Self Archive Version
Cite this article as: Fuersattel, P., Placht, S., Maier, A. Riess, C - "Geometric Primitve Refinement for Structured Light Cameras". Machine Vision and Applications (2018) 29: 313.

Geometric Primitive Refinement for Structured Light Cameras

Peter Fuersattel · Simon Placht · Andreas Maier · Christian Riess

Received: date / Accepted: date

Abstract 3-D camera systems are useful sensors for several higher level vision tasks like navigation, environment mapping or dimensioning. However, the raw 3-D data is for many algorithms not the best representation. Instead, many methods rely on a more abstract scene description, where the scene is represented as a collection of geometric primitives like planes, spheres or even more complex models. These primitives are commonly estimated on individual point measurements which are directly affected by the measurement errors of the sensor.

This paper proposes a method for refining the parameters of geometric primitives for structured light cameras with spatially varying patterns. In contrast to fitting the model to a set of 3-D point measurements, we propose to use all information that belongs to a particular object simultaneously to directly fit the model to the image, without the detour of calculating disparities.

To this end, we propose a novel calibration procedure which recovers the unknown internal parameters of the range sensors and reconstructs the unknown projected pattern. This is particularly necessary for consumer structured light sensors whose internals are not available to the user. After calibration, a coarse model fit is considerably refined by comparing the observed structured light dot pattern with a predicted virtual view of the projected virtual pattern.

The calibration and the refinement methods are evaluated on three geometric primitives: planes, spheres and cuboids. The orientations of the plane normals are improved by more than 60 %, and plane distances by more than 30 % compared to the baseline. Furthermore, the initial parameters of spheres and cuboids are refined by more than 50 % and 30 %. The method also operates robustly on highly textured plane segments, and at ranges that have not been considered during calibration.

Keywords Structured light · Range imaging · Geometric Primitives

1 Introduction

Accurate scene information is a critical component for numerous vision-based applications, such as dimensioning, robot navigation, floor detection, SLAM algorithms or generally any map building task. These applications greatly benefit from the recent proliferation of low-cost cameras that capture 2-D images with additional depth information [20, 4, 22]. Well-known depth cameras are based on Time-of-Flight, stereo vision, and structured light. These cameras capture dense, ordered point clouds of the scene at high frame rates with reasonable accuracy.

Higher level vision tasks like robot navigation or mapping oftentimes compute an intermediate representation of the data. The segmentation and accurate detection of planes is a common task. For example, in the context of robotics, several algorithms for synchronous localization and mapping (SLAM) operate on 3-D planes

This work was supported in part by the Research Training Group 1773 "Heterogeneous Image Systems", funded by the German Research Foundation (DFG), and in part by the Erlangen Graduate School in Advanced Optical Technologies (SAOT) by the German Research Foundation (DFG) in the framework of the excellence initiative.

Peter Fuersattel[1], Andreas Maier[1], Christian Riess[1]
{first}.{last}@fau.de

Simon Placht[2]
simon.placht@metrilus.de

[1] Friedrich-Alexander University Erlangen-Nuremberg
[2] Metrilus GmbH, Erlangen

that have been detected and segmented from the raw sensory information [22,23,4,3]. Plane representations have also bee used for the calibration of range sensors to color sensors [11,9]. The extrapolation of missing measurements at object boundaries as demonstrated by Moerwald *et al.* is another example for the usefulness of abstract representations of objects in the scene [17]. Fitting models to specific regions in the captured point cloud is also useful in a dimensioning context, e.g. if the size of box-like objects needs to be measured. Naturally this applies to all types of models which can be fit to a list of 3-D measurements, for example in a least-squares fashion or via RANSAC.

Abstract representations are popular because they address three challenges of working with 3-D data:

1. Higher level vision algorithms oftentimes do not operate on the raw sensory measurements, but require a model of the environment anyways. Furthermore, abstract representations simplify many common calculations like determining the size of objects or computing the intersections of multiple objects.
2. The additional depth dimension increases the amount of data to be processed (which may be a bottleneck, especially in real-time applications like SLAM). By computing a simplified geometric representation, the data rate can be considerably reduced.
3. Current range cameras provide only limited accuracy. For structured light and Time-of-Flight cameras, these errors range from several millimeters up to multiple centimeters [14,8]. Fitting models of geometric primitives to segments of the point cloud can reduce the impact of noise on subsequent calculations, especially if the measurement errors stem from temporal noise sources.

In this work, we present a highly accurate method for fitting a 3-D model to a segment in a point cloud using a consumer structured light camera. Well-known examples of this class of sensors are the Microsoft Kinect, Asus Xtion and the more recently released Orbbec Astra[1]. These cameras use a single infrared camera to capture a projector-generated infrared dot pattern. The distortion of the dot pattern is used to compute depth. One advantage of structured light sensors over stereo systems is that they provide dense depth data on surfaces with homogeneous texture. Typically, structured light cameras calculate distances via block matching of small patches. In contrast to that, we propose a method that exploits the complete 2-D image information that represents an object to find an accurate parametric object for this object.

[1] `https://orbbec3d.com`

Our method is based on the idea that, for any parametric model in 3-D, one can simulate the projection and observation of the emitted dot pattern. We propose to create a virtual view of the object based on a model that defines the object. If the model describes the observed object perfectly, then the virtual image and the observed image will be identical. Thus, by measuring the similarity between these two images and adjusting the model parameters such that the similarity increases, one can optimize for more accurate parameters.

A fundamental requirement for creating such virtual views is the knowledge of all internal parameters that describe the structured light sensor: a model of the projector, the spatial relation between projector and camera and the emitted dot pattern. The challenge with off-the-shelf structured light sensors, like those mentioned above, is that the user typically has no access to these parameters. To tackle this problem, we propose a novel calibration method, which is able to recover the intrinsic parameters of the projector, the spatial relation between the projector and the camera, and to reconstruct the unknown emitted dot pattern from two or more images.

The contributions of this work consist of three parts:

1. We present a novel calibration technique to estimate pinhole model parameters for the projector, the extrinsic parameters of camera and projector, and the unknown dot pattern.
2. We present a new refinement approach for geometric primitives for structured light cameras. In contrast to the regular pixel-wise disparity calculation, we propose to use all pixels that belong to the object simultaneously. This image information is combined with constraints posed by the model's shape to refine the initial parameters.
3. We show that initial estimates for three different geometric primitives, namely planes, spheres and cuboids, can be refined significantly with the proposed method. In the case of planes, we show that the angular and distance accuracy of plane models are increased by more than 60 % and 30 % compared to the baseline. Initial estimates of spheres can be improved by up to 50 %. Furthermore, even complex models, like cuboids can be refined by up to 30 %.

In Section 2, we present related work. The notation used in this work is explained in Section 3. Section 4 contains a detailed description on the calibration method. In Section 5 we give information on how to obtain an initialization for an object which can be refined with the method described in Section 6. The performance of the method is evaluated in Section 7. We summarize and conclude our findings in Section 8.

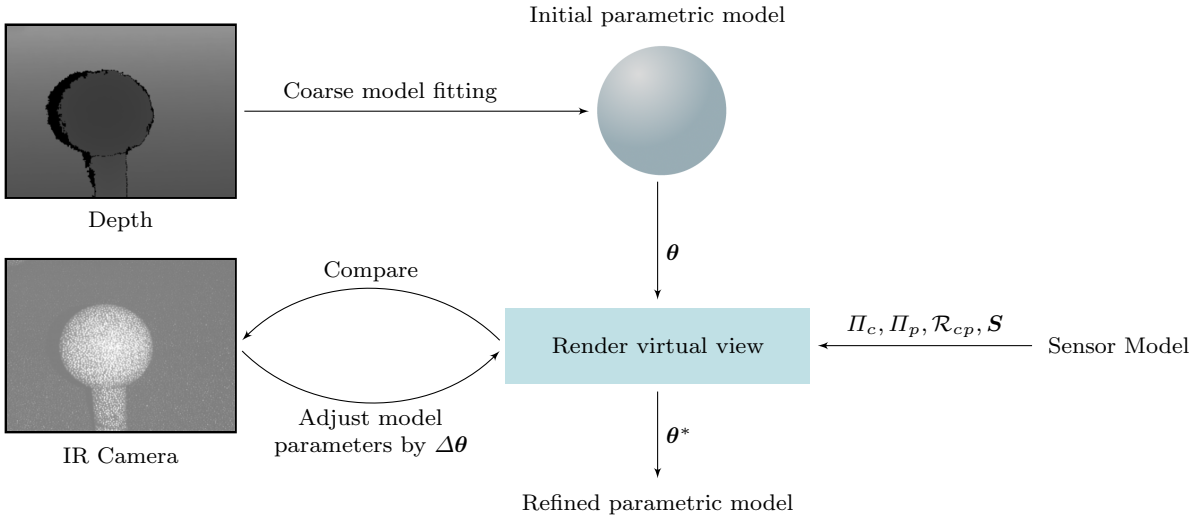


Fig. 1: Geometric Primitive Refinement. Parameters $\boldsymbol{\theta}$ of an initial parametric model are refined in an iterative process. By creating virtual views of the object and comparing it to the observed image, $\boldsymbol{\theta}$ is adjusted such that the similarity increases and an optimal parameter set $\boldsymbol{\theta}^*$ can be obtained. The sensor model, that is summarized with the parameters $\Pi_c, \Pi_p, \mathcal{R}_{cp}, \boldsymbol{S}$, is obtained with the calibration method described in Section 4.

2 Related Work

A structured light camera system is defined by a number of parameters: the intrinsic parameters of the camera and the projector, their spatial relation and finally the projected pattern. In custom setups, these parameters can either be directly controlled by the user, or obtained via calibration.

Moreno and Taubin published a calibration toolbox for such systems to find the spatial relation by using a calibration target [16]. Ye and Song recently proposed a method for refining an initial calibration result based on control points in 3-D space [27].

Yamazaki *et al.* presented a method for estimating the spatial relation between the camera and the projector as well as the intrinsic parameters of both devices [26]. The method does not require a calibration target but is consequently not able to determine the scale. Another target-less approach for setups with at least two cameras and a projector have been proposed by Bird and Papanikolopoulos [2].

The above-mentioned methods are designed for custom structured light systems, or setups which allow the user to control the projected patterns. However, these methods are not applicable for off-the-shelf sensors, like the Microsoft Kinect or Orbbec Astra, as these cameras do not provide access to their internal parameters. In this work we relax these requirements and present a method which is capable of calibrating a generic structured light camera with an unknown pattern.

In principle, this work is closely related to the primitive detection and segmentation in point clouds. However, a significant difference lies in the focus. In our work, the objective is to obtain an estimate for the location and the parameters that define an object as accurate as possible. In contrast to that, the primary objective of the detection methods lies in a fast, coarse segmentation, typically with a strong focus on real-time applications.

There exists a large body of work on geometric primitive detection for point clouds that have been captured with, for example, a structured light camera. A review on point cloud segmentation can be found in [18]. Poppinga *et al.* segment a scene into planes by performing region growing on the 3-D point cloud [20]. The authors present an efficient method to calculate least squares fits of planes to the points. The authors of [12] extend this method by a multilateral filtering step to increase robustness against measurement noise. The authors also propose an alternative, less accurate segmentation method to reduce computational complexity. Trevor *et al.* reduce the number of plane fits by segmenting the image first and only fitting planes to regions which fulfill a size criterion [24]. Mörwald *et al.* identify regions that can be modeled with B-Splines [17]. This method jointly uses range and color information to calculate missing range information in the scene, which may happen in pixels where the correspondence search fails. Georgiev *et al.* propose to first search for line segments in the point cloud to reduce the search space [10]. In a second step, these line segments are analyzed and combined to obtain the plane segments in the scene. In

[6] a real-time capable, graph-based approach is presented. Graph nodes represent non-overlapping image regions. These nodes are analyzed separately for planarity and merged if they belong to the same segment. Then, region growing is used to determine the exact pixels which belong to the individual models. Borrmann *et al.* present a new Hough transform accumulator for 3-D plane detection and compare it to other variants of the Hough transform [5].

The plane estimate in the methods above is relatively straightforward. It oftentimes consists of a least-squares fit to the 3-D point cloud using singular value decomposition or variants thereof. However, such a fast approach is quite prone to outliers or other estimation errors. All in all, applications can benefit from our method if high accuracy is required, but runtime requirements are somewhat less stringent.

One important difference to the methods above lies in the fact that our approach does not operate on the 3-D points, but instead directly on the dot pattern of the structured light sensor. This allows to jointly fit several dots of the pattern to a geometric primitive, and hence to avoid individual errors in the stereo matching process.

The idea to directly operate on the dot pattern has also been used in other applications. Fanello *et al.* pose the correspondence search problem as a classification problem, and solve it for each pixel individually with random forests [21]. The projector pattern is obtained from a calibration and training phase. One notable difference of this work to ours is that we do not operate on each dot individually. Instead, we include non-local cues by jointly optimizing for all dots on one surface. McIlroy *et al.* also operate directly on the dot pattern [15]. The authors propose a low-cost tracker based on the Kinect camera. The authors demonstrate how the projection of the dot pattern onto a fixed multi-planar surface can be used to determine the pose of the projector. We use a similar idea, but we solve for the projection surface using a fixed camera-projector setup.

Abstract representations of geometric primitives, e.g. planes, have been used by a number of works on higher-level vision tasks. Methods of this type can potentially benefit from the high accuracy of the presented algorithm. An extension of the work by Holz *et al.* [12] uses a map of planes for localization or place recognition [7]. In [25], planes are used as features to create a map of the environment. These planes are used to effectively reduce the computational complexity of the mapping process and to increase the accuracy of the map and localization tasks. Birk *et al.* demonstrate how planar surface patches can be used to estimate the robot motion efficiently via a deterministic, non-iterative method [3]. Biswas *et al.* present a RANSAC-based plane segmentation for obstacle avoidance and navigation, which relies only on depth information [4]. Taguchi *et al.* show that the use of planes is important to achieve high frame rates for 3D reconstruction with the Kinect sensor [23]. Salas *et al.* present a SLAM system that recovers the boundaries of planes in the scene over time [22].

3 Notation

We denote images by bold capital letters, for example image $\boldsymbol{I}$. Bold lowercase letters denote vectors, for example $\boldsymbol{x} = (X, Y, Z)$. If required, a superscript specifies the coordinate system of the vector, for example, $\boldsymbol{x}^{(c)}$ denotes a point as seen from camera c.

Geometric primitives, for example planes, spheres or cylinders, are represented as parametric 3-D models. All parameters that describe such a model are summarized in a parameter vector $\boldsymbol{\theta}$. In the case of a plane, we use the Hesse normal form to describe a plane with four parameters: three values to represent the unit normal, and one value to represent the distance between the origin and the plane.

A translation vector and a rotation matrix form the rigid body transformation $\mathcal{R}_{cp}$ that, for example, relates the coordinate systems of the camera and the projector. We use Π to describe a projection from 3-D space to the 2-D image domain. This projection is implemented as a pinhole camera model, or a pinhole camera model combined with a lens distortion model. $\Pi_{\boldsymbol{\theta}}^{-1}$ describes an inverse projection. In this work, inverse projections are implemented as the intersection of a ray with a parametric 3-D model defined by $\boldsymbol{\theta}$.

4 Calibration of the Structured Light Camera and Reconstruction of the Dot Pattern

The complete calibration process is illustrated in Figure 2. We first describe the acquisition of a set of calibration images in Section 4.1. In Section 4.2, we demonstrate how a coarse initialization of the sensor's extrinsic parameters can be approximated. Next, we create an initial reconstruction of the dot pattern as described in Section 4.3. Hereafter, we show how the initial parameters and the initial dot pattern can be refined in a non-linear optimization scheme based on multiple calibration images. The refinement is described in Section 4.5.

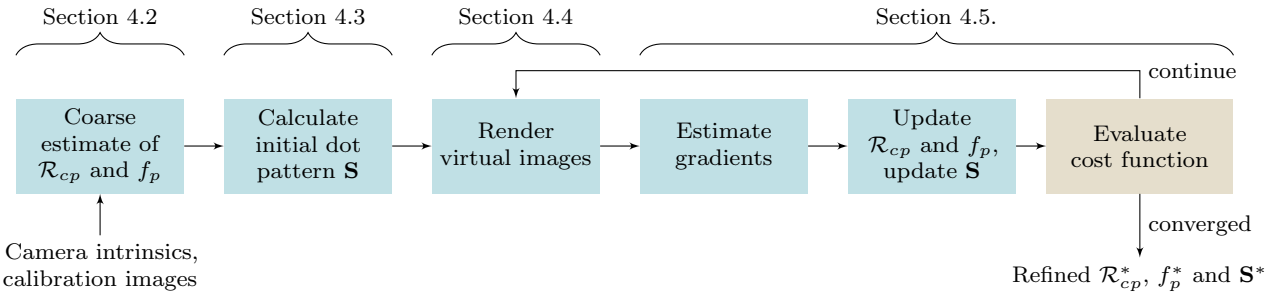


Fig. 2: Calibration process. Starting from a set of calibration images and camera intrinsics, the extrinsic parameters, the intrinsic parameters of the projector and the unknown dot pattern are estimated.

4.1 Calibration Data and Reference Distance Information

In this section we describe the calibration data and how it can be acquired.

The calibration procedure requires at least two images, each consisting of a dot pattern image and a corresponding point cloud. The images need to show multiple planar scenes. The planes may be parallel but not coplanar.

A homogeneous white wall captured at different distances will ensure the best possible estimation of the internal parameters and of the dot pattern. Generally, more images captured at different distances will result in better estimates.

The calibration protocol requires that the position and orientation of an imaged plane is known. Theoretically, one can use the distance information of the point cloud to estimate the plane model. However more accurate plane models can be obtained with calibration patterns. The dot pattern is captured best in a two-step process that consists of estimating the plane with a calibration pattern, and removing the calibration pattern before capturing the dot pattern.

In preliminary experiments, we have compared different calibration patterns with each other, including checkerboards with ROCHADE [19] and RUNETags [1] with the implementation provided by the authors. In these experiments, checkerboards were detected over a wider range of distances and more accurately than RUNETags, which is why we use checkerboards.

4.2 Coarse Initialization of the Intrinsic and Extrinsic Parameters

A first coarse calibration is obtained in two steps. First, we estimate the relative position between projector and camera $\mathcal{R}_{pc}$. In the second step, we calculate the parameters of a pinhole camera model that represents the projector.

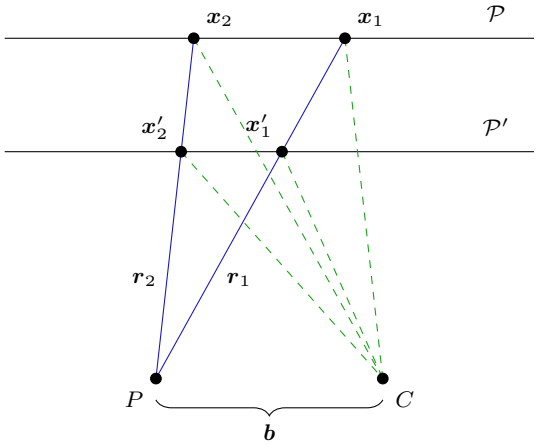


Fig. 3: Initial baseline estimation. The rays $\boldsymbol{r}_i$ generated from two corresponding point pairs $\boldsymbol{x}_i$ and $\boldsymbol{x}'_i$ are used to calculate the initial baseline $\boldsymbol{b}$.

We slightly simplify the estimation of the relative position between camera and projector. Since the principal rays of camera and projector are parallel in almost all off-the-shelf structured light cameras, the spatial relation $\mathcal{R}_{pc}$ of camera and projector reduces to a translation $\boldsymbol{b}$. We call $\boldsymbol{b}$ the baseline.

The estimation of the baseline is visualized in Figure 3. Here, the projector emits rays $\boldsymbol{r}_i$, that eventually hit a plane. The illustration shows that the same ray $\boldsymbol{r}_i$ can also be calculated from two corresponding points $\boldsymbol{x}_i$ and $\boldsymbol{x}'_i$ which lie on two different planes $\mathcal{P}$ and $\mathcal{P}'$. Thus, the center of projection of the projector P, and consequently the baseline $\boldsymbol{b}$, can be computed by finding the intersection of two or more rays.

For this step the intrinsic parameters of the camera Π_c are assumed to be known or that they can be calculated in a separate calibration step [28].

The derivation requires that the positions of the two planes $\mathcal{P}$ and $\mathcal{P}'$ are known. Their parameters can be

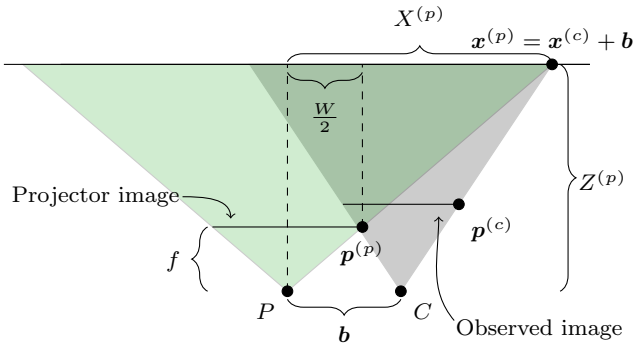


Fig. 4: Focal length initialization. The projector's focal length f_p is computed from the known camera intrinsics, a known plane and the initial baseline $\boldsymbol{b}$. Shaded areas visualize the two fields of view for camera C and projector P.

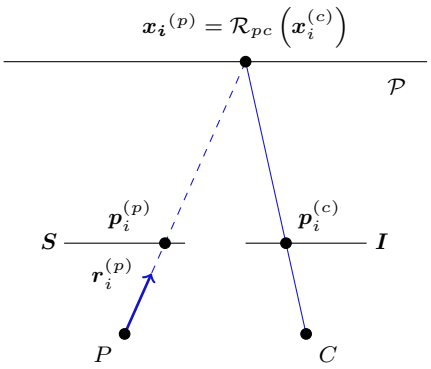


Fig. 5: Virtual image generation. The ray $\boldsymbol{r}_i^{(p)}$ for the point $\boldsymbol{p}_i^{(p)}$ is intersected with the plane $\mathcal{P}$ to obtain $\boldsymbol{x}_i^{(p)}$. This point is transformed to the coordinate system of the camera, and finally projected with Π_c to get $\boldsymbol{p}_i^{(c)}$.

estimated from the structured light depth measurements or (preferred) from a calibration pattern attached to their surfaces.

Point correspondences $(\boldsymbol{x}_i, \boldsymbol{x}_i')$ are obtained via block matching. If camera and projector positions differ only in the x-coordinate, the block search can be constrained to the x-axis.

This approach can be extended easily to more than two rays to increase the accuracy of the estimate. In practice, five to ten rays were sufficient. Best initializations are obtained if the correspondences are well spread in the image, for example equally among the four quadrants of the image.

The approximation of the baseline allows the estimation of the pinhole model parameters of the projector. We assume that the projector has an ideal lens and that the principal point is at the image center. Hence, finding the projector model reduces to estimating the focal length.

We calculate the focal length such that a pixel that is observed in a corner of the camera image is also located in the same corner of the projector's image. Figure 4 illustrates this scenario. In order to capture the complete observed pattern, we choose the corner $\boldsymbol{p}^{(c)}$ that corresponds to the projection of a 3-D point $\boldsymbol{x}^{(c)} = \left(X^{(c)}, Y^{(c)}, Z^{(c)}\right)$ that maximizes the distance to the projector. With $\boldsymbol{x}^{(p)} = \boldsymbol{x}^{(c)} + \boldsymbol{b}$ and its respective projection $\boldsymbol{p}^{(p)}$, the focal length can be computed directly with the pinhole camera model:

$$f_p = \left(W - \frac{W}{2}\right) \frac{Z^{(p)}}{X^{(p)}} \quad . \tag{1}$$

Computing f_p requires the width W of the fixed image to be known. In practice, W should be set to a value equal or larger than the width of the camera image.

The resulting focal length directly depends on the distance between the observed plane and the camera, with closer planes resulting in a smaller focal length. Therefore, we use the calibration image with the smallest plane-camera distance for calculating f_p in order to include the largest possible section of the projected dot pattern.

4.3 Dot Pattern Reconstruction

In this section, we describe how the static image, that the emitter projects into the scene, can be reconstructed. We call the reconstructed image the virtual dot pattern $\boldsymbol{S}$. Below we describe how the intensities for all pixels in $\boldsymbol{S}$ can be obtained from corresponding pixels in $\boldsymbol{I}$ by using the previously calculated intrinsic and extrinsic parameters.

The reconstruction process is visualized in Figure 5. Here, the intensity for a point $\boldsymbol{p}_i^{(p)}$ is calculated from its corresponding point $\boldsymbol{p}_i^{(c)}$ in $\boldsymbol{I}$. The coordinates $\boldsymbol{p}_i^{(c)}$ are obtained by intersecting $\boldsymbol{r}_i$ with the plane, transforming the intersection to the camera's coordinate system, and finally by projecting it onto the image plane. The intensity in the virtual dot pattern at $\boldsymbol{p}_i^{(p)}$ is now interpolated from $\boldsymbol{I}$. This is expressed by interp$(\cdot)$. Mathematically, the intensity lookup for a single pixel of $\boldsymbol{S}$ is given by

$$\begin{aligned} \boldsymbol{S}\left(\boldsymbol{p}_i^{(p)}, \boldsymbol{\theta}\right) &= \text{interp}\left(\boldsymbol{I}, \boldsymbol{p}_i^{(c)}\right) \\ &= \text{interp}\left(\boldsymbol{I}, \Pi_c\left(\mathcal{R}_{pc}\left(\Pi_{p,\boldsymbol{\theta}}^{-1}\left(\boldsymbol{p}_i^{(p)}\right)\right)\right)\right) \end{aligned} \tag{2}$$

The estimation of the initial dot pattern can be extended to use multiple images, analogously as described in Section 4.2. However, similar as in Section 4.2, it

turned out that a coarse estimate of the pattern is sufficient if an additional refinement step is performed.

4.4 Virtual Image Generation

With known dot pattern, extrinsic and intrinsic parameters, it is possible to generate a virtual image $\boldsymbol{V}$ of a parametric model. In the context of the calibration, the goal is to generate a virtual image of a plane in the calibration scene. To this end, we use the inverse reconstruction pipeline given in Equation 2, with $\mathcal{R}_{cp}$ being the inverse of $\mathcal{R}_{pc}$. The plane is defined by $\boldsymbol{\theta}$. By using the previously calculated virtual dot pattern $\boldsymbol{S}$ as source for the lookup, pixel values for all coordinates $\boldsymbol{p}_i^{(c)}$ can be obtained:

$$\boldsymbol{V}\left(\boldsymbol{p}_i^{(p)}, \boldsymbol{\theta}\right) = \text{interp}\left(\boldsymbol{S}, \Pi_p\left(\mathcal{R}_{cp}\left(\Pi_{c,\boldsymbol{\theta}}^{-1}\left(\boldsymbol{p}_i^{(c)}\right)\right)\right)\right) \quad . \tag{3}$$

Note that the image generation pipeline expressed in Equation 3 allows the adjustment of all parameters which describe either the scene or the structured light sensor. This property will be exploited for refining the sensor parameters and the initial virtual dot pattern in Section 4.5. Also note that although the calibration operates on planes, the application of the refinement can operate on arbitrary geometric structures that can be formulated as a parametric model. The sole requirement is the implementation of a method which calculates the intersection of a ray with the respective model.

The virtual image generation can be parallelized as all pixel values can be computed independently. Depending on the used geometric primitive and camera properties additional simplifications may become possible. For example, if planar regions are refined and lens distortion is negligible, then the mapping between dot pattern and virtual image can be represented by a homography.

4.5 Optimization of Intrinsic Parameters and the Dot Pattern

The coarse calibration is based only on a small number of point correspondences. During refinement, the complete image information of M calibration images is utilized to obtain more accurate intrinsic and extrinsic parameters as well as a more accurate reconstruction of the dot pattern. We assume, that if the true sensor parameters are known, then it is possible to create virtual view of the scene that is identical to the observed camera image.

We use gradient descent to refine the rigid body transformation $\mathcal{R}_{cp}$, focal length f_p of the projector and the dot pattern $\boldsymbol{S}$ as seen from the projector. In each iteration, two steps are performed: first, the refinement of the extrinsic parameters and the focal length, and second, an update of the dot pattern. This iteration is visualized in Figure 2.

Gradient descent requires a cost function to measure the dissimilarity of two images. To this end, we calculate the mean squared differences between an observed image $\boldsymbol{I}$ and a virtual image $\boldsymbol{V}$, which itself depends on a particular f_p and $\mathcal{R}_{cp}$.

To increase the comparability, each pixel is normalized by the mean value of its neighborhood. This can be implemented efficiently using integral images. The cost function for a normalized image $\boldsymbol{I'}$ and its corresponding normalized virtual image $\boldsymbol{V'}$ is given by

$$e_{\mathcal{R}_{cp},f_p}(\boldsymbol{I}, \boldsymbol{V}(\boldsymbol{\theta})) = \frac{1}{N}\sum_i^N \left(\boldsymbol{I}'_i - \boldsymbol{V}'_i\left(\boldsymbol{\theta}\right)\right)^2 \quad . \tag{4}$$

In this equation, N denotes the number of pixels. We obtain the gradients numerically by altering the current parameters individually by small deltas. Note that the pixel-wise differences and the required normalization prevent the usage of analytic derivatives.

Once the gradients are known, the current parameter set is adjusted and used to estimate a new dot pattern.

From each of the M calibration images, a dot pattern $\boldsymbol{S}_j$ is derived as outlined in Section 4.3. The updated dot pattern $\boldsymbol{S'}$ is computed as the pixel-wise median image, i.e.,

$$\boldsymbol{S'} = \text{median}\left([\boldsymbol{S}_1, \boldsymbol{S}_2, \ldots, \boldsymbol{S}_M]\right) \quad . \tag{5}$$

The median filter limits the impact of sensor noise on the dot pattern reconstruction. During the next iteration, $\boldsymbol{S'}$ will serve as the current dot pattern $\boldsymbol{S}$.

The optimal parameter set, consisting of $\mathcal{R}^*_{cp}$, f^*_p and $\boldsymbol{S}^*$ can be obtained by minimizing Equation 6. The function minimizes the dissimilarity between the M normalized virtual views $\boldsymbol{V}'_j$ of known planes with their corresponding normalized observations $\boldsymbol{I}'_j$.

$$\left(\mathcal{R}^*_{cp}, f^*_p, \boldsymbol{S}^*\right) = \underset{\mathcal{R}_{cp},f_p,\boldsymbol{S}}{\text{argmin}} \left(\sum_j^M e_{\mathcal{R}_{cp},f_p}\left(\boldsymbol{I}'_j, \boldsymbol{V}'_j(\boldsymbol{\theta}_j)\right)\right) \tag{6}$$

With the aforementioned method, the projector parameters, baseline, rotation and dot patterns can be estimated and saved for future use. Recalibration is only required if the camera/projector setup changes which typically does not occur during operation.

5 Geometric Primitive Detection

The refinement method requires an initial estimate of the object's model whose parameters should be refined. There are various approaches for obtaining such an initialization, for example a complete decomposition of the point cloud into geometric primitives, or a semi-automatic segmentation which is based on seed values or a region of interest. In Section 2, we listed example works for finding different primitives like planes [20,12, 24] or B-spline models [17].

In this work we do not put any constraints on how the initial parameters for a particular model are obtained. Instead, we only require a parameter vector $\boldsymbol{\theta}$ that describes the model analytically. In this work, three different models are investigated: planes, cuboids and spheres. A general limitation of the proposed method is that it assumes that the geometric model is a sufficiently accurate representation of a pre-segmented area of the scene. Situations where, for example, parts of the primitive are occluded by another object, have to be addressed in a separate processing step. In our implementation, we used masks to ignore such outlier regions.

6 Geometric Primitive Refinement

The core idea of the refinement algorithm is to create a virtual view of the object. If all parameters that describe the object are either known or correctly estimated, the virtual view matches the observed image. Hence, the objective is to estimate the unknown parameters in a way that minimizes the deviations between the virtual view and the observed image.

For generating a virtual view, a complete model of the imaging system is required. In this section, we assume that the intrinsic parameters of the camera and the projector, their spatial relation and the projected dot pattern have already been calculated with the calibration procedure described in Section 4.

The parameters of an object in the scene, for example a plane, are summarized in the parameter vector $\boldsymbol{\theta}$. In this context, we assume that there exists an initial estimate of the object which is obtained by some initial detection algorithm. In this section, we present how these parameters can be further improved, such that their virtual view matches the observed image more accurately.

The optimal model parameters $\boldsymbol{\theta}^*$ are obtained via nonlinear optimization. Similar as during calibration, we use a gradient descent approach to find the optimum. A metric e is used to measure the dissimilarity between the observed image $\boldsymbol{I}$ of a camera and a virtual image $\boldsymbol{V}(\boldsymbol{\theta})$. Here, we reuse a slightly altered version of the error metric defined in Equation 4. In the optimization phase of the calibration procedure, this error metric is used to find $\mathcal{R}^*_{cp}$ and f^*_p based on fixed plane model parameters. During refinement of object parameters, we fix the camera's system parameters and vary the object parameters $\boldsymbol{\theta}$. Therefore, the dissimilarity in terms of the mean squared error for a given $\boldsymbol{\theta}$ is defined as:

$$e_{\mathcal{R}^*_{cp},f^*_p}(\boldsymbol{I},\boldsymbol{V}(\boldsymbol{\theta})) = \frac{1}{N}\sum_i^N \left(\boldsymbol{I}'_i - \boldsymbol{V}'_i(\boldsymbol{\theta})\right)^2 \quad . \tag{7}$$

The goal of the refinement step is to find parameters $\boldsymbol{\theta}^*$ that maximize the similarity (or minimize the dissimilarity, respectively) between $\boldsymbol{I}$ and $\boldsymbol{V}(\mathcal{R}_{cp},\boldsymbol{\theta})$. More precisely, we seek

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left(e_{\mathcal{R}^*_{cp},f^*_p}(\boldsymbol{I}',\boldsymbol{V}'(\boldsymbol{\theta}))\right) \quad . \tag{8}$$

Several effects, like scene texture, varying surface reflectance, or the angle of incidence at lateral surfaces may negatively affect the optimization in Equation 8. Thus, to increase the robustness of the optimization, we first normalize each pixel by dividing it by the mean value of its neighborhood. This can be implemented efficiently using integral images. Let $\boldsymbol{I}'$ and $\boldsymbol{V}'$ be the normalized images of $\boldsymbol{I}$ and $\boldsymbol{V}$. Then, Equation 8 becomes

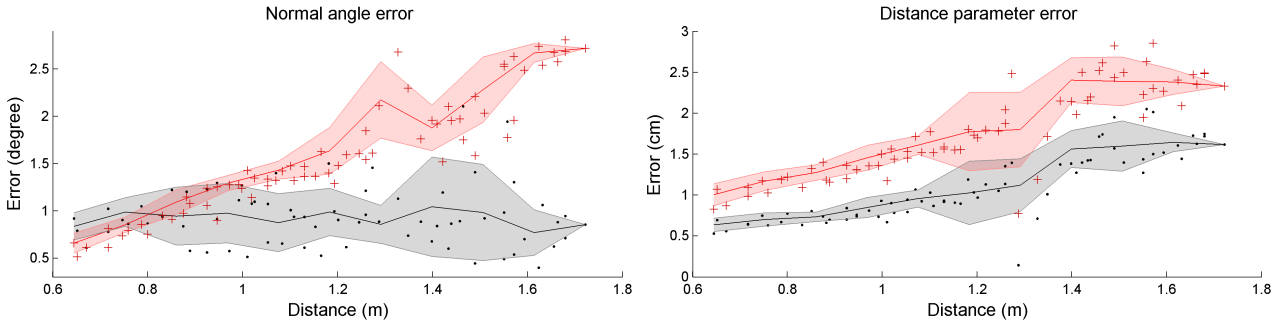
$$\begin{aligned}\boldsymbol{\theta}^* &= \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left(e_{\mathcal{R}^*_{cp},f^*_p}(\boldsymbol{I}',\boldsymbol{V}'(\boldsymbol{\theta}))\right) \\ &= \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left(\frac{1}{N}\sum_i^N (\boldsymbol{I}'_i - \boldsymbol{V}'_i(\boldsymbol{\theta}))^2\right) \quad ,\end{aligned} \tag{9}$$

which can be optimized via gradient descent. We observed that a number of outliers can be expected at the border of a segment. Thus, applying an additional erosion to the segment boundaries increases the overall accuracy of the estimate.

7 Evaluation

This section presents a thorough evaluation of the proposed method. We evaluate three different aspects of the method: its accuracy with respect to varying scenes, the impact of the calibration data and its applicability to different geometric primitives.

We use planar segments in our accuracy study, as calibration patterns allow a highly accurate estimation of the reference planes for quantitative evaluation. In this study, we compare our method to the results of a least-squares fitting method for planes. In these experiments, we investigate the influence of the distance between camera and plane, the impact of the size of the refined plane, the robustness with respect to texture and the



(a) Angle error (b) Distance error

Fig. 6: Normal angle and distance parameter errors. The red plot shows the error before refinement, black the error after refinement. Crosses and dots show the individual results. The shaded area reflects the standard deviation. With the proposed method, the plane parameters can be refined effectively.

impact of the maximum number of iterations for the refinement.

The second part of the evaluation contains a study of different aspects of the calibration procedure. In this context, we evaluate the impact of the number of calibration images that are used for calibrating the system. Additionally, we investigate how well the calibration results generalize with respect to distances which have not been covered during the calibration.

Finally, we evaluate the proposed method for two other primitives to demonstrate its applicability for geometric models other than planes. In these examples, initial estimates of spheres and cuboids are refined such that both their position and model parameters become more accurate.

In this evaluation, an Orbbec Astra structured light camera is used. The camera comes with a generic factory calibration which is used for all camera models, independent of individual model variations. For achieving the highest possible accuracy, we have estimated the intrinsic parameters for the specific camera using [28] and [19].

7.1 Evaluation Data Sets and Experimental Setup

The system is calibrated with a set of 25 calibration images that show a white wall at distances ranging from 0.7 m to 1.7 m. The wall is approximately perpendicular to the optical axis. Accurate reference planes are calculated from checkerboard patterns which are removed before capturing the dot pattern.

Four different data sets with different scenes are used in this evaluation. The first dataset is captured similarly as the calibration data and consists of 70 images of a white wall at distances ranging from $\sim$ 0.65 m to $\sim$ 1.75 m. We denote this data set as the *wall* data set. The individual images are evenly spread throughout the evaluated distance range. For each image the ground truth plane is estimated with a checkerboard. Before capturing the dot pattern, the checkerboard is removed from the scene.

The *texture* data set consists of images that show six boards with different surface properties. This data set is used to investigate the robustness of the refinement method with respect to texture and surface properties.

Two additional data sets contain images of spheres and boxes respectively. The *ball* data set contains 40 images of a white coated ball with a radius of 10.5 cm from different perspectives and at different distances. The *cuboid* data set contains 40 images of a wooden box with 32.4 cm $\times$ 26.4 cm $\times$ 10 cm. The box is captured at various poses, such that different faces point towards the camera. The initial detection methods for spheres and planes are described in Section 7.8 and Section 7.9.

In the case of planes, we evaluate results by comparing the plane parameters with the ground truth planes before and after refinement. Note that the ground truth planes are calculated from checkerboards, and thus are accurate only up to some uncertainty. However, preliminary results showed that the uncertainties of the checkerboard-based plane estimates are at least 5 times smaller than the remaining errors of our refinement results.

We use the following notation to describe the results: the angle between the directions of the evaluated normal and ground truth normal is denoted by α. The difference between the distance parameter of the evaluated and ground truth plane is represented by Δd. The subscripts i and r refer to the initial and the refined values. During optimization, no information from the checkerboard is used.

The initial plane estimates are obtained with the probabilistic plane segmentation by [12]. This method

segments the scene into its individual planar segments. The segmentation method calculates a least squares fit for each planar segment, as well as a mask that delimits the particular segment. We use the proposed method to refine the initial estimate of the plane model. In this step, only valid pixels of the segment's mask are considered.

During optimization, observed and generated virtual images are normalized pixel-wise by their mean (window-size 11 pixels). By default, up to 60 iterations are performed during optimization with gradient descent. Optimization terminates early when the mean squared error for all optimized pixels changes by less than 10^{-5} during one iteration.

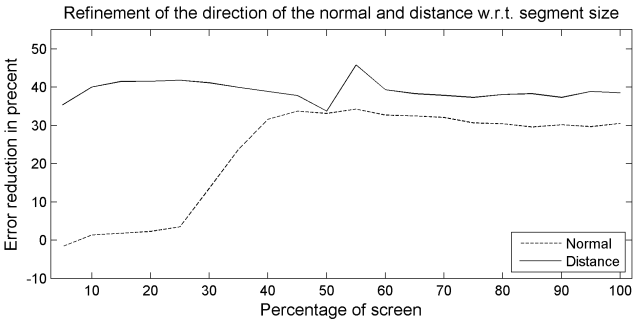


Fig. 7: Average error reduction for different segment sizes. This plot shows the accuracy gains for the normal and the distance for segments which cover a specific area of the total image.

7.2 Distance Between Plane and Camera

The first experiment aims at evaluating the impact of the distance between the camera and the plane. In this evaluation we analyze the deviations of the normal vector and the distance parameter for both the initialization and the refined model.

We use the *wall* data set for this experiment. The images are subdivided into 10 cm wide bins with respect to the distance from the center pixel. In the resulting Figure 6, we show the mean deviation from the reference normal as an angle and the difference between the distance parameters averaged over all planes that belong to a specific bin.

The results show that by using the proposed method, the error of the least squares plane estimate can be reduced across almost the complete range. The distance parameter is always improved by more than 30 %. In the close-range, our method fails to improve the normal direction and returns slightly worse normals than the initialization. We explain this with the saturation of the projected pattern. Whenever the camera gets very close to the wall, and thus to the limits of its operational range, the spatial extent of the individual dots of the pattern grows until they eventually merge. Once the camera has a certain distance (> 0.8 m) to the wall, the proposed method reliably improves the initial direction of the normal. As the distance increases, improvements by 60 % and more are possible (> 1.6 m).

7.3 Influence of the Size of the Plane Segment

Another interesting characteristic is the relationship between accuracy and segment size. The proposed method exploits the information of all pixels that belong to a plane segment. Therefore, it can be expected that higher accuracies become possible if more pixels are used during refinement. We study this relationship with the images of the *wall* data set.

We iteratively increase the segment size by 5 % starting from 5 % of the complete image region. All segments are positioned in the image center. Figure 7 shows the average gains across all images. For the whole range of segment sizes, the accuracy of the distance parameter of the initial plane models can be improved by ≈ 40 %. Assuming a correct normal angle, a wrong distance parameter would evenly affect all evaluated pixels due to wrong disparity values. In contrast, the direction of the normal affects all pixels of the plane differently. This effect can be noticed best at the border regions of the segments, and will be even more distinct if these regions cover large portions of the image. This explains why the accuracy gains for the normal direction decrease for smaller segments. Segments with at least 40 % of the image size still allow improvements of ≈ 30 %, while it is barely possible to refine segments that cover less than 25 % of the image.

Another interesting observation can be made from this plot: for segments that cover 50 % to 60 % of the image region, the best accuracies can be computed. A possible explanation for this effect is inhomogeneous scene illumination of the projector. Following this argumentation, we think that the central 50 % to 60 % of the image is illuminated best, and thus offers more information for optimization than the outer image regions.

7.4 Maximum Number of Iterations

In the current, simple implementation the optimization terminates if the preset 60 iterations are reached or if the cost function does not change by more than 10^{-5}. These parameters need to be adjusted depending on the required refinement accuracy.

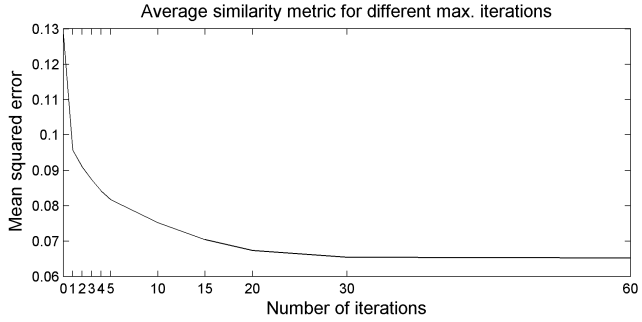


Fig. 8: Similarity metric vs. number of iterations. Only little accuracy gains are possible after more than 30 iterations.

Seg.	α_i	α_r		Δd_i	Δd_r	
1	1.68°	1.35°	−19 %	0.66 cm	0.35 cm	−47 %
2	0.72°	0.59°	−17 %	1.02 cm	0.41 cm	−60 %
3	1.37°	0.95°	−31 %	0.50 cm	−0.09 cm	−81 %
4	1.59°	1.20°	−25 %	0.87 cm	0.74 cm	−15 %
5	1.33°	1.01°	−24 %	0.85 cm	0.53 cm	−37 %
6	1.22°	0.88°	−28 %	0.75 cm	0.34 cm	−54 %
Avg.	1.32°	1.0°	−24 %	0.76 cm	0.41 cm	−46 %

Table 1: Normal and distance refinement for textured surfaces. $\Delta\alpha$ and Δd are the deviations from the reference normal and distance. ϕ is the ratio of the initial value and the refined value.

The following experiment investigates the decrease of the average cost function across the iterations with the *wall* data set. We evaluate the results after fixed numbers of iterations, starting with 0 (no refinement), to 60 iterations. The results of this experiment are shown in Figure 8. The figure shows how the similarity metric, in our implementation the mean squared difference of intensities, decreases with additional iterations. No large gains in accuracy can be observed after more than 30 iterations.

7.5 Performance on Textured Surfaces

In this section the robustness with respect to surface properties is evaluated. In this example, the *texture* data set is used. The scene, shown in Figure 9, consists of six different boards with different, challenging surfaces, which are placed on a table. Table 1 shows the results of this experiment. The reference plane of the planar segments is calculated with a checkerboard pattern ahead of the experiment.

For all planar segments the least-squares plane fit parameters, that are returned by the method of Holz *et al.* [12], can be refined with the proposed method. Segment 3 is, due to its strong texture, one of the more challenging examples in this evaluation. Even in this case the plane's normal direction and distance can be refined by 31 % and 81 % respectively. Additionally huge accuracy gains for the distance parameter are possible. Another challenging example is the highly specular black segment (segment 2) for which the deviation from the reference plane can be reduced by 17 % for the normal direction and 60 % for the distance parameter. On average, the direction of the normal can be improved by 24 % and the distance parameter by even 44 %.

7.6 Number of Calibration Images

The number of required calibration images directly affects the calibration effort. In this experiment, we order the calibration images (see Section 7.1) by their distance at the center pixel. From this list, N images are sampled uniformly to cover the complete calibration range. Each of the resulting sets is used as input to the proposed calibration method.

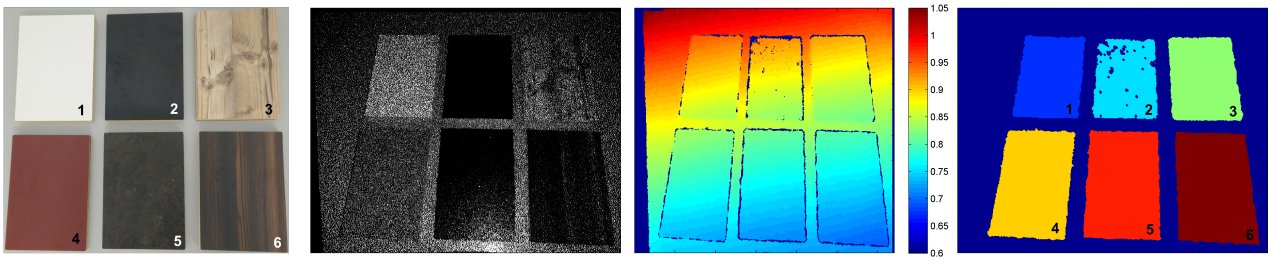
In this experiment, all images of the *wall* data set are refined with different calibrations. For each calibration, we calculate the average of all final cost function values to measure how well the final virtual image matches the observed image. Figure 10 shows the results of this experiment for $N = \{2, 3, 5, 10, 15, 25\}$ calibration images.

As expected, more calibration images will also increase the overall accuracy of the algorithm. However, it is also valid to trade accuracy for the amount of effort one would like to invest during calibration. Even with only a few images, for example less than ten, considerable improvements are possible. With a metric value of 0.129 for the initializations, gains of more than 40 % are already possible with only five calibration images and almost 50 % with 25 images.

7.7 Importance of Calibrating the Whole Range of Operation

Goal of this experiment is to investigate the possibility to get refined plane parameters at distances which are not covered during calibration. This is an important question as it also directly affects the calibration effort and the re-usability of calibration data.

For this evaluation, we reused the ordered list of calibration images of Section 7.6. From this list, we draw the first ten images for calibration, which results in a range from 0.70 m to 1 m. This calibration is used to refine the planes of the *wall* data set.



(a) Color photo (b) Infrared image (c) Z-Image (d) Segments

Fig. 9: Scene with six differently textured plane segments. Figure (a) shows a top-down color photography of the scene, (b) shows the infrared image captured by the camera, (c) shows the corresponding z-image (distances in meters) and (d) shows the plane segments which are determined during initialization. The results for the individual segments are given in Table 1. The segments 1,3,5 and 6 have a low specularity, segment 4 is moderately specular and segment 2 has high specularity.

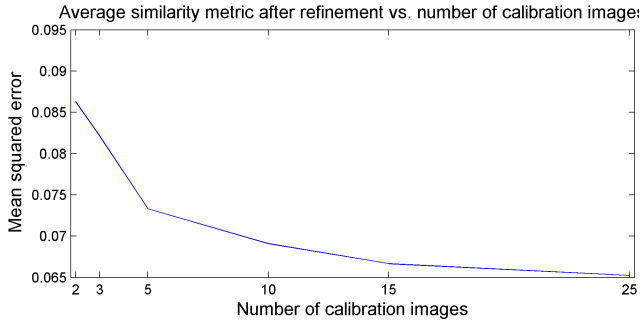


Fig. 10: Mean squared intensity differences for different numbers of calibration images. For reference: the initial mean squared error, without refinement, equals 0.129.

In Figure 11 we compare the residuals for two different set of calibration images: first all calibration images and, second the close range subset. The impact of the limited calibration range is not very severe, but can be observed nonetheless. The error of the normal direction increases for planes that are more distant than 1.2 m as shown in Figure 11a. The distance parameter (Figure 11b) is very stable, leading to barely noticeable differences between the two calibration sets.

For calibration with the close range subset, one could even argue that the distance parameter, gets even more accurate within the range of the calibration data. We hypothesize that the dot pattern, which has been derived during calibration, resembles the observed pattern in this range best. In contrast, the other dot pattern will likely reflect the observed patterns of the complete calibration range.

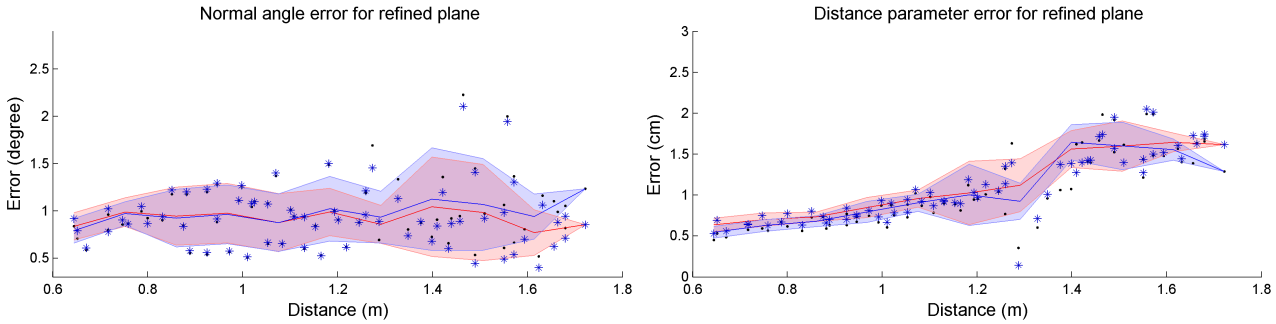
7.8 Evaluation on Spheres

This experiment evaluates the performance of the proposed method with respect to refining the position and the radius of spheres. Spheres are represented in terms of a center point and a radius, thus $\boldsymbol{\theta}$ consists of four unknowns: (c_X, c_Y, c_Z, r).

The initial spheres have been estimated with RANSAC with a distance threshold of 2.5 mm. Using smaller thresholds for RANSAC results in less robust initial estimates as the discretization error of the camera is larger than a millimeter. The 3-D points which were included in the estimation process have been obtained from a tightly selected rectangular region of interest around the sphere. Furthermore, all pixels which belong to the background plane have been identified with the plane segmentation method by Holz *et al.* [12] and excluded from the estimation process, thus resulting in a set of points with only very few outliers.

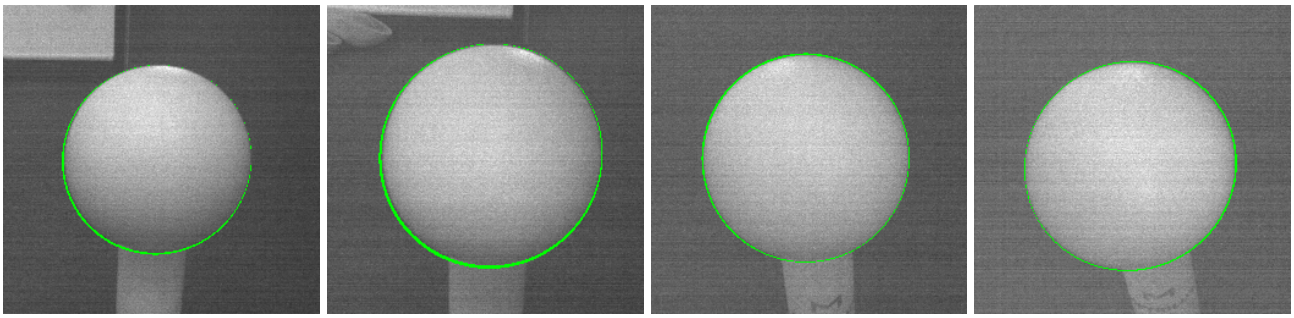
Table 2 reports the experimental results for the *spheres* data set. In the table, we list the absolute difference between the true radius of the sphere and the radius of the respective estimate. The average absolute error of the radius accounts for almost a millimeter for the RANSAC estimate. With the proposed refinement method, this error can be reduced by more than 50 % to 0.46 mm.

Figure 12 shows four examples taken from the data set. For these examples, the projector has been temporarily disabled for capturing an image without the overlaying dot pattern. In all cases, the initial RANSAC estimates are improved such that the resulting spheres match the observed spheres more closely. The noise, which is observable in both examples, stems from the low illumination in the infrared spectrum whenever the emitter is disabled.



(a) Angle error (b) Distance error

Fig. 11: Normal angle and distance parameter errors for a subset of calibration images. Only the 5 images with the smallest distance between camera and wall have been used for calibration. The results of the subset are shown in blue, the results of all calibration images (red) are shown for reference. Stars and dots are used to show individual results.



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

Fig. 12: Qualitative examples from the spheres data set. The green overlays represent the difference between the mask of the initial detection and the refined sphere. In all examples the initial sphere position and radius have been refined to a more accurate parameters which truly match the observed image.

Method	Mean Abs. Err. + Std.Dev
RANSAC	0.933 ± 0.760 mm
RANSAC + Proposed	0.455 ± 0.420 mm

Table 2: Errors for initial sphere detection and remaining errors after refinement.

7.9 Evaluation on Cuboids

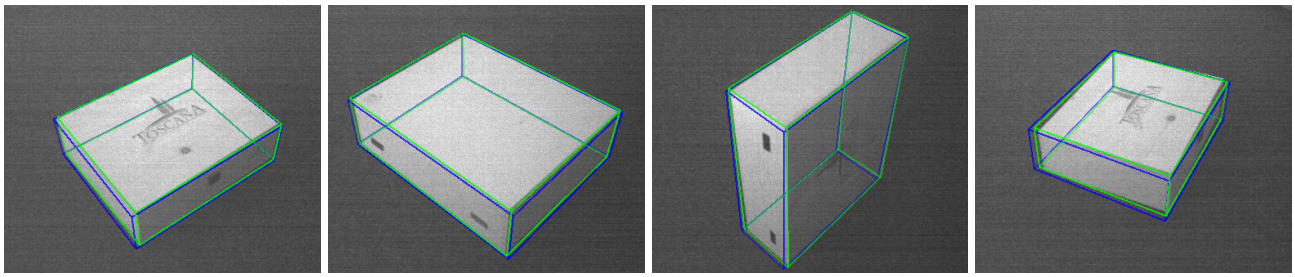
Cuboids are another interesting primitive that can be refined with the proposed method. In this evaluation we represent a cuboid by its dimensions (length × width × height) and its position in terms of a rigid body transformation that relates the cuboid's coordinate system with the coordinate system of the camera. Consequently, $\boldsymbol{\theta}$ contains nine parameters that need to be refined.

The method is evaluated on the *cuboid* data set. Figure 13 shows four selected examples of this data set. The figures show the contours of the initial and refined cuboid models in blue and green respectively.

The initial cuboid is found in a three step process: first the top plane and floor plane are detected. Second, the dimensions of the cuboid are obtained. In the last step, the rotation and translation between the cuboid and the camera is calculated.

The top plane is calculated with the plane segmentation method by Holz *et al.* [12], initialized with a seed coordinate on the top of the box. The floor plane is calculated from all points, except for those points who support the top plane, via RANSAC. For the estimation we assume that the floor is parallel to the top plane of the box and consequently fix the normal to be identical to the top plane's normal. As the floor resembles the majority of the point cloud, the floor plane estimates are very accurate.

Length and width of the box are calculated as the two major dimensions of the minimum bounding box of all pixels that support the top plane. The height is obtained by calculating the distance of one of the cuboid's top corner coordinates to the floor plane.



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

Fig. 13: Qualitative examples from cuboid refinement from the *cuboid* data set. The contours of the initial and refined cuboid are outlined in blue and green respectively. The examples show that a more accurate initialization leads to more accurate results. The box dimensions and positions in (a) and (b) are refined noticeably. The refinement result in (c) shows an improvement from the initial detection, but still has a small offset at the left contour. The initial estimate in (d) has inaccurate rotation parameters which can only be corrected to some extent. However, the dimensions of the box have been adjusted to more accurate parameters.

The cuboid's coordinate system is defined such that the origin is centered in the cuboid, with the coordinate axes aligned with the length, width and height dimensions of the cuboid. This enables an easy calculation of the cuboid's corner coordinates in its own coordinate system. The rotation and translation is calculated from corresponding corner coordinates with Horn's method [13].

In this evaluation, we measure the error of the cuboids detection as the sum of the absolute differences from the true values.

The average error of the initial detections equals 0.94 cm. After refinement, this error is reduced by approximately 30 % to 0.66 cm. Note that this error metric does only consider the dimensions of the boxes but not their position with respect to the camera. For the task of dimensioning, the box position may be only of secondary interest, however, for localization tasks refined spatial information turn out to be equally important as the cuboid's dimensions.

7.10 Runtime

The runtime for refining a single object depends on the number of pixels the object covers, the maximum number of iterations and the number of parameters that are required to describe the object. Out of these factors, the last has the largest impact, as calculating the derivatives numerically requires to render the the object multiple times in each optimization iteration. In our implementation we calculate the central difference, which results in the generation of two virtual images per parameter.

We have evaluated our two variants of our implementation on a consumer notebook. The first variant only uses the Intel I7-4700 CPU, whereas the second variant generates the virtual image on the GPU (Nvidia 750M) in a simple CUDA implementation. The runtime experiments have been conducted with the *wall* data set, which consists of planes that fill the complete field of view. As the number of iterations per image may vary due to early stop criteria in the gradient descent optimizer we only provide measurements for the average time required for a single iteration.

The CPU implementation requires on average 340 ms per iteration, which involves the generation of nine virtual images (eight for the central differences of the the four parameters, plus one image for evaluating the updated parameters). In this implementation we use the complete rendering pipeline, without any approximations like a plane-to-plane mapping based on homographies (see Section 4.4). By moving the rendering process to the GPU, the runtime for a single iteration could be reduced to 40 ms, resulting in a speedup of more than 8. We believe that the runtime can be reduced even further if all features of the GPU would be exploited (interpolation and texture memory). An additional speedup could be achieved by porting the complete optimization to the GPU.

8 Conclusion

In this paper we have presented a novel method for refining model parameters for geometric primitives by exploiting the dot pattern of structured light sensors.

The method exploits that knowing the exact parameters of a geometric primitive in the scene allows to re-render the primitive such that the rendered image is identical to the camera image of the structured light sensor. Unlike regular fitting methods, which work

on individual 3-D point measurements, the proposed algorithm uses all pixels that belong to the geometric primitive simultaneously. Furthermore, the a priori knowledge of the object shape helps to constrain the optimization problem.

As a prerequisite to the estimation of the geometric primitive, we present a robust calibration method for off-the-shelf structured light cameras like the Microsoft Kinect or the Orbbec Astra. With the proposed method, the unknown camera dot pattern, the projector's focal length, and the extrinsic parameters between camera and the projector can be calculated from a small set of calibration images.

We exemplarily demonstrate the performance of the proposed method on plane segments, spheres and cuboids. In the case of planes we use a state-of-the-art plane detection method to find planar segments in the scene. Next, the parameters of these initial planes are refined with the proposed method. It turns out that the method can robustly and highly accurately estimate the normal angle and the distance of a plane segment. For example, we show in our segmentation that plane parameters of highly reflective and diffuse planar segments can be improved by more than 60 % for the normal direction and up to 30 % for the distance parameter. The proposed method is very robust towards texture, plane distance and the size of the plane segment. Also in these very challenging scenarios, the performance gain is in a comparable order of magnitude. In the case of spheres and cuboids, the initially found parameters can be improved by up 50 % and 30 % respectively.

References

1. Bergamasco, F., Albarelli, A., Rodola, E., Torsello, A.: Rune-tag: A high accuracy fiducial marker with strong occlusion resilience. In: 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 113–120 (2011)
2. Bird, N., Papanikolopoulos, N.: Optimal image-based euclidean calibration of structured light systems in general scenes. IEEE Transactions on Automation Science and Engineering **8**(4), 815–823 (2011)
3. Birk, A., Pathak, K., Vaskevicius, N., Pfingsthorn, M., Poppinga, J., Schwertfeger, S.: Surface representations for 3d mapping. KI - Künstliche Intelligenz **24**(3), 249–254 (2010)
4. Biswas, J., Veloso, M.: Depth camera based indoor mobile robot localization and navigation. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 1697–1702 (2012)
5. Borrmann, D., Elseberg, J., Lingemann, K., Nüchter, A.: The 3d hough transform for plane detection in point clouds: A review and a new accumulator design. 3D Research **2**(2) (2011)
6. Feng, C., Taguchi, Y., Kamat, V.R.: Fast plane extraction in organized point clouds using agglomerative hierarchical clustering. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 6218–6225 (2014)
7. Fernndez-Moral, E., Mayol-Cuevas, W., Arvalo, V., Gonzlez-Jimnez, J.: Fast place recognition with plane-based maps. In: 2013 IEEE International Conference on Robotics and Automation, pp. 2719–2724 (2013)
8. Fuersattel, P., Placht, S., Balda, M., Schaller, C., Hofmann, H., Maier, A., Riess, C.: A comparative error analysis of current time-of-flight sensors. IEEE Transactions on Computational Imaging **2**(1), 27–41 (2016)
9. Geiger, A., Moosmann, F., Car, Ö., Schuster, B.: Automatic camera and range sensor calibration using a single shot. In: Robotics and Automation (ICRA), IEEE International Conference on, pp. 3936–3943. IEEE (2012)
10. Georgiev, K., Creed, R.T., Lakaemper, R.: Fast plane extraction in 3d range data based on line segments. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3808–3815 (2011)
11. Herrera, D., Kannala, J., Heikkilä, J.: Accurate and practical calibration of a depth and color camera pair. In: International Conference on Computer analysis of images and patterns, pp. 437–445. Springer (2011)
12. Holz, D., Behnke, S.: Approximate triangulation and region growing for efficient segmentation and smoothing of range images. Robotics and Autonomous Systems **62**(9), 1282–1293 (2014)
13. Horn, B.K.P.: Closed-form solution of absolute orientation using unit quaternions. Journal of the Optical Society of America **4**(4), 629 (1987)
14. Khoshelham, K., Elberink, S.O.: Accuracy and resolution of kinect depth data for indoor mapping applications. Sensors **12**(2), 1437–1454 (2012)
15. McIlroy, P., Izadi, S., Fitzgibbon, A.: Kinectrack: 3d pose estimation using a projected dense dot pattern. IEEE Transactions on Visualization and Computer Graphics **20**(6), 839–851 (2014)
16. Moreno, D., Taubin, G.: Simple, accurate, and robust projector-camera calibration. In: 2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), pp. 464–471 (2012)
17. Mörwald, T., Richtsfeld, A., Prankl, J., Zillich, M., Vincze, M.: Geometric data abstraction using b-splines for range image segmentation. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 148–153 (2013)
18. Nguyen, A., Le, B.: 3d point cloud segmentation: A survey. In: 6th International Conference on Robotics, Automation and Mechatronics (RAM), pp. 225–230 (2013)
19. Placht, S., Fürsattel, P., Mengue, E.A., Hofmann, H., Schaller, C., Balda, M., Angelopoulou, E.: Rochade: robust checkerboard advanced detection for camera calibration. In: Computer Vision – ECCV 2014, *Lecture Notes in Computer Science*, vol. 8692, pp. 766–779. Springer International Publishing (2014)
20. Poppinga, J., Vaskevicius, N., Birk, A., Pathak, K.: Fast plane detection and polygonalization in noisy 3d range images. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3378–3383 (2008)
21. Ryan Fanello, S., Rhemann, C., Tankovich, V., Kowdle, A., Orts Escolano, S., Kim, D., Izadi, S.: Hyperdepth: Learning depth from structured light without matching. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
22. Salas-Moreno, R.F., Glocken, B., Kelly, P.H.J., Davison, A.J.: Dense planar slam. In: IEEE International Symposium on Mixed and Augmented Reality (ISMAR), pp. 157–164 (2014)

23. Taguchi, Y., Jian, Y.D., Ramalingam, S., Feng, C.: Point-plane slam for hand-held 3d sensors. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 5182–5189 (2013)
24. Trevor, A.J.B., Gedikli, S., Rusu, R.B., Christensen, H.I.: Efficient organized point cloud segmentation with connected components. Semantic Perception Mapping and Exploration (SPME) (2013)
25. Weingarten, J., Siegwart, R.: 3d slam using planar segments. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3062–3067 (2006)
26. Yamazaki, S., Mochimaru, M., Kanade, T.: Simultaneous self-calibration of a projector and a camera using structured light. In: 2011 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops), pp. 60–67 (2011)
27. Ye, Y., Song, Z.: A practical means for the optimization of structured light system calibration parameters. In: 2016 IEEE International Conference on Image Processing (ICIP), pp. 1190–1194 (2016)
28. Zhang, Z.: A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence **22**(11), 1330–1334 (2000)