CIRCULAR MARKERS FOR CAMERA POSE ESTIMATION

Alain Pagani, Johannes Koehler and Didier Stricker

Augmented Vision - DFKI

ABSTRACT

This papers presents a new system using circular markers to estimate the pose of a camera. Contrary to most markers-based systems using square markers, we advocate the use of circular markers, as we believe that they are easier to detect and provide a pose estimate that is more robust to noise. Unlike existing systems using circular markers, our method computes the exact pose from one single circular marker, and do not need specific points being explicitly shown on the marker (like center, or axes orientation). Indeed, the center and orientation is encoded directly in the marker's code. We can thus use the entire marker surface for the code design. After solving the back projection problem for one conic correspondence, we end up with two possible poses. We show how to find the marker's code, rotation and final pose in one single step, by using a pyramidal cross-correlation optimizer. The marker tracker runs at 100 frames/second on a desktop PC and 30 frames/second on a hand-held UMPC.

1. INTRODUCTION

Markers can be used in many applications, like *e.g.* Augmented Reality (see Figure 1). Since the early years of marker-based pose estimation, several marker designs and pose estimation approaches have been proposed. Among them, a vast majority uses square markers [1, 2, 3, 4], as the pose estimation algorithms for points or lines are easier to implement and to maintain. However, we believe that the use of circular markers has several advantages over square markers. First, the perspective projection of a circle is an ellipse, which is easy to detect and to fit in the image. Second, the pose estimation of a square marker relies on a good estimation of only four points (the four corners) whereas the whole ellipse can be used in the case of circular markers. Third, the case of occlusion is easily treated for circular markers, as an ellipse fitting algorithm does not require the entire contour to be visible.

In this paper, we present our circular markers, and show how to use them to compute the pose of the camera. The novelty of our approach resides in the fact that our markers are defined only by their circular contour and a 16-bit code (see Figure 2). No indication of the marker's center nor the marker orientation on its underlying plane is needed. This was made possible thanks to our new code extraction strategy, which solves several issues at the same time: identification of the marker through its code, disambiguation in a small set of hypothetical poses and extraction of the correct local orientation. These steps are completed efficiently thanks to a pyramidal search of the best cross-correlation among possible marker codes.

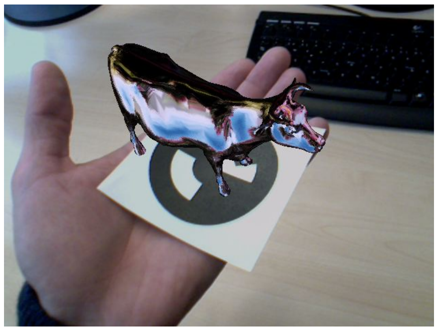

Fig. 1. Circular marker for Augmented Reality

Related work. Most systems use square markers [2, 1, 4, 3]. To the best of our knowledge, only two systems use circular markers: Intersense's VIS-Tracker ([5]) and a subpart of Cantag [3]. In [5], the system treats the markers as single points, and at least four markers have to be recognized in an image to recover the camera pose. Moreover, the marker center is approximated by the center of the ellipse. The subsequent error can be neglected if the markers are small, but leads to wrong poses when markers are seen in close up scenes. On the contrary, our method can recover the pose with any number of visible markers, and we use the correctly projected center of the marker to recover the pose. Moreover the marker design proposed by Foxlin et al. explicitly shows the center of the marker as a white disk, and the two axes of the ellipse are given by black disks ("eyes" in the marker). This inevitably reduces the payload of the marker, as less space is left for the marker's code.

The use of circular markers is also possible in [3]. The marker design has a white disk in the center, which is used to recover the camera pose. Cantag has been specified to compare different markers designs, and the authors suggest that circular markers give better results than square ones. Several methods have been proposed to calibrate a camera and compute its pose from circular patterns. In [6], a camera is calibrated using two concentric circles. [7] use the image of the absolute conic to compute the pose from one imaged circle. However, the center of the circle has to be provided. In [8],

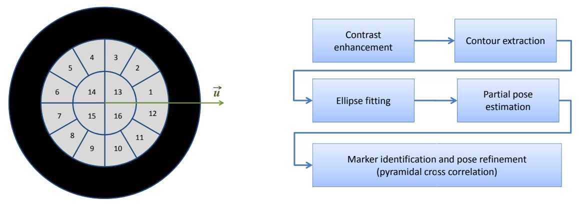

Fig. 2. (left) Marker design. (right) Our pipeline

two concentric circles and a extra orientation mark are necessary to find the pose of the camera. [9] proposes the computation of the camera pose and calibration from the image of circles, but at least two circles have to be visible in the image. In [10], a camera is calibrated using two coplanar circles. Our approach is different from these, in that we use only one circle without explicit orientation nor center to compute the camera pose. The main idea is that a one-dimensional family of possible poses can be computed from one single ellipse. We then find the correct pose in this family while identifying the marker.

The remainder of this paper is organized as follows: Section 4 presents the pose estimation algorithm from one marker. Section 3 shows how the correct marker identification is found along with the final pose and rotation. In Section 4 we show how ton compute the pose of a set of markers. We show our results in Section 5 before concluding.

2. POSE ESTIMATION FROM ONE MARKER

Marker definition. Our marker is a black circle and consists on three rings with equal width. The outer ring is always black and serves for the marker localization. The inner ring is divided into 4 angular bins and the middle ring into 12 angular bins. Each bin can be white or black. Thus, the code can store 16 bits of information. When designing markers, care has to be taken that the code does not have a periodicity (other than 2Π), as the marker orientation $\boldsymbol{u}$ has to be found from the code itself. Figure 2 shows the marker with its 16 bits and the principal orientation $\boldsymbol{u}$.

Image processing pipeline. In this section, we show how we can recover the marker pose pose from one single view of the marker. Figure 2 shows the image processing pipeline we apply to each video frame. In order to better distinguish the markers even under difficult light conditions, we first apply a contrast enhancement function to the image, based on homomorphic image processing [5]. Edges in the image are then extracted using a Canny filter with non maximal suppression (NMS). A SVD-based ellipse fitting algorithm is applied to all found closed contours, and we keep only the found ellipses for which the error returned by the fitting function is small. For each remaining ellipse, we then apply first a partial pose computation, then a marker identification algorithm. These two steps are further detailed in the next subsections.

Dimensionality reduction from one imaged circle. We now show how to reduce the pose computation to a one dimensional problem when the image of a circle has been found. In this derivation, we follow the work of [10], where further details can be found. We assume that the camera follows the pinhole camera model and is calibrated (the calibration matrix $\boldsymbol{K}$ is known). Without loss of generality, we can therefore assume that the image plane has equation $Z = 1$ in the camera coordinate system and that the image is centered on the principal point (see Figure 3). We define the Marker Coordinate System (MCS) as centered on the marker center, having a Z axis perpendicular to the supporting plane and a X axis defined by the marker's orientation $\boldsymbol{u}$. The Intermediate Coordinate System (ICS) is by definition a translation of the MCS so that the origin is at the camera optical center (see Figure 3).

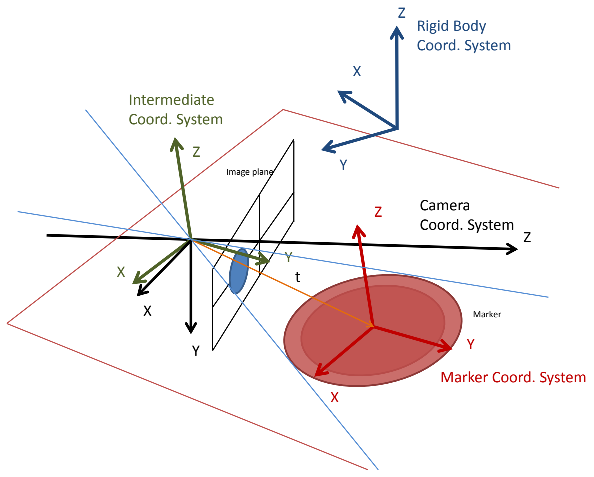

Fig. 3. Two views of the oblique elliptical cone

Let us define the unknown transformation from the MCS to the CCS by the rotation matrix and the translation vector $(\boldsymbol{R}, \boldsymbol{t})$, such as a point $\boldsymbol{X}_M$ in the MCS transforms to $\boldsymbol{X}_C$ in the CCS as

$$\boldsymbol{X}_C = \boldsymbol{R}\boldsymbol{X}_M + \boldsymbol{t} \tag{1}$$

We will now show how to retrieve the correct $\boldsymbol{R}$ and $\boldsymbol{t}$ from image measurements. The idea of the derivation is that the oblique cone defined by the bundle of straight lines (*i.e.* the light rays) passing through the optical center, the ellipse and the circle can be described in two different manners, once in the CCS, once in the ICS (see [10] for details).

An ellipse found in the image is defined by a symmetric matrix $\boldsymbol{C}$. In the CCS, a point $\boldsymbol{X}_C$ is on the cone if

$$\boldsymbol{X}_C^{\mathrm{T}}\boldsymbol{C}\boldsymbol{X}_C = 0 \tag{2}$$

In the ICS, a 3D point $\boldsymbol{X}_I$ is on the cone if

$$\boldsymbol{X}_I^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{X}_I = 0 \tag{3}$$

where

$$\boldsymbol{Q} = \begin{pmatrix} 1 & 0 & -x_0/z_0 \\ 0 & 1 & -y_0/z_0 \\ -x_0/z_0 & -y_0/z_0 & (x_0^2 + y_0^2 - r^2)/z_0^2 \end{pmatrix} \tag{4}$$

and $\boldsymbol{t} = (x_0, y_0, z_0)^{\mathrm{T}}$. By construction, the rotation matrix $\boldsymbol{R}$ transforms $\boldsymbol{X}_C$ to $\boldsymbol{X}_I$ as following

$$\boldsymbol{X}_C = \boldsymbol{R}\boldsymbol{X}_I \tag{5}$$

so that $\boldsymbol{C}$ and $\boldsymbol{Q}$ are related by following equation

$$\mu \boldsymbol{R}^{\mathrm{T}} \boldsymbol{C} \boldsymbol{R}_c = \boldsymbol{Q} \tag{6}$$

where μ is a scale factor. Equation (6) is solved using the eigendecomposition of the matrix $\boldsymbol{C}$: if $\lambda_1, \lambda_2, \lambda_3$ are the (sorted) eigenvalues and $\boldsymbol{V}$ the matrix of eigenvectors, then the one-dimensional family of solutions $(\boldsymbol{R}, \boldsymbol{t})$ is ([10]):

$$\begin{aligned} \boldsymbol{R}(S_1, S_2, S_3, \alpha) &= \boldsymbol{V} \begin{pmatrix} g\cos\alpha & S_1 g \sin\alpha & S_2 h \\ \sin\alpha & -S_1\cos\alpha & 0 \\ S_1 S_2 h \cos\alpha & S_2 h \sin\alpha & S_1 g \end{pmatrix} \\ \boldsymbol{t}(S_1, S_2, S_3, \alpha) &= \begin{pmatrix} -S_2 S_3 \sqrt{\frac{(\lambda_1-\lambda_2)(\lambda_2-\lambda_3)}{-\lambda_1\lambda_3}}\, r\cos\alpha \\ -S_1 S_2 S_3 \sqrt{\frac{(\lambda_1-\lambda_2)(\lambda_2-\lambda_3)}{-\lambda_1\lambda_3}}\, r\sin\alpha \\ S_3 \frac{\lambda_2 r}{\sqrt{-\lambda_1\lambda_3}} \end{pmatrix} \end{aligned} \tag{7}$$

where α is a free variable, S_1, S_2 and S_3 are undetermined (discrete) signs, and g and h are defined by

$$\begin{aligned} g &= \sqrt{\tfrac{\lambda_2-\lambda_3}{\lambda_1-\lambda_3}} \\ h &= \sqrt{\tfrac{\lambda_1-\lambda_2}{\lambda_1-\lambda_3}} \end{aligned} \tag{8}$$

The free variable α reflects the rotation invariance of the oblique cone, and fixing α boils down to fixing the rotation of the marker around its normal. We chose to fix $\alpha = 0$ and to find the true rotation of the marker in a later step. We further reduce the number of solutions by stating that among the 8 possible combinations of the signs S_i, exactly two meet the visibility constraints (the center of the marker must be in front of the camera, *i.e.* in the CCS $\boldsymbol{c}_z > 0$ *and* the normal of the marker must be oriented towards the camera *i.e.* in the CCS $\boldsymbol{n}_z < 0$). We end up with exactly two possible poses $(\boldsymbol{R}^1, \boldsymbol{t}^1)$ and $(\boldsymbol{R}^2, \boldsymbol{t}^2)$ for each ellipse.

3. MARKER IDENTIFICATION

In order to identify the markers among the found ellipses, we use a one-dimensional Normalized Cross Correlation between the expected true markers and the image pixel values. In our application, the user can define which markers are used by the means of an editable XML file. For each ellipse E_j, we go successively over the 2 poses $(\boldsymbol{R}_j^k, \boldsymbol{t}_j^k)$, $k \in [1, 2]$, and then for each expected marker $\boldsymbol{M}_i$ we compute the normalized cross correlation $NCC(i, j, k, \alpha)$ by sampling pixel values on points regularly distributed over two rings on the marker. We find the position of these points in the image using the transformation $(\boldsymbol{R}_j^k, \boldsymbol{t}_j^k)$ and the camera intrinsic matrix $\boldsymbol{K}$. Note that we know the pose of the marker only up to a rotation around its normal, so that we have to sample several times using different values for the starting rotation angle α. Sampling for a sufficient number of starting angles α would require a prohibitive amount of computation time, and we therefore suggest to use a pyramidal normalized cross correlation approach as follows: i, j and k being fixed, we compute $NCC(\alpha)$ for 12 different values: α_0 to α_{11}, starting at zero and increasing with an angular increment $\delta = 2\pi/12$. We keep $\alpha_{max} = argmax_{\alpha_p}\ NCC(\alpha_p)$ and sample again with 5 values for α around α_{max} using half of the increment δ. We repeat the last step, halving each time the increment δ until the computed NCC value does not change anymore. This pyramidal cross correlation is much faster than an exhaustive search and proves to produce the same results in terms of angular accuracy.

For each ellipse E_j-marker $\boldsymbol{M}_i$ combination, we compute the score s_{ij} as the maximal NCC over all possible poses k and angles α. This gives us a correlation matrix between ellipses and markers that we use as input to an assignment algorithm. In practice, we use the Hungarian assignment method, and reject the ellipses whose score is below a given threshold as non-marker. As a result, we have for each marker $\boldsymbol{M}_i$ in the definition file: (1) a one-bit answer to the question *is the marker visible in the image* and (2) if the answer is yes, the pose of the marker $(\boldsymbol{R}_i, \boldsymbol{t}_i)$ as defined in equation (1), computed as the concatenation of the ellipse pose $(\boldsymbol{R}_j^{k_{max}}, \boldsymbol{t}_j^{k_{max}})$ and the rotation around the Z axis with the angle α_{max}.

4. POSE OF A SET OF MARKERS

In the previous section, we have shown how to compute the pose of one single marker. Here, we address the more complex case where several markers are defined in the same rigid body. A rigid body is a set of markers with a fixed relative position over time. Our system allows any combination of markers, including rigid bodies with markers lying on different planes and any number of markers per plane. We propose to compute one global pose using all the available data (in form of circle-ellipse correspondences). Our algorithm is as follows: first, for each plane in a rigid body, we compute a homography between the 3D plane and the imaged plane. Second, we use the homographies from all the planes in a rigid body for computing the global pose of the rigid body.

Homography of an imaged plane from circular markers. From the marker definition file, we can deduce which markers belong to the same 3D plane. For each plane of a rigid body, we count the number of visible markers on that plane.If one marker only is visible in the image, then we use the retrieved pose $(\boldsymbol{R}_{i,k}, \boldsymbol{t}_{i,k})$ to compute the plane homography $\boldsymbol{H}$. If more than one marker is visible, we use the method of [9] to compute the homography of the plane from conic correspondences. Note that we use two different algorithms depending on the number of correspondences (see [9] for details). In case of multiple solutions, we apply visibility constraints and choose the remaining solutions that minimizes the reprojection error (reprojection of the marker center).

Rigid body pose from planar homographies. If only one plane has visible markers within the rigid body, it is straightforward to recover the global pose from the homography, using the fact that if the plane has equation $Z = 0$, then the homography is equal to the matrix $\boldsymbol{K}\,[\boldsymbol{r}_1|\boldsymbol{r}_2|\boldsymbol{t}]$, where $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are the two first columns of the final rotation matrix $\boldsymbol{R}$. When several planes have been found with visible marker(s), we apply following multiple plane pose estimation algorithm: Let $\boldsymbol{T}$ be a transformation matrix transforming a given plane to a plane with equation $Z = 0$ and $\langle\boldsymbol{T}\rangle$ the same matrix without its third column. Then we have the homogeneous equation:

$$[\boldsymbol{R} \mid \boldsymbol{t}]\langle\boldsymbol{T}\rangle\boldsymbol{H}^{-1}\boldsymbol{K} \sim \boldsymbol{I}_{3\times 3}$$

where $\boldsymbol{I}_{3\times 3}$ is the 3×3 identity matrix.
Calling $\boldsymbol{d}_i, i = 1...3$ the columns of the matrix $\langle\boldsymbol{T}\rangle\boldsymbol{H}^{-1}\boldsymbol{K}$ and $\boldsymbol{q}_i^{\mathrm{T}}$ the (unknown) rows of the matrix$[\boldsymbol{R} \mid \boldsymbol{t}]$, we have:

$$\begin{pmatrix} \boldsymbol{q}_1^{\mathrm{T}} \\ \boldsymbol{q}_2^{\mathrm{T}} \\ \boldsymbol{q}_3^{\mathrm{T}} \end{pmatrix} \begin{pmatrix} \boldsymbol{d}_1 & \boldsymbol{d}_2 & \boldsymbol{d}_3 \end{pmatrix} \sim \boldsymbol{I}_{3\times 3}$$

This last 3×3 matrix homogeneous equation leads to 8 equations in the elements of the $\boldsymbol{q}_i$'s. This system can be solved from 2 planes on.

5. RESULTS

Tracking one marker. In order to assess the developed theoretical framework, we implemented the algorithms on a standard PC. Figure 4 shows the results of tracking one marker over 500 consecutive frames. The translation vector is provided in mm and the rotation by means of Euler angles. From the charts, we can see that the tracking is smooth and regular, and we can track the pattern from a distance of 200 mm to 750 mm (the marker radius was 35 mm in this experiment).

Time consumption. Our marker tracking algorithm works in a tracking-by-detection paradigm. This means that we search for all the markers and compute the pose independently in each frame, without taking the last frames into consideration. However, thanks to the fast ellipse detection method and to the linear algorithms for pose computation, the tracker is very fast. We achieve about 100 frames per second on a standard PC and 30 frames per second on a small Ultra Mobile PC.

6. CONCLUSION

In this paper, we presented a new marker design for marker-based applications. This is the first circular marker design

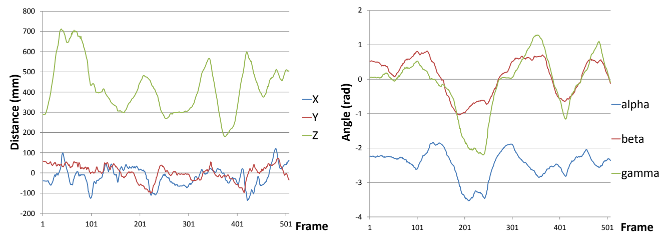

Fig. 4. Translation and rotation over 500 frames

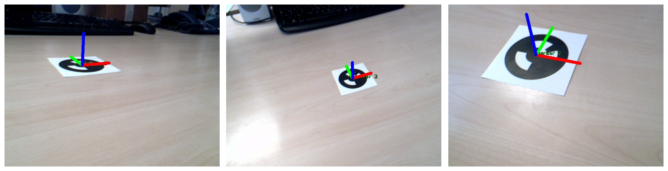

Fig. 5. Examples of tracked frames

where the pose can be computed from one single marker. Our method relies on a pose estimation from conic correspondences and is accelerated by a pyramidal normalized cross correlation algorithm. In case several markers are used in a rigid setup, we have shown how to compute the global pose using all conics in correspondence. Our system is fast and robust, and can be used as alternative to square markers in marker-based augmented reality applications.

Acknowledgment. This work has been partially funded by the project CAPTURE (01IW09001) and the German BMBF project AVILUSplus (01M08002).

7. REFERENCES

[1] H. Kato, M. Billinghurst, S. Weghorst, and T. Furness, "A mixed reality 3d conferencing application," Tech. Rep., HIT lab, UW, 1999.

[2] D. Schmalstieg, A. Fuhrmann, G. Hesina, Z. Szalavari, L.M. Encarnacao, M. Gervautz, and W. Purgathofer, "The studierstube augmented reality project," *Presence: Teleoperators & Virtual Environments*, vol. 11, no. 1, pp. 33–54, 2002.

[3] A.. Rice, A. Beresford, and R. Harle, "Cantag: an open source software toolkit for designing and deploying marker-based vision systems," in *Pervasive Computing and Communications*, 2006.

[4] M. Fiala, "Artag, a fiducial marker system using digital techniques," in *CVPR*, 2005.

[5] L. Naimark and E. Foxlin, "Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker," in *ISMAR*, 2002.

[6] J. Kim, H. Kim, and I. Kweon, "A camera calibration method using concentric circles for vision applications," in *ACCV*, 2002.

[7] G. Wang, Q. Wu, , and Z. Ji, "Pose estimation from circle or parallel lines in a single image," in *ACCV*, 2007.

[8] F. Abad, E. Camahort, and R. Viv, "Camera calibration using two concentric circles," in *ICIAR*, 2004.

[9] J. Kannala, M. Salo, and J. Heikkil, "Algorithms for computing a planar homography from conics in correspondence," in *BMVC*, 2006.

[10] Q. Chen, H. Wu, and T. Wada, "Camera calibration with two arbitrary coplanar circles," in *ECCV*, 2004.