Int J CARS
DOI 10.1007/s11548-015-1318-7

ORIGINAL ARTICLE

Time-of-flight-assisted Kinect camera-based people detection for intuitive human robot cooperation in the surgical operating room

Tim Beyl[1] · Philip Nicolai[1] · Mirko D. Comparetti[2] · Jörg Raczkowsky[1] · Elena De Momi[2] · Heinz Wörn[1]

Received: 29 April 2015 / Accepted: 21 October 2015

Abstract

Background Scene supervision is a major tool to make medical robots safer and more intuitive. The paper shows an approach to efficiently use 3D cameras within the surgical operating room to enable for safe human robot interaction and action perception. Additionally the presented approach aims to make 3D camera-based scene supervision more reliable and accurate.

Methods A camera system composed of multiple Kinect and time-of-flight cameras has been designed, implemented and calibrated. Calibration and object detection as well as people tracking methods have been designed and evaluated.

Results The camera system shows a good registration accuracy of 0.05 m. The tracking of humans is reliable and accurate and has been evaluated in an experimental setup using operating clothing. The robot detection shows an error of around 0.04 m.

Conclusions The robustness and accuracy of the approach allow for an integration into modern operating room. The data output can be used directly for situation and workflow detection as well as collision avoidance.

Keywords Digital operating room · Environment supervision · Surgical robotics · 3D vision · RGB-D cameras · ToF cameras

✉ Tim Beyl
tim.beyl@googlemail.com

[1] Institute for Anthropomatics and Robotics (IAR), Intelligent Process Control and Robotics (IPR), Karlsruhe Institute of Technology, Karlsruhe, Germany

[2] NeuroEngineering and Medical Robotics Laboratory, Department of Electronics, Information and Bioengineering, Politecnico di Milano, Milan, Italy

Introduction

In the past years medical robotics has evolved to a point where not only teleoperated systems are considered to improve the medical workflow. Current robotics research focuses more on assistant systems, robots in close cooperation with the personnel or semi-autonomous systems. Those systems have in common that the usage has to be as intuitive as possible. A future goal in research should be to have robots that can be intuitively controlled and that do not require direct attention. More specifically, the system has to behave like the surgeon expects it to, yet to ensure a safe usage of the system especially in the case when conventional multi-purpose (e.g., industrial) robots are being used and adapted to the scenario.

With robots specially designed for surgery, some safety requirements can be fulfilled in the design phase of the kinematics. An example is the remote center of motion for minimally invasive robot surgery that can be solved in hardware with specially designed robots. For example, the DaVinci system (Intuitive Surgical, USA) only moves around a given remote center of motion.

Surgical systems using multi-purpose robots have to cope with issues like safety or remote center of motion in software, e.g., using special supervision systems. A common advantage of systems such as the DLR Miro [15], the LARS robot [4] and some research systems using the KUKA LWR 4 (KUKA, Germany) [3] is the huge and flexible workspace and a higher applicable payload of the robots that allows usage in multiple applications. This offers new applications like ultrasound (US) probe steering, open surgery with increased accuracy and a suitable workspace, orthopedics interventions using milling devices and saws, as well as the field of minimally invasive robotic surgery that is widely explored, e.g., by Intuitive Surgical [11] and the University of Washington [12].

Published online: 14 November 2015

Our approach aims at raising the safety of multi-purpose robots inside of the operating room. Moreover it offers the base for a comprehensive operating room supervision that can also be used with specially designed surgical robots in order to improve the intuitiveness and the efficiency of the system. Use cases of the system range from simple high-speed collision avoidance and on-line path planning based on point clouds, workflow detection focused on the operating room environment to probabilistic and rule-based inference from the perceived situation and workflow-based control of the operating room and the surgical robots. These concepts can, for example, be used in minimally invasive laparoscopic surgery. The transitions between hands-on movement toward the patient, telemanipulation and hands-on movement or automatic movement away from the patient can be detected.

To the best of the authors knowledge there is no publication about similar approaches yet. However, the problem of multi-depth camera room supervision for other applications is tackled by [16] who developed a multi-Kinect system for interaction with the environment and a past self, by [28] who are focusing on the "The room is the computer" and the "Body as display" approach as well as [8] who are tackling the problem of interference induced by using multiple structured light Primesense (Primesense, Tel-Aviv, Israel) devices. Other approaches like [17] and [29] focus on multi-Kinect people tracking and dynamic scene reconstruction from asynchronous depth cameras.

From a technical point of view, our system is composed of three different camera systems. Seven time-of-flight (ToF) cameras (six PMD[vision] S3 and one PMD[vision] CamCube 2.0) are dedicated to low-latency scene supervision. These cameras are used for collision avoidance. Four Kinect cameras (Microsoft, USA) that offer larger resolution, but have the drawback of introducing more latency into the processing chain, are used for semantic interpretation of the scene. The marker-based tracking system ARTtrack2 (ART, Germany) is used for calibration purposes and accurate tracking of marker equipped objects. Our supervision system is developed in the context of OP:Sense [18,21], which is mainly dedicated to safe and intuitive workflow controlled human robot interaction in the operating room [2] as well as exploring new applications for the operation. OP:Sense is based on ROS [24].

In the following we describe our approach to processing the high amount of data coming from the Kinect cameras, dealing with registration issues between the cameras and our approach to find corresponding humans in the camera's field of view. We outline how we integrated the Kinect system with the combined Photonic Mixer Devices and marker-based optical tracking system. Finally, our method for detecting the location of robots in the scene is described as well as a CUDA [10]-based distance calculation for point clouds.

Materials and methods

The OP:Sense system is composed of two KUKA lightweight robots LWR4 (KUKA, Augsburg, Germany) that are the central part of our robotic system, a ceiling-mounted camera rig holding an optical tracking system ARTtrack2 (Advanced Realtime Tracking GmbH, Weilheim, Germany), PMDtec cameras (PMD technologies, Siegen, Germany) as well as four Kinect cameras, custom-built attachable surgical instruments, milling devices, an ultrasound device, a high-precision Stäubli (Stäubli International AG, Freienbach, Switzerland) RX90 robot, the special endoscope steering robot Viky (EndoControl, Grenoble, Frankreich) and an endoscope. The software framework is based on ROS. On top of that, a custom-built framework based on a java implementation for ROS is being developed in order to cope with probabilistic, rule- and workflow-based control of the complete system.

As described in [2] and [27], which are expanded through this work, a Kinect camera acquires 3-channel RGB images with a resolution of 8 bit per channel (dimensions: 640×480) as well as a 11 Bit depth map (dimensions: 640×480). The capturing frequency is 30 Hz, which results in a data rate of around 40 MB/s to be transferred between the Kinect and its host computer via USB. This requires the use of a dedicated USB 2.0 host controller per Kinect which means that multiple Kinects cannot share the same USB bus. In order to build up a scalable supervision system with a flexible number of cameras, we separated the capturing and processing of Kinect data. Capturing is performed using small form-factor PCs (Zotac ZBOX nano AD10) with an AMD E-350 CPU, to which two Kinects are connected each. Due to the small footprint of these PCs and their (spatial) separation from the processing server, no unnecessary clutter is introduced close to the region of interest, e.g., the OR table, and we gain a high flexibility in the number and position of the Kinects. The proposed algorithm for processing the acquired data runs online with up to four cameras using an Intel Core i7 3770 and 8 GB of RAM.

The Kinect cameras and the processing chain introduce a notable time delay into the system. In order to cope with fast movements and time critical tasks such as collision detection and avoidance, the Kinect system is complemented by ToF cameras that observe the same workspace as the Kinects and feature a higher. The ToF cameras provide a low dense uncolored point cloud with low delay. For precise 6 DOF tracking of small objects such as surgical instrument, they can be equipped with optical markers and tracked via the ART track system.

The PMD subsystem consists of six ceiling-mounted PMD S3 cameras that surround the OR table. They provide a depth map as well as an amplitude image with a resolution of 64 pixels $\times$ 48 pixels. The value of each pixel in the depth map

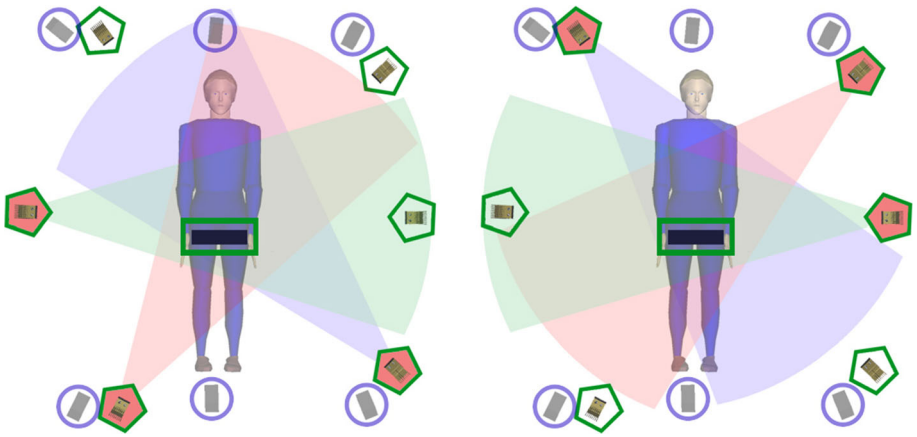

Fig. 1 Triggering scheme of PMD cameras (*colors* represent different modulation frequencies); *left* camera group one; *right* camera group two

provides the distance to the corresponding 3D point in the scene, while the pixel value in the amplitude map gives the intensity of the reflected IR light. Located centrally above the OR table, a PMD CamCube provides depth map, amplitude image and a grayscale image with a resolution of 204 pixels $\times$ 204 pixels. The PMD S3 cameras, which can be controlled and accessed via Ethernet, are connected via a dedicated subnetwork using a 1 GBit/s switch.

Due to the measurement principle of ToF cameras, they are very prone to crosstalk when multiple cameras are used in a shared environment. While other works such as [5] focus on employing this for further calculations, we chose to eliminate crosstalk by implementing a time- and frequency-multiplexing synchronization scheme that prevents the PMD cameras (Low Resolution ToF) from influencing each other. The six PMD S3 cameras, which are ceiling-mounted around the operating table, are divided into two groups. Inside each group, the cameras use different modulation frequencies which are set when the system is started. Both camera groups are triggered alternatively to each other with the PMD CamCube (high-resolution ToF) triggered in between. Figure 1 shows the triggering scheme of the two camera groups.

The coordinate frames of the independent camera systems are registered to each other to acquire a multi-modal scene representation. The layout of the supervision system can be seen in Fig. 2. A close to reality setup with two LWR4 robots attached to an OR table is depicted in Fig. 3. The network setup for the multicamera system can be seen in Fig. 4.

The proposed approach was implemented using the OpenNI framework for full body tracking [20,23] and ROS. Generally, it can be used with any people tracking algorithm that delivers depth maps. The number of used Kinect cameras was set to four and can be parameterized at the start up of the system. In the following paragraphs, the detection and correspondence matching as well as distance computation works are described.

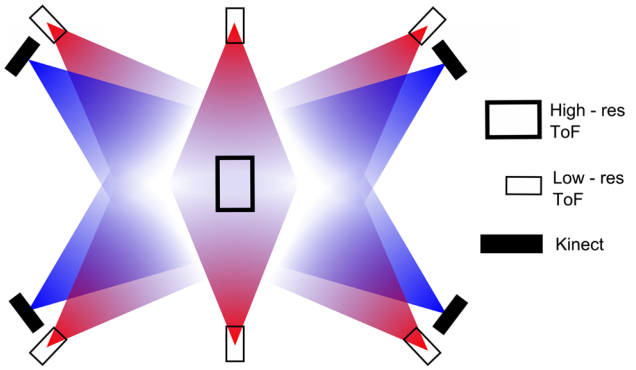

Fig. 2 Layout of the camera rig from a bird's view. High-resolution (204 $\times$ 204 pixels) ToF is pointing downward and all other cameras are pointing toward the floor with an angle between 30° and 45° compared to the floor

Registration of Kinect cameras with respect to a reference frame

One of the Kinect cameras is (arbitrarily) chosen as the reference camera K_{ref}. We perform a pairwise registration between K_{ref} and each other camera (K_n). A scheme showing the calibration is given in Fig. 5.

A laser-printed checkerboard with 8 by 5 fields (8 cm $\times$ 8 cm edge length) on DIN A3 standard paper is used to provide a large area in which to detect the corners. The printout is attached to a wooden flat panel by adhesive in order to provide a completely undistorted checkerboard.

The following steps are performed for all pairs of (K_{ref} and K_n $[n = 2...4]$), using always the same camera as reference:

1. Detection of the checkerboard using the RGB camera (K_{ref} and K_n): the OpenCV [6] checkerboard detection algorithm is used to detect the corners of a checkerboard. Resulting coordinates of the corners are stored locally for every RGB image.

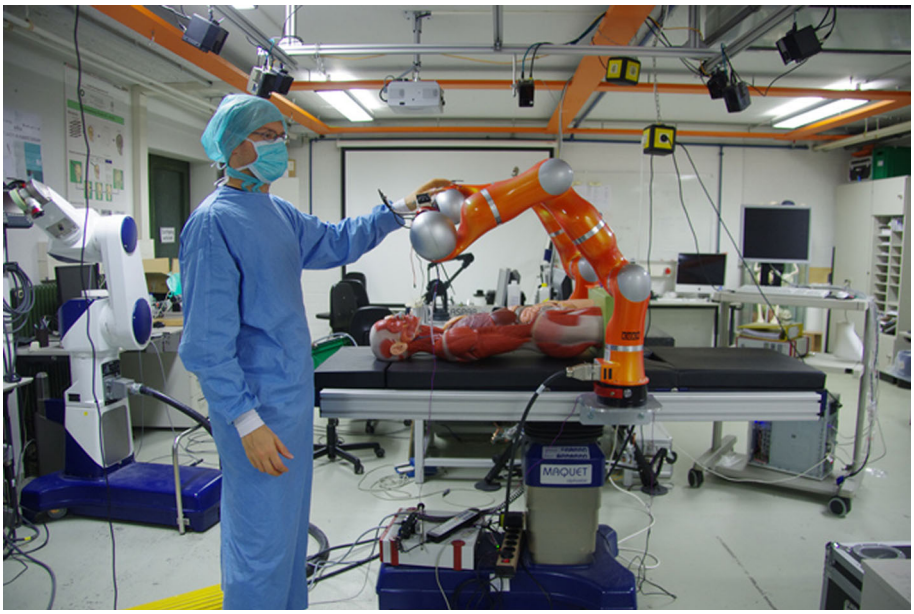

Fig. 3 Setup in the IPR lab showing the surgeon, the operating table and the minimally invasive robotics setup [2]

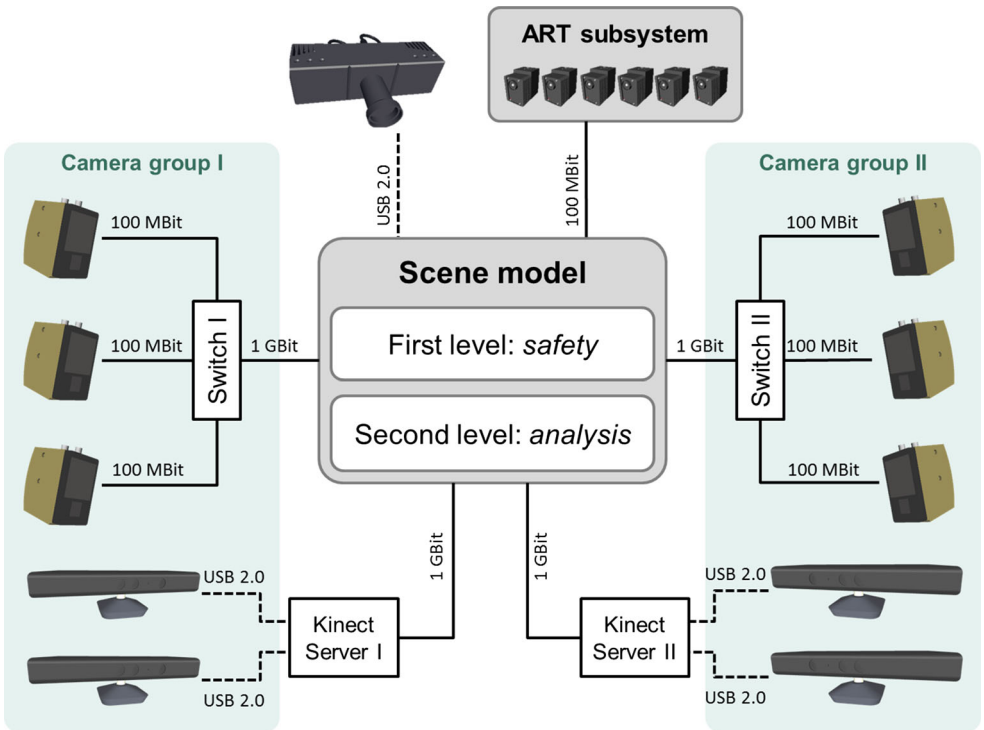

Fig. 4 Ethernet setup for Kinect, ToF (PMD) and optical tracking system cameras (ART) [2]

2. Calculation of the corresponding depth value for every checkerboard corner (K_{ref} and K_n): Based on the known relation between depth and rgb sensor in each Kinect, an organized point cloud is created for each camera. An intrinsic and extrinsic calibration of cameras of the Kinect cameras has not been performed, as the registration errors have been found to be already sufficiently low enough for the application. The 2D coordinates of the checkerboard corners detected in the previous step can now be used to directly look up the corresponding 3D point in the 2D-indexed point cloud. However, it is very likely that due to lighting conditions or reflection of the material, not all 2D points correspond to a valid depth measurement and/or that the noise level is high. To encounter the noise problems, we acquire the depth data at every checkerboard corner for 300 subsequent frames and cal-

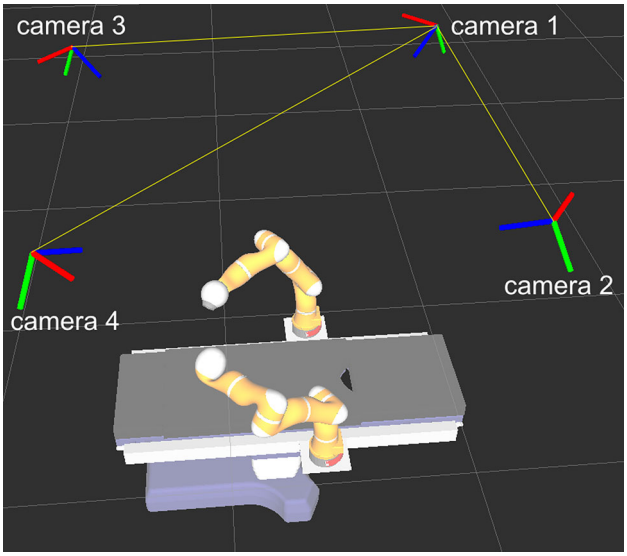

Fig. 5 Location of Kinect cameras over the operating table. Every Kinect camera is calibrated with reference to camera 1

culate the average of all valid depth measurements for each point. Compared to conventional calibration methods, where the checkerboard pose is computed from the known geometry of the board, the depth error of the Kinect cameras [14] is taken into account by using the measured depth for the calibration.

3. Correspondences between the source and the target point cloud are stored: after collecting the depth data for every checkerboard corner, we evaluate how many checkerboard corners with valid depth information (3D checkerboard corner) have successfully been detected. If a 3D checkerboard corner has been acquired by both K_{ref} and K_n, the respective data are added to a correspondence list. Otherwise, e.g., if one depth value within a correspondence (K_{ref} or K_n) is missing, the complete correspondence is dropped. This leads to a set of correspondences that is later used as an input for the transformation estimation.
4. Repositioning of the checkerboard: The checkerboard has to be placed in different poses in the field of view (fov) of the Kinect cameras in order to estimate further correspondences, make the transformation estimation more robust to outliers and increase the calibrated workspace. The implementation supports unlimited checkerboard positions. The registration results benefit if the planes defined through the checkerboard positions are not parallel to each other (multiple orientations). After the re-placement, steps 1–3 are repeated until enough correspondences have been collected. In our experiments we registered the cameras using 12 poses in a volume of approximately 1.8 m × 1.0 m × 1.5 m. Twelve poses were experimentally found to lead to a reproducible registration result as specified in "Kinect camera system" section.
5. Estimation of the transformation: the correspondences are used to estimate a transformation between the camera frames using the point cloud library (PCL) [25]. The estimation is based on incrementally building a covariance list of the correspondences and the means of the correspondences. The rotation is estimated using eigenvalue decomposition of the covariance. The translational component is estimated using the means of the point sets, in a similar way to a standard ICP [1] iteration.

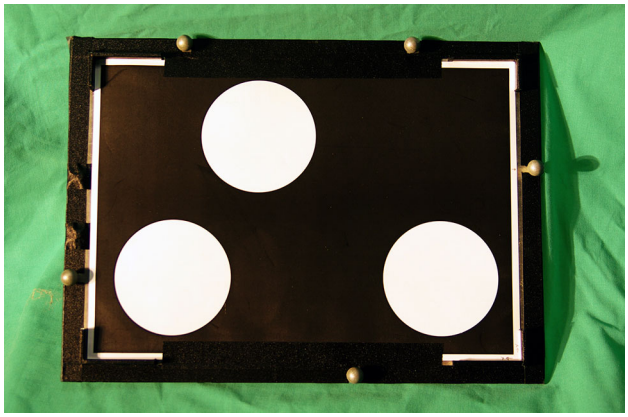

Fig. 6 Registration object for registration of PMD cameras against optical tracking system (ART)

Registration of depth cameras with respect to an optical tracking system (OTS)

As the OTS used in our setup features a big working volume and a precise localization of rigid bodies, it is desirable to also register depth cameras to the OTS. Apart from allowing for fusion of data obtained by both OTS and depth cameras, registering depth cameras with reference to the OTS eases the registration process for cameras with small overlapping fields of view. Two different registration objects can be used: A custom one especially designed for (low resolution) PMD cameras or a checkerboard. The custom registration target features large white circles on a black background (see Fig. 6), which can be easily and precisely located by the PMD cameras, and retroreflective marker spheres that are tracked by OTS.

In order to collect corresponding 3D points between OTS and depth cameras, the following steps are performed for each camera:

1. Detection of the registration object by depth camera: The features of the registration object, e.g., center points of the circles on the custom registration target or checkerboard corners, are automatically located by the depth cameras. In case of PMD S3 cameras, feature detection is per-

formed based on the amplitude image, and in case of PMD CamCube, feature detection is performed based on the intensity image. For RGB-D cameras, the RGB image is used. As in the previous section, feature detection yields the 2D coordinate of a detected feature in camera space.
2. The corresponding distance value for each feature detected is calculated: We average the depth image over 10 consecutive frames in order to smooth out noisy data. Based on the location of the features in the amplitude image, the corresponding distance value is acquired via a lookup in the averaged depth map. If a checkerboard or similar is used instead of the custom registration target, the detected features (e.g., checkerboard corners) are located at positions neighboring both white and black regions. As this can introduce incorrect distance readings by PMD cameras, the plane of the checkerboard is determined from the averaged depth image using a constrained RANSAC [9]. At each 2D position indicated by feature detection above, the corresponding depth value is retrieved and projected onto the plane. The resulting noise-free 3D feature position is then stored.
3. Detection of the registration object by OTS: The custom registration target with attached optical markers is already calibrated against the OTS and can therefore be tracked in 6D. Based on the known properties of the registration pattern, the 3D position of features (w.r.t. the OTS), corresponding to those detected by depth cameras, can be directly calculated. For registration objects that are not equipped with optical markers, features can be manually annotated using a special rigid body ("pointer") whose tip has been calibrated to the OTS. Each feature's position is acquired by positioning the tip of the pointer at the feature location.
4. Correspondences between features located by depth camera and OTS are stored.
5. Repositioning of the registration object: As in the pairwise registration of Kinect cameras, the steps above are repeated 12 times with different locations of the registration object. Due to the narrow field of view of the ToF cameras and the resulting, rather small volume visible to both ToF camera and OTS, this number of iterations has been determined as the "sweet spot" where the volume is completely sampled in the given system setup.
6. Estimation of the transformation: Based on the stored correspondences, the transformation between depth camera and OTS is calculated using either the method of [13] or the method provided by PCL (again, see pairwise Kinect registration for details).

After performing the registration for each PMD camera, their known transformations w.r.t. the OTS are used to establish the registration between the PMD cameras themselves.

Registration of the reference Kinect camera with respect to the OTS

The Kinect camera subsystem is designed to work independently from any reference system. However, both PMD and Kinect camera system need to be registered to each other in order to enable a fusion of their data. The registration of the reference Kinect camera is performed using the same method as the registration of the PMD cameras. Due to the lower resolution of the PMD cameras compared to the Kinect cameras, registration objects with larger patterns and therefore less details have to be used for the PMD camera system registration.

Detection of a reference plane within the reference camera's frame

The correspondence detection for users in different cameras and the robot base detection both depend on knowledge about the plane equation that describes the floor of the room the system is used in. The floor is needed for the detection of the robot at the OR table and to project the centroids on. A common method to detect a plane is to use a RANSAC algorithm in combination with a plane model. We use the RANSAC plane estimator that is part of PCL. RANSAC delivers a plane equation of the dominant plane inside a point cloud. The plane equation is stored for future processing. This is implemented as an interactive process to allow the user to verify the registration result at any time in order to assure that the correct plane has been found.

Robot localization

Contrary to, e.g., a typical industrial application, the position of a robot in the clinical scenario is highly likely to not remain constant. For example, even in the same type of intervention it might be placed at different positions along the operating table rail (e.g., based on the patients anatomy or the indication) to provide optimal workspace in the situs. In other types of intervention such as in the ACTIVE scenario for neurological interventions, the robots are not mounted to the OR table at all. For this reason, the robot's position has to be determined before the system can be used for human–robot interaction. We have defined two different approaches for localization of the robot's position in a 3D scene acquired from several registered depth cameras: A passive localization based on landmarks and an active localization throughout which the robot is moving. In the following, a short explanation of the localization methods is given. For more information we refer the reader to [19]

Both methods perform the following steps which can be applied to the point cloud from a single depth cameras or a full 3D scene acquired by multiple registered cameras:

1. Approximate detection of the robot's base: for passive localization, a landmark like the operating table is detected first. In the reduced search space, the robot's base is determined with an appropriate method, e.g., by localizing it along the operating table rail using circle fitting on horizontal slices of the 3D scene. For active localization, the robot performs a predefined motion. Via spatial change detection, the area of motion is detected in the 3D scene. Based on the known geometric properties of the robot and the performed motion, the base of the robot can be estimated.
2. Localization refinement: a simplified model of the robot is placed in the 3D scene at the previously determined position of the robot. The scene is then segmented into (a) points inside the robot hull ("inliers"), (b) points within a certain distance to the robot's hull ("outliers") and (c) points farther away from the robot. Using a metric based on this segmentation (ratio of inliers to outliers, combined with total numbers of inliers), the accuracy of the currently estimated position is rated. The scene segmentation and subsequent rating of the current position estimate are repeated for multiple robot positions. These positions are determined using the following algorithm: First, a local optimization is performed. Predefined positions within a certain distance to the current position are rated and the best rated position is used as the basis for the next steps. Then, for each segment of the robot, an optimization vector is calculated based on the center of the segment and the center of the potential outliers. The robot's position is then transformed by an average of the corresponding optimization vectors, yielding a new position at which local optimization is performed.

Human detection and tracking

Human detection using OpenNI

OpenNI and NITE (PrimeSense, ISRAEL) are frameworks for the development of 3D sensing technologies using the Primesense sensor that is built into the Kinect camera. For a single Kinect camera it offers user detection and full body tracking without the requirement of a calibration pose. We use OpenNI for user detection in every single Kinect camera. Per camera, up to 16 users can be detected. For each user, we extract the corresponding depth values and add them to an empty depth map that is used for further processing. The result is a set of h depth maps per camera where h is the number of users detected in the camera frame. Every depth map holds the depth values corresponding to a single detected user. This detection step is performed in real time, e.g., repeated for each frame at 20–30 frames per second, and triggers the execution of the complete processing chain.

Removal of noise pixels in depth image

A common problem of depth sensors is noise at edges present inside an image. Using ToF cameras, the term "jumping pixels" is commonly used for pixel that changes position from foreground to background and vice versa. We eliminate the resulting noise by first performing Sobel-based edge detection on the depth map acquired by each ToF camera and then filtering all "edge pixels" from the acquired point clouds. Kinect cameras do not show the same phenomenom but exhibit a behavior where several pixels at borders lie between foreground and background. This noise makes it hard to perform a robust brute force distance computation and affects the computation of centroids. Again, we use two steps of noise removal process. In the first step the morphological erosion filter is applied to each depth image holding a user. As filter a rectangular kernel shape with size 2 has been chosen. This removes pixels at the borders of the users in these images, thereby eliminating most of the noise. Step two is described in paragraph 2.6.4.

Point cloud computation for every detected user

Using the known focal length of the Primesense sensor's lens, the resolution of the sensor and the function that maps depth maps to meters, one can compute a point cloud from every depth map.

Denoising in point clouds

The resulting point cloud is still not completely free of noise. In particular artifacts that occur during movements cannot be completely removed by the erosion operation. In order to cope with the remaining noise, statistical outlier removal is being performed. The used algorithm is described in [25] and uses k neighbors of a point to find inliers and outliers. Points that are closer to the mean μ of the k neighbors locations of a point in the point cloud than α (standard deviation multiplier) * σ (standard deviation) of the k neighbors locations are considered as inliers. The other points are considered as outliers and are removed from the point set. We set α to 1.0 and we used 50 nearest neighbors to be considered for the filtering. The result is a point cloud representing a single user in a single camera with almost no visible noise at the borders. This provides a good base for centroid computation, sensor fusion and distance computations.

Find correspondences based on centroids

In order to determine corresponding users, the centroid of the point cloud representing the complete user is computed. The centroid c of a point cloud can be computed by summing up the translational components of every point in a cloud.

Dividing the resulting vector through the number of points gives the centroid of a point cloud as shown in Eq. (1)

$$c = \begin{bmatrix} \frac{p_{1,x}+p_{2,x}+\cdots+p_{n,x}}{n} \\ \frac{p_{1,y}+p_{2,y}+\cdots+p_{n,y}}{n} \\ \frac{p_{1,z}+p_{2,z}+\cdots+p_{n,z}}{n} \end{bmatrix} \quad (1)$$

where p_i is a point in the point set and n is the number of points.

In our approach, the centroids are used for the fusion of user point clouds, as they are usually placed in areas of the point clouds that represent the thoracal or abdominal regions of the human and can therefore characterize a human's location. Alternatively thorax objects as computed from skeleton tracking approaches could serve this purpose. However, the shown approach purely relies on point clouds and the corresponding centroids for the fusion of user representations, as skeleton tracking can be unstable or less available compared to the point clouds.

To establish a shared reference frame, all user point clouds as captured by the cameras (configuration can be seen in Fig. 5) are transformed into the reference frame of camera K_{ref}. Afterward, corresponding users are determined based on distance between every computed centroid. Please note that, if not explicitly stated otherwise, we use the Euclidean metric for all distance calculations throughout this article. If the distance of the centroids of two user point clouds is below a certain threshold, both point clouds are classified as belonging to the same user. We compute a matrix holding the correspondences (in the following: correspondence matrix) that is used for concatenation of the point clouds. However, the centroid is a metric that is not very robust to movements of the user. Bending of the user's upper body or articulation of a joint causes the centroid to shift. As each camera covers a different part of the scene due to its position and field of view, partial occlusions are possible and only parts of a user may be visible to a single camera. The movement of a centroid in a single camera may result in a distance between corresponding centroids that is above the defined threshold which results in false-negative detection of corresponding centroids. A possible solution is to increase the threshold for correspondence detection. However, this results in a higher false-positive detection rate. Instead, we tackle the problem using the fact that the erosion operation from the removal of noisy points does not only remove noise but also points corresponding to a user. The impact on image parts representing limbs is higher compared to the impact on abdominal and thoracic regions. The reason for this is that in a 2D view (depth image) of the users a longer contour is present for limbs compared to the contour of upper body parts. The erosion operation removes pixels along every contour and therefore strongly affects the limbs because of the lower volume to surface ratio of the limbs compared to thoracic and abdominal regions.

The reduction in limb points has the effect of moving the centroid more into the center of the body of the detected user as the limbs compared to thoracic of abdominal regions are farther away from the center of body. The overall number of points inside thorax and abdomen exceeds the number of the points of the limbs which additionally helps to pull the centroid toward the center of a user. However, the problem of the movement of the centroid through bending of the upper body is not solved. We reduce the problem to a two-dimensional one by projecting the centroids to the floor plane which was detected in step 2.4. By doing this we now have to consider the trace of the user on the floor plane instead of the trace of the centroid in 3D space. This eliminates errors along the longitudinal axis of the user and helps for cases where only parts of a user are visible in a single camera image. The computation of the distance between the centroids is performed on these projected centroids and the final correspondence matrix for users in different cameras is being stored.

An example for a common situation in which projection to the floor improves the correspondence detection is a situation where the complete body of a user is visible in one of the cameras but only an arm is visible in another camera. In the three-dimensional case, the centroid of the fully user in the full body cloud is close to the center of mass of the user, but in the cloud representing only the arm it is close to the center of mass of the arm. Projection to the floor in this case eliminates all errors along the z-axis (where z is along the normal of the floor) and thereby allows for lower thresholds.

Projection can simply be performed by computing the normal of a plane (in our case the floor) and moving the centroid along this normal until the plane is being crossed.

A high threshold for correspondence estimation is necessary in order to reduce the false-negative detection rate, especially as the view ports of the Kinects are different which results in partial point clouds of the user. Due to the fact that all parts of the body that are not facing the camera cannot be captured, the centroid is biased along the axis through the observed user and the Kinect, toward the Kinect camera. In order to cope with registration errors and the aforementioned problems, a threshold of 0.4 m has been chosen for our experiments.

Concatenation of point clouds per user

Using the correspondence matrix calculated in step 2.6.5, we perform a lookup on the computed point clouds of the users detected in different cameras. All point clouds that belong to the same user are concatenated into a single point cloud. This results in overlapping regions being represented by points of several cameras and several viewpoints and regions that are only represented by the points of only one camera.

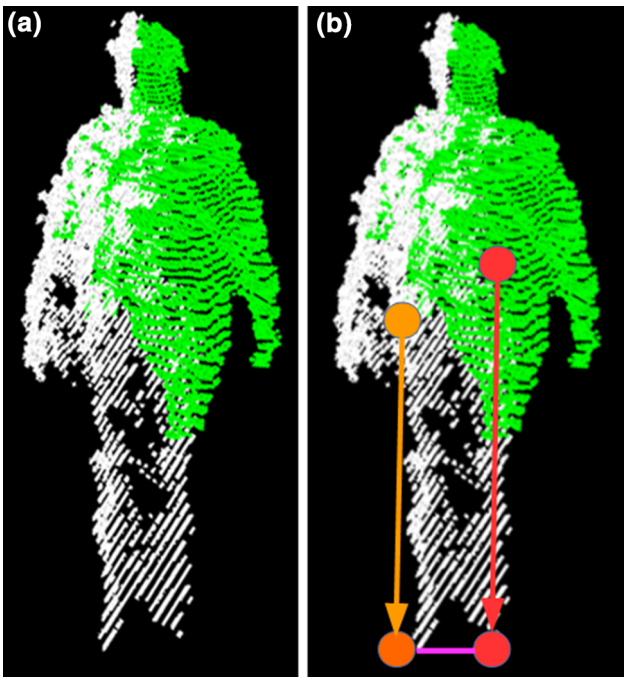

Fig. 7 *Left* user detected by two cameras, *different colors* represent data from different cameras; *right* illustration of computed centroids of the respective point clouds and their projections on the ground plane

Regions with overlapping point clouds introduce redundancy into the concatenated point cloud. In order to reduce the size of the point cloud and to remove redundancy, we downsample the point cloud using a voxel grid filter of PCL with a leaf size of 0.005 m for x, y and z. This reduces the amount of points in areas where points are dense and does not reduce the amount of points in areas with a lower point density. The downsampled point cloud is being used for distance computations and situational inference in subsequent steps of the process. Figure 7 shows a user detected by two Kinect cameras in different positions using the camera geometry seen in Fig. 5.

Distance computation using GPGPU

In order to calculate a measure the distance between humans, robots, as well as humans and robots, we implemented a CUDA-based algorithm that operates on point clouds without using any shape information. Collision checks with meshes are not included yet because in the current target scenario there is no need for mesh-based collision checking. As described in "Robot localization" section, each robot's position is detected in the scene. After successful detection, we use the CAD model of the robot and information from the position encoders of the robots joints for distance processing. We update the CAD model using Denavit–Hartenberg forward kinematics and the measured joint angles from the robot's internal sensors. In order to not rely solely on the correctness of the measured joint angles, the robots' pose is continuously monitored by the PMD camera subsystem (see "Robot pose monitoring" section). From the CAD model, we drop all triangle information and extract all vertices which can be considered to form a point cloud. Both the extracted point cloud from the CAD model and the point cloud from the processing chain, e.g., that of the users, are being loaded into CUDA pinned memory on the host computer in every updated cycle. From there, all cloud data are uploaded to the graphics RAM of a NVIDIA Geforce GTX Titan. A kernel on the NVIDIA Geforce GTX Titan computes Eq. (2)

$$d(p,q)^2 = ||q - p||^2 \tag{2}$$

which is the squared distance between points p and q. Using a brute force approach, which was fast enough for the point cloud sizes used in the experiments, we employ this kernel to compute all squared distances between the source (user) and target cloud (robot). Afterward, all resulting squared distances are compared on the graphics adapter to determine the smallest one. The points for which the squared distance is minimal are closest in the original point clouds, e.g., user and robot point cloud. We transfer back the smallest squared distance together with point indices of the points in source and target cloud to the main process where the square root for this value is being computed on the CPU, thereby calculating the distanced between both clouds. This gives a measure of the closest distance between two point clouds, which can be used for simple collision avoidance. Two thresholds have been defined. If the distance is below threshold 1 (0.4 m in our experiments), a warning is displayed to the user. If the distance is below threshold 2 (0.2 m in our experiments), the robot can be stopped in order to avoid hazardous situation like collisions. The use of thresholds describing safe and unsafe regions removes the requirement of performing a collision check in our scenario. Small distances between a user and the robot can be considered unsafe and uncomfortable for a user, especially in the case of surgical robots where the robot acts as an assisting system. However, using only the distance between user and robot is a huge over-simplification (as close distances may not always be unwanted), so in a future step the distance will be used as a feature vector for a probabilistic approach to infer about the safety.

This brute force approach only works if the data are nearly free of outliers (points that represent noise or other errors in the free space, or background), as those would also be considered as points representing parts of users. If outliers and noisy pixels cannot be completely removed, the quality of the distance computation decreases, resulting in wrong distance computation and a higher false-positive detection rate of hazardous events. Figure 8 shows the detection process from detection of a user in a single camera to final system output representing the user in the scene. Figure 9 shows the running system.

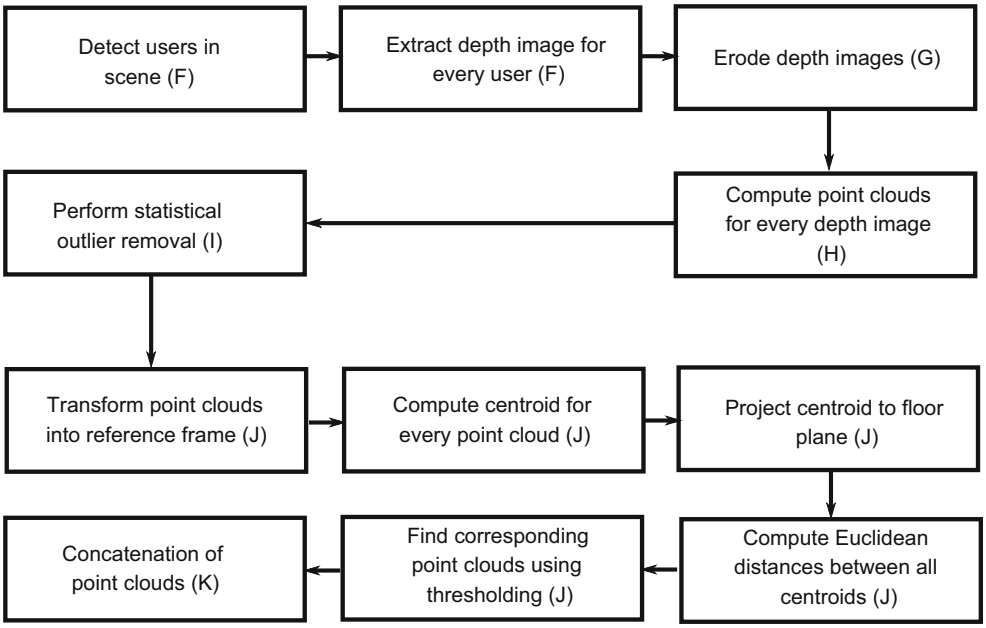

Fig. 8 Flowchart of the user detection process from detection of users in a single camera to the final point clouds representing the users. The *letter* in *brackets* represents the paragraph in which the algorithms are described

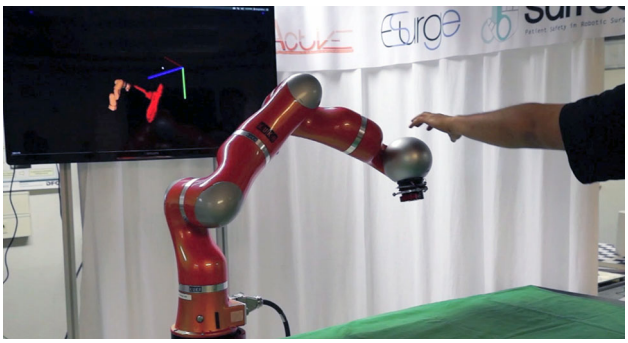

Fig. 9 Complete system during run time; foreground: supervised scene with users arm and robot; background: scene representation with user detected being too close to the robot (depicted as user colored in *red*) [2]

Skeleton fusion

Most tracking algorithms (OpenNI/NITE as well) deliver poses and orientation of the skeletal joints of a user. OpenNI uses a model-based approach to detect the joints. If two or more skeletons of one and the same user are available, a fusion of the skeletons is possible. The quality of the skeletal information is highly dependent of the orientation of the user with respect to the camera. Most current tracking algorithms deliver best results if the camera is facing the front of a user. In our approach we count the number of available skeletons for a user based on the correspondence matrix computed in 2.6.5. As the operating table is centered under the rectangular configuration, the assumption can be made that the front of the user is likely to be seen by two Kinect cameras (user on one side of the table). It is unlikely that the personnel is seen by all cameras at any time and most of the time we have to rely on one or two cameras. We observed that tracking quality decreases if a camera can see less of a user's front side. Three cases for the skeleton fusion have to be handled:

1. Skeleton tracked in only one camera: the skeletal configuration of a user is the skeletal data of the camera tracking the user.
2. Skeleton of a user tracked by two cameras: the skeleton of a user is computed using the means of the joint positions tracked by the cameras. The joint orientation is computed using quaternion slerping described in [22].
3. Skeleton of a user tracked by more than two cameras: based on the assumption that a maximum of two cameras at once can view the user's front side, the two best-fitting skeletons are computed (based on the translational component only). For every pair of skeletons (i.e., the skeleton of a user tracked in two cameras), we compute the distances between the joints. These distances result from the registration and tracking error. We find the pair of cameras in which the distances between the joint positions are the smallest, neglecting the lower body joints as they often are occluded by the operating table and other operating room equipment. We assume that the two skeletons with the smallest distances represent the actual skeleton position best. With the resulting pair of cameras the steps of 2 are performed.

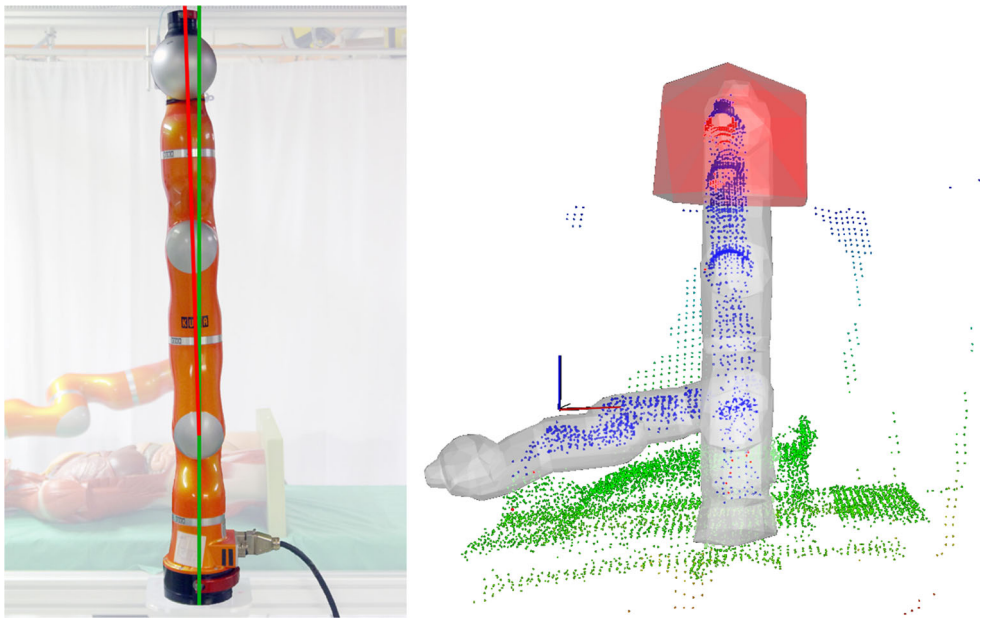

Fig. 10 Detection of a deviation of the robot pose; *left* real robot pose with deviation (marked by *central red axis*) and desired upright pose (marked by *central green axis*); *right* resulting violation of the safe zone

Robot pose monitoring

The pose of the robot is continuously supervised by the PMD camera subsystem in order to guarantee that there is no deviation between the planned and the real robot pose. This is achieved by an algorithm we call *shape cropping* in which the point cloud of the immediate surroundings of a robot is segmented into two zones. The robot zone contains all data points that belong to the robot surface itself. The safe zone can be thought of as an inflated hull of the robot that includes all points in the immediate surroundings of the robot. Both zones are created in each time step based on a simplified CAD model of the robot and the current joint values. If the robot joint values are reported incorrectly, the real robot pose in the scene deviates from the CAD model. This leads to an increased number of inliers in the safe zone which can be detected easily. Using this method, we monitor the robot continuously for deviations between planned and real pose. Figure 10 shows the detection of incorrect calibration of a robot in the first joint, leading to a violation of the safe zone. Furthermore, the same concept is used to estimate potential collisions (which is not in the scope of this article).

Results

Kinect camera system

We measured the frequency of the received image data collected and transmitted via Ethernet by the mini computer using built-in ROS mechanisms. The resulting frequency without any computation was 25 Hz. The accuracy evaluation for the checkerboard-based registration has been performed using 12 poses for the checkerboard in a volume of 1.8 m × 1.0 m × 1.5 m. The locations of the corners of the checkerboard at each of the locations for the checkerboard have been captured by both the reference camera and the camera, whose registration error had to be evaluated. Using the known transformation between each camera and the reference camera, the locations of the checkerboard corner have been transformed to the reference camera frame. Then the Euclidean distance between each corner as captured by the reference camera and the camera to be evaluated have been computed. These Euclidean distances are a measure for the registration error between the point clouds of the cameras. The data have been statistically analyzed using a Kruskal–Wallis method to show the registration error of the pairwise registered Kinects. In the following, Kinect pairs of a source K_n and the reference camera K_{ref} are called a pair.

In Table 1, the median and inter-quartile ranges of the accuracy evaluation are reported. In subsequent tables and

Table 1 RMS and Median errors in mm and quartile range for the accuracy evaluation of each pair of Kinect cameras [2]

Variable	RMS	Median value	1st quartile	3rd quartile
Pair 1	25.1064	19.8392	12.7431	28.2076
Pair 2	25.2948	26.8867	19.9021	34.8021
Pair 3	26.0307	26.6838	20.2448	35.1427

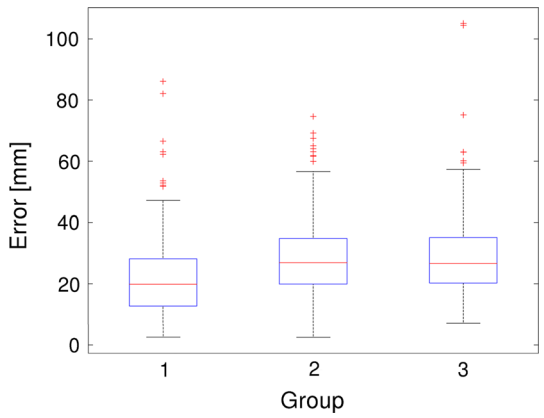

Fig. 11 Boxplot of the registration error for every checkerboard position and each pair of Kinect cameras [2]

figures, pair 1 is the pair of K_{ref} and K_1, pair 2 is the pair of K_{ref} and K_2, pair 3 is the pair of K_{ref} and K_3. The statistical analysis shows that the results for pair 1 are significantly different from the other two pairs because of the spatial configuration of the cameras, e.g., a smaller distance between the cameras.

Figure 11 shows the results of the accuracy evaluation for each group (=pair) in all the tested checkerboard positions. As can be seen on the picture, the data exhibit statistical differences in the tested workspace, which is induced by different angles and distances in between the cameras.

Figure 12 shows the results of the accuracy evaluation for each position of the checkerboard in each pair of Kinect cameras.

When introducing the detection and fusion chain, performance measurements show that an Intel Core i7 3770 can process up to four Kinect cameras using the proposed approach.

Our approach successfully removes noisy pixels at the edges of detected users but also removes parts of the useful information as well. We did not observe any voxels in free space (through noise or jumping pixels) during our experiments which made distance computation without knowledge about the objects possible.

To evaluate the delay of the camera system, we used a high-speed camera (SpeedCam MacroVis) that simultaneously captured the real scene and the resulting 3D scene displayed on a monitor. A series of events was performed and captured by the high-speed camera. The delay was calculated by measuring the time between an event in the scene and the monitor showing the result. The delay is about 950 ms on average using Kinect cameras in the setup detailed in this paper, e.g., transmission via ROS and ethernet. However, the exact delay between actual action and finished computation can only be deducted, as the time for visualization and displaying on the monitor is included in this measurement.

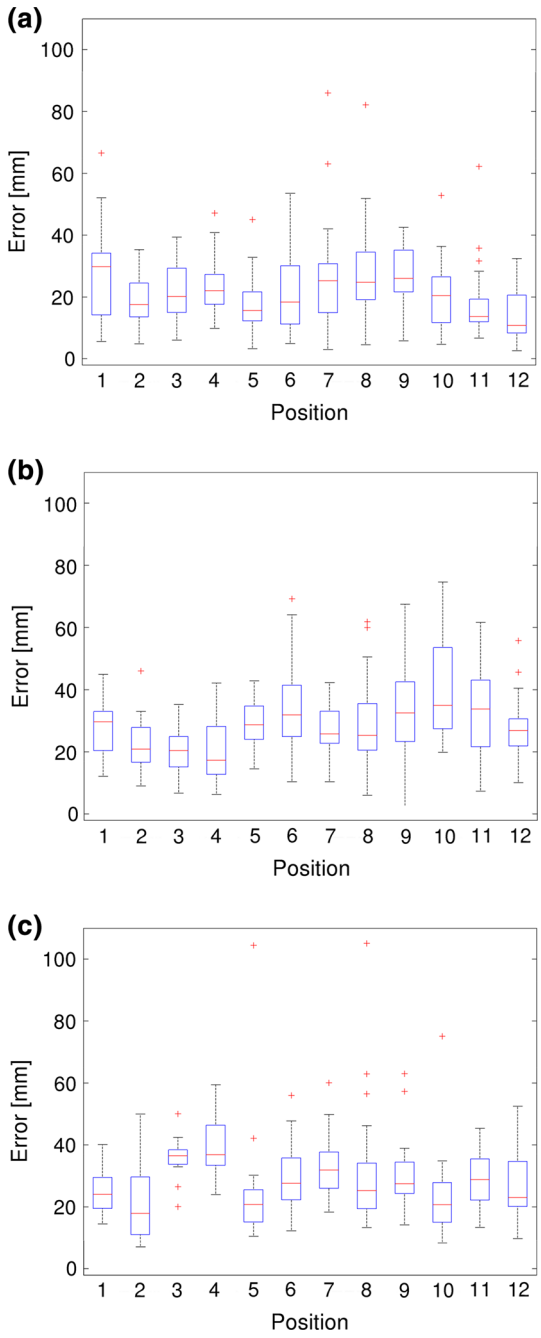

Fig. 12 Boxplot of the error for each pair of Kinects in each of the 12 checkerboard positions. **a** Pair 1, **b** pair 2, **c** pair 3

Until further experiments are carried out, we assume a worst-case scenario in which visualization and displaying introduce no significant delay and therefore estimate the delay of the Kinect system to 950 ms.

PMD camera system

The hybrid frame rate of the PMD camera system was evaluated to be 18.4 fps. The delay was evaluated with the same

method described in "Kinect camera system" section, resulting in a delay of 175 and 244 ms for camera group one and two. Both frame rate and delay are directly related to the time for scene acquisition needed by each camera. The registration error was evaluated to be below 3 cm per camera. Depending on the robot's pose, deviations from the correct pose were detected from angular errors of 2° upward. Further results have been presented in [19].

Robot localization

The robot position was localized by the Kinect camera system with the active localization method as described in "Robot localization" section. For evaluation, the robot was placed at seven different positions inside the field of view of both the Kinect camera system and the optical tracking system (OTS). All localizations were performed using only information obtained by the Kinect camera system. The reference frame for all cameras was located at the position of K_{ref} with the z-axis parallel to the normal of the floor plane (as determined in "Detection of a reference plane within the reference camera's frame" section).

For the purpose of evaluation, the Kinect cameras were additionally registered against the OTS with the method described in 2.2 in order to provide a ground truth for all measurements.

For each position, the following steps have been performed:

1. Position acquisition with OTS: The robots position was determined using the OTS. The determined position was stored as ground truth for this position.
2. Initial localization by spatial change detection: Spatial change detection was performed in the scene view of K_{ref} on the predefined motion of the robot. Based on the resulting change point cloud, the initial robot base position relative to the reference frame was calculated.
3. Refinement of initial localization: Based on the initial localization, the localization optimization was performed for each camera separately.
4. Localization at additional positions: The robot was moved to five different configurations. For each configuration, the localization optimization was performed in each camera separately.
5. Calculation of final localization result: Over all 24 positions estimations p_i (initial estimation plus five configurations, seen from 4 cameras each), a weighted average position vector p_{final} was calculated as in Eq. 3, taking into account the total number of both inliers in and inliers/outliers ratio $\frac{\text{in}}{\text{out}}$ per measurement:

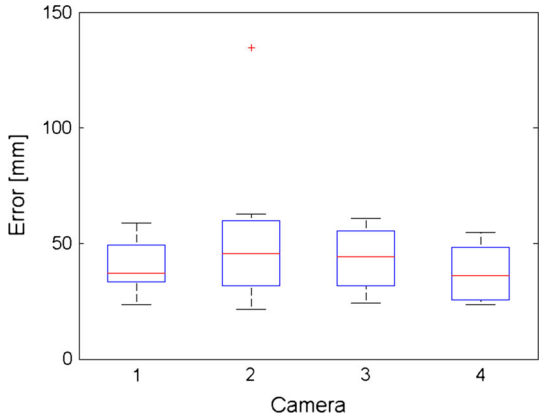

Fig. 13 Robot localization error per Kinect camera as euclidean distance between correct robot position and detected position by camera

Table 2 RMS and median errors in mm and quartile range for the robot localization accuracy of the Kinect system

	RMS	Median value	1st quartile	3rd quartile
First localization	109.5	79.5	76.9	80.4
Result by averaging	60.2	48.7	28.7	52.5
Final weighted result	44.7	38.0	29.3	53.4

$$p_{\text{final}} = \frac{1}{\sum_{i=1}^{n} \text{in}_i \cdot \frac{\text{in}_i}{\text{out}_i}} \sum_{i=1}^{24} \text{in}_i \cdot \frac{\text{in}_i}{\text{out}_i} \cdot p_i \tag{3}$$

The acquired data were first evaluated against the robot position acquired by OTS, which served as a ground truth, for each camera separately: For Kinect $K_{1-n}, n < 4$, the position detected by the camera was transformed into the OTS frame (based on the direct camera-to-OTS registration as in "Registration of depth cameras with respect to an optical tracking system (OTS)" section). The Euclidean distance between the obtained position and the ground truth was used as error metric. The median error calculated over all cameras was 41 mm. Figure 13 shows the error distribution for each camera separately.

To evaluate the robot localization accuracy of the whole Kinect camera system, we compared the localization result of the camera system (registered with the method described in 2.1) against the ground truth. Table 2 shows the respective results obtained by the initial pose refinement, calculated by a plain average over all single camera results and the final result obtained by weighting the results as shown in Eq. 3.

Kinect-based user detection

The system's overall human detection performance was evaluated using the test protocols described below:

1. One person in blue surgical clothing moves in the field of view of the camera system with a speed of 0.0–0.5$\frac{m}{s}$. A human operator observes both the movement of the person in the scene and the output of the supervision system. We measure the time and the amount of losses (system unable to detect the human) of the human in the field of view. Three iterations with a duration of 3 min have been performed.
2. The same person moves in the field of view of the system and completely stops movement, while its position is close to the operating table. We measure the time until the user is lost by the system. Thirteen iterations have been performed.
3. As before, the person is standing close to the operating table. He/she is mimicking the movement of the hands during surgical tasks by moving hands in a workspace of 0.3 m × 0.3 m × 0.3 m. As before, the time until the person is lost is measured. As no dedicated surgical intervention has been chosen, the person has been advised to perform hand movements in the workspace. The focus of this experiment is to determine whether there is a significant difference between no movement and small hand movements in terms of detection quality. Thirteen iterations have been performed.
4. After the person has been lost, he/she starts to move again. We measure the time until the person is detected by the system. Thirteen iterations have been performed.
5. Two persons are moving in the field of view of the camera system. During the experiment, the persons move close to each other until the Kinect system incorrectly identifies them as one person only. We measure the distances below which both users are identified as one and the same person (false-positive detection). Thirteen iterations have been performed.

To test the assumption that detection quality is related to the height and figure of a person, all experiments have been repeated for two persons. We have chosen a male with height 1.93 m (test person 1) and a female with height 1.62 m (test person 2) to evaluate the influence of body characteristics on detection quality.

To mimic realistic occlusions in the operating theater, a robot, an ultrasound stand and an operating bed have been placed in the field of view of the Kinect system. As the data are annotated by visual observation of the virtual scene in comparison with reality, errors are introduced due to reaction time of the observer, so all values have been rounded to full seconds. The delay of the system from action to visualization of about 950 ms has not been taken into account during data collection, so all values have been corrected afterward using a rounded delay of 1 s.

For experiment 1 and test person 1, we observed one detection loss of less than 3 s for iteration 1, one detection loss of 2 s in iteration two and no detection loss in iteration three. For experiment 1 and test person 2, we observed no detection losses in two of three iterations and one detection loss of less than 2 s in the third iteration.

Table 3 Median/mean values in sec and quartile range for the time until loss of tracking test when standing still (experiment 2)

Test person	Median value	Mean value	1st quartile	3rd quartile
Test person 1	14.0	14.3846	11.0	16.75
Test person 2	15.0	12.3846	6.0	17.25

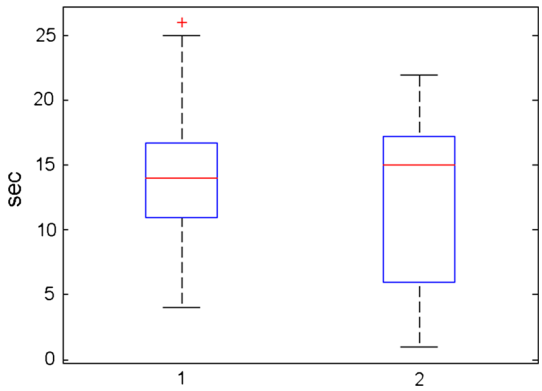

Fig. 14 Boxplot showing the results of experiment 2: time until tracking is lost while standing still with both test persons

Table 4 Median/mean values in sec and quartile range for the time until loss of tracking test when moving hands (experiment 3)

Test person	Median value	Mean value	1st quartile	3rd quartile
Test person 1	9.0	18.0769	6.0	27.0
Test person 2	16.0	17.9231	11.75	23.5

Results of experiment 2, time until loss of tracking for a previously detected person after he/she stops moving, are depicted in Table 3 and Fig. 14.

Experiment 3 shows results similar to experiment 2 which is given in Table 4, which means that there is no difference in detection quality between a person standing next to the operating table without moving and a person standing next to the operating table and moving the hands. For further evaluation, a Kruskal–Wallis test with null hypothesis "No significant difference between standing still and moving hands" has been performed. The p value of the performed test is 0.7380 when performed with the data of test person 1 and 0.1815 when performed with the data of test person 2, which indicates that there is no presumption against the null hypothesis and therefore no significant difference in between the samples of experiment 2 and experiment 3 can be assumed on the 10 % significance level.

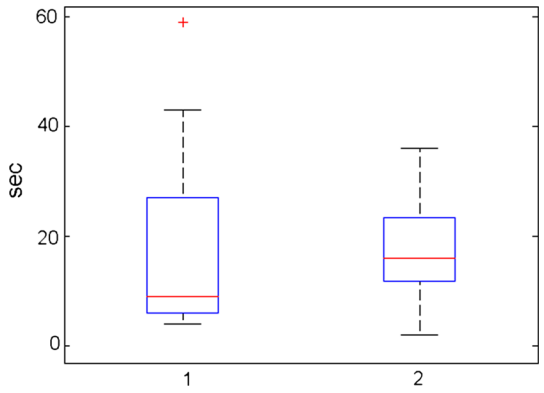

Fig. 15 Boxplot showing the results of experiment 3: time until tracking is lost while moving hands with both test persons

Table 5 Median/mean values in sec and quartile range for the time until user detected test (experiment 4)

Test person	Median value	Mean value	1st quartile	3rd quartile
Test person 1	1.0	0.6923	0.0	1.0
Test person 2	1.0	1.0769	1.0	1.0

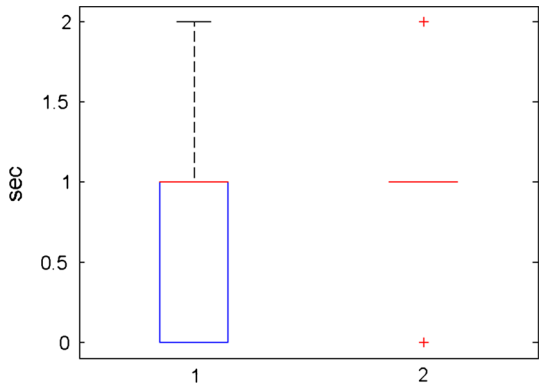

Fig. 16 Boxplot showing the results of experiment 4: time until detection of test person for both test persons

The results of experiment 3, detection robustness when moving hands only, are depicted in Table 4 and Fig. 15.

The results of experiment 4, re-detection of previously untracked person, are depicted in Table 5 and Fig. 16.

A Kruskal–Wallis test with null hypothesis "No significant difference between samples" was performed to find statistically significant differences in between the detection quality of the test persons. We compared the results of test person 1 with the results of test person 2. The p values were 0.7776 for experiment 2, 0.5207 for experiment 3 and 0.0878 for experiment 4. There are no presumptions against the null hypothesis for experiment 2 and 3, and there is only low presumption against the null hypothesis for experiment 4 on the 10 % significance level.

In experiment 5, testing at which distance two persons are incorrectly detected as one, we were not able to provoke such a mis-detection (false positive) for test person 1 in 12 of 13 samples. In the remaining sample the distance for a false positive was 0.4 m. For test person 2, in 8 samples we were not able to provoke a detection of two users as one and the same user (false positive). In two samples we observed a false-positive detection distance of 0.0 m. In two samples a false-positive detection distance of 0.1 m was measured and in one sample we observed a false-positive detection distance of 0.4 m.

Furthermore, the Kinect system has been integrated into a demonstration and validation system for the ACTIVE FP7 project that aims at improving robotics surgery for neurosurgical interventions and is integrated with the system described in [7]. The system is used for optimizing the robot's nullspace configuration based on the position of the personnel, for workflow detection and hazard detection and performed well in all use cases. The scenario allows a surgeon and a scrub nurse being close to the operating table. A third person is considered as a safety risk. Using the methods described above, we were able to successfully raise a safety alert when more than two people were closer to the robot than the allowed thresholds. However, we observed a false-positive hazard detection when moving very fast (e.g., running) in the field of view of the cameras. While this is usually not the case in a surgical setting, it can lead to a person being detected twice at different positions as the Kinect camera does not allow for active synchronization.

The false-positive and false-negative detection rates for the estimation of correspondences between a person tracked by different cameras are correlated via the threshold for correspondence estimation. A higher threshold results in a lower false-negative detection rate, but increases the false-positive detection rate. The proposed threshold value of 0.4 m led to a good performance in the aforementioned scenario.

Discussion

We propose a novel approach for fusion of detected users inside 3D depth maps as well as a registration method for RGB-D sensors. The approach is scalable, robust to interferences and noise introduced by the use of multiple Kinect cameras, and independent from models to compare the captured data with. Therefore the approach is highly flexible, can run with different detection algorithms and works also for other objects than humans and with other RGB-D cameras than the Kinect and the PMD. With regard to the latency of the PMD cameras, it has to be noted that the models of PMD cameras which are currently integrated in the setup are outdated by now. Using newer camera revisions is expected to further raise the frame rate and lower the delay. This flexibility

allows the system to be adaptable to many environments. The cameras can be positioned in the operating room where they have the best view on the scene to be supervised and where the occlusions can be minimized. The approach is scalable, which allows for the use of more cameras when complexity in the scene increases. An integration in a real operating theater would require small technical modifications, but is feasible and planned for future works.

The approach shows a registration error that makes it usable for high-level scene supervision for workflow detection and for inferring about the situation but is not suited for high precise measurements of body area, volumetric or exact distance calculation. Additionally, the removal of noisy pixels increases the model quality but removes also part of the surface around an object being detected. The use of better sensors like precise stereo cameras could reduce the error, but introduces problems like lighting and the need for stereo calibration and feature matching. The delay between the action being performed and the scene being completely processed by the Kinect subsystem is notably high. This is not a problem for inferring about long-lasting processes like workflow detection in the operating room, but makes the use of faster systems like the PMD subsystem for time critical tasks like collision avoidance necessary. In order to reduce the threshold for correspondence estimation, a shape-based model for computing a point that is closer to the center of mass of a user compared to the centroid may help.

The results of the detection and tracking tests show that the system and the integrated algorithms are more reliable when the test persons are moving in the field of view instead of standing still. This can be explained through the use of the OpenNI algorithm which is based on detecting users based on moving objects. Also during our experiments we observed that a curtain, the operating bed, a server rack and a stand for an ultrasound device were infrequently detected as a person. To overcome these shortcomings, in future iterations the Microsoft Kinect V2 camera with the Microsoft algorithms [26] will be used. The results show that moving persons in the operating theater can be detected in very short time; however, it has to be taken into account that the data are adjusted using the delay (rounded 1 s) resulting in higher values for detection time. The results also show that the system is prone to false-positive detection of two users as one user. Regarding the outcome of the Kruskal–Wallis tests comparing the system performance for different body heights and figures, we have to assume that the system does not perform significantly different for the different body heights and figures of the test persons.

Future work will include situation-based information into the process. Measuring distances to infer about hazardous situation is not enough as in cooperative control modes that may occur during operations the robot is allowed to be in contact with the surgeon. This is being worked on using probabilistic approaches based on the Kinect subsystem data. The system will be included into a workflow management system that flexibly controls an operating room using enviromental data and a knowledge-based system that models all system components as well as the medical workflow.

The final output data will be used to infer about the current situation in the operating room, to switch between workflow steps and will form the base for workflow-based assistance in the operating room. The target system is a completely integrated neurosurgical platform developed in the scope of the ACTIVE project as well as an integrated multi-purpose operating room in the scope of OP:Sense. In both projects several robots work close together with humans and share the same workspace. The system is intended to follow the workflow during the operation and provide safe and intuitive human–robot interaction during the whole intervention.

Acknowledgments This research was funded by the European Commissions Seventh Framework program within the projects Patient Safety in Robotic Surgery (SAFROS) under Grant No. 248960 and Active Constraints Technologies for Ill-defined or Volatile Environments (ACTIVE) under Grant No. 270460. The authors thank the EU for its financial support. The authors thank NVIDIA (USA) for providing two NVIDIA Geforce GTX Titan graphic adapters for the research shown in this paper.

Compliance with ethical standards

Conflict of interest The authors state that there are no conflicts of interest.

References

1. Besl P, McKay ND (1992) A method for registration of 3-D shapes. IEEE Trans Pattern Anal Mach Intell 14(2):239–256. doi:10.1109/34.121791
2. Beyl T, Nicolai P, Raczkowsky J, Worn H, Comparetti M, De Momi E (2013) Multi kinect people detection for intuitive and safe human robot cooperation in the operating room. In: 2013 16th international conference on advanced robotics (ICAR), pp 1–6. doi:10.1109/ICAR.2013.6766594
3. Bischoff R, Kurth J, Schreiber G, Koeppe R, Albu-Schaeffer A, Beyer A, Eiberger O, Haddadin S, Stemmer A, Grunwald G, Hirzinger G (2010) The KUKA-DLR lightweight robot arm—a new reference platform for robotics research and manufacturing. In: 2010 41st international symposium on robotics (ISR) and 2010 6th German conference on robotics (ROBOTIK), pp 1–8
4. Cadeddu JA, Bzostek A, Schreiner S, Barnes AC, Roberts WW, Anderson JH, Taylor RH, Kavoussi LR (1997) A robotic system for percutaneous renal access. J Urol 158(4):1589–1593
5. Castaneda V, Mateeus D, Navab N (2013) Stereo time-of-flight with constructive interference. In: 2013 IEEE transactions on pattern analysis and machine intelligence, p 1
6. Culjak I, Abram D, Pribanic T, Dzapo H, Cifrek M (2012) A brief introduction to opencv. In: 2012 proceedings of the 35th international convention MIPRO, pp 1725–1730
7. Daniele Comparetti M, Beretta E, Kunze M, De Momi E, Raczkowsky J, Ferrigno G (2014) Event-based device-behavior switching in surgical human–robot interaction. In: 2014 IEEE international conference on robotics and automation (ICRA), pp 1877–1882. doi:10.1109/ICRA.2014.6907106

8. Faion F, Friedberger S, Zea A, Hanebeck U (2012) Intelligent sensor-scheduling for multi-kinect-tracking. In: 2012 IEEE/RSJ international conference on intelligent robots and systems (IROS), pp 3993–3999. doi:10.1109/IROS.2012.6386007
9. Fischler MA, Bolles RC (1981) Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun ACM 24(6):381–395. doi:10.1145/358669.358692
10. Garland M, Le Grand S, Nickolls J, Anderson J, Hardwick J, Morton S, Phillips E, Zhang Y, Volkov V (2008) Parallel computing experiences with CUDA. IEEE Micro 28(4):13–27. doi:10.1109/MM.2008.57
11. Guthart G, Salisbury JJ (2000) The intuitivetm telesurgery system: overview and application. In: Robotics and automation. Proceedings. ICRA '00. IEEE international conference on, vol 1, pp 618–621. doi:10.1109/ROBOT.2000.844121
12. Hannaford B, Rosen J, Friedman D, King H, Roan P, Cheng L, Glozman D, Ma J, Kosari S, White L (2013) Raven-II: an open platform for surgical robotics research. IEEE Trans Biomed Eng 60(4):954–959. doi:10.1109/TBME.2012.2228858
13. Horn BKP (1987) Closed-form solution of absolute orientation using unit quaternions. Opt Soc 4(4):629–642
14. Karan B (2015) Calibration of Kinect-type RGB-D sensors for robotic applications. FME Trans 47:47–54
15. Konietschke R, Hagn U, Nickl M, Jorg S, Tobergte A, Passig G, Seibold U, Le-Tien L, Kubler B, Groger M, Frohlich F, Rink C, Albu-Schaffer A, Grebenstein M, Ortmaier T, Hirzinger G (2009) The DLR Mirosurge—a robotic system for surgery. In: Robotics and automation. ICRA '09. IEEE international conference on, pp 1589–1590. doi:10.1109/ROBOT.2009.5152361
16. Lou Y, Wu W, Zhang H, Zhang H, Chen Y (2012) A multi-user interaction system based on kinect and wii remote. In: 2012 IEEE international conference on Multimedia and Expo workshops (ICMEW), pp 667–667. doi:10.1109/ICMEW.2012.123
17. Nakazawa M, Mitsugami I, Makihara Y, Nakajima H, Habe H, Yamazoe H, Yagi Y (2012) Dynamic scene reconstruction using asynchronous multiple kinects. In: 2012 21st international conference on pattern recognition (ICPR), pp 469–472
18. Nicolai P, Beyl T, Monnich H, Raczkowsky J, Worn H (2011) Op:sense—an integrated rapid development environment in the context of robot assisted surgery and operation room sensing. In: 2011 IEEE international conference on robotics and biomimetics (ROBIO), pp 2421–2422. doi:10.1109/ROBIO.2011.6181667
19. Nicolai P, Raczkowsky J, Wrn H (2014) A novel 3D camera based supervision system for safe human–robot interaction in the operating room. J. Autom. Control Eng. 3(5):410–417
20. OpenNI organization: OpenNI (2010). http://www.openni.org. Last viewed 24-01-2014
21. Nicolai P, Brennecke T, Kunze M, Schreiter L, Beyl T, Zhang Y, Mintenbeck J, Raczkowsky J, Wörn H (2013) The OP: Sense surgical robotics platform: first feasibility studies and current research. Int J Comput Assist Radiol Surg 8:136–137
22. Pennec X (1998) Computing the mean of geometric features application to the mean rotation. Tech. Rep. RR-3371, INRIA. http://hal.inria.fr/inria-00073318
23. PrimeSense Inc.: Prime Sensor NITE 1.3 Algorithms notes (2010). http://www.primesense.com. Last viewed 24-01-2014
24. Quigley M, Conley K, Gerkey B, Faust J, Foote T, Leibs J, Wheeler R, Ng AY (2009) Ros: an open-source robot operating system. In: ICRA workshop on open source software, vol 3
25. Rusu RB, Cousins S (2011) 3D is here: Point Cloud Library (PCL). In: Proceedings of the IEEE international conference on robotics and automation (ICRA), Shanghai, China
26. Shotton J, Fitzgibbon A, Cook M, Sharp T, Finocchio M, Moore R, Kipman A, Blake A (2011) Real-time human pose recognition in parts from single depth images. In: 2011 IEEE conference on computer vision and pattern recognition (CVPR), pp 1297–1304. doi:10.1109/CVPR.2011.5995316
27. Beyl Tim, Nicolai Philip, Raczkowsky Jörg, Wörn Heinz (2012) Ein Kinect basiertes Überwachungssystem für Workflowerkennung und Gestensteuerung im Operationssaal. In: Tagungsband der 11. Jahrestagung der Deutschen Gesellschaft für Computer- und Roboterassistierte Chirurgie e.V. (CURAC)
28. Wilson AD, Benko H (2010) Combining multiple depth cameras and projectors for interactions on, above and between surfaces. In: Proceedings of the 23rd annual ACM symposium on user interface software and technology, UIST '10. ACM, New York, NY, USA, pp 273–282. doi:10.1145/1866029.1866073
29. Zhang L, Sturm J, Cremers D, Lee D (2012) Real-time human motion tracking using multiple depth cameras. In: 2012 IEEE/RSJ international conference on intelligent robots and systems (IROS), pp 2389–2395. doi:10.1109/IROS.2012.6385968