Control Algorithms for 3-DoF Handheld Robotic Devices Used in Orthopedic Surgery

Martin Klemm[*,‡], Uwe D. Hanebeck[†,§], Harald Hoppe[*,¶]

[*]*Faculty of Electrical Engineering and Information Technology*
Offenburg University, Badstrasse 24, Offenburg 77652, Germany

[†]*Institute for Anthropomatics and Robotics*
Karlsruhe Institute of Technology, Karlsruhe 76131, Germany

Nowadays, robotic systems are an integral part of many orthopedic interventions. Stationary robots improve the accuracy but also require adapted surgical workflows. Handheld robotic devices (HHRDs), however, are easily integrated into existing workflows and represent a more economical solution. Their limited range of motion is compensated by the dexterity of the surgeon. This work presents control algorithms for HHRDs with multiple degrees of freedom (DOF). These algorithms protect pre- or intraoperatively defined regions from being penetrated by the end effector (e.g., a burr) by controlling the joints as well as the device's power. Accuracy tests on a stationary prototype with three DOF show that the presented control algorithms produce results similar to those of stationary robots and much better results than conventional techniques. This work presents novel and innovative algorithms, which work robustly, accurately, and open up new opportunities for orthopedic interventions.

Keywords: Handheld robotic device; orthopedic surgery; control algorithms.

1. Introduction

Since the first intervention was performed with ROBODOC (Curexo, USA) 20 years ago [1], robotic systems became an integral part of many orthopedic surgeries. Whereas ROBODOC executed the milling procedure automatically, modern robotic milling devices, such as Mako RIO (Stryker, USA), are guided by the surgeon. Higher accuracy, reproducibility and the ability to execute complex milling procedures are just some of the advantages of these cooperative robots. On the one hand, they perform the milling procedures (semi-)automatically and increase the final accuracy. On the other hand, they produce increasing costs [2] and cause additional training time since the surgical workflow has to be adapted to the robot. Handheld Robotic Devices (HHRDs), however, are easily integrated into existing workflows and represent a more economical solution. Within the scope of this paper, an HHRD is defined to be a tool with an end effector such as a burr or a saw blade that can be dynamically relocated using one or several joints between the end effector and the handle (see Fig. 1). Whereas stationary robots have more degrees of freedom (DoF), HHRDs have only a limited range of motion but instead make use of the dexterity of the surgeon.

Stationary cooperative robots are often force-controlled: the surgeon (or a virtual constraint) applies a force to the handle of the end effector and the robot reacts with an appropriate motion, respectively, counterforce. HHRDs are obviously not able to apply counterforces to the surgeon's hand. Therefore, control systems for HHRDs are normally position-controlled considering the position of the HHRD's handle, the position of the end effector and the positions of the virtual constraints.

A high level collaborative control strategy, called virtual fixture (VF), provides assistance to the user by controlling the robot in such a way that predefined

Received 2 October 2017; Revised 28 May 2018; Accepted 5 July 2018; Published 30 August 2018. This paper was recommended for publication in its revised form by Editor, Jaydev P. Desai.
Email Addresses: ‡*martin.klemm@hs-offenburg.de*, §*uwe.hanebeck@kit.edu*, ¶*harald.hoppe@hs-offenburg.de*

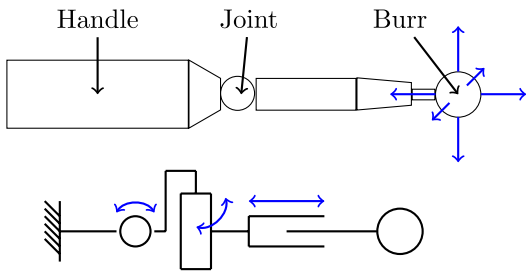


Fig. 1. A HHRD with 3-DOF and a milling end effector. This device can be modeled using two rotational and one translational joint.

regions or targets are either protected or approached. The name VF was originally published by Rosenberg [3] and is inspired by mechanical fixtures that anisotropically limit the motion of tools, e.g., a ruler. However, VFs are more flexible according to the positioning and modification. VFs can be split into two categories:

- Regional VF restricts the pose of the device to predefined regions in order to prevent the device from harming other regions.
- Guidance VF assists the user in moving the device along a specific path or towards a specific target.

Conventionally, these VFs (constraints) are evaluated by calculating the proximity (or collision) between the VF and the robot's end effector. Subsequently, the robot motion is determined. Rosenberg states that VFs reduce mental workload, execution time and errors. An extensive survey of VFs and active constraints is given in [4]. VFs were already applied on many types of robots in different fields. Rosenberg originally developed VF for teleoperation tasks, therefore it is reasonable that VF was implemented on teleoperated surgical robots such as ZEUS (Computer Motion Inc., USA) [5]. Xia *et al.* [6] and Haidegger *et al.* [7] describe the skull base neurosurgery project of Johns Hopkins University, which implements VF on a stationary cooperative milling robot. Another robotic system of Johns Hopkins University implementing VF is the microsurgical steady-hand eye robot [8]. Becker *et al.* [9] describe a handheld microsurgical robot based on guidance VF.

First approaches in the field of handheld surgical robotic devices concentrated on navigated and controlled milling devices. Although these devices are rigid, big parts of the control algorithm are comparable to those of robots. Kneissler *et al.* [10] present such a system used for spine surgery in which the milling speed of the device is controlled depending on its position. The workspace of the device is defined by the preoperative plan which is based on a CT or MRI scan. Even though not explicitly mentioned, the closed milling volume represents a regional VF. The surgeon holds and guides the milling device as usual. As soon as the device's end effector touches the virtual boundaries, the milling speed is set to 0. They state that this approach results in accuracies comparable to robot-controlled executions. Additionally, they report high-frequency speed changes at the boundaries of the workspace. Every sudden change caused a jerk at the hand of the operator which complicated the handling and produced ragged edges.

In contrast to that, an HHRD allows controlling the milling speed as well as the end effector's position. Touching the boundaries of the virtual constraints results in a deflection of the end effector such that it does not penetrate the boundary. Only if the maximum deflection is reached, the milling speed is set to 0. This decreases the frequency of speed changes and improves the accuracy. Brisson *et al.* [11] introduced the precision freehand sculptor that used a retractable rotary blade to control which part of the bone is removed. The surgeon simply glides the sculptor over the bone surface and the tool removes the defined parts. It can handle pre- and intraoperatively defined plans and uses an optical tracking system for navigation. However, their system does not support a partial blade retraction. The NAVIO handpiece (Smith & Nephew, UK) represents a similar device: a retractable milling device that either controls the speed or the exposure of the end effector depending on the proximity to the constraints. Lonner [12] states that this HHRD is able to place implants in unicompartmental knee arthrosplasty as well as stationary robots and more accurately than conventional techniques. The availability of only one DoF limits the ease of use especially with regard to more complex working volumes including cavities or indentations. Even a simple cylindrical hole with a diameter slightly bigger than that of the milling cutter gets demanding if the direction of the tool is not perfectly aligned with the hole. Therefore, a 3-DoF HHRD is much more desirable.

Riviere *et al.* [13] present Micron: a 3-DoF micromanipulator with a 6-DoF inertial sensor able to compensate the surgeon's tremor. Becker *et al.* [9] implemented different control modes for Micron. Besides others, a "standoff-regulation" mode implements a guided VF and prevents from accidental unwanted contact by repulsing the end effector within a defined range to the center of a sphere. This does not prevent the end effector from penetrating the area but only repulses it from the center. This mode only works for points, respectively, spheres and is not applicable for arbitrary VF.

Although similar systems already exist for related types of robots such as stationary cooperative, teleoperated or microsurgical robots, the authors are not aware of works in the field of handheld milling robots. Therefore — to the best of the authors' knowledge — this work introduces novel and innovative control strategies and includes the following contributions:

(1) A regional VF control system for arbitrary surgical milling HHRDs with at least 3-DoF;

(2) working with bilateral constraints, more specifically, arbitrary, nonclosed triangle meshes and
(3) which is based on position control rather than force control.
(4) The control system also includes acoustic signals supporting the surgeon if the HHRD has to be repositioned and
(5) the paper comprises the evaluation of the presented algorithms by means of an accuracy comparison between different implemented modes (rigid, evasive and smoothing) using a table-top robotic milling device.

2. Material and Methods

2.1. *Overview*

The applied software comprises two components: OrthoCAD is used for calibrating the HHRD, for visualization, and for generating the constraints needed for the tests. The Active Tool Control (ATC) is used for controlling the HHRD. Figure 2 shows the interaction between OrthoCAD and the ATC.

2.2. *OrthoCAD*

OrthoCAD is a CAD system adapted to orthopedic interventions enabling surgeons to intraoperatively plan and execute orthopedic surgeries. The surgeon can switch between planning and execution as often as necessary. The planning is generic, which means that the software offers a selection of elementary objects such as points, lines and planes. Subsequently, these objects can be assembled to more complex structures. Surfaces can be recorded and modified. Distance and angle measurements are offered. A snapping functionality allows digitizing new objects in relation to already created objects (e.g., perpendicular, parallel, or through). The surgeon interacts with the system using a pointing device and a touch screen. Since all these digitized or assembled structures can be used as virtual constraints, closed milling volumes are rather the exception than the rule. Therefore, the following algorithms are designed to work with arbitrary triangle meshes. This implies that constraints are treated as impenetrable walls without distinction between a fly zone and a no-fly zone.

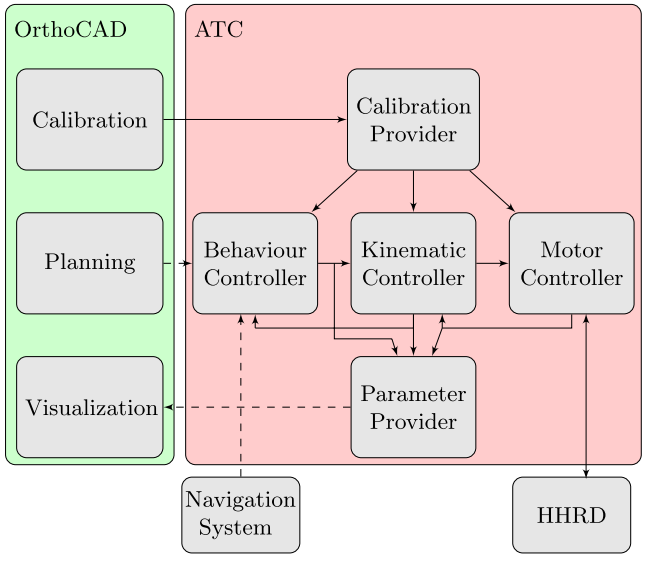


Fig. 2. Overview of the software components of the ATC and OrthoCAD. The BC receives the planning data from OrthoCAD and passes the computed speed and target position to the KC. The KC calculates the joint parameters and passes them to the MC. The MC updates the HHRD and reads the current joint parameters. The calibration provider imports the calibration parameters and updates the controllers. The parameter provider collects calculated parameters and supplies them to the visualization on request.

2.3. *Active tool control*

The ATC ensures that the end effector does not penetrate the given constraints consisting of arbitrary triangle meshes. There are two ways to react in case that the end effector touches one of these constraints: As long as possible, the end effector deflects in order not to penetrate the constraint (see Fig. 3). If this is not possible anymore, the end effector's power is turned off to put it into a safe state.

The end effector is located at its home position (no deflection) if no constraints are touched. Otherwise, the end effector is located at a to be calculated target position where it touches the constraints such that it is nearest to its home position.

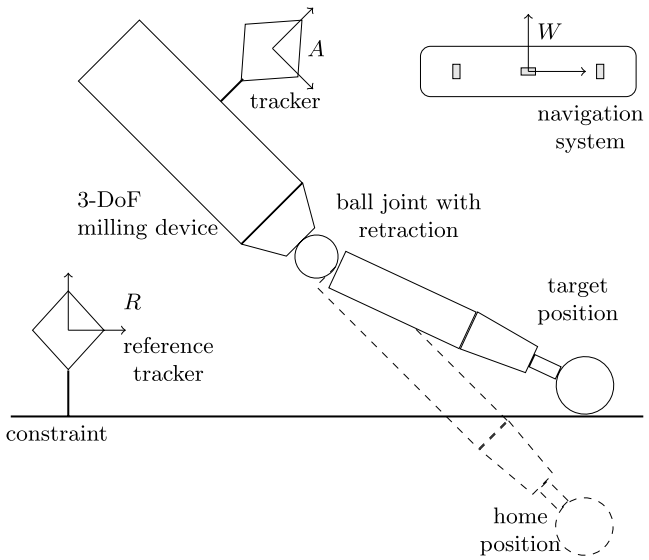


Fig. 3. A 3-DoF milling device that collided with a constraint and adjusted its end effector position such that the distance between home position and target position gets minimal.

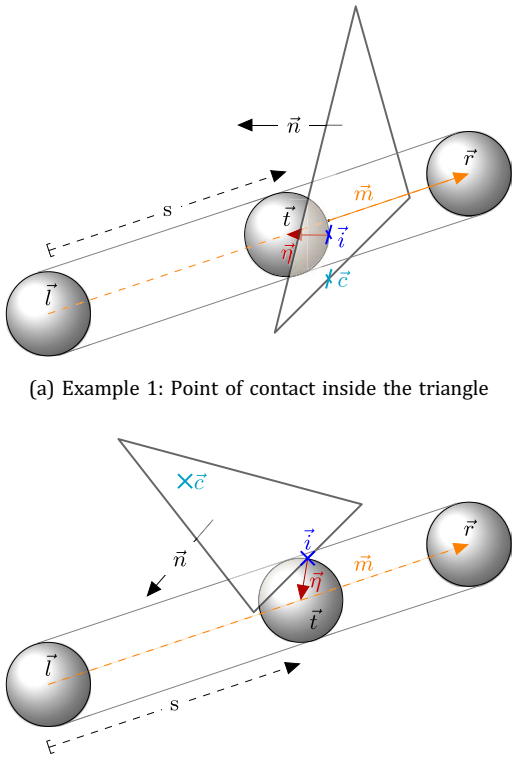
The ATC consists of several components: The Behavior Controller (BC), the Kinematic Controller (KC) and the Motor Controller (MC) are responsible for controlling the HHRD. Additional to the controllers, there are two components for reading input data and providing data to components outside the ATC. The BC receives the planning data and the current transformations and computes the target position as well as the end effector speed (see Secs. 2.5 and 2.6). These parameters are passed to the KC that calculates the joint parameters according to a kinematic model (inverse kinematics). These joint parameters are sent to the MC that communicates with the motors. Besides this forwarding pipeline, there is also one for feedback. The MC reads the current joint parameters and passes them to the KC. The KC computes the current position and passes it to the BC (forward kinematics).

The BC is completely independent from the hardware and can be used for arbitrary robotic devices. The position control algorithm performs a collision detection in order to find a valid target position where the end effector does not penetrate any constraint. The kinematic model of the KC has to be adapted to the hardware. Additional to the forward and the inverse kinematics, it must also perform plausibility checks for the reachability of a calculated target position. It simply ensures that the computed parameters are within a range that the hardware can reach. In case that these parameters are outside the reachable range, the MC switches off the end effector. The ATC currently expects spherical burrs, however, differently shaped burrs can also be used after adapting the collision detection algorithm.

2.4. *Collision detection algorithm*

The collision detection algorithm calculates all collisions between the given constraints and the end effector being moved from its last valid target position to its home position. A typical discrete approach (*a posteriori*) finds a collision after it occurred. A continuous approach (*a priori*) predicts the movement of the objects and finds a collision before it occurs. In the presented setup, a continuous approach cannot be applied since there are two movements: the movement of the surgeon's hand and the movement of the end effector (caused by the joints). Whereas the movement of the end effector is controlled, the movement of the surgeon's hand is difficult to predict since there is only an optical navigation system without additional sensors (tremor prediction normally uses high-speed 6-DoF gyroscopes, see for example, [9]). Therefore, a discrete approach is used that calculates the collision for a given sphere movement (spherical end effector).

The convex hull of a linear sphere movement trajectory is a capsule. Therefore, the underlying algorithms are optimized for finding collisions of capsules with



(a) Example 1: Point of contact inside the triangle

(b) Example 2: Point of contact at the edge of the triangle

Fig. 4. Two examples of a sphere movement colliding with a triangle. All parameters calculated in the collision detection are shown. Note the difference between the direction vectors $\vec{\eta}$ and $\vec{n}$, respectively, the points $\vec{i}$ and $\vec{c}$.

given triangle meshes. The sphere movement is defined by the start position $\mathbf{l}$ and the home position $\vec{r}$ of the sphere center (see Fig. 4). The algorithm returns a factor s that indicates how far the sphere can move from $\mathbf{l}$ to $\vec{r}$ until it touches the triangle. $s = 0$ means that the sphere at position $\mathbf{l}$ already touches the triangle, $s = 1$ indicates that the sphere can move from $\mathbf{l}$ to $\vec{r}$ without touching it. Further parameters (see Fig. 4) are

- $\vec{n}$: the normal vector of the triangle;
- $\vec{c}$: the nearest point to the sphere on the triangle;
- $\vec{m}$: the move vector from the last valid position $\mathbf{l}$ to the home position $\vec{r}$ with $\vec{m} = \vec{r} - \mathbf{l}$;
- $\vec{t}$: the (target) position of the sphere center where the sphere touches the triangle on its way from $\mathbf{l}$ to $\vec{r}$;
- $\vec{i}$: the corresponding point of contact of the sphere on the triangle;
- $\vec{\eta}$: the direction vector pointing from the point of contact $\vec{i}$ to the sphere center $\vec{t}$ with $\vec{\eta} = \vec{t} - \vec{i}$.

Subsequently, $\vec{\eta}$ is called **move plane normal** since it describes the normal vector of the plane through the sphere center defining the half space in which the sphere can move without intersecting the triangle. $\vec{\eta}$ equals $\vec{n}$ if the point of contact lies inside (not on the edge) the triangle.

First of all, the nearest point $\mathbf{c}$ to the sphere on the triangle is calculated. Given that the angle between the desired move vector $\vec{m}$ and the vector pointing from $\mathbf{l}$ to this nearest point $\mathbf{c}$ is smaller than $90°$, a collision may occur. If this is the case, the position of the sphere center $\vec{t} = \mathbf{l} + s_{\min} \cdot (\vec{r} - \mathbf{l})$ where the sphere first touches the triangle and the corresponding point of contact $\vec{i}$ are calculated. The resulting move plane normal is $\vec{\eta} = \vec{t} - \vec{i}$.

2.5. *Position control algorithm*

Calculating and correcting the end effector position such that the constraints are not penetrated is the elementary task of this component. As long as the end effector does not touch any constraint, the target position $\vec{t}$ equals the home position $\vec{r}$. If the user places the HHRD such that the home position collides with the constraints, the target position is adapted. The subsequently described algorithm minimizes the distance between the target position $\vec{t}$ and the home position $\vec{r}$ by sliding along all constraints currently being touched by the end effector sphere. The resulting target position is stored and used as last valid position $\mathbf{l}$ for the next iteration.

Figure 5(a) shows an example of a movement $\mathbf{l} \rightarrow \vec{r}$ that collides with the constraint after a certain distance. In this case, the algorithm moves the sphere until it touches the constraint and terminates.

Consider the example in Fig. 5(b). The last valid position $\mathbf{l}$ already touches one of the constraints. Therefore, the desired move vector $\vec{m} = \vec{r} - \mathbf{l}$ is projected onto the touched constraint resulting in a new and allowed move vector $\vec{m}_1$. $\vec{\tau}_1 = \vec{l} + \vec{m}_1$ is called the projected home position, where the subscript identifies the phase number k with $0 \leq k \leq 2$. The new movement $\mathbf{l} \rightarrow \vec{\tau}_1$ could intersect other constraints and another collision detection has to be performed. Given that $s_{\min} > 0$ for this new move vector, the target position equals $\vec{t} = \vec{l} + s_{\min} \cdot (\vec{\tau}_1 - \mathbf{l})$.

Since the projected movement could intersect other constraints, the collision detection has to be performed once in each phase. Performing a collision detection up to three times per iteration guarantees that the end effector position is optimized with respect to the home position and that the executed movement does not penetrate any constraint. In each iteration, the home position and the last valid position have to be considered since they are moving relative to each other.

This iterative position control algorithm can be summarized in the following steps (see Fig. 6):

(1) Calculate the projected home position $\vec{\tau}_k$ according to the already touched triangles from the previous phases and the movement from $\mathbf{l}$ to $\vec{r}$. In the first phase $\vec{\tau}$ equals $\vec{r}$.
(2) In case the projected home position $\vec{\tau}_k$ equals the last valid position $\mathbf{l}$, the algorithm is terminated.
(3) Otherwise, a collision detection for the sphere movement from the last valid position to the projected home position $\mathbf{l} \rightarrow \vec{\tau}_k$ is performed (see Sec. 2.4).
(4) The smallest allowed movement is determined (triangles from the previous phases are ignored).
(5) If there is a possible movement ($s_{\min} > 0$), the target position is set and the algorithm is terminated.
(6) Otherwise, the next phase is started.

2.5.1. *Move plane normal versus triangle normal*

Every touched triangle defines a move plane normal $\vec{\eta}$ and a factor s which indicates how far the end effector can move from $\mathbf{l}$ to $\vec{\tau}_k$. Remind that the move plane normal defines the normal vector of the plane through the sphere center in which the end effector is allowed to move relative to the triangle without intersecting it. If there are two different move plane normals with $s = 0$, the space is restricted to the intersection line of the two corresponding planes. In this case, the end effector can be moved along this line (see Figs. 5(c) and 5(d)). Given the case that there are three different move plane normals with $s = 0$, they intersect in one point which results in the fact that the projected home positions equal the last valid position and that the end effector cannot be moved any more. One example would be a movement through the tip of a tetrahedron while touching the three faces of the tip.

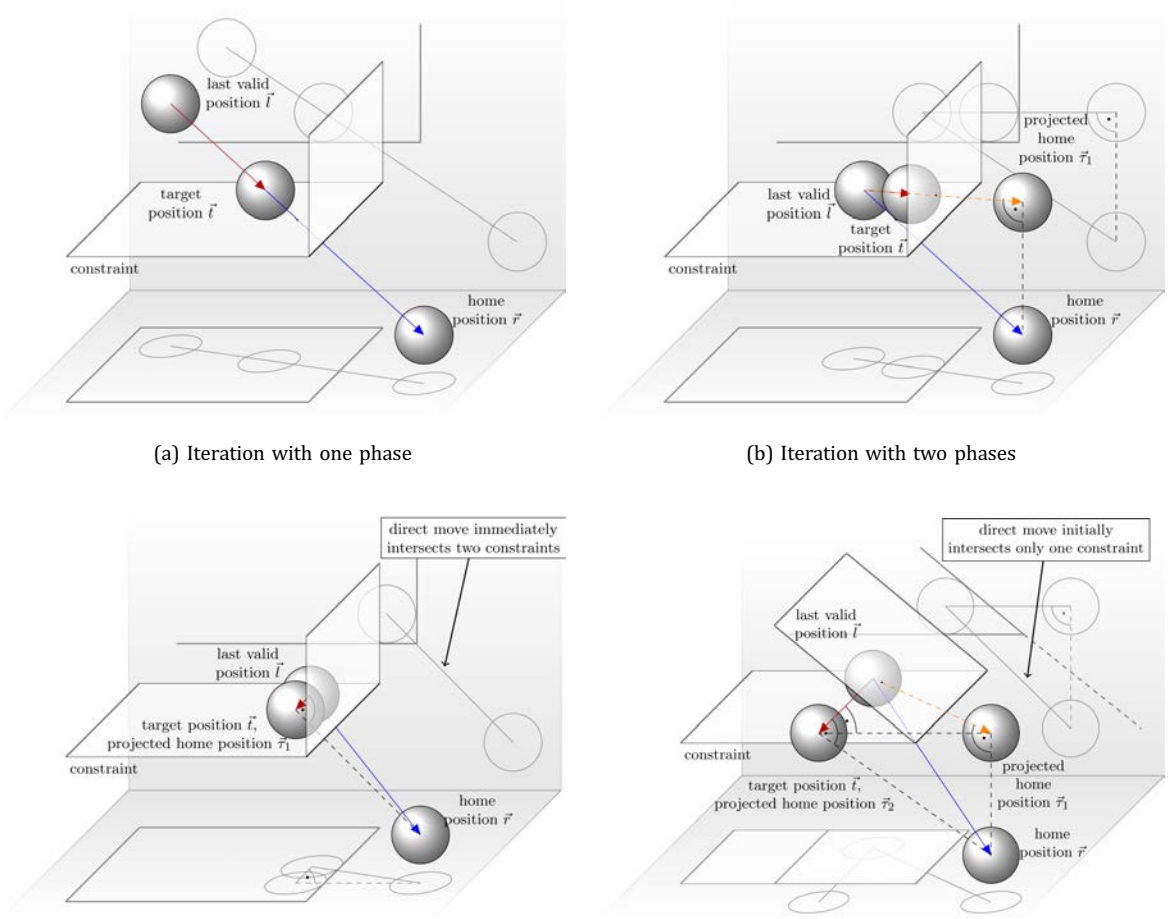
The move plane normal (as defined in Sec. 2.4) is parallel to the triangle's normal if the point of contact lies inside the triangle and differs if it lies on one of the triangles' edges or corners. Figure 7 describes the importance of using the move plane normal instead of the triangle's normal. Using the move plane normal guarantees that the sphere slides around the edge of a triangle which results in smoother movements around corners and edges.

2.5.2. *Calculating the allowed movement*

The collision detection algorithm returns a move plane normal $\vec{\eta}_{k,i}$ with corresponding factor $s_{k,i}$ ($0 \leq i < C_k$) for every collision. Since the nearest collision is the most important, only the move plane normals corresponding to the smallest factor $s_{\min,k}$ are stored. If $s_{\min,k} > 0$, the target position $\mathbf{t} = \mathbf{l} + s_{\min,k} \cdot (\vec{\tau}_k - \mathbf{l})$ is calculated and the iteration is terminated (see Fig. 5(a)).

If $s_{\min,k} = 0$, the end effector is not allowed to move from $\mathbf{l}$ to $\vec{\tau}_k$ and the next phase is started. In many cases, several triangles are immediately touched with $s_{\min,k} = 0$. The corresponding move plane normals are stored in matrix Λ_k:

$$\Lambda_k = (\vec{\eta}_{k,0} \ldots \vec{\eta}_{k,M_k-1}) \quad \text{with} \quad M_k \leq C_k. \tag{1}$$



(a) Iteration with one phase

(b) Iteration with two phases

(c) Iteration with two phases

(d) Iteration with three phases

Fig. 5. Four individual sphere movements $\mathbf{l} \rightarrow \vec{r}$ and how the position control algorithm finds the target position $\vec{t}$. The blue movement ($\mathbf{l} \rightarrow \vec{r}$) would be the optimum without considering constraints. The red one ($\mathbf{l} \rightarrow \vec{t}$) is the final movement of the end effector. The dashed orange ones ($\mathbf{l} \rightarrow \vec{\tau}$) are theoretical intermediate movements. In (a), the end effector can move partially and on a direct way towards the home position. In (b), a direct movement is not possible. A collision is detected and a second phase is started. In the second phase, the move vector is projected onto the constraint and the end effector can move partially until it hits the second constraint. In (c), a direct movement is not possible since it intersects both constraints. In the second phase, the move vector is projected onto the intersection line of the two constraints and the end effector can move. In (d), a direct movement is not possible. A collision with the lower constraint is detected and a second phase is started. In the second phase, the move vector is projected onto the constraint. The resulting move vector causes another collision and a third phase is started. In the third phase, the move vector is projected onto the intersection line of the two constraints and the end effector can move.

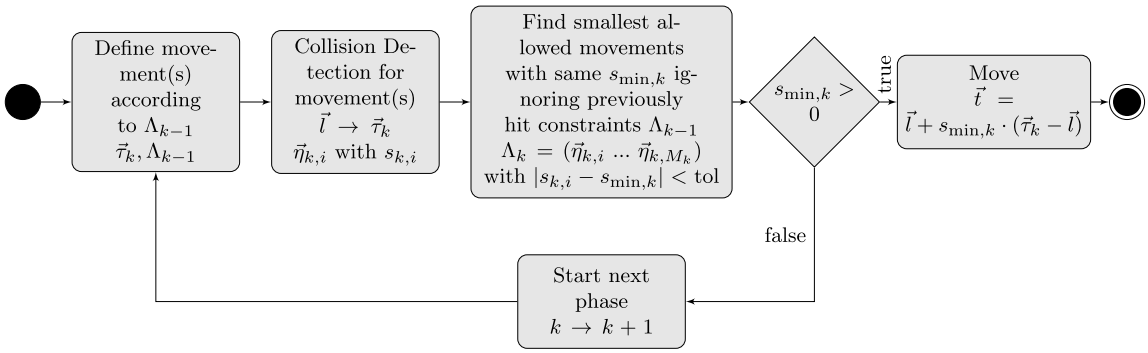
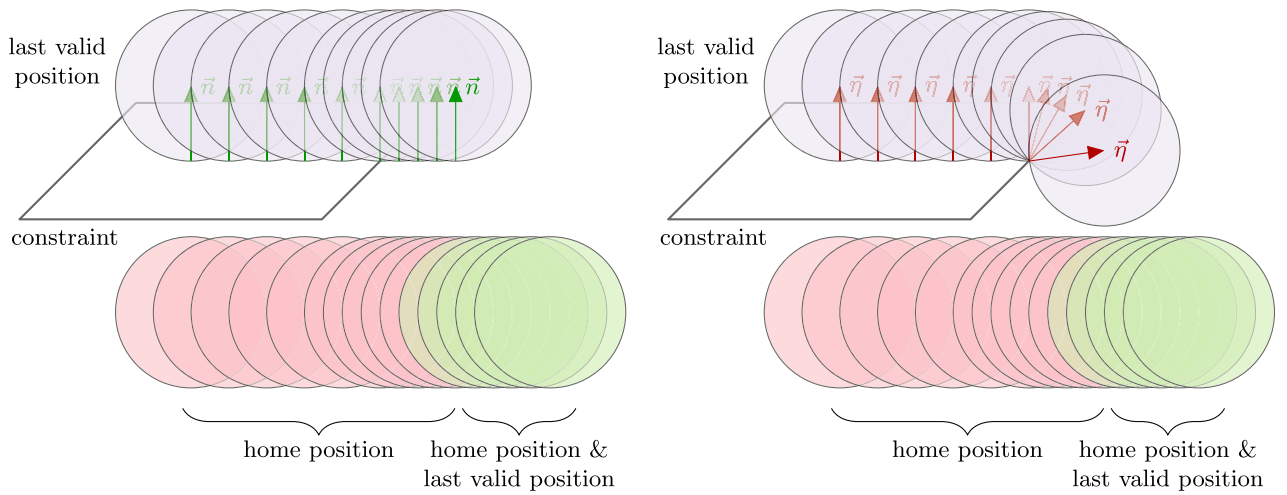


Fig. 6. The position control algorithm. Inputs are the home position $\vec{r}$ and the last valid position $\mathbf{l}$. The output is the target position $\vec{t}$. Every collision of the current movement from $\mathbf{l}$ to the projected home position $\vec{\tau}_k$ produces one pair of $\vec{\eta}_{k,i}$ with corresponding $s_{k,i}$. After the smallest allowed movement is found, only the move plane normals $\vec{\eta}_{k,i}$ with corresponding $s_{k,i} = s_{\min,k}$ are stored in Λ_k.



(a) Sphere movement when using the triangle's normal

(b) Sphere movement when using the move plane normal

Fig. 7. The difference between using the triangle's normal **n** (left) and its move plane normal $\vec{\eta}$ (right). Initially, both approaches show the same results but they differ as soon as the sphere passes the edge of the triangle. $\vec{\eta}$ is the vector from the nearest point on the triangle to the sphere center and defines the normal vector of the plane through the sphere center in which the sphere is allowed to move without intersecting the triangle. Using the move plane normal guarantees that the sphere moves around the edge whereas using the triangle's normal results in the fact that it moves straight on until it can pass the triangle without intersecting it.

2.5.3. *Defining the movement according to detected constraints*

Depending on the found constraints, the move vector $\mathbf{l} \rightarrow \vec{r}$ has to be projected onto a plane, line, or point. Figure 8 shows a flowchart of the projection algorithm (Sec. 2.5.4 explains the outlined cases).

Before the move vector can be projected, it has to be determined, how restricted the space already is. **In the first phase** ($k = 0$), there are no restrictions and $\vec{\tau}_k$ equals $\vec{r}$.

In the second phase ($k = 1$), the space is restricted by the move plane normals found in the first phase $\Lambda_0 = (\vec{\eta}_{0,0} \dots \vec{\eta}_{0,M_0-1})$. In order to find the vector space defined by these move plane normals, a singular value decomposition (SVD) is applied:

$$\mathrm{SVD}(\Lambda_0^T) = U_0 \Sigma_0 V_0^T, \tag{2}$$

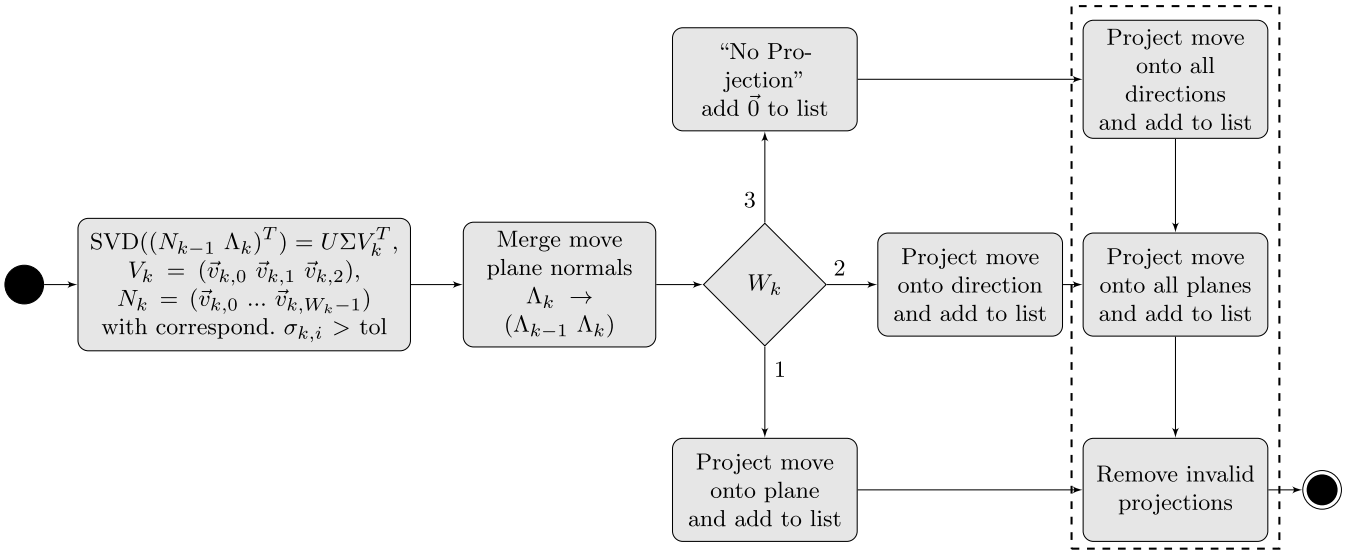


Fig. 8. The algorithm that defines the movements according to the given restrictions. For clarity reasons, the indices are incremented by 1. Inputs are the right-singular vectors N_{k-1} and the move plane normals Λ_k as well as the movement from the last valid position **l** to the home position $\vec{r}$. The output is a list of possible projected home positions $\{\vec{\tau}_{k+1,\gamma}\}$ with corresponding move plane normals $\Lambda_{k,\gamma}$. The dashed part is executed when the special cases are considered.

$$\Sigma_0 = \begin{pmatrix} \sigma_{0,0} & 0 & 0 \\ 0 & \sigma_{0,1} & 0 \\ 0 & 0 & \sigma_{0,2} \\ 0 & 0 & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & 0 \end{pmatrix}, \tag{3}$$

$$V_0 = (\mathbf{v}_{0,0}\ \mathbf{v}_{0,1}\ \mathbf{v}_{0,2}). \tag{4}$$

The interesting part of the decomposition is matrix V_0 which is a 3×3 matrix whose column vectors are the right-singular vectors of Λ_0. Each right-singular vector $\mathbf{v}_{0,i}$ corresponds to one singular value $\sigma_{0,i}, i = 0, 1, 2$. Those vectors belonging to nonzero singular values define the restrictions of the movement. They are stored in matrix $N_0 = (\mathbf{v}_{0,0}, \ldots, \mathbf{v}_{0,W_0-1})$, where W_0 equals the number of nonzero singular values $\sigma_{0,i}$.

Depending on W_0, three cases have to be considered:

(1) $W_0 = 1$: $\mathbf{v}_{0,0}$ defines the normal vector of the plane onto which the move vector has to be projected (see Fig. 5(b)).
(2) $W_0 = 2$: $\mathbf{v}_{0,0} \times \mathbf{v}_{0,1}$ defines the direction vector of the straight line onto which the move vector has to be projected (see Fig. 5(c)).
(3) $W_0 = 3$: The end effector cannot move, the move vector equals $\mathbf{0}$.

The third phase ($k = 2$) resembles the second one. In addition to the move plane normals from the second phase $\Lambda_1 = (\vec{\eta}_{1,0} \ldots \vec{\eta}_{1,M_1-1})$, the right-singular vectors corresponding to nonzero singular values of the first phase N_0 are used as input for the SVD:

$$\mathrm{SVD}((\Lambda_1\ N_0)^T) = U_1 \Sigma_1 V_1^T. \tag{5}$$

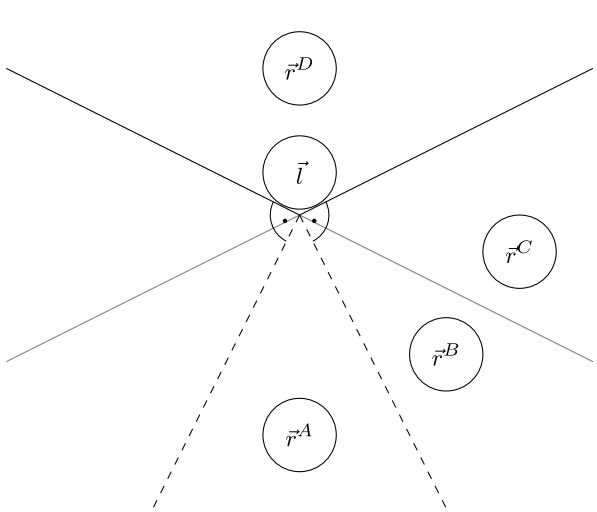
This results in a number of W_1 right-singular vectors $\mathbf{v}_{1,i}$ belonging to nonzero singular values of $(\Lambda_1\ N_0)^T$. Those are stored in N_1 which implicitly includes the vector space of N_0. Therefore, there are only two possible projection cases depending on W_1:

(1) $W_1 = 2$: $\mathbf{v}_{1,0} \times \mathbf{v}_{1,1}$ defines the direction vector of the straight line onto which the move vector has to be projected (see Fig. 5(d)).
(2) $W_1 = 3$: The end effector cannot move, the move vector equals $\mathbf{0}$.

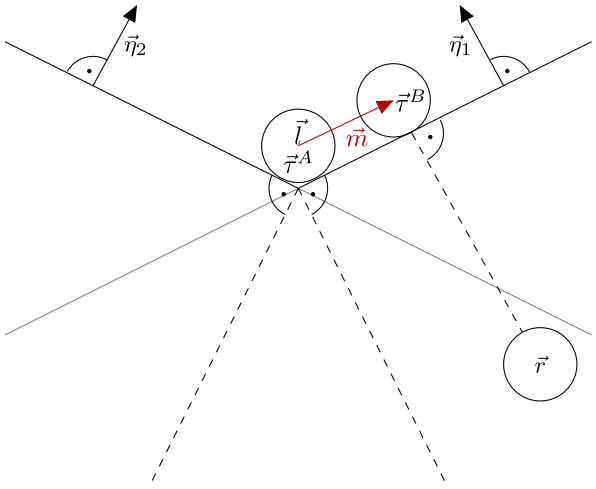
2.5.4. *Special projection cases*

The algorithm described in the previous section generates valid move vectors. However, there are certain cases where it is not sufficient to check projections defined by the right-singular vectors of the constraints if a natural interaction with the HHRD is desired.

Further valid projections can be found using the individual move plane normals. The example in Fig. 9 shows two planes defining a projection onto a line. It is obvious that certain home positions cause different projections. The movement $\mathbf{l} \rightarrow \vec{r}^B$ has to be handled differently than the movement $\mathbf{l} \rightarrow \vec{r}^A$ since the end effector would stick on the line even if the user movement



(a) All possibilities



(b) $\vec{\tau}^A$ is the projection of $\mathbf{r}$ onto the intersection line, $\vec{\tau}^B$ is the projection of $\mathbf{r}$ onto plane 1 which is valid and preferable

Fig. 9. Different projection cases. Two constraints are touched by the end effector (see (a)). The shown home positions $\vec{r}^A, \vec{r}^B, \vec{r}^C, \vec{r}^D$ result in different projections. $\mathbf{l} \rightarrow \vec{r}^D$ represents a movement away from the constraints, therefore, no projection is necessary. $\mathbf{l} \rightarrow \vec{r}^C$ intersects only with one of the two constraints and the movement is projected onto the plane defined by the constraint. $\mathbf{l} \rightarrow \vec{r}^A$ intersects both constraints, which is why it has to be projected onto the intersection line defined by the constraints. Even $\mathbf{l} \rightarrow \vec{r}^B$ causes a collision with both constraints but the desired movement can also be projected onto one of the two constraints and the resulting move vector does not collide with any of the two (see example in (b)). Therefore, it is important to also check the projections onto the individual constraints even when two different constraints are intersected.

indicates that he wants to slide the end effector along the constraint. Hence in this case, the better movement results from projecting onto one of the two planes instead of projecting onto the line.

As explained above, the number of right-singular vectors in N_k indicates how to project. In addition to the standard cases from the previous section (see Fig. 8), different movement projections are calculated and added to a list of possible projections:

(1) If W_k equals 2, the movement $\mathbf{l} \rightarrow \vec{r}$ has to be projected onto the individual constraints (the corresponding move plane normals are stored in Λ_k).
(2) If W_k equals 3, the movement has to be projected onto the individual constraints and all possible combinations of two move plane normals resulting in projections onto the corresponding lines.

Finally, the projection with the longest allowed movement is chosen.

2.6. *Acoustic signals to maintain responsiveness of the device*

In order to prevent the end effector from penetrating a constraint, the device deflects. Although the surgeon could realize this deflection, he normally concentrates on the intervention and not the deflection of the HHRD. Moreover, the maximum deflection is not directly apparent. Since the milling cutter has to be turned off when the maximum deflection is reached, it is important to inform the surgeon about approaching the maximum deflection in order to guarantee a smooth operation. This is realized by linearly reducing the speed of the milling cutter from 60,000 rpm to 50,000 rpm on the last 4 mm before the maximum deflection is reached. These speed changes are audible and the surgeon can react by repositioning his hand such that the milling speed again reaches 60,000 rpm.

2.7. *Table-top prototype*

In order to be able to test the presented algorithms, an *xyz*-linear stage was used since the actual HHRD was not available yet. The prototype (see Fig. 10) is composed of four linear stages driven by three motors and an orthopedic milling device. The linear stages are produced by CCM rails (www.ccmrails.com). The motors are intelligent NEMA23 motors by Technosoft Motion (www.technosoftmotion.com). The communication between the motors and the PC is realized via CAN. The CAN bus uses the maximal baud rate of 1 Mbps. The attached surgical milling device is produced by Stryker (www.stryker.com) and allows the surgeon to easily exchange the end effector. The linear stages have a pitch of 75 mm per revolution. The motors can run with up to 11.7 revolutions

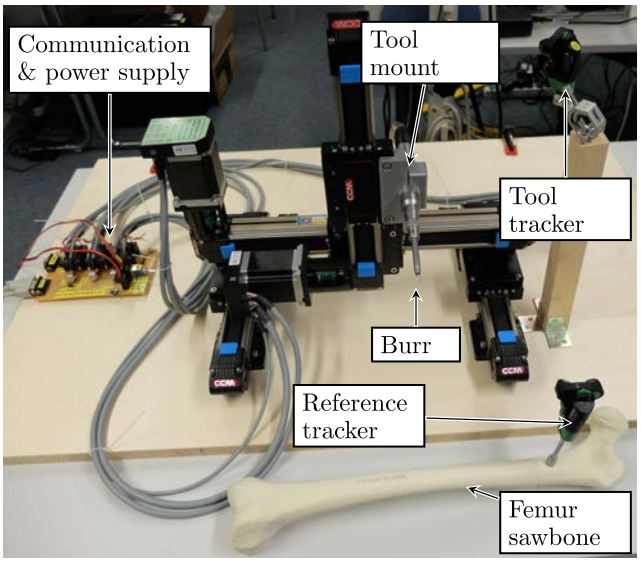


Fig. 10. A prototype consisting of four translation stages driven by three motors.

per second which allows speeds of up to 877 mm per second. Although the working volume of the table-top prototype is much bigger, it is restricted (by software) to a cube with an edge length of 25 mm which resembles the working volume of the target device.

The calibration of the translational table-top prototype consists of three steps:

(1) Homing the translation stages.
(2) Calibration of the transformation F_{CA} from tracker coordinates A to linear stages coordinates C.
(3) Pivoting of the end effector to find the home position $\vec{r}_A$.
(4) As a last step, the radius of the end effector has to be specified manually. In order to calibrate the radius (or the shape) of the end effector, an optical calibration, as presented by Klemm *et al.* [14], could be applied.

2.8. *Accuracy tests*

2.8.1. *Foam block tests*

Two different geometries, used in different surgical interventions, are milled and evaluated. The milling task is carried out by one experienced and two first-time users (engineers without surgical background). Each geometry is milled using the power-controlled (PCM), the evasive (EM), and the smoothing (SM) modes. The PCM simply turns off the tool's power and stops the end effector as soon as a constraint is touched (position control is turned off). In the evasive mode, the complete control system, as presented in this work, is active. In the smoothing mode, the same control system as in the evasive mode is running while the end effector oscillates

around the shaft. The geometries are loaded as CAD files in order to guarantee the reproducibility of test results. Polyurethane hard foam blocks (240 kg/m^3) are used as milling medium since their characteristics are comparable to those of bones. A burr with a diameter of 5 mm is used.

First, a cuboid of 30 mm × 30 mm × 15 mm is removed from the test block. Such a bone removal could be used to deepen the bone surface in order to place an implant. The cuboid's bottom plane is recorded, its regression plane is calculated and analyzed. Second, a half sphere with a radius of 15 mm is milled. Such a geometry could be used in hip arthroplasties (hip replacement surgery) to increase the diameter of the acetabulum in order to place an implant. For this test, a sphere with the given radius of 15 mm is matched onto the point cloud of the half sphere.

The milled geometries are sampled using the previously mentioned navigation system FP6000 by Stryker with the corresponding pointer device. The pointer tip can be localized with a standard deviation of 0.15 mm.

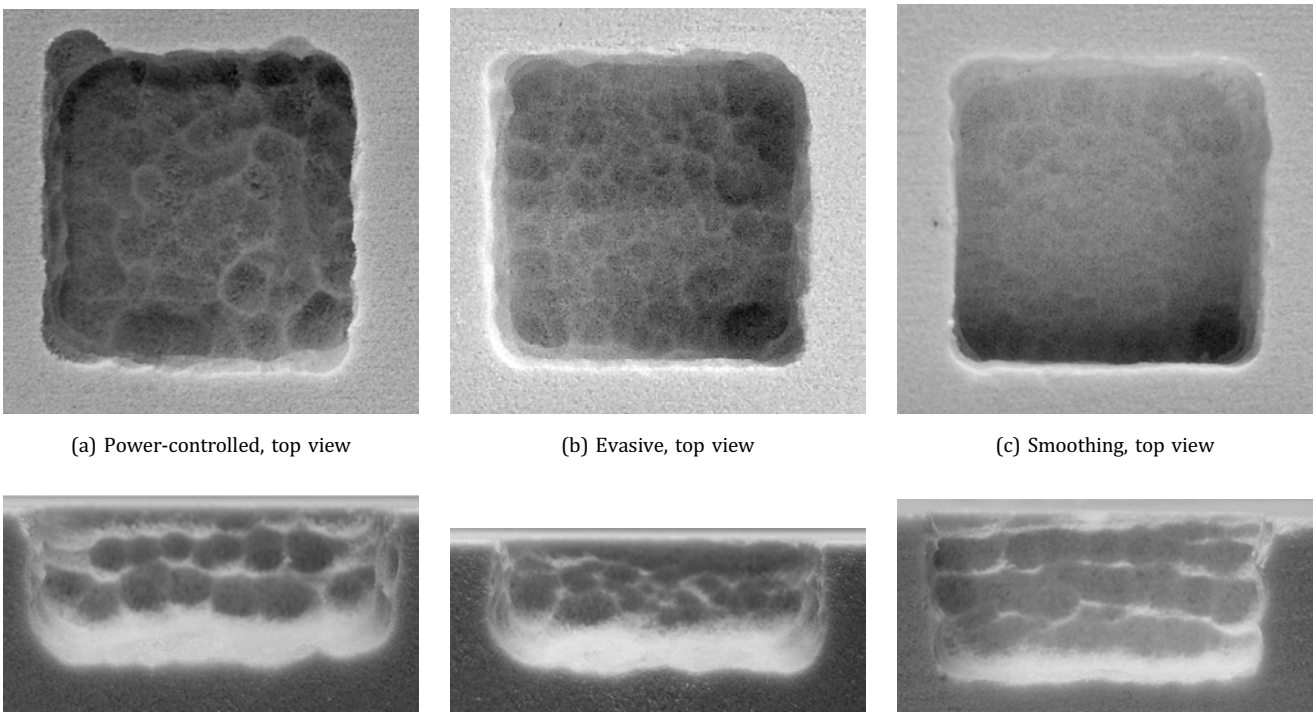
2.8.2. *Femoral neck osteotomy intervention*

The previous tests evaluate the system's accuracy. The goal of the following section is to evaluate the control system in collaboration with OrthoCAD. For this purpose, tests on bone imitations were performed. These imitations consist of hard foam comparable to the consistency of real bones. As in the previous tests, the table-top prototype with a 5 mm burr is used.

In order to show the functionality of the developed control system together with OrthoCAD, a femoral neck osteotomy is performed. This procedure represents a preparation for a hip arthroplasty (hip replacement) in which the femur's head is completely replaced by an implant. Before the implant can be inserted into the femur, the head and the neck have to be removed. This resection is performed by removing a layer with a thickness of 1 cm from the femur neck. This removal is necessary for gaining space to remove the femur head from the acetabulum. Since the femur head is disposed after surgery, only one tracker attached to the femur is required.

3. Results

Figure 11 shows the difference between a cuboid milled with the power-controlled, the evasive and the smoothing modes. While the power-controlled mode produced ragged edges and cratered surfaces, the evasive mode produced clearer edges and smoother surfaces. The smoothest surfaces are milled using the smoothing mode.



(a) Power-controlled, top view (b) Evasive, top view (c) Smoothing, top view

(d) Power-controlled, cross-sectional view (e) Evasive, cross-sectional view (f) Smoothing, cross-sectional view

Fig. 11. Cuboid milled with a 5 mm burr. (a) and (d) show the results using the power-controlled mode, (b) and (e) are milled in evasive mode, (c) and (f) correspond to the smoothing mode. Note that the evasive and smoothing mode produce smoother surfaces and cleaner edges.

Table 1. Deviations to regression plane for different trials and different modes.

Trial	Mode	Number of points	δ_{rms} [mm]	δ_{var} [mm^2]	δ_{max} [mm]	δ_{min} [mm]
1	PCM	2010	0.44	0.20	−1.31	1.70
1	EM	1411	0.36	0.13	−1.08	1.05
1	SM	1567	0.16	0.03	−0.62	0.61
2	PCM	1294	0.49	0.24	−1.92	1.42
2	EM	1596	0.41	0.17	−1.19	1.24
2	SM	2657	0.21	0.04	−0.80	0.64
3	PCM	2366	0.49	0.24	−1.31	1.59
3	EM	2640	0.39	0.15	−1.28	1.34
3	SM	2367	0.18	0.03	−0.54	0.70

Table 2. Deviations after sphere-fit for different users and different modes, radius was given (from CAD file), center was fitted.

Trial	Mode	Number of points	δ_{rms} [mm]	δ_{var} [mm^2]	δ_{max} [mm]	δ_{min} [mm]
1	PCM	2077	0.40	0.16	−1.35	1.59
1	EM	3444	0.29	0.08	−0.88	1.15
1	SM	941	0.32	0.11	−0.85	1.11
2	PCM	1778	0.54	0.29	−1.95	1.67
2	EM	1489	0.43	0.19	−1.24	1.49
2	SM	1648	0.37	0.14	−0.98	1.39
3	PCM	2455	0.60	0.36	−1.94	1.91
3	EM	2175	0.58	0.34	−1.35	1.96
3	SM	1466	0.51	0.26	−1.28	1.56

Tables 1 and 2 show the error measures for the cuboid and the half sphere tests. The corresponding boxplots are shown in Figs. 12 and 13. For each test, 1200 to 3500 points were digitized. The median of the boxplot is indicated by the central mark, the 25th and 75th percentiles are the edges of the box, the whiskers extend to the most extreme data points not considered as outliers and actual outliers are plotted individually as crosses. The results show that all three modes produce RMS errors of less than 0.6 mm. The deviations and their variances show that the smoothing mode produces the best results and that the evasive mode is more accurate than the power-controlled mode. The improvement is best visible in the minimal and the maximal deviations. In the cuboid tests, the smoothing mode produces much smoother surfaces than the evasive mode. However, in

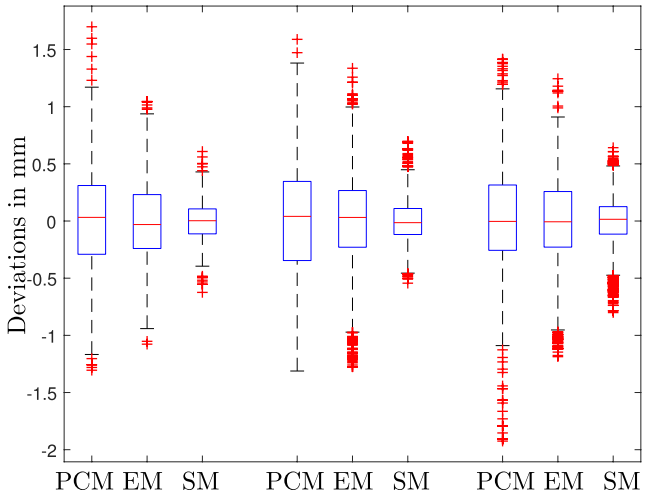


Fig. 12. Deviations to regression plane of different trials and different modes.

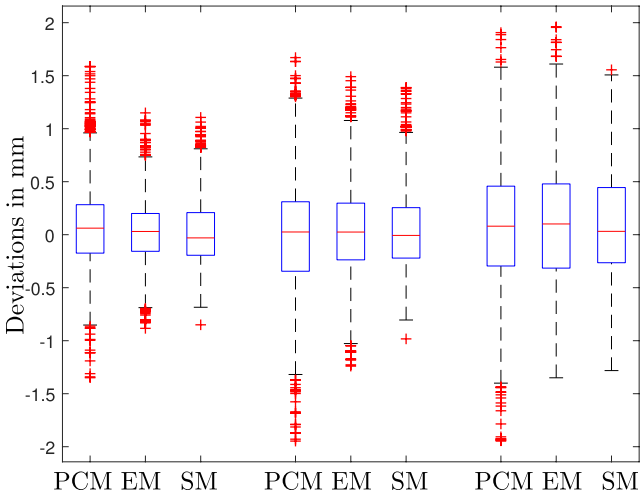
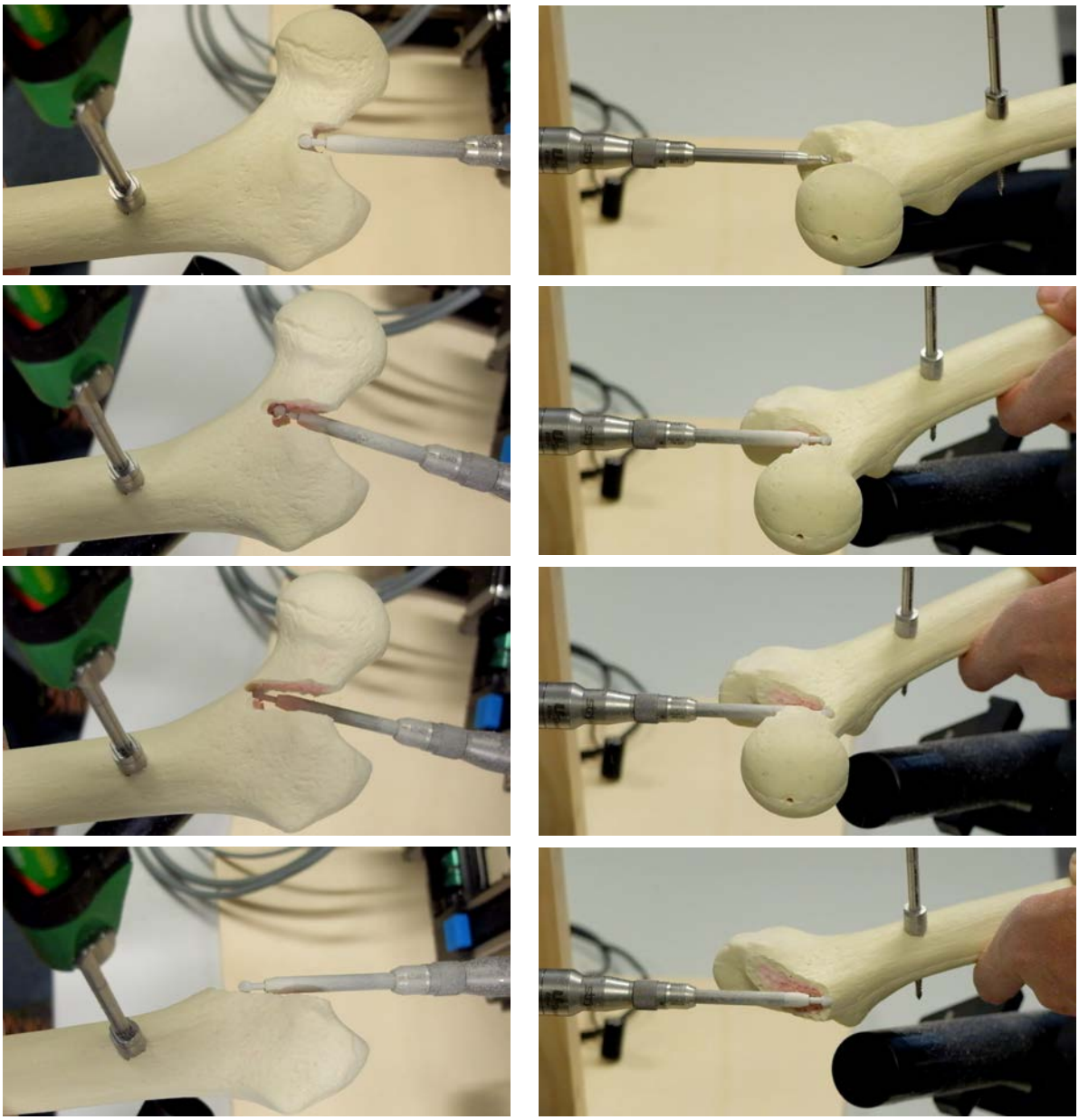


Fig. 13. Deviations after sphere-fit for different users and different modes, radius was given (from CAD file), center was fitted.



(a) First trial (b) Second trial

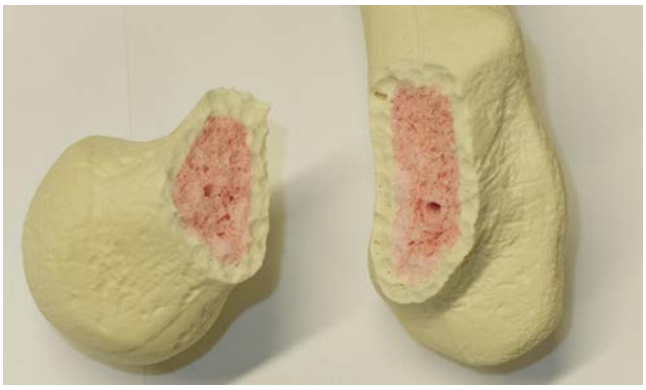
Fig. 14. Milling procedure of femoral neck osteotomy in several steps using the table-top robot prototype.

the half sphere tests, this difference is notably smaller. In one test, the RMS error using the smoothing mode is even worse than using the evasive mode. This is probably caused by the fact that the end effector oscillates around its shaft and that the shaft, in case of the cuboid tests, points perpendicular to the surface, whereas in case of the half sphere tests, it does not. Therefore, it can be concluded that the smoothing mode only produces smoother surfaces if the shaft points perpendicular to the surface.

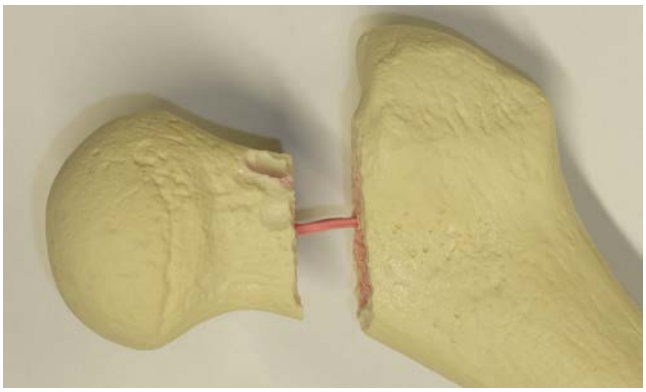
Figure 14 shows the milling procedure of the femoral neck osteotomy in several steps. This image sequence illustrates how the end effector evades from the defined constraints. Figure 15 shows the results of the intervention.

4. Discussion

This work presents a robot control system for HHRDs used in orthopedic surgery. The system protects previously defined regions from being penetrated by the HHRD. Different modes can be used depending on the hardware and the current task. A power-controlled mode simply turns off the device's speed when touching a



(a) First trial



(b) Second trial

Fig. 15. The results of the femoral neck osteotomy using the table-top robot prototype.

constraint, whereas the evasive mode adapts the end effector's position to evade the constraint. A smoothing mode, similar to the evasive mode, oscillates the end effector around its home position. Constraints can be arbitrarily shaped (convex, concave) and do not have to form a closed volume. A table-top robotic milling device is used as test platform. The accuracy analysis shows that the presented control algorithms work robustly and accurately and that the quality of the milled objects depends on the used mode. The power-controlled mode shows the biggest deviations, the smoothing mode produces the best results with smallest deviations. An exemplary femoral neck osteotomy illustrates the advantages and flexibility of the presented control system.

The experiments show that the RMS deviations are below 0.6 mm in evasive mode. The minimal and maximal deviations range from -1.4 mm to 2.0 mm. One source for this high range of deviations are the latencies of the prototype and the navigation camera. A future prototype will improve these latencies. The smoothing mode further decreased these deviations to a range of -0.8 mm to 0.7 mm. The results for the evasive mode are comparable to the ones presented by Brisson *et al.* [11] using a handheld device with retractable rotary blade. They state the minimal and maximal deviations with -1.7 mm and 1.2 mm. Xia *et al.* [6] use a cooperative stationary robot for skull base surgery. They perform accuracy tests on a cadaver and state penetration errors of 1–2 mm and maximal errors of up to 3 mm. They further state that previous tests on foam blocks produced better results.

A femoral neck osteotomy was planned in OrthoCAD and executed using the presented control system and the table-top robot. All tests were successful and demonstrated the perfect functioning of the whole system. For the described tests, the control system ran on a regular workstation (HP Z420, Intel Xeon quad-core with 3.6 GHz, 20 GB RAM) at approximately 1000 Hz. Hence, the presented algorithm does not require specialized hardware.

The surgeon holds the device in his hand and therefore places it in 6-DoF. The milling volumes are not just planar but also contain cavities and edges. Therefore, controlling the end effector's position just along one axis (1-DoF) is insufficient especially for milling large cavities with small entry points. Even a simple cylindrical hole with a diameter slightly bigger than that of the milling cutter gets demanding if the direction of the tool is not perfectly aligned with the hole. Therefore, HHRD with 3-DoF is much more preferable.

Tests have shown that the milling speed of the device has a strong influence on the accuracy in power-controlled mode: the higher the speed, the better the results. The presented tests were executed with a milling speed of 50,000 rpm. If the burr stops turning because of touching a constraint and subsequently starts again, it might happen that the burr gets caught in the bone. Therefore, low speeds may result in a movement of the whole tool whereas high speeds remove material.

A semi-automatic mode could further improve the usage and handling of such devices. The HHRD automatically removes material inside its workspace while the user simply holds the device close to the milled structures. As soon as the reachable material is completely removed, the user moves the HHRD to the next position. For such a semi-automatic mode, it is essential that the device can move in all three dimensions since moving along a line or a plane might be impractical.

Further research concerning the execution time of the milling tasks needs to be conducted in order to find out if the surgeon is faster using the evasive or smoothing mode.

Simulating a HHRD with a table-top robotic tool is sufficient for testing the algorithms, however, it is difficult for the operator to estimate the possible range of motion of the tool. In case of an HHRD being held by the operator, the range of motion is directly visible. In case of the table-top robotic tool, the theoretical range of motion

is much bigger and is only limited by software bounds. Nevertheless, several untrained users were straightforwardly able to mill complex bone volumes with the presented robot. Currently, the final HHRD prototype is in development and usability studies will be conducted as soon as it is available.

To the best of the authors' knowledge, the presented algorithms are novel, innovative and — once understood — straightforward to be implemented. They work robustly, accurately, and open up new opportunities for orthopedic interventions.

Acknowledgment

This work was funded by Stryker Leibinger GmbH & Co. KG, Freiburg, Germany.

References

1. P. Kazanzides, B. Mittelstadt, B. Musits, W. Bargar, J. Zuhars, B. Williamson, P. Cain and E. Carbone, An integrated system for cementless hip replacement, *IEEE Eng. Med. Biol. Mag.* **14**(3) (1995) 307–313.
2. J. E. Lang, S. Mannava, A. J. Floyd, M. S. Goddard, B. P. Smith, A. Mofidi, T. M. Seyler and R. H. Jinnah, Robotic systems in orthopaedic surgery, *Bone Joint J.* **93-B**(10) (2011) 1296–1299.
3. L. Rosenberg, Virtual fixtures: Perceptual tools for telerobotic manipulation, in *Proc. IEEE Virtual Reality Annual Int. Symp.* (IEEE, New York, 1993), pp. 76–82.
4. S. A. Bowyer, B. L. Davies and F. R. Y. Baena, Active constraints/virtual fixtures: A survey, *IEEE Trans. Robot.* **30**(1) (2014) 138–157.
5. S. S. Park, R. D. Howe and D. F. Torchiana, Virtual fixtures for robot-assisted minimally-invasive cardiac surgery, in *Proc. Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, (Springer, Berlin, 2001), pp. 1419–1420.
6. T. Xia, C. Baird, G. Jallo, K. Hayes, N. Nakajima, N. Hata and P. Kazanzides, An integrated system for planning, navigation and robotic assistance for skull base surgery, *Int. J. Med. Robot. Comput. Assist. Surg.* **4**(4) (2008) 321–330.
7. T. Haidegger, T. Xia and P. Kazanzides, Accuracy improvement of a neurosurgical robot system, in *Proc. 2nd Biennial IEEE/RAS-EMBS Int. Conf. Biomedical Robotics and Biomechatronics, BioRob 2008* (IEEE, New York, 2008) pp. 836–841.
8. A. Üneri, M. A. Balicki, J. Handa, P. Gehlbach, R. H. Taylor and I. Iordachita, New steady-hand eye robot with micro-force sensing for vitreoretinal surgery, in *2010 3rd IEEE RAS and EMBS Int. Conf. Biomedical Robotics and Biomechatronics (BioRob)* (IEEE, New York, 2010) pp. 814–819.
9. B. C. Becker, S. Voros, R. A. Maclachlan, G. D. Hager and C. N. Riviere, Active guidance of a handheld micromanipulator using visual servoing, in *IEEE Int. Conf. Robotics and Automation (ICRA)* (IEEE, New York, 2009), pp. 339–344.
10. M. Kneissler, A. Hein, M. Mätzig, U. W. Thomale, T. C. Lueth and C. Woiciechowsky, Concept and clinical evaluation of navigated control in spine surgery, in *IEEE/ASME Int. Conf. Advanced Intelligent Mechatronics, AIM* (IEEE, New York, 2003) pp. 1084–1089.
11. G. Brisson, T. Kanade, A. Digioia and B. Jaramaz, Precision freehand sculpting of bone, in *Medical Image Computing and Computer-Assisted Intervention* (Springer, Berlin, 2004), pp. 105–112.
12. J. H. Lonner, Robotically assisted unicompartmental knee arthroplasty with a handheld image-free sculpting tool, *Orthop. Clin. North Am.* **47**(1) (2016) 29–40.
13. C. N. Riviere, W. T. Ang and P. K. Khosla, Toward active tremor canceling in handheld microsurgical instruments, *IEEE Trans. Robot. Autom.* **19**(5) (2003) 793–800.
14. M. Klemm, F. Seebacher and H. Hoppe, Flexible three-dimensional camera-based reconstruction and calibration of tracked instruments, in *2016 19th Int. Conf. Information Fusion (FU-SION)* (IEEE, New York, 2016), pp. 861–867.

Martin Klemm received his M.Sc. degree in Electrical Engineering and Information Technology from Offenburg University, Germany, in 2013. Following, he received his Ph.D. in Informatics at the Karlsruhe Institute of Technology (KIT), Germany, in 2018. His research interests are in the areas of robot control, computer vision, HMD calibration and augmented reality.

Harald Hoppe received his diploma in physics (1999) and a Ph.D. degree (2003) in Computer Science from the Karlsruhe Institute of Technology (KIT), Germany. He is a full professor in medical informatics at Offenburg University, Department of Medical Engineering and the scientific leader of the Laboratory for Computer-Assisted Medicine. His major research interests include navigated surgical interventions and augmented reality applications for medical interventions with focus on precice calibration methods for cameras, handheld robotic devices, see-through glasses, and navigated surgical instruments.

Uwe D. Hanebeck is a Chaired Professor of Computer Science at the Karlsruhe Institute of Technology (KIT) in Germany and director of the Intelligent Sensor-Actuator-Systems Laboratory (ISAS). He obtained his Ph.D. degree in 1997 and his habilitation degree in 2003, both in Electrical Engineering from the Technical University in Munich, Germany. His research interests are in the areas of information fusion, nonlinear state estimation, stochastic modeling, system identification, and control with a strong emphasis on theory-driven approaches based on stochastic system theory and uncertainty models. He is the author and coauthor of more than 400 publications in various high-ranking journals and conferences and an IEEE Fellow.