EE368 Final Report

TJ Melanson

December 2016

1 Introduction

Common feature tracking algorithms, such as SIFT and SURF, are fairly slow in runtime due to the processing of a large amount of external data. If the object is sufficiently small, outside noise may throw off the object detection device without prior knowledge. Machine learning, especially Markov chains, can use prior knowledge to turn a computationally expensive task into a faster, stochastic one. This project attempts to implement the mapping portion of SLAM, i.e. to develop a mobile application that could create a planar grid given a known reference, then track an object along that grid. In order to constrain the object field with a fixed camera. the algorithm produced adequately tracks a cube with image refinement. Opportunities exist to create a better template for homography estimation than the first image, which will reduce noise in detecting keypoints, to have a generalized algorithm so a template can be chosen on Android without further tuning, and to further integration of the masking region.

2 Procedure

2.1 Initial Shape Estimation via Edge Detection and Ellipse fitting

To estimate the initial shape of the template, the shape was computed as an actual geometry. The shape estimation algorithm used OpenCV's contour estimation from Suzuki combined with a conic least-squares estimation method created by FitzGibbon.

As a prefilter, the template was first denoised via a 3x3 Gaussian kernel with standard variance. Then, a Canny edge detector was used to find the image binary. To find the optimal threshold, the upper threshold was kept at 2.5 times the lower threshold. The lower threshold was first placed at 10, then increased by 10 until all images could detect the shape in question. In the case of the experiment, the optimal lower threshold was 50, and upper threshold was 125.

The contour fitting took the edge map and translated it to a series of points rather than a binary image. This allows the shape fitting algorithms to more easily process the shapes (as the edge points are explicit rather than implicit).

In short, the conic fitting method takes the contour points, and estimates the conic shape (in this case, an ellipse) that minimized the linear distance between the shape and the sets of points. Similar to a Hough transform, it fits a shape to the set of points. However, it uses regression of parameters instead of a most-likely estimate in a discrete parameter set.

Initially, the goal was to use Hough transforms to create a set of lines, then use RANSAC to determine the rectangle. However, Hough transforms could not accurately find all four edges without extraneous noise. In addition, the lines would have to be chosen such that exactly two sets of two lines were approximately parallel, and those two sets were perpendicular to each other, which proved less practical in implementation. To estimate a full 2-D curve, sufficient parameters (in this case, 5) for the curve would need to be placed into the Hough transform. In this case, using least-squares fitting becomes simpler, both in terms of complexity and space usage.

2.2 Testing the Kalman Filter with Simple Ellipse Estimation

Because the homography estimation can be prone to error, the Kalman filter for the image parameters was first used purely against the geometry of the detected ellipse. Instead of the homography vector $\vec{h}_k$, there was a geometry vector $\vec{x}_k = x, y, w, h, \theta$, where (x, y) is the center of the shape, w,h are the width and height, and θ is the shape angle. Because the eraser in the image was a prominent feature in the background, the shape in question was estimated as the largest detected shape in the image.

2.3 The Kalman filter

The Kalman filter is divided into two steps: a prediction and an estimation. The prediction value of the filter relies on the previous estimate. In this case , the predicted value is the expected value of the previous rate of change plus a random walk, which is simply the value of the previous estimate (as we assume the walk is zero-mean). In other words, the prediction equation can be modeled as follows:

$$\bar{v}_{P,k+1} = \bar{v}_{E,k} + E\{w\} = \bar{v}_k$$

where $\bar{v}_{P,k+1}$ is the average value of the velocity at the prediction stage of iteration $k+1$, and $\bar{v}_k$ is the estimation stage expected value of the current iteration.
In the estimate step, the Kalman filter uses the estimate z_k to get a better estimate of the actual state. According to the Kalman filter, the estimation

stage updates the estimated value according to a gain G, the measurement, and the prediction stage:

$$\hat{v}_{E,k} = \hat{v}_{P,k} + G(z_k - \hat{v}_{P,k}) = \hat{v}_{P,k} + G\alpha_k$$

where $\hat{v}_{E,k}$ is the estimated value of v_k at the estimation stage, $\hat{v}_{P,k}$ is the estimated value at the prediction stage.

The optimal value of G is determined based on the estimated variance and the updated noise variance:

$$G = \frac{1}{1 + \frac{\sigma_w^2}{\sigma_P^2}}$$

The variance of the update noise, σ_w^2, is assumed constant. The prediction noise σ_P^2, however, is the result of two errors: the estimation error noise and the measurement noise. Because these two values are independent of each other, the noise can be described as the sum of the noise variances:

$$\sigma_P^2 = \sigma_E^2 + \sigma_v^2$$

σ_E^2 was determined recursively starting from 0, and σ_v^2 was determined as the difference between the overall variance of the measurement error minus the variances of the noise and prediction.

2.4 State Model of the Experiment

For this project, the state model used was the discrete rate of change between the homography parameters of the current frame and the previous frame. In other words, the current measurement for the stage z_k is defined by:

$$z_k = \vec{h}_k - \vec{h}_{k+1}$$

where $\vec{h}_k$ is the vector of the eight homography vectors at iteration k, calculated by estimation of the homography at point k.

The hidden variable is the estimated rate of change v_k. Because the state is a vector in this case, the state is now expressed as a vector $\vec{v}_k$. This means that the predicted value of the homography vector, $\hat{\vec{h}}_k$, is estimated as a sum of the previous measured homography and the current prediction of the homography change rate:

$$\hat{\vec{h}}_k = \vec{h}_{k-1} + \hat{\vec{v}}_{P,k}$$

2.5 Region Masking

Region masking allows the algorithm to make use of the information provided by the Kalman filter. Because the reference paper did not explicitly mention a method to rank the most likely regions of the next image, each input frame was masked for feature detection. The resulting descriptor extraction and matching with the template images came from the modified image features. Although it could be possible to rank the features by distance from the estimated homography, the region masking method greatly simplified the image processing steps.

As a heuristic, the region mask was approximately 1.5x the size of the original template shape along each dimension, resulting in 2.25x the area of the original image. This was chosen on two bases: that there would be some error, which would mean a region exactly the size of the original template image could exclude important edge features in the current frame; and that the center point would at least be within the prediction region, which would make the region mask twice the size of the original frame. Because there was no significant scaling in the image, the size of the predicted shape was approximated as the size of the original shape, although the scaling would could be expressed as multiplication of the size vector by the homography matrix $H\vec{s} = H\{w, h, 0\}^T$.

The key difficulty in implementing region masking was making sure the keypoint detection was not including the mask edges as keypoints. When just running the image through a mask before running it through the algorithm, this would be a major issue, especially since the background color was light. The issue showed itself when the features detected were along a line inside the shape of the eraser. To fix this, the mask was provided as an input to the keypoint detector.

2.6 Determing Error

Error was determined by the accuracy of the predicted geometry versus the actual geometry of the system. For the majority of the experiment, this error was determined visually (see the results). The new ellipse geometry was written on the image, and compared with the location of the shape. However, in order to quantify the error, the exact centers were hand-chosen via Matlab, and then compared to the predicted values via the squared L2 norm.

3 Results

In order to test the algorithm without extra noise, the algorithm was tested on a series of 17 images taken in rapid succession by an Android camera, then downsampled to a 900 by 500 image. Because it was assumed that the background of the image was constant, the camera was moved instead of the shape (an eraser), which simulated translation of the eraser roughly along the negative y direction.

The Kalman filter was much better at dealing with the noise of the outside image. In particular, the non-masked feature detector would detect features on the shadows on the desk, such as one on the upper left corner and those cast by the eraser. The masked image, however, retained only the features of the shape itself, resulting in a much more accurate homography estimation.

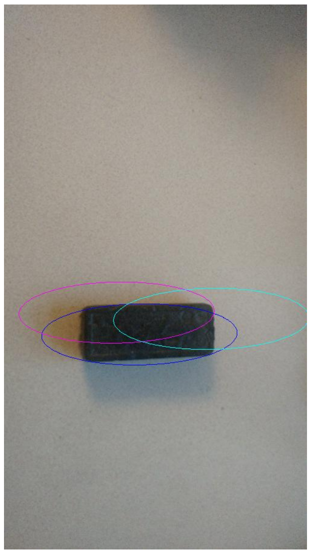

Figure 1: Image at iteration 10. Blue: Kalman filter predicted shape, approximated as an ellipse. Magenta: Result of pure homography estimation at frame 10. Cyan: using the original region mask as the region mask for the image

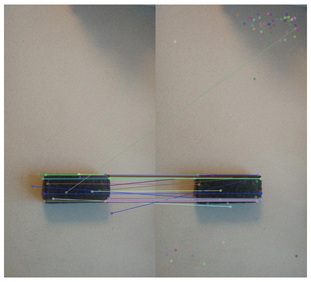

Figure 2: Masked image feature matching

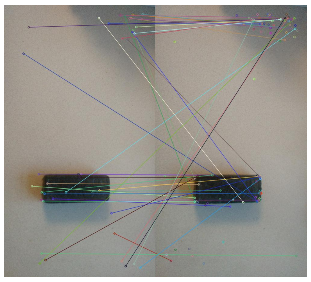

Figure 3: Non-masked image feature matching

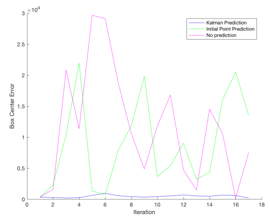

Figure 4: Comparison of three methods of homography with Kalman filter prediction and region masking, region masking based on the first image, and blind estimation (control).

The second hypothesis was the possible speed increase due to the decrease in area with which to detect features. To time the functions, the system time in nanoseconds was recorded just before and after the declaration of the processing method. The "masked" method was the final Kalman filtered output, using the predicted region, while the "non-masked" method calculated the homography with a mask the size of the entire image.

Figure 4 shows the error of the algorithms' estimated center point of the shape as determined by the squared L2 norm of the distance between the predicted and a ground truth. The ground truth was determined by manually finding the center point of the image. The Kalman masking algorithm (blue), compared to the pure homography estimation (magenta), is near 0 (< 100 square pixels as opposed to order 10^4).

One issue that could be raised was the relevance of the Kalman filter to the improvement. If the region masking was used without using predictive methods, it would ruin the purpose of the algorithm. Therefore, a third algorithm, which used the initial mask estimate for all frames, was implemented. As shown in Figure 4, the region masking alone (cyan) did not significantly improve accuracy of the algorithm, even after only 2 or 3 frames. This shows that adaptive prediction was a key factor to the success of the algorithm.

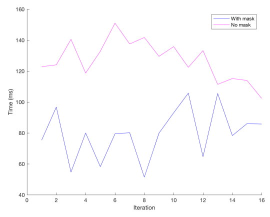

Figure 5: Comparison of homography estimation runtimes with region masking and without region masking. On average, region masking cut runtime by almost 40%

The average runtime of the masked function was 79.6 ms, while the average runtime of the non-masked function was 127 ms. Although this speed would still make it a bottleneck in the backend calculations, it shows the potential to run at almost double the speed of continuous homography estimation.

4 Conclusion

The Kalman filter and region masking algorithm significantly enhanced the accuracy of homography estimation, while providing a strong processing speedup. The algorithm went beyond that of using the region masking, which did not significantly increase accuracy alone.

Overall, this algorithm is well suited for real time tracking of an initially given shape. It is both robust to noise from the outside image, as well as moderately faster than pure homography estimation via ORB.