Detection of Aruco Markers Using the Quadrilateral Sum Conjuncture

José Ferrão[1], Paulo Dias[1,2], and António J. R. Neves[1,2(✉)]

[1] DETI, University of Aveiro, 3810-193 Aveiro, Portugal
{joseferrao,paulo.dias,an}@ua.pt
[2] IEETA, University of Aveiro, 3810-193 Aveiro, Portugal

Abstract. Fiducial Markers are heavily used for pose estimation in many applications from robotics to augmented reality. In this paper we present an algorithm for the detection of aruco marker at larger distance. The algorithm uses the quadrilateral sum conjecture and analyzes the sum of the cosine of the internal angles to detect squares at larger distances. Experiments conducted showed that the developed solution was able to improve the detection distance when compared to other methods that use similar marker while keeping similar pose estimation precision.

Keywords: Robotics · Digital images · Computer vision · Marker detection
Aruco

1 Introduction

Absolute pose estimation is a common problem in robotics. One possible solution to tackle this problem is the use of odometry information (coming from encoders and Inertial Units). However, this solution suffers significantly from drift error.

An alternative commonly used not only in robotics but also in areas like augmented reality is the usage of markers to estimate the position and orientation of the camera. A visual marker is something that can be easily distinguished from the rest of the ambient and have characteristics that allow it to be easily detected and identified.

The motivation for this work was the need for a marker-based solution to easily identify charging stations for robots and correcting odometry drifting using the pose estimated by the marker detector.

One of the main problems found when experimenting already existing solutions (ROS Aruco [1] and Alvar [2]) was that they are not able to identify the markers from far away or under hard perspective distortion making it difficult to discover the charging station. In this paper we propose an algorithm for the detection of aruco markers. Our approach uses the quadrilateral sum conjecture to improve the detection distance of these markers.

The remain of this paper is structured as described next. Section 2 presents base concepts related with marker encoding and detection, Sect. 3 presents the proposed algorithm and discusses pose estimation. Section 4 presents experimental results, and, finally, Sect. 5 contains result analysis and conclusion.

A. Campilho et al. (Eds.): ICIAR 2018, LNCS 10882, pp. 363–369, 2018.
https://doi.org/10.1007/978-3-319-93000-8_41

2 Aruco Markers

Aruco markers are geometrically square, they have a black border and an inner grid that is used to store a numeric identifier in binary code.

A dictionary is used to identify the marker [3]. The dictionary defines a set of rules used to calculate the marker identifier, perform validation and apply error correction.

We use the original aruco dictionary [1], that uses bits from the marker 2nd and 4th columns to store the marker identifier in natural binary code, the remaining bits are used for parity checking. Figure 1 presents the first four markers in this dictionary.

Fig. 1. Aruco markers with id's 0, 1, 2 and 3.

A signature matrix is used to validate the marker. Each row of this matrix encodes a possibility of 2 bits. An aruco marker is valid only if each of its rows are equal to one of the rows of the signature matrix, ensuring that the marker only have one valid rotation. Table 1 represents the signature matrix used in this dictionary.

Table 1. Signature matrix, used to validate aruco markers.

Value	Data				
0	1	0	0	0	0
1	1	0	1	1	1
2	0	1	0	0	1
3	0	1	1	1	0

By analyzing the signature matrix, it is possibly to verify that it is not enough to guarantee that there is only one possible rotation for each marker, in the Fig. 2 we can see the marker 1023 that is horizontally symmetric.

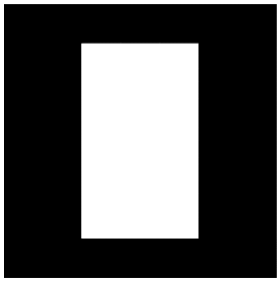

Fig. 2. Aruco marker 1023.

3 Detection Algorithm

The detection algorithm was implemented using the OpenCV library, since it provides a large set of image processing algorithms. Figure 3 shows the steps applied to detect and identify markers.

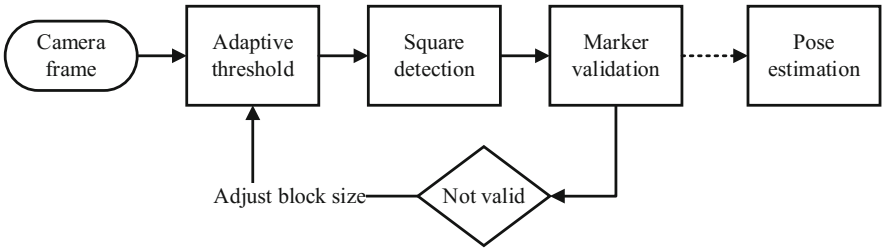

Fig. 3. Algorithm diagram

The algorithm starts by applying adaptive threshold [4] to the image, this algorithm consists in calculating for each pixel a threshold value using the histogram of its neighborhood. It is of particular interest for situations with multiple lighting conditions. Figure 4 shows the results after applying adaptive thresholding.

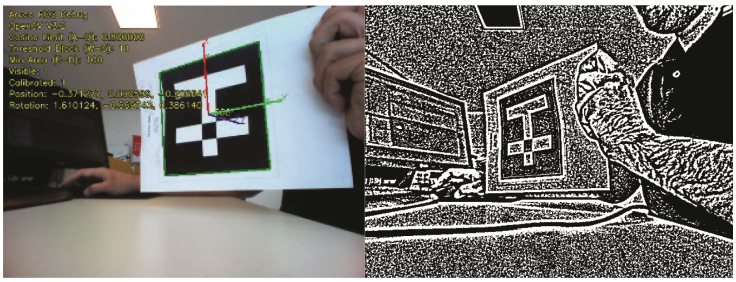

Fig. 4. Adaptive threshold result.

To determine the threshold block (neighborhood size), one block size is tested on each frame, the block size chosen is the average size from all block sizes were the maximum number of markers were found, the block size is retested when there are no markers visible.

After threshold is applied to the image, we perform square detection contours are detected using a border-following algorithm [5], followed by the Douglas-Peucker contour simplification algorithm [6].

Based on the detected contours, the Quadrilateral Sum Conjecture is used as a criterion to detect squares. Even under significant perspective distortion a square is always a convex quadrilateral. Our second criteria will be to make sure that the sum of the cosine of all inner angle is below a defined threshold.

To filter noise a third criterion was added: all contours composing a geometry with an area bellow a defined threshold will be discarded.

These three criteria allow to properly filter squares even under heavy distortion from the contour list. Figure 5 represents the obtained result for a maximum sum of cosine of 0.25 and a minimum area of 100px.

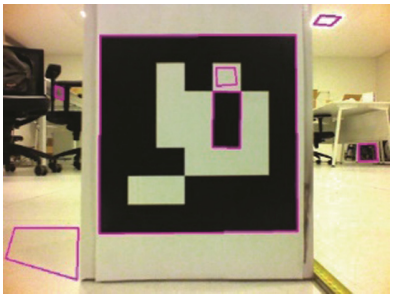

Fig. 5. Result of square detection algorithm

Perspective distortion is corrected in the detected squares, then they are resampled into a 7×7 matrix using linear interpolation, threshold is applied using the Otsu's Binarization algorithm [7], at this point we obtain a matrix with the marker data in it. Figure 6 represents the matrix obtained after the binarization process.

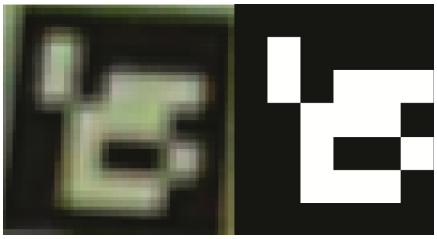

Fig. 6. Marker reading result

At this stage, the marker data is validated as aruco using the signature matrix. Markers might be detected in any orientation. The algorithm tests the data with different rotations (90º, 180º, 270º), if the marker is not recognized for any rotation it is then discarded.

For pose estimation the method solvePnp from OpenCV was used, in iterative mode using Levenberg-Marquardt optimization [8].

To obtain the camera position, markers need to be registered into the program, a marker is represented by its identifier and a real-world pose (position and rotation).

Corners obtained from all visible known markers are used to estimate the camera pose.

4 Algorithm Evaluation

We created a testing environment to compare the developed solution with the ones already existing in the literature. Two test markers were printed: one Aruco maker and one ARTag maker. Both markers had exactly 20 cm in size and the camera was placed

on top of a box with the marker aligned with the camera. The marker was fixed with transparent tape.

A measuring tape with a millimeter scale was used to measure the distance between the camera and the markers. An image was taken for each distance tested and the markers were moved 30 cm each time until none of the algorithms was able to detect the marker. Figure 7 represents some samples of the testing images used during the experiments.

Fig. 7. Some images used to test the developed algorithm.

To measure the tolerance of the detector to perspective distortion a second testing environment was created. A marker was placed on a box and the camera was positioned 2.0 m away. The marker was rotated in steps of 10º from 0º to 80º. Table 2 presents the results obtained for marker rotation showing that the proposed method performed better than the other two algorithms used for comparison, obtaining lower error.

Table 2. Results obtained for maker rotation.

Rotation	Proposed solution		ROS Aruco		Alvar	
	Dist. (m)	Error (%)	Dist. (m)	Error (%)	Dist. (m)	Error (%)
0	1.981	0.980	2.013	0.631	1.986	0.727
10	1.983	0.872	2.032	1.583	2.017	0.858
20	1.985	0.741	2.031	1.537	2.011	0.564
30	1.989	0.528	2.029	1.415	1.974	1.308
40	2.001	0.040	2.029	1.425	1.971	1.493
50	2.002	0.121	2.03	1.468	2.028	1.357
60	2.004	0.213	2.033	1.639	1.974	1.322
70	1.993	0.341	2.029	1.421		

Camera calibration was performed using a chessboard pattern and the values obtained were stored to be used for the tests. Figure 8 presents the results obtained. It is possible to observe an improvement in the maximum detection distance when using our algorithm.

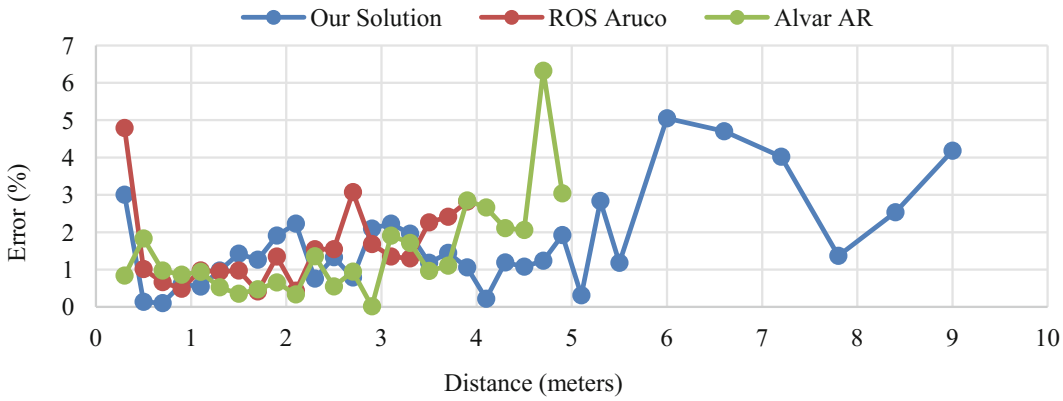

Fig. 8. Samples of the testing data

5 Conclusion

Two well known and used existing algorithms were tested (ROS Aruco and Alvar) and compared with the one proposed in this paper. All algorithms use a similar approach for marker detection: adaptive threshold, detect squares, refine corners, apply distortion correction, and decode the marker data. The proposed method uses a different approach for square detection allowed to detect markers at larger distances and more immune to rotations.

Experimental results show that algorithm detects the marker up to 9 m, when compared with 4 m and 5 m for the state of art algorithms referred before, representing a 44% increase in detection distance, with similar precision values.

The method also reveals more tolerance to perspective distortion, obtaining better precision results when detecting rotated markers.

To improve the algorithm further a corner refinement method with subpixel accuracy could be added improving the marker corner position estimation.

An implementation of the algorithm described in this document can be found at www.github.com/tentone/aruco.

Acknowledgments. This work was partially funded by FEDER (Programa Operacional Factores de Competitividade - COMPETE) and by National Funds through the FCT - Foundation for Science and Technology in the context of the project UID/CEC/00127/2013.

References

1. Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F.J., Marín-Jiménez, M.J.: Automatic generation and detection of highly reliable fiducial markers under occlusion. Pattern Recogn. **47**(6), 2280–2292 (2014)
2. Siltanen, S.: Theory and Applications of Marker-Based Augmented Reality. VTT Science 3
3. Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F.J., Medina-Carnicer, R.: Generation of fiducial marker dictionaries using mixed integer linear programming. Pattern Recogn. **51**(C), 481–491 (2016)

4. Sezgin, M., Sankur, B.: Survey over image thresholding techniques and quantitative performance evaluation. J. Electron. Imaging **13**(1), 146–165 (2004)
5. Suzuki, S., Abe, K.: Topological structural analysis of digitized binary images by border following. Comput. Vis. Graph. Image Process. **30**(1), 32–46 (1985)
6. Wu, S.-T., da Silva, A.C.G., Márquez, M.R.G.: The Douglas-peucker algorithm: sufficiency conditions for non-self-intersections. J. Braz. Comp. Soc. **9**(3), 67–84 (2004)
7. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Trans. Syst. Man Cybern. **9**(1), 62–66 (1979)
8. Levenberg, K.: A method for the solution of certain non-linear problems in least squares. Q. Appl. Math. **2**(2), 164–168 (1994)