High Precision Augmented Reality Haptics

G. Bianchi[*] B. Knoerlein[†] G. Székely[‡] M. Harders[§]

Computer Vision Laboratory
ETH Zürich Switzerland

Abstract

In our current research we examine the integration of haptic interfaces into augmented reality setups. The ultimate target of these endeavours is the application of the framework to training of manipulative skills in surgical environments. To this end, highly accurate calibration, system stability, and low latency are indispensable prerequisites. Therefore, we developed a new calibration method to exactly align the haptic and world coordinate systems. Moreover, a distributed framework was created, which ensures low latency and component synchronization. Finally, to demonstrate our results, we integrated all elements into an augmented reality haptics ping-pong game.

Keywords: haptic, calibration, synchronization, augmented reality

1 Introduction

The underlying idea of augmented reality (AR) systems is the combination of real and virtual objects into one environment. This is in contrast to virtual reality (VR) frameworks, where the whole surrounding of a user is computer-generated. In contrast to the latter, AR systems are of higher complexity, since the real world represents a reference frame into which the virtual elements have to be perfectly integrated. Any errors in this process would greatly diminish the usability of a system, and compromise user interaction and immersion [20].

Recent research has examined the possibility of adding haptic feedback to AR frameworks. A few proof-of-concept systems have been developed in this context, for instance [16, 23, 1]. However, lag reduction, exact alignment, or error minimization is only seldomly addressed. This is a significant shortcoming, since depending on the actual application area of the AR haptic system, these factors can play a fundamental role.

The driving force of our current research is the development of medical training systems using AR techniques. In this setting, high accuracy and stability are a prerequisite. Misalignment of overlaid virtual objects would greatly compromise manipulative fidelity and the sense of presence, and thus reduce the overall training effect. This situation becomes even more critical, if haptic feedback is integrated into the system. Calibration errors or excessive latencies in generating feedback would lead to the loss of spatial and/or temporal synchronization making the interaction disturbingly unnatural. Unfortunately, so far no systems are available, which would meet the high demands for stability, accuracy and response time. Therefore, we have developed a high precision augmented reality haptics environment. The two main components are the accurate calibration of the haptic device in world coordinates, as well as a distributed, low latency framework for synchronization. Moreover, we have integrated all components into a first simple application to demonstrate the high fidelity of our system. In the following, we present a detailed description of the haptic device integration into our AR demo application.

Following the overview of previous work and the description of our system setup, in section 4, we present an efficient calibration and registration procedure for incorporating the device into our system. The registration procedure covers the whole haptic workspace and only needs to be performed once. No recalibration is necessary after a restart of the system. In addition, we provide a detailed analysis of the calibration accuracy. Thereafter, in section 5 a description of the synchronization approach for our distributed system is given. In section 6 we illustrate the accuracy of the registration and show the synchronization results of the system by developing an AR-based ping-pong game. This scenario has been selected due to the high requirements, such as fast interaction with the user, accurate alignment between haptic feedback and virtual ball motion, and interaction of the virtual ball with the real environment. Finally, we conclude with a discussion of our results and an outlook to future work.

2 Previous Work

Only a limited number of groups have attempted to incorporate haptic rendering into AR applications, and even less have provided an analysis of the haptic registration accuracy, or dealt with latency issues.

Research pioneering the integration of haptic devices into AR systems was conducted by Vallino and Brown [21]. They determined the relationship between the haptic and the world coordinate system by measuring four points in both frames and estimating an affine transformation between them. However, affine representation does not provide any metric information, which prevents an analysis of the accuracy of their calibration procedure.

Another way to estimate the haptic-world transformation is to measure points with an external metrology system. In [5] a precision surveying theodolite is used to measure the position of markers placed in the haptic space. Unfortunately, the haptic calibration only provided a low accuracy with errors up to $1.5cm$. This is due to the difficulty of uniformly distributing markers over the whole workspace.

Some attempts have been made to improve the position accuracy of the PHANToM haptic device by correcting the position errors from measurements of grid points. A planar calibration grid is used in [17] to correct the haptic position, and then extend this correction to the entire haptic volume by extrapolation. However, the results reveal that the position error increases as the tip moves away from the grid.

An improvement of the grid approach has been suggested in [10] by placing two perpendicular planar grids in the haptic workspace. In addition, the authors adjust the joint angles of the PHANToM with correction values for rectifying the imprecise initialization of

[*] e-mail: bianchi@vision.ee.ethz.ch

[†] e-mail: knoerleb@vision.ee.ethz.ch

[‡] e-mail: szekely@vision.ee.ethz.ch

[§] e-mail: mharders@vision.ee.ethz.ch

the optical encoders. Those values are computed by minimizing the difference between the position of the grid points and the corresponding haptic measurements. The minimization results show a significant improvement of the haptic-grid calibration.

Although the estimated accuracy provided by the former two methods is on the order of a few millimeters, the results are only valid for points close to the calibration grid with the assumption that the grid is perfectly planar. Unfortunately, grid-based approaches only partially cover the haptic workspace, and can thus only locally provide sufficient accuracy.

Concerning the used display technology, integration of haptic rendering into AR systems, has mostly focused on desktop systems, e.g. [13, 19]. These setups create a virtual environment, which is projected on a semi-transparent mirror placed between the user and the haptic interface. Thus, virtual objects are apparently colocated with the haptic workspace. Often, head tracking is included to integrate simulation of motion parallax. These system exhibit similar calibration issues as in real AR systems.

Finally, games implemented with augmented reality techniques have also been a topic of reseach. In [22] an virtual table-tennis setup with real rackets has been developed. However, they did not include haptic feedback, and only used the rackets as an interface to position as virtual racket on the screen. In [18], a collaborative, mobile AR-based billard game has been discussed, which includes tactile feedback.

3 Augmented Reality Setup

The basic idea of our AR setup is to capture a real scene with a head-mounted camera, superimpose virtual objects in the image, and display the scene with a head mounted display. To ensure exact alignment between real and virtual world, the system needs to estimate the relative position between the virtual objects and the user's head. Therefore, accurate estimation of the head pose with respect to an arbitrary world coordinate system in which the virtual objects are placed is necessary.

Our AR system comprises an optical position tracking device OPTOTRAK 3020 manufactured by Northern Digital Inc. [8], a head-mounted FireWire camera and a camera-mounted marker. The optical tracker consists of three fixed linear cameras which detect the infrared LEDs attached to a marker. By triangulation, the optical system measures the 3D LED position with a RMS accuracy of $0.2mm$ at an optimal distance of $2.25m$. From these measurements, the orientation and position of the marker is computed. Since the camera and the marker are rigidly attached, the camera-marker transformation is fixed and estimated by an offline calibration process. Given this transformation and the marker pose, the AR system can estimate the camera pose with respect to the tracker coordinate frame. The estimated camera pose allows us to align the virtual and the real worlds.

Into our described AR setup, we integrated a SensAble PHANToM 1.5 haptic device. An overview of the resulting system is presented in Figure 1. A view of the scene as displayed to the user is rendered on an additional monitor.

To enable faithful haptic interaction with the virtually augmented objects, we need to accurately estimate the transformation between the haptic coordinate system and the optical tracking system. This will be explained in more detail below.

4 Calibration of Haptic System

4.1 Overview

In order to draw the virtual representation of the haptic interaction point at the correct physical location in the real world, we need to estimate the relationship between the haptic and the world coordinate system. In our case, the world coordinate system is provided by the OPTOTRAK. This procedure is called *haptic-world calibration*. We have to ensure, that errors in determining this transformation are minimized.

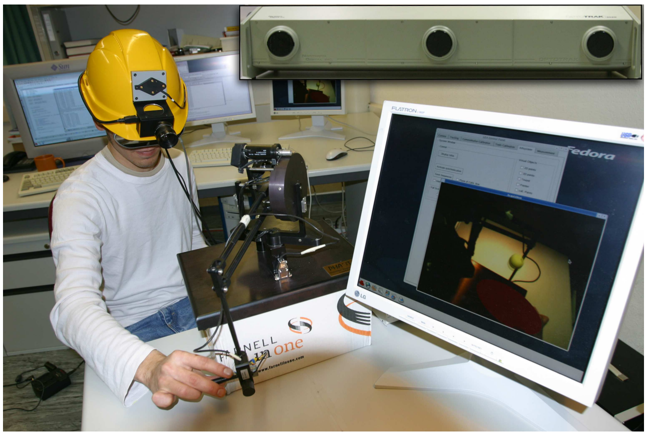

Figure 1: View depicting all components of AR system: head-mounted camera, marker, haptic device and optical tracking system (in the small window)

The underlying idea of the calibration procedure is to collect 3D point measurements in both coordinate systems. Based on these data, the rigid transformation between the two different Cartesian coordinate systems A and B can be determined. In photogrammetry, this is referred to as the *absolute orientation* problem. Let X_a and X_b be points expressed in A and B, respectively. The transformation between X_a and X_b is given by

$$X_b = RX_a + T \tag{1}$$

where R is the 3x3 rotation matrix and T the translation vector. R and T denote the rigid transformation between A and B. In our case, the set of points $(X_b)_{1\leq i\leq N}$ represents the measurements obtained from the tracking system, while the set of points $(X_a)_{1\leq i\leq N}$ describe positions determined from the haptic device. The *absolute orientation* method estimates the best rotation matrix and translation vector in the sense of distance error between both point sets. This leads to the least square problem

$$\min_{R,T} \sum_{i=1}^{N} \|X_b^i - RX_a^i - T\|^2 \tag{2}$$

Several algorithms have been suggested to solve this problem [12]; we follow a least-squares fitting method as discussed in [2]. Thus, the overall procedure consists of two main steps. First, we collect points measured in both coordinate systems by using a calibrated marker-tip extension attached to the haptic stylus. Thereafter, the haptic-world transformation is computed with an optimization approach.

With regard to both these steps, it should be noted, that the precision of encoder readings for the haptic device position depends on the reset procedure of the PHANToM, and therefore influences the calibration results. To avoid this problem, we reset only once the encoder values before the initial marker-tip calibration, and do not perform a re-initialization thereafter.

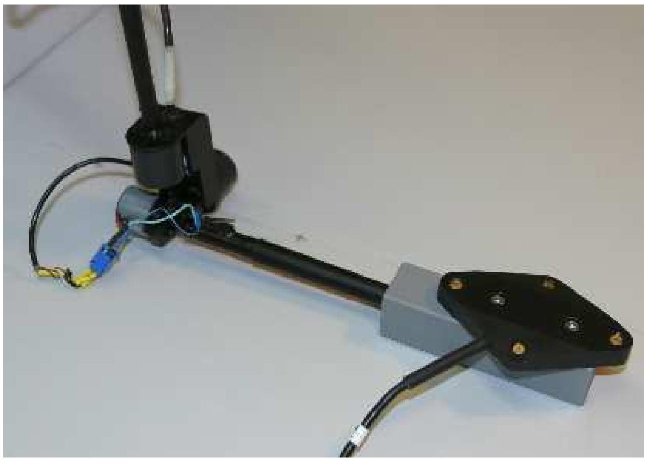

Figure 2: Calibration marker attached to tip of PHANToM

4.2 Tip-marker calibration

4.2.1 Calibration procedure

As discussed above, we first have to collect a set of points measured in both reference frames. To this end we rigidly attached a marker to the tip of the PHANToM (Figure 2). In this configuration, the tracking system only reports the marker pose H_{world}^{marker}. In order to measure the tip position with respect to the world coordinate system, we need to determine the translation T_{marker}^{tip} between the marker and the tip of the haptic device. This procedure is referred to as the *tip-marker calibration*. Thereafter, the tip pose H_{tip}^{world} with respect to the world coordinate system is computed from the composition of H_{world}^{marker} and T_{marker}^{tip}. Figure 3 illustrates the different transformations involved in the calibration process in detail, where the notation H_{o}^{p} is the rigid transformation from the frame O to the frame P.

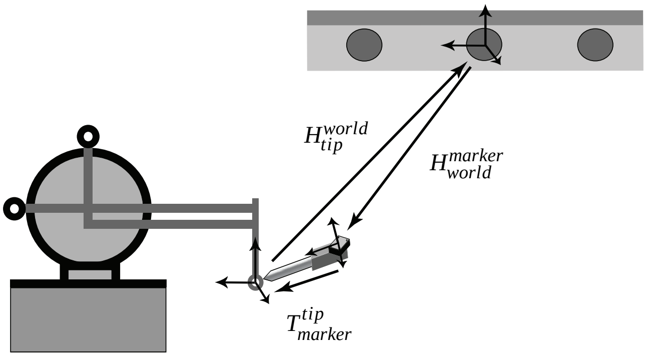

Figure 3: PHANToM and OPTOTRAK coordinate frames

The tip-marker calibration procedure consists of rotating the tip-mounted marker around the joint intersection of the gimbal. To do this, we stabilize the tip position in the middle of the haptic workspace by rendering a fixation force with the device. The tip-marker is then manually displaced following a spherical movement in space, while the marker poses are recorded. Given the recorded set of points, we used the pivot calibration method described in [11] to obtain the best sphere fitting the data.

In our case, the center of the sphere corresponds to the tip position with respect to the world frame, and the radius of the sphere represents the distance between the marker and the tip. From these data, the translation T_{marker}^{tip} defined in figure 3 can be obtained. Unfortunately, the estimation of the tip-marker relationship depends on the accuracy of the point measurements.

4.2.2 Simulation of calibration error

Two main sources of measurement errors can be identified - marker pose accuracy, and the stability of the fixed tip in the haptic workspace while rotating the marker. In order to determine the influence of the point measurement errors on the calibration, we performed a simulation of the procedure. In previous work [3], we have described a statistical noise model for the former error source. It models the marker pose error, which estimates the position error introduced by the tracking device. The model reports the marker pose with a systematic space-dependent offset. For the latter source of errors, we used a Gaussian noise to shift the tip position from the center of the sphere in order to model the instability of the haptic tip location while rotating the marker.

The simulations consisted of varying the magnitude of the introduced errors within a given range. For each noise value, we ran the simulation 1000 times and computed the statistical data. In order to be close to the real case, we modeled the marker-tip assembly from real measurements. In addition, we placed the haptic device at the optimal range of the tracking system.

Figure 4 shows the influence of the marker noise on the estimated sphere center. For the RMS accuracy of the tracking system, the center position error is lower than $0.05mm$. This low value is due to the advantage of recording the marker pose on a complete sphere. Other simulations in which the marker movement was restricted to a quarter of a sphere provided an error of the center position up to $0.2mm$. Thus, the actual point collection procedure also has an influence on the accuracy of tip-marker calibration.

The influence of the tip noise is depicted in figure 5. The simulation results reveal that the instability of haptic tip position dominates the calibration results. During experiments, we observed that the stylus moves about $1-2mm$. Adding to the resulting error the one of the marker pose, we obtain an accuracy of the calibration with a global error smaller than $1mm$.

It should be noted, that attaching the marker to the tip enables us to record points in the entire working volume of the haptic device, whereas grid-based methods are limited to a planar surface. Moreover, a grid might not even be perfectly flat, which would also introduce additional errors in the calibration.

4.3 Haptic-world calibration

4.3.1 Outline

With the marker-tip calibration described in the previous section, we are now able to obtain corresponding point measurements in both the haptic and world coordinate system. This allows us to carry out the actual haptic-world calibration. As discussed above, this denotes the determination of the rigid transformation between both reference frames. Unfortunately, additional errors are introduced due to the inaccuracies in haptic encoder initialization. Therefore, we have to carry out a two-staged optimization process. First we determine the rigid transformation, and thereafter we perform an encoder joint angle correction.

The initial step is the acquisition of points in both coordinate systems. We obtain point measurements by manually moving the marker-tip assembly in the entire haptic workspace. We record the tip position with respect to both haptic and tracker coordinate systems. In an ideal noise-free situation, two points would be identical. However, in reality, points are not aligned due to errors introduced by the haptic device encoder readings. Therefore, the goal of the calibration procedure is to correct the misalignment between both data sets.

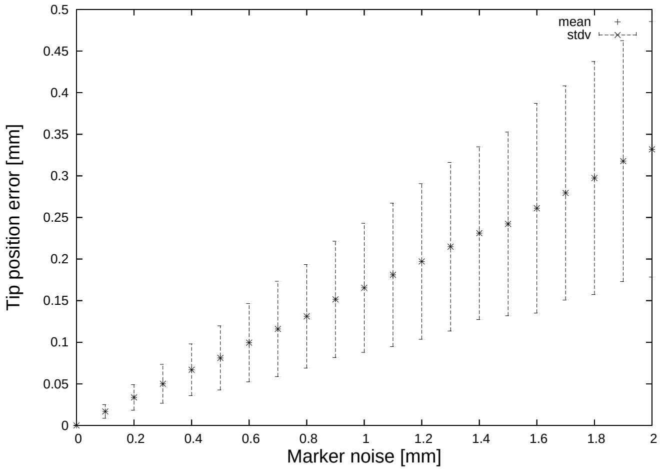

Figure 4: Influence of the marker noise on Marker-Tip calibration

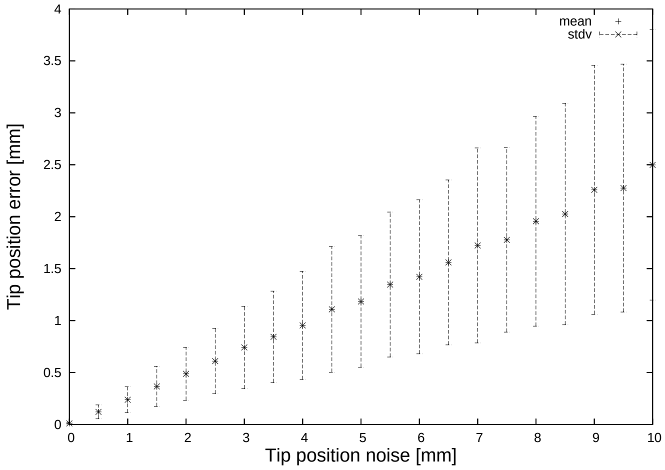

Figure 5: Influence of the tip noise on Marker-Tip calibration

$$h(\theta,l_1,l_2)=\begin{bmatrix} cos(\theta_1) & -sin(\theta_1)sin(\theta_3) & sin(\theta_1)cos(\theta_3) & sin(\theta_1)(l_1cos(\theta_2)+l_2sin(\theta_3)) \\ 0 & cos(\theta_3) & sin(\theta_3) & l_2-l_2cos(\theta_3)+l_1sin(\theta_2) \\ -sin(\theta_1) & -cos(\theta_1)sin(\theta_3) & cos(\theta_1)cos(\theta_3) & -l_1+cos(\theta_1)(l_1cos(\theta_2)+l_2sin(\theta_2)) \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

4.3.2 Calibration procedure

Apart from the *absolute orientation* problem, we also have to correct the joint angles of the haptic device in order to align the two datasets. To this end we will follow an *open-loop method*, as defined in [7]. By moving the joints and recording both tip pose and corresponding joint angle values, the parameters of a forward kinematics model are determined by carrying out a non-linear optimization. Inspired by [10], the correction technique is based on the registration between the two set of points expressed in the same coordinate system by adjusting the joint angles.

We choose the forward kinematics model of the PHANToM described in [4], which is given by equation 3. Here, θ is the vector of joint angles $(\theta_1,\theta_2,\theta_3)^T$ and (l_1,l_2) the length of the device arms.

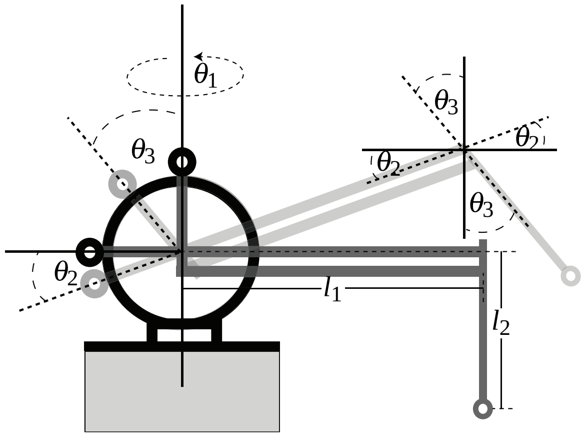

Figure 6: Joint angle configuration

Figure 6 illustrates the notation of the joint angles and the length of the arms. In our case, we only investigate the tip location because the structure of the tip-mounted marker is not able to measure the orientation of the tip. As a consequence, the kinematic parameters are solely estimated from the tip position. From equation 3, we extract the tip location $g(\theta,l_1,l_2)$ given by:

$$x_{haptic}=g(\theta,l_1,l_2)=\begin{bmatrix} sin(\theta_1)(l_1cos(\theta_2)+l_2sin(\theta_3)) \\ l_2-l_2cos(\theta_3)+l_1sin(\theta_2) \\ -l_1+cos(\theta_1)(l_1cos(\theta_2)+l_2sin(\theta_2)) \end{bmatrix} \quad (4)$$

Due to the discussed initialization inaccuracies, a deviation will be present between θ and the actual encoder readings. To address this, we model joint angle correction as a simple linear function:

$$\theta^i = k\psi^i+\gamma \quad (5)$$

where i is the pose number for $1\leq i\leq N$, ψ^i is the vector of the joint angles encoder readings, γ is the vector of the joint angle offsets, and k is the vector of joint angle gains. With this we substitute (5) into (4) and obtain:

$$x^i_{haptic}(\psi^i,\phi)=g\left(\theta(k,\psi^i,\gamma),l_1,l_2\right) \quad (6)$$

with $\phi=(k,\gamma,l_1,l_2)$.

Let $(x^i_{world})_{1\leq i\leq N}$ be the corresponding point set with respect to the world coordinate system. To align the point set, we follow a two-staged optimization. In the first step we perform the minimization based on equation 2 to solve the absolute orientation problem. This estimation is a closed-form solution, and does not require an iterative optimization. Thus, this step allows us to determine values for R and T with an initial guess for ϕ.

Thereafter, the joint angle correction is performed by adjusting the parameters of ϕ, such as to minimize the residual function $r(\phi)$.

$$r(\phi)=\min_{R,T}\sum_{i=1}^{N}\|x^i_{world}-Rx^i_{haptic}(\psi^i,\phi)-T\|^2 \quad (7)$$

Thus, the optimal correction parameter set ϕ^* is obtained by solving:

$$\phi^* = arg\min r(\phi) \quad (8)$$

We use the MATLAB implementation of the Levenberg-Marquardt algorithm for minimizing equation 8.

The two-staged process is iterated alternatingly until the residual drops below a given threshold. The next section illustrates the results of this optimization process.

4.4 Calibration analysis

4.4.1 Test protocol

The validity of the optimization process was assessed in an experiment. The goal of this step was on the one hand to examine the lower bound of the calibration error, and on the other hand, to determine the number of point measurements necessary to reach this minimal error.

To this end, we collected measurements for 140 manually selected points in both the haptic and world coordinate system. While acquiring the data, we tried to cover the entire haptic workspace by moving the device arm to the borders of the volume in order to reach the extreme values of the joint angles. These data were then randomly divided into two sets of 70 points. The first represented a *training set*, which we used for estimating the correction parameters of ϕ. The second was our *testing set*, which we utilized to determine the quality of the calibration. In order to avoid any bias introduced by the selection of these two sets, we repeated this random subdivision 1000 times, and determined the errors for each one of them. The statistical analysis takes all this data into account.

Since the optimization algorithm is based on an iterative approach, we need to provide an initial guess for the parameters to be optimized. In our case, the joint angle gains k were set to 1, the joint angle offsets γ to 0, and the length of the device arms l_1, l_2 to the manufacturer defined values of $209.55mm$.

Moreover, it should be mentioned, that the results were obtained for a device initialization position close to the usual reset configuration. However, as we will discuss below, the algorithm is robust against selection of the starting position, and thus the results representative for the whole workspace.

4.4.2 Analysis of correction parameters

Figure 7 shows the distance error between the haptic and the tracking measurements against the number of points used for optimizing the correction parameters. As revealed by the plot, the optimization process is able to reduce the error from 3 to 1.3mm. The remaining error is probably introduced while calibrating the marker-tip assembly. In addition, a minimum of 30 points can be used for precisely estimating the correction parameters.

Figure 8 depicts the offset values against the number of points used. It can be seen, that the optimization is successful for γ_2 and γ_3, whereas no solution can be found for offset γ_1. This indicates, that the joint angles θ_2 and θ_3 can be corrected. In contrast to this, the optimization algorithm does not find any solution for the joint angle θ_1. This is due to the fact, that correcting the angle θ_1 implies moving the haptic coordinate system from the reset position to the true position. However, the latter is unknown and cannot be estimated by the marker-tip assembly since its position is rigidly depending on the reset position. Thus, it is only possible to optimize the two joint angle offsets γ_2 and γ_3.

These findings are in line with the results obtained for the joint angle gains k_i as depicted in figure 9. In contrast to θ_1, the gains for the joints θ_2 and θ_3 are slightly deviating from 1. While the change of k_1 is negligible, the deviation of the others can not be omitted. Even though those changes are small, an optimization without joint angle gains has revealed a higher residual value of the equation 7.

Furthermore, to test whether the optimization process is robust against the reset position or not, we carried out additional independent optimizations at different starting positions using varying joint angles of the device arms. In order to examine the effect of different angles for θ_2 and θ_3 during calibration, we carried out the process using combinations of right or obtuse angles, respectively. For each angle combination, we again recorded 140 measurements and repeated the procedure described above. As revealed by figure 10, the distance error between the haptic and world points converges for any combination towards low error values. As mentioned earlier these values can probably be attributed to the calibration error of the marker-tip assembly caused by the noisy tracking data and the instability of the haptic fixation. Additional experiments, which did not focus on specific angles, but used reset locations in the extreme top and bottom, as well as left and right regions in the workspace resulted in the same outcome.

Finally, as a further result, we also noticed, that the optimized length values of the arms l_1, l_2 are not modified by the optimization process. This implies that the nominal values defined by the manufacturer are sufficiently precise.

4.4.3 Calibration results

In figures 11 and 12, the two sets of 3D points are plotted in the haptic coordinate frame before and after the optimization process. The circles represent the tracking measurements, and the stars are the haptic data. The picture on the left reveals the structure of the measurement errors reported by the haptic device. One can notice that the deformation between both clouds of points cannot be represented by any rigid transformation. A radial distance error with respect to the base of the haptic arm is probably the source of this. The picture on the right illustrates the results after estimating the correction parameters. The two clouds are nearly aligned to each other. This result implies that the optimized joint angles allow estimating an accurate transformation between the haptic and the tracking system.

5 DISTRIBUTED FRAMEWORK

5.1 System architecture

One of the main challenges in an AR system is to maintain low latency in order to synchronize the augmented images with the user motion also in time. Already a standard AR setup has to meet high computational demands, since image acquisition, image processing, virtual overlay, and drawing of the output image should be done with delays as small as possible. Adding a haptic interface to such a setup and performing physics simulations to allow user interaction with virtual objects would exhaust the computational power of currently available computing hardware. Therefore, we have developed a distributed system to meet the related requirements. As discussed in [14], several possibilities for distribution of such an application exist. In our case, we use a *graphics server* and a *physics server*.

The former carries out all tasks typical to a standard AR setup, e.g. image acquisition, camera pose computation, or rendering. Thus, in our context the only external input data needed are position and orientation of the haptic interface, as well as of the virtual objects in the scene. This is true, as long as we are using rigid objects. Switching for instance to deformable objects would require an update of a complete mesh. Nevertheless, as discussed below, sufficient time would be available to also transmit larger packets of data. The latter takes care of force-feedback computation, and physical simulation of virtual objects. Thus, the server is completely independent from tracking or image acquisition. Communication between the two is accomplished via an ethernet connection with the

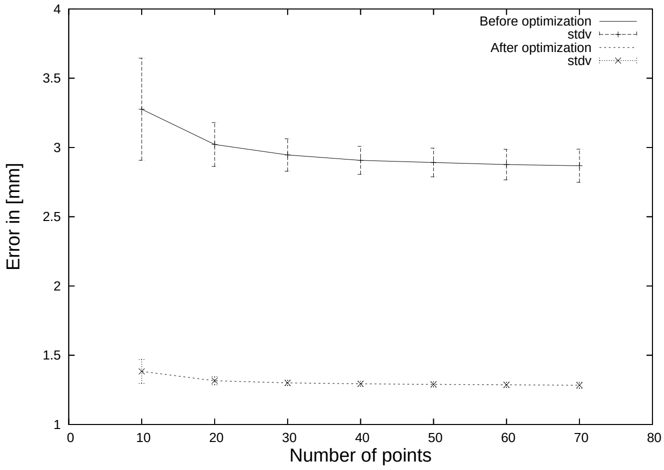

Figure 7: Tip position error using the test set

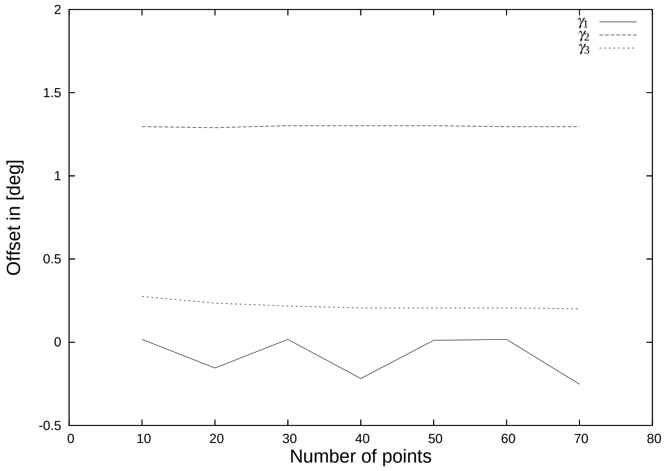

Figure 8: Angle correction after optimization

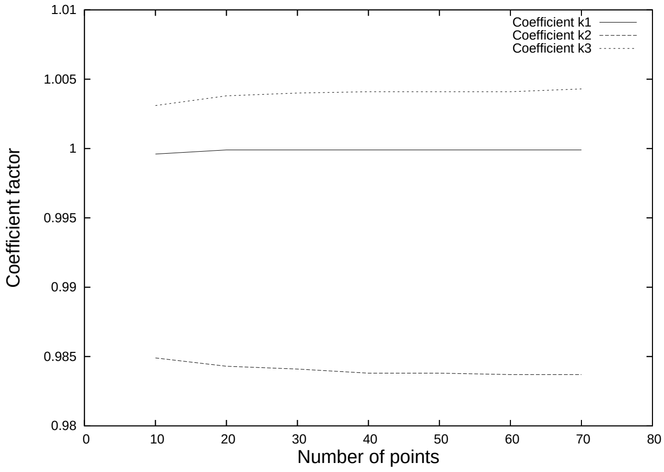

Figure 9: Joint angle gain after optimization

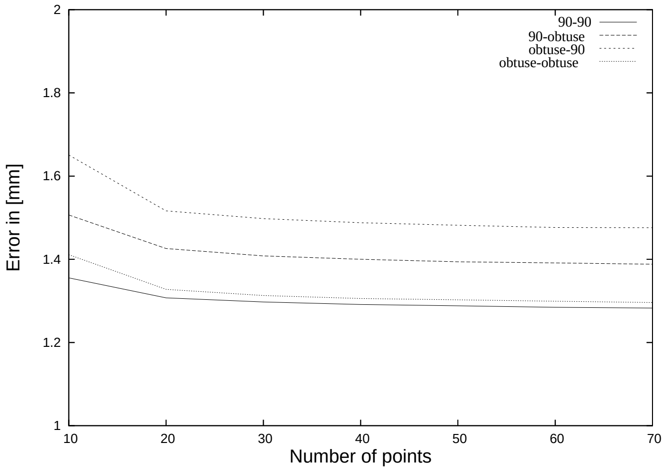

Figure 10: Calibration results for four combinations of joint angles during reset procedure

TCP protocol. The Nagle algorithm has been disabled to reduce the round trip time.

The described separation provides sufficient resources to meet the low-latency requirements, however, two additional components are needed to ensure coordination during run-time. First, we have to guarantee synchrony between events in the haptic and visual virtual world. This problem can be tackled by synchronizing both machines in time, which allows the correlation of recorded images to specific timesteps in the physical simulation. Secondly, the system requires an appropriate communication model, which does not block or interfere otherwise with the physics or graphics loop.

In our testbed implementation described below, we have used two dual PCs with $2.8GHz$ and $2.4GHz$ CPUs, respectively. Both machines have $2GB$ RAM with $512kb$ cache, and are running with a Linux OS. On the haptics server, the haptic loop is updated with a fixed refresh rate of $1kHz$. The physics computation runs in a separate loop and can achieve higher refresh rates. However, we also limited it to $1kHz$. On the graphics server the AR pipeline is carried out with an update of about $30Hz$. This is mainly limited by the frame acquisition time of the camera. The visualization is rendered with an NVIDIA Geforce 6600 GT card.

It should be mentioned, that the distributed system also exhibits some additional advantages. It provides high flexibility and can easily be adapted to other applications. Moreover, the individual components can be developed separately from other framework components. Finally, due to the decoupling, different operating systems could be used on the servers, which could be useful, if a haptic device other than the PHANToM should be integrated.

5.2 Synchronization

Synchronizing both servers in time is essential for our AR setup. Time-lagged movement of virtual objects with respect to the recorded background image would result, if no or inaccurate synchronization would be used. Consequently, user interaction would be greatly compromised. To tackle this problem we reduced the difference of both machine clocks by using a closely located Network Time Protocol (NTP) server, to which we measured a round trip time below $1ms$.

For the synchronization, we have to determine round trip time and clock difference of both machines every time we start our application. In order to achieve this, packets are sent and received on each machine, in which the actual time on both machines is stored. The clock difference can then be computed from the round trip time RTT and the difference of the times taken, with the physics server as the reference [6]. With T_1 being the recorded time on the ma-

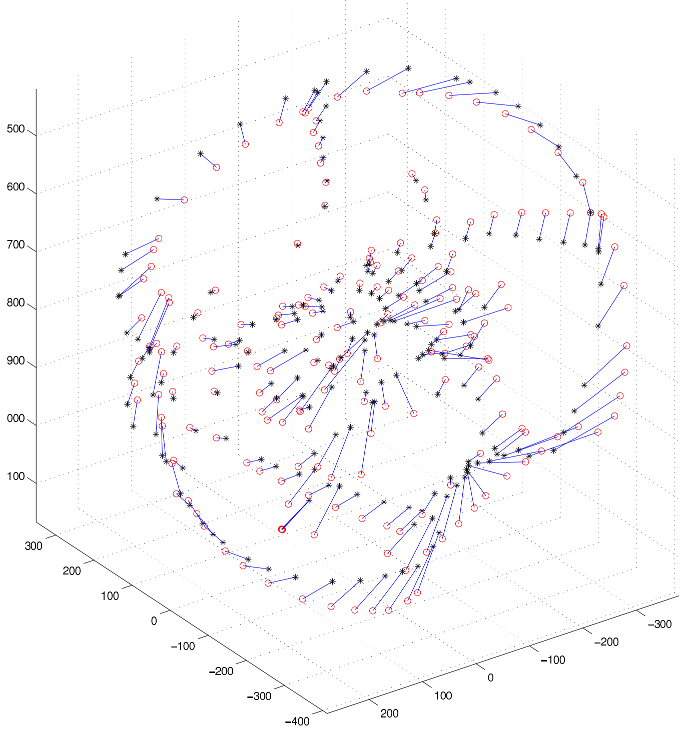

Figure 11: The two sets of points before optimization

chine starting the communication, and T_2 being the one of the other machine, the clock difference ΔT for the first processor is given by:

$$\Delta T = \frac{RTT}{2} + (T_2 - T_1) \tag{9}$$

To avoid the effect of network jitter, this measurement is carried out multiple times, and the mean of the results is used for the application. In our testbed system, round trip time was in the range of $200-300\mu s$. Thus, with our procedure the clocks of both machines can be synchronized with an accuracy of about $100\mu s$.

5.3 Communication model

For communication between the servers in our distributed framework, three requirements have to be fulfilled. First, object data to be transmitted have to correspond to a given timestep [9] as precisely as possible, so that the optimal correlation between image and virtual objects is given. Secondly, the communication should not limit the computations on both machines to a specific update rate, or affect them in any other way even if packages arrive too late. Lastly, the model should be easily adaptable for other applications.

To meet these requirements, a ring data buffer has been implemented, which stores the information of the physics computation at $1kHz$. Thus, data about interactive objects can be provided at any timestep with $500\mu s$ accuracy. When the process of generating a new image starts on the graphics server, a request is sent to update the object data for the time the image was taken. After sending the request, the computation on the graphics server can continue. The physics server receives the request, retrieves the data for the specified time from the ring buffer, and sends them back via the network. Again, this does not interrupt or affect the physics computation. In order to provide the best possible synchronization, the clock difference - determined as previously described - is also taken into account. When the data arrive at the graphics server, they are stored temporarily, and can be retrieved when finally needed for visualization at the end of the graphics loop. Figure 13 illustrates the communication for one iteration on the graphics server. It should be noted, that in the rare case that a packet arrives belatedly, the previously received information is used for rendering, so that the loop is not forced to wait.

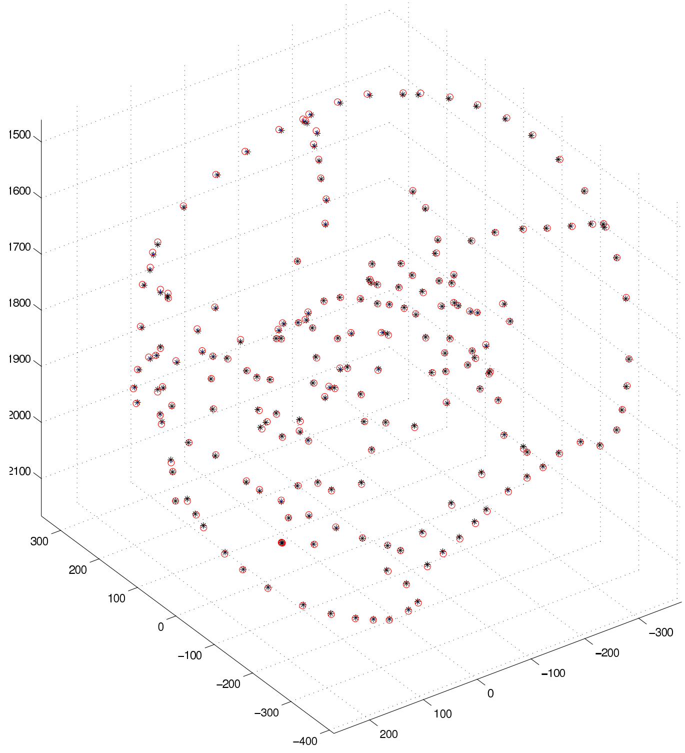

Figure 12: The two sets of points after optimization

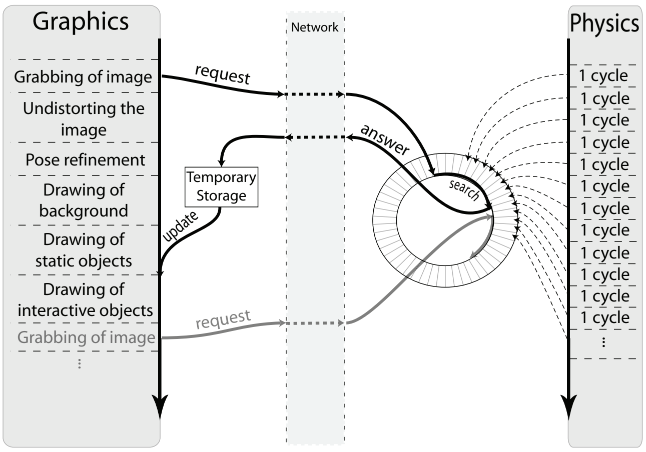

Figure 13: Communication model to exchange simulation data

Usually, enough time is available for communication during image undistortion, camera pose refinement, and drawing of background and static objects. One round trip of the communication takes about $300-800\mu s$ for small-sized packets. Of this, $100-500\mu s$ are necessary to find the most accurate answer in the ring buffer. This compares well to the time spent on the other tasks. For instance, drawing the recorded image to the display buffer takes about $3-5ms$. Therefore, the time window is usually sufficient to transmit the requested data without running into network latency problems. Additional time is even available, which could be used to transmit further information if desired.

6 Augmented Reality Haptics Ping-Pong

6.1 Overview

In order to provide a testbed scenario for the high precision augmented reality haptic system, we selected the game of ping-pong. This choice has been driven by the high level of interactivity and necessary precision to enable natural interaction. The main idea is to play with a virtual ball in a real environment. The ball collides with the real table and backwall, while the user interacts with it by using a virtual bat attached to the haptic device. This allows the user to feel the impact of the simulated ball on the virtual bat. The augmented visual scene is displayed to the user via a head mounted display. Snapshots of the application taken from an external camera are illustrated in Figure 14.

6.2 Physics engine

As mentioned above, the physical simulation has to provide updates of position and rotation to the graphics server. In our intial prototype we only use a simplified model for rigid body dynamics. The ball is considered as a non-deformable sphere, and represented by a dimensionless point in its center. The overall movement is computed with standard equations of motion:

$$x_{t+1} = x_t + v_t \cdot dt + \frac{1}{2} a_t \cdot dt^2 \tag{10}$$

$$v_{t+1} = v_t - a_t \cdot dt \quad a_{t+1} = g + \frac{F_t}{m}$$

In the current version, ball rotation and surface friction is not taken into account. Ball mass and size were obtained from a real ping-pong ball. Mass and size of the bat, as well as damping coefficients were manually set. Simple energy dissipation is implemented when the ball collides with the table, backwall, or bat. To ensure realistic interaction, fast collision detection has to be performed.

6.3 Collision detection

Our collision detection method is based on the intersection of the ball trajectory with triangular meshes representing objects in the scene. In order to enable collisions with fixed, real objects, we first have to register their positions in world coordinates. This is done with a procedure similar to the tip-marker calibration. A tracked, handheld pointer is used to define four corners both for table and backwall, respectively. This allows us to represent the surfaces of these real objects with fixed triangle meshes in the simulation. The virtual bat is also modelled with a triangular mesh, and moves with the haptic device. The stylus of the PHANToM can be thought of as the handle of the bat.

In order to detect collisions, we extrapolate position and velocity of the ball one timestep into the future. If the anticipated trajectory of the ball intersects with the triangle meshes of the bat or the fixed objects, we determine the appropriate collision repsonse. Since the bat is moving, we also have to extrapolate its kinematics, and in case of a collision, render haptic feedback. To determine a possible collision, we compute an algebraic solution of the ray-triangle intersection problem according to [15]. The barycentric coordinates of the intersection are determined, which are either inside or outside of the triangle area. This test has to be done for all triangles of the meshes. Since only simple geometries are used in our initial testbed, no enhanced collision detection methods are currently applied.

6.4 Collision response

We model the collisions of the ball with the triangular representations of real or virtual objects with a damped mass-spring system (DMSS). If a collision is detected, the energy dissipation of the ball due to the contact is modelled with the DMSS. Moreover, appropriate interaction forces are also obtained from this model.

Generally speaking, the process represents a damped oscillation along one axis, since during contact the spring is compressed and released only once. In analytical form the oscillation is given by:

$$x(t) = A \cdot e^{-\frac{b \cdot t}{2m}} \cdot \sin(w_d \cdot t) \tag{11}$$

where A is the amplitude of the oscillation, b the damping coefficient, w_d the period of the pure oscillation, and m the mass attached to the spring. According to Newton's second law, we can thus obtain the forces from this motion:

$$F(t) = -b \cdot \dot{x} - k \cdot x \tag{12}$$

Inserting equation 11 into 12 we get:

$$F(t) = \frac{1}{w_d} \cdot e^{-\frac{b \cdot t}{2m}} \Big((\frac{b^2}{2m} - k) \cdot \sin(w_d \cdot t) - b \cdot w_d \cdot \cos(w_d \cdot t) \Big) \tag{13}$$

Here, the damping coefficient b is set, so that the collision time equals that of a real ping-pong interaction - i.e. about $4ms$. In the time period, during which the spring is compressed and released, we use equation 13 to render forces to the haptic device. Since the computation is independent of current velocity or penetration depth, we can precompute the values at the start of the simulation. Therefore, we sample the equation at individual time-steps and create a look-up table (LUT). During interaction we can then retrieve these data. Moreover, to incorporate the influence of the ball velocity, we scale the values appropriately, thus determining the force feedback according to:

$$F_{feedback} = |v_t^{ball}| \cdot LUT(\tilde{t}) \qquad \tilde{t} \in [0, t_{dwell}] \tag{14}$$

Here, $\tilde{t}$ represents discretized steps of the time period t_{dwell} during which the ball is in contact with the bat.

7 Conclusion and Future Work

We have developed a high precision augmented reality haptics system to be applied for medical training scenarios. To this end, an accurate calibration procedure for integrating the haptic device, as well as a distributed framework has been created. As a testbed we have built a ping-pong game, which allows visual and haptic interaction with real and virtual objects. One shortcoming of our current system is the lack of stereo rendering. Since depth cues are only present due to shadows and motion parallax, some difficulties in judging the flight path of the ball became apparent. This can easily be overcome in future versions, by adding a stereo camera to the setup.

Acknowledgement

The authors would like to thank Roberto Celli and Markus Berner for the development and integration of the ping pong game into the AR haptics system.

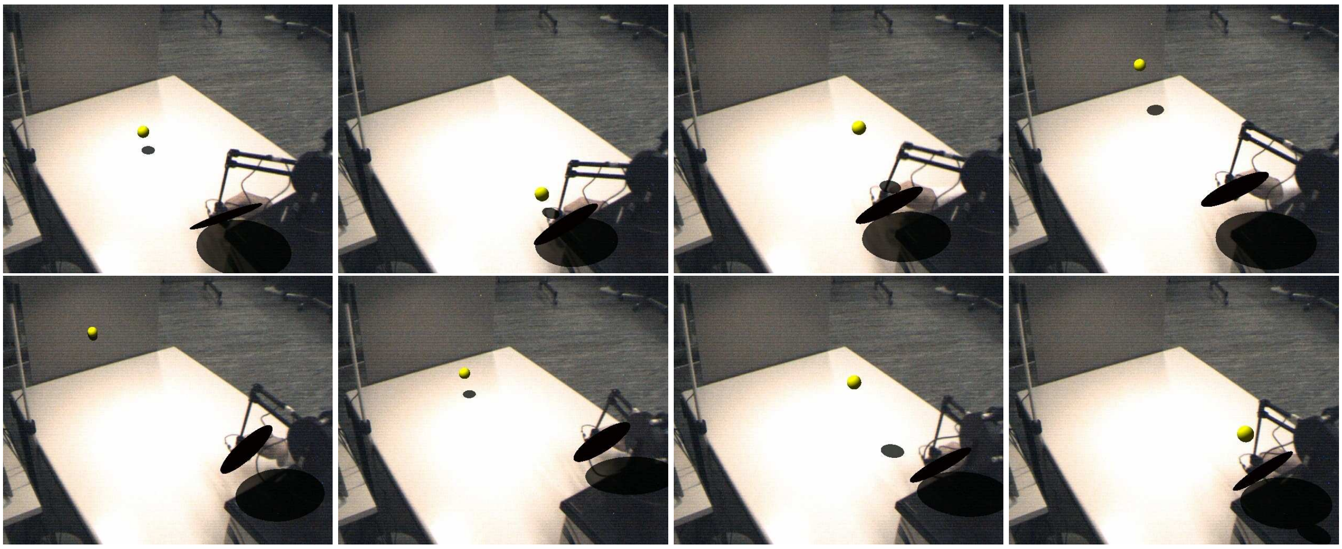

Figure 14: Snapshots of user playing AR ping-pong

This work has been performed within the frame of the Swiss National Center of Competence in Research on Computer Aided and Image Guided Medical Interventions (NCCR Co-Me) supported by the Swiss National Science Foundation.

References

[1] Matt Adcock, Matthew Hutchins, and Chris Gunn. Augmented reality haptics: Using ar toolkit for display of haptic applications. In *Proceedings of 2nd IEEE International Augmented Reality Toolkit Workshop*, 2003.

[2] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d point sets. *IEEE Trans. Pattern Anal. Mach. Intell.*, 9(5):698–700, 1987.

[3] G. Bianchi, C. Wengert, M. Harders, P. Cattin, and G. Székely. Camera-Marker Alignment Framework and Comparison with Hand-Eye Calibration for Augmented Reality Applcations. In *ISMAR*, 2005.

[4] M. C. Cavusoglu, D. Feygin, and F. Tendick. A critical study of the mechanical and electrical properties of the phantom haptic interface and improvements for high-performance control. *Presence: Teleoper. Virtual Environ.*, 11(6):555–568, 2002.

[5] B. Grant, A. Helser, and II Taylor, R.M. Adding force display to a stereoscopic head-tracked projection display. In *Virtual Reality Annual International Symposium, 1998. Proceedings IEEE 1998*, pages 81–88, Atlanta, GA, 1998.

[6] F. G. Hamza-Lup and J. P. Rolland. Scene synchronization for real-time interaction in distributed mixed reality and virtual reality environments. *Presence: Teleoper. Virtual Environ.*, 13(3):315–327, 2004.

[7] J. M. Hollerbach and C. W. Wampler. The calibration index and taxonomy for robot kinematic calibration methods. *Int. J. Rob. Res.*, 15(6):573–591, 1996.

[8] http://www.ndigital.com.

[9] R. M. Taylor II, T. C. Hudson, A. Seeger, H. Weber, J. Juliano, and A. T. Helser. VRPN: a device-independent, network-transparent vr peripheral system. In *VRST '01: Proceedings of the ACM symposium on Virtual reality software and technology*, pages 55–61, New York, NY, USA, 2001. ACM Press.

[10] M. Ikits, C.D. Hansen, and C.R. Johnson. A comprehensive calibration and registration procedure for the visual haptic workbench. In *Joint Immersive Projection Technology and Eurographics Virtual Environments Workshop*, pages 247–254, 2003.

[11] S. Lavalle, P. Cinquin, and J. Troccaz. *Computer Integrated Surgery and Therapy: State of the Art*. IS Press, Amsterdam, NL, in C. Roux and J.L. Coatrieux edition, 1997.

[12] A. Lorusso, D. Eggert, and R.B. Fisher. A comparison of four algorithms for estimating 3-d rigid transformations. In *British Machine Vision Conference*, 1995.

[13] C. Luciano, P. Banerjee, L. Florea, and G. Dawe. Design of the immersivetouch^{TM}: a high-performance haptic augmented virtual reality system. In *11th International Conference on Human-Computer Interaction, Las Vegas, NV*, 2005.

[14] William R. Mark, Scott C. Randolph, Mark Finch, James M. Van Verth, and Russell M. Taylor II. Adding force feedback to graphics systems: Issues and solutions, 1996.

[15] T. Möller and B. Trumbore. Fast, minimum storage ray-triangle intersection. *journal of GRAPHICS-TOOLS*, 2(1):21–28, 1997.

[16] T. Nojima, D. Sekiguchi, M. Inami, and S. Tachi. The smarttool: a system for augmented reality of haptics. In *Proceedings of IEEE Virtual Reality, 2002*, pages 67–72, 2002.

[17] K. Reinig, R. Tracy, H. Glimore, and T. Mahalik. Some calibration information for a phantom 1.5 a. In *Proceedings PHANTOM User Group Workshop*, 1997.

[18] Usman Sargaana, Hossein S. Farahani, Jong Weon Lee, Jeha Ryu, and Woontack Woo. Collaborative billiARds: Towards the ultimate gaming experience. In *ICEC*, pages 357–367, 2005.

[19] C. Scharver, R. Evenhouse, A. Johnson, and J. Leigh. Pre-surgical cranial implant design using the PARIS/spl trade/ prototype. *2004. Proceedings. IEEE Virtual Reality*, pages 199–291, 2004.

[20] Valerie A. Summers, Kellogg S. Booth, Tom Calvert, Evan Graham, and Christine L. MacKenzie. Calibration for augmented reality experimental testbeds. In *SI3D '99: Proceedings of the 1999 symposium on Interactive 3D graphics*, pages 155–162. ACM Press, 1999.

[21] J. Vallino and C. Brown. Haptics in augmented reality. In *Multimedia Computing and Systems, 1999. IEEE International Conference on*, volume 1, pages 195–200, Florence, 1999.

[22] C. Woodward, P. Honkamaa, J. Jppinen, and E.-P. Pykkimies. Camball - augmented virtual table tennis with real rackets. In *ACE Singapore*, pages 275–276, July 2004.

[23] Guangqi Ye, Jason J. Corso, Gregory D. Hager, and Allison M. Okamura. Vishap: Augmented reality combining haptics and vision. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, pages 3425–3431, 2003.