 ISBN 978-3-200-02588-2

INITIALIZING 3D MODEL-BASED TRACKING OF DICE

Fuensanta Torres, Walter G. Kropatsch, and Nicole M. Artner*

PRIP, Vienna University of Technology, Austria

ABSTRACT

The initialization process of a 3D model-based tracking method with an extremely fast checking time is presented. One initialized, the algorithm will execute a 3D model-based tracking with pose determination. The tracking method is a two-steps approach, a prediction-correction method. We are studying the possibility of finding the minimum group of interest points and a set of rules which form all the possible movements of a 3D tracked object. Thereby, the computation time-memory for the tracking task could be reduced. We consider rigid objects which can be approximately modeled by a convex polyhedral shape. Our approach works in monocular videos and where the position of the camera is fixed.

Index Terms— initialization, tracking, pose estimation, interest points

1. INTRODUCTION

Shape appearance and motion of real-world objects obey the laws of physics and also their changes happen in a smooth manner. This implies that there exists a strong correlation in the temporal evolution of image content. Therefore, in our approach we plan to use a 3D model of the object to deal with changes in appearance due to the movement of this object in front of the camera. It integrates a 3D model-based tracking, as described in [1] and pose determination. Furthermore, we are studying the possibility of finding the minimum group of key characteristics and a set of rules which form all the possible movements of a 3D tracked object.

1.1. Related work

Point tracking is one of the earliest examples of computer vision tracking. Although after the early 1990s, the visual tracking of global features became more popular than the methods that use localized features. Nowadays, the point trackers are reemerging. These techniques can use a collections of interest points to follow higher level objects [2]. The methods that use just simple points instead of higher level objects, which has several advantages. They matching individual interest points across images and they provide advantages to deal with partial occlusions, matching errors and illumination invariance. Moreover, they allow fast motions [3]. This approach requires an external mechanism to detect the objects in every frame. The most common point-based tracking methods are: multiple hypothesis tracking (MHT), hidden Markov models (HMM), artificial neural network (ANN), particle filter, Kalman filtering (KF) and mean shift (MF), according to [4]. We are working on the possibility of building a novel feature-based tracking algorithm, which could be computationally cheaper both in time and memory than the actual feature-based methods.

1.2. Method

Our approach requires an initialization process. One initialized, the algorithm executes a 3D model-based tracking with pose determination, as proposed in [5]. Unlike of [5] the pose computation is predictive in our algorithm. The tracking method is a two-steps approach, a prediction-correction method. The prediction method uses the pose determination and a 3D affine motion model [7] to predict the future states of the tracked object. The correction step uses the current observations of the object to update its state. We are studying the possibility of finding the minimum group of interest points and a set of rules which form all the possible movements of a 3D tracked object. We are planing to obtain a correction method approximated enough, only checking the binary value of a small group of interest points. The 3D model-based tracking with pose determination method is currently being worked on and will be described in a later paper. We have been studying the motion of rigid objects which can be approximately modeled by a convex polyhedral shape. The direct extension to non polyhedral object can be contemplate if a 3D description of the object is available. This paper deals with the initialization process, which automatically obtains the 3D model. The input of this task is a video sequence of a convex polyhedron and the output is the 3D model and information about its pose and its position.

The rest of the paper is organized as follows: Section 2 presents the process to extract the position and the orientation. Section 3 describes the 3D model. The experimental results revealing the efficacy of the method are shown in Section 4. Finally, the paper concludes along with discussions and future work in Section 5.

*Thanks to Doctoral College on Computational Perception (Vienna University of Technology, Austria) for funding.

2. POSITION AND POSE ESTIMATION

The relevant steps to obtain the shape, the position and the orientation of 3D tracked object in a video sequence are summarized in this section.

2.1. 3D Shape

The proposed process to extract the 3D shape of a convex polyhedron runs four consecutive steps which are described below.

2.1.1. 2D silhouette (Canny edge detector)

To obtain the 2D shape or the 2D silhouette of the polyhedron we use the Canny edge detector [6, 8]. An edge detector is the most common approach for detecting meaningful discontinuities in intensity values.

2.1.2. Parallel edges (Hough Transform)

Ideally, the Canny Edge Detector discussed in the previous section should yield pixels laying only on edges. In practice, the resulting pixels seldom characterize an edge completely because of noise, breaks in the edge from non uniform illumination, and other effects that introduce spurious intensity discontinuities. Thus edge-detection algorithms typically are followed by linking procedures to assemble edge pixels into meaningful edges. Perhaps the most often used approach to find and link line segments in an image is the Hough transform [6].

The Hough transform of a line segment in 2D can be described with two real-valued parameters using the Hessian normal form for representing lines (see Equation 1).

$$x \cdot cos\theta + y \cdot sin\theta = \rho; \tag{1}$$

The ρ value is the distance between the line and the origin, while θ is the angle of the vector from the origin to this closest point (x, y).

Finding parallel lines in a Hough transform can be challenging [9]. Ideally, the parallel lines have the same θ, which means that they must lie in the same column in the Hough domain. In practice, this is not always the case because of the quantization in the image space, the quantization in parameter space of the Hough transform, as well as the fact that edges in typical images are not parallel due to non ideal parallel projection and the discretization. One strategy to overcome this problem is our three-stage strategy:

1. We introduce a discretization step of the θ -axis equal to 5 degrees, which is enough to suppress the effect of the a non ideal parallel projection.
2. The total sum of each column of the Hough transform, is always the same value and equal to the total numbers of points in the image domain. However there are peaks with high values which correspond to the edges. Therefore once each element of the Hough domain is raised to the power of a high value, the total sum of each column does not have anymore the same number, the columns with parallel edges have the highest numbers.
3. The parallel edges in the 2D shape are the cells containing the highest value in each column highlighted in the last step.

2.1.3. Edges between visible faces (Histogram)

Between two visible faces always there is a visible edge. In the case of three visible faces these edges are the lines segments joining two cut-off points between the edges in its silhouette. We use the gray-scale value histogram when the polyhedron has exactly two visible faces. An edge in the image space represents the directionality of the brightness changes in the image. Therefore, we run over the whole length of the silhouette and repeatedly dividing its silhouette in two parts. We calculate the histogram of these two parts. Finally we compare how similar are these two histograms. The line which splits the polyhedron in two parts with the most dissimilar histograms is the edge between the two visible faces.

2.1.4. Corners

Knowing all the visible edges, it is possible to extract the visible corners of the polyhedron. The visible corners are the cut-off points between the edges. In order to calculate the coordinates of the non visible corners we calculate the third coordinate (z) of the visible corners. First of all, it is necessary to establish the origin of the z-axis, the origin will be the closest point to the camera. In the case of only one visible face all the visible corners have z equal to 0. In the case of two or three faces we use the aspect graph to identify the current viewing angle. To calculate the z coordinate of the rest of corners we use the knowledge of the geometry of the polyhedron. For instance in the case of a cube (see Figure 1) the sizes of its edges must be always the same (see Equation 2) and its vectors must be orthogonal (see Equation 3).

$$|\vec{a}| = \left|\vec{b}\right|; \tag{2}$$

$$\vec{a} \cdot \vec{b} = 0; \tag{3}$$

With the edges $\vec{a} = (a_x\ a_y\ a_z)$ and $\vec{b} = (b_x\ b_y\ b_z)$.

Hence, we solve a system of two equations with two unknown values, a_z and b_z.

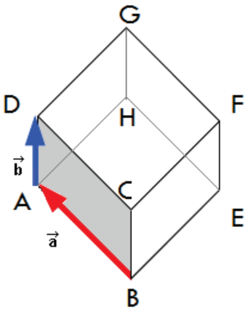

Fig. 1. Geometry of a cube.

To calculate the coordinates of the invisible corners we apply again the knowledge of the geometry. In the cube of the Figure 1 the invisible corner is H (see Equation 4).

$$H = E + \vec{a}; \tag{4}$$

2.2. The Position

The position of the polyhedron can be described by the center of gravity, this is the center of mass (C). Center of Mass for a convex polyhedron (see Equation 5).

$$C = \frac{\Sigma_{i=1}^{k} x_i}{k}; \tag{5}$$

x_i is a corner of the polyhedron and k is the total number of corners.

2.3. The Orientation

In geometry the orientation or the pose of an object, is part of the description of how it is placed in the space. Euler angles parameterize orientation using only three numbers (α, β, σ). These are the rotation angles around the Z, Y, and X axis respectively [10].

3. CONSTRUCTION OF THE 3D MODEL

We build a multi-view appearance model, as described in [1]. This 3D model has the same geometry and the same surface topology as the tracked object. We extract the geometry of the tracked object from the video sequence. We look for one frame of the video sequence in which only one visible face appears. We extract the corners of the visible face and thereby we obtain the dimensions of this face. We build a 2D template per each face with the set of its representative points. Finally, we place the templates in their corresponding faces of the 3D model.

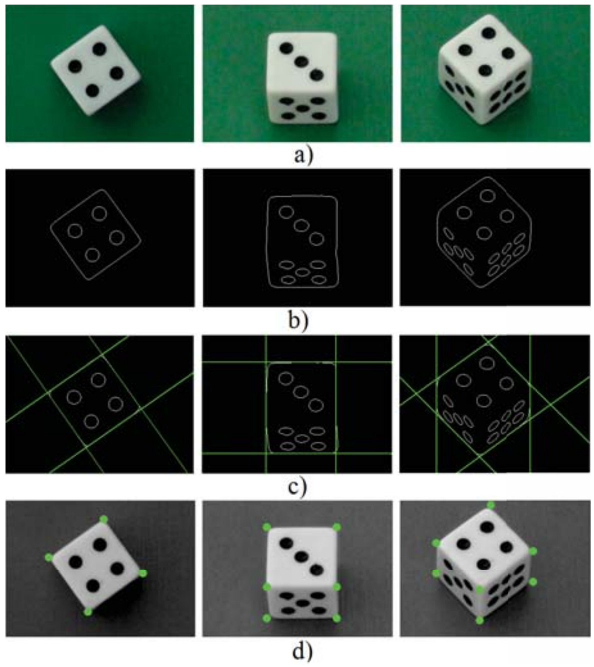

Fig. 2. (Best viewed in color) a) different images of a die; b) result of Canny edge detector; c) edges in the 2D silhouette; and d) visible corners (see text for details).

4. EXPERIMENTAL RESULTS

Figure 2. a) shows three images of a die (a convex polyhedron) from different views, with one, two and three visible faces respectively; b) the output of the Canny edge detector superimposed on the images in a); c) the parallel edges of the 2D silhouette are highlighted (green lines); and d) the visible corners are highlighted (green dots). Figure 3. a) shows the visible corners; b) the visible corners as well as the non visible corner; and c) all the corners and the center of gravity, marked with a red "x". Figure 4. is an example to find the representative points of a die. Figure 5. is the 3D model of a die. Figure 6. is another example of a cube, a poker die where our approach successfully extracts the corners.

When two parallel edges of the die project into lines having an angle of more than 5 degrees, the parallel projection assumption is violated and the method is unable to detect the parallelism. Further on, edge detection relies on local contrast, which if not strong enough leads to missing detections.

5. CONCLUSION

This paper has proposed the starting point of a novel approach. Our method integrates pose estimation and tracking. It works in monocular videos, where the position of the camera is fixed and the 3D tracked object is known. The starting point has been an initialization approach, which obtains the shape, the position, the orientation and builds a 3D multi-

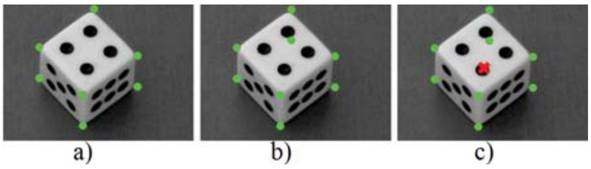

Fig. 3. (Best viewed in color) a) the visible corners; b) all the corners; and c) all the corners and the center of gravity (with a red "x"), (see text for details).

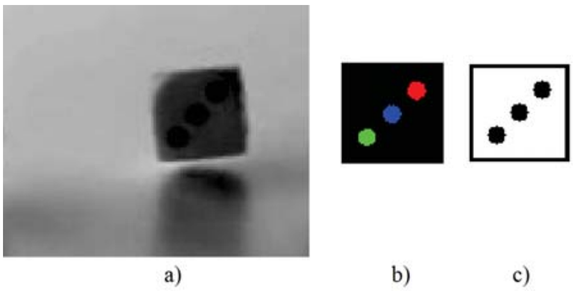

Fig. 4. a) image of a die; b) connected components; and c) geometry and representative points of the face 3.

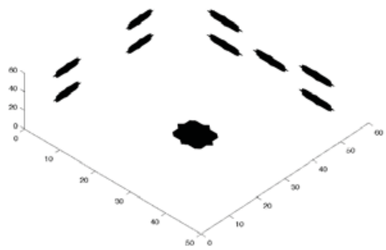

Fig. 5. 3D multi-view appearance model of a die.

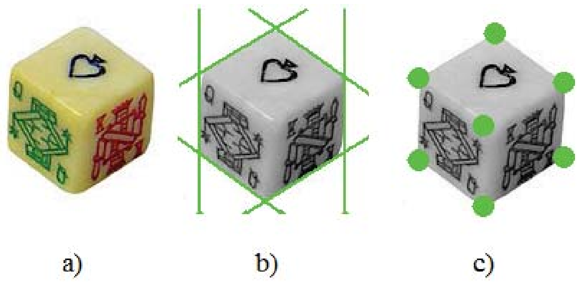

Fig. 6. a) one view of a poker die; b) edges in the 2D silhouette are highlighted; and c) the visible corners are highlighted.

view appearance model of a convex polyhedron. The second part will be the tracking with pose determination algorithm. This one is currently being worked on and will be described in a later paper. The tracking algorithm method looks for the minimum number of points needed to predict future poses and positions and also to correct possible errors or variations. Moreover, our final goal is to use this technique for tracking real world objects (e.g. cars).

6. REFERENCES

[1] O. Javed A. Yilmaz and M. Shah, "Object tracking: A survey," *ACM Comput. Surv., vol. 38, no. 4, pp. 1-45*, vol. 13, no. 1, pp. 234–778, 2006.

[2] K. Cannons, "A review of visual tracking," Tech. Rep., Technical Report CSE-2008-07, 2008.

[3] V. Lepetit and P. Fua, "Monocular model-based 3d tracking of rigid objects: A survey," *Foundations and Trends in Computer Graphics and Vision*, 2005.

[4] H. Zhou, Y. Yuan, and C. Shi, "Object tracking using sift features and mean shift," *Computer Vision and Image Understanding, vol. 113, pp. 345352*, 2009.

[5] S. Ravela, B. Draper, J. Lim, and R. Weiss, "Adaptive tracking and model registration across distinct aspects," *International Conference on Intelligent Robots and Systems, pp. 174-180*, 1995.

[6] R. C. Gonzalez, R. E. Woods, and S. L. Eddins, *Digital Image Processing*, Prentice-Hall, 2004.

[7] L. S. Shapiro, *Affine Analysis of Image Sequences*, Ph.D. thesis, Oxford University, 1993.

[8] M. Lv, H. Su, and Y. Li, "An adaptive canny detector with new differential operator," *Wireless Communications Networking and Mobile Computing (WiCOM), pp. 1- 4*, 2010.

[9] T. Tuytelaars, M. Proesmans, and L. Van Gool, "The cascaded hough transform," *Proc. IEEE Int'l Conf. on Image Processing (ICIP-97), pp.736-739*, 1997.

[10] F. Dunn and I. Parberry, *3D math primer for graphics and game development*, Wordware Publishing, 2002.