Approach to 3D Object Recognition Using a Collaborative Multi-View Camera System

A. Martinière and R. Kleihorst

Abstract—**Distributed smart camera networks allow multiple views of the same scene, which provide us more information for 3D object recognition than a single camera. Such a system can deal with occlusions by considering different scene views. The aim is to describe an approach to 3D object recognition using 3D features, by combining detected 2D features from different cameras. The object is modeled by its 3D features and its center of mass. The Intersecting Line Technique extended to 3D is used for the recognition process. Object recognition using a collaborative multi-view camera system can be used for many applications, such as gaming, surveillance or locating lost objects.**

Index Terms—**Multiple views, Smart cameras, Object recognition, 3D corner, Intersecting Line Technique**

I. INTRODUCTION

THE field of 3D object recognition has been investigated extensively in the recent years. While human beings can easily recognize arbitrary 3D objects in arbitrary situations, computer vision algorithms can only solve the object recognition problem in constrained conditions.

This paper introduces an approach to recognize objects in the 3D space using a collaborative multi-view camera system. The system which has been used for the experiments is the Wireless Camera (WiCa) platform, developed by NXP Research (see Figure 1). Each camera is equipped with a Xetal 3D processor, dedicated for video processing, and a communication module using ZigBee protocol. The Xetal is combined with a 30 frames per second color VGA-format image sensor. The processor is fully programmable and therefore able to run a variety of computer vision algorithms. Xetal 3D is able to achieve high computational performances (up to 50 GOPS) with very modest power consumption.

The aim of the method described here is to detect 3D objects, using 3D features and its center of mass. By combining several 2D feature descriptors obtained from different views, we can define a new type of 3D feature. The application of the Intersecting Line Technique [1] extended to 3D would allow us to recognize the object. We assume our camera network calibrated in space. Each smart camera uses corner detection for feature detector and corner orientation for feature descriptor. In order to find out the 3D features, we need to compute a correspondence rate between our corner descriptors. We use then the 3D corners detected in order to find the center of mass of the object in 3D space. We show that fusing color and grayscale for the Intersecting Line Technique significantly improves on the results of 2D.

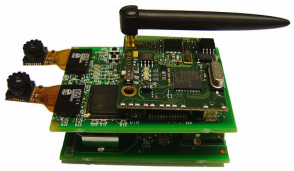


Figure 1: The WiCa (Wireless Camera) platform version 1.0.

The paper is organized as follows. Related work is described in section 2. A 3D corner definition based on 2D local features detection is explained in section 3. The Intersecting Line Technique extended to 3D is introduced in section 4. Finally, section 5 concludes this paper.

II. RELATED WORK

Most model-based 3D object recognition systems use information from only a single view [2][3][4]. However, a single view may not contain sufficient information to recognize the object, because the detected features depend on the camera viewpoint and the viewing geometry. Therefore, a single view based approach may not be suitable for 3D object recognition. One solution to this problem is to use information from different views. There has been research on combining data from several cameras [5][6][7] in order to recognize the object of interest, but none of them is based on the definition of a 3D feature.

A 3D object recognition based on the combination of 2D features detected from completely different angle of views implies that reliable 2D features have to be used. Moreels and Perona [8] compared the most popular feature detectors and descriptors using a benchmark designed to assess their performance in recognition of 3D objects. They found that the Hessian-affine detector performed well for viewpoint change and scale change, and the Harris-affine detector performed

Manuscript received September 20th, 2007.

A. Martinière works at NXP Research, High Tech Campus 32, office 140, 5656 AE Eindhoven, The Netherlands (phone: +31-40-27-45551; fax: +31-40-27-44639; e-mail: anthony.martiniere@nxp.com).

R. Kleihorst works at NXP Research, High Tech Campus 32, office 142, 5656 AE Eindhoven, The Netherlands (phone: +31-40-27-25900; fax: +31-40-27-44639; e-mail: richard.kleihorst@nxp.com).

well for lighting change and scale change [9]. The Scale-Invariant Feature Transform (or SIFT) descriptor was the most reliable globally [10]. Moreels and Perona proved then a significantly low stability of 3D features, namely only a small fraction of all features (less than 3%) can be matched for viewpoint changes beyond 30° using the SIFT descriptor. That means we have to find another kind of correspondence calculation to define a new 3D feature. The idea here is to compare the corner orientations computed from different views. The position of our WiCa systems in space is known a priory. Indeed, a spatial calibration of our distributed camera system needs to be executed beforehand.

III. 3D CORNER DEFINITION

A. 2D Local features

1) Corner Detector

We aim in defining a 3D corner based on the real-time detection of 2D corners from each WiCa. The basic idea is to use a common corner detector which is the Harris-Stephens corner operator [11]. The Harris-Stephens algorithm is not only sensitive to corners, but also to local image regions which have a high degree of variation in all directions. Therefore, all interest points are detected in the image, containing all corner orientations. It is also easier to implement this algorithm on the WiCa system than a SIFT feature for a real-time execution (30 frames per second). The problem of using a blob detection approach is computation time and the need of a higher number of line memories than a normal corner detection approach.

2) Corner Descriptor

As soon as a 2D corner is detected, we need to assign a descriptor for this corner. Our idea is to use the corner orientation, defined as the direction from the corner to the object. A relevant corner is detected on the object boundary. We compute the orientation by looking at the angle of the edges around the interest point, and by performing a background subtraction algorithm on the current scene. A comparison between edge orientations only gives us the direction of our corner but not its orientation, depending on the position of our object.

Figure 2 shows a corner (C) and the corresponding edges (E1, E2). For each corner point, we need to define on which side of the edges our object of interest is positioned. This is done by background subtraction, which defines the relative position of the object with respect to the position of the edges. The dotted vector represents the other orientation detected if no background subtraction is applied.

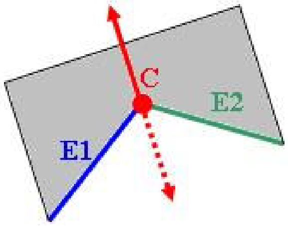
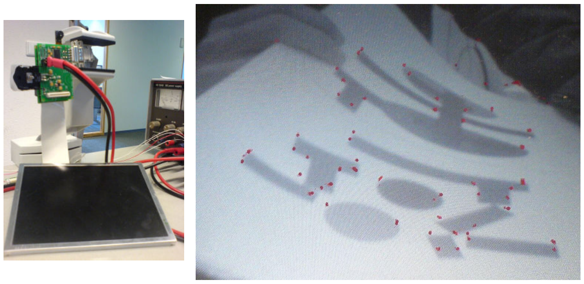


Figure 2: Corner orientation deducted after a background subtraction.

Alternatively, we could also use any of the object's grey level values instead of background subtraction, but this would be more sensitive to permanent luminosity change.

3) Experimental Results of 2D feature detection

The 2D corner detector and descriptor have been implemented on the WiCa platform. Figure 3(b) indicates the detected Harris corners by red dots. Those corners are stable when the object is translated or rotated. Their orientations change when the object is rotated. These detected interest points are displayed with a 5 line synchronization delay on the LCD screen compared to the actual corners. We use the neighborhood of the corners to deduct their orientation. In our experiments, we use 4 different Sobel kernels (0°, 45°, 90° and 135°) with 8 edge orientations, which allow the deduction of corner orientations.



(a) (b)

Figure 3: Oriented Harris corners detection with the WiCa system.
(a) WiCa system with a LCD screen for debugging. – (b) Detected corners displayed on the LCD screen.

The WiCa version 1.0 has a DPRAM memory of size 128 Kbytes. Two corners are coded in one byte. We use 4 bits to save one of our eight 2D corner types. First, a frame is saved in memory and then, the type and positions (line and column) of the corners are retrieved, to be used consequently for the 3D corner definition.

B. Approach to 3D Corner

Figure 4 illustrates the idea of combining 2D corner orientations in order to find a 3D feature. Assuming a calibrated system in space, we can interpret our 2D vectors and deduct the presence of a 3D corner.

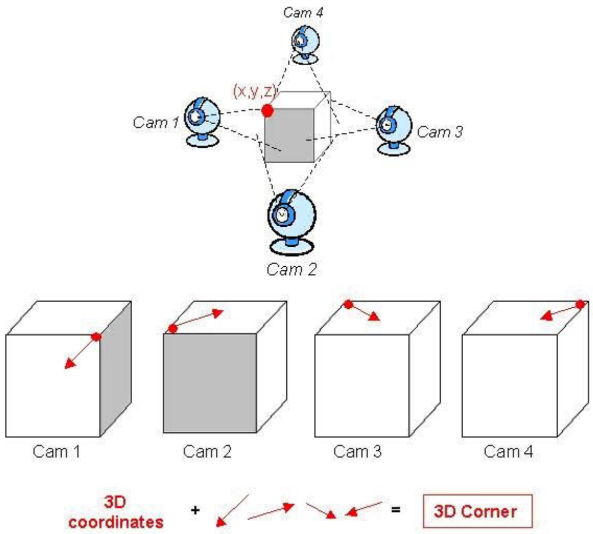


Figure 4: General example of a 3D corner detection with a 4 camera setup.

In practice, for each corner detected in one camera, we consider the pixels on the other cameras corresponding to the same position in space. If a corner exists at this position, we use its orientation to compute a correspondence rate. By applying the same process in each camera, the system can deal with occlusions. The algorithm is given on Figure 5. Prior to this algorithm, a 3D background subtraction method [12] may remove any detected corners not observed by all the cameras, for example if the corner located behind a camera is detected.

```
For each camera i,
  table = 0;//Initialisation of 'table'
  For each detected corner,
    table[i] ← Orientation(Pi);
    For each camera j, with $j \neq i$,
      //Pj corresponds to Pi in space known by calibration
      If Pj = detected corner,
      Then table[j] ← Orientation(Pj);
    Result ← Correspondence(table);
    If Result > Threshold,
    Then a 3D corner is detected
```

Figure 5: 2D corners orientations comparison algorithm.

In Figure 5, ‘Pi’ and ‘Pj’ are pixels located on cameras i and j, respectively. The variable ‘table’ is a table containing the corner orientations from each camera. The length of ‘tab’ is equal to the number of cameras used. The function ‘Correspondence()’ computes a correspondence rate between our 2D corner orientations. The goal here is not to minimize the distance between our feature vectors, but to compare their difference relative to the positions of our cameras in space. The correspondence rate gets higher if the relation between the different orientations is verified. This rate depends on the number of distributed cameras and their layout in space. The positions of the cameras in space are assumed to be known by calibration.

IV. 3D INTERSECTING LINE TECHNIQUE

A. *Extension from 2D to 3D*

The proposed 3D recognition process is based on the Intersecting Line Technique [1]. This method has the advantage of being simple, fast, scale-invariant and robust to occlusions. It uses a database of our object(s) which contains, for each feature point, the type of the feature and its 3D line gradient to the object center of mass. In Figure 6 an example of how to create such a database for a simple cubic object is shown. We have 8 different types of 3D corners and the corresponding 3D line gradients.

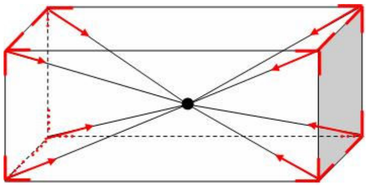


Figure 6: A database contains for each object its 3D corners information (number of 3D features, types, line’s gradients).

The centre of mass is calculated by taking the average x, y and z values of the object feature point coordinates. So, for an object consisting of n 3D feature points, the centre of mass is expressed as

$$CoM_x = (x_1 + \ldots + x_n)/n$$

$$CoM_y = (y_1 + \ldots + y_n)/n,$$

$$CoM_z = (z_1 + \ldots + z_n)/n$$

where CoM_x, CoM_y and CoM_z are its coordinates.

Once the shape is defined in the database and the feature points have been extracted from the image, the two must be combined to recognize the object. Each feature point detected in the image is considered separately. When a feature from the image is processed, then all occurrences of the sane feature type in the database are retrieved. From these retrieved database entries the associated line is drawn on the image, starting at where the feature point has been detected. More than one line can emerge from a specific image feature point if this type of feature is used multiple times in the object shape stored in the database. The newly drawn line emanates from the image feature point in the direction of the expected location of the centre of mass.

B. *Tests and Results in 2D*

The Intersecting Line Technique has been implemented on the WiCa platform and tested in 2D with a simple ‘T’ shape 2D object. As soon as the line gradients are computed, we seek the intersections points. Indeed, each line is drawn on a different bit plane. That allows us to count the number of lines intersecting on a specific pixel. The computation tasks are spread amongst the processors of the WiCa. The Xetal processor detects the corners and saves them in DPRAM. To monitor the process, a microcontroller Atmel 8051 is in charge of drawing the lines on the LCD screen. This operation is not done in real time but on user request. In Figure 7, we see the result of a simple ‘T’ shape displayed in front of our camera sensor where several lines are crossing each other. If the number of intersecting lines is greater than or equal to a threshold, which has a maximum equal to the number of object features, then we conclude that the object ‘T’ is present in the scene.

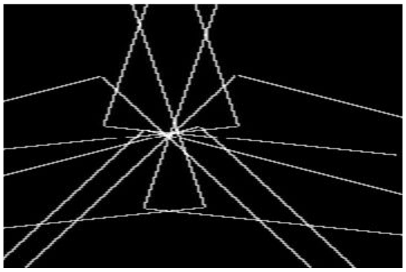


Figure 7: Intersecting Line Technique results in 2D on a simple ‘T’ shape.

We propose next the use of color to obtain more information than grayscale in certain conditions. This leads to a more accurate detection in certain cases. If we consider our specific algorithm, the goal is to decrease the number of lines in order to decrease the number of false detections. Therefore, if the feature color is known, we can segment our image and keep only the interest region. This way, we detect a smaller amount of corners; therefore fewer line gradients will be drawn on the screen.

Tests have been carried out on 20 different images using either color or grayscale, trying to recognize a colored T-Shape (red, green or blue). The test bench included real images and artificial images with different shapes, and colors and in different positions. The images have been displayed in front of our camera sensor. We obtained the ROC (Receiver Operating Characteristic) curves seen in Figure 8. The “True Positive”, “False Positive”, “True Negative” and “False Negative” rates have been computed for different thresholds applied to the number of line intersects (from 3 to 8). The ROC curves have been created with 20 different images. Even the low number of images proves the fact that color is definitely useful in certain condition. Our experiments show that in low luminosity conditions, grayscale detection performs better.

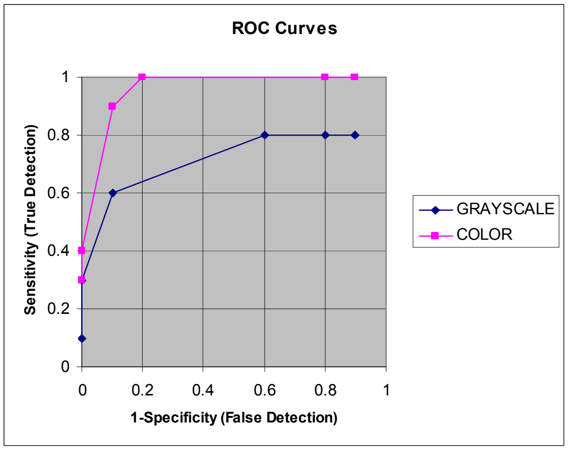


Figure 8: Recognition of a ‘T’ shape using either grayscale or color.

Therefore, the idea is to fuse grayscale and color depending on the luminosity condition. On the other hand, using color requires significantly more resources from our system, because it has to take into account 3 dimensional (RGB) pixel values.

V. CONCLUSION

We presented an approach to 3D object recognition using a multiple-view system, based on wireless camera (WiCa) platforms.

Our contribution was the introduction of a 3D feature, combining 2D corner orientations from different view points. We extend the Intersecting Line Technique to 3D using those 3D features. We show that color information gives more accurate measures for a 2D recognition process, by decreasing the number of features detected, hence the number of lines drawn on the screen.

Our experimental results show that at 10% of the false detection rate, we can achieve a 90% True Detection rate when color is used, and a 60% True Detection Rate when using gray-scale. That means color has to be taken into account in certain case, depending on the ambient luminosity.

So far, results concerning our 3D features detection are promising but more experiments have to be carried on, with a different number of cameras positioned in space. Additional work will be directed on the high-level reasoning to compute our correspondence rate concerning our 2D descriptors combination.

Future work may be extending the Harris-Stephens algorithm to a multi-scale approach, introduced by Mykolajczyk and Schmid [13]. They have also extended this method to affine transformation invariance. In practice, this multi-scale corner detector is often complemented by a scale selection step, using the scale-normalized Laplacian operator [14].

REFERENCES

[1] D. Rankin, “Embedded Object Recognition Using Smart Cameras and the Relative Position of Feature Points”, Master Thesis Report, University of Glasgow, pp 54 – 59, 2007.

[2] I. Weiss and M. Ray, “Model-based recognition of 3D objects from single images”, IEEE. Trans. on Pattern Analysis and Machine Intell, 23(2): pp 116 – 128, 2001.

[3] I. K. Park, K. M. Lee and S. U. Lee, “Recognition and Reconstruction of 3D Objects Using Model Based Perceptual Grouping”, Proc. 15th Int. Conf on Pattern Recognition, 1: pp 720 – 724, 2000.

[4] J. Ponce, S. Lazebnik, F. Rothganger, and C. Schmid, “Toward True 3D Object Recognition”, Proc. CVPR, Vol. II, pp 272 – 277, 2003.

[5] M. F. S. Farias, and J. M. de Carvalho, “Multi-view Technique For 3D Polyhedral Object Recognition Using Surface Representation”, Revista Conlrole & Automacao, 10(2): pp 107 – 117, 1999.

[6] J. Mao, P. J. FIynn and A. K. Jain, “Integration of Multiple Feature Groups and Multiple Views into a 3D Object Recognition System”, Computer Vision and Image Understanding, 62(3): 309-325, 1995.

[7] M.K. Osman M.Y. Mashor M.R. Arshad, “3D object recognition system using multiple views and cascaded multilayered perceptron network”, Cybernetics and Intelligent Systems, IEEE Conference, Vol. 2, pp 1011 – 1015, 2004.

[8] P. Moreels and P. Perona, “Evaluation of features detectors and descriptors based on 3d objects”, In Proceedings IEEE International Conference on Computer Vision, pp 800 – 807, 2005.

[9] K. Mikolajczyk and C. Schmid, “An affine invariant interest point detector”, ECCV, In Proceedings European Conference on Computer Vision, 2002.

[10] D. G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints”, International Journal of Computer Vision 60 (2): pp 91 – 110, 2004.

[11] C. Harris and M.J. Stephens, A combined corner and edge detector, Alvey Vision Conference, pp 147 – 152, 1988.

[12] H. Lee, C. Wu, and H. Aghajan, "Nonstationary Background Removal via Multiple Camera Collaboration", in Proc. of 1st International Conference on Distributed Smart Cameras, Vienna, Austria, Sept 2007.

[13] K. Mikolajczyk and C. Schmid, “Scale and affine invariant interest point detectors”, International Journal of Computer Vision 60 (1): pp 63 – 86, 2004.

[14] T. Linderberg, “Feature detection with automatic scale selection”, IJCV 30(2), p 79 – 116, 1998.