# Automatic Detection of Calibration Grids in Time-of-Flight Images

Miles Hansard, Radu Horaud, Michel Amat, Georgios Evangelidis

# Automatic Detection of Calibration Grids in Time-of-Flight Images

Miles Hansard[a,b], Radu Horaud[a,*], Michel Amat[a,c], Georgios Evangelidis[a]

[a]*INRIA Grenoble Rhône-Alpes, 38330 Montbonnot Saint-Martin, France*
[b]*School of Electronic Engineering and Computer Science, Queen Mary, University of London, Mile End Road, United Kingdom*
[c]*SuperSonic Imagine, 510 Rue René Descartes, 13857 Aix-en-Provence, France*

## Abstract

It is convenient to calibrate time-of-flight cameras by established methods, using images of a chequerboard pattern. The low resolution of the amplitude image, however, makes it difficult to detect the board reliably. Heuristic detection methods, based on connected image-components, perform very poorly on this data. An alternative, geometrically-principled method is introduced here, based on the Hough transform. The projection of a chequerboard is represented by two pencils of lines, which are identified as oriented clusters in the gradient-data of the image. A projective Hough transform is applied to each of the two clusters, in axis-aligned coordinates. The range of each transform is properly bounded, because the corresponding gradient vectors are approximately parallel. Each of the two transforms contains a series of collinear peaks; one for every line in the given pencil. This pattern is easily detected, by sweeping a dual line through the transform. The proposed Hough-based method is compared to the standard OpenCV detection routine, by application to several hundred time-of-flight images. It is shown that the new method detects significantly more calibration boards, over a greater variety of poses, without any overall loss of accuracy. This conclusion is based on an analysis of both geometric and photometric error.

*Keywords:* Range imaging, time-of-flight sensors, camera calibration, Hough transform

## 1. Introduction

Time-of-flight (TOF) cameras [1] produce a *depth image*, each pixel of which encodes the distance to the corresponding point in the scene. These devices emit pulsed infrared illumination, and infer distances from the time taken for light to reflect back to the camera. The TOF sensor can therefore be modelled, geometrically, as a pinhole device. Furthermore, knowledge of the TOF camera-parameters can be used to map raw depth-readings (i.e. distances along lines of sight) into Euclidean scene-coordinates. The calibration thereby enables these devices to be used as stand-alone 3-D sensors, or to be combined with ordinary colour cameras, for complete 3-D modelling and rendering [2, 3, 4, 5, 6, 7, 8].

TOF cameras can, in principle, be calibrated with any existing camera calibration method. For example, if a known chequerboard pattern is detected in a sufficient variety of poses, then the internal and external camera parameters can be estimated by standard routines [9, 10, 11]. It is possible to find the chequerboard vertices, in ordinary images, by first detecting image-corners [12], and subsequently imposing global constraints on their arrangement [13, 14, 15]. This approach, however, is not reliable for low-resolution images (e.g. in the range 100–500 px$^2$) because the local image-structure is disrupted by sampling artefacts, as shown in fig. 2. Furthermore, these artefacts become worse as the board is viewed in distant and slanted positions, which are essential for high quality calibration [16, 17].

*The central motivation of this work is to detect a greater number and variety of calibration board-poses, in* TOF *images, without increasing the geometric error of the vertices*. The geometric error can be conveniently defined with respect to the known geometry of the board, as will be shown in section 3.

TOF sensors provide low-resolution depth and amplitude images. This is because relatively large detector-elements are required in order to allow accumulation of electrons, which increases the signal-to-noise ratio and yields accurate depth-estimates [18] but, in turn, limits the spatial resolution of the devices. This explains the poor performance of heuristic detection-methods, when applied to TOF camera calibration. For example, the amplitude signal from a typical TOF camera [19, 20] resembles an ordinary greyscale image, but is of very low spatial resolution (e.g. 176×144 for the SR4000 camera, or 160 × 120 for the PMD PhotonICs on-chip sensor) , as well as being noisy. A 712×496 CMOS color-depth sensor is currently being developed, but the resolution of the TOF image delivered by this sensor is only 356×248 pixels [21]. A 340×96 pixels TOF camera has also been developed, for driving applications [22]. Lindner at al. [17] used the calibration module included in the OpenCV library [15] to estimate the parameters of a 200 × 200 PMD depth-camera and noticed a high dependency between the intrinsic and extrinsic parameters. To overcome these issues, a calibration method that combines a TOF camera with colour cameras was proposed [3, 17]. While this method yields very accurate parameters, it requires a multiple-sensor setup composed of both TOF and standard cameras.

Calibration grids are essentially composed of two pencils of

---

*Corresponding author
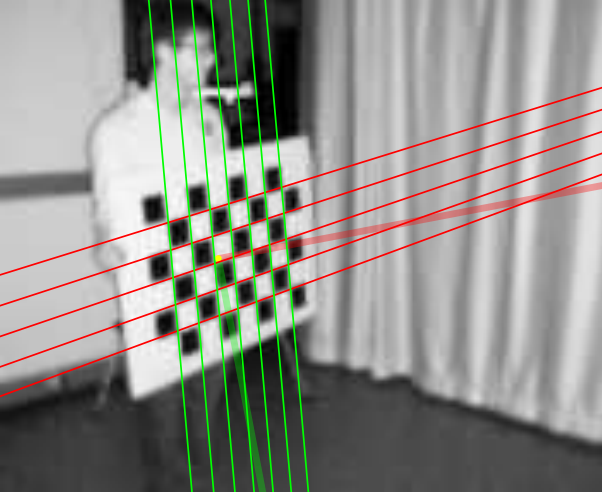Email address:* Radu.Horaud@inria.fr (Radu Horaud)

Figure 1: An example of a ᴛᴏꜰ amplitude image taken in the infrared range in a normal neon-lit room. The original $176 \times 144$ image has been magnified and smoothed for display. The OpenCV software is unable to detect the vertices of the chequerboard in this case and in many other images as the board is viewed in distant and slanted positions. The red and green lines are the initial pencils detected by the proposed Hough-based method (prior to the refinement described in section 3). The thick lines are the local Hough coordinate-systems, which are automatically established, as described in section 2.3.

lines, therefore chequerboard detection should explicitly take this structure into account. For instance, the method in [23, 24] starts by extracting points of interest, followed by eliminating those points that do not have a local chequerboard pattern, and finally by grouping together points lying along lines. This method puts a lot of emphasis on interest points, which are difficult to detect in low-resolution images, and does not take full advantage of the global structure of the calibration grid.

Two families of mutually orthogonal lines may also be detected by finding a dominant pair of vanishing-points. In [25] it is proposed to represent image lines on the Gaussian sphere (a unit sphere around the optical center of the camera). Under perspective projection, an image line projects onto a great circle on the Gaussian sphere, and a pencil of lines corresponds to a family great circles that intersect at antipodal points (see for example fig. 1 in [26]). Therefore, a vanishing point may be found by detecting the intersections, provided that the camera's internal parameters are known. Vanishing point detection was implemented using a quantized Gaussian sphere and a hierarchical (scale-space) Hough method, e.g. [27]. In general, Gaussian sphere-based methods require the detection of edges or of straight lines which are then projected as point sets (circles) on an azimuth-elevation grid, which may also produce spurious vanishing points [26].

More recently, vanishing-point detection was addressed as a clustering problem in the parameter space, using maximum likelihood and the EM algorithm [28], which requires suitable initialization. Alternatively, parameter-space clustering can be implemented using minimal sets [29] and random sampling. A method that combines [29] with an EM algorithm was recently proposed to find the three most orthogonal pencils of lines in indoor and outdoor scenes [30]. We tested this method using

the software provided by the author[1] but found that the algorithm was not able to reliably extract and label edges from the low-resolution ᴛᴏꜰ amplitude images. We conclude that vanishing point methods, e.g., [30, 31] fail to extract pencils of lines because they require accurate edge detection that is difficult to accomplish in low resolution, noisy images.

The method described in this paper is also based on the Hough transform [32], but it effectively fits a specific model to the chequerboard pattern, e.g., fig. 1. This process is much less sensitive to the resolution of the data, for two reasons. Firstly, information is integrated across the source image, because each vertex is obtained from the intersection of two fitted lines. Secondly, the structure of a straight edge is inherently simpler than that of a corner feature. However, for this approach to be viable, it is assumed that any lens distortion has been pre-calibrated, so that the images of the pattern contain straight lines. This is not a serious restriction, because it is relatively easy to find enough boards (by any heuristic method) from which to obtain adequate estimates of the internal and lens parameters. Indeed there exist lens-calibration methods that require only a single image [33, 34, 35]. The harder problems of reconstruction and relative orientation can then be addressed after adding the newly detected boards, ending with a bundle-adjustment that also refines the initial internal parameters. Furthermore, the ᴛᴏꜰ devices used here have fixed lenses, which are sealed inside the camera body. This means that the internal and lens-distortion parameters from previous calibrations can be re-used.

Another Hough-method for chequerboard detection has been presented by de la Escalera and Armingol [36]. Their algorithm involves a *polar* Hough transform of all high-gradient points in the image. This results in an array that contains a peak for each line in the pattern. It is not, however, straightforward to extract these peaks, because their location depends strongly on the unknown orientation of the image-lines. Hence all local maxima are detected by morphological operations, and a second Hough transform is applied to the resulting data in [36]. The true peaks will form two collinear sets in the first transform (cf. sec. 2.4), and so the final task is to detect two peaks in the second Hough transform. This iteration makes it hard to determine an appropriate sampling scheme, and also increases the computation and storage time of the procedure [37].

The method described in this paper is quite different. It makes use of the gradient *orientation* as well as magnitude at each point, in order to establish an axis-aligned coordinate system for each image of the pattern. Separate Hough transforms are then performed in the *x* and *y* directions of the local coordinate system. By construction, the slope-coordinate of any line is close to zero in the corresponding *Cartesian* Hough transform. This means that, on average, the peaks occur along a fixed axis of each transform, and can be detected by a simple sweep-line procedure. Furthermore, the known $\ell \times m$ structure of the grid makes it easy to identify the optimal sweep-line in each transform. Finally, the two optimal sweep-lines map directly back to pencils of $\ell$ and $m$ lines in the original image, owing to the

---

Cartesian nature of the transform. The principle of the method is shown in fig. 3.

It should be noted that the method presented here was designed specifically for use with TOF cameras. For this reason, the *range*, as well as intensity data are used to help segment the image in sec. 2.1. However, this step could easily be replaced with an appropriate background subtraction procedure [15], in which case the new method could be applied to ordinary RGB images. Camera calibration is typically performed under controlled illumination conditions, and so there would be no need for a dynamic background model.

## 1.1. Overview and Contributions

The new method is described in section 2; preprocessing and segmentation are explained in sections 2.1 and 2.2 respectively, while section 2.3 describes the geometric representation of the data. The necessary Hough transforms are defined in section 2.4, and analyzed in sections 2.5 and 2.6. The new method is evaluated on over 700 detections in section 3, and shown to be substantially better, for TOF images, than the standard OpenCV method. Conclusions are stated in section 4. [2].

The main contributions of this paper are the use of a double-angle mapping to segment the gradient vectors (2.2), the splitting of detection process into a pair of Cartesian Hough transforms (2.4), and the sweep-line method of analyzing these transforms (2.5, 2.6).

## 1.2. Notation

Matrices and vectors will be written in bold, e.g. $\boldsymbol{M}$, $\boldsymbol{v}$, and the Euclidean length of $\boldsymbol{v}$ will be written $|\boldsymbol{v}|$. Equality up to an overall nonzero-scaling will be written $\boldsymbol{v} \simeq \boldsymbol{u}$. Image-points and lines will be represented in homogeneous coordinates [10], $\boldsymbol{p} \simeq (x, y, 1)^{\top}$ and $\boldsymbol{\lambda} \simeq (\alpha, \beta, \gamma)$, such that $\boldsymbol{\lambda p} = 0$ if $\boldsymbol{\lambda}$ passes through $\boldsymbol{p}$. The intersection-point of two homogeneous lines can be obtained from the cross-product $(\boldsymbol{\lambda} \times \boldsymbol{\mu})^{\top}$. An assignment from variable $a$ to variable $b$ will be written $b \leftarrow a$. It will be convenient, for consistency with the pseudo-code listings, to use the notation $(m : n)$ for the sequence of integers from $m$ to $n$ inclusive. The 'null' symbol $\varnothing$ will be used to denote undefined or unused variables in the algorithms.

## 2. Method

It is convenient to begin with an overview of the complete algorithm, before describing the exact form of the input data. Following this, subsections 2.1–2.6 will describe each step in detail. The individual stages of the algorithm are as follows:

A. *Preprocessing.* The background of the image is roughly identified, by depth-thresholding or image-differencing, and discarded. The gradient of the remaining image is then computed.
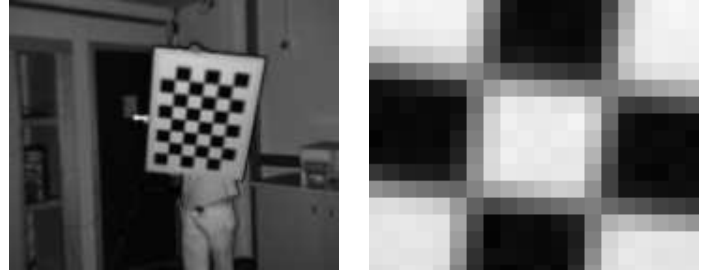
Figure 2: Example chequers from a TOF amplitude image (shown on the left). Note the variable appearance of the four junctions at this resolution; '× like' at lower-left vs. '+ like' at top-right.

B. *Gradient Clustering.* Two gradient clusters, one for each parallel set of edges on the board, are identified. All weak gradients are discarded.

C. *Local Coordinates.* A pair of orthogonal axes, centred on the board, are constructed from the two gradient clusters.

D. *Hough Transform.* Two Cartesian Hough transforms are performed, one for each gradient cluster. The local coordinate system ensures that all edge directions can be properly represented.

E. *Hough Analysis.* A line is swept across both transforms, until a collinear set of peaks is found in each case. The two sets of peaks correspond to two pencils of image-lines, the intersections of which are the estimated vertices of the calibration grid.

F. *Decision Functions.* The solution is accepted if two tests are passed: Firstly, the separation of adjacent lines in each pencil must not be too variable. Secondly, there must be black squares on both sides of the outermost lines of each pencil.

The form of the input data will now be detailed. Suppose that the chequerboard has $(\ell + 1) \times (m + 1)$ squares, with $\ell < m$. It follows that the *internal* vertices of the pattern are imaged as the $\ell m$ line-intersections

$$\boldsymbol{v}_{ij} = \boldsymbol{\lambda}_i \times \boldsymbol{\mu}_j \quad \text{where} \begin{cases} \boldsymbol{\lambda}_i \in \mathcal{L} \text{ for } i = 1 : \ell, \text{ and} \\ \boldsymbol{\mu}_j \in \mathcal{M} \text{ for } j = 1 : m. \end{cases} \quad (1)$$

The sets $\mathcal{L}$ and $\mathcal{M}$ are *pencils*, meaning that the $\boldsymbol{\lambda}_i$ all intersect at a point $\boldsymbol{p}$, while the $\boldsymbol{\mu}_j$ all intersect at a point $\boldsymbol{q}$. Note that $\boldsymbol{p}$ and $\boldsymbol{q}$ are the *vanishing points* of the grid-lines, which may be at infinity in the images (cf. pencil $\mathcal{M}$ in fig. 3).

It is assumed that the imaging device, such as a TOF camera, provides a range map $D_{xy}$, containing distances from the optical centre, as well as a luminance-like amplitude map $A_{xy}$, where $(x, y)$ represents a pixel. The images $D$ and $A$ are of size $X \times Y$. All images must be undistorted, as described in the introduction.

## 2.1. Preprocessing

The amplitude image $A$ is roughly segmented, by discarding all pixels that correspond to very near or far points. This gives
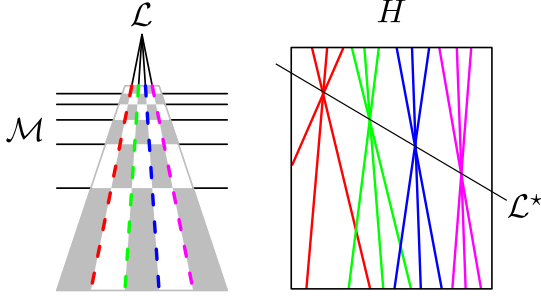
Figure 3: **Left:** A perspective image of a calibration grid is represented by line-pencils $\mathcal{L}$ and $\mathcal{M}$, which intersect at the $\ell \times m = 20$ internal vertices of this board. Strong image-gradients are detected along the dashed lines. **Right:** The Hough transform $H$ of the image-points associated with $\mathcal{L}$. Each high-gradient point maps to a line, such that there is a pencil in $H$ for each set of edge-points. The line $\mathcal{L}^\star$, which passes through the $\ell = 4$ Hough-vertices, is the Hough representation of the image-pencil $\mathcal{L}$.
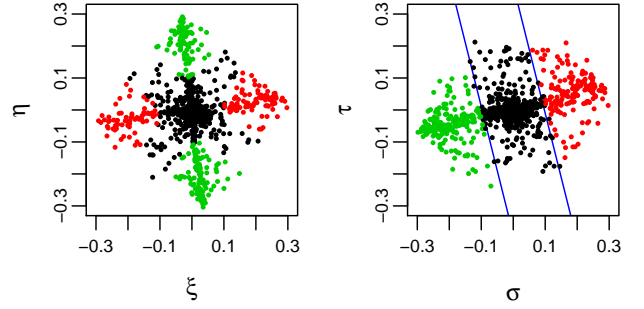


Figure 4: **Left:** the cruciform distribution of image gradients, due to black/white and white/black transitions at each orientation, would be difficult to segment in terms of horizontal and vertical components $(\xi, \eta)$. **Right:** the same distribution is easily segmented, by eigen-analysis, in the double-angle representation of equation (4). The red and green labels are applied to the corresponding points in the original distribution, on the left.

a new image $B$, which typically contains the board, plus the person holding it:

$$B_{xy} \leftarrow \begin{cases} A_{xy} \text{ if } d_0 < D_{xy} < d_1 \\ \varnothing \text{ otherwise.} \end{cases} \quad (2)$$

The near-limit $d_0$ is determined by the closest position for which the board remains fully inside the field-of-view of the camera. The far-limit $d_1$ is typically set to a value just closer than the far wall of the scene. These parameters need only be set approximately, provided that the interval $[d_1, d_0]$ covers the possible positions of the calibration board.

It is useful to perform a morphological erosion operation at this stage, in order to partially remove the perimeter of the board. In particular, if the physical edge of the board is not white, then it will give rise to irrelevant image-gradients. The erosion radius need only be set approximately (a value of 2px was used here), assuming that there is a reasonable amount of white-space around the chessboard pattern.

The gradient of the remaining amplitude image is now computed, using the simple kernel $\Delta = (-1/2, 0, 1/2)$. The horizontal and vertical components are

$$\begin{aligned} \xi_{xy} &\leftarrow (\Delta \star B)_{xy} \\ &= \rho \cos\theta \end{aligned} \quad \text{and} \quad \begin{aligned} \eta_{xy} &\leftarrow (\Delta^\top \star B)_{xy} \\ &= \rho \sin\theta \end{aligned} \quad (3)$$

where $\star$ indicates convolution, and $(\rho, \theta)$ is the polar representation of the gradient. No pre-smoothing of the image is performed, owing to the low spatial resolution of the data.

### 2.2. Gradient Clustering

The objective of this section is to assign each gradient vector $(\xi_{xy}, \eta_{xy})$ to one of three classes, with labels $\kappa_{xy} \in \{\lambda, \mu, \varnothing\}$. If $\kappa_{xy} = \lambda$ then pixel $(x, y)$ is on one of the lines in $\mathcal{L}$, and $(\xi_{xy}, \eta_{xy})$ is perpendicular to that line. If $\kappa_{xy} = \mu$, then the analogous relations hold with respect to $\mathcal{M}$. If $\kappa_{xy} = \varnothing$ then pixel $(x, y)$ does not lie on any of the lines.

The gradient distribution, after the initial segmentation, will contain two elongated clusters through the origin, which will be approximately orthogonal. Each *cluster* corresponds to a

gradient orientation (mod $\pi$), while each *end* of a cluster corresponds to a gradient polarity (black/white vs. white/black). Two methods of identifying these clusters are described below; a principal component method, and a RANSAC method.

The *principal component method* begins with a double-angle mapping of the data [38], which will be expressed as $(\xi, \eta) \mapsto (\sigma, \tau)$. This mapping results in a *single* elongated cluster, each *end* of which corresponds to a gradient orientation (mod $\pi$). A real example of this mapping is shown in fig. 4.

The double-angle coordinates are obtained by applying the trigonometric identities $\cos(2\theta) = \cos^2\theta - \sin^2\theta$ and $\sin(2\theta) = 2\sin\theta\cos\theta$ to the gradients (3), so that

$$\sigma_{xy} \leftarrow \frac{1}{\rho_{xy}}(\xi_{xy}^2 - \eta_{xy}^2) \quad \text{and} \quad \tau_{xy} \leftarrow \frac{2}{\rho_{xy}}\xi_{xy}\eta_{xy} \quad (4)$$

$$\text{where} \quad \rho_{xy} = \sqrt{\xi_{xy}^2 + \eta_{xy}^2}$$

for all points at which the magnitude $\rho_{xy}$ is above machine precision. Let the first unit-eigenvector of the $(\sigma, \tau)$ covariance matrix be $(\cos(2\phi), \sin(2\phi))$, which is written in this way so that the angle $\phi$ can be interpreted in the original image. The cluster-membership is now defined by the orthogonal projection

$$\pi_{xy} = (\sigma_{xy}, \tau_{xy}) \cdot (\cos(2\phi), \sin(2\phi)) \quad (5)$$

of the data onto this axis. It is now straightforward to classify the gradient-vectors $(\xi_{xy}, \eta_{xy})$, as shown in fig. 4, according to a threshold $\pi_{\min}$;

$$\kappa_{xy} \leftarrow \begin{cases} \lambda & \text{if } \pi_{xy} \geq \pi_{\min} \\ \mu & \text{if } \pi_{xy} \leq -\pi_{\min} \\ \varnothing & \text{otherwise.} \end{cases} \quad (6)$$

Hence the symbol $\varnothing$ is assigned to all strong gradients that are *not* aligned with either axis of the board, as well as to *all* weak gradients.

The above method is robust to moderate perspective effects, because only the first principal component is needed, and this is well-defined for any elongated distribution. However, in order

4

to include boards with extreme perspective distortion, an even more robust RANSAC method can be used, as described below.

The *RANSAC method* is based on the fact that two lines through the origin can be defined by two points; one on each line. These two points are randomly sampled in the $(\xi, \eta)$ gradient space, and used to define two normal vectors $(-\eta_\lambda, \xi_\lambda)$ and $(-\eta_\mu, \xi_\mu)$. The projections of the all $(x, y)$ gradient points, onto each unit vector, are defined as

$$\pi_{xy}^\lambda = \frac{(-\eta_\lambda, \xi_\lambda) \cdot (\xi_{xy}, \eta_{xy})}{\sqrt{\xi_\lambda^2 + \eta_\lambda^2}} \tag{7}$$

$$\pi_{xy}^\mu = \frac{(-\eta_\mu, \xi_\mu) \cdot (\xi_{xy}, \eta_{xy})}{\sqrt{\xi_\mu^2 + \eta_\mu^2}} \tag{8}$$

The gradients are are now classified, in relation to a slab of thickness $2 \times \pi_{\min}$ around each line:

$$\kappa_{xy} \leftarrow \begin{cases} \lambda & \text{if } |\pi_{xy}^\lambda| \le \pi_{\min} \text{ and } |\pi_{xy}^\mu| > \pi_{\min} \\ \mu & \text{if } |\pi_{xy}^\mu| \le \pi_{\min} \text{ and } |\pi_{xy}^\lambda| > \pi_{\min} \\ \varnothing & \text{otherwise.} \end{cases} \tag{9}$$

Points that are in *both* slabs (i.e. around the intersection) are given the null label, which means that more points are excluded as the two lines become more parallel. This has desirable effect of automatically excluding more weak gradients as the perspective distortion increases. The quality of the classification is defined as the number of non-null labels, and the best solution is chosen, as usual, from the ensemble of samples.

In general, the RANSAC method is more robust that the the principal component method; it does, however, have two drawbacks. Firstly, it is necessary to set the number of random samples to be drawn, which introduces an additional parameter. Secondly, the final result of the calibration will not be exactly repeatable, unless the same random-number generator and seed are employed each time.

Suppose that the gradients have now been classified, using either the principal component method or the RANSAC method, as described above. The next task is to resolve the respective *identities* of the clusters, with respect to labels $\lambda$ and $\mu$. In principle, the class that contains the greater number of gradients should correspond to $\mathcal{L}$, which was defined as the pencil containing fewer lines. This is because, for a fronto-parallel board, the total *lengths* of the edges in $\mathcal{L}$ and $\mathcal{M}$ are proportional to $\ell(m-1)$ and $m(\ell-1)$ respectively, with $\ell < m$. This prediction is unreliable, in practice, owing to foreshortening and other image-effects. For this reason, the correspondence $\{\lambda, \mu\} \Leftrightarrow \{\mathcal{L}, \mathcal{M}\}$ between labels and pencils will be resolved more robustly, in section 2.5.

### 2.3. Local Coordinates

A coordinate system will now be constructed for each image of the board. Recall from (2) that amplitude-image $B$ typically contains the board, as well as the person holding it. The very low amplitudes $B_{xy} \approx 0$ of the *black* squares tend to be characteristic of the board itself (i.e. $B_{xy} \gg 0$ for both the white

squares and for the rest of $B$). Hence a good estimate of the board-centre can be obtained by normalizing $B$ to the range $[0, 1]$ and then computing a centroid using weights $(1 - B_{xy})$. The centroid, together with the angle $\phi$ from (5) defines the Euclidean transformation $E$ into local coordinates, centred on and aligned with the board. From now on, unless otherwise stated, it will be assumed that this simple transformation has been performed.

Let $(x_\kappa, y_\kappa, 1)^\top$ be the local coordinates of given point, after transformation by $E$, with the label $\kappa$ inherited from $\kappa_{xy}$. Now, by construction, any labelled point is hypothesized to be part of $\mathcal{L}$ or $\mathcal{M}$, such that that $\lambda(x_\lambda, y_\lambda, 1)^\top = 0$ or $\mu(x_\mu, y_\mu, 1)^\top = 0$, where $\lambda$ and $\mu$ are the local coordinates of the relevant lines. These lines can be expressed as

$$\lambda \simeq (-1, \beta_\lambda, \alpha_\lambda) \quad \text{and} \quad \mu \simeq (\beta_\mu, -1, \alpha_\mu) \tag{10}$$

with inhomogeneous forms $x_\lambda = \alpha_\lambda + \beta_\lambda y_\lambda$ and $y_\mu = \alpha_\mu + \beta_\mu x_\mu$, such that the slopes $|\beta_\kappa| \ll 1$ are *bounded*. In other words, the board is axis-aligned in local coordinates, and the perspective-induced deviation of any line is less than $45°$. Furthermore, if the board is visible, then the intercepts $|\alpha_\kappa| \ll \frac{1}{2}(X + Y)$ are bounded in relation to the image size.

Recall from (1) that the vertices $v_{ij}$ of the board are computed as the intersections of $\mathcal{L}$ with $\mathcal{M}$. The resulting points can be expressed in the *original* image coordinates as $v_{ij} \simeq E^{-1}(\lambda_i \times \mu_j)^\top$, via the inverse transformation.

### 2.4. Hough Transform

The Hough transform, as used here, maps *points* from the image to *lines* in the transform. In particular, points along a line are mapped to lines through a point. This duality between collinearity and concurrency suggests that a *pencil* of $n$ image-lines will be mapped to a *line* of $n$ transform points, as in fig. 3.

The transform is implemented as a 2-D histogram $H(u, v)$, with horizontal and vertical coordinates $u \in [0, u_1]$ and $v \in [0, v_1]$. The point $(u_0, v_0) = \frac{1}{2}(u_1, v_1)$ is the centre of the transform array. Two transforms, $H_\lambda$ and $H_\mu$, will be performed, for points labelled $\lambda$ and $\mu$, respectively. The Hough variables are related to the image coordinates in the following way; let

$$u(x, y, v) = u_0 + x - y(v - v_0), \quad \text{and}$$

$$u_\kappa(x, y, v) = \begin{cases} u(x, y, v) & \text{if } \kappa = \lambda \\ u(y, x, v) & \text{if } \kappa = \mu \end{cases} \tag{11}$$

Here $u(x, y, v)$ is the $u$-coordinate of a line (parameterized by $v$), which is the Hough-transform of an image-point $(x, y)$. The Hough intersection point $(u_\kappa^\star, v_\kappa^\star)$ is found by taking two points $(x, y)$ and $(x', y')$, and solving $u_\lambda(x, y, v) = u_\lambda(x', y', v)$, with $x_\lambda$ and $x'_\lambda$ substituted according to (10). The same coordinates are obtained by solving $u_\mu(x, y, v) = u_\mu(x', y', v)$, and so the result can be expressed as

$$u_\kappa^\star = u_0 + \alpha_\kappa \quad \text{and} \quad v_\kappa^\star = v_0 + \beta_\kappa \tag{12}$$

with labels $\kappa \in \{\lambda, \mu\}$ as usual. A peak at $(u_\kappa^\star, v_\kappa^\star)$ evidently maps to a line of intercept $u_\kappa^\star - u_0$ and slope $v_\kappa^\star - v_0$. Note that if the

perspective distortion in the images is small, then $\beta_\kappa \approx 0$, and all intersection points lie along the horizontal midline $(u, v_0)$ of the corresponding transform. The Hough intersection point $(u_\kappa^\star, v_\kappa^\star)$ can be used to construct an image-line $\lambda$ or $\mu$, by combining (12) with (10), resulting in

$$\lambda \leftarrow (-1, \; v_\lambda^\star - v_0, \; u_\lambda^\star - u_0)$$
$$\mu \leftarrow (v_\mu^\star - v_0, \; -1, \; u_\mu^\star - u_0). \tag{13}$$

The two Hough transforms are computed by the procedure in fig. 5. Let $H_\kappa$ refer to $H_\lambda$ or $H_\mu$, according to the label $\kappa$ of the point $(x, y)$. For each accepted point, the corresponding line (11) intersects the top and bottom of the $(u, v)$ array at points $(s, 0)$ and $(t, v_1)$ respectively. The resulting segment, of length $w_1$, is evenly sampled, and $H_\kappa$ is incremented at each of the constituent points.
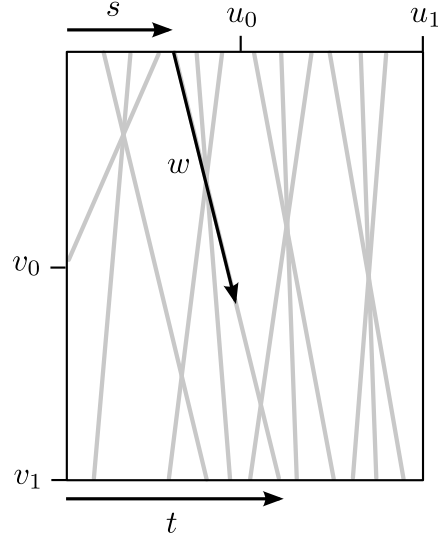
The procedure in fig. 5 makes use of the following functions. Firstly, $\text{interp}_\alpha(\boldsymbol{p}, \boldsymbol{q})$, with $\alpha \in [0, 1]$, returns the convex combination $(1 - \alpha)\boldsymbol{p} + \alpha\boldsymbol{q}$. Secondly, the 'accumulation' $H \oplus (u, v)$ is equal to $H(u, v) \leftarrow H(u, v) + 1$ if $u$ and $v$ are integers. In the general case, however, the four pixels closest to $(u, v)$ are updated by the corresponding bilinear-interpolation weights (which sum to one). This weighting-scheme, combined with the large number of gradient-vectors that are processed, tends to produce a relatively smooth histogram.

### 2.5. Hough Analysis

The local coordinates defined in sec. 2.3 ensure that the two Hough transforms $H_\lambda$ and $H_\mu$ have the same characteristic structure. Hence the subscripts $\lambda$ and $\mu$ will be suppressed for the moment. Recall that each Hough cluster corresponds to a line in the image space, and that a collinear set of Hough clusters corresponds to a pencil of lines in the image space, as in fig 3. It follows that all lines in a pencil can be detected simultaneously, by *sweeping* the Hough space $H$ with a line that cuts a 1-D slice through the histogram.

Recall from section 2.4 that the Hough peaks are most likely to lie along a horizontal axis (corresponding to a fronto-parallel pose of the board). Hence a suitable parameterization of the sweep-line is to vary one endpoint $(0, s)$ along the left edge, while varying the other endpoint $(u_1, t)$ along the right edge, as in fig. 6. This scheme has the desirable property of sampling more densely around the midline $(u, v_0)$. It is also useful to note that the sweep-line parameters $s$ and $t$ can be used to represent the apex of the corresponding pencil. The local coordinates $\boldsymbol{p}$ and $\boldsymbol{q}$ are $\boldsymbol{p} \simeq (\lambda_s \times \lambda_t)^\top$ and $\boldsymbol{q} \simeq (\mu_s \times \mu_t)^\top$ where $\lambda_s$ and $\lambda_t$ are obtained from (10) by setting $(u_\lambda^\star, v_\lambda^\star)$ to $(0, s)$ and $(u_1, t)$ respectively, and similarly for $\mu_s$ and $\mu_t$.

The procedure shown in fig. 6 is used to analyze the Hough transform. The sweep-line with parameters $s$ and $t$ has the form of a 1-D histogram $h_\kappa^{st}(w)$. The integer index $w \in (0 : w_1)$ is equal to the Euclidean distance $|(u, v) - (0, s)|$ along the sweep-line. The procedure shown in fig. 6 makes further use of the interpolation operator that was defined in section 2.4. Each sweep-line $h_\kappa^{st}(w)$, constructed by the above process, will contain a number of isolated clusters: $\left|\text{clusters}(h_\kappa^{st})\right| \geq 1$. These



$$
\begin{aligned}
&\textbf{for } (x, y) \textbf{ in } (0 : X) \times (0 : Y) \\
&\quad \textbf{if } \kappa_{xy} \neq \varnothing \\
&\qquad \kappa \leftarrow \kappa_{xy} \\
&\qquad s \leftarrow u_\kappa(x, y, 0) \\
&\qquad t \leftarrow u_\kappa(x, y, v_1) \\
&\qquad w_1 \leftarrow \left|(t, v_1) - (s, 0)\right| \\
&\qquad \textbf{for } w \textbf{ in } (0 : \text{floor}(w_1)) \\
&\qquad\quad H_\kappa \leftarrow H_\kappa \oplus \underset{w/w_1}{\text{interp}}((s, 0), (t, v_1)) \\
&\qquad \textbf{end} \\
&\quad \textbf{endif} \\
&\textbf{end}
\end{aligned}
$$

Figure 5: **Left:** Hough transform space. **Right:** Constructing the transform. Each gradient pixel $(x, y)$ labelled $\kappa \in \{\lambda, \mu\}$ maps to a line $u_\kappa(x, y, v)$ in transform $H_\kappa$. The operators $H \oplus \boldsymbol{p}$ and $\text{interp}_\alpha(\boldsymbol{p}, \boldsymbol{q})$ perform accumulation and linear interpolation, respectively. See section 2.4 for details.

clusters are simply defined as runs of non-zero values in $h_\kappa^{st}(w)$. The existence of separating zeros is, in practice, highly reliable when the sweep-line is close to the true solution. This is simply because the Hough data was thresholded in (6), and strong gradients are not found *inside* the chessboard squares. The representation of the clusters, and subsequent evaluation of each sweep-line, will now be described.

The label $\kappa$ and endpoint parameters $s$ and $t$ will be suppressed, in the following analysis of a single sweep-line, for clarity. Hence let $w \in (a_c : b_c)$ be the interval that contains the $c$-th cluster in $h(w)$. The score and location of this cluster are
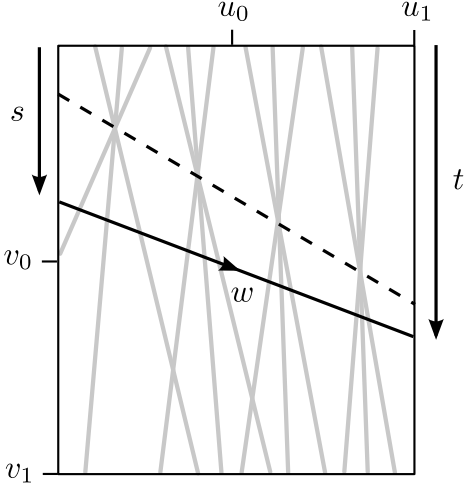
```
for (s, t) in (0 : v₁) × (0 : v₁)
    w₁ = |(u₁, t) − (0, s)|
    for w in (0 : floor(w₁))
        (u, v) ← interp((0, s), (u₁, t))
                    w/w₁
        h_λ^st(w) ← H_λ(u, v)
        h_μ^st(w) ← H_μ(u, v)
    end
end
```

Figure 6: **Left:** Hough transform space. **Right:** Searching the transform. A line $h_\kappa^{st}(w)$, with end-points $(0, s)$ and $(u_1, t)$, is swept through each Hough transform $H_\kappa$. A total of $v_1 \times v_1$ 1-D histograms $h_\kappa^{st}(w)$ are computed in this way. See section 2.5 for details.

defined as the mean value and centroid, respectively:

$$\operatorname*{score}_{c}(h) = \frac{\sum_{w=a_c}^{b_c} h(w)}{1 + b_c - a_c} \qquad (14)$$

$$w_c = a_c + \frac{\sum_{w=a_c}^{b_c} h(w)\, w}{\sum_{w=a_c}^{b_c} h(w)} \qquad (15)$$

More sophisticated definitions are possible, based on quadratic interpolation around each peak. However, the mean and centroid give similar results in practice. A total score must now be assigned to the sweep-line, based on the scores of the constituent clusters. If $n$ peaks are sought, then the total score is the sum of the highest $n$ cluster-scores. But if there are fewer than $n$ clusters in $h(w)$, then this cannot be a solution, and the score is zero:

$$\Sigma^n(h) = \begin{cases} \sum_{i=1}^{n} \operatorname*{score}_{c(i)}(h) & \text{if } n \leq |\text{clusters}(h)| \\ 0 & \text{otherwise} \end{cases} \qquad (16)$$

where $c(i)$ is the index of the $i$-th highest-scoring cluster. The optimal clusters are those in the sweep-line that maximizes (16). Now, restoring the full notation, the score of the

optimal sweep-line in the transform $H_\kappa$ is

$$\Sigma_\kappa^n \leftarrow \max_{s, t} \operatorname*{score}_n(h_\kappa^{st}). \qquad (17)$$

One problem remains: it is not known in advance whether there should be $\ell$ peaks in $H_\lambda$ and $m$ in $H_\mu$, or vice versa. Hence all four combinations, $\Sigma_\lambda^\ell$, $\Sigma_\mu^m$, $\Sigma_\mu^\ell$, $\Sigma_\lambda^m$ are computed. The ambiguity between pencils $(\mathcal{L}, \mathcal{M})$ and labels $(\lambda, \mu)$ can then be resolved, by picking the solution with the highest *total* score:

$$(\mathcal{L}, \mathcal{M}) \Leftrightarrow \begin{cases} (\lambda, \mu) & \text{if } \Sigma_\lambda^\ell + \Sigma_\mu^m > \Sigma_\mu^\ell + \Sigma_\lambda^m \\ (\mu, \lambda) & \text{otherwise.} \end{cases} \qquad (18)$$

Here, for example, $(\mathcal{L}, \mathcal{M}) \Leftrightarrow (\lambda, \mu)$ means that there is a pencil of $\ell$ lines in $H_\lambda$ and a pencil of $m$ lines in $H_\mu$. The procedure in (18) is based on the fact that the complete solution must consist of $\ell + m$ clusters. Suppose, for example, that there are $\ell$ good clusters in $H_\lambda$, and $m$ good clusters in $H_\mu$. Of course there are also $\ell$ good clusters in $H_\mu$, because $\ell < m$ by definition. However, if only $\ell$ clusters are taken from $H_\mu$, then an additional $m - \ell$ weak or non-existent clusters must be found in $H_\lambda$, and so the total score $\Sigma_\mu^\ell + \Sigma_\lambda^m$ would not be maximal.

It is straightforward, for each centroid $w_c$ in the optimal sweep-line $h_\kappa^{st}$, to compute the 2-D Hough coordinates

$$(u_\kappa^\star, v_\kappa^\star) \leftarrow \operatorname*{interp}_{w_c/w_1}((0, s), (u_1, t)) \qquad (19)$$

where $w_1$ is the length of the sweep-line, as in fig. 6. Each of the resulting $\ell m$ points are mapped to image-lines, according to (13). The vertices $v_{ij}$ are then computed from (1). The order of intersections along each line is preserved by the Hough transform, and so the $ij$ indexing is automatically consistent.

Finally, in order to minimize any effects of discretization, it would be possible to perform quadratic (or other) interpolation around the maximal sweep-line score, with respect to the endpoint parameters. In practice, owing to the high resolution of the sweep-procedure around zero-slope (as described above), this does not prove to be necessary.

*2.6. Decision Functions*

The analysis of section 2.5 returns estimated pencils $(\mathcal{L}, \mathcal{M})$, even if the board is not visible in the image. Hence it is necessary to evaluate the quality of the solution. One possibility would be to test a statistic based on the scores in (18). However, a more robust approach is to test whether the solution $(\mathcal{L}, \mathcal{M})$ satisfies certain geometric constraints that were *not* used in the estimation process.

There are, in practice, two types of mis-detections. In the first case a minority of the lines are not aligned with the chessboard pattern, and so the solution is **corrupted**. In the second case the lines include an external edge of the chessboard pattern, and so the entire solution is **displaced**. These two types of error, which may co-occur, are addressed below.

Recall that vertex $v_{ij}$ is the intersection between lines $\lambda_i$ and $\mu_j$, as in (1). Corrupted estimates can be detected by noting that *cross ratios* of distances between quadruples of vertices should

7

be near unity, given that the observations are projectively related to a regular grid [10, 24]. In practice, extremely foreshortened boards cannot be detected (owing to limited resolution), and so it suffices to use an *affine* test. In particular, for line $\lambda_i$, consider the ratio of the vertex-intervals starting at positions $j$ and $k$:

$$F_{jk}(\lambda_i) = \frac{|v_{i(j+1)} - v_{ij}|}{|v_{i(k+1)} - v_{ik}|}. \tag{20}$$

These ratios are tested along the first and last line in each pencil, and so a suitable set of decision functions is

$$\left| 1 - F_{jk}(\lambda_i) \right| \leq f \quad \text{for all } i, j, k \tag{21}$$

$$i = \{1, \ell\}, \quad j, k \in (1 : m - 1), \quad j \neq k$$

for a small positive threshold $f$. The analogous tests are applied to the other pencil, $\mathcal{M}$. If any of the tests (21) are failed, then the estimate $(\mathcal{L}, \mathcal{M})$ is rejected.

Displaced estimates can be determined as follows. Let $w \in \lambda_i$ be an image-point on the line segment between vertices $v_{i1}$ and $v_{im}$ that lies 'inside' the pencil $\mathcal{M}$. If this segment is strictly inside the chessboard pattern, then there should be equal numbers of black-white and white-black transitions across it. If, on the other hand, the segment is along the perimeter of the pattern, then the two types of transition will be very imbalanced, as the black squares are 'missing' on one side of the perimeter. Let $(\xi_w, \eta_w)$ be the gradient at edge-point $w$, and let $\lambda^\star = (\alpha, \beta, \gamma)$ be the normalized line coordinates, such that $\alpha^2 + \beta^2 = 1$. Now the dot-product $\lambda^\star(\xi_w, \eta_w, 0)^\top$ is the projection of the gradient onto the edge-normal at $w$, and so the amount of black-white vs. white-black transitions can be measured by the ratio of the summed positive and negative projections,

$$G(\lambda_i) = \frac{\sum_{k=0}^{k \leq d} \text{pos}(\lambda_i^\star (\xi_w, \eta_w, 0)^\top)}{\sum_{k=0}^{k \leq d} \text{neg}(\lambda_i^\star (\xi_w, \eta_w, 0)^\top)} \quad \text{where} \tag{22}$$

$$d = |v_{im} - v_{i1}| \quad \text{and} \quad w = \underset{k/d}{\text{interp}}(v_{i1}, v_{im}). \tag{23}$$

Here the function $\text{pos}(x)$ returns $x$ if $x > 0$, or zero otherwise, and $\text{neg}(x)$ returns $|x|$ if $x < 0$, or zero otherwise. The ratio $G(\lambda)$ should be close to unity for all internal segments, and so an appropriate set of decision functions is

$$\left| 1 - G(\lambda_i) \right| \leq g \quad \text{for all } i = 1 : \ell \tag{24}$$

where $g$ is a small positive threshold. The same test is applied to the $j = (1 : m)$ segments between $v_{1j}$ and $v_{\ell j}$, in pencil $\mathcal{M}$.

The thresholds $f$ and $g$ in tests (21) and (24) were fixed once and for all in the experiments, such that no false-positive detections (which are unacceptable for calibration purposes) were made, across *all* data-sets. Furthermore, the same thresholds were used for all cameras.

## 3. Results

The new method is evaluated in two ways. Firstly, in section 3.1, the robustness of the gradient-labelling method to perspective distortion is investigated. Secondly, in section 3.2 the overall detection performance is compared, over a data-set of several hundred real images, to the most commonly-used (OpenCV) detection method.

It is important, at the outset, to clarify the nature of the evaluation. The ultimate objective of a board-detection algorithm, for calibration purposes, is to detect as many boards as possible, in the greatest variety of poses. In particular, *good coverage of the entire 3D scene-volume*, by the board vertices, is required for subsequent extrinsic calibration procedures. However, it is also essential that the vertices be *accurate*, in the sense of geometric error; in particular, there must be no 'false' detections or mis-labellings of the vertices. In practice, this means that the parameters of any detection algorithm must be set conservatively. Having done this, then any additional detection is beneficial for subsequent extrinsic calibration, provided that it does not increase the overall geometric error. The evaluation in 3.2 is based on large real-world calibration sets, which inevitably contain many problematic images (including those in which the board is not fully visible to one or more cameras). In order to make a real evaluation, there was no attempt to avoid or subsequently remove these images from the data-sets.

### 3.1. Geometric robustness

This section investigates whether the method can reliably identify two pencils of lines, in the presence of perspective distortion. In particular, it must be shown that the method does not require the midlines of the two pencils to be orthogonal. Ideally, the evaluation would be performed on a controlled image-set, showing a board at all possible orientations. In practice, it is very hard to obtain such an image-set, without using a mechanical mounting for the board. However, it is possible to accurately simulate the *geometric* effects of foreshortening, by applying a 2D homography to real image-data. This makes it straightforward to construct a uniform distribution of board orientations, including all slants $\varphi$. Furthermore, each classified pixel, labelled $\kappa_\varphi$, is *in correspondence* with the fronto-parallel 'best case', labelled $\kappa_0$. The latter can therefore be used as ground-truth measurements for the evaluation, as explained below. This procedure, importantly, allows us to isolate *geometric* effects from *sampling* effects, which leads to a better understanding.

In more detail, a single fronto-parallel (zero slant) image is processed as described in section 2.1, yielding a gradient vector $(\xi_0, \eta_0, 1)$, and a label $\kappa_0$ at each pixel. A rotation matrix $\boldsymbol{R}_\varphi = \boldsymbol{R}(\varphi, w(\vartheta)) \boldsymbol{R}(\omega, z)$ is then constructed, using angle-axis factors $\boldsymbol{R}(\cdot, \cdot)$. The rotation has the effect of *slanting* the board by an angle $\varphi$, around an axis $w(\vartheta)$. This axis is perpendicular to the optical axis $z$, with $\vartheta$ being the *tilt* angle. The second factor, $\boldsymbol{R}(\omega, z)$, is a cyclo-rotation around the optical axis $z$. The important variable is the slant angle, $\varphi$, which is systematically varied from zero (fronto-parallel) to $90°$. At each slant, the tilt and cyclorotation angles $\vartheta$ and $\omega$ are sampled 100 times from the uniform distribution on $[0, 2\pi]$.

A homography $\boldsymbol{H}_\varphi$ is now constructed from the rotation matrix $\boldsymbol{R}_\varphi$, and applied to the gradient-vectors, so that $(\xi_\varphi, \eta_\varphi, 1) = (\xi_0, \eta_0, 1)\boldsymbol{H}_\varphi^{-1}$. The transformed gradients are then re-classified, and the 'slanted' label $\kappa_\varphi$ is compared to the 'fronto-parallel'
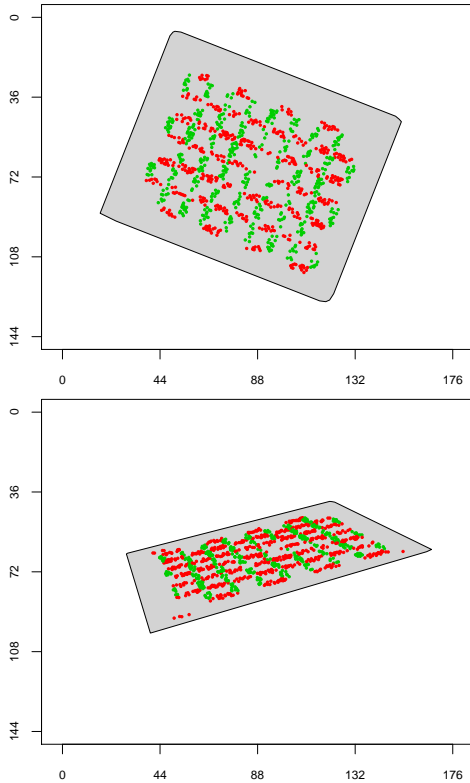
8

Figure 7: Example trials from the slant experiment. Top: A cyclo-rotation of the original gradient-vectors, which are easily classified (except for saturated highlight in the centre), as there is zero slant $\varphi$. Bottom: the classification is more difficult at a slant of $\varphi = 70°$. A total of 5000 gradient vectors were sampled, but only the labelled pixels are shown. The grey rectangles are the convex hulls of the 5000 samples. A total of 1200 trials were performed; the results are shown in fig. 8.

label $\kappa_0$, at each pixel. This can be visualized as a comparison between the two classifications in fig. 7. Consistency of the slanted classification is defined as the proportion of unchanged labels, over all edges. If $\delta(\kappa_\varphi, \kappa_0)$ is the indicator function that is 1 if $\kappa_\varphi = \kappa_0$, and 0 otherwise, then

$$\text{consistency} = \frac{\sum_{\{\kappa_0 \neq \varnothing\}} \delta(\kappa_\varphi, \kappa_0)}{\left|\{\kappa_0 \neq \varnothing\}\right|}$$

where $\{\kappa_0 \neq \varnothing\}$ is the set of edge-pixels (i.e. large gradients), and $\left|\{\kappa_0 \neq \varnothing\}\right|$ is the number of pixels in this set. This statistic is relatively strict, in that it ignores the easy task of classifying the non-edge pixels, for which $\kappa = \varnothing$.

The results of the experiment are shown in fig. 8, from which it can be seen that the labelling is robust to slants of more than 70°. This finding validates the basic detection principle, in relation to the geometry of perspective projection. However, in this experiment, the low-level image-processing was performed in the original fronto-parallel images. This was to avoid confounding geometric effects with sampling effects. A complete evaluation, using a real calibration data-set, is performed in the following section.
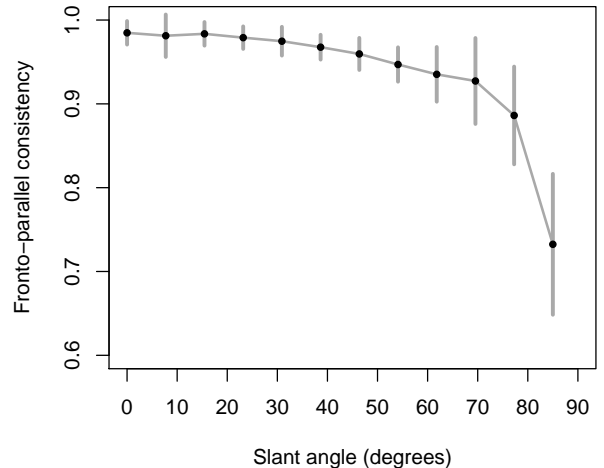


Figure 8: Edge-classification accuracy, over 1200 trials, as a function of slant-angle. The labelling becomes less accurate as the foreshortening increases, and eventually becomes undefined at 90° (when only the edge of the board can be seen). The labelling is robust to slants of more than 70°. The error bars show ±1 standard deviation per 100 trials; the variation is due to the random tilt-angle on each trial.

### 3.2. Comparison with heuristic methods

The method was formally tested on five multi-view data-sets, captured by Mesa Imaging Swiss-Ranger SR-4000 ToF cameras [19]. We used between two and four cameras per data-set (see table 1), all performing simultaneous capture. The sizes $u_1$ and $v_1$ of the Hough array were set to 1.5 times the average dimension, $(X + Y)/2$, of the input images.

We compare our results to the widely-used OpenCV detector [15]. Both the OpenCV and Hough detections were refined by the OpenCV subpixel routine, which adjusts the given point to minimize the discrepancy with the image-gradient around the chequerboard corner [15, 16]. Specifically, if the true vertex has coordinates $(x_0, y_0)$, and $(\xi_{xy}, \eta_{xy})$ is the image-gradient at a nearby point $(x, y)$, then

$$(\xi_{xy}, \eta_{xy}) \cdot (x - x_0, y - y_0) \approx 0. \qquad (25)$$

This is because if the magnitude of $(\xi_{xy}, \eta_{xy})$ is not negligible, then $(x, y)$ must be *on* a black-white edge. Hence $(\xi_{xy}, \eta_{xy})$ must be perpendicular to the edge, while $(x - x_0, y - y_0)$ must be parallel to it.

Table 1 shows the number of true-positive detections by each method, as well as the number of detections common to both methods. The geometric error is the discrepancy from the 'ideal' board, after fitting the latter by the optimal homography (initialized by the DLT method, and refined by Levenberg-Marquardt optimization [10]). This is by far the most useful measure, as it is directly related to the role of the detected vertices in subsequent calibration algorithms (e.g. bundle-adjustment [10]), and also has a simple interpretation in pixel-units. The photometric error is the RMS gradient residual (25) computed over a $5 \times 5$ window. This measure is worth considering, because it is the criterion minimized by the subpixel optimization, but it is much less useful than the geometric error.

| Set / Camera | Number detected | | | Geometric error | | Photometric error | |
|---|---|---|---|---|---|---|---|
| | OCV | HT | Both | OCV | HT | OCV | HT |
| 1 / 1 | 19 | 34 | 13 | 0.2263 | 0.1506 | 0.0610 | 0.0782 |
| 1 / 2 | 22 | 34 | 14 | 0.1819 | 0.1448 | 0.0294 | 0.0360 |
| 1 / 3 | 46 | 33 | 20 | 0.1016 | 0.0968 | 0.0578 | 0.0695 |
| 1 / 4 | 26 | 42 | 20 | 0.2044 | 0.1593 | 0.0583 | 0.0705 |
| 2 / 1 | 15 | 27 | 09 | 0.0681 | 0.0800 | 0.0422 | 0.0372 |
| 2 / 2 | 26 | 21 | 16 | 0.0939 | 0.0979 | 0.0579 | 0.0523 |
| 2 / 3 | 25 | 37 | 20 | 0.0874 | 0.0882 | 0.0271 | 0.0254 |
| 3 / 1 | 14 | 26 | 11 | 0.1003 | 0.0983 | 0.0525 | 0.0956 |
| 3 / 2 | 10 | 38 | 10 | 0.0832 | 0.1011 | 0.0952 | 0.1057 |
| 3 / 3 | 25 | 41 | 21 | 0.1345 | 0.1366 | 0.0569 | 0.0454 |
| 3 / 4 | 18 | 23 | 10 | 0.1071 | 0.1053 | 0.0532 | 0.0656 |
| 4 / 1 | 16 | 21 | 14 | 0.0841 | 0.0874 | 0.0458 | 0.0526 |
| 4 / 2 | 45 | 53 | 29 | 0.0748 | 0.0750 | 0.0729 | 0.0743 |
| 4 / 3 | 26 | 42 | 15 | 0.0954 | 0.0988 | 0.0528 | 0.0918 |
| 5 / 1 | 25 | 37 | 18 | 0.0903 | 0.0876 | 0.0391 | 0.0567 |
| 5 / 2 | 20 | 20 | 08 | 0.2125 | 0.1666 | 0.0472 | 0.0759 |
| 5 / 3 | 39 | 36 | 24 | 0.0699 | 0.0771 | 0.0713 | 0.0785 |
| 5 / 4 | 34 | 35 | 19 | 0.1057 | 0.1015 | 0.0519 | 0.0528 |
| 6 / 1 | 29 | 36 | 20 | 0.1130 | 0.1203 | 0.0421 | 0.0472 |
| 6 / 2 | 35 | 60 | 26 | 0.0798 | 0.0803 | 0.0785 | 0.1067 |
| Mean: | **25.75** | **34.8** | 16.85 | **0.1157** | **0.1077** | 0.0547 | 0.0659 |

Table 1: Results over six multi-ToF camera-setups. Total detections for the OpenCV ($N = 515$) vs. Hough Transform ($N = 696$) method are shown, as well as the accuracy of the estimates. Geometric error is in pixels. The chief conclusion is that the HT method detects 35% more boards, and slightly reduces the average geometric error. The increased number and variety of detected boards is very beneficial for extrinsic calibration procedures, which require the input-points to be spread throughout the entire scene-volume.

The proposed Hough-based method detects 35% more boards than the OpenCV method, on average. There is also a slight reduction in average geometric error, even though the additional boards were more problematic to detect. Even if OpenCV had detected these boards, it is likely that their inclusion would have increased the geometric error even further. It may also be noted that the new method is *dramatically* better in five cases, in the sense that it detects an additional ten or more boards, while also *reducing* the geometric error. These results should not be surprising, because the new method uses a very strong model of the global board-geometry.

There were zero false-positive detections (100% precision), as explained in sec. 2.5. The number of true-negatives is not useful here, because it depends largely on the configuration of the cameras (i.e. how many images show the back of the board). The false-negatives do not provide a very useful measure either, because the definition of these would depend on an arbitrary judgement about which of the very foreshortened boards 'ought' to have been detected (i.e. whether an edge-on board is 'in' the image or not). It is emphasized that the test-data are actual calibration sets, containing many problematic images, and so the evaluation is based in a real-world application. Some example detections are shown in figs. 9–11, including a variety of difficult cases.

## 4. Discussion

A new method for the automatic detection of calibration grids in time-of-flight images has been described. The method is
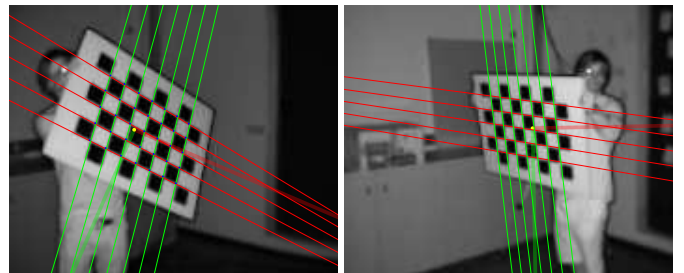


Figure 9: Example detections in $176 \times 144$ ToF amplitude images. The yellow dot (one-pixel radius) is the estimated centroid of the board, and the attached thick translucent lines are the estimated axes. The board on the right, which is relatively distant and slanted, was not detected by OpenCV.
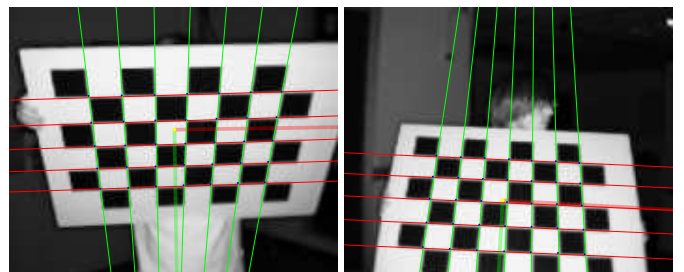


Figure 10: Example detections (cf. fig. 9) showing significant perspective effects.

based on careful reasoning about the global geometric structure of the board, before and after perspective projection. The
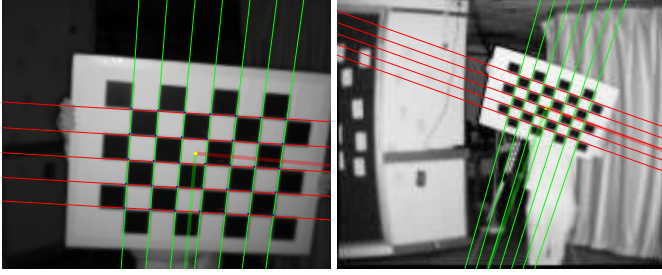
Figure 11: Example detections (cf. fig. 9) showing significant scale changes. The board on the right, which is in an image that shows background clutter and lens distortion, was not detected by OpenCV.

method detects many more boards than existing heuristic approaches, which results in a larger and more complete data-set for subsequent calibration algorithms. This is achieved while also *reducing* the overall geometric error. The increased number and variety of detections is of great benefit to subsequent extrinsic calibration procedures.

### 4.1. Future work

The Hough transform was developed, in section 2.4, as a dense 2-D array. This presentation has the advantage of making the method easy to visualize and implement. However, it also raises issues of resolution and scalability [39, 40]. In particular, the implementation in section 2.4 is inefficient, both in space and time. These issues could be addressed by the use of a *randomized* Hough transform [41]. This approach avoids building a dense transform array, in favour of a dynamic data structure [42]. Future work will examine the advantages of randomized methods, in relation to the increased complexity of implementation. Another possible direction for future work would be to perform a global refinement of the line-pencils, in the geometric parameterization, but by minimizing a photometric cost-function with respect to the original images.

In a more general view, the present work suggests that it would be useful to have a data-set of calibration images with known 3D poses. These could, for example, be acquired using a robotic mounting of the physical board. This would enable a fine-grained comparison of algorithms, thereby encouraging future work in this area.

### References

[1] M. Hansard, S. Lee, O. Choi, R. Horaud, Time-of-Flight Cameras: Principles, Methods and Applications, Springer, 2013.

[2] J. M. Dubois, H. Hügli, Fusion of Time of Flight Camera Point Clouds, ECCV Workshop on Multi-Camera and Multi-modal Sensor Fusion Algorithms and Applications (2008).

[3] I. Schiller, C. Beder, R. Koch, Calibration of a PMD camera using a planar calibration object together with a multi-camera setup, in: Int. Arch. Soc. Photogrammetry, Remote Sensing and Spatial Information Sciences XXI, 2008, pp. 297–302.

[4] J. Zhu, L. Wang, R. G. Yang, J. Davis, Fusion of time-of-flight depth and stereo for high accuracy depth maps, in: Proc. CVPR, 2008, pp. 1–8.

[5] U. Hahne, M. Alexa, Depth Imaging by Combining Time-of-Flight and On-Demand Stereo, in: Proc. DAGM Workshop on Dynamic 3D Imaging, 2009, pp. 70–83.

[6] R. Koch, I. Schiller, B. Bartczak, F. Kellner, K. Köser, MixIn3D: 3D Mixed Reality with ToF-Camera, in: Proc. DAGM Workshop on Dynamic 3D Imaging, 2009, pp. 126–141.

[7] A. Kolb, E. Barth, R. Koch, R. Larsen, Time-of-Flight Cameras in Computer Graphics, Computer Graphics Forum 29 (1) (2010) 141–159.

[8] M. Hansard, R. Horaud, M. Amat, S. Lee, Projective Alignment of Range and Parallax Data, in: Proc. IEEE CVPR, 2011, pp. 3089–3096.

[9] Z. Zhang, A flexible new technique for camera calibration, IEEE Trans. PAMI 22 (11) (2000) 1330–1334.

[10] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University Press, 2000.

[11] V. Douskos, I. Kalisperakis, G. Karras, E. Petsa, Fully Automatic Camera Calibration using Regular Planar Patterns, in: Int. Arch. Soc. Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. 37(5), 2008, pp. 21–26.

[12] C. Harris, M. Stephens, A combined corner and edge detector, in: Proc. 4th Alvey Vision Conference, 1988, pp. 147–151.

[13] L. Krüger, C. Wöhler, A. Würz-Wessel, F. Stein, In-factory calibration of multiocular camera systems, in: SPIE Photonics Europe, 2004, pp. 126–137.

[14] Z. Wang, W. Wu, X. Xu, D. Xue, Recognition and location of the internal corners of planar checkerboard calibration pattern image, Applied Mathematics and Computation 185 (2) (2007) 894–906.

[15] G. Bradski, A. Kaehler, Learning OpenCV, O'Reilly, 2008.

[16] A. Datta, K. Jun-Sik, T. Kanade, Accurate camera calibration using iterative refinement of control points, in: Workshop on Visual Surveillance, Proc. ICCV 2009, 2009, pp. 1201–1208.

[17] M. Lindner, I. Schiller, A. Kolb, R. Koch, Time-of-flight sensor calibration for accurate range sensing, Computer Vision and Image Understanding 114 (12) (2010) 1318–1328.

[18] B. Buttgen, P. Seitz, Robust optical time-of-flight range imaging based on smart pixel structures, IEEE Transactions on Circuits and Systems I: Regular Papers 55 (6) (2008) 1512–1525.

[19] Mesa Imaging AG, Swiss Ranger, http://www.mesa-imaging.ch/ (2010).

[20] S. Foix, G. Alenya, C. Torras, Lock-in Time-of-Flight (ToF) Cameras: A Survey, IEEE Sensors 11 (9) (2011) 1917–1926.

[21] S.-J. Kim, J. D. Kim, B. Kang, K. Lee, A CMOS image sensor based on unified pixel architecture with time-division multiplexing scheme for color and depth image acquisition, IEEE Journal of Solid-State Circuits 47 (11) (2012) 2834–2844.

[22] C. Niclass, M. Soga, H. Matsubara, S. Kato, M. Kagami, A 100-m range 10-frame/s 340 96-pixel time-of-flight depth sensor in 0.18-CMOS, IEEE Journal of Solid-State Circuits 48 (2) (2013) 559–572.

[23] J.-E. Ha, Automatic detection of calibration markers on a chessboard, Optical Engineering 46 (10) (2007) 107203.

[24] J.-E. Ha, Automatic detection of chessboard and its applications, Optical Engineering 48 (6) (2009) 067205.

[25] S. T. Barnard, Interpreting perspective images, Artificial Intelligence 21 (1983) 435–462.

[26] J. A. Shufelt, Performance evaluation and analysis of vanishing point detection techniques, IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (3) (1999) 282–288.

[27] L. Quan, R. Mohr, Determining perspective structures using hierarchical Hough transform, Pattern Recognition Letters 9 (4) (1989) 279–286.

[28] J. Košecká, W. Zhang, Video compass, in: European Conference on Computer Vision, Springer, 2006, pp. 476–490.

[29] R. Toldo, A. Fusiello, Robust multiple structures estimation with j-linkage, in: European Conference on Computer Vision, Springer, 2008, pp. 537–547.

[30] J. P. Tardif, Non-iterative approach for fast and accurate vanishing point detection, in: IEEE International Conference on Computer Vision, 2009, pp. 1250–1257.

[31] M. Antunes, J. P. Barreto, A global approach for the detection of vanishing points and mutually orthogonal vanishing directions, in: IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 1336–1343.

[32] J. Illingworth, J. Kittler, A survey of the Hough transform, Computer Vision, Graphics and Image Processing 44 (1988) 87–116.

[33] F. Bukhari, M. Dailey, Robust radial distortion from a single image, in: Proc. Int. Conf. on Advances in Visual Computing, 2010, pp. 11–20.

11

[34] D. Gonzalez-Aguilera, J. Gomez-Lahoz, P. Rodriguez-Gonzalvez, An automatic approach for radial lens distortion correction from a single image, IEEE Sensors 11 (4) (2011) 956–965.

[35] R. Melo, M. Antunes, J. Barreto, G. Falcao, N. Gonçalves, Unsupervised intrinsic calibration from a single frame using a plumb-line approach, in: IEEE International Conference on Computer Vision, 2013, pp. 537–544.

[36] A. de la Escalera, J. Armingol, Automatic Chessboard Detection for Intrinsic and Extrinsic Camera Parameter Calibration, Sensors 10 (2010) 2027–2044.

[37] T. Tuytelaars, M. Proesmans, L. V. Gool, The Cascaded Hough Transform as Support for Grouping and Finding Vanishing Points and Lines, in: Proc. International Workshop on Algebraic Frames for the Perception-Action Cycle, 1997, pp. 278–289.

[38] G. Granlund, In search of a general picture processing operator, Computer Graphics and Image Processing 8 (1978) 155–173.

[39] M. Zhang, On the discretization of parameter domain in hough transformation, in: Proc. Int. Conf. on Pattern Recognition (ICPR), 1996, pp. 527–531.

[40] L. Guo, O. Chutatape, Influence of discretization in image space on hough transform, Pattern Recognition 32 (4) (1999) 635–644.

[41] H. Kälviäinen, P. Hirvonen, L. Xu, E. Oja, Probabilistic and non-probabilistic Hough transforms: overview and comparisons, Image and Vision Computing 13 (4) (1995) 239–252.

[42] L. Xu, E. Oja, P. Kultanen, A new curve detection method: Randomized hough transform (rht), Pattern Recognition Letters 11 (5) (1990) 331–338.