Detection of Street Signs

Tom Kawchak

CMPEN 454 Honors Project

# Introduction

For the honors option project, I decided to create a program in MATLAB that detects street signs in images. The program uses many of the concepts that we used in class in order to match template street signs in the images that are presented. The program works by identifying SURF features in an image and the template street sign, matching points that might be correspondences, and then using a RANSAC-based homography estimation algorithm.

# Program Details

The first part of the program that I needed to create was a function to turn an image into a template of desired size. I created a makeTemplate function for this purpose. This task was done by first converting a colored image into a gray-scale image. Secondly, the image was scaled to the desired size and then padded with a border of white. I used this function to produce template images of a stop sign and a speed limit sign. These were saved to a file to be used later. It should be noted that all of the filters are a white square with an image of a sign on them.

The second and most comprehensive step was to make a function that would take an input image and a template image and attempt to detect the template image in the input image. For this purpose, I created the matchTemplate function. This was accomplished by first converting an image to grayscale and then detecting SURF features in both the template image and the input image. MATLAB functions were used to both detect SURF features and then estimate possible point matches between the images. After this, I used the points to estimate a homographic transformation of the template sign into the scene image. First, I sorted the matched points based on the goodness of the match and then took up to the top twelve matchings to use for estimating the transformation. This choice of twelve came from experimentation and run time constraints since the number of possible transformations to check for fitness increases exponentially. For this homography estimation, I took all possible combinations of size 4 of the top twelve matching pairs of points and ran a modified RANSAC algorithm. I took the combination that resulted in the greatest number of inliers with all matched points. Once the best homographic transformation was determined, I had some checks to make sure that the estimated homography was reasonable or not. I checked to see if the detected bounding shape for the sign was approximately square to get rid of bad estimations that could result from bad point matches. After that, I plotted the bounding shape from the projection of the template onto the scene image and displayed the results.

The last step for this algorithm was to run the matchTemplate algorithm for all of the combinations of templates and scene images. All of the images for this project were taken from the results of searching google for images. These images that I downloaded can be split into three categories. They can be described as images that have speed limit signs in them, images that have stop signs in them, and images that have neither stop signs nor speed limit signs in them.

# Results

The program was able to give good results for sign detection. All of the tests can be found as images in the same folder as this file. The output image files all have the following format for name: <template>\_<image type>.jpg. Looking at the images will show which signs the program detected a sign for. Each picture shows the point matches between the template and the scene image on the left side. The right side of the picture shows the scene image with a green quadrilateral around the detected sign if one was detected. If no sign was detected, then no right image is shown. We can see easily see the results of the algorithm in the following confusion matrix:

|  |  |
| --- | --- |
| 19 | 7 |
| 0 | 46 |

In other words, 19 images that contained the corresponding template image were classified as containing the template image. Additionally, 7 images that contained the template image were classified as not containing the template image, and 0 images that did not have the template images were classified as having the template image. Finally, 46 images that did not have the given template image were classified as not having the template image.

# Conclusion

As can be seen by the results above, the algorithm correctly detected the presence of a sign 73% of the time. The algorithm never detected a sign that was not present in the image. This shows that the algorithm used would be better if it detected the template sign in the scene image more often, assuming that it should. The algorithm seemed to perform best when the MATLAB SURF feature detection and matching function returned more matching values. One possible improvement for the algorithm would be to use more filters for the detection of SURF features, and another would be returning more matching features between the template and the scene image. One prohibiting factor for the algorithm was run time. I found that the standard RANSAC function often did not detect the sign because it sampled points that were not correct matches. I modified the algorithm to use the best matches, but trying many of these combinations resulted in a longer run time. Finally, another improvement that could be made to the algorithm would be in the evaluation of a match. The evaluation criteria for the goodness of the homography could have been too strict. This would have resulted in matches being discarded even if they should not have. These are all directions that could be explored to improve the algorithm.