EG UK Theory and Practice of Computer Graphics (2009)
Wen Tang, John Collomosse (Editors)

Markerless visual human movement tracking for HCI: what frequency?

Ferenc Kahlesz and Reinhard Klein

Computer Graphics Group
Universität Bonn
Römerstr. 164, 53117 Bonn, Germany
{fecu,rk}@cs.uni-bonn.de

Abstract

This paper tries to establish a minimal tracking frequency limit for visual human movement tracking algorithms that intend to be useful for the realization of some kind Human-Computer-Interaction (HCI) metaphor. More specifically, we examine the question of this minimal frequency for Augmented/Virtual Reality (AR/VR) navigation and 3D object manipulation. We approach the question from three different perspectives: shortly reviewing non-visual and visual marker-based solutions integrated regularly into AR/VR systems, spectral analysis of human movement and latency implications for AR/VR settings. Finally, we conclude the paper by combining and discussing the results from these different areas. We find that tracking with update rates as low as 12.5Hz can provide a usable basis for interaction. The most important message of the paper is that stable and working (even if slow, when compared to other techniques) markerless tracking algorithms are desperately needed because only working online with and based on such systems can the pros and cons of markerless tracking be evaluated.

Categories and Subject Descriptors (according to ACM CCS): I.3.6 [Computer Graphics]: Interaction Techniques I.3.7 [Computer Graphics]: Virtual Reality I.4.8 [Image Processing and Computer Vision]: Tracking

1. Introduction

Markerless visual human motion tracking has been a main area of interest in computer vision already from the beginning of the 1990s [EBN*07] [MHK06]. Applications that could possibly benefit from advancements in the field include surveillance, medical analysis, motion capture, AR/VR and other HCI settings and telerobotics. Unlike other tracking methods, including visual, but marker-based approaches, markerless techniques have the promise to free the user, patient or actor of cumbersome marker-placement procedures. In some of the target applications, most prominently automated surveillance, markerless tracking could prove to be the only feasible solution.

Due to the complexity of pose estimation of highly articulated or deformable objects (for which the human body or hand are perfect examples) apart from a few exceptions, results reported in literature usually cannot claim "real-time" performance. For applications that inherently allow offline processing (e.g. medical analysis or motion capture) this does not pose a problem and the advantages of markerless tracking mentioned above are easier to put at disposal.

In case of other applications, most notably HCI, offline evaluation is out of question, as "offline interaction" simply does not exist. These can be considered to have "real-time" requirements. But what does exactly "real-time" mean? Is it 25-30Hz, because the framegrabber or the camcorder connected to the computer provides these framerates? Or must it be well above 50-100Hz as is provided by most of the established non-markerless/non-visual tracking methods and suggested by some in the human movement research community [Ste04]? Or can it be as low as 10Hz, as according to animation compression literature, human motion is vastly dominated by low-frequency components [LOiT08] [AHM02] and according to the Nyquist-Shannon criterion sampling with twice of the baseband-width allows signal reconstruction? Or is waveform-preserving sampling necessary, thus up to 3-10 times [MT05] of the Nyquist-frequency is required, putting the estimate about 50-100Hz?

DOI: 10.2312/LocalChapterEvents/TPCG/TPCG09/133-140

Although HCI as target application is often cited as motivation for papers on markerless tracking systems and – as described above – a lot of recommendations exist for other fields, we did not find in the literature any required minimal frequency guidelines for building usable tracking systems for HCI. Therefore, we try to fill this gap via a requirement analysis of tracking update rates for two basic interaction scenarios for AR/VR applications: navigation and 3D manipulation of objects. The problem of the minimal update rate is examined from three different points of views: surveying what other, commercially available tracking methods are used by VR system engineers as building blocks; examining the spectral density of human movement in navigation and manipulation tasks and finally giving an insight into the latency problem in VR.

Contributions of the paper are the following: first, we find that low update rate algorithms ($\sim$12.5Hz) can have the potential for interaction. This statement will be arrived at by frequency domain analysis of the dataset presented in [LaV03], which is freely downloadable from the Internet. The motions contained in the dataset are typical in VE settings: navigation and object manipulation. However, examining the problem of latency in VEs reveals that 80ms lag induced by 12.5Hz tracking is intolerable, therefore a combined tracking and motion prediction framework is proposed to yield usable update rates.

It is obvious that working markerless trackers are needed, because unless such systems are available, potential user studies cannot be carried out and it cannot be evaluated whether the additional comfort of not needing markers is worth the effort. Unfortunately, due to the computational complexity of the problem to be solved, regarding speed markerless tracking currently cannot be on par with other methods. However, our findings suggest that – at least for the purpose of interaction with VEs, if not for general motion capture applications –, it might not be necessary to provide the same tracking rates as non-markerless trackers.

The rest of the paper is organized according to the three main aspects mentioned previously, followed by a discussion that led to the above conclusions. We also identify application setups that would greatly benefit from the availability of markerless tracking solutions.

2. Non-markerless tracking

Tracking is considered as a building block in Virtual Environments. When building a VE, system engineers concentrate on integrating via custom application code the different subsystems, basically tracking, physical simulation, rendering and display into a coherent application. The interplay between these blocks and the user then realize an interaction metaphor with the virtual environment that allows the whole to fulfill its purpose. Clearly, as the available technologies for the different building blocks are diverse, this gives rise to a large variety of VEs, ranging from desktop "Fish Tank VR" to room-sized CAVES. As tracking is considered a building element, rather than the goal, usually some commercially available solution is chosen.

Such tracking systems have clearly defined interfaces (e.g. RS-232) and protocol to communicate the tracking results or have access APIs that hide the communication details to the host computer. VR input device access frameworks (e.g. OpenTracker [RS01]) provide either own drivers for these trackers or wrappers for company supplied APIs. VE development toolkits (like VR Juggler [BJH*01] or FlowVR [LR08]) then in turn utilize device access frameworks, which allows easily integrating a tracking subsytem into VE applications or changing between different tracking sytems, much like a mouse is available for the desktop.

Available tracking methods can be categorized according different taxonomies, like target application type (e.g. interactive visualization and design or virtual assembly/maintenance training) [WF02], sensor and marker/source placement (sensor or marker placed on the user) [ZH04] and the type of physical phenomenon used for tracking (e.g. magnetic or inertial) [Fox02]. As we are interested in the achievable tracking frequencies and this is correlated with the physical phenomenon used for tracking, we also choose this for categorizing different trackers. Table 1 shows the three main wide-spread categories, typical update rates and references to manufacturers, without the aim of being exhaustive. Also, though we are interested in framerates, these are not the qualitative feature of tracking systems [Fox02]. For an excellent introductory material on tracking for VEs, please see [WF02].

method	typical update (Hz)	example
electromagnetic	120-240	[POL09] [Asc09]
acoustic + inertial	180	[Int09]
visual	30-2000	[Gmb07] [Sys07] [3rd09]

Table 1: *Different tracking methods and typical capabilities.*

3. PSD analysis of human motion data

Although there are several free motion capture databases available on the Internet, we have chosen to use the one described in [LaV03]. This database contains 16 real-life recordings of 6DOF head and hand motion during navigational and spatial interaction tasks in AR/VR settings. Each record is about 20 second in length, sampled at about 215Hz and resampled about 1KHz rate to provide ground truth data for predictive tracking algorithm development. For more information, please see the cited paper. Though other datasets provide much more data, both in amount and also in diversity, this dataset fits perfectly with the scenario that we want to examine. Moreover, the analysis presented here can be easily carried out for other motion datasets.

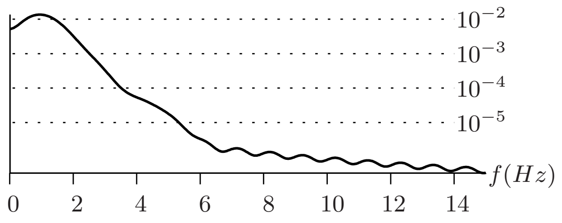

Figure 1: *Typical PSD for the [LaV03] dataset.*

We estimated the Power Spectral Density (PSD) of all the translation and rotation data from the database using the Welch method [Wel67] as implemented in GNU Octave [EBH08] Forge. The averaged periodograms corresponded to about 1 second human movement (1050-1060 samples) and FFT was carried out on the Hamming-windowed data with double sized zero padding. The mean has been removed from every segment used for periodogram estimation. This resulted in PSDs with frequency resolution of about 0.25Hz. The PSD shown in Fig. 3 was typical: the bulk of the energy clearly concentrated well below 20Hz. Please note that as the datasets are sampled above 1kHz, the PSD frequency-axis contains frequencies up to above 500Hz, still the meaningful parts of the PSD histogram are below 10Hz. These results are well in sync with to the ones presented in [AHM02]. Actually, the spectral content above 6Hz is probably due to the spectral leakage caused by windowing.

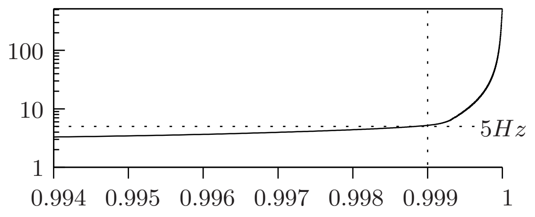

Figure 2: *The normalized total energy (x-axis) and frequency range it is contained in (y-axis, logarithmic).*

Finally, all the PSDs were combined into one common histogram, where the bins were corresponding to total signal energy percent and the values of the bins held the spectral component needed to contain the amount of energy represented by the bin. This combination was done using a "worst case" approach: all the PSDs of all the 16 datasets with the 6DOFs each were examined and the highest frequency value was selected for the bin. Fig. 2. As it can be seen, almost all of the energy is contained up to 5Hz. Table 2 shows our findings about the amount motion energy that can captured using different kinds of off-the-shelf cameras. As it can be seen, almost all power can be captured using a simple webcam.

camera type	framerate	captured energy (%)
webcam	12.5	99.88
camcorder PAL	25	99.94
camcorder NTSC	30	99.95
DCAM	60	99.97
	120	99.98

Table 2: *Different tracking methods and typical capabilities.*

4. Latency in VEs

The responsiveness of a VR system is determined by its end-to-end *latency* or *lag*: the delay between the user's input action and the reaction to it presented by the system, *i.e.* rendering the scene from a new viewpoint upon head-movement or updating the 3D cursor position upon hand-movement. As some kind of processing happens every time between the user's action and the rendered result, lag is inherently present in any VE. In this section, latency will be examined with a bit bias toward trying to answer the question: how fast should a hand-tracking system be in order to be usable in a VE?

4.1. Negative effects

The problem with latency is that it introduces a contradiction between the proprioceptic and the exteroceptive senses of the user. The most apparent result of such contradiction is visual oscillopsia [AHJ*01], when the virtual world seems to float around or oscillate in space; this is caused by the mismatch of the actual sight shown by the renderer and expected sight based on *e.g.* the motion sensors in the inner ear in case of head movement or the expected position of one's hand reported by the "bodypart-postion-sensors". Apart from visual mismatch, auditory or tactual oscillopsia can be considered in a VE setting, too.

The effects of system latency have been extensively examined in psychophysical studies [MW93] [WB94] [SMK98] [AHJ*01] [MAEH04] [RM07]. Basically, these experiments can be divided into two categories: the first type of experiments are designed to explicitly investigate the connection between the absolute amount of latency and some kind of absolute measure of loss of simulation fidelity (the feeling of presence) or user performance degradation induced by the latency [MW93] [WB94] (usually the increase in the time needed to position or select objects or the decrease in precision of these tasks given a fixed amount of time to accomplish them). The second kind of studies analyze what amount of relative latency changes can be detected by the user [AHJ*01] [MAEH04] [EMAH04].

The first kind of studies concerning a hand-based manipulation task have shown that lag gradually degrades the performance in positioning a 2D cursor [MW93] or in reaching 3D positions in VE [WB94]. Longer exposure to VEs with relatively large latency (*ca.* $200-1000$ms) can even lead to simulator- or cybersickness [SMK98] (similar symptoms as motion sickness). Fortunately, to some degree, the human brain can learn to use VE systems that are not as responsive as accustomed to from everyday experiences. In VEs with latency, users can adopt a "move-and-wait" strategy [MAEH04] for object manipulation tasks (which, of course, increases the time needed to accomplish the given task) or can exhibit sensorimotor adaptation to the delay (*e.g.* turning a car earlier than in the real world, *c.f.* the drive-simulation investigated in [CCvdHB01]). Such adaption can be rapid if latency is constant or can not happen at all with variable lag [SMK98]. This is why the second kind of latency experiments are significant, because they give hints to VE system engineers about with what tolerance should latency be tuned.

4.2. Sources of lag

Fig. 3 depicts the sources of lag in a typical VE pipeline. T_{track} is the lag introduced by the tracking system itself, this is the time the tracker needs to be aware of changes in the tracked state and update it, *i.e.* computing the new state from the observations. T_{comm}^{track} time elapses during the communication of the updated state to the application, this can vary from a simple DMA transfer from a periphery to sending data on some platform, *e.g.* USB or Ethernet. After acquiring the tracking information, the application has to update itself accordingly, *e.g.* to run physical simulation of the virtual world, this is T_{sim}. Then the new configuration of the world is communicated to the rendering engine, this takes T_{comm}^{sim} time. Once again, as is the case with T_{comm}^{track}, T_{comm}^{sim} can widely vary if an actual VE system is considered: if simulation and rendering happens on the same computer, it is simply the delay introduced by the data-transfer through the system bus from memory to the graphics HW (which can be considered negligible), if distributed rendering is needed *e.g.* for a Powerwall [CPS*97], this time can include network transfer delays. T_{render} in the VE pipeline decomposition depicted in Fig. 3 means the actual time needed by the graphics HW to render the updated scene and T_{update} is the amount of time that elapses until the display is actually updated. Of course, if multidisplay rendering is considered, both T_{render} and T_{update} are the sums of the delays introduced by the individual rendering HWs.

f_{track}, f_{sim} and f_{render} in Fig. 3 denote the execution frequencies or *throughput* [Wlo95] of the different stages and are meant to emphasize the fact that the blocks of the pipeline execute concurrently – of course, simple applications can execute the stages sequentially (in this case the tracking device is polled in the beginning of a track-simulate-render cycle), more sophisticated VEs, however, have to employ multiple CPUs/CPU cores (possibly connected over network) and exploit parallelism in order to tackle computational challenges.

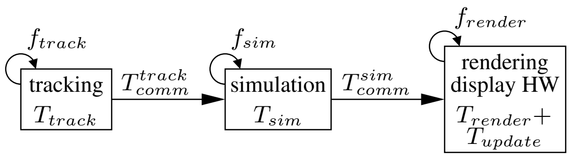

Figure 3: *Sources of lag in a typical track-simulate-render VE pipeline. $T_\star$ are the processing/transfer-times of the pipeline stages, $f_\star$ are the execution frequencies.*

4.3. Lag and throughput

Though the corresponding $T_\star$ and $f_\star$ are related, it does not necessarily hold that $f_\star = \frac{1}{T_\star}$, at least not for tracking; an example for this is an optical tracker with $f_{track} = 12.5$Hz images per second sampling rate from a webcam that requires only $T_{track} = 5$ms for image processing (and not $\frac{1000\text{ms}}{12.5\text{Hz}} = 80$ms). Moreover, the pipeline stages other than tracking can have an additive extra waiting time, if the preceding stage does not produce results fast enough, *i.e.* if the stage has to wait for its input data. This is called *synchronization lag* in [Wlo95]. In order to avoid this kind of lag, $f_{track} \geq f_{sim} \geq f_{render}$ should hold [MJvR03], however, if the simulation is computationally expensive, it can be advantageous to also directly "short-cut" tracking data to the rendering stage and thus maintain a low-lag cursor/viewpoint feedback independent from the simulation state [Wlo95].

Although not explicitly present in Fig. 3, the data-throughput of the communication channels – that affect T_{comm}^{track} and T_{comm}^{sim} – must be high enough to avoid frame-drops. The total throughput or *framerate* of the system cannot be better than the weakest link in the chain and is thus $\min(f_{track}, f_{sim}, f_{render})$. It is important to maintain a "high-enough" total system framerate, because low framerates introduce a sample-and-hold artifact. This artifact has similar psychophysical effects as lag [Wlo95], strictly seen, however, low total system throughput is a related, but distinct problem: using the webcam-based optical tracker from the example above on a single computer ($T_{track} = 5$ms, $T_{comm}^{track} = T_{comm}^{sim} \approx 0$ms) in a simple scene setting ($T_{sim} = T_{render} = 5$ms) with 120Hz update rate monitor ($T_{update} = 8.3$ms), system lag is 23.3ms, but system throughput is determined by $f_{track} = 12.5$Hz (and is not $\frac{1000\text{ms}}{23.3\text{ms}} = 42.9$Hz).

As low framerates induce lag-like effects, high framerates are important. It should be noted, however, that high framerate does not mean low ($\frac{1000}{\text{system framerate}}$ms) lag. This is illustrated in Fig. 4, where a system maintains a constant 100Hz total throughput in a pipelined configuration and still exhibits 60ms latency (not $\frac{1000\text{ms}}{100\text{Hz}} = 10$ms). This 60ms is based on the premise of perfect synchronization, if some pipeline stages are implemented by multithreading, additional delays can be expected, because the wake-up time of the threads depends on the OS scheduler policy. The same system in a non-pipelined operation mode would maintain the same lag (apart from possible scheduling delays), but system framerate would be only $\frac{1000\text{ms}}{60\text{ms}} = 16.6\text{Hz}$.

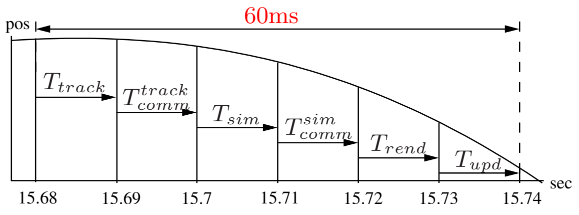

Figure 4: *Considering a conceptual system where all the lag sources are* 10*ms, the concurrently pipelined VE runs at* 100*Hz, but the environment at* 15.74 *(when the user sees it) is rendered according to the tracking results at* 15.68*.*

4.4. Combating lag

When designing a VE, it is clearly advantageous to select pipeline elements that induce small lag (faster tracker, simulation and rendering and communication channels) and also to appropriately minimize the synchronization lag between the pipeline steps [Wlo95] [MMD*08]. However, no matter how carefully the subsystems are selected and synchronization is designed, some latency will remain.

The effects of this remaining lag can be alleviated using predictive tracking. The idea is to somehow model the movement of the user (*e.g.* through some polynomials), fit this model to past observations and extrapolate her movement from the fitted model into the future when the image will be displayed. This predicted tracking result is used for the simulation and when the scene is rendered it will hopefully according to the user's state at the actual time of rendering.

Of course, the look-ahead time for the prediction must be known. This is usually measured in carefully timed trial-runs of the system. It is also clear that the smaller the look-ahead time is, the better (or rather, less erroneous) the prediction will be. *If the lag is constant* and the user's motion is not "too fast", look-ahead times up to 100ms can be successfully handled. In Distributed Virtual Environments (DVE), however, where users participate over large distances (*i.e.* over the Internet), network lag can easily reach 1000ms [CLL08]. Moreover, DVEs are much more exposed to the problem of *non-constant* lag, though locally installed VEs has this problem, too, as measurements of *e.g.* thread/pipeline-node synchronization delays cannot 100% be relied upon. The problem of non-constant lag can be tackled by predicting the lag (could be used both in VEs and DVEs); such approaches were presented in [AB94] and [TAS07], respectively.

Despite the mentioned difficulties, lag masking via prediction is a viable and useful tool to cope with latency in VEs and VE system-engineers are using this technique successfully to provide a better user experience or to reduce lag to an affordable level.

5. Discussion

Unfortunately, though the negative effects are well documented in human surveys, few explicit statements can be made about what latency or system framerates are still acceptable and what are not.

[THC*96] designates a 100 – 1000ms lag range as tolerable. [Wlo95] states that in a VR setting, lags higher than 300ms destroy presence, [MRWFPB03] indicates that 120ms are unacceptable, [PSLJ99] allows at most 40ms lag in order not to be noticed by the user. Regarding lag in a hand-tracking scenario (interactive motor-sensory tasks), 75ms lag has easily measurable lag, while 225ms degrades performance substantially [MW93]. In AR applications using an optical see-through display, however, perceived lag must be less than 30ms according to [Wlo95], or even lower: 10ms or 5ms [PSLJ99].

Regarding framerates, there is a consensus in the VR community that at least 10Hz is required to achieve "smooth-enough" animation [MW93] [MJvR03]. As stated previously, in order to avoid synchronization (or data-wait) lags, this value in itself would allow 10Hz sampling rate for the tracking device, but this does not actually mean that 100ms processing time is acceptable, because that would mean at least 100ms lag, not counting the processing delay of other stages. Moreover, advances in graphics accelerators today allow for rendering framerates above 100Hz (depending on scene complexity and rendering quality), which would suggest a tracking cycle less than 10ms.

Humans' adaptation capabilities to latency complicates giving a clear answer to the question: what latency is tolerable? Another difficulty is that there is more to human performance issues than only latency or frame-rate: field-of-view, image-quality, fewer depth cues in VR [SPB06]. The vast amount of interrelated ergonomical problems is clearly illustrated in [CHB07], a relatively recent survey of more than 150 papers about diverse human performance issues, including also latency. Though the main focus of this paper is robot-teleoperation, this topic is closely related to VEs and in fact, the paper cites VE user studies.

Concerning the starting question of this section, how fast should tracking be, the only answer that can be given is that as tracking is part of the VE pipeline, a general purpose tracker should be as fast as possible. Tracking limits the available rendering framerate, thus 100Hz is better than 30Hz. Tracking with 100Hz with 1ms latency is better than 100Hz with 10ms. Of course, for a specific application (given interaction metaphor, display type, *etc.*) concrete requirements can be elaborated.

5.1. A combined tracking and motion prediction framework

The results of the PSD analysis clearly indicate that capturing motion at 12.5Hz satisfies the Nyquist-criterion of signal reconstructability. This means, that even at these tracking framerates, enough information is captured about the user's movement. This information then could be fed into a black-box that predicts the user's motion until the next update based on motion modeling techniques. This way, slow tracking is decoupled from the problem of lag. Different interaction modes (*e.g.* fine object manipulation, navigation or interaction with dialog boxes, buttons and menus) could be modeled independently and the appropriate model could be selected based on the user's hand/head position and the state of the application. Such submodels of motion could be trained using training data collected with established, high-speed tracking methods. The rest of the VE would be able to poll such a blackbox for motion data at any point in time (*c.f.* Fig. 5).

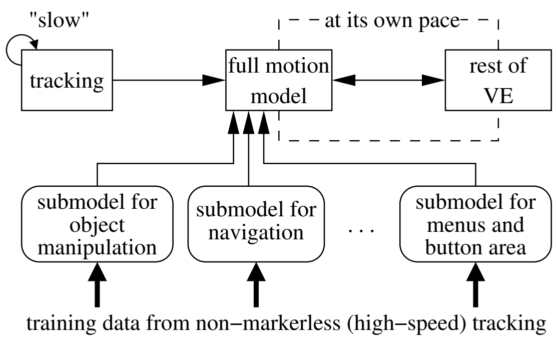
"slow"
tracking
at its own pace
full motion model
rest of VE
submodel for object manipulation
submodel for navigation
. . .
submodel for menus and button area
training data from non–markerless (high–speed) tracking

Figure 5: *Decoupling tracking from the rest of the VE.*

Of course, if tracking frequency is higher than the bare minimum required, it automatically ensures lower latency and also makes it easier to implement the predictor, as lookahead times are smaller. However, two important points to make are that:

- Using the proposed predictive setup low tracking framerates are allowable (and an tracking system in itself can use up all processing time available between frames).
- The proposed predictive setup makes image processing algorithms independent of the motion modeling logic.

In our opinion, using a combination of stable, albeit low framerate tracking and reliable human motion modeling and prediction can lead to stable markerless tracking systems. This setting has the advantage that markerless motion tracking could inherently benefit from independent advances is both fields, let it be faster tracking or better modeling.

5.2. Specific VE settings that could benefit from markerless tracking

At exhibitions, technology fairs or in museums, where there are a large number of users who interact with some VE for a short amount of time, it is definitely advantageous not to require users to spend extra time with putting on and of hand-tracking devices. Another aspect in such settings are the hygienic benefits.

Another important possible application area are VE settings where head-tracking is not necessarily required: visualization for a group of users on a Powerwall (*e.g.* for design reviews in the automotive industry) and one glasses-based person desktop VR. Though HMDs are not used in these VEs, the user(s) in these cases still have to put on shutter, polarized or anaglyhp glasses for 3D rendering, but these glasses do not need to be tracked. As glasses require only a matter of seconds to put on or off, extra inconvenience of wearing and unwearing hand-tracking devices is clearly disadvantageous.

There are also VEs where head tracking is solved in itself. Unlike when tracking is solved by *e.g.* EM technology, in these settings it is not possible to simple attach another sensor to the tracking system and thus provide pose tracking for the hand. Examples for this kind of displays include the now discontinued BOOM hands-free display [HF08] or the upcoming rotation tracked 3D glasses, like CyberfaceX from LeepVR [Cyb08]. Doing only rotation tracking perfectly fits the use case of a desktop VR system, when the user sits still (like before a computer monitor) and only rotates her head but the position is approximately constant. The advantage of this setting is that sourceless (no external sensor or calibration needed) 3D rotational tracking is feasible and is probably the way for consumer HMD devices [WF02]. For this "head-tracking solved in itself" VEs bare handed interaction has the same advantages as in the no head-tracked case.

Finally, and most importantly, there are VE applications where the users do not need to wear any glasses. The simplest example for this are desktop VR systems where 3D rendering is done like in ubiquitous 3D computer games. The more important case is, however, when holographic or other automultiscopic displays are used: these displays provide true 3D images for the naked eyes of the users. Although this kind of visualization is currently emerging technology, there already exist a number of commercially available products and this kind of visualization is clearly the future. Having the user only to use her bare hands for interaction with bare-eyes visualization in exhibition-like settings or other environments, without any pre- or post-hand-interaction requirements, as long as force-feedback is not needed, has without doubt unmatchable benefits.

6. Conclusions

As it has been shown via PSD analysis, tracking in a VE setting with just 12.5Hz would capture all important aspects of the user's motion. This would still allow for the bare minimum rendering framerate 10Hz, however, latency would probably introduce problems. Therefore, we also proposed a framework that combines motion modeling and stable tracking and is able to compensate for the slow tracking update rate. For such a setting to work, motion modeling independent from visual tracking should in itself be considered an important research field that can contribute to markerless tracking development.

Due to the complexity of the problem, creating a stable markerless tracker with automatic initialization that work at the same tracking rates as non-markless trackers seems to be infeasible with state-of-the-art algorithms. Our findings suggest that – for the purpose of interaction – trying to be on par with non-markerless methods might not be the right goal to set and it could be possible to get away with much lower framerates.

References

[3rd09] 3rdTech: http://www.3rdtech.com/. WWW, March 2009.

[AB94] Azuma R., Bishop G.: Improving static and dynamic registration in an optical see-through hmd. In *Proceedings of the 21st Annual Conference on Computer Graphics and interactive Techniques SIGGRAPH ACM* (NY, USA, October 1994), pp. 197–204.

[AHJ*01] Allison R. S., Harris L. R., Jenkin M., Jasiobedzka U., Zacher J. E.: Tolerance of temporal delay in virtual environments. *vr 00* (2001), 247.

[AHM02] Ahmed A., Hilton A., Mokhtarian F.: Adaptive compression of human animation data. In *Eurograhics - Short Paper* (Sep 2002).

[Asc09] Ascension C.: http://www.ascension-tech.com/. WWW, March 2009.

[BJH*01] Bierbaum A., Just C., Hartling P., Meinert K., Baker A., Cruz-Neira C.: Vr juggler: A virtual platform for virtual reality application development. In *VR '01: Proceedings of the Virtual Reality 2001 Conference (VR'01)* (Washington, DC, USA, 2001), IEEE Computer Society, p. 89.

[CCvdHB01] Cunningham D. W., Chatziastros A., von der Heyde M., BÃijlthoff H. H.: Driving in the future: Temporal visuomotor adaptation and generalization. *J. Vis. 1*, 2 (10 2001), 88–98.

[CHB07] Chen J. Y. C., Haas E. C., Barnes M. J.: Human performance issues and user interface design for teleoperated robots. *IEEE Transactions on Systems, Man, and Cybernetics, Part C 37*, 6 (2007), 1231–1245.

[CLL08] Chan A., Lau R., Li L.: Hand motion prediction for distributed virtual environments. *Visualization and Computer Graphics, IEEE Transactions on 14*, 1 (Jan.-Feb. 2008), 146–159.

[CPS*97] Czernuszenko M., Pape D., Sandin D., DeFanti T., Dawe G. L., Brown M. D.: The immersadesk and infinity wall projection-based virtual reality displays. *SIGGRAPH Comput. Graph. 31*, 2 (1997), 46–49.

[Cyb08] CyberfaceX L.: http://www.leepvr.com/15cyberfacex.html. WWW, December 2008.

[EBH08] Eaton J. W., Bateman D., Hauberg S.: *GNU Octave Manual Version 3*. Network Theory Ltd., 2008.

[EBN*07] Erol A., Bebis G., Nicolescu M., Boyle R., Twombly X.: Vision-based hand pose estimation: A review. *CVIU 108*, 1-2 (October 2007), 52–73.

[EMAH04] Ellis S. R., Mania K., Adelstein B. D., Hill M. I.: Generalizeability of latency detection in a variety of virtual environments. In *Human Factors and Ergonomics Society 48th Annual meeting* (2004).

[Fox02] Foxlin E.: Motion tracking technologies and requirements. In *Handbook of Virtual Environment Technologies*, Stanney K. M., (Ed.). Lawrence Erlbaum Associates, 2002, pp. 163–210.

[Gmb07] GmbH A. R. T.: http://www.ar-tracking.de. WWW, September 2007.

[HF08] HF F. S. B.: http://www.inition.co.uk/inition/product.php?URL_=product_hmd_fakespace_boomhf\&SubCatID_=34. WWW, December 2008.

[Int09] InterSense: http://www.isense.com/. WWW, March 2009.

[LaV03] LaViola J. J.: A testbed for studying and choosing predictive tracking algorithms in virtual environments. In *EGVE '03: Proceedings of the workshop on Virtual environments 2003* (New York, NY, USA, 2003), ACM, pp. 189–198.

[LOiT08] Li S., Okuda M., ichi Takahashi S.: Compression of human motion animation using the reduction of interjoint correlation. *EURASIP Journal on Image and Video Processing 2008* (2008).

[LR08] Limet S., Robert S.: Flowvr-vrpn: first experiments of a vrpn/flowvr coupling. In *VRST '08: Proceedings of the 2008 ACM symposium on Virtual reality software and technology* (New York, NY, USA, 2008), ACM, pp. 251–252.

[MAEH04] Mania K., Adelstein B. D., Ellis S. R., Hill M. I.: Perceptual sensitivity to head tracking latency in virtual environments with varying degrees of scene complexity. In *APGV '04: Proceedings of the 1st Symposium on Applied perception in graphics and visualization* (New York, NY, USA, 2004), ACM, pp. 39–47.

[MHK06] MOESLUND T. B., HILTON A., KRÜGER V.: A survey of advances in vision-based human motion capture and analysis. *Comput. Vis. Image Underst. 104*, 2 (2006), 90–126.

[MJvR03] MULDER J. D., JANSEN J., VAN RHIJN A.: An affordable optical head tracking system for desktop vr/ar systems. In *EGVE '03: Proceedings of the workshop on Virtual environments 2003* (New York, NY, USA, 2003), ACM, pp. 215–223.

[MMD*08] MYALL D. J., MACASKILL M. R., DAVIDSON P. R., ANDERSON T. J., JONES R. D.: Design of a modular and low-latency virtual-environment platform for applications in motor adaptation research, neurological disorders, and neurorehabilitation. *IEEE Transactions on neural systems and rehabilitation engineering 16*, 3 (June 2008), 298–309.

[MRWFPB03] MEEHAN M., RAZZAQUE S., WHITTON M. C., FREDERICK P. BROOKS J.: Effect of latency on presence in stressful virtual environments. In *VR '03: Proceedings of the IEEE Virtual Reality 2003* (Washington, DC, USA, 2003), IEEE Computer Society, p. 141.

[MT05] MAGEE P., TOOLEY M.: *The Physics, Clinical Measurement, and Equipment of Anaesthetic Practice*. Oxford University Press, USA, 2005. 37.

[MW93] MACKENZIE I. S., WARE C.: Lag as a determinant of human performance in interactive systems. In *CHI '93: Proceedings of the INTERACT '93 and CHI '93 conference on Human factors in computing systems* (New York, NY, USA, 1993), ACM, pp. 488–493.

[POL09] POLHEMUS: http://www.polhemus.com/. WWW, March 2009.

[PSLJ99] PASMAN W., SCHAAF A. V. D., LAGENDIJK R. L., JANSEN F. W.: Accurate overlaying for mobile augmented reality. *Computers and Graphics 23* (1999), 875–881.

[RM07] ROBINSON A., MANIA K.: Technological research challenges of flight simulation and flight instructor assessments of perceived fidelity. *Simul. Gaming 38*, 1 (2007), 112–135.

[RS01] REITMAYR G., SCHMALSTIEG D.: Opentracker-an open software architecture for reconfigurable tracking based on xml. In *VR '01: Proceedings of the Virtual Reality 2001 Conference (VR'01)* (Washington, DC, USA, 2001), IEEE Computer Society, p. 285.

[SMK98] STANNEY K. M., MOURANT R. R., KENNEDY R. S.: Human factors issues in virtual environments: A review of the literature. *Presence: Teleoper. Virtual Environ. 7*, 4 (1998), 327–351.

[SPB06] SPRAGUE D. W., PO B. A., BOOTH K. S.: The importance of accurate vr head registration on skilled motor performance. In *GI '06: Proceedings of Graphics Interface 2006* (Toronto, Ont., Canada, Canada, 2006), Canadian Information Processing Society, pp. 131–137.

[Ste04] STERGIOU N.: *Innovative Analyses of Human Movement*. Human Kinetics, 2004. 236.

[Sys07] SYSTEMS V. M.: http://www.vicon.com. WWW, September 2007.

[TAS07] TUMANOV A., ALLISON R., STUERZLINGER W.: Variability-aware latency amelioration in distributed environments. *Virtual Reality Conference, 2007. VR '07. IEEE* (March 2007), 123–130.

[THC*96] TAYLOR V. E., HUANG M., CANFIELD T., STEVENS R., REED D., LAMM S.: Performance modeling of interactive, immersive virtual environments for finite element simulations. *Int'l J. Supercomputer Applications and High Performance Computing 10* (1996), 141–151.

[WB94] WARE C., BALAKRISHNAN R.: Reaching for objects in vr displays: lag and frame rate. *ACM Trans. Comput.-Hum. Interact. 1*, 4 (1994), 331–356.

[Wel67] WELCH P.: The use of fast fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms. *Audio and Electroacoustics, IEEE Transactions on 15*, 2 (Jun 1967), 70–73.

[WF02] WELCH G., FOXLIN E.: Motion tracking: No silver bullet, but a respectable arsenal. *IEEE Computer Graphics and Applications 22*, 6 (2002), 24–38.

[Wlo95] WLOKA M. M.: Lag in multiprocessor virtual reality. *Presence 4* (1995), 50–63.

[ZH04] ZHOU H., HU H.: *A Survey – Human Movement Tracking and Stroke Rehabilitation*. Tech. rep., Department of Computer Sciences, Univeristy of Essex, December 2004.