Sweeping-Based Volumetric Calibration and Registration of Multiple RGBD-Sensors for 3D Capturing Systems

Stephan Beck* Bernd Froehlich†

Virtual Reality and Visualization Research Group, Faculty of Media, Bauhaus-Universität Weimar, Germany

ABSTRACT

The accurate calibration and registration of a set of color and depth (RGBD) sensors into a shared coordinate system is an essential requirement for 3D surround capturing systems. We present a method to calibrate multiple unsynchronized RGBD-sensors with high accuracy in a matter of minutes by sweeping a tracked checkerboard through the desired capturing space in front of each sensor. Through the sweeping process, a large number of robust correspondences between the depth and the color image as well as the 3D world positions can be automatically established. In order to obtain temporally synchronized correspondences between an RGBD-sensor's data streams and the tracked target's positions we apply an off-line optimization process based on error minimization and a coplanarity constraint. The correspondences are entered into a 3D look-up table which is used during runtime to transform depth and color information into the application's world coordinate system. Our proposed method requires a manual effort of less than one minute per RGBD-sensor and achieves a high calibration accuracy with an average 3D error below 3.5 mm and an average texture reprojection error smaller than 1 pixel.

Index Terms: I.4 [Image Processing and computer vision]: Digitization and Image Capture—Camera calibration

1 INTRODUCTION

Camera-based 3D capturing systems form the technical basis for many applications in the context of virtual reality and computer graphics. A very active application domain is 3D telepresence [5, 10, 16], where users are typically captured and reconstructed in real-time in order to be represented as virtual humans. This allows groups of users from different locations to meet and collaborate in a shared virtual environment [5]. Another application domain of 3D capturing systems is skeleton tracking or 3D tracking in general. In such scenarios, the captured measurements either serve as input for 3D interaction tasks, e.g. by tracking the user's hand [27] and body [19, 21], or for the generation and animation of virtual characters [23]. In any application domain, the accuracy of the measurements from the underlying 3D capturing system is essential for the achieved quality and strongly depends on the calibration and registration of the involved camera sensors.

Our application domain is immersive 3D telepresence, where a 3D capturing system is used as input for the real-time 3D reconstruction of users. The 3D capturing system is formed by a cluster of color and depth (RGBD) sensors. In addition, the users and their input devices are tracked by a high-precision tracking system which typically defines or is rigidly linked to the application's world coordinate system. A specific challenge for 3D capturing systems is that the RGBD-sensors have to be calibrated and registered into the coordinate system which is shared with the application, i.e. all sensor contributions have to match as closely as possible in a shared coordinate system for an artifact-free surround reconstruction of the captured content. The reconstruction in the (virtual) coordinate system of the application enables direct interaction with the virtual content, e.g. natural bare-handed pointing at features of virtual objects can be directly and consistently observed by remote users [5].

In 2015, we presented an integrated method for the precise calibration and registration of RGBD-sensors into a shared coordinate system which fulfills the specific calibration challenges for 3D telepresence [4]. The calibration was performed by placing a tracked checkerboard at different locations inside the capturing volume to capture a set of correspondences (reference samples) which were then used to obtain the calibration. While the achieved accuracy is very high, the main limitation of our original method is the relatively high manual effort, i.e. it takes about 30 minutes to obtain enough reference samples per RGBD-sensors. This effort becomes infeasible if the calibration has to be repeated often, e.g. because the sensor setup has to be changed or is slightly misaligned due to accidental movements.

To overcome this limitation, our new method significantly speeds up the manual operation of capturing reference samples. Instead of placing the checkerboard target at a set of *static sampling* locations, the user *sweeps* the target at reasonable speed through the desired capturing volume for only about one minute per sensor. While the basic concept of the calibration process remains similar to our original approach, our new method has the specific challenge that we need to obtain synchronized correspondences between the color and depth sensor's image streams and the tracked checkerboard's positions. This is not a trivial task since the tracking system and the RGBD-sensor cannot be synchronized by hardware. If the unknown latency differences between the sensor streams are ignored, the resulting calibration would be significantly distorted. However, we observed that the latency differences are quite constant and developed an efficient off-line optimization process which computes these latency differences and synchronizes all involved data streams. The optimization process maximizes the coplanarity of the reconstructed checkerboards' crossing points or minimizes their 3D and 2D reprojection error.

The main contributions of our work are

- a highly accurate, *sweeping-based* calibration method which requires a manual calibration effort of only a few minutes for typical capturing setups consisting of 4 or 5 RGBD-sensors,
- an optimization process to establish synchronized correspondences from non-synchronized data streams and
- a filter chain for robust identification of outliers during the crossing point detection in RGBD-sensor image streams.

We compared our proposed *sweeping-based* method to our initial method [4] that uses *static sampling*. Our observations reveal that *sweep sampling* is only slightly less accurate than *static sampling*. We also improved the alignment of the checkerboard's local coordinate system with the tracking target, which was a major source of error in our original method.

*e-mail: stephan.beck@uni-weimar.de

†e-mail: bernd.froehlich@uni-weimar.de

Table 1: Comparison of the characteristics of state-of-the-art multi-sensor calibration methods. Intrinsic relates to the explicit identification of the intrinsic parameters of the involved color and depth sensors. Depth calibration relates to whether an explicit correction of the sensors' depth measurements is performed. Extrinsic relates to the calibration of the rigid transformation between the depth and color coordinate systems. Registration relates to whether multiple RGBD-sensors are externally registered to a (shared) reference coordinate system or in a camera-to-camera fashion (inter-camera). Geometric preservation relates to whether the calibration preserves shapes, lengths, and angles of the captured scene geometry.

Method	Intrinsic	Depth calibration	Extrinsic	Registration	Geometric preservation	Manual effort
Maimone et al. [16]	optical	no	optical	reference	no	low
Maimone et al. [17]	optical	no	optical	ref. + inter-camera	no	low
Kainz et al. [13]	optical	no	optical	ref. + inter-camera	no	medium
Beck et al. [5]	optical	yes	optical	reference	yes	high
Deng et al. [8]	optical	no	optical + geom.	inter-camera	no	medium
Avetisyan et al. [3]	optical	yes	optical	reference	yes	high
Avetisyan et al. [2]	optical	yes	optical + geom.	reference	yes	low
Beck et al. [4]	integrated volumetric color and depth calibration to a shared reference				yes	high
Our	integrated volumetric color and depth calibration to a shared reference				yes	low

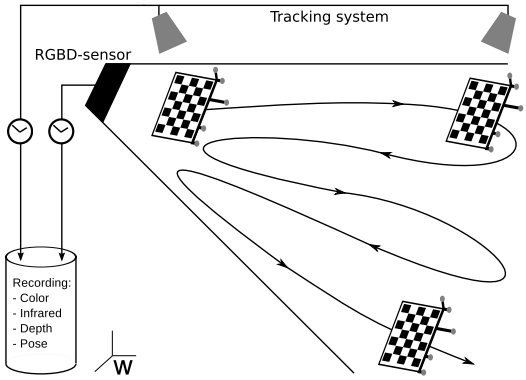


Figure 1: Fast capturing of reference samples: The users *sweeps* the tracked checkerboard at moderate speed of approx. 20 to 40 cm/s through the capturing space in front of the sensor. The *sweeping* path should overlap, i.e. the user has to *sweep* over the same regions from different directions, in order to enable our proposed off-line optimization process. Simultaneously, the sensor's color, infrared and depth image streams as well as the tracked pose are recorded to a file along with time-stamps.

2 Related Work

3D capturing systems are often based on RGBD-sensors like the Microsoft Kinect (version 1 or 2), e.g. [5, 8, 13, 16, 19, 21, 31], some systems [2, 14] combine Kinect sensors with more sophisticated color cameras. The objective of calibration is to determine the intrinsic parameters of an underlying projection model for each sensor as well as extrinsic parameters for a mapping between the sensors and to a common coordinate system. Raposo et al. [20] give a detailed definition of all involved parameters and the camera's projection model. Although most parameters are pre-calibrated by the manufacturer, more dedicated techniques are known to improve calibration accuracy [12]. Moreover, the depth sensor's distance measurement has to be calibrated too since it is affected by non-linear distortion [5, 12, 15].

State-of-the-art camera calibration methods employ a checkerboard in order to capture a set of feature points to identify the camera's intrinsic parameters based on Zhang's [29] model. Several calibration techniques have been proposed to address the specific challenges of RGBD-sensors [5, 12, 20, 25, 26, 31]. Raposo et al. [20] improved the accuracy of the joint calibration method of Herrera et al. [12] and additionally speed up of the calibration process. Staranowicz et al. [26] use a spherical object instead of a checkerboard which simplifies the calibration for non-expert users.

The calibration of multi-sensor 3D capturing systems is far more challenging than calibrating a single sensor since a large set of matching intrinsic and extrinsic parameters have to be identified to achieve high accuracy. Schmidt et al. [22] present a method for the calibration of a multi-sensor tracking system based on inter-camera error minimization, however, depth sensors are not addressed. Moreover, some application domains have specific requirements, e.g. the calibration and registration of all sensor contributions into the shared coordinate system of the application [5]. Table 1 compares the characteristics of the calibration methods that are most closely related to our work. All of them focus on the registration of multiple RGBD-sensors and aim for a perfect fusion throughout a large capturing volume.

In 2011, Maimone et al. [16] introduced a telepresence system based on multiple Kinects. They reported a 3D error ranging from approx. 1.8 cm at a distance 0.7 m to approx. 3 cm at a distance of 1.8 m from the cameras, which was mainly caused by the inaccuracy of the Kinect's depth measurement. In order to improve the 3D matching of multiple sensors, Maimone et al. [17] proposed an inter-camera-based calibration method. They captured a set of 3D correspondences inside the capturing space which were then used to fit an affine transformation for each sensor that minimized the distances between the correspondences resulting in a 3D error of approx. 1 cm. In 2012, Kainz et al. [13] presented a similar approach using a calibration sphere to obtain a set of 3D correspondences. The correspondences where then used to fit a three-dimensional polynomial function that maps depth values to world space positions at runtime for each sensor. However, they did not report quantitative results on the resulting accuracy.

In 2013, Beck et al. [5] proposed a method for the correction of the depth measurement. An optical tracking system was used as reference to obtain a look-up volume that maps from a sensor's raw depth values to metric values. As a result, the accuracy of the depth measurement was significantly improved compared to the standard approach. However, only a single rigid body transformation was used to register the sensors into the application's coordinate system. As a result, high fusion accuracy could only be achieved in the areas where the calibration target was positioned, whereas in other areas the contributions of multiple sensors matched poorly. To address this, Deng et al. [8] suggested a field of rigid transformations which is

interpolated during runtime. In 2014, Avetisyan et al. [3] presented an approach for the calibration of depth sensors conceptually similar to the one presented by [5], but slightly more accurate (0.8-1.2 cm vs. 1-2 cm).

Most calibration methods either apply explicit depth calibration or rely on inter-camera error-minimization. Consequently, these methods depend on the accuracy of a large set of interdependent and error-prone parameters. In general, the limitations of inter-camera based approaches as presented by [8, 13, 17, 22] are that the sensors have to overlap (at least two) and that the registration might result in geometrical distortions since the sensors are not registered to a metric reference space. To overcome this, Beck et al. [4] presented an integrated calibration method which is independent of any specific lens or camera model, implicitly compensates depth distortion, and, registers the sensors into a shared coordinate system which is defined by an already calibrated high-precision tracking system. A major benefit of their method is its high local fusion quality and that it avoids runtime image rectification. Zhou et al. [31] propose a volumetric calibration to compensate the depth sensor's non-linear distortion similar to [4]. However, their depth sensor calibration is performed in depth sensor space and does not register the sensor into a shared coordinate system. Moreover, they do not address color sensing.

More recently, Avetisyan et al. [2] presented a refined approach based on the ideas of [4] and [3]. Similar to [4], they place a tracked checkerboard target at different locations in front of each involved sensor. Based on a sequence of captured correspondences they calculate the sensors' intrinsic parameters and 3D lookup tables to correct the distortions of the depth sensors. They reported an average reprojection error below 0.5 pixels for each sensor. However, the extrinsic calibration was defined by a single rigid transformation per sensor, which can cause varying fusion quality throughout a larger capturing volume [8].

Our research is inspired by our earlier work where we addressed the specific challenges of multi-sensor 3D capturing systems in an integrated manner [4]. Our initial method compensates both, lens and depth distortions, and, it provides almost perfect local fusion quality for multiple sensors. However, a major drawback is the time-consuming reference sampling process which results in a very high manual effort. Our motivation is to significantly speed up the calibration process by replacing the manual reference sampling step with *sweeping* (Figure 1) while still providing high accuracy. Our proposed method has the specific challenge that we need to obtain robust and synchronized correspondences between the color and depth sensor's image streams and the position of the tracked checkerboard. This challenge was already inherent in the depth-correction methods of earlier work [2,3,5]. However, synchronization was not addressed and distortions due to differences in latency of the involved sensors were ignored. The alignment of unsynchronized sensor streams was investigated in different domains and several solutions have been presented [11,18,24,30]. Zhou et al. [30] presented a method for the estimation of the temporal offsets between unsynchronized video streams in the context of stereo vision. Sinha et al. [24] presented a visual-hull-based 3D capturing system and a method for the temporal alignment of unsynchronized video cameras. Most existing approaches are based on an optimization process (error minimization or maximum likelihood estimation) using inter-sensor correspondences. However, each approach differs from our specific challenge, either in preconditions (e.g. the knowledge of sensor intrinsics [11]) or in the measurement systems (video sensors [11, 18, 24, 30] vs. RGBD-sensors in combination with an optical tracking system).

Our new approach was developed to allow the calibration of RGBD-sensors in a very short time using *sweep sampling*, which required the synchronization between the tracking system, and the color and depth sensor streams.

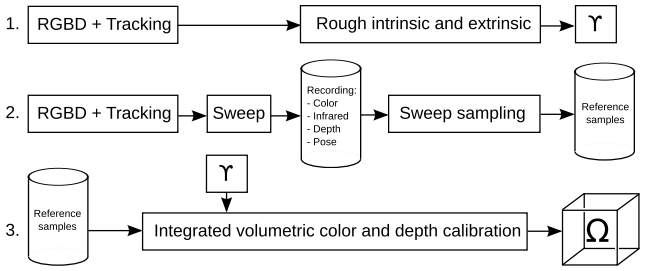


Figure 2: Overview of our proposed method. First, a rough initial calibration Υ is computed using a set of captured frames of the tracked checkerboard. Second, the user *sweeps* the tracked checkerboard inside the desired capturing space in front of the sensor. The recorded *sweep* is then used to establish a set of synchronized and robust reference samples (*sweep sampling*). Third, the final calibration Ω is computed from the reference samples by an integrated correction of Υ of using scattered data interpolation.

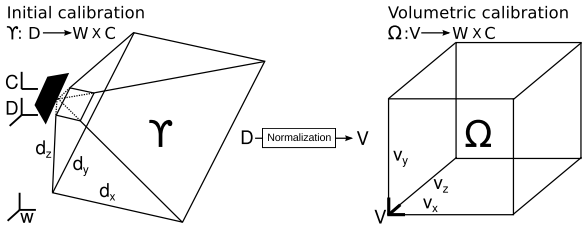


Figure 3: Relationship between the initial calibration Υ and the volumetric calibration Ω.

3 Calibration Method Overview

The central idea of our calibration method is similar to our original method [4]: for each RGBD-sensor, a rough initial calibration Υ is computed and then locally corrected by a set of reference samples $\{R\}$ to obtain our final calibration Ω. Figure 3 and 5 illustrate the related coordinate systems. The initial calibration Υ is a function that maps pixels $\mathbf{d} \in D$ from the depth sensor space D to 3D positions $\mathbf{p} \in W$ in our shared coordinate system, and to texture coordinates $\mathbf{c} \in C$ of the corresponding color space C. W is defined by the tracking system. Our final calibration Ω is defined in (normalized) volume space V and, similarly, maps to W and C.

As an overview, our proposed method is performed by the steps depicted in Figure 2. First, a rough initial calibration is computed using the state-of-the-art method from Zhang [29], however, no image rectification is applied. Second, a large set of reference samples $\{R\}$ is captured inside the capturing volume at the crossing points of a tracked checkerboard in front of each sensor. Finally, the calibration is performed by correcting the initial calibration locally and applying the correction offsets that are calculated at the reference samples.

The concept of a reference sample is illustrated in Figure 5 along with the involved coordinate systems. Each reference sample contributes to the integrated correction of the initial calibration with two correction offsets: δ_p, which corrects the calibration from the depth sensor space to world space, and δ_c which corrects the calibration from depth sensor space to color space. Figure 6 illustrates the calibration in terms of a local correction of the initial calibration which is performed in normalized volume space. As a result, the final calibration Ω maps values from depth sensor space directly to coordinates of the color sensor and to positions in the shared coordinate system of the application.

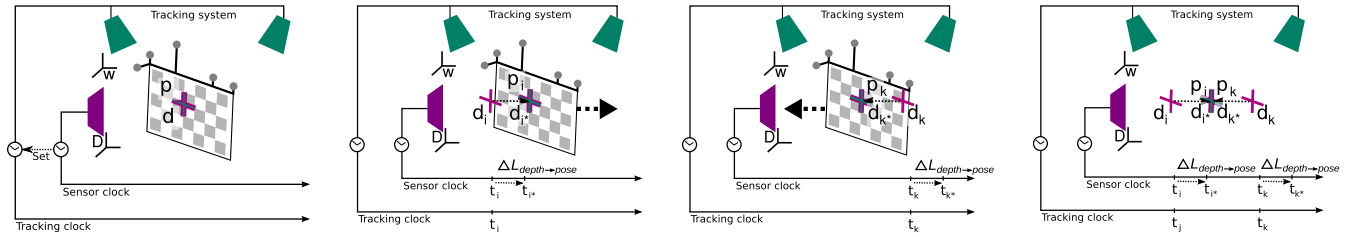


Figure 4: Illustration of the latency difference $\Delta L_{depth \to pose}$ between the depth sensor and the tracking system. (a) For a static checkerboard the measurement of the depth sensor and the tracking system coincides at $\mathbf{d}$ and $\mathbf{p}$. ((b) and (c)) If the checkerboard is in motion (during *sweeping*), the measurement of the sensor is delayed by a latency difference $\Delta L_{depth \to pose}$. As a consequence, the measured locations $\mathbf{d}_i$ and $\mathbf{d}_k$ at points in time t_i and t_k do not correspond to the measurements $\mathbf{p}_i$ and $\mathbf{p}_k$ of the tracking system. (d) Hence, the correct measurements $\mathbf{d}_{i*}$ and $\mathbf{d}_{k*}$, which correspond to $\mathbf{p}_i$ and $\mathbf{p}_k$, can only be identified if the latency difference $\Delta L_{depth \to pose}$ is known. Similarly, the measurement between the depth and the color sensor is not consistent since they have a slight latency difference $\Delta L_{depth \to color}$ too.

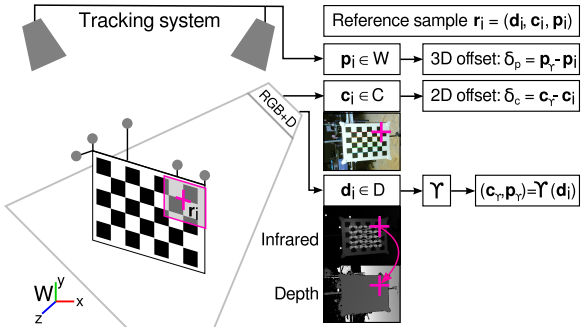


Figure 5: Schematic illustration of the involved coordinate systems during reference sampling and the correction offsets δ_p and δ_c which are used during calibration. A reference sample $\mathbf{r}_i$ is located at a crossing point of the checkerboard. The coordinates of the crossing point can be simultaneously detected in the depth space ($\mathbf{d}_i \in D$), the color space ($\mathbf{c}_i \in C$) and in world space ($\mathbf{p}_i \in W$). At $\mathbf{d}_i$, the initial calibration Υ maps to ($\mathbf{c}_\Upsilon \in C$) and ($\mathbf{p}_\Upsilon \in W$). Hence, Υ can be locally corrected using δ_p and δ_c. Please note that $\mathbf{d}_i$ is defined by the coordinate of the crossing point in the infrared image and the corresponding $z-$value in the depth image.

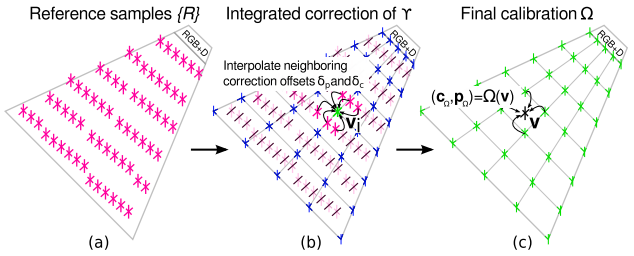


Figure 6: (a) The process of *sweep sampling* generates a large set of reference samples $\{R\}$ inside the frustum of the sensor. (b) Our integrated calibration corrects the initial calibration Υ at each voxel $\mathbf{v}_i \in V$ by applying the interpolated offsets δ_p and δ_c that are retrieved from its neighboring reference samples. (c) Our final calibration Ω maps values $\mathbf{d} \in D$ to positions $\mathbf{p}_\Omega \in W$ and $\mathbf{c}_\Omega \in C$ using (trilinear) interpolation in normalized volume space V.

4 Sweeping-based Reference Sampling

In order to speed up the manual process of taking reference samples, we developed a method that we refer to as *sweep reference sampling*. As an overview, *sweep reference sampling* is performed by the following three steps:

1. *Sweeping* the tracked checkerboard target through the desired capturing volume in front of the sensor (Figure 1).
2. Filtering and robust extraction of reference samples from the RGBD-sensor stream and the pose stream.
3. Optimization of the calibration based on 3D/2D error minimization and a coplanarity constraint.

During *sweeping*, we record the sensor's color, infrared and depth image stream and assign a time-stamp to each frame. Simultaneously, we record the tracked checkerboard's pose together with time stamps to a separate file. In order to establish a common timing of the data streams we set the clock of the tracking system to the clock of the workstation where the sensor is attached when the recording starts. Obviously, this does not synchronize the measurements of the RGBD-sensor and the tracking system since the two systems have different latencies. Figure 4 illustrates the situation: If the checkerboard is moved away from the sensor (to the right), we can suppose that at the time when a depth sensor's frame arrives in the application, its currently tracked position is already farther to the right and vice versa. In principle, it would be possible to measure the latency of a system following the method from e.g. [9]. However, we are not interested in the absolute latencies. Instead, we are only interested in the latency difference $\Delta L_{depth \to pose}$ between the depth sensor and the tracking system. Similarly, the RGBD-sensor itself typically has a slight latency difference $\Delta L_{depth \to color}$ between its depth and color measurement.

In order to resolve this issue, the latency differences $\Delta L_{depth \to pose}$ and $\Delta L_{depth \to color}$ between the captured data streams have to be identified. We present an implicit solution to this problem in Section 4.2. Please note that during *sweeping*, the image streams of the sensor as well as the pose stream are recorded to files without any on-line processing, in particular, without any image rectification or filtering.

Notations In the following, we give a few notations that we use throughout the next sections:

- $\mathbf{c}_i(t)$ is the i-th entry in the list of 2D crossing points that are detected in the color image at time t.
- $\mathbf{ir}_i(t)$ is the i-th entry in the list of 2D crossing points that are detected in the infrared image at time t.
- $\mathbf{d}_i(t)$ is the i-th entry in the list of 3D crossing points (u_i, v_i, z_i) that are detected in the corresponding depth and infrared image at time t, where $(u_i, v_i) = \mathbf{ir}_i(t)$ and $z_i = depth(u_i, v_i)$.

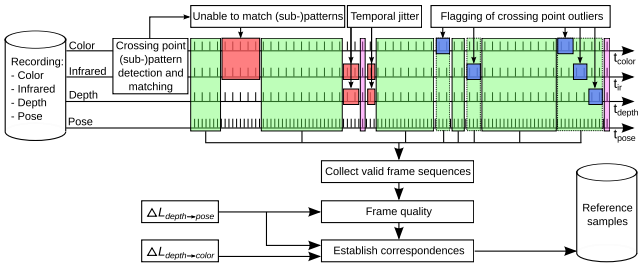


Figure 7: Schematic overview of the robust extraction of reference samples (*sweep sampling*). First, crossing point (sub-)patterns are detected in the color and infrared images and matched. If (sub-) patterns cannot be matched, both frames are marked as invalid (red). Second, frames with significantly too much temporal jitter are also marked as invalid (red). Next, crossing point outliers are detected, and flagged (blue). Of course, the illustrated erroneous frames or outliers can occur anywhere in the streams. After filtering, the remaining frame sequences (green) are collected, too short sequences (purple) are discarded. Next, frame quality is computed based on the current motion speed. Finally, the reference samples are extracted from the filtered frames at the valid (non-flagged) correspondences incorporating the latency differences $\Delta L_{depth \to pose}$ and $\Delta L_{depth \to color}$.

- $\mathbf{p}_i(t)$ is the i-th entry in the list of 3D crossing points (x_i, y_i, z_i) in world space W measured by the tracking system.

A reference sample $\mathbf{r}_i(t)$ at time t is then defined as:

$$\mathbf{r}_i(t) = (\mathbf{d}_i(t), \mathbf{c}_i(t), \mathbf{p}_i(t)) \tag{1}$$

The set $R(t)$ of reference samples per checkerboard location then is:

$$R(t) = \{\mathbf{r}_i(t)\}_{1 \le i \le k} \tag{2}$$

The set $d(t)$ of 3D crossing points per checkerboard location in the combined depth and infrared image at time t then is:

$$d(t) = \{\mathbf{d}_i(t)\}_{1 \le i \le k} \tag{3}$$

The set $c(t)$ of 2D crossing points per checkerboard location in the color image at time t then is:

$$c(t) = \{\mathbf{c}_i(t)\}_{1 \le i \le k} \tag{4}$$

The set $p(t)$ of 3D crossing points per checkerboard location in world space at time t then is:

$$p(t) = \{\mathbf{p}_i(t)\}_{1 \le i \le k} \tag{5}$$

According to above notations, we refer to the sets contained in the recorded *sweep* as $\{R\}$, $\{d\}$, $\{c\}$ and $\{p\}$.

4.1 Robust Extraction of Reference Samples

In order to obtain a set of reference samples $\{R\}$, the correspondences $\{d\}$, $\{c\}$ and $\{p\}$, have to be extracted robustly from the recorded *sweeping*. Therefore, all possible corrupt or outlying correspondences have to be detected first. While $p(t)$ is robust, we apply several filters on the sets $d(t)$ and $c(t)$. The most problematic step during *sweep sampling* is the checkerboard crossing point detection since it is very sensitive to noise and motion blur which can lead to outliers. After the filter chain is applied, all erroneous crossing points, $\mathbf{c}_i(t)$ or $\mathbf{d}_i(t)$, from $d(t)$ and $c(t)$ are flagged as invalid. Fortunately, the recorded sequence typically contains enough frames with spatial overlap such that several thousand correspondences can be established. As an overview, the following filtering steps are applied on a frame-by-frame basis (Figure 7):

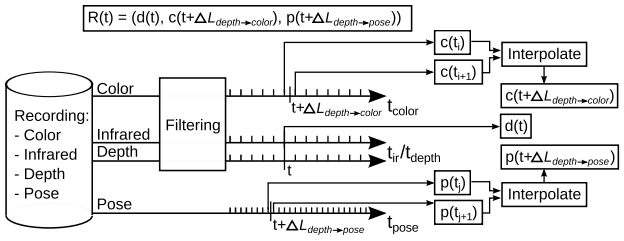


Figure 8: The establishment of the reference samples $R(t)$ at a certain time t has to incorporate the latency differences $\Delta L_{depth \to pose}$ and $\Delta L_{depth \to color}$. As a consequence, the corresponding intermediate coordinates $c(t + \Delta L_{depth \to color})$ and $p(t + \Delta L_{depth \to pose})$ have to be interpolated from adjacent frames.

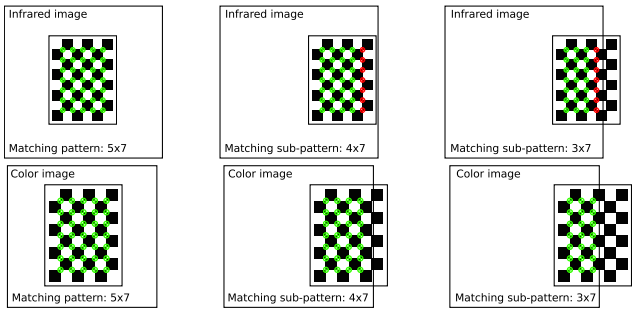


Figure 9: Cascading (sub-)pattern detection and matching between the infrared and color image. (a) The full pattern can be detected in both images. (b) and (c) When the checkerboard pattern is swept toward the border, only (sub-)patterns can be detected due to the different fields of view of the sensors. Detected crossing points which do not match between the (sub-)patterns are flagged as invalid (red).

1. Cascading (sub-)pattern detection.
2. Invalidation of frames with to much temporal jitter.
3. Statistical outlier detection.
4. Calculation of frame quality based on motion speed.

First, we try to detect the captured crossing point patterns in the corresponding color and infrared images and try to match them. This is performed in a cascading fashion, (Figure 9): if the full pattern (e.g. a pattern of 5×7 crossing points) cannot be detected, we fall back to the detection of smaller (sub-)patterns, e.g. 4×7 or 3×7. Crossing points which do not match within the two (sub-) patterns are flagged as invalid. If no matching is possible, both frames are invalidated. The situation of non-matching crossing point patterns can occur if the smallest checkerboard sub-pattern cannot be detected any more, or, when the corner detection completely fails due to noise or motion blur. In this step, we also ensure that the ordering of crossing point detection is consistent.

Second, we invalidate complete frames which exhibit too much temporal jitter since our method relies on a temporally smooth image sequence with constant latency on average. Temporal jitter can be caused by data loss during the transmission of frames from the sensor to the driver. We discard frames having a temporal deviation larger than twice the standard deviation.

Next, two types of crossing point outliers are detected, and, if apparent, the outliers are flagged as invalid. The first type of outliers are corrupt depth values due to holes in the depth image. These outliers are sometimes caused by very bright reflections of the emitted infrared light (into the depth sensor). We detect these corrupt depth values by fitting a plane through the set of $d(t)$ and flag crossing

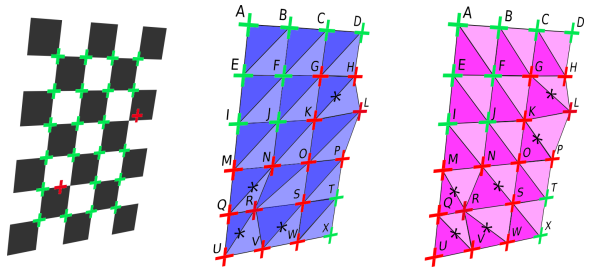


Figure 10: Robust crossing point outlier detection based on image space triangle area ratios. First, the crossing points are clustered into quadrilaterals which are formed by two types of triangle pairs (blue and purple). Next, the ratios of the triangle area pairs and the two means ($mean_h$ and $mean_v$) are computed as follows: $mean_h$ (blue triangle pairs) and $mean_v$ (purple triangle pairs), with $mean_h = mean(ratio_{AEB/FBE}, ..., ratio_{SWT/XTW})$ and $mean_v = mean(ratio_{EFA/BAF}, ..., ratio_{WXS/TSX})$. In a frame without noise, both, $mean_h$ and $mean_v$ are close to 1.0 and the corresponding standard deviations are close to 0.0. In this example, 15 ratios (blue quadrilaterals) are evaluated and compared to $mean_h$, and further, 15 ratios (purple quadrilaterals) are evaluated and compared to $mean_v$. Obviously, crossing points L and R cause several triangle ratio outliers (indicated by asterisks) which are detected. Our algorithm does not identify the crossing points L and R directly. Instead, it greedily invalidates each crossing point (marked in red) which is part of an outlying triangle pair, hence, false positives are accepted.

points if individual distances deviate more than twice the standard deviation. The second type of outliers are crossing points with wrongly detected crossing point coordinates in the infrared or color images. We greedily identify these outliers based on a statistical outlier detection filter which is illustrated in Figure 10.

Finally, we compute quality values for each set $d(t)$, $c(t)$ and $p(t)$ based on the actual motion speed. The slower the actual speed compared to the average motion speed of the whole sequence, the higher the quality and vice versa.

After the filter chain is applied, the set of reference samples $\{R\}$ is established from the remaining valid frames and the non-flagged crossing points as illustrated in Figure 8. As a basis, the frames of the depth sensor stream are used. The correspondences are then established at frames $d(t)$ with the interpolated set of $c(t + \Delta L_{depth\to color})$ from the color sensor's stream and the interpolated set $p(t + \Delta L_{depth\to pose})$ from the pose stream taking into account the latency differences $\Delta L_{depth\to pose}$ and $\Delta L_{depth\to color}$. In a last step, spatially overlapping reference samples are merged to their quality-weighted average.

4.2 Optimization of the Latency Differences

The final calibration Ω can be considered as a function of the latency differences $\Delta L_{depth\to pose}$ and $\Delta L_{depth\to color}$. In order to obtain synchronized correspondences we have to identify the optimal latency differences. The central idea of our optimization process is as follows: If we establish correspondences based on a set of reference samples, using non-optimized latency differences $\Delta L_{depth\to pose} = 0$ and $\Delta L_{depth\to color} = 0$, the resulting calibration Ω will be distorted, i.e. the calibration from depth sensor space to positions in world space $\Omega_p(d(t), \Delta L_{depth\to pose})$ as well as to the positions in color space $\Omega_c(d(t), \Delta L_{depth\to color})$ will be distorted. The distortion increases with the amount of overlap of the *sweeping* path (Figure 1 and 4). Hence, at each checkerboard location of the filtered input $d(t)$ we will observe deviations depending on $\Delta L_{depth\to pose}$ and $\Delta L_{depth\to color}$. We minimize the deviations and implicitly optimize the latency differences in our optimization using the following constraints:

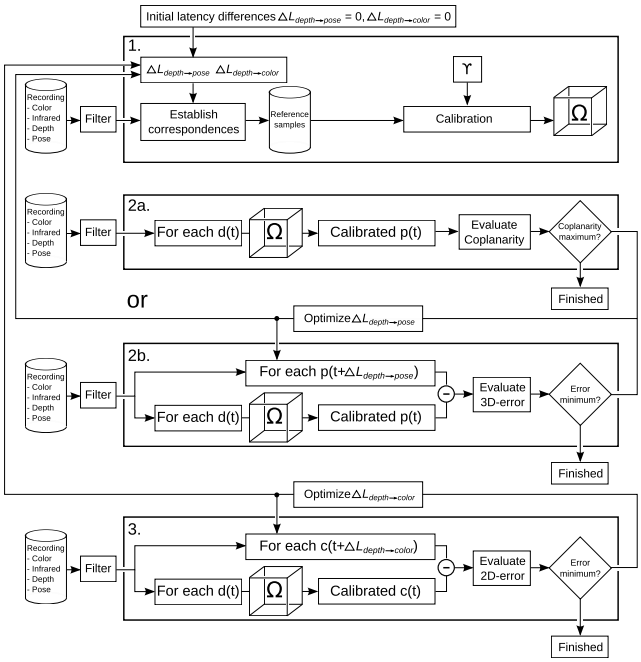


Figure 11: Schematic illustration of the optimization processes for $\Delta L_{depth\to pose}$ and $\Delta L_{depth\to color}$. (1.) The optimization starts with initial latency differences (both are set to 0.0) and the reference samples are generated in order to obtain the calibration Ω. Please note that the establishment of the correspondences depend on both, $\Delta L_{depth\to pose}$ and $\Delta L_{depth\to color}$ (Figure 8). $\Delta L_{depth\to pose}$ can be optimized using two different approaches: By maximizing the average coplanarity (2a.) or by minimizing the average 3D error (2b.) for every input $d(t)$. The optimization is finished if a coplanarity maximum (3D error minimum) is reached, otherwise $\Delta L_{depth\to pose}$ is updated and the process restarts at (1.). (3.) In a similar way, $\Delta L_{depth\to color}$ is optimized by minimizing the average 2D error for every input $d(t)$.

- Coplanarity constraint: The set of calibrated 3D points $\Omega_p(d(t), \Delta L_{depth\to pose})$ should be coplanar since d(t) is coplanar.
- 3D reprojection constraint: The set of calibrated 3D points $\Omega_p(d(t), L_{depth\to pose})$ should coincide with the corresponding 3D reference points $p(t + \Delta L_{depth\to pose})$.
- 2D reprojection constraint: The set of calibrated 2D points $\Omega_c(d(t), L_{depth\to color})$ should coincide with the corresponding 2D reference points $c(t + \Delta L_{depth\to color})$.

Given the above constraints, we are able to optimize the latency difference $\Delta L_{depth\to pose}$ in two ways: first, by searching for the maximum average coplanarity and second, by minimizing the average 3D reprojection error. The latency difference $\Delta L_{depth\to color}$ can be optimized by minimizing the average 2D reprojection error respectively. We measure the amount of coplanarity (CP) for a frame $d(t)$ using principal component analysis:

$$CP(t, \Delta L_{depth\to pose}) = planeFit(\{\Omega_p(d(t), \Delta L_{depth\to pose})\}) \quad (6)$$

The function $planeFit$ internally computes the best fitting plane of the set of 3D positions and returns a normalized fitting quality between 0.0 and 1.0, whereas 1.0 indicates a perfect fit and hence perfect coplanarity. The 3D reprojection error is measured as the average of the corresponding Euclidean distances between the 3D points $\mathbf{p}_i(t + \Delta L_{depth\to pose})$ and $\Omega_p(\mathbf{d}_i(t), \Delta L_{depth\to pose})$. The 2D reprojection error is measured as the average of the corresponding Euclidean distances between the 2D points $\mathbf{c}_i(t + \Delta L_{depth\to color})$

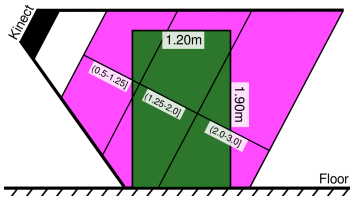


Figure 12: Schematic illustration of the capturing spaces which were evaluated. The green area is the space where a user is typically captured by our system. The purple area is the space where *sweeping* is possible. We provide results of the achieved calibration accuracy for the green area in Table 2 and for the purple space in Figure 13.

and $\Omega_c(\mathbf{d}_i(t), \Delta L_{depth \rightarrow color})$. Figure 11 gives a schematic overview of the three optimization processes. The optimization processes are independent of each other and can be performed in a brute force fashion or iteratively using the method of gradient descent. We implemented both approaches and found that curve fitting is a good approximation.

5 Results and Discussion

In our evaluation we used Kinects of version 2 with an open source driver [28]. We cropped the resolution of the color image to 1280×1080 in order to fit the field of view of the depth sensor, performed our volumetric calibration for a volume size of $128 \times 128 \times 128$ voxels and clipped the sensor's depth to a range of 0.5 m to 3.0 m. Our proposed method was implemented in C/C++, using OpenCV [6] for the crossing point detection. We used CGAL [7] for natural neighbor interpolation and for the function *planeFit* from equation (6). We used a tracking system from A.R.T. [1] for tracking the pose of the checkerboard with a very high accuracy in the range of 1-2 mm throughout a large space of about 4 m x 3 m x 2.5 m. The pattern on our checkerboard has 7×5 crossing points which we printed onto a warp-resistant board mounted on a custom stand. The initial calibration was performed using OpenCV. Please note that image rectification is not performed since the method implicitly compensates image distortion in the volumetric calibration. Please also note that we do not perform any on-line processing in terms of filtering during *sweeping*. Further, we save the acquired streams to a RAM disk in order to avoid writing stalls.

5.1 Evaluation Approach

We were interested in the accuracy of our *sweeping-based* calibration method and compared it to the accuracy of our original approach using *static sampling* [4]. We evaluated both methods with the same set of ground truth reference samples $\{R_{eval}\} = (\{d_{eval}\}, \{c_{eval}\}, \{p_{eval}\})$. The 3D accuracy is defined in terms of the average Euclidean distance between the calibrated world positions $\{p_{calib}\} = \Omega_p(\{d_{eval}\})$ and the corresponding ground truth positions $\{p_{eval}\}$. Hence, the reference for the 3D error is the tracking system. Similarly, the 2D accuracy is defined in terms of the average Euclidean distance between the calibrated color coordinates $\{c_{calib}\} = \Omega_c(\{d_{eval}\})$ and the corresponding ground truth color coordinates $\{c_{eval}\}$, which is equivalent to the 2D reprojection error used in the literature [12].

As an overview, the evaluation was performed as follows:

1. Computation of an initial calibration Υ using OpenCV.
2. *Static reference sampling* to obtain a set $\{R\}$.
3. Split of $\{R\}$ into disjunctive sets $\{R_{static}\}$ and $\{R_{eval}\}$.
4. *Sweeping* and *Sweep-based reference sampling* to obtain the set $\{R_{sweep}\}$.
5. Computation of the final calibration based on the correction of Υ using $\{R_{static}\}$.

Table 2: Average absolute errors for a Kinect v2 sensor in 3D world space and 2D texture space for *static* and *sweep sampling* and the two different scattered data interpolation schemes inverse distance weighting (*IDW*) and natural neighbor interpolation (*NNI*). For *sweep sampling*, we evaluated the accuracy without any optimization of the latency differences ($Sweep_{IDW}$) and with our proposed optimization methods using the *coplanarity constraint* ($Sweep_{IDW_{CP}}$, $Sweep_{NNI_{CP}}$) and the proposed *3D/2D error minimization* ($Sweep_{IDW_{3D/2D}}$, $Sweep_{NNI_{3D/2D}}$). For $Sweep_{IDW_{CP}}$ and $Sweep_{NNI_{CP}}$ we set $\Delta L_{depth \rightarrow color}$ to $10ms$. The resolution of the calibration volume was $128 \times 128 \times 128$. For *IDW*, only positions inside the convex hull of the input set were evaluated. 3D errors are in mm, 2D errors in pixels, standard deviations in parentheses.

Method	3D error	2D error
$Static_{IDW}$	2.29 (1.52)	0.282 (0.162)
$Static_{NNI}$	2.1 (1.53)	0.278 (0.154)
$Sweep_{IDW}$	6.25 (3.52)	1.37 (1.11)
$Sweep_{IDW_{CP}}$	3.02 (2.21)	0.951 (0.834)
$Sweep_{IDW_{3D/2D}}$	3.02 (2.21)	0.836 (0.722)
$Sweep_{NNI_{CP}}$	3.01 (2.22)	0.953 (0.858)
$Sweep_{NNI_{3D/2D}}$	3.01 (2.22)	0.842 (0.727)

6. Computation of the final calibration based on the correction of Υ using $\{R_{sweep}\}$.

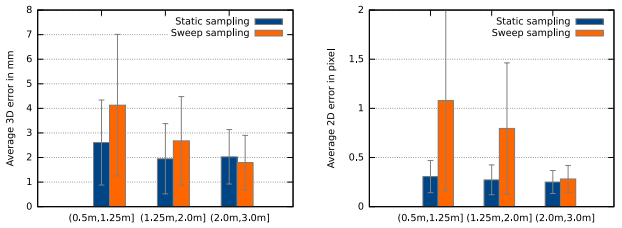
We evaluated both methods using inverse distance weighting (*IDW*) and natural neighbor interpolation (*NNI*) for scattered data interpolation following [4]. In addition, for our new *sweep sampling* approach, we evaluated our proposed optimization processes to identify the optimal latency differences in order to obtain synchronized reference samples for $\{R_{sweep}\}$.

5.2 Calibration Accuracy

In our evaluation, *static sampling* took approx. 25 minutes yielding in the set $\{R_{static}\}$ of 2100 reference samples, whereas *sweep sampling* took only about one minute yielding in the set $\{R_{sweep}\}$ of approx. 50,000 reference samples. The set $\{R_{eval}\}$ contained 2000 reference samples.

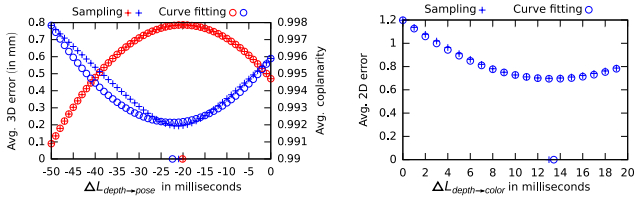
We performed our evaluation inside the two capturing areas illustrated in Figure 12. Table 2 lists the resulting accuracy of the different methods inside the capturing space which is typically covered by a Kinect in a 3D capturing setup for telepresence. The main benefit of *sweep sampling* is that a very large number of reference samples can be captured in a very short amount of time. Of course, the quality of the *sweeping-based* reference samples is affected by motion blur and inherent noise which cannot be filtered out completely. The original method using *static sampling* results in a very high accuracy ($Static_{IDW}$ and $Static_{NNI}$), confirming the results of [4]. Our new method based on *sweep sampling* results in a slightly higher 3D error of approx. 3.0 mm ($Sweep_{NNI_{3D/2D}}$) compared to approx. 2.1 mm ($Static_{NNI}$). The 2D reprojection error is slightly higher too, however, still lower than one pixel on average.

In addition, our evaluation shows that the influence of the method for scattered data interpolation ($Sweep_{IDW_{3D/2D}}$ vs.$Sweep_{NNI_{3D/2D}}$) is very low, which is mainly due to the large number of reference samples. The method for the optimization of $\Delta L_{depth \rightarrow pose}$, however, seems to have only a marginal influence. Further, it can be seen that the optimization of the latency difference $\Delta L_{depth \rightarrow color}$ between the depth and the color sensor stream is beneficial since the 2D reprojection error is lower with the 2D error minimization enabled (e.g. $Sweep_{NNI_{3D/2D}}$ vs. $Sweep_{NNI_{CP}}$). Further, the benefit of our optimization process is obvious: without optimization ($Sweep_{IDW}$), the calibration accuracy is drastically decreased.



(a) (b)

Figure 13: Comparison of the 3D errors (a) and 2D reprojection errors (b) for *static* and *sweep sampling* clustered to three depth ranges. For both methods we used NNI-based scattered data interpolation, for *sweep sampling* we used the proposed 3D/2D error minimization optimization.



(a) (b)

Figure 14: Optimization of the latency differences $\Delta L_{depth \to pose}$ (a) and $\Delta L_{depth \to color}$ (b). The crosses indicate the values of the respective error function (c.f. Section 4.2) whereas the circles indicate the values of the fitted curve. The extrema are marked on the horizontal axis. The maximum of coplanarity (red) is very close to the minimum of the 3D error (blue) (approx. at -21 ms). For $\Delta L_{depth \to pose}$, parabolas were fitted through the points at 0, -25 and -50 ms., for $\Delta L_{depth \to color}$, a parabola was fitted through the points at 0, 10 and 20 ms.

We also evaluated the accuracy of *sweep sampling* for different depth ranges in front of an RGBD-sensor as illustrated in Figure 12 and compared it to *static sampling*. The input sets $\{R_{eval}\}$, $\{R_{static}\}$ and $\{R_{sweep}\}$ were the same as in the previous evaluation and $\{R_{eval}\}$ was clustered into three depth ranges. Figure 13 shows the results. The 3D accuracy as well as the 2D accuracy of *sweep sampling* are lower in the range close to the sensor. This is mainly due to the fact that motion blur gets much higher close to the sensor during *sweeping*. E.g. if the users *sweeps* the checkerboard close to the sensor, the motion of the checkerboard pattern in the acquired images is relatively high and the detection of the checkerboard's crossing points gets less stable in this region such that many crossing points are invalidated by our filter chain and fewer reference samples can be obtained close to the sensor. However, the accuracy of our *sweeping-based* approach increases in the ranges $(1.25m, 2.0m]$ and $(2.0m, 3.0m]$. Furthermore, we recommend to simply sweep slower close to the sensor to improve the accuracy in this region.

5.3 Optimization Process

First, we evaluated the stability of the frame times, since our optimization process relies on stable latencies. Therefore, we measured the temporal deviations in the sensor streams which were recorded during *sweeping* based on the frames' time stamps. Both, the depth and color sensor acquisition takes approx. 33 ms. We measured a standard deviation of approx. 2.7 ms in the recorded color and depth sensor stream. Depth and color frames are discarded if their temporal deviation is larger than twice the standard deviation, which results in discarding about five percent of the recordings. The tracking system runs at an internal frequency of 150 Hz and our measurements have a standard deviation of 0.5 ms in the pose stream. Hence, the latency difference $\Delta L_{depth \to pose}$ and $\Delta L_{depth \to color}$ can be considered as almost constant since our filter chain discards frames with too much temporal jitter.

Next, we evaluated the three methods for the automatic optimization of the latency differences. In particular, we were interested in two aspects: first, we wanted to evaluate whether the optimizations based on coplanarity and 3D error minimization differ and, second, we wanted to evaluate whether curve fitting is a sufficient approximation of the error functions. Figure 14(a) illustrates the optimization of the latency differences between the sensor and the tracking system using the coplanarity constraint and 3D error minimization. The two optima are very close to each other. This trend was similar in other evaluations, however, the optima vary slightly for different recordings, and, we therefore recommend to perform the optimization for each calibration. In our system, $\Delta L_{depth \to pose}$ ranges between -15 ms and -30 ms. In this example, the optimal latency difference $\Delta L_{depth \to pose}$ is approx. -21 ms. A negative latency difference indicates that the tracking system runs ahead of the sensor. This is plausible since the tracking system operates at 150 Hz whereas the RGBD-sensor operates at only approx. 30 Hz. Further, curve fitting seems to be a practical choice since the parabolas approximate the 3D error function and the function for the measurement of the coplanarity very well. By fitting a parabola, the functions have to be evaluated at only three samples. Figure 14(b) illustrates the optimization of the latency difference between the depth and color sensor stream. In this example, the optimum of $\Delta L_{depth \to color}$ is approx. 13 ms, which indicates that the color stream is slightly behind the depth stream. It can be seen that curve fitting is a very good approximation of the 2D error function too.

Further, we measured the processing times of the proposed optimization methods and compare them briefly to the processing time for the computation of the calibration using *static sampling*. We used a workstation equipped with two Intel® Xeon® ten-core CPU processors running at 3.10 GHz and 128 GiB of main memory. In the case of *static sampling*, the computation of the final calibration takes approx. 5 seconds for an input of approx. 2000 reference samples. The number of reference samples depends on the manual capturing process. In the case of our proposed *sweeping* approach, the final calibration is computed off-line from the recorded *sweeping*. This step can be divided into two stages: first, the detection and filtering of the crossing points and, second, the computation of the calibration which includes the optimization process. The first stage takes approx. 33 seconds for processing approx. 50,000 reference samples. The second stage takes approx. 5 seconds using the optimization based on the coplanarity constraint and approx. 8 seconds using the 3D/2D error minimization, both, using curve fitting as an approximation. The computation times of our *sweeping-based* method are higher than in our initial method [4] since significantly more reference samples are generated.

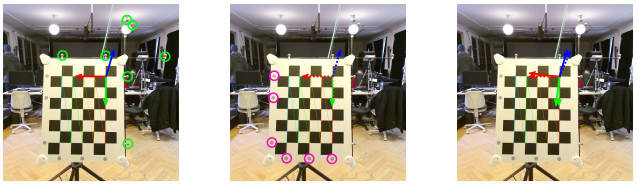
5.4 Considerations for 3D Capturing Systems

Our research focuses on the development of 3D capturing systems for 3D telepresence. We calibrated a setup of 4 Kinect v2 sensors using the original method based on *static sampling* and our proposed *sweeping*-based method to compare the resulting accuracy in terms of visual quality. Figure 15 shows screen shots of real-time 3D reconstructions which were calibrated with our proposed *sweeping-based* method and, for comparison, with our original method [4]. Differences in visual quality of both calibration approaches are largely imperceptible. The objective of our method is to register the contributions of multiple RGBD-sensors into the shared coordinate system of the application which is typically linked to a tracking system by a rigid-body transformation. Hence, the reconstructed remote users and their interactions with the virtual content are precisely registered to the shared coordinate system of the application. Figure 16 shows screen shots of a reconstructed remote user who is holding

(a) (b) (c) (d)

Figure 15: Comparison of 3D reconstructions using a capturing system formed by four Kinect v2 sensors. The real-time 3D reconstruction was performed based on the method described in [5]. For (a) and (c), the calibration was performed using *static sampling* and took approx. 90 minutes. For (b) and (d), the calibration was performed using our proposed *sweep sampling* approach and took only approx. 5 minutes. The optimization was performed using 3D/2D error minimization based on curve fitting. The detail in (c) and (d) shows the region where multiple sensor contributions overlap. The visual quality of both approaches is very similar, in fact, differences are in general imperceptible.

Figure 16: Screen shots of a reconstructed user captured by a system of 4 Kinect v2 sensors which were calibrated using our method. The virtual sword is rendered at the location of the user's input device which is tracked by our optical tracking system. Both, the virtual content and the reconstructed user are registered to the shared coordinate system of the application which enables a consistent viewing of the scene.



(a) (b) (c)

Figure 17: Precise alignment of the coordinate system on the checkerboard: (a) The spherical markers (highlighted in green) are attached to the checkerboard and define the primary local coordinate system of the checkerboard (red/green/blue). (b) An additional set of flat markers (highlighted in purple) is glued onto the checkerboard and defines a secondary coordinate system on the checkerboard (red/green/blue dashed). (c) Registration of the primary local coordinate system using the secondary coordinate system as a reference.

a virtual sword. The sword is rendered at the location of a tracked input device. Please note that the reconstructed remote user can be directly and consistently observed by local users. Please also refer to the video figure.

An important practical aspect of the proposed method is the construction of the checkerboard plate. Our calibration process relies heavily on a precise mounting of the attached tracking target since it links the checkerboard crossing points to the coordinate system of the tracked checkerboard. Rotation and translation errors result in a misalignment between the local registration of the crossing points to the tracking system. To ensure the best possible registration of the (primary) local coordinate system on the checkerboard, we use a secondary coordinate system (Figure 17). The secondary coordinate system is defined by an additional set of tracking targets which are directly glued onto the checkerboard, and, therefore is very robust and precise. In order to compensate for potential misalignments of the tracking markers, the primary coordinate system is re-calibrated using the secondary coordinate system as a reference at start up. Please note that the primary coordinate system can be also tracked from behind and from the side while the secondary coordinate system can only be tracked from ahead.

Finally, we would like to mention some additional aspects of our new approach. Since our approach uses an optical tracking system based on infrared lighting, a potential source of error is interference. Fortunately, we did not encounter interference between the Kinect v2 and our tracking system. We also tested two other tracking systems which could be potentially utilized by our proposed method, namely those delivered with the Oculus Rift™ and the HTC Vive™. Initial experiments revealed strong interference between the Kinect v2 and the HTC Vive™'s tracking system, however, we did not observe interference between the Kinect v2 and the tracking system of the Oculus Rift™. Hence, it would be possible to calibrate Kinect v2 sensors into the coordinate system of the Oculus Rift™ using our proposed method. A minor limitation is that older sensor types, like the Kinect v1, cannot be directly calibrated using our *sweeping* approach, because the sensor's depth and infrared stream cannot be transferred simultaneously from the sensor to the driver, at least in current versions. In general, *static sampling* and *sweeping* can be used in combination because the only difference lies in the process of capturing reference samples. For example, it might be valuable to capture a few additional reference samples using *static sampling* in the areas where *sweeping* is problematic, e.g. at the borders of the viewing frustum or very close to the sensor. The resulting two sets of reference samples could then be integrated and merged to a combined set, using higher quality weights for $\{R_{static}\}$ and lower weights for $\{R_{sweep}\}$. However, this would increase the manual effort but would result in improved accuracy.

6 Conclusion and Future Work

The quality of 3D capturing systems based on multiple RGBD-sensors strongly relies on a precise calibration and registration of the involved sensors. The main contribution of our work is the development of an entire software pipeline to perform a *sweeping-based* calibration and registration of multiple RGBD-sensors. As a result, our method allows users to calibrate a set of four RGBD-sensors in a matter of only five to ten minutes in contrast to our original method [4] using *static sampling* which required one to

two hours. A fundamental requirement for our *sweeping-based* calibration method is the establishment of synchronized reference samples from non-synchronized data streams. To address this, we developed and evaluated an automatic optimization process that computes the unknown latency differences between the RGBD-sensors and the tracking system. Our evaluation shows that we are able to register the sensors with an average 3D accuracy of about 3 mm and an average 2D reprojection error of less than 1 pixel for the Kinect version 2 throughout a large capturing space. While the achieved accuracy based on *sweep sampling* is slightly lower than the accuracy using *static sampling*, the perceived visual quality of exemplary 3D reconstructions appears to be very similar.

Additional noise reduction filters for increasing the robustness of the crossing point detection of *sweep sampling* have the potential to increase the accuracy of our method. Furthermore, the calibration process could be further accelerated by simultaneously capturing reference frames from all involved sensors at once. However, the occasionally occurring interferences of overlapping sensors would have to be considered in the filtering process. Immediate feedback on the achieved coverage of the capturing space during sweeping would be certainly a desirable feature of our calibration software in order to avoid large variations in calibration and registration accuracy.

Acknowledgements

This work was supported by the German Federal Ministry of Education and Research (BMBF) under grant 03IPT704X (project Big Data Analytics).

References

[1] ART. Advanced realtime tracking gmbh, 2017. http://www.ar-tracking.com/home/.

[2] R. Avetisyan, C. Rosenke, and O. Staadt. Flexible calibration of color and depth camera arrays. In *Proc. of Conference on Computer Graphics, Visualization and Computer Vision (WSCG)*, 2016.

[3] R. Avetisyan, M. Willert, S. Ohl, and O. Staadt. Calibration of depth camera arrays. In *Proc. of the 13th SIGRAD 2014 Conference of the Swedish Eurographics Chapter, Eurographics Association*, 2014.

[4] S. Beck and B. Froehlich. Volumetric calibration and registration of multiple rgbd-sensors into a joint coordinate system. In *Proc. of IEEE Symposium on 3D User Interfaces (3DUI)*, pp. 89–96, March 2015.

[5] S. Beck, A. Kunert, A. Kulik, and B. Froehlich. Immersive group-to-group telepresence. *In Transactions on Visualization and Computer Graphics, IEEE*, 19(4):616–625, April 2013.

[6] G. Bradski. Opencv. *Dr. Dobb's Journal of Software Tools*, 2000.

[7] CGAL, Computational Geometry Algorithms Library. http://www.cgal.org.

[8] T. Deng, J. C. Bazin, T. Martin, C. Kuster, J. Cai, T. Popa, and M. H. Gross. Registration of multiple rgbd cameras via local rigid transformations. In *Proc. of International Conference on Multimedia and Expo (ICME), IEEE*, pp. 1–6, 2014.

[9] S. Friston and A. Steed. Measuring latency in virtual environments. *In Transactions on Visualization and Computer Graphics, IEEE*, 20(4):616–625, Apr. 2014.

[10] H. Fuchs, A. State, and J.-C. Bazin. Immersive 3d telepresence. *Computer*, 47(7):46–52, July 2014.

[11] T. Gaspar, P. Oliveira, and P. Favaro. Synchronization of two independently moving cameras without feature correspondences. In *Proc. of Computer Vision – ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Part I*, pp. 189–204. Springer International Publishing, 2014.

[12] D. Herrera C, J. Kannala, and J. Heikkila. Joint depth and color camera calibration with distortion correction. *Trans. Pattern Anal. Mach. Intell., IEEE*, 34(10):2058–2064, Oct. 2012.

[13] B. Kainz, S. Hauswiesner, G. Reitmayr, M. Steinberger, R. Grasset, L. Gruber, E. Veas, D. Kalkofen, H. Seichter, and D. Schmalstieg. Omnikinect: real-time dense volumetric data acquisition and applications. In *Proc. of VRST 2012*, pp. 25–32. ACM Press, New York, 2012.

[14] C. Kuster, J.-C. Bazin, C. ztireli, T. Deng, T. Martin, T. Popa, and M. Gross. Spatio-Temporal Geometry Fusion for Multiple Hybrid Cameras using Moving Least Squares Surfaces. *Computer Graphics Forum*, 33(2):1–10, 2014.

[15] E. Lachat, H. Macher, M.-A. Mittet, T. Landes, and P. Grussenmeyer. First experiences with kinect v2 sensor for close range 3d modelling. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pp. 93–100, 2015.

[16] A. Maimone and H. Fuchs. Encumbrance-free telepresence system with real-time 3d capture and display using commodity depth cameras. In *Proc. of ISMAR 2011*, pp. 137–146. IEEE Computer Society, Washington, DC, USA, 2011.

[17] A. Maimone and H. Fuchs. A first look at a telepresence system with room-sized real-time 3d capture and large tracked display. In *Proc. of ICAT 2011*. ACM Press, New York, NY, USA, 2011.

[18] B. Meyer, T. Stich, and M. Pollefeys. Subframe temporal alignment of non-stationary cameras. In *Proceedings of the British Machine Vision Conference*, pp. 11.1–11.10. BMVA Press, 2008.

[19] M. Otto, P. Agethen, F. Geiselhart, and E. Rukzio. Towards ubiquitous tracking: Presenting a scalable, markerless tracking approach using multiple depth cameras. In *Proceedings of EURO VR*, 2015.

[20] C. Raposo, J. Barreto, and U. Nunes. Fast and accurate calibration of a kinect sensor. In *Proc. of the International Conference on 3D Vision - 3DV 2013*, pp. 342–349, June 2013.

[21] M. Rietzler, F. Geiselhart, J. Thomas, and E. Rukzio. Fusionkit: a generic toolkit for skeleton, marker and rigid-body tracking. In *Proceedings of the 8th ACM SIGCHI Symposium on Engineering Interactive Computing Systems, EICS 2016, Brussels, Belgium, June 21-24, 2016*, pp. 73–84, 2016.

[22] A. Schmidt, A. Kasiski, M. Kraft, M. Fularz, and Z. Domagaa. Calibration of the multi-camera registration system for visual navigation benchmarking. *International Journal of Advanced Robotic Systems*, 11(6), 2014.

[23] A. Shapiro, A. Feng, R. Wang, H. Li, M. Bolas, G. Medioni, and E. Suma. Rapid avatar capture and simulation using commodity depth sensors. *Computer Animation and Virtual Worlds*, 25(3-4):201–211, 2014.

[24] S. N. Sinha and M. Pollefeys. Visual-hull reconstruction from uncalibrated and unsynchronized video streams. In *Proceedings of the 3D Data Processing, Visualization, and Transmission, 2nd International Symposium*, 3DPVT '04, pp. 349–356. IEEE Computer Society, Washington, DC, USA, 2004.

[25] J. Smisek, M. Jancosek, and T. Pajdla. 3d with kinect. In *IEEE International Conference on Computer Vision Workshops, ICCV 2011 Workshops, Barcelona, Spain*, pp. 1154–1160, 2011.

[26] A. Staranowicz, G. R. Brown, F. Morbidi, and G. L. Mariottini. Easy-to-use and accurate calibration of rgb-d cameras from spheres. In *Proc. of Image and Video Technology: 6th Pacific-Rim Symposium, PSIVT 2013*, pp. 265–278. Springer Berlin Heidelberg, 2014.

[27] J. Taylor, L. Bordeaux, T. Cashman, B. Corish, C. Keskin, T. Sharp, E. Soto, D. Sweeney, J. Valentin, B. Luff, A. Topalian, E. Wood, S. Khamis, P. Kohli, S. Izadi, R. Banks, A. Fitzgibbon, and J. Shotton. Efficient and precise interactive hand tracking through joint, continuous optimization of pose and correspondences. *ACM Trans. Graph.*, 35(4):143:1–143:12, July 2016.

[28] L. Xiang, F. Echtler, C. Kerl, T. Wiedemeyer, Lars, hanyazou, R. Gordon, F. Facioni, laborer2008, R. Wareham, M. Goldhoorn, alberth, gaborpapp, S. Fuchs, jmtatsch, J. Blake, Federico, H. Jungkurth, Y. Mingze, vinouz, D. Coleman, B. Burns, R. Rawat, S. Mokhov, P. Reynolds, P. Viau, M. Fraissinet-Tachet, Ludique, J. Billingham, and Alistair. libfreenect2: Release 0.2, April 2016.

[29] Z. Zhang. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11):1330–1334, Nov. 2000.

[30] C. Zhou and H. Tao. Dynamic depth recovery from unsynchronized video streams. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 351–358, 2003.

[31] Q.-Y. Zhou and V. Koltun. Simultaneous localization and calibration: Self-calibration of consumer depth cameras. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 454–460, June 2014.