Improving RGB Image Consistency for Depth-Camera based Reconstruction through Image Warping

Fernando Reyes-Aviles, Philipp Fleck, Dieter Schmalstieg, Clemens Arth
Institute for Computer Graphics and Vision (ICG)
Graz University of Technology
Inffeldgasse 16/2, 8010 Graz
[fernando.reyes-aviles, philipp.fleck, schmalstieg, arth]@icg.tugraz.at

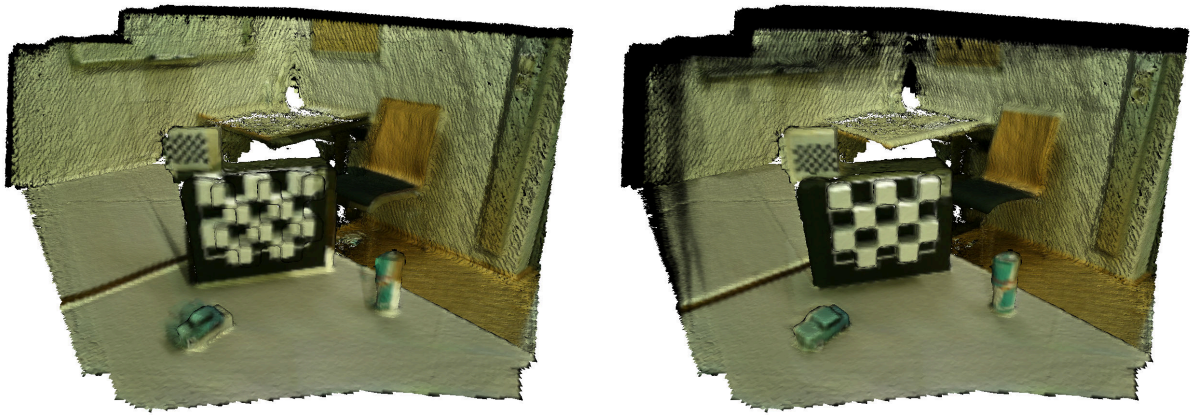


Figure 1: A partial room reconstruction using InfiniTAMv3. Left: result without applying a warping function. Right: result using our warping method.

ABSTRACT

Indoor reconstruction using depth camera algorithms (*e.g.,* InfiniTAMv3) is becoming increasingly popular. Simple reconstruction methods solely use the frames of the depth camera, leaving any imagery from the adjunct RGB camera untouched. Recent approaches also incorporate color camera information to improve consistency. However, the results heavily depend on the accuracy of the rig calibration, which can strongly vary in quality. Unfortunately, any errors in the rig calibration result in apparent visual discrepancies when it comes to colorization of the 3D reconstruction. We propose an easy approach to fix this issue for the purpose of image-based rendering. We show that a relatively simple warping function can be calculated from a 3D checkerboard pattern for a rig with poor calibration between cameras. The warping is applied to the RGB images online during reconstruction, leading to a significantly improved visual result.

Keywords

Depth camera, image warping, calibration, image-based rendering.

1 INTRODUCTION

Indoor reconstruction algorithms for depth cameras, like InfiniTAMv3, are becoming increasingly popular. Such an algorithm performs camera pose estimation based on the depth maps and feature extraction from the color images to build 3D models from image observations. In order to provide textured or colored 3D reconstructions, a large amount of approaches were developed adopting RGB-D cameras as input devices.

One of the practical issues encountered in using such approaches is the need for accurate calibration of the camera rig. The pose information from the color camera and the depth camera have to be correct in order to enable proper color and texture projection onto the 3D point cloud or mesh. RGB-D devices are often calibrated in the factory. The calibration parame-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ters are stored inside a nonvolatile memory [BMP18]. The depth-to-color registration process uses the intrinsic and extrinsic parameters stored in the nonvolatile memory, but the parameters themselves are inaccessible due to the closed nature of the sensor design.

Only a fraction of the large variety of depth camera types is shipped with an accurate internal calibration (*e.g.,* Microsoft Kinect v2). Some products provide inaccurate calibrations; custom rigs always require manual calibration. In any case, a noticeable misalignment between the depth maps and the color images makes a full calibration of the RGB-D sensor necessary. A survey of calibration literature suggests that creating a proper calibration for RGB-D rigs is surprisingly difficult because of the heterogeneous nature of the involves cameras[1].

Projection-based depth cameras use a projector casting infrared (IR) patterns observed by an IR cameras (or at least a camera fitted with IR filters). The depth maps accessible through the device driver or middleware like OpenNI[2] are generated using closed algorithms, which use a nonlinear, depth-dependent mapping that cannot be explained with a rigid transformation between the two sensors. One could compare the raw IR image delivered by the depth camera to the RGB image [BF15]. However, this assumes that the IR image and the depth image use the same projection, which is not the case for the sensors we have examined. Indeed, there are pairwise nonlinear mappings between color image, IR image and depth image.

This work was motivated by a problem encountered during the scanning of real environments for a Virtual Reality application. When reconstructing model geometry with InfiniTAMv3 [KPR+15] for an image-based rendering (IBR) system [EHH+19], we noted a significant misalignment between depth and color information (Figure 1, left).

Diving deeper into this issue (see Figure 2) makes the discrepancies more apparent. The color-depth offset is non-linearly decreasing with the distance from the camera center and increasing with distance from the optical axis of the sensor (see Figure 3). Attempts to calibrate the RGB-D sensor with standard camera calibration technique using a checkerboard pattern failed to produce usable results. We also tested the depth-sensor calibration algorithm by Ferstl *et al*. [FRR+15], but neither of those calibration methods led to a reduction of the depth-color offset.

In this work, we devise a method to calibrate RGB-D sensors using a polynomial warping function. The warping function registers the RGB images to the depth

[1] For the rest of the discussion, we focus on projector-based depth cameras only.

[2] OpenNI Github Repository

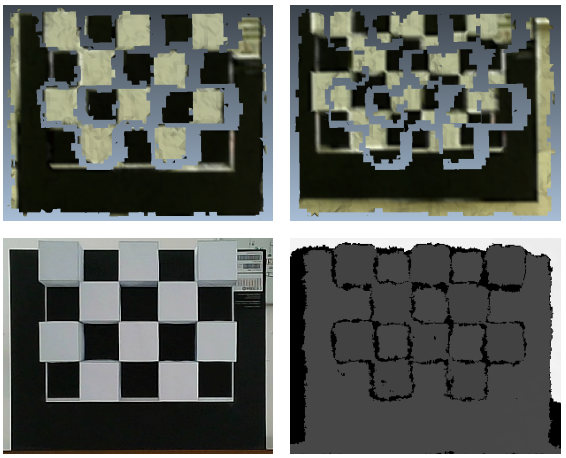


Figure 2: Visualizing depth discontinuities for Orbbec Astra Pro sensor. First row: Left, example of expected result. Right, the actual result. Second row: a corresponding pair of color image and depth map.

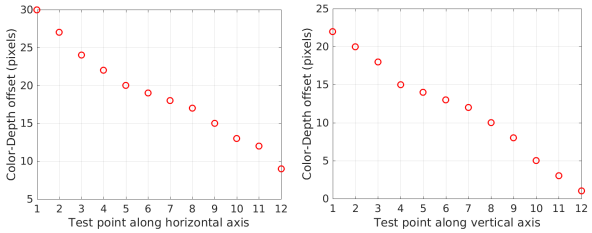


Figure 3: Non-linear offset between depth maps and color images, varying over the distance from the optical axis. These offsets were obtained by moving a known object in a static scene and measuring the depth to color offset. We moved it along the horizontal (left plot) and vertical (right plot) axis of the image plane.

maps, which produces a 1:1 correspondence between pixels in the depth map and the color image. We use a 3D equivalent of a conventional checkerboard calibration pattern, which has its white squares raised and turned into small cubes. The resulting 3D target can be reliably detected with a depth sensor. After computing the translational offset between corresponding corners in the RGB and depth images, we fit an nth degree polynomial to the resulting differences and use it to compute the warped image $RGB'(x,y) = RGB(x_w, y_w)$, with (x,y) being pixel coordinates and (x_w, y_w) denoting the coordinates retrieved from the warping function. Thus, the primary contributions of this paper can be summarized as:

1. The implementation of a calibration algorithm to retrieve translational offsets

2. An image warping registration algorithm which achieves 1:1 correspondence of color to depth

3. The demonstration of the plausibility of the calibration approach on the popular Orbbec sensors

We apply the proposed algorithm to an IBR algorithm [EHH+19] and provide some experimental results (see *e.g.,* right of Figure 1).

2 RELATED WORK

The problem of proper calibration of RGB-D cameras has been mentioned multiple times in the literature. The factory calibration of commercial RGB-D sensors, such as Kinect and Structure Sensors, is not suitable for applications requiring high quality 3D data, *i.e.*, 3D building models of centimeter-level accuracy [DTLC17]. Thus, several studies have extensively analyzed the sources of errors of such sensors [GVS18, SPB04, KBR14, HHEM16].

Simultaneous localization and mapping (SLAM) using depth sensors has become immensely popular after the publication of the KinectFusion [NIH+11] algorithm. Later work improved or extended the original capabilities [KPR+15, WSF18]. However, these methods generally assume a properly calibrated RGB-D rig.

Other work uses IBR to create high-quality results despite poor alignment of color images, essentially correcting the problem after acquisition is completed. This usually involves generating new views from recorded video frames [HRDB16, DH18, EHH+19]. The rig calibration problem was encountered in these works, but, again, no direct solutions were proposed. For the rest of the discussion, we thereby focus on the calibration problem itself.

2.1 RGB to IR camera calibration

Basic camera calibration is usually done through well-known algorithms, such as Heikkila and Silven's calibration method [HS97], Zhang's algorithm [Zha00], Bouguet's Matlab camera calibration toolbox [Bou01], or more recent work, such as Rojtberg and Kuijper's method [RK17].

Chen *et al*. [CYS+18] and Darwish *et al*. [DLT+19] calibrate RGB-D sensors with a flat checkerboard pattern via images from the RGB and IR cameras. The former calibrates multiple RGB-D sensors into a single coordinate system, while the latter improves the depth measurement accuracy.

Geiger *et al*. [GMCS12] developed a method for fully automatic camera-to-camera calibration that uses images of planar checkerboard patterns as calibration targets. They detect checkerboard corners in an image by computing a corner likelihood at each pixel in the image using corner prototypes. To produce a list of corner candidates, they apply non-maxima suppression, followed by a scoring based on gradient statistics that they threshold to obtain a final list of corners.

2.2 RGB to depth map calibration

Like most approaches, Zhang and Zhang [ZZ11] use a checkerboard pattern, but they do not calibrate the depth camera through the detection of checkerboard points in the images from the IR camera. They propose a maximum-likelihood method based on the fact that points on the checkerboard in the depth map shall be co-planar, and the plane is known from color camera calibration. They use point correspondences between the depth and color images that may be manually specified or automatically established.

Herrera C. *et al*. [HKH12] present an algorithm that simultaneously calibrates the intrinsics and extrinsics of two color cameras and a depth camera. They use a planar checkerboard pattern for calibration. In the color images, the checkerboard corners are extracted, and, in the depth maps, the four corners of the checkerboard plane are located. For the depth camera calibration, the user selects the corners of the calibration plane in the depth maps.

Jin *et al*. [JLG14] proposed an intrinsic calibration of depth sensors in which they use a set of cuboids with known sizes. Their approach consists of an iterative plane fitting and non-linear optimization that takes around five minutes to accomplish the calibration. The objective function for calibration is based on the length, width, and height of cuboids and its angle between the neighboring surfaces.

Staranowicz *et al*. [SBMM15] developed an RGB-D camera calibration algorithm for the estimation of intrinsic and extrinsic parameters. They use a recorded sequence of a moving sphere to calibrate the sensor. Ellipse and sphere-fitting algorithms are used to detect the sphere in the RGB images, and the center of the fitted sphere is reprojected onto the depth map. A great advantage is that no a-priori knowledge of the sphere's size is required.

Basso *et al*. [BMP18] developed a method to calibrate RGB-D sensors by estimating the rigid body transformation that relates the two sensors, while inferring the depth error correction function. Their method requires the depth sensor to be coupled with a calibrated RGB camera that frames approximately the same scene. First, they use the pre-calibrated RGB camera to detect a checkerboard on a wall. Then, they infer the location of the latter based on the pose of the checkerboard in the RGB images, and finally they estimate the rigid body transformation that relates the two sensors.

2.3 Learning-based approaches

Ferstl *et al*. [FRR+15] use a random regression forest to optimize the manufacturer supplied depth measurements. They detect feature points in both the RGB and the depth maps using a planar target with known dimensions.

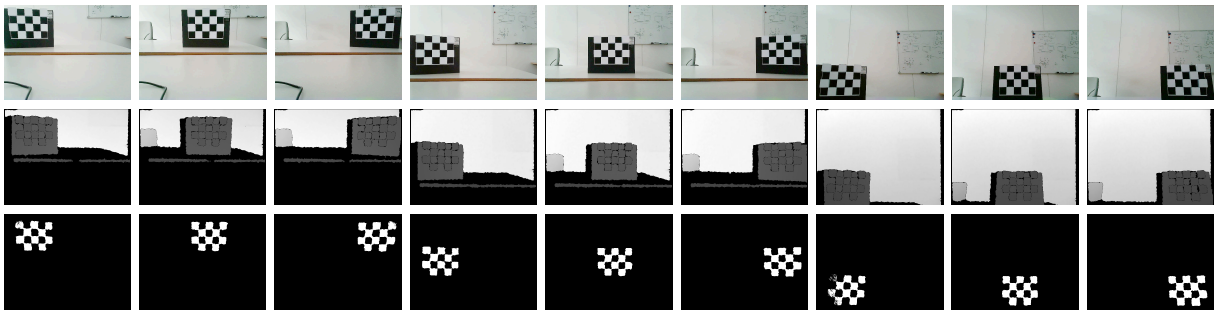


Figure 4: Sample images. RGB images I_C (top). Depth maps I_D (middle). Binary images I_T (bottom).

Teichman *et al*. [TMT13] propose an unsupervised intrinsic calibration approach built on top of a robust SLAM system. The RGB-D sensors they used to test their system gave them correct extrinsic parameters, *i.e.*, a correct depth-to-color registration. The main advantage is that no specialized hardware, calibration target, or hand measurement is required. However, their optimization approach takes several hours to converge.

2.4 Discussion

Calculating a correct differential pose of the depth map and the RGB camera is not possible directly. At least, a calculated geometric pose correction only works for a dedicated physical distance examined, but does not hold throughout the entire range of distances observed in a practical scenario.

Our approach aims at creating suitable imagery for IBR. We do not aim for improving the reconstruction directly through the use of image observations. However, while our algorithm has similarities to existing approaches, it is much more straight-forward to apply, only requires a small amount of images and works exclusively in the image domain.

3 CALIBRATION ALGORITHM

In order to describe our calibration algorithm, we first introduce some basic notation here. An RGB camera provides an RGB image denoted by I_C; a depth sensor provides a depth image denoted by I_D. From an RGB image of a scene that contains a checkerboard, it is possible to extract the checkerboard corners ${}^{I_C}B$, where the superscript I_C denotes the fact that the corners are expressed in two-dimensional pixel coordinates, *i.e.*, ${}^{I_C}B = (x,y)_1, ..., (x,y)_n$.

We do not use images from the IR sensor to calibrate the RGB-D camera for the reasons explained earlier. Instead, we directly go for a method to relate RGB images and the corresponding depth maps. For our algorithm, we require a 3D checkerboard target, which can be detected in both the RGB image and the depth map. Bamji *et al*. [BOE+15] reported a depth resolution of $5cm$ at $7.5m$ and a precision of $2.7cm$ over a range of $0.2-6m$ for commercial RGB-D sensors. Thus, we chose a minimum cube size of $125cm^3$ as features to detect in the depth image.

We record color and depth images of the 3D checkerboard in different positions (see Figure 4). We need to automatically detect as many points as possible to obtain dense 3D-to-3D correspondences. This requirement deviates from the method of Zhang and Zhang [ZZ11], who only need a few points because they estimate a rigid transformation with only six degrees of freedom. In contrast, we estimate a warping function to achieve a 1:1 correspondence of color and depth.

From the imagery, we extract the checkerboard points in all the k color-depth image pairs, with $k = 1, 2, ..., K$. Once we have the coordinates of the checkerboard points ${}^{I_C}B$ and ${}^{I_D}B$ in both color and depth images, we calculate the offsets (in pixels). Two warping functions $S^{\mathbf{x}}$ and $S^{\mathbf{y}}$ are estimated from these measurements, which warp the image in $\mathbf{x}$ and $\mathbf{y}$, respectively.

3.1 Corner detection

We use the algorithm of Geiger *et al*. [GMCS12] to detect the checkerboard corner points in both color and depth. Corner detection in RGB images is straightforward, but depth frames need pre-processing before checkerboard detection works.

First, we estimate surface normals from the raw depth image using the surface normal determination algorithm of Ückermann *et al*. [UEHR12]. This procedure results in edges becoming more distinct. The method uses the 2D image coordinates and depth value to yield valid three-dimensional vectors for every image point. The surface normals are calculated from the plane spanned by three image points in a vicinity of 5×5 pixels. Using the cross product

$$\vec{n} = (\vec{b} - \vec{a}) \times (\vec{c} - \vec{a}) \tag{1}$$

and the normal to the image plane $\vec{n}_{xy}$, we obtain the angle $\angle(\vec{n}_i, \vec{n}_{xy})$ which gives us the degree of deviation of the $\vec{n}_i$ normals from the optical center of the RGB-D sensor. We heuristically determined a threshold value of $\theta_{max} = 1^\circ$ to remove the edge pixels in the raw depth map.

Figure 5: Left: 3D checkerboard target blob. Right: Binary image I_T.

We use the output of the edge detection step to detect sets of connected components. As with most checkerboard detection algorithms, we require to board to be non-square, *i.e.*, one board dimension should be even, the other, odd. Consequently, blobs should be nearly rectangular in screen space, and we can reject outliers based on their aspect ratio.

From this rejection test, we obtain a bounding box E that delimits the blob containing the 3D checkerboard target. We use its boundary to remove all the pixels in the depth map that lie outside of E and get an image that only contains the 3D target (see left of Figure 5).

We threshold the depth map containing only the bounding box E (see left of Figure 5) to obtain the binary images I_T (see right of Figure 5 and bottom of Figure 4). To do so, we determine a plane $P(x,y)$ with three non-colinear points (see Figure 6) which allows us to remove the board's background (checkerboard black squares).

The 3D checkerboard position cannot be parallel to the image plane because, when it is moved too far away from the optical axis, the side faces of the cubes interfere with the checkerboard detection algorithm [GMCS12]. Thus, the board has to be slightly rotated when moving away from the center. Hence, we cannot use a mask which is parallel to the image plane.

Given the center E_C (red point in Figure 6) of the bounding box E, the plane points are determined as

$$\begin{aligned} P_1 &= \left(E_C^x - d, \quad E_C^y - d, \quad G_1^{min}\right) \\ P_2 &= \left(E_C^x, \quad E_C^y + d, \quad G_2^{min}\right) \\ P_3 &= \left(E_C^x + d, \quad E_C^y, \quad G_3^{min}\right) \\ d &= E_H/5, \quad e = E_H/5 \\ G_i^{min} &= min\{D_j > 0\} + \theta_T, \end{aligned} \tag{2}$$

with E_H being the height of the bounding box E, and D_j, all the depth values inside the green square G_i in Figure 6. The threshold θ_T lets us decide the depth values the plane $P(x,y)$ masks out. We experimentally determined a threshold $\theta_T = 4cm$, since we know that the height of the cube is $5cm$ (given a cube size of $125cm^3$).

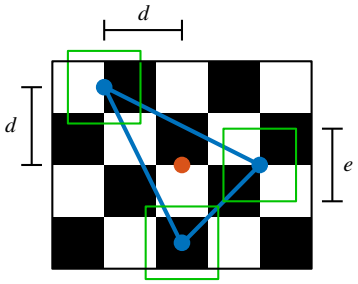


Figure 6: Points (blue) defined around the bounding box center (red) to determine the plane used as a filter to remove the board's background (checkerboard black squares).

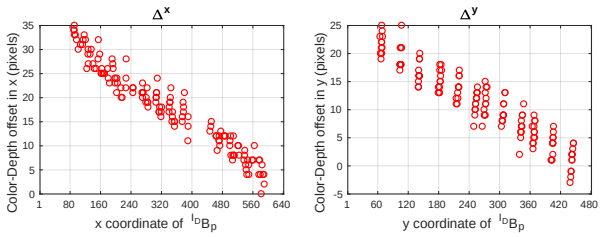


Figure 7: Sorted lists of points $\Delta_s^{x,y}$ (non-linear offset between ${}^{I_C}B$ and ${}^{I_D}B$ checkerboard corner points) obtained from the sample data set of Figure 4.

We analyze the depth value of all the pixels inside the bounding box E and binarize them to obtain I_T. Pixels with a depth value greater than $P(x,y)$ are discarded.

$$I_T(x,y) = \begin{cases} 0 & \text{if } E(x,y) \geq P(x,y) \\ 1 & \text{if } E(x,y) < P(x,y) \end{cases} \tag{3}$$

After this, we obtain binary images containing only the checkerboard (see right of Figure 5 and bottom of Figure 4) that we use as input for the detection algorithm [GMCS12].

3.2 Color-to-depth offset

From a board with $M \times N$ corner points, we obtain a set of p points, with $p = 1,2,...,M \cdot N$. Once we have the coordinates of the checkerboard points ${}^{I_C}B$ and ${}^{I_D}B$ in both color and depth images, we calculate the offset (in pixels) between each corner point ${}^{I_C}B_p$ in the I_{Ck} image and ${}^{I_D}B_p$ in the I_{Dk} image.

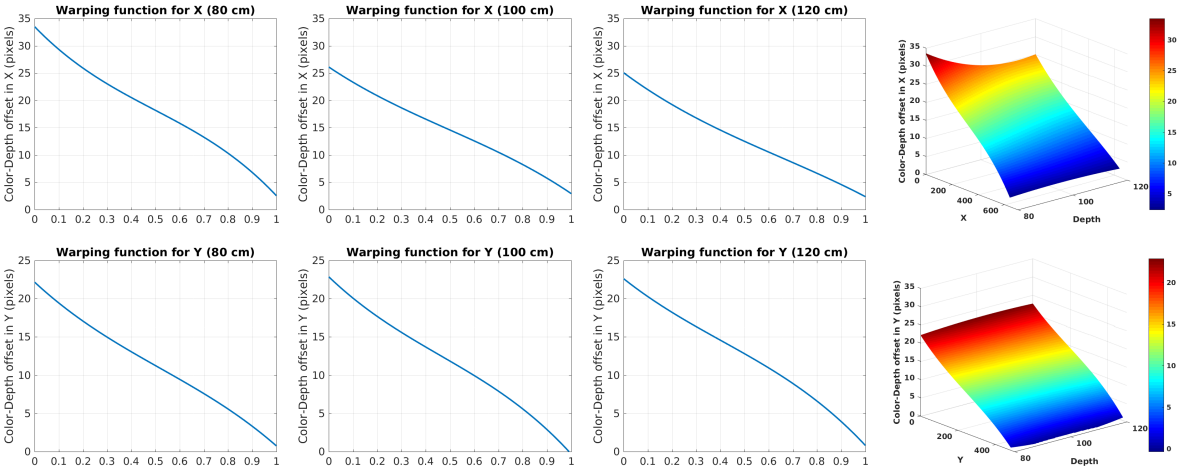
$$\Delta_p^x = {}^{I_D}B_p^x - {}^{I_C}B_p^x \quad \text{with} \quad x = 1,2,...,n \tag{4}$$

$$\Delta_p^y = {}^{I_D}B_p^y - {}^{I_C}B_p^y \quad \text{with} \quad y = 1,2,...,m \tag{5}$$

We sort the points $\Delta_p^{x,y}$ according to the coordinates of its associated point ${}^{I_D}B_p$ (see Figure 7).

3.3 Image warping

From the sorted lists of points $\Delta_s^{x,y}$, we estimate the warping functions $W^{\mathbf{x}}$ and $W^{\mathbf{y}}$. We fit cubic polynomials to Δ_s^x and Δ_s^y to obtain the warping functions $W^{\mathbf{x}}$



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

Figure 8: Warping functions $W^{\mathbf{x}}$ and $W^{\mathbf{y}}$ for different distances for the Orbbec Astra Pro sensor. (a): 80cm calibration distance. (b): 100cm calibration distance. (c): 120cm calibration distance. (d): Interpolated warping functions $S^{\mathbf{x}}$ (top) and $S^{\mathbf{y}}$ (bottom).

and $W^{\mathbf{y}}$, respectively (after the investigation of multiple sensors and setups, a cubic polynomial turned out to be sufficient). We solve the polynomial regression problem using the least squares method

$$\mathbf{A}^{\mathbf{T}}\mathbf{A}\mathbf{c} = \mathbf{A}^{\mathbf{T}}\mathbf{b}, \tag{6}$$

For $W^{\mathbf{x}}$, the vector $\mathbf{b} = \Delta_s^x$ and the matrix

$$\mathbf{A} = \begin{bmatrix} 1 & {}^{I_D}B_1^{x\,1} & \dots & {}^{I_D}B_1^{x\,n} \\ 1 & {}^{I_D}B_2^{x\,1} & \dots & {}^{I_D}B_2^{x\,n} \\ \vdots & & \ddots & \vdots \\ 1 & {}^{I_D}B_s^{x\,1} & \dots & {}^{I_D}B_s^{x\,n} \end{bmatrix} \tag{7}$$

The functions $W^{\mathbf{x}}$ and $W^{\mathbf{y}}$ can only be correctly approximated, when the 3D target in all the I_C and I_D images is at the same distance d from the image plane of the RGB-D sensor. Hence, it is required to estimate warping functions $W_d^{\mathbf{x}}$ and $W_d^{\mathbf{y}}$ for multiple distances d, respectively.

We interpolate all the available warping functions $W_d^{\mathbf{x}}$ and $W_d^{\mathbf{y}}$ to model the warping of $\mathbf{x}$ and $\mathbf{y}$ as a function of (x,D) and (y,D), respectively. We use cubic spline interpolation to obtain the functions $S^{\mathbf{x}}$ and $S^{\mathbf{y}}$ which warp all (x,y) pixels over all the depth distances D.

The function $S^{\mathbf{x},\mathbf{y}}$ is used to compute the warped image $I'_C(x,y) = I_C(x_w,y_w)$, such that

$$x_w = x + S^{\mathbf{x}}(x,D) \tag{8}$$

$$y_w = y + S^{\mathbf{y}}(y,D) \tag{9}$$

where D is the depth distance value at the pixel (x,y) of the I_D associated to I_C.

4 EXPERIMENTS

We conducted a set of experiments with different RGB-D cameras (Orbbec Astra Pro, Orbbec Astra Embedded S, *etc.*). An exhaustive overview about individual products is beyond the scope of this paper. However, in the following we discuss our findings with respect to the Orbbec Astra Pro sensor. Applying the algorithm to others led to results showing the same trends, however.

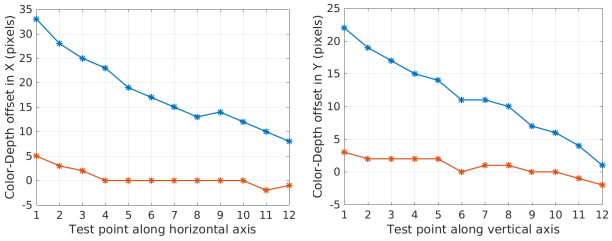


Figure 9: Offset error between depth and color images. Blue points represent the offset error, for 12 different points along the horizontal and vertical axes, before the calibration. Red points represent the error for the same 12 points after the calibration.

4.1 Calibration results

In Figure 8, the calculated warping functions for different target distances, 80cm, 100cm and 120cm, are depicted. With increasing distance from the sensor, the warping is observed to become slightly more linear. In Figure 9, the offset between ${}^{I_C}B$ and ${}^{I_D}B$ before the calibration (blue), and the offset after using the warping function on the RGB images (red). The error is significantly reduced in both the x and y dimension.

At the top of Figure 10, the offset for a test data set of 12 images is depicted. We took twelve images in different positions. The red circles are the ${}^{I_C}B$ board corners in RGB. The blue stars are the ${}^{I_D}B$ board corners in depth. The figure at the top shows the offset before applying

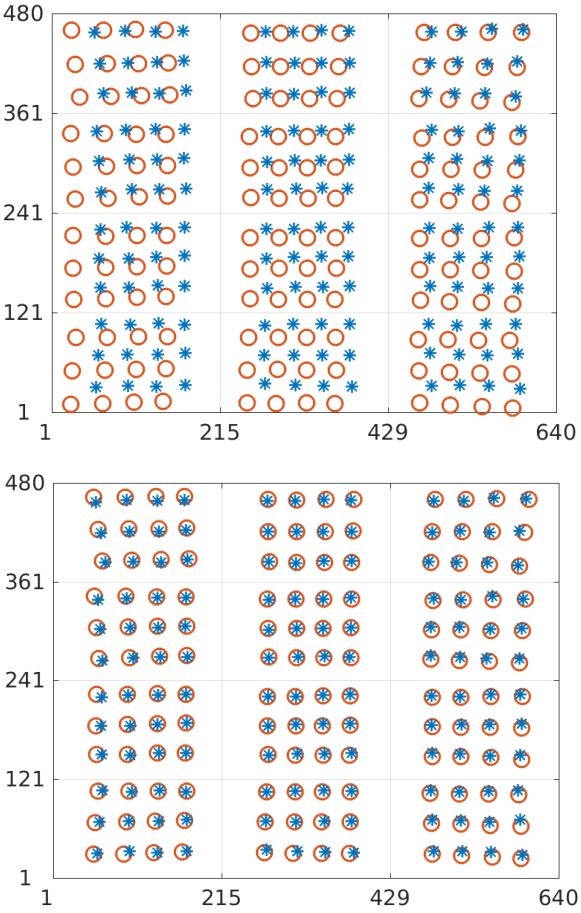


Figure 10: Scatter plot of the depth-to-color checkerboard corners offset for two data sets of 12 images. Red circles represent the ${}^{I_C}B$ corners, and blue stars, the ${}^{I_D}B$ ones. Top: offset before calibration. Bottom: offset after calibration.

the $S^{\mathbf{x}}$ and $S^{\mathbf{y}}$ functions. Similarly, for a second testset, the result is shown at the bottom of Figure 10.

4.2 Colored point clouds

The warping can be done online or offline. The $S^{\mathbf{x},\mathbf{y}}$ functions are used to create a lookup table that can be consulted at runtime in order to apply the correction. We added this functionality to InfiniTAMv3 without any noticeable performance drop. We compute $I'_C(x,y) = I_C(x_w, y_w)$ for every input frame I_C. An example of a colorized point cloud is showed in Figure 1, without and leveraging our warping method. Another example is shown in the top row of Figure 2, denoting the result after (expected) and before (original) warping.

Another example is shown in Figure 11. The borders of the tables in the scene prominently stick out, as well as the partial erroneous texturing of parts of the mobile computer, the mouse next to it, and the break of the lamp in the back. Using the warping method, the overall visual quality is significantly improved.

4.3 Image-based rendering

Erat *et al.* proposed an algorithm for generating an unstructured lumigraph in real-time from an image stream. Despite calculating the texture beforehand, the texture mapping onto a 3D model is performed online based on the actual view point onto the scene. For a more explicit description of the approach, the interested reader is referred to [EHH+19].

The overall quality of the approach is heavily dependent on the accuracy of the RGB poses given to the algorithm. Passing the original RGB images from the sensor to the system results in clear visual artefacts and inaccuracies, as depicted on the left of Figure 12, while the results of our warping-based approach for the same scenes are depicted on the right.

For the scene depicted in Figure 12, the most obvious improvements can be observed at the 3D checkerboard pattern itself. Another visible inaccuracy is present at the left boundary of the monitor, which is completely cropped in the original approach, but nicely preserved in our new method.

5 CONCLUSION

In this paper, we presented a method to calibrate RGB-D sensors using a polynomial warping function. The warping function registers the color images to the depth maps producing a 1:1 correspondence between pixels in the former and the latter. This method is particularly useful when the camera extrinsics determination, through both color and depth intrinsics, does not give a satisfactory result.

We presented a set of experiments using the popular Orbbec Astra Pro RGB-D sensor along reconstruction and IBR systems, showing the quality improvement obtained after performing the warping.

The main drawback of our approach is the need to take multiple images of the same 3D target, at different distances from the sensor, to obtain a dense enough 3D warping function. The more 2D warping functions it contains the better the results the warping produces, though, a substantial improvement is noticeable with only a few warping entries.

The main advantage of the proposed method is that a depth-to-color correspondence between unrelated (or factory attached) RGB cameras and depth sensors could be achieved. For example, it could be used to build a camera rig made up of multiple depth sensors and a high definition color camera. This is subject to further research and development in the future.

6 ACKNOWLEDGMENTS

The authors wish to thank Okan Erat. This work was supported by FFG grant 859208.

Figure 11: Results from a reconstruction recorded with InfiniTAMv3. Left: original results. Right: results generated with the proper warping functions.

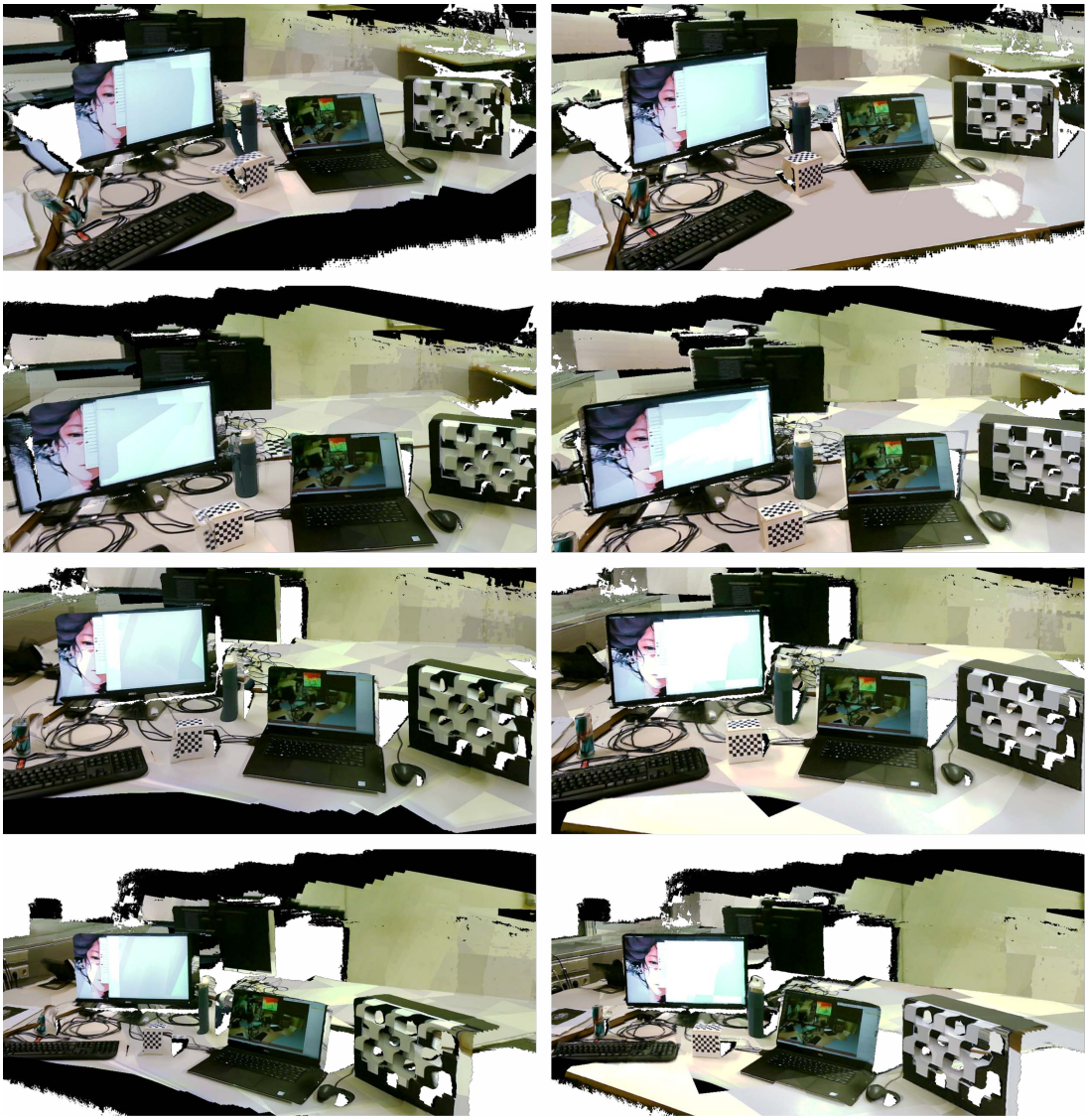


Figure 12: Results from the IBR pipeline of [EHH+19]. Left: original results. Right: results generated with the proper warping functions.

7 REFERENCES

[BF15] S. Beck and B. Froehlich. Volumetric calibration and registration of multiple rgbd-sensors into a joint coordinate system. In *3DUI*, pages 89–96, March 2015.

[BMP18] F. Basso, E. Menegatti, and A. Pretto. Robust intrinsic and extrinsic calibration of rgb-d cameras. *IEEE Trans. on Robotics*, 34(5):1315–1332, Oct 2018.

[BOE+15] C. S. Bamji, P. O'Connor, T. Elkhatib, S. Mehta, B. Thompson, L. A. Prather, D. Snow, O. C. Akkaya, A. Daniel, A. D. Payne, T. Perry, M. Fenton, and V. Chan. A 0.13 μm cmos system-on-chip for a 512 x 424 time-of-flight image sensor with multi-frequency photo-demodulation up to 130 mhz and 2 gs/s adc. *IEEE Journal of Solid-State Circuits*, 50(1):303–319, Jan 2015.

[Bou01] J. Y. Bouguet. Camera calibration toolbox for matlab. `http://www.vision.caltech.edu/bouguetj/calib_doc/`, 2001.

[CYS+18] C. Chen, B. Yang, S. Song, M. Tian, J. Li, W. Dai, and L. Fang. Calibrate multiple consumer rgb-d cameras for low-cost and efficient 3d indoor mapping. *Remote Sensing*, 10(2), 2018.

[DH18] S. Dong and T. Höllerer. Real-time re-textured geometry modeling using microsoft hololens. In *VR*, pages 231–237, 2018.

[DLT+19] W. Darwish, W. Li, S. Tang, B. Wu, and W. Chen. A robust calibration method for consumer grade rgb-d sensors for precise indoor reconstruction. *IEEE Access*, 7:8824–8833, 2019.

[DTLC17] W. Darwish, S. Tang, W. Li, and W. Chen. A new calibration method for commercial rgb-d sensors. *Sensors*, 17(6), 2017.

[EHH+19] O. Erat, M. Hoell, K. Haubenwallner, C. Pirchheim, and D. Schmalstieg. Real-time view planning for unstructured lumigraph modeling. *TVCG*, 25(11):3063–3072, Nov 2019.

[FRR+15] D. Ferstl, C. Reinbacher, G. Riegler, M. Rüther, and H. Bischof. Learning depth calibration of time-of-flight cameras. In *BMVC*, pages 102.1–102.12. BMVA Press, September 2015.

[GMCS12] A. Geiger, F. Moosmann, Ö. Car, and B. Schuster. Automatic camera and range sensor calibration using a single shot. In *ICRA*, pages 3936–3943, May 2012.

[GVS18] S. Giancola, M. Valenti, and R. Sala. *A Survey on 3D Cameras: Metrological Comparison of Time-of-Flight, Structured-Light and Active Stereoscopy Technologies*. SpringerBriefs in Computer Science. Springer Int. Pub., 2018.

[HHEM16] R. Horaud, M. Hansard, G. Evangelidis, and C. Ménier. An overview of depth cameras and range scanners based on time-of-flight technologies. *Machine Vision and Applications*, 27(7):1005–1020, Oct 2016.

[HKH12] D. Herrera C., J. Kannala, and J. Heikkila. Joint depth and color camera calibration with distortion correction. *PAMI*, 34(10):2058–2064, 2012.

[HRDB16] P. Hedman, T. Ritschel, G. Drettakis, and G. Brostow. Scalable Inside-Out Image-Based Rendering. *ACM Trans. Graph.*, 35(6):231:1–231:11, 2016.

[HS97] J. Heikkila and O. Silven. A four-step camera calibration procedure with implicit image correction. In *CVPR*, pages 1106–1112, June 1997.

[JLG14] B. Jin, H. Lei, and W. Geng. Accurate intrinsic calibration of depth camera with cuboids. In *ECCV*, pages 788–803, Cham, 2014. Springer International Publishing.

[KBR14] A. Kadambi, A. Bhandari, and R. Raskar. *3D Depth Cameras in Vision: Benefits and Limitations of the Hardware*, pages 3–26. Springer International Publishing, Cham, 2014.

[KPR+15] O. Kähler, V. A. Prisacariu, C. Y. Ren, X. Sun, P. H. S. Torr, and D. W. Murray. Very high frame rate volumetric integration of depth images on mobile devices. *TVCG*, 21(11):1241–1250, 2015.

[NIH+11] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *ISMAR*, pages 127–136, 2011.

[RK17] P. Rojtberg and A. Kuijper. [poster] efficient pose selection for interactive camera calibration. In *ISMAR (adj. material)*, pages 182–183, 2017.

[SBMM15] A. N. Staranowicz, G. R. Brown, F. Morbidi, and G. Mariottini. Practical and accurate calibration of rgb-d cameras using spheres. *CVIU*, 137:102 – 114, 2015.

[SPB04] J. Salvi, J. Pages, and J. Batlle. Pattern codification strategies in structured light systems. *Pattern Recognition*, 37(4):827 – 849, 2004. Agent Based Computer Vision.

[TMT13] A. Teichman, S. Miller, and S. Thrun. Unsupervised intrinsic calibration of depth sensors via slam. In *Proceedings of Robotics: Science and Systems*, Berlin, Germany, June 2013.

[UEHR12] A. Uckermann, C. Elbrechter, R. Haschke, and H. Ritter. 3d scene segmentation for autonomous robot grasping. In *IROS*, pages 1734–1740, Oct 2012.

[WSF18] R. Weilharter, F. Schenk, and F. Fraundorfer. Globally consistent dense real-time 3d reconstruction from rgbd data. In *Proceedings of the OAGM Workshop 2018*, pages 121–127, 2018.

[Zha00] Z. Zhang. A flexible new technique for camera calibration. *PAMI*, 22(11):1330–1334, 2000.

[ZZ11] C. Zhang and Z. Zhang. Calibration between depth and color sensors for commodity depth cameras. In *2011 IEEE Int. Conference on Multimedia and Expo*, pages 1–6, 2011.