Methods for Enhancing Corner Detection

Alexander Honda
Advisor: Todd Wittman

March 17, 2010

Abstract

In this paper we explore several experimental techniques for increasing the precision of corner detection. The presented methods are based on the idea of using local image information associated with predetected corners to achieve subpixel resolution. All the methods discussed used the Harris corner point algorithm for the initial detection, but they are modularly compatible with any corner detection algorithm.

1 Introduction

Image registration is a common requirement for most image processing applications. It is necessary for image fusion, change detection, recognition, and pattern matching. Methods exist for achieving this automatically, however they are far from robust. Many of these methods rely on feature point extraction. A specific subset of feature point extraction methods are those which detect corners. We are particularly interested in corner points because they are defined locally, usually in very small neighborhoods. The methods in this paper attempt to use a detected corner and local image information to increase the precision of the corner, with the goal of improving registration efforts.

Many feature based registration methods use more the minimal number of points and minimize a sum of squares error function to mitigate the effects of noise. This is the quantity over quality approach to registration. It makes the assumption that error in feature detection is zero mean. For smaller collections of points it becomes more unlikely that this assumption as true. In these cases a quality over quantity approach should be taken. By devising a method to locate corners with subpixel precision we intend to provide a means to improve registration. In the case where there is an abundance of points, applying registration with precise corner points should yield equivalent results if the zero mean assumption was correct, or better results if it was wrong, and it should not degrade the results.

2 Background

2.1 The Moravec Detector

The Moravec approach is to compare the error between shifted patches with the original image using sum of squared differences[5]. When a shifted patch is similar to the original image the region is homogeneous in the shifted direction. A flat patch will thus produce a very low value for all shifts. A single edge patch will yield a large change in the directions perpendicular to the edge, and almost no change for shifts parallel to the edge. A corner should produce a large change in any direction since it has two non-parallel edges. The equation can be formulated as shown below:

$$E_{u,v}(x,y) = \sum_{x,y} \left(I(x+u, y+v) - I(x,y)\right)^2$$

The Moravec Detector selects the location of the minimum of $E_{u,v}(x,y)$ for

$$(u,v) = \{(-1,-1), (-1,0), (-1,1), (0,-1), (0,1), (1,-1), (1,0), (1,1)\}$$

2.2 The Harris Detector

The Harris Detector[2] is a commonly used method for extracting corner point locations. It is based on the Moravec Operator, but uses a first order approximation of patch shifts to calculate the sum of squared differences, as a bilinear function of the shift, rather than directly computing it for select values:

$$\begin{aligned} E_{u,v}(x,y) &= \begin{bmatrix} u & v \end{bmatrix} \sum_{x,y} w(x,y) H(x,y) \begin{bmatrix} u \\ v \end{bmatrix} \\ &= \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} \end{aligned}$$

The Harris method uses a weighting function $w(x,y)$ for the window. Common functions for $w(x,y)$ are step functions and Gaussians. Gaussian functions are popular because they treat the image data more symmetrically, rather than preferring directions parallel to either of the axes.

Since the goal of the corner detector is to identify when $E_{u,v}$ varies in all directions, the next step is to associate a value to this amount. The eigenvalues of the matrix M are a good indication of this. When both eigenvalues are small there is little change for any (u,v). When one is large and the other is small it indicates that there is an edge, since one direction has high change, while the orthogonal direction has small change. When both eigenvalues are large it indicates a corner.

The Harris Detector avoids computing the eigenvalues directly, and opts for a method of computation which does not require the square root operator:

$$C(M) = det(M) + k * tr^2(M)$$

This is equivalent to:

$$C(M) = (1 + 2k)\sigma_1\sigma_2 + k(\sigma_1^2 + \sigma_2^2)$$

Due to the squared terms this function is large when both singular values are large. The k term is a tuning parameter, which was empirically determined to be 0.04 in the original paper.

2.3 Other Methods

There are many other methods for detecting corners. The Sh-Tomasi Detector [8] uses $C(M) = \min(\sigma_1, \sigma_2)$ for affinely transformed images. The Noble Detector [6] is more similar to the Harris method, but uses explicitly computed singular values. $C(M) = \frac{\sigma_1\sigma_2}{\sigma_1+\sigma_2+\epsilon}$. The method chosen is often simply a matter of preference by the researcher.

3 Line Tracing Method

In this section we present and evaluate a method of detecting corners using edges in images. The goal is to locate nearby lines that would intersect close to the predetected corner point and try to accurately model them so that we can calculate a more precise corner point. It relies on the images to be of man made objects, which tend to have strait edges not found in images of naturally formed objects, but has the benefit of aggregating data over several pixels which should offset noise and pixelization error.

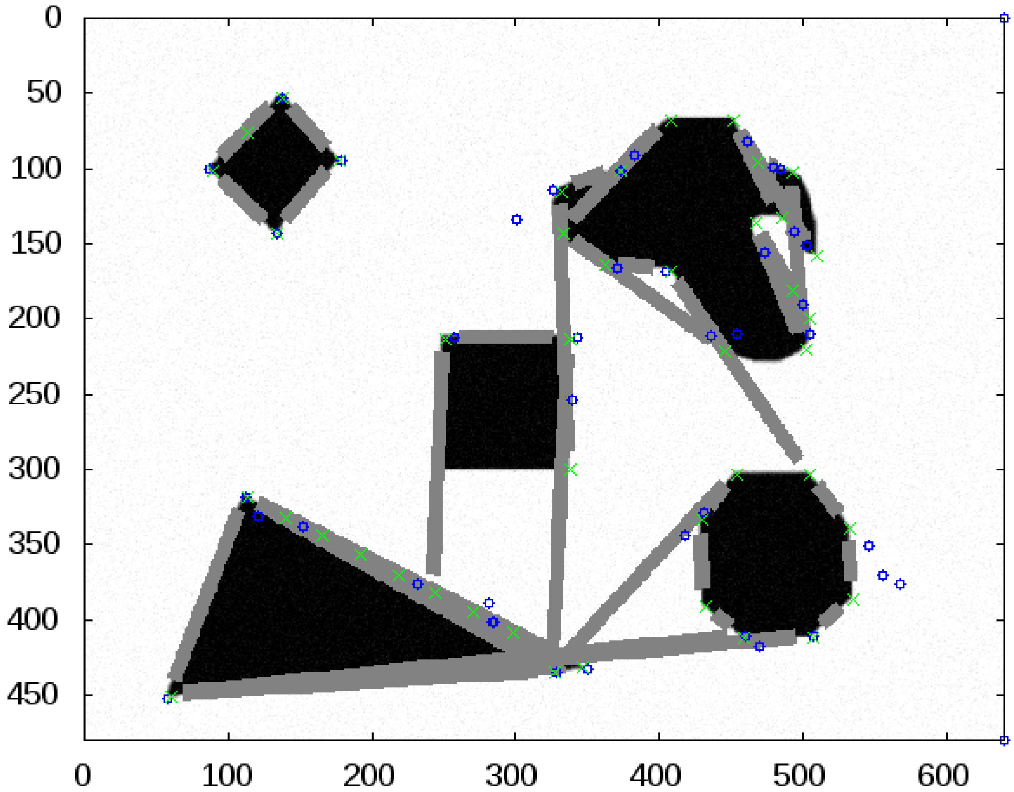
3.1 Implementation

A mask of a given width is constructed for each pair of corner points. The mask indicates where an edge between two corner points would run if it exists. It is truncated near the corner points with the intention of capturing only one edge in each mask. For each mask a score is calculated by averaging either the gradient, or the Harris corner score in the masked region. Since the average of the gradient or Harris corner score is large in this region it is likely for these regions to contain lines.

The top two highest scoring masks are taken to be the strongest edges. A weighted linear regression is performed on the set of all pixels in the mask, with the weights being the gradient (or Harris corner score). This yields parameters for the standard form of the line equation for each mask. Solving for the intersection then yields the enhanced corner point.

3.2 Test

A synthetic two class image with strait lines, curves, and corners with various angles was created for preliminary testing. A Gaussian blur was applied, and noise added to mimic real data. Then the algorithm was run on the resulting image.



In this image the gray boxes are the masks. The green x marks are the original corner points as detected by the Harris method. The blue circles are the updated locations of corner points.

3.3 Conclusion

There are inherent problems with this method that make it non-ideal for use on real images. The gradient and Harris corner measure can vary by orders of magnitude, throwing off the average value of masks that cross through high valued regions, as can be seen by the long masks which cross large regions of whitespace.

Forcing two edges from each detected corner point doesn't make sense when the detect points are not corners, which can be seen in the top right shape. There is not always sufficient room between corner points to draw enough data to correctly identify a edge, as seen in the octagon in the bottom right. This is especially the case for satellite and aerial images, where large buildings may only be a few pixels long.

This method also suffers when the corner detection provides false corners, or fails to provide real corners which can be seen in the bottom left triangle and the center square, respectively. This is a common occurrence with real images, especially low resolution images where corner details get cut out and pixelization creates corner-like artifacts.

Finally, from the tests run it is clear that the lines calculated are overly dependent on the masks used. Very few lines do not run perfectly down the center of the mask. These problems lead to several cases of making the corner point worse than it initially was. While this is an interesting method, it is not worth pursuing as a practical solution to the problem.

4 Local Image Enhancement

This section presents and evaluates two methods which try to solve the problem in a very small neighborhood around the detected corner. The goal of these methods is to enhance an image locally around a detected corner point, in order to increase the precision of the corner point localization. In particular, we will be focused on increasing the resolution by four, since satellites such as IKONOS and QuickBird produce four times higher resolution for their panchromatic sensors.

4.1 TV with Neighborhood Constraints

This method is similar the image enhancement method used by Guo, Wittman, and Osher[1]. It minimizes an energy functional which is the total variations[7] cost with an added penalty term for deviating from the mean of a local neighborhood. This local neighborhood is defined as the 4x4 pixel region which represents a single pixel in the low resolution image patch which is being expanded. The functional to be minimized is:

$$S(u, u_x, u_y) = \int \sqrt{u_x^2 + u_y^2} + \lambda \left(\frac{1}{\|N\|} \sum_{N(x)} u(x) - f(x) \right)^2 dx$$

By the Euler-Lagrange equation we have:

$$\frac{\partial u}{\partial t} = \nabla \cdot \frac{\nabla u}{|\nabla u|} - \frac{2\lambda}{\|N\|} S^{\mathrm{T}}(Su - f)$$

Here we use S to denote a symmetric square selection matrix whose side length is equal to the area of the patch being considered. It each row index corresponds to a pixel when the original patch is stacked into a vector. Each element of that row is one if the pixel corresponding to the column is in the same neighborhood and zero otherwise. This $\frac{\partial u}{\partial t}$ term is used to iteratively update the solution a fixed number of times.

4.2 Simulated Annealing

To improve the image enhancement it makes sense to add addition information. We consider the case that we have hyperspectral data, and we are given unmixing information in addition to our grayscale image. This additional information tells us what percentage of each local neighborhood belongs to a collection of predefined classes, such as buildings, foliage, water. We now attempt to use this new information to assist in improving the image patches.

Simulated annealing [3] is a combinatorial approach where we can ensure that the constraints are always satisfied. The trade off is that the method generally runs much slower.

This is because a combinatorial approach is generally intractible. Simulated annealing attempts to find a close solution in a predetermined amount of time. It models the physical processes of the metallurgic process of annealing, and thus uses terminology such as temperature and cooling rate. Since simulated annealing uses random choices, it requires many iterations to find a good solution.

Our implementation of simulated annealing follows the general form, but we elaborate on the state transition function. In order to encourage meaningful transitions we weight the chance of selecting a new state by the change in potential it causes. We use total variations for the potential of the image. This prevents swapping two pixels of the same class, which has no net effect. We limit pixel swaps so that nothing enters of leaves a local neighborhood which is how we ensure the constraints are met at all times. We do not limit pixel swaps to adjacent pixels, since many times a better solution would require passing through one or more higher potential states using the adjacent swap model.

4.3 Tests

For these tests a satellite image of a harbor from Google Earth was used. The image was blurred and downsampled to $\frac{1}{4}$ the original resolution. This image was used as the test image. The original image was used as the ground truth image. The Harris corner detection method was applied to the test image and the coordinates of the corners were converted back to the ground truth resolution. A 12x12 patch, which is equivalent to a 3x3 patch in the test image, was extracted and passed to the two methods.

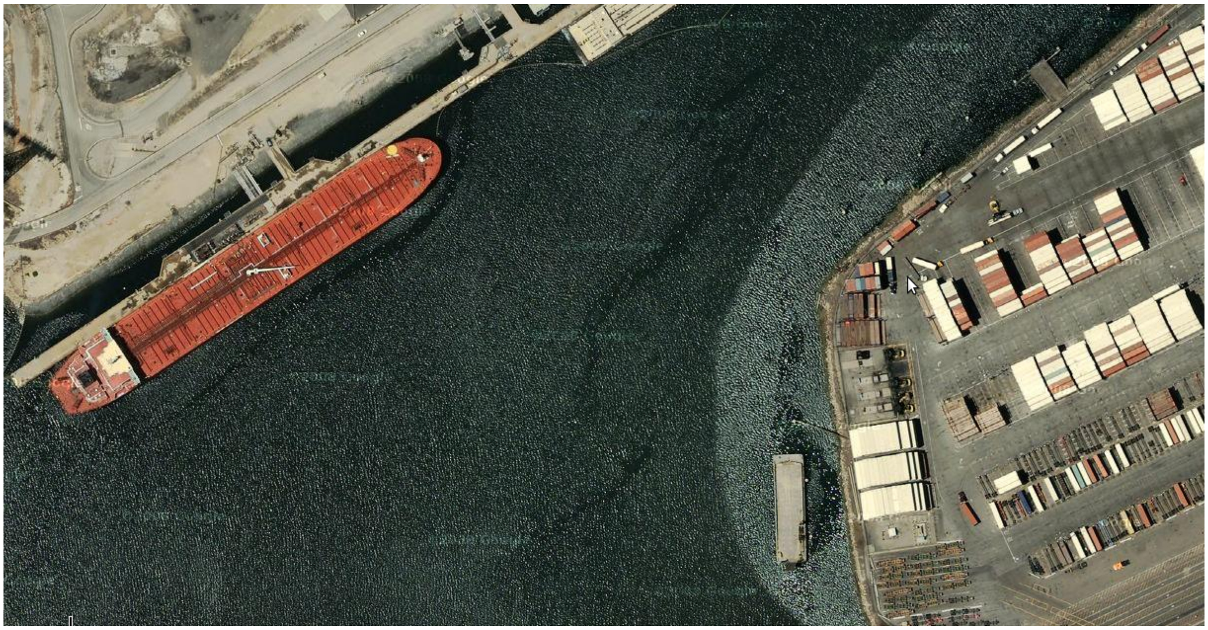
4.3.1 TV with Neighborhood Constraints

The image patches of size 12x12 were averaged across 4x4 neighborhood block to reobtain the equivalent test image patch, but upscaled to fit the target size. It was used as both the initialization image and the reference image for the minimization method. The method was run for 500 iterations using $\lambda = 0.2$ and a step size of 1.

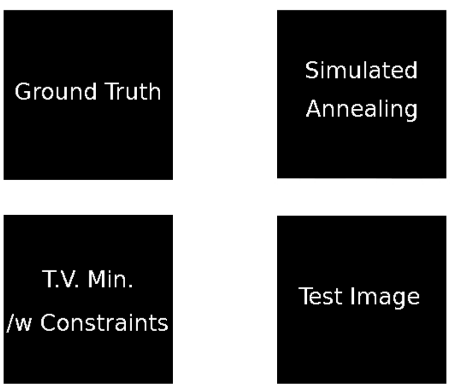
4.3.2 Simulated Annealing

To simulate unmixing information we start with the ground truth pixels. We then randomly select to add 1, subtract 1, or not change each pixel and create a histogram of the values in each local neighborhood. Then each neighborhood is recreated with the knowledge of the values, but not the locations. Each pixel is placed randomly within its original window. Then simulated annealing is run for 15,000 iterations with a starting temperature of 200, and a cooling rate of 0.999. The run time per patch for simulated annealing was approximately 90 seconds.

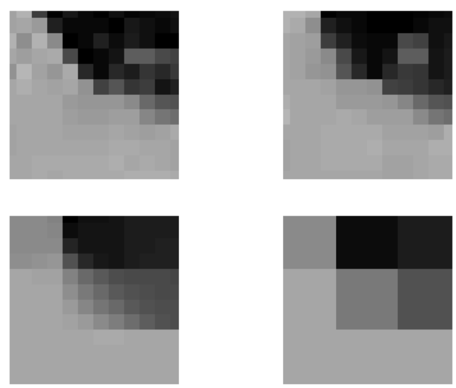
4.3.3 Test Results



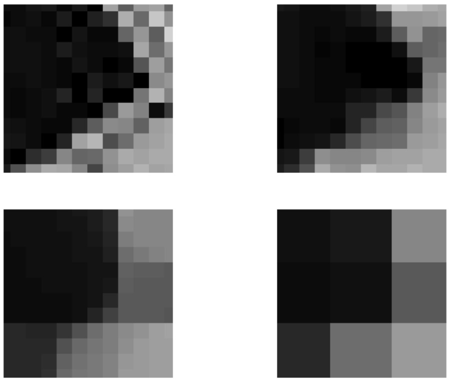
Original image



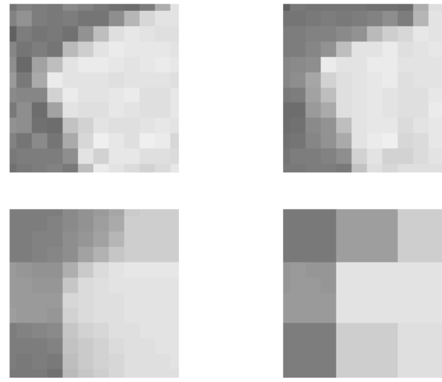
Results layout



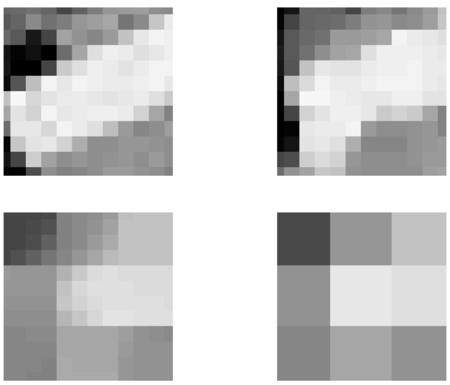
Edge



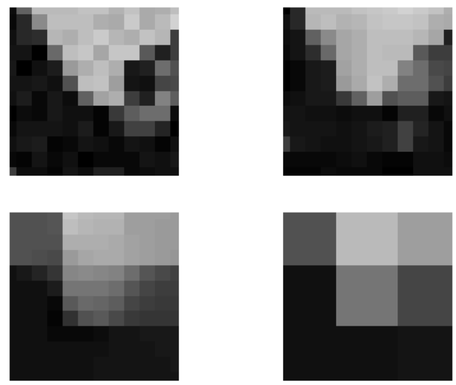
Black on white corner



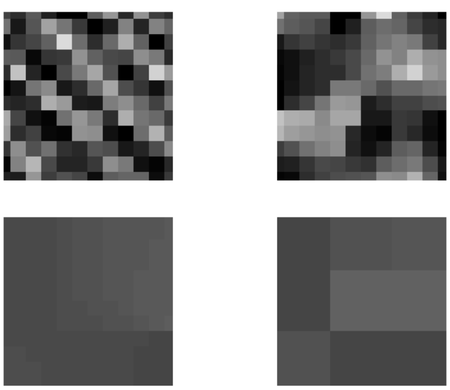
White on black corner



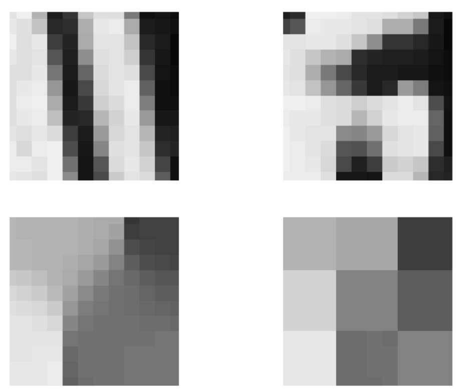
Bar shape



Sharp Corner



Textured patch for which methods do not work



Striped patch for which methods do not work

4.4 Conclusions

4.4.1 TV with Neighborhood Constraints

This method does not work well most of the time. It seems that many times it diverges wildly. It may be due to the fact that the selection matrix is causing a large discontinuity, making the derivative unexpectedly large and the process unstable. Even when the method does not diverge, it is often too blurry or too similar to the starting image to be useful. While this method runs very fast, it does not perform well as it is. It seems like there is room for improvement but it is not a useful tool in the way it is currently formulated.

4.4.2 Simulated Annealing

This method performs very slowly, with an average run time of about 90 seconds. For image patches with edges and corners it performed very well most of the time. There were some cases where the image patch was more texture than edge or corner. In these cases simulated annealing suffered, producing nonexistent edges or amorphous blobs that did not resemble the original image. While this is an error on the part of the corner detector more than the corner enhancer, it is sometimes impossible to discern between corners and some textures at low resolution, and is a case that should be handled appropriately.

5 Future Work

The line tracing method failed to produce consistent results, and the results it did produce did not seem to be a large improvement. It seems unlikely to ever be dependable enough for use in any real world application. The TV minimization with neighborhood constraints method did not work as formulated, but could possibly be improved. It is reasonable that it might work if we can ensure that it will converge. The simulated annealing method was slow, and had problems with textured areas, but was strong with edges and corners, and seemed to be fairly invariant to rotation. Future work with this method would be to increase the robustness to textures, and compare how it improves registration. Currently it is capable of giving 0.25 pixel resolution, but this could be increased if coupled with quadratic interpolation.

References

[1] Z. Guo, T. Wittman, and S. Osher. "L1 Unmixing and its Application to Hyperspectral Image Enhancement." *Proc. SPIE Conference on Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XV*. Orlando, Florida. April 2009.

[2] C. Harris and M.J. Stephens. "A combined corner and edge detector". *Alvey Vision Conference*, (1988), 147152.

[3] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi *Science*, New Series, Vol. 220, No. 4598 (May 13, 1983), pp. 671-680

[4] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. "Equations of State Calculations by Fast Computing Machines". *Journal of Chemical Physics*, (1953), 21(6): 1087-1092.

[5] H. Moravec. "Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover". *Tech Report CMU-RI-TR-3 Carnegie-Mellon University, Robotics Institute.*, (1980)

[6] A. Noble, "Finding Corners", *Image and Vision Computing Journal*. 1988, 6(2): 121-128.

[7] L. Rudin, S. Osher, E. Fatemi. "Nonlinear total variation based noise removal algorithms". *Physica D* 60 (1992), 259268.

[8] J. Shi and C. Tomasi. "Good Features to Track". *IEEE Conference on Computer Vision and Pattern Recognition*. 1994, 593-600.