Computer Vision Tools for Locating Nitrogen-Vacancy Centers

Eric Wadkins, Michael Walsh, Dirk Englund
Quantum Photonics Laboratory, Massachusetts Institute of Technology, Cambridge MA, 02139, USA
ewadkins@mit.edu

Abstract: Presented in this paper is my work on the detection and auto-focus algorithms used by the Quantum Photonics Laboratory at RLE to locate nitrogen-vacancy centers. I discuss the design, implementation, and evaluation of the system, as well as its future use in improving instrument localization for self-driving microscopy.

1. Introduction

The Quantum Photonics Laboratory uses a confocal camera to detect nitrogen-vacancy (NV) centers in a diamond surface. Locating these NV centers requires the examination of auxiliary QR codes that contain location information, as well as other information regarding the NV centers. The detection method, however, must be tolerant to noise and variations in the surface of the diamond. Another challenge faced by the lab is finding the focal plane in which the QR codes are visible. The region in which the codes are visible (and focused enough to extract the necessary information) is generally very narrow, making it difficult to focus the confocal camera manually.

As evidenced by these challenges in the detection and focusing of the QR codes, the lab stands to benefit significantly from a system which automates these tasks. Therefore, my work for the Quantum Photonics Laboratory includes an automatic detection and focusing system. This paper details the design and implementation of the system, a brief performance evaluation, and a discussion of its future use in self-driving microscopy.

2. Detection

In order to extract information from the QR codes, the system requires a detection method that is tolerant to a noisy environment. To achieve this, a technique known as image correlation is used, which is closely related to image convolution. This technique calculates how closely a region of the input image resembles the correlation kernel. Therefore, using a gray-scale model of a QR code as the correlation kernel (see Figure 1) allows for the accurate detection of regions of the image that closely resemble a QR code. Image correlations are calculated by fast Fourier transform (FFT), one method of efficiently calculating convolutions/correlations which provides an especially large speedup for large kernels. [1]

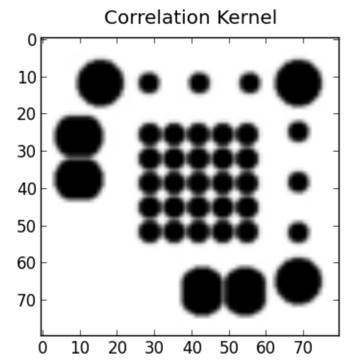

Fig. 1: The correlation kernel used in detection of QR codes near NV-centers.

One problem with using image correlation, though, is that the background gradient of the images results in some of the darker regions having an output value similar to that of an actual QR code (see (b) in Figure 3). To counter this problem, the result of the image correlation is passed through a notch filter, which filters out the solid high-value regions, leaving only the high-value regions representative of QR codes (see (c) in Figure 3). [2]

At this point in the pipeline, all there remains to do is to group together the high-value points that correspond to each individual QR code. This is done by thresholding the image and using blob detection, which is a simple way to group regions together based on value. However, since different images may result in drastically different values, a static threshold cannot be used. Therefore, I have created a process to dynamically determine an appropriate threshold. The process works by initially setting the threshold at a low, preset value proportional to the average value of the image. Then, the thresholding is performed and the number of different regions is calculated. Should the number of regions be too high (greater than some *k*) or any of

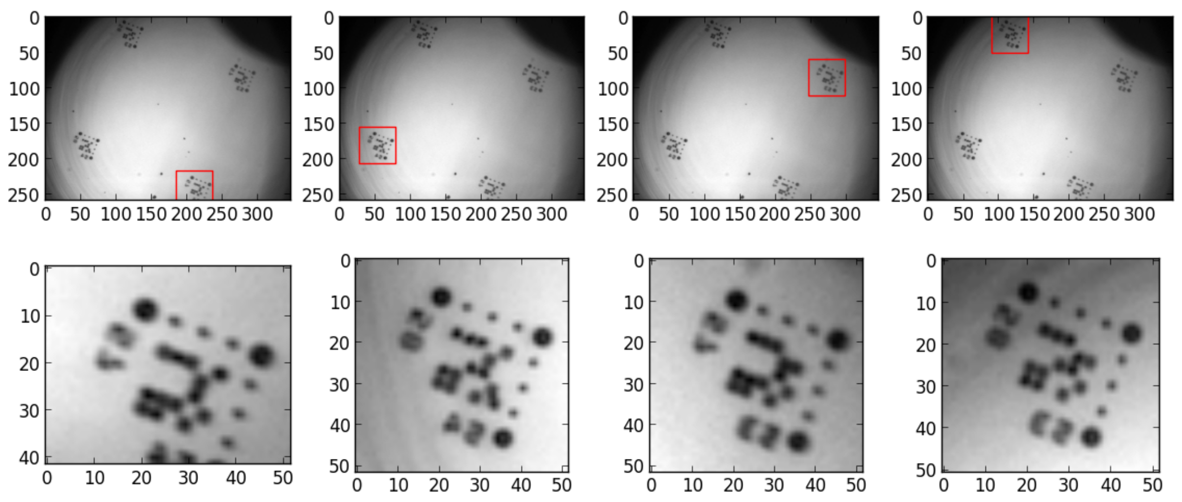

Fig. 2: The automatic detection of four QR codes on a diamond surface; one of which (left) is partially obscured.

the regions be too large, the threshold is increased and the process repeats. I have found this to be an effective way to dynamically determine an accurate threshold, while at the same time excluding false positives. Unfortunately, this comes with the restriction that there may not be more than k different regions detected, which limits the number of QR codes that can be detected at once. This, however, is an acceptable restriction for the use case of the Quantum Photonics Laboratory.[1]

Figure 2 (above) shows example results of the detection algorithm. Figure 3 (below) shows the processed image in each step of the detection pipeline.

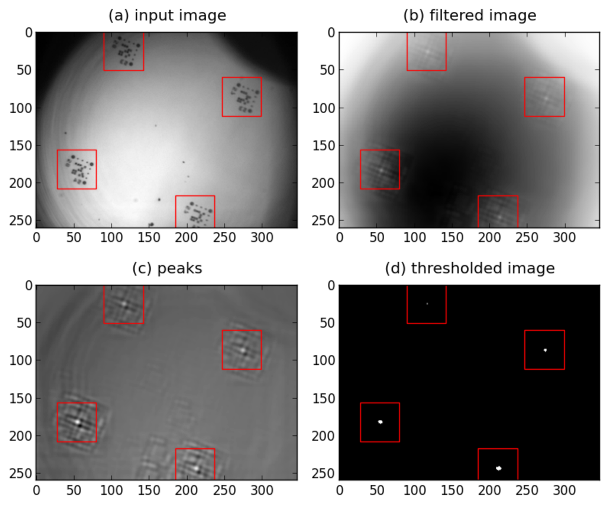

Fig. 3: The four steps in the detection pipeline. Brighter regions depict regions of higher value.

3. Calibration

The detection algorithm detailed in section 2 works given specific values for two parameters: the scale and orientation of the QR codes. Therefore, there must be a way to determine the scale and orientation prior to running the detection algorithm (as well as the auto-focusing algorithm discussed in section 4).

One way to determine if a pair of parameters is able to identify the QR codes is to pass the image through a portion of the detection algorithm with those parameters. The higher the output, the more likely it is that a detected QR code was found with those parameters. A major challenge, though, is that if either one of the parameters is incorrect (eg. scale is correct, but orientation is not), the detection algorithm will fail to detect any QR codes.

Therefore, the calibration process performs a beam search over both domains simultaneously. First, a search with a large step size is performed in order to minimize the number of tests required. As the search proceeds, the size of the search space is significantly reduced, along with the step size. Figure 4 (below) shows an example of the calibration search space and this reduction procedure.

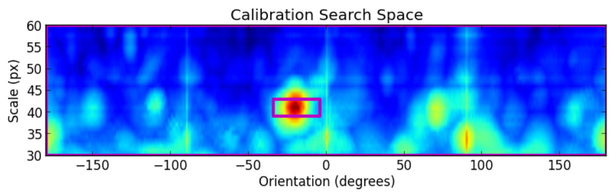

Fig. 4: A heatmap of the detection algorithm output over the calibration search space. The magenta rectangle shows the size of the search space after just one iteration.

[1] We use a limit of 8. This is reasonable, as most of the time there will only be 4 codes visible at once.

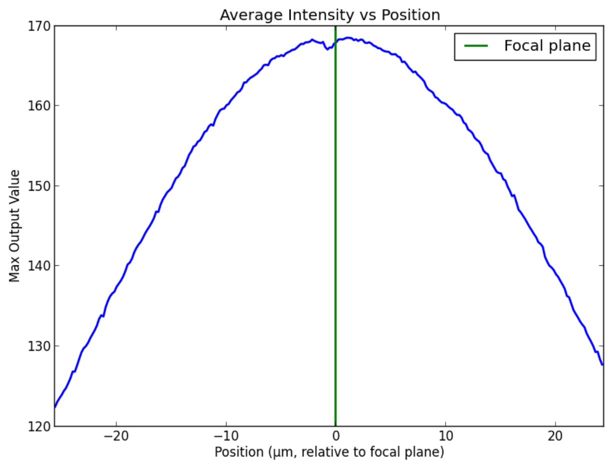

Fig. 5: The average intensity as position varies relative to the focal plane.

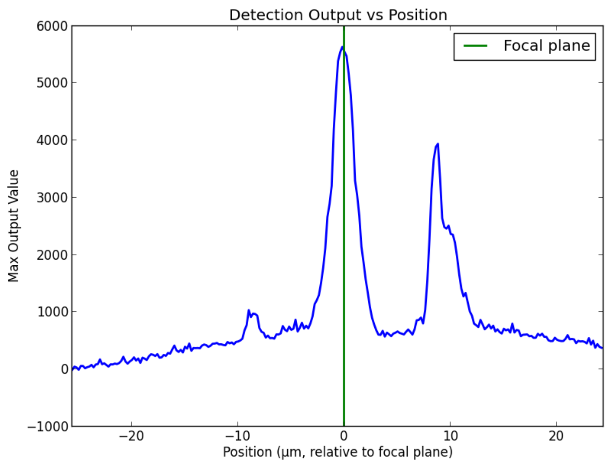

Fig. 6: The max detection output as position varies relative to the focal plane. Secondary maxima often exist as a result of regions of in-focus noise.

This search space reduction repeats until the step size reaches a predefined lower limit depending on the level of accuracy desired. The parameter pair with the maximum output is then stored for later use by the detection and auto-focusing algorithms.

Using this method to search for the correct pair of parameters decreases the runtime of the calibration process and increases the accuracy, since the allowed error can be set to be arbitrarily small. In this way, accuracy can be traded off for improved runtime as needed.

4. Auto-Focus

The process of automatically focusing the confocal camera is similar to the calibration process described in section 3, except that it requires physical adjustments to the position of the camera, rather than simply performing operations on the same input image. As such, it is critical to minimize the number of steps that must be taken to focus the camera.

Fortunately, there are two metrics that this process may take advantage of when focusing the camera: average intensity and detection output. As the camera approaches the focal plane, the average intensity of the image tends to increase (see Figure 5). This realization provides us a quick way to get *close* to the focal plane. This also has the added benefit of ensuring that the correct local maximum of the detection output is explored (see Figure 6). However, analyzing intensities alone is not an accurate enough method of focusing the camera.[2] For the level of accuracy desired, we must also use the output of the detection algorithm (see Figure 6).

The process begins by receiving input from the camera at some arbitrary initial position. Then, the first step is always to instruct the camera to move away from the surface (to avoid a potential collision, should the camera's initial position be too near the surface). The intensity of each input image is calculated, providing us with two data points. The intensities can then be compared to determine the direction of the maximum relative to the current position.

Once the direction of the maximum is determined, the camera is instructed to move in that direction with a relatively large step size (we chose 2 μm), and collect intensity data as it moves. This continues until several decreasing intensity values are observed, indicating that the camera is now moving away from the focal plane. Then, using the changes in position and the intensity values observed at those positions, a quadratic curve can be fit to the collected data. The maximum of this curve, which provides a strong estimate of the actual maximum, can be calculated, after which the camera is instructed to move to that position.

The auto-focusing process now switches to using the detection output as a measure of how focused the QR codes are. Similar to the first step of this process, the camera is then instructed to move slightly away from the surface to collect a second data point. Using these two points, the direction of the maximum can be determined. The camera is then instructed to move towards the maximum with a small step size (depending on the level of accuracy

[2]This is partially because as the QR codes come into focus, the intensity decreases slightly due to their dark appearance. Intensity is also simply a weak indicator of the presence of QR codes.

desired; we chose 200 nm), collecting the maximum output values as it does so. The maxima of the detection output tend to be unimodal and positioned far apart from each other, as seen in Figure 6. For this reason, once the output values are observed to be decreasing, we can be certain that we have observed the maximum, after which we instruct the controller to position the camera at the maximum and signal that the auto-focus process is complete.

Careful consideration should be given when choosing the step sizes. The step size in the first part of the process, for which we chose 2 µm, determines how quickly the data is collected and how far the camera must move. As the step size increases, the number of steps taken decreases, but the distance the camera must overshoot the focal plane in order to fit the quadratic increases. The step size in the second part of the process, for which we chose 200 nm, should be significantly smaller as it determines the minimum accuracy of the auto-focus algorithm. Again, though, the smaller it is, the larger the number of steps that must be taken to find the focal plane.

5. Evaluation

In this section, I evaluate the performance and accuracy of the three essential components of the system: the detection, calibration, and auto-focus algorithms.

5.1. *Detection*

To evaluate the accuracy of the detection algorithm described in section 2, I put it to the test using a wide range of input images. It is generally able to detect any QR code that has at least two-thirds of the expected shape visible. Whether or not partially obscured QR codes are useful is not the concern of this paper. In the rare case that one or more easily-recognizable QR codes appear with a less-recognizable code (eg. too close to edge, malformed shape, etc.), the dynamic thresholding technique (see section 2) may result in failure to detect the less recognizable code. However, the vast majority of QR codes are able to be correctly identified by the detection algorithm.

Arguably as important as correctly finding QR codes is the ability to ignore the surrounding noise. This is precisely the reason that the dynamic thresholding technique (see section 2) is utilized. Figure 7 shows the results of the detection algorithm run on an image with an extreme amount of noise (more noise than would ever be observed in practice) — much of which could easily be mistaken for a QR code due to the similar pattern. The QR code is also poorly positioned near the edge of the image. However, the detection algorithm is still able to correctly find the one actual QR code in the image and generates no false positives.

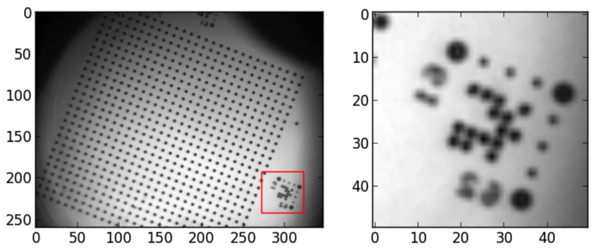

Fig. 7: Results of the detection algorithm run on an image with an extreme amount of noise and a poorly positioned QR code.

This illustrates that the detection algorithm is able to recognize and subsequently ignore what may have been false positives using other detection methods.

It is also important that the detection algorithm is able to run in real-time. Due to several optimizations,[3] a 460 x 344 image (the size commonly used by the Quantum Photonics Lab) takes on average just 70 milliseconds to run. Consequently, the detection algorithm can run in real-time with little or no noticeable delay.

5.2. *Calibration*

The calibration process, which the detection and auto-focus algorithms are dependent on, works reasonably well. Given a moderately clean image, such as the image in Figure 2, the calibration process succeeds without any problems. Given an image like the one in Figure 7, however, results in the failure of the calibration process to correctly determine the scale and orientation of the QR codes. With that said, it is expected that the calibration process would require a less-noisy image than the detection algorithm, as it must determine the scale and orientation of the QR codes, rather than already having that information available. Therefore, the calibration process, while not flawless, definitely performs within the accuracy required of it. Also, the runtime of the calibration process is not a concern, since it will only be run upon re-orientation of the surface. Even then, it generally takes under a minute to complete.

5.3. *Auto-Focus*

The auto-focus algorithm performs exceptionally well. The steps taken to find the focal plane depend entirely on the initial position of the camera. However, regardless of starting position, the algorithm is always able to adjust the camera into focus, and do so with little risk of a collision with the surface. The algorithm also has very little variation in the end result. While implementing the auto-focus algorithm, I was concerned of the possibility that a

[3]The most significant optimization was the use of FFT to perform the image correlation. FFT performs several orders of magnitude faster than other, more frequently used methods.

starting position too close to the focal plane would result in a poorly-fitted quadratic curve, which would in turn result in either an incorrect end result, or a correct result that took much longer to converge to than normal. Fortunately, this does not seem to be the case, as the quadratic fits quite well regardless of the starting position.

Using 2 µm and 200 nm as the large and small step sizes, respectively, we tested the system with an initial distance of 10 µm from the focal plane. Just 10 large steps are required to fit the quadratic, and 5 small steps to find the focal plane. An initial position that is already at the focal plane, however, requires 4 large steps and 12 small steps to find the focal plane. Due to irregularities in intensity near the focal plane and the algorithm quickly realizing it is moving away from the focal plane, there are fewer steps to collect intensity data. As a result, the quadratic less accurately estimates the position of the focal plane, resulting in a larger number of small steps to find it. While unexpected, this behavior is beneficial considering the distance the camera must move: many small steps are preferred to many large steps. Table 1 (below) shows some of the test results when varying initial distance to the focal plane.

Initial Distance	# Large Steps	# Small Steps
0 µm	4	12
2 µm	6	4
5 µm	7	4
10 µm	10	4

Table 1: The number of steps required to find the focal plane for various initial distances.

I also explored the use of edge detection in determining whether an object was focused. However, this appraoch was determined to be inferior to the chosen approach because some noise may have well-defined edges, and edge detection alone cannot distinguish between noise with well-defined edges and focused QR codes. Using the detection algorithm as a metric is therefore much better suited for this task.

6. Conclusion & Future Work

As demonstrated by the results of the system evaluation (see section 5), the methods described in this paper are successfully able to automate the tasks of detecting QR codes near NV centers and automatically focusing the confocal camera on these QR codes. The next steps for this system include performing integration tests to observe how the system behaves in conjunction with the laboratory equipment. While I have already implemented and tested (via a simulation) the module allowing the auto-focus algorithm to communicate with the camera controller, the system has yet to be tested beyond simulations. Also, it would be worthwhile to further explore the use of edge detection in the auto-focusing process, likely in conjunction with the detection algorithm currently in use. While I provided reasoning against using edge detection alone (see section 5.3), using both methods may provide a way to boost the results to an even higher level of accuracy.

The work presented has the potential for use in other applications within the Quantum Photonics Laboratory. As part of a year-long research project throughout the 2017-2018 academic year, I will be conducting research into how the detection algorithm detailed here can be used to improve instrument localization for self-driving microscopy. Coupled with Bayesian inference, it is possible to more accurately determine the location of instruments over a surface encoded with location information by taking into account the inherent level of uncertainty that results from the use of physical instruments. The proposed system will build upon, and in many ways be a natural extension of, the work presented in this paper.

References

1. Schreier H., Orteu J., Sutton M.A. (2009) Image Correlation for Shape, Motion and Deformation Measurements: Basic Concepts, Theory and Applications. Springer, Berlin, Heidelberg. doi:10.1007/978-0-387-78747-3. http://link.springer.com/book/10.1007%2F978-0-387-78747-3
2. Koh HW., Hildebrand L. (2010) Automated Gaussian Smoothing and Peak Detection Based on Repeated Averaging and Properties of a Spectrums Curvature. In: Hllermeier E., Kruse R., Hoffmann F. (eds) Information Processing and Management of Uncertainty in Knowledge-Based Systems. Theory and Methods. IPMU 2010. Communications in Computer and Information Science, vol 80. Springer, Berlin, Heidelberg. doi: 10.1007/978-3-642-14055-6_39. http://link.springer.com/chapter/10.1007/978-3-642-14055-6_39