Technische Universität München
Fakultät für Informatik
Prof. Gudrun Klinker, Ph.D.
Andreas Langbein, Adnane Jadid, David Plecher

Summer Term 2018
Exercise Sheet 4
May 9th, 2018

Introduction to Augmented Reality

Exercise 8 (P,H) Line fitting

From exercise 7, you should have six estimated points for each of the marker edges.

(a) Now you need to fit a line through these points. Use the `OpenCV` function `cv::fitLine` for this task. Hint:

```
cv::Point2f points[6];
...
float lineParams[16];
cv::Mat lineParamsMat( cv::Size(4, 4), CV_32F, lineParams);
                                            // lineParams is shared
...
cv::Mat point_mat( cv::Size(1, 6), CV_32FC2, points);
cv::fitLine ( point_mat, lineParamsMat.row(i), CV_DIST_L2, 0, 0.01, 0.01 );
```

A line is an array of four floats, containing the direction.

(b) At this point, you have four lines. Intersect every two consecutive lines to get the new marker corners. Consult a book on planar geometry on how to achieve this.

(c) In order to check your results, draw a small circle at the original corner points from `cvApproxPoly` and a second one of different color at the new corner locations.

Exercise 9 (P,H) Marker identification

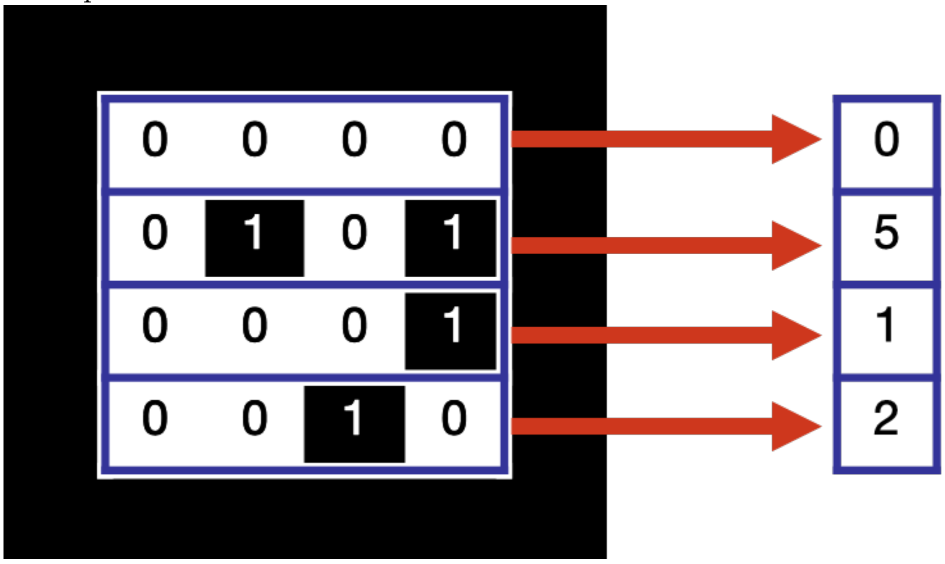
To make sure that the quadrangle you have found is actually the marker you are looking for, the perspective transformation now has to be undone in order to rectify the marker.

(a) The function `cv::getPerspectiveTransform` (or `cv::warpPerspectiveQMatrix`) takes two sets of four 2D points and calculates a $3 * 3$ matrix representing the perspective transformation. For the first set, use the points $(-0.5, -0.5)(-0.5, 5.5)(5.5, 5.5)(5.5, -0.5)$, for the second set, use your points from exercise 8.

(b) Create a new image of size $6 * 6$. For each of the pixels of this image, multiply its coordinates with the matrix from the previous step. Use `cv::warpPerspective` to get the right image.

(c) Try to identify the marker. Discard all markers which don't have a black border. For all others, generate a 16-bit identifier from the innermost $4 * 4$ subimage. Print all marker identifiers on the console.
Note: remember that there are four different marker identifiers per marker, depending on the orientation. Try to think of a way to get a rotation-independent id.

Example:


0 0 0 0
0 1 0 1
0 0 0 1
0 0 1 0
0
5
1
2