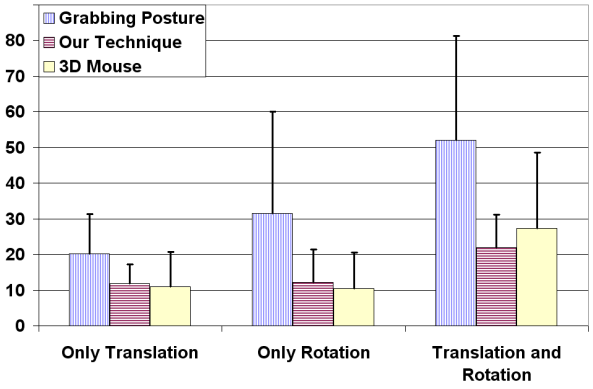

Figure 5: Task completion times (in seconds).

Considering both the times of our technique and the 3D mouse it can be seen that the 3D mouse performs slightly better if the amount of degrees of freedoms is restricted, but inferior if all 6 DOFs are available. This was confirmed in our observations during the experiments. The simultaneous control of several DOFs was significantly more difficult with the 3D mouse.

In Fig. 6 the mean errors and their standard deviations for translational and rotational positioning of the virtual object are depicted. This diagram shows, that our technique is suitable for precisely releasing virtual objects. The bad results for using the grabbing posture are caused by overhasty releasing while the hand was still attached to the virtual object.

6.3 Discussion

After the user experiments, the subjects could practice with the whole interface and we quested them for their subjective impression concerning the visual feedback box. Most subjects told us that the proposed visual feedback box (see Sec. 4.1) with its modifications for different interaction modes helped them notably for familiarizing with the respective interaction mode and for ensuring that their handling of the hand-tracking device is correct (e.g. whether the hand is still in the working volume and forms the correct posture). In particular for supporting the steering of the camera, we got a clear positive feedback, what we think results from the less intuitive handling of the steering technique. Note that more objective testing of the visual feedback goes beyond the scope of our work, because as it supports mainly the familiarization with an interface, we can not compare the performance with and without it for one and the same subject. However, the timings and precision strongly depends on the individual, so an objective study would need a great many of subjects.

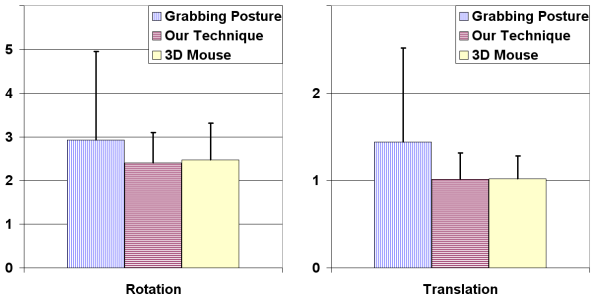

Figure 6: Positioning errors. In degrees for the rotational error. The translational error can only serve as a relative measure as it depends on the adopted mapping from real to virtual space.

The roll click (see Sec. 4.2) could instantly be handled by everyone. Due to the proposed pointing direction modifications very precise object selection operations could be performed even if the virtual pointer metaphor was used. Note that these modifications are only applicable, because the roll-axis angle is used for triggering the button events.

For moving objects by using the jerky release technique (see Sec. 4.3), most users needed a short adaptation phase until they developed a sense for the different kinds of motion (smooth movements for moving the object and fast/jerky for releasing it). But subsequently, they could easily perform different complex tasks.

Obviously, a limitation of the grabbing and releasing technique is the fact that a virtual object can not be moved fast and jerky any more. However, in practice such movements are utilized rarely for manipulation tasks. To quantify this problem, we analyzed the movements of both hand and virtual object in our user experiments for the cases that the grabbing posture was employed for grabbing and releasing instead of our technique. We computed the percentage of virtual ob-

ject movement that occurred while the 'Not attached' state would have been occupied, if our technique would have been used. On average, less than 5% of the virtual object movements would have been filtered out by our technique. Moreover, we observed such movements often to be unintended by the subjects, for example if the virtual object should be released by switching from the grabbing to the standard posture but moving the hand overhasty while the object is still attached to the hand.

7 CONCLUSIONS AND FUTURE WORK

We presented novel techniques for 3D interaction particularly suitable if markerless hand-tracking is employed. These techniques were integrated into a novel user interface and tested by many people. We showed that the proposed technique for grabbing and releasing enables efficient manipulations and precise releasing. A short user study was presented to compare this technique to traditional controllers and techniques.

In the future, we want to integrate further manipulation tools into our interface such as object deformation. Thereby, the visual feedback box will also be adapted.

Furthermore, we wish to improve the immersiveness of our interface by using additional tracking technology as for example head or gaze tracking. Gaze tracking could additionally be exploited for investigation of the usefulness of the visual feedback; it could be quantified how much time the user spends on looking at the visual feedback box in certain situations.

REFERENCES

[BH97] Doug A. Bowman and Larry F. Hodges. An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In *SI3D '97: Proceedings of the 1997 symposium on Interactive 3D graphics*, pages 35–38, New York, NY, USA, 1997. ACM.

[BI05] Christoph W. Borst and Arun P. Indugula. Realistic virtual grasping. In *VR '05: Proceedings of the 2005 IEEE Conference 2005 on Virtual Reality*, pages 91–98, Washington, DC, USA, 2005. IEEE Computer Society.

[BKLP04] Doug A. Bowman, Ernst Kruijff, Joseph J. LaViola, and Ivan Poupyrev. *3D User Interfaces: Theory and Practice*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 2004.

[Bol80] Richard A. Bolt. "put-that-there": Voice and gesture at the graphics interface. *SIGGRAPH Comput. Graph.*, 14(3):262–270, 1980.

[BRC05] Joan De Boeck, Chris Raymaekers, and Karin Coninx. Are existing metaphors in virtual environments suitable for haptic interaction. In *Proceedings of Virtual Reality International Conference*, pages 261–268, 2005.

[DH07] Raimund Dachselt and Anett Hübner. Virtual environments: Three-dimensional menus: A survey and taxonomy. *Comput. Graph.*, 31(1):53–65, 2007.

[FK05] Scott Frees and G. Drew Kessler. Precise and rapid interaction through scaled manipulation in immersive virtual environments. In *VR '05: Proceedings of the 2005 IEEE Conference 2005 on Virtual Reality*, pages 99–106, Washington, DC, USA, 2005. IEEE Computer Society.

[GFF+04] C. Grätzel, T. Fong, T. Fong, S. Grange, and C. Baur. A non-contact mouse for surgeon-computer interaction. *Technol. Health Care*, 12(3):245–257, 2004.

[GWB05] Tovi Grossman, Daniel Wigdor, and Ravin Balakrishnan. Multi-finger gestural interaction with 3d volumetric displays. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, pages 931–931, New York, NY, USA, 2005. ACM.

[JS07] Alejandro Jaimes and Nicu Sebe. Multimodal human-computer interaction: A survey. *Comput. Vis. Image Underst.*, 108(1-2):116–134, 2007.

[MF04] N. Murray and T. Fernando. An immersive assembly and maintenance simulation environment. In *DS-RT '04: Proceedings of the 8th IEEE International Symposium on Distributed Simulation and Real-Time Applications*, pages 159–166, Washington, DC, USA, 2004. IEEE Computer Society.

[MFPBS97] Mark R. Mine, Jr. Frederick P. Brooks, and Carlo H. Sequin. Moving objects in space: exploiting proprioception in virtual-environment interaction. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 19–26, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[Osa06] Noritaka Osawa. Automatic adjustments for efficient and precise positioning and release of virtual objects. In *VRCIA '06: Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications*, pages 121–128, New York, NY, USA, 2006. ACM.

[PBWI96] Ivan Poupyrev, Mark Billinghurst, Suzanne Weghorst, and Tadao Ichikawa. The go-go interaction technique: non-linear mapping for direct manipulation in vr. In *UIST '96: Proceedings of the 9th annual ACM symposium on User interface software and technology*, pages 79–80, New York, NY, USA, 1996. ACM.

[PIWB98] I. Poupyrev, T. Ichikawa, S. Weghorst, and M. Billinghurst. Egocentric object manipulation in virtual environments: Empirical evaluation of interaction techniques. *Computer Graphics Forum*, 17(3):41–52, 1998.

[PSP99] Jeffrey S. Pierce, Brian C. Stearns, and Randy Pausch. Voodoo dolls: seamless interaction at multiple scales in virtual environments. In *I3D '99: Proceedings of the 1999 symposium on Interactive 3D graphics*, pages 141–145, New York, NY, USA, 1999. ACM.

[SCP95] Richard Stoakley, Matthew J. Conway, and Randy Pausch. Virtual reality on a wim: interactive worlds in miniature. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 265–272, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

[SKSK07] M. Schlattmann, F. Kahlesz, R. Sarlette, and R. Klein. Markerless 4 gestures 6 dof real-time visual tracking of the human hand with automatic initialization. *Computer Graphics Forum*, 26(3):467–476, September 2007.

[VB05] Daniel Vogel and Ravin Balakrishnan. Distant freehand pointing and clicking on very large, high resolution displays. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, pages 33–42, New York, NY, USA, 2005. ACM.

[WO90] Colin Ware and Steven Osborne. Exploration and virtual camera control in virtual three dimensional environments. In *SI3D '90: Proceedings of the 1990 symposium on Interactive 3D graphics*, pages 175–183, New York, NY, USA, 1990. ACM.

[WP03] Andrew Wilson and Hubert Pham. Pointing in intelligent environments with the worldcursor. In *INTERACT*, 2003.

[Zac00] G. Zachmann. *Virtual Reality in Assembly Simulation – Collision Detection, Simulation Algorithms, and Interaction Techniques*. PhD thesis, Darmstadt University of Technology, Germany, Department of Computer Science, may 2000.