Robust detection of n-fold edges using convolution with a complex kernel

Henrik Skov Midtiby

12. marts 2020

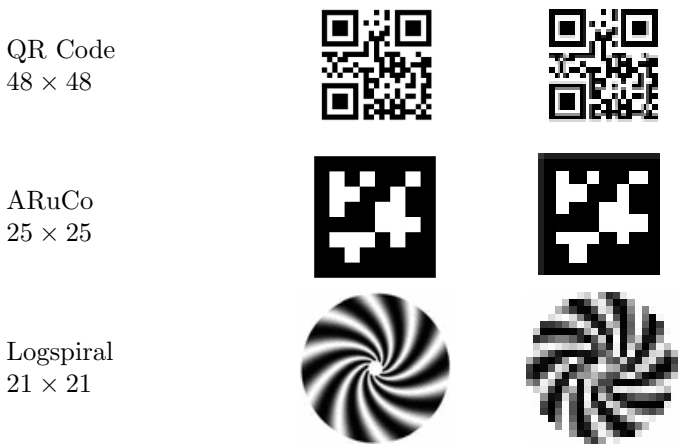
Resumé

In some cases it is required to track a certain point / marker on an object in an image / a sequence of images. This paper suggest to use a marker (an n-fold edge) constructed by repeating a pattern of white and black regions n times. The center of the marker can then be located by a convolution with a kernel containing complex numbers.

1 Introduction

An often occuring task in computer vision is to locate a certain object or position in an image. It can either be a generic object, or an object (a marker) that the user have placed actively in the field of view. Robust detection of one or more markers can give precise pose estimates which are required for applications like Augmented Reality.

Many types of markers have been suggested, and they can be divided into two types of markers. The first type of markers are intended for estimating the pose of the object on which the marker is placed. This requires at least four points to be identifiable in the marker, which often consists of a square



Figur 1: *Examples of different marker types and images with a minimal resolution in which the marker can be detected (and verified).*

or rectangular array of black and white pixels. QR Codes and ARuCo markers are both of this type. The second type of markers identifies specific points in images. These markers resemble eg. chess board corners and log spirals.

QR Codes are a 2D bar code, that can contain a significant amount of information. The largest QR codes contain almost 3000 bytes of information. QR Codes was developed by DENSO WAVE. The minimum resolution for reading a QR code containing four bytes of information using the `https://zxing.org/w/decode.jspx` QR code decoder around 50 x 50 pixels (personal experiments). ARuCo markers consists of a black border with a $n \times n$ pattern of black and white pixels inside the border [1]. The smallest ARuCo marker consists of 4×4 pixel within an two pixel wide border made of black and white pixels. To detect this marker a resolution of at least 18 pixels is likely needed. Chess board corners are regularly used for camera calibration. The corners can be detected using different techniques, eg. the Harris corner detector [2]. A marker formed as a logarithmic spiral was suggested in [3]. A visual comparison of the mentioned marker types are given in figure 1.

This paper suggests a new marker design with the following properties 1) robust detection of the marker, 2) the algorithm requires no segmentation of the input image prior to marker detection and 3) the algorithm can locate low resolution markers. This is achieved by analysing the neighbourhood around each pixel in the image, like in the FAST corner dector http://www.edwardrosten.com/work/fast.html. The analysis is done by convolution of the image with a kernel containing complex.values. This approach limits the algorithm to only locate markers with a certain fingerprint.

2 Materials and methods

2.1 Template matching

It is possible to use template matching to locate objects with a fixed orientation in an image. https://docs.opencv.org/trunk/d4/dc6/tutorial_py_template_matching.html

2.2 Fourier transform

Given N observations $(x_0, x_1, \dots)$, the kth term in the discrete Fourier transform is given by the equation:

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-i 2\pi k n / N}$$

Notice that the Discrete Fourier transform is a weighted sum over a set of observations, that is a convolution. In the standard situation the set of observations is sampled along a linear dimension. Instead of sampling along a linear dimension, the sampling will be done over a 2D area, the kernel window. This is similar to

a 2 D convolution, which is defined as follows:

$$f[x,y]* g[x,y] = \sum_{k=-w}^{w} \sum_{l=-w}^{w} f[k,l] \cdot g(x-k, y-l)$$

The task is now to design a pattern to add to the object that should be tracked and which is possible to detect with the convolution based approach described above.

2.3 Square wave

A square wave $x(t)$ with amplitude 1 and frequency f can be represented with the Fourier series

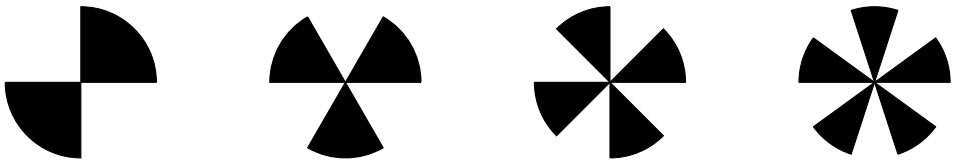
$$x(t) = \frac{4}{\pi}\left(\sin(2\pi ft) + \frac{1}{3}\sin(6\pi ft) + \frac{1}{5}\sin(10\pi ft) + \dots\right)$$

Given the function $x(t)$, the Fourier transform can be used to determine the elements of the Fourier series.

2.4 The plain marker

Instead of locating a square wave in an image, the pattern is bent around a certain point and then replaced with high and low intensities. This is illustrated in figure 2. The generated pattern has a well defined spatial center and as will be seen later, the pattern can be detected using convolution. By altering the number of repetitions of the black and white pattern around the central point, a set of different markers can be generated. The number of repetitions of the pattern is denoted the order (n) of the pattern. In figure 3 three patterns are visualised, the patterns have the orders $n = 2$, $n = 3$ and $n = 4$.

Figur 2: *Four repetitions of a square wave pattern is bent into a circular pattern around a central point. The direction from the central point out to the low levels of the square wave is coloured black.*



Figur 3: *Markers with different orders ($n = 2 \ldots 5$).*

2.5 Detection of a marker

Detection of a square wave with n repetitions using Fourier analysis, relies on a convolution of the N measurement of the signal with the kernel Y_n:

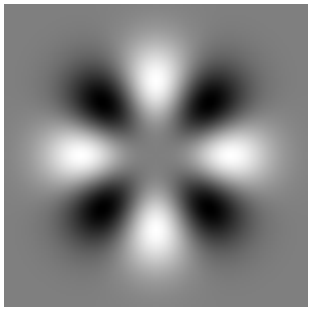
$$Y_n = e^{-i2\pi kn/N} \qquad k \in [0, \ldots, N-1]$$

A somewhat similar kernel is used to detect the n-fold edge markers. The kernel is specified using polar coordinates as follows:

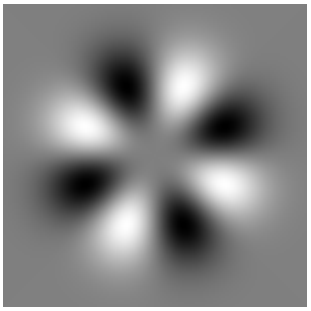
$$Z_n = e^{-in\theta} \cdot r^n \cdot e^{-8r^2}$$

where θ is the direction and r is the distance to the actual position in the kernel. The center of the polar coordinates are placed in the middle of the kernel and is scaled such that a circle with radius 1 is the largest circle that can be placed inside the kernel. Four different views of a kernel with order $n = 4$ is shown in figure 4.

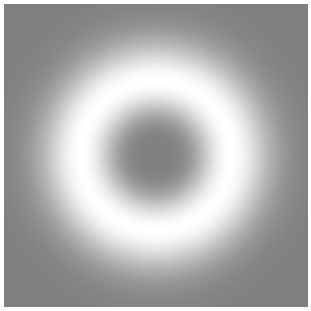
To detect a marker with a known order, the input image is converted to



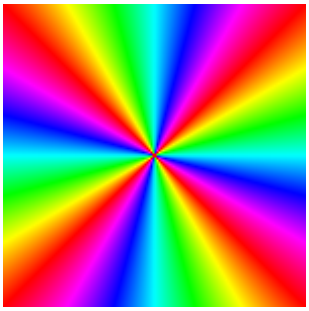
(a) *Real part of kernel.*



(b) *Imaginary part of kernel.*

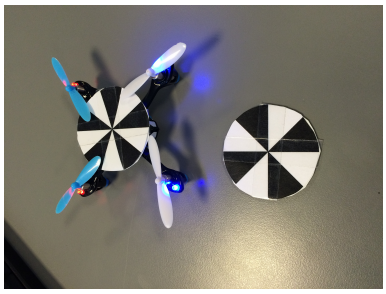


(c) *Magnitude of kernel.*

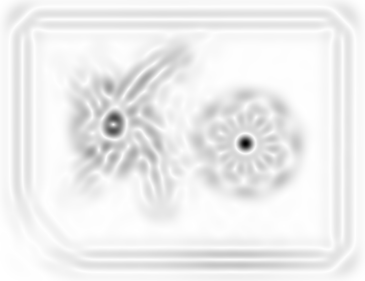


(d) *Argument of kernel.*

Figur 4: *Different views of the complex kernel used for detecting n-fold markers ($n = 4$).*



(a) *Input image containing markers of order 4 and 5.*



(b) *Marker detection response in inverted colors. Black denotes a high response to the marker.*

Figur 5: *Marker detection example response.*

a grayscale image and then convolved with the Z_n kernel. As the Z_n kernel contains complex weights, the resulting image will contain complex values. The magnitude of the complex values in the resulting image, tells us how well the input picture matches the used kernel, this is visualised in figure 5; the argument of the complex value tells the orientation of the pattern, to best match the input image.

2.6 Estimating the quality of the detected marker

Interpreting the magnitude response to the Z_n kernel poses an issue when markers with different (but nearby) orders are present in the input image. As can be seen in figure 5, where a marker of order $n = 4$ is being detected, there is a moderate response around the marker mounted on the Hubsan UAV with order $n = 5$. This issue can of course be reduced by avoiding markers with similar orders in the same image, but a better solution is to check that the algorithm actually found a marker with the requested order; that is to assign some kind of quality score of the detection result.

The used approach for estimating the quality of a detected marker, is to utilise information about the orientation of the marker (from the argument of

the kernel response) to align the orientation of the located marker with the expected pattern of white and black regions. An example of a template for the position of white and black markers are shown in figure 6. For all pixels in the white / black regions of the template, the average image intensity and standard deviation is calculated, this gives the values: μ_w, μ_b, σ_w and σ_b.

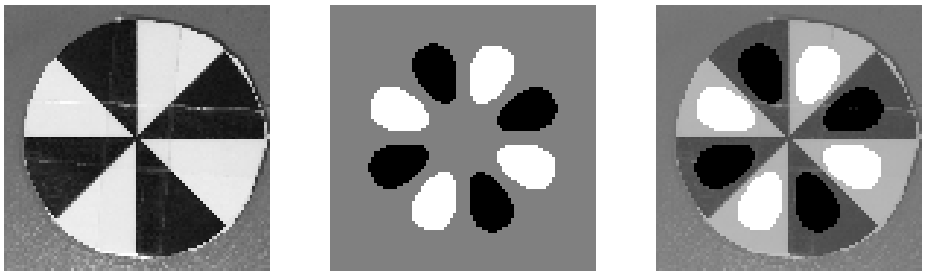
A marker that matches the pattern that has been searched for (regarding order of the kernel) and is positioned correctly above the center of the pattern, will have well separated grayscale values for pixels in the white and black regions respectively. Whether this is the case is quantified by calculating the normalised difference (t) between the grayscale values in the white and black regions:

$$t = \frac{\mu_w - \mu_b}{0.5 \cdot \sigma_w + 0.5 \cdot \sigma_b} \tag{1}$$

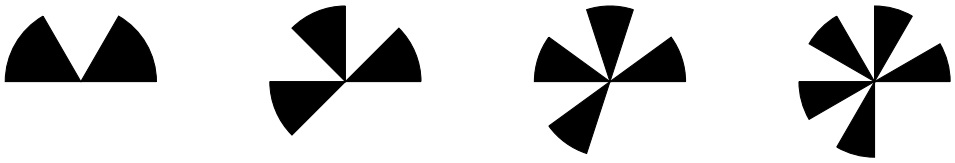
If t has a value above 7, it indicates that there is a very large difference between the grayscale values in the white and black regions of the template. In an attempt of making the estimated quality easier to interpret, the following mapping between the t value and the resulting quality score is utilised.

$$\text{quality} = 1 - \frac{1}{1 + e^{0.75 \cdot (t-7)}} \tag{2}$$

The quality score gives a number between zero and one. A low score indicates that the detected marker does not match what was searched for and is likely to be a random match. When the quality score gets above 0.5 the tracker is quite confident that the detected marker is actually what was searched for.



Figur 6: *Visualisation of how the quality of the detected marker in figure 5a is assessed. A region centered above the detected marker is extracted, the quality template divides the extracted part of the image into three regions: expected white pixels, expected black pixels and don't care pixels. The template is rotated so it is aligned with the detected marker.*



Figur 7: *Markers where one of the black legs have been removed to indicate an orientation of the marker. The markers have the orders* $(n = 3 \dots 6)$.

2.7 The oriented marker

Even though the plain marker contains some information about the orientation of the marker (as it is possible to discriminate between markers with different orientations), is is not possible for the pattern to point in a certain direction, for eg. specifying the orientation of a tracked object. By changing one of the black legs of the pattern to white, the pattern is given a unique orientation. This is illustrated in figure 7.

3 Results

Results on detecting markers with different sizes. Mention that markers can be detected reliably with 16×16 kernels, which is quite small.

Give results on the running time of the marker detector and how it scales with image and kernel size.

A python module implementing the algorithm is available on https://github.com/henrikmidtiby/markerlocator.

4 Conclusion

Litteratur

[1] S. Garrido-Jurado m.fl. "Automatic generation and detection of highly reliable fiducial markers under occlusion". I: *Pattern Recognition* 47.6 (jun. 2014), ss. 2280–2292. ISSN: 0031-3203. DOI: 10.1016/J.PATCOG.2014.01.005. URL: https://www-sciencedirect-com.proxy1-bib.sdu.dk/science/article/pii/S0031320314000235?via%7B%5C%%7D3Dihub.

[2] Chris Harris, Chris Harris og Mike Stephens. "A combined corner and edge detector". I: *IN PROC. OF FOURTH ALVEY VISION CONFERENCE* (1988), ss. 147–151. URL: http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.231.1604.

[3] Stefan M Karlsson og Josef Bigun. "Synthesis and detection of log-spiral codes". I: *Swedish Symposium on Image Analysis (SSBA2011)*. 2011.