Smooth Object State Updates in Distributed Haptic Virtual Environments

Clemens Schuwerk and Eckehard Steinbach
Institute for Media Technology
Technische Universität München, Germany
Email: {clemens.schuwerk, eckehard.steinbach}@tum.de

***Abstract*—In a multi-user distributed haptic virtual environment (DHVE), object state update messages are necessary to maintain a consistent virtual environment at every entity involved. Traffic control schemes applied to reduce the usually high number of update packets lead to inconsistencies among the participants. Whenever a new state update is received, the discrepancy between the old and new object state has to be resolved. For this purpose, various interpolation methods can be used to smoothly correct the state error. In this paper, we investigate and compare three such methods using subjective experiments to determine a haptically preferred method that leads to a convincing force feedback. Besides basic linear interpolation, we test two other methods from literature which are based on cubic polynomials. Our experimental evaluation, based on a pursuit-tracking task and pairwise comparisons, shows that linear interpolation performs better than the two cubic approaches and is subjectively preferred.**

I. Introduction

Enhancing human-machine or human-human interaction mediated by technical systems by incorporating the haptic modality is a growing research area. As humans rely heavily on their kinesthetic and tactile perception during interaction with their environment, the realism of interaction via technical systems is increased substantially if haptic feedback is provided to the user [1]. Additionally, the sense of immersion into distant real or virtual environments is improved [2]. Also, it has been shown that the integration of the haptic modality into technical systems leads to an increased task performance [3].

Advances in existing communication infrastructure like the Internet enable physical (haptic) interaction or manipulation of objects in distant real or virtual environments. The transmission of haptic information over today's communication networks, however, faces several challenges [4]. Firstly, the high temporal resolution of the human haptic perception system requires a stimulus refresh rate of $1kHz$, resulting in the need for high-rate information exchange on the communication channel. Additionally, the communication network closes a control loop between the local human and the remote entity being controlled, e.g. a remote robot or a virtual avatar in a remote virtual environment. The stability of this control loop necessities low-delay communication of haptic data samples. To address these issues in telepresence and teleaction (TPTA) systems, lossy perceptual haptic data reduction schemes have been introduced [5].

Similar challenges also apply for so-called Shared Haptic Virtual Environments (SHVEs), wherein several users, possibly spread around the globe, jointly manipulate objects within a shared virtual world. Different communication architectures based on either the client-server or the peer-to-peer paradigms can be conceived for building SHVEs. As more than one user is involved in the interaction, also scalability and consistency issues arise. The haptic rendering algorithms need to run at a minimum rate of $1kHz$ to display high-fidelity force feedback to the user. To allow true collaboration between users, the virtual scene perceived by all of them should be consistent. Therefore, low-delay and high-rate information exchange between collaborating users is necessary causing the communication network to quickly become a bottleneck.

Contrary to a common real world TPTA system, where contact forces with the remote environment are measured by a force sensor and sent back to the operator, in virtual applications, parts of the rendering pipeline can be distributed and calculated either on a centralized server or locally at the user. A commonly used architecture, proposed by Hikichi et al. in 2001 [6], is shown in Fig. 1a and is referred to as the *Consistency Client-Server Architecture* throughout this paper. Here, the centralized server is the only entity in the system running a physics engine which simulates the current object state (position, velocity, rotation and rotational velocity) at a rate of $1kHz$. This state is multicasted to the clients, where the local object state database is updated and interaction forces are rendered locally. By shifting the force rendering to the clients, the transmission of force samples is avoided and stable local force rendering is achieved. However, perceptual transparency is lost with increasing delay [7].

In the so-called *Fully-Distributed P2P Architecture* shown in Figure 1b, a local physics engine is running at each participating peer. Users only exchange their current device state which is then used as input to the local physics simulations. Obviously, the locally calculated object states will diverge in presence of communication delay, as the physics simulations are all based on delayed information. Thus, consistency control mechanisms are needed to correct this inconsistency.

Both architectures have drawbacks in terms of either delay, scalability or consistency. Additionally, the load on the communication network increases with the number of participating users. Traffic control mechanisms, shortly reviewed in Section II, are applied to reduce the communication load and to avoid congestion in the network. For the Consistency Client-Server Architecture, such traffic control schemes lead to fewer object state updates sent from the server to the clients. For the P2P architecture, inconsistency resolution mechanisms need to resolve discrepancies between the currently rendered object state and the corrected state. If the local object state database is immediately updated, sudden jumps in rendered force feedback

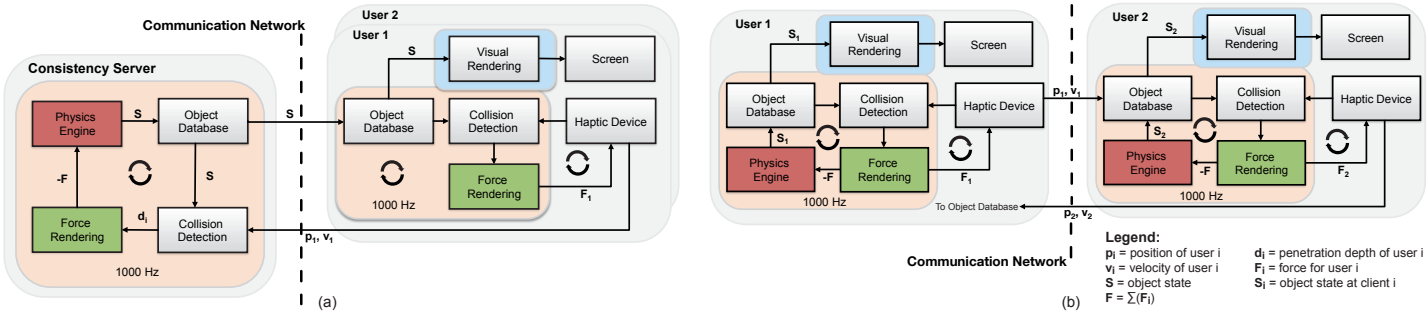

Fig. 1. Distributed architectures for SHVEs: (a) Consistency Client-Server Architecture with centralized physics engine and local force rendering (adopted for our experiments). (b) fully-distributed P2P architecture with physics engine running at every peer.

will occur. Such jumps can be avoided by using appropriate interpolation methods to smoothly converge from the old to the new state. Different methods, based on linear interpolation or higher-order polynomials [8], [9], [10], have been proposed for distributed applications without haptic feedback. For DHVEs, there is a lack of knowledge about which methods should be applied to achieve perceptually convincing force feedback. Thus, this paper investigates and compares different interpolations techniques, known from computer graphics and networked games in a distributed haptic virtual environment. The Consistency Client-Server Architecture depicted in Fig. 1a is deployed for this purpose. Subjective experiments are conducted to determine the method that leads to the perceptually most convincing force feedback. Thus, the comparison is based solely on user preference, not considering other factors as, e.g., computational complexity.

The remainder of this paper is structured as follows. Relevant traffic control schemes are reviewed in Section II, followed by an introduction of the three investigated state interpolation methods in Section III. The experimental design, the analyses as well as the gathered results are described in Sections IV-VI. Finally, in Section VII we summarize our results and give an outlook on future work.

II. Traffic Control in Shared Haptic VEs

The concept of *dead reckoning* (DR), supported by the *IEEE Standard for Distributed Simulations* [8], is a popular technique used in distributed simulations to reduce the number of transmitted state updates. It is based on local prediction of object states using the last received update. The entity issuing the state updates, e.g. the centralized server in the afore-mentioned client-server architecture, implements the same prediction method and triggers new updates only if the error between the predicted and the current real state exceeds certain thresholds. Here, the globally valid real object state is calculated by the physics engine on the centralized server. Differences in position and rotation of the objects are compared to the corresponding thresholds and updates are triggered accordingly. Both thresholds have to be tuned according to the simulation's characteristics to find a trade-off between the fidelity of the rendered objects and packet rate reduction.

In virtual environments enhanced with the haptic modality, the choice of these thresholds heavily influences the fidelity of the force feedback displayed to the human. If a user is in contact with an object, the displayed force is rendered locally, commonly based on Hooke's Law $F = k * d$, where k is the rendered stiffness and d the device's penetration depth into the object [11]. Any object state update leads to a sudden change in rendered force, as the penetration depth changes abruptly due to the new object position. A dead reckoning approach for DHVEs which takes the perceptibility of the resulting force changes into account and adjusts the error threshold accordingly is presented in [12]. According to Weber's law, the just noticeable difference (JND) in a comparison of two force magnitudes is proportional to the magnitude of the first stimulus [13]. By keeping the force changes caused by sudden object state updates below the human perception limits, a perceptually transparent force rendering at the clients can be achieved.

An interpolation scheme might be used at the clients to smoothly correct the state of an object, rather than updating it abruptly. Approaches based on first-order [8] or higher-order polynomials [9] are used in the literature for VEs without haptic feedback. There is, however, no well-accepted model for a perceptually convincing method, even for visual-only applications [10]. By smoothly correcting state inconsistencies in DHVEs, higher error thresholds can be chosen, leading to a higher packet rate reduction, while satisfactory force feedback quality is still maintained. In [6], the packet rate on the channel from the centralized consistency server to the clients is adjusted according to the current network load to prevent congestion. Missing updates are absorbed by predicting and interpolating the objects' positions and rotations. The used interpolation techniques, however, are not described.

III. Object State Interpolation

The three interpolation techniques used in this paper to smoothly correct the inconsistency after a state update is received at the clients are briefly introduced in the following. The given formulas correspond to the interpolation of an object's position. Similarly, they can be used with Euler angles to correct rotation errors [8], [9]. Let $\mathbf{S}_* = \{\mathbf{p}_*, \mathbf{v}_*\}$ denote an object state, where $\mathbf{p}_* \in \Re^3$ is the position and $\mathbf{v}_* \in \Re^3$ the velocity in global coordinates. The current object state at time $t = t_0$ when a new update is received at a client is denoted by $\mathbf{S}_0$, and the newly received object state by $\mathbf{S}_1$. The time used for the convergence from the old to the new position is denoted as t_n and is pre-decided before the interpolation starts. With this, $i(t) = (t - t_0)/t_n$ where $t_0 \leq t \leq t_0 + t_n$ can

be defined. Assuming a linear prediction at the client-side, the target position $\widehat{\mathbf{p}}_1$ can be calculated with $\mathbf{S}_1$ as $\widehat{\mathbf{p}}_1 = \mathbf{p}_1 + \mathbf{v}_1 t_n$.

A. Linear interpolation

The most straightforward interpolation between the current position $\mathbf{p}_0$ and the target position $\widehat{\mathbf{p}}_1$ is the linear interpolation [8], which is given for the object position $\mathbf{p}(t)$ at time t as follows:

$$\begin{aligned}\mathbf{p}(t) &= \mathbf{p}_0 + (\widehat{\mathbf{p}}_1 - \mathbf{p}_0) i(t) \\ &= \mathbf{p}_0(1 - i(t)) + \mathbf{p}_1 i(t) + \mathbf{v}_1 t_n i(t)\end{aligned} \tag{1}$$

Obviously linear interpolation leads to abrupt changes in the object velocity at the times $t = t_0$ and $t = t_0 + t_n$. Thus the object's trajectory in the virtual world is not C^1-continuous.

B. Hermite interpolation

Alternatively, the trajectory can be smoothed with a hermite curve segment, which takes also the old and new velocity, $\mathbf{v}_0$ respectively $\mathbf{v}_1$, into account. Thereby C^1 continuity is maintained. With the common hermite curve segment [14] and the predicted target position $\widehat{\mathbf{p}}_1$, the object position $\mathbf{p}(t)$ is given as:

$$\begin{aligned}\mathbf{p}(t) = \mathbf{p}_0(2i(t)^3 - 3i(t)^2 + 1) + \mathbf{p}_1(-2i(t)^3 + 3i(t)^2) \\ + \mathbf{v}_0 t_n(i(t)^3 - 2i(t)^2 + i(t)) \\ + \mathbf{v}_1 t_n(-i(t)^3 + 2i(t)^2)\end{aligned} \tag{2}$$

The drawback of hermite interpolation is the tendency to show an oscillatory behaviour [9]. Although the object trajectory is smooth, higher acceleration is needed during interpolation to correct the inconsistency within the given interpolation time t_n.

C. Projective velocity blending

The so-called projective velocity blending for dead reckoning in virtual environments proposed in [9] uses two combined linear interpolations. According to the authors, this method leads to a visually convincing interpolation in networked games that tends to show less oscillations than the hermite interpolation. The resulting curve segment, following the equations given in [9], can be denoted as follows:

$$\begin{aligned}\mathbf{p}(t) = \mathbf{p}_0(1 - i(t)) + \mathbf{p}_1 i(t) \\ + \mathbf{v}_0 t_n(i(t)^3 - 2i(t)^2 + i(t)) \\ + \mathbf{v}_1 t_n(-i(t)^3 + 2i(t)^2)\end{aligned} \tag{3}$$

The first part of Eq. 3 consists of a linear interpolation between the old and new object position, equal to the first two terms in Eq. 1. The prediction is incorporated into the latter two terms as in Eq. 2. Obviously, it is a mixture of the two afore-mentioned approaches and C^1-continuity is not achieved. However, the velocity used for extrapolation is blended from $\mathbf{v}_0$ to $\mathbf{v}_1$ compared to Eq. 1 where only $\mathbf{v}_1$ is used in the extrapolation term.

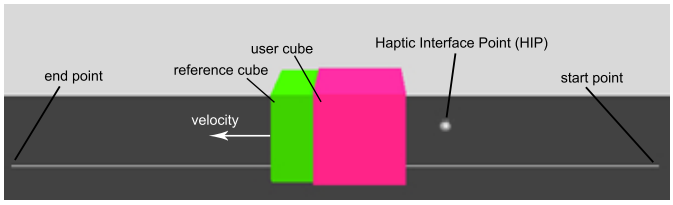

Fig. 2. Pursuit tracking task with user cube (pink, haptically active) and reference cube (green, only visual).

In the following we describe two experiments conducted to evaluate the impact of the interpolation scheme on the haptic feedback quality.

IV. Experimental Setup

Both subjective experiments are based on the same apparatus and task described in the following.

A. Apparatus

The PHANToM Omni haptic device (Sensable Technologies Corp, Woburn, MA) is used for the experiments, together with Chai3D [15], an open source library for haptic rendering. As the target of both experiments is to investigate the perceptual quality of the haptic feedback for the different interpolation techniques, only a single user is connected to the server and his haptic device position is transmitted to the server at the full rate of $1kHz$. On the return channel from the server to the client, the traffic control scheme proposed in [12] is used to reduce the number of object state updates. In both experiments, a new state update is only triggered if the expected force change due to a sudden object position change exceeds 25% of the current force displayed to the user. This parameter was determined in pilot tests, where we found it to be a reasonable trade-off between operability and perception of artefacts. According to the known human perceptual limitations, this force change is easily perceivable [13]. However, due to the applied object state interpolation, the abrupt jump in object position is smoothed out before being displayed to the user. For the task described below and the applied traffic control scheme, only 4.1 packets per second on average are needed from the server to the client. Additionally, the server and the client are running on the same machine to avoid any effect of delay in the communication between them, as the impact of delay is not the subject of this study. During the experiments, the subjects wore headphones with active noise cancellation to prevent any auditory feedback from the haptic device or distractions from the environment.

B. Task description

It is important to have a simple to accomplish task which avoids as much variability between runs and subjects as possible to compare artefacts introduced by different interpolation schemes. Therefore we choose a pursuit tracking task, wherein the subject has to push a cube from a start location to an end location following a reference cube as closely as possible (see Figure 2). The reference cube's movement is experimentally controlled to show constant acceleration in the first half of the trajectory and constant deceleration afterwards. Additionally, the movement of both the device and user cube is limited

TABLE I. USED RATING SCHEME

5	4	3	2	1
no difference	perceptible, but not disturbing	slightly disturbing	disturbing	strongly disturbing

to one degree-of-freedom. This simplifies task execution and subjects can pay more attention to the perception of artefacts. Without this constraint task and input behaviour the comparison between different parameter settings is difficult as subjective ratings might be influenced by individual interaction patterns of the subjects.

V. INTERPOLATION INTERVAL EXPERIMENT

The *interpolation interval experiment* evaluates the time parameter t_n that is used for the interpolation. The position inconsistency can be corrected more smoothly with larger t_n. However, the interpolation should be finished before the next state update arrives. The inter-arrival time between two consecutive updates is not constant as the server applies traffic control which non-uniformly triggers new update packets. In case of an update arriving during interpolation, we immediately stop the current interpolation and apply the newly received state. Otherwise the client falls behind the intended movement and the inconsistency between the server and the clients increases.

A. Procedure

We test a range of interpolation times from $10ms$ to $250ms$ in our experiment. Additionally, we test an adaptive method, wherein t_n is chosen based on the latest packet arrival times. For this, we use the mean of the last five packet inter-arrival times and, to be more conservative, choose 75% of this mean as the new t_n. Moreover, whenever an interpolation still needs to be aborted due to the arrival of a new update, our heuristic halves t_n.

The linear interpolation is fixed for all subjects as we only want to study the impact of the interpolation time in this first experiment. We use a within-subjects experimental design with all subjects performing the same task under the same parameter settings. Each setting is tested in a single run, which is presented randomly. The subjects are asked to rate their perception of force feedback on the rating scale shown in Table I. For this purpose we conduct a training phase to familiarize the subjects with the task, the device, the experimental procedure and two references corresponding to a rating of 5 and 1. One reference is a run without any traffic control scheme applied and, thus, 1000 packets per second transmitted to the client. The second reference has the same traffic control scheme enabled as all the other runs, but no interpolation is used to smoothly correct the object state. During the experiment, the subjects are allowed to perform both reference runs whenever they desire to calibrate their ratings.

B. Participants

Fourteen subjects participated in the experiment, eleven male and three female with an age between 24 and 30. Five subjects are daily users of haptic devices. Four subjects have participated in earlier subjective experiments in this research area and were familiar with the device. The remaining five subjects are novice users and although a training phase was conducted with all of the subjects prior to the experiment, they had a more extensive training.

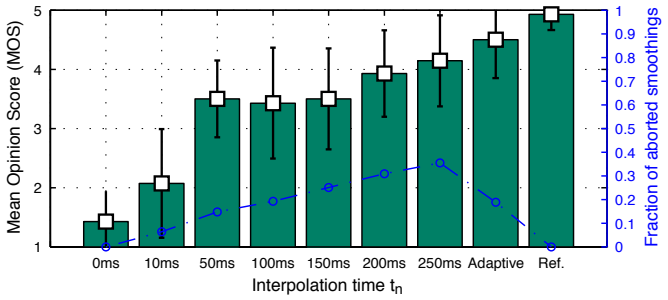

Fig. 3. User ratings (green), their standard deviation (black) and fraction of aborted interpolations (blue) for the *interpolation interval experiment*.

C. Results and analysis

The results of the *interpolation interval experiment* are shown in Fig. 3. As expected, the perceived force feedback quality improves with increasing interpolation time t_n. A big increase to a MOS of 3.5 can be observed for $t_n = 50ms$. From $t_n = 50ms$ to $t_n = 150ms$ the MOS remains nearly constant and increases again for higher t_n. The adaptive method performs best compared to all fixed interpolation times with a MOS of 4.5. In general, the standard deviation in the user ratings is high. This lies in the nature of such subjective experiments, but might also come from the difficulty to adjust the rating scale consistently between subjects in the training phase.

Statistical significance of the differences in user ratings for the different interpolation times is evaluated with a non-parametric Kruskal-Wallis test as normal distribution in the population could not be assumed due to the small sample size ($\chi^2(8) = 81.42, p < .001$)). Tukey's honestly significant difference (HSD) test indicated that all differences compared to the reference without interpolation ($t_n = 0ms$) are significant ($p < .05$), except for $t_n = 10ms$. For $t_n = 200ms$, $t_n = 250ms$ and the adaptive method there is no significant difference to the 1000 packets/s reference, however there is a difference compared to $t_n = 10ms$. No significant difference was found between $t_n = 50ms$ and the adaptive method.

The fraction of aborted interpolations increases linearly from around 10% at $t_n = 10ms$ to nearly 40% at $t_n = 250ms$. The adaptive method reduces this fraction as expected and leads to 20%, which is about the same fraction as for $t_n = 50ms$. The mean interpolation time used in the adaptive method is $\bar{t}_n = 148ms$. This clearly shows the usefulness of the proposed heuristic to choose the interpolation time, as a high t_n with a low interpolation abortion ratio is achieved.

From these results we conclude that for our specific setting (task and traffic control) we achieve a substantial improvement in force feedback quality if we use at least $50ms$ as interpolation time t_n. On average, the adaptive method is rated higher while keeping the fraction of aborted interpolations at a similar level. The adaptive method performs best compared

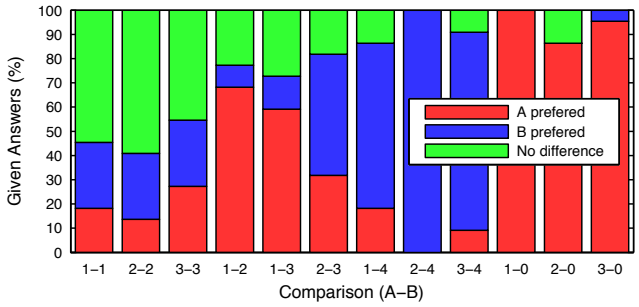

Fig. 4. User preference in the *preference experiment* (0: no interpolation, 1: linear interpolation, 2: cubic hermite, 3: velocity blending, 4: reference).

to the reference case and, thus, will be used in the second experiment described in the following.

VI. Preference experiment

The *preference experiment* uses several pairwise comparisons of the methods described in Section III to determine a haptically preferred approach, following the methodology described in [16].

A. Procedure

Two runs (denoted as run A and run B) as explained in Section IV-B are presented sequentially in each comparison. Afterwards, the subject is asked to select his preferred run. Additionally, the subject has the option to state that there is no perceived difference between both runs. The comparisons include all possible combinations between the three interpolation methods, as well as comparisons to the two references explained in Section V-A. Additionally, there are comparisons with both runs A and B set to be identical to determine an identity norm. Each comparison is evaluated twice, leading to an overall number of 24 comparisons per subject. The order of comparisons and runs within a comparison is selected randomly to avoid any systematic error in the results. Repetition of runs within a comparison or the repetition of the comparison itself on behalf of the subject is not allowed. Following the results of the *interpolation interval experiment*, the adaptive method is used to decide on the interpolation time t_n.

B. Participants

Twelve subjects participated in the second experiment. Two of them are novice users of haptic devices. Ten subjects already participated in the *interpolation interval experiment*, which was conducted several weeks before, and thus, were familiar with the device and the task. Nevertheless, a training session was conducted with every subject before the actual experiment started. One subject was excluded from the later analyses as the two repeated comparisons were judged differently by more than 50%, indicating random answers.

C. Results and analysis

The user preference for all comparisons is shown in Fig. 4. The first three bars show the comparisons for the case when the runs A and B use the same interpolation mode. The subjects correctly choose the *no difference* answer only in 50% to 60% of the cases, showing the subjects' difficulties of comparing two separate runs. Besides the uncertainty of every subjective experiment, we think the following characteristic could be a reason for this. Although we carefully built our experiment around a special and simplified task to reduce the variation between different experimental runs, a human is not able to exactly repeat a run and to perfectly follow the reference cube. Occasionally, a subject realizes during task execution that the error between the user cube and the reference cube is growing too big and quickly adapts his movement accordingly. Due to the nature of the applied traffic control scheme, new state updates are only triggered if the linear client-side prediction fails, meaning whenever sudden accelerations in movement occur. New updates lead to new interpolations on the client-side and as a result perceived artefacts in the force feedback displayed to the user. These first three comparisons are explicitly included in the experiment to counteract this phenomena. We use them as identity norms as suggested in [16] and investigate if other comparisons differ from them by applying Pearson's chi-squared test. With this we can conclude if any interpolation mode performs differently compared to the corresponding norm. We further investigate if there is a significant preference by applying the statistical method described in [16].

The comparisons 1-2, 1-3 and 2-3 shown in Fig. 4 are respective direct pairwise comparisons between linear interpolation (1), cubic hermite interpolation (2) and velocity blending (3). Linear interpolation seems to be clearly preferred compared to the other two interpolation modes. Compared to cubic hermite interpolation (1-2), it is preferred in 68% of the comparisons, while in 9% the cubic approach was preferred and in 23% of the runs the subjects perceived no difference (in short 68%:9%:23%). Pearson's chi-squared test indicates a significant difference compared to the identity norm 1-1 ($\chi^2(2, N = 22) = 37, p < .001$), indicating that method 1 and 2 do not perform equally. The follow up analysis shows a significant preference for linear interpolation ($\chi^2 = 10.89, p = .0008$ with $\alpha = .05$). Similar results are reported for the comparison to velocity blending (1-3) with significant difference to the identity norm (59%:14%:27%, $\chi^2(2, N = 22) = 24.75, p < .001$) and a preference for linear interpolation ($\chi^2 = 6.70, p = .009$). The comparison between cubic hermite interpolation and velocity blending (2-3) is significantly different from the identity norm 2-2 (32%:50%:18%, $\chi^2(2, N = 22) = 15.73, p = .0003$). However, the follow up analysis shows no clear preference for one of these two candidates ($\chi^2 = 0.90, p = .34$).

The comparisons of all methods to the two references are shown in the last six bars in Fig. 4. Clearly, the reference case with 1000 packet/s is preferred in nearly all of the runs (see 1-4, 2-4, 3-4 bars). Similarly, every interpolation scheme is preferred to the reference without interpolation (compare 1-0, 2-0, 3-0). These results are not surprising, as we choose a relatively large perceptual threshold for the traffic control in our experiment as explained Section IV-A. It is the subject of future work to adopt the preferred linear interpolation and to tune the deadband parameter in the traffic control scheme to find an optimal trade-off between packet rate reduction and satisfying force feedback displayed to the user.

VII. Conclusion

This work investigates object state interpolation in distributed haptic virtual environments. Two experiments, dealing with the interpolation time and the interpolation method are presented. The first experiment indicates that for the given task, traffic control scheme and communication architecture, at least $50ms$ are needed for a significant improvement in displayed haptic force feedback in comparison to sudden object state updates. However, this time cannot be seen as a general boundary, as it heavily depends on the task and system parameters, for example the error thresholds in the traffic control scheme. With more update packets sent to the client, the object state error will obviously decrease. But, there will be less time available for the interpolation as it should be finished before the next update arrives. The best result was achieved with the proposed heuristic to adaptively determine the interpolation time based on the latest packet inter-arrival times.

The three different interpolation methods presented in Section III are compared to each other subjectively in the second experiment. The subjects show a clear preference for the linear interpolation. We think the reason for this is the acceleration needed in both cubic methods to correct the inconsistency. During linear interpolation there is a sudden change in velocity only at the beginning and in the end of the interpolation. The other two schemes adapt more severely to the old (new) velocities at these time instances. Thus, they need to catch up during the interpolation itself. This acceleration was also reported after the experiment to be a driving factor in the subjective preference decisions. Although e.g. *projective velocity blending* might lead to a convincing visual object movement as stated in [9], in terms of displayed force feedback, the users prefer a trajectory with less acceleration. These results do not depend on the underlying communication architecture nor any consistency maintenance scheme, as in any case, the interpolation needs to be performed locally and the goal is always to smoothly converge from an old to a new object state. Thus, in summary, if inconsistency in a distributed haptic virtual environment has to be corrected, we conclude that a linear interpolation is haptically preferred. The time used for the interpolation should be chosen adaptively according to the latest update arrivals to ensure a timely end of the interpolation.

Although the preference was evaluated with a simplified one degree-of-freedom task, we hypothesise that these insights also hold for more realistic three dimensional tasks. The cubic interpolation methods tend to show an oscillatory behaviour, e.g., if the movement direction changes sharply. We assume that the linear interpolation is haptically preferred as it outperforms cubic methods even in the evaluated most basic task, where we actually expected the more smooth cubic methods to be superior. This hypothesis, however, needs to be validated with further experiments.

By using object state interpolation at the client we need to trigger fewer state updates in distributed haptic virtual environments, while still achieving convincing force feedback, by avoiding sudden jumps in the rendered force displayed to the human. We use the limitations of the human haptic perception to hide these force changes within the interpolation time without explicitly knowing these limits. Their determination is an interesting direction for future work. Similar to the perception of increased weight of virtual objects due to delay in the communication network [7], a large reduction of object state updates in our adopted client-server architecture leads to a loss of transparency. Thus, the influence of the traffic control scheme including state interpolation on perceived transparency should also be investigated in detail.

Acknowledgment

This work has been supported by the European Research Council under the European Unions Seventh Framework Programme (FP7/2007-2013) / ERC Grant agreement no. 258941.

References

[1] M. Srinivasan and C. Basdogan, "Haptics in virtual environments: Taxonomy, research status, and challenges," *Computers & Graphics*, vol. 21, no. 4, pp. 393 – 404, April 1997.

[2] C. Basdogan, C.-H. Ho, M. A. Srinivasan, and M. Slater, "An experimental study on the role of touch in shared virtual environments," *ACM Transactions on Computer-Human Interaction (TOCHI) - Special issue on human-computer interaction and collaborative virtual environments*, vol. 7, pp. 443–460, December 2000.

[3] J. T. Dennerlein, D. B. Martin, and C. Hasser, "Force-feedback improves performance for steering and combined steering-targeting tasks," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, April 2000, pp. 423–429.

[4] E. Steinbach, S. Hirche, M. Ernst, F. Brandi, R. Chaudhari, J. Kammerl, and I. Vittorias, "Haptic communications," *Proceedings of the IEEE*, vol. 100, no. 4, pp. 937–956, April 2012.

[5] P. Hinterseer, S. Hirche, S. Chaudhuri, E. Steinbach, and M. Buss, "Perception-based data reduction and transmission of haptic data in telepresence and teleaction systems," *IEEE Trans. on Signal Processing*, vol. 56, no. 2, pp. 588–597, February 2008.

[6] K. Hikichi, H. Morino, I. Fukuda, S. Matsumoto, Y. Yasuda, I. Arimoto, M. Iijima, and K. Sezaki, "Architecture of haptics communication system for adaptation to network environments," in *IEEE International Conference on Multimedia and Expo, ICME*, August 2001, pp. 563–566.

[7] S. Lee and J. Kim, "Transparency analysis and delay compensation scheme for haptic-based networked virtual environments," *Computer Communications*, vol. 32, no. 5, pp. 992–999, March 2009.

[8] IEEE Computer Society, "IEEE Standard for Distributed Interactive Simulation–Application Protocols," *IEEE Std 1278.1-2012 (Revision of IEEE Std 1278.1-1995)*, pp. 1 – 747, December 2012.

[9] C. Murphy, "Believable Dead Reckoning for Networked Games," in *Game Engine Gems 2*, E. Lengyel, Ed. A K Peters/CRC Press, 2011, pp. 307–328.

[10] X. Zhang, T. E. Ward, and S. Mcloone, "An information-based dynamic extrapolation model for networked virtual environments," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 8, no. 3, pp. 1–19, July 2012.

[11] C. Basdogan and M. A. Srinivasan, *Haptic Rendering in Virtual Environments, Handbook of Virtual Environments: Design, implementation, and applications*. Lawrence Erlbaum Associates Publishers, 2002.

[12] C. Schuwerk, R. Chaudhari, and E. Steinbach, "Perceptually robust traffic control in distributed haptic virtual environments," in *Haptics: Perception, Devices, Mobility, and Communication*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, June 2012, vol. 7282, pp. 469–480.

[13] E. Weber, *Die Lehre vom Tastsinn und Gemeingefuehl, auf Versuche gegruendet*. Friedrich Vieweg und Sohn Verlag, 1978.

[14] D. Salomon, *Curves and surfaces for computer graphics*. Springer New York, 2006.

[15] F. Conti, F. Barbagli, D. Morris, and C. Sewell. (2005) Chai 3d: An open-source library for the rapid development of haptic scenes. [Online]. Available: http://www.chai3d.org

[16] D. M. Ennis and J. M. Ennis, "Accounting for no difference/preference responses or ties in choice experiments," *Food Quality and Preference*, vol. 23, no. 1, pp. 13–17, January 2012.