Object Position and Orientation Detection For Mars Science Laboratory Helicopter Test Imagery

Michael Johnson
EE 368 – Stanford University
mpjohnso@stanford.edu

***Abstract*—This paper describes image processing methods for identifying the position and orientation of a rover mock-up in images taken during a Mars Science Laboratory field test campaign. Identification targets were affixed to the top deck of the mock-up, and the methods discussed involve the detection of these targets, followed by associating them with the known positions of the targets.**

I. Introduction

Mars Science Laboratory Terminal Descent Sensor, the rover's landing radar, underwent a field test campaign in the Mojave Desert. One purpose of the field test was to gauge how the rover's position during the sky crane descent phase might impact the return of the radar signal during this crucial phase of landing operations. To test this, the landing radar was affixed to a helicopter, and a mock up of the rover, with targets attached to the top deck, was attached to a wench underneath the helicopter. The radar was operated while the position of the rover mock up was varied. A camera was positioned near the wench, facing down, to image the rover as the test was underway. Time-tagged images were captured at a rate of 6 frames per second, which will be used to estimate the position of the rover during the test and matched with specific times of radar test data.

II. Problem Statement

The identification targets affixed to the rover top deck will be the primary means of solving for the rover position and orientation in the imagery.

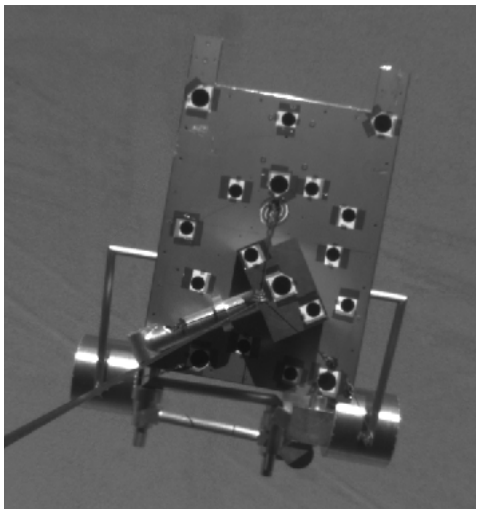

Figure 1: The rover mock-up with targets.

The targets are black circles on various places on the rover top deck. The contrast between the targets and the rover top deck should aid in proper identification, however the task is made more difficult by the extra clutter of the white square shape around the target, and the dark-colored tape used to attach them to the rover.

Similarly, the highly-reflective nature of the rover surface may cause unwanted side effects such as glare that can cause problems in identifying some of the targets. It would be more desirable if the rover surface was painted a non-reflective white and the targets did not have the extra clutter of tape around them. However, as the images are this way and nothing can be done about it, these problems must be dealt with in post-processing. Shadows cast by the helicopter and the rover mast may also cause contrast issues that will pose problems for target identification.

In each image, the rover can be considered to have undergone three translations:

1. Translated along the X and Y axes.
2. Scaled in size depending on the distance from the camera.
3. Rotated due to helicopter maneuvers, wind, and other factors.

Roll and pitch effects are present, but assumed to be minimal due to the nature of the test and will be ignored in processing.

III. Proposed Solution

It is proposed to implement a three step process to identify the targets in each image.

A. Edge Detection via the Sobel Operator

The first step in detecting the targets is to separate them from the background and the rover shape. The proposed method to do this is to perform edge detection via the Sobel Operator [1] applied in both the horizontal and vertical axes. Detection in both directions is necessary to properly detect circular edges.

The Sobel Operator is a 3x3 convolution kernel that is applied to each pixel in an image to approximate the derivative of the image along a direction. For the horizontal direction, the kernel is defined as:

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \qquad (1)$$

For the vertical direction, the kernel is defined as:

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} \qquad (2)$$

The kernels defined in (1) and (2) are convolved with every pixel in the image to detect edges. A threshold is applied to thin the edges as well as avoid detection of non-circular shapes, which will aid in further steps. For this problem, a threshold value of 0.08 was used, which provided good performance.

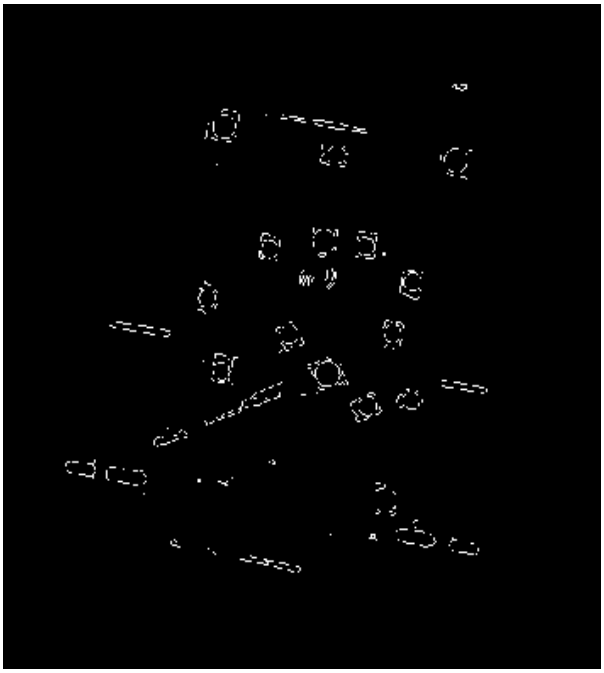

Figure 2: The rover mock-up after Sobel edge detection.

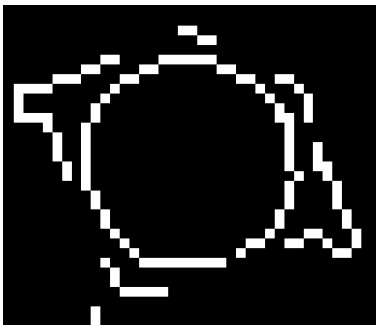

Figure 3: The close-up of one of the targets.

As is shown in figures 2 and 3, this method does an adequate job at detecting the circular targets, while rejecting some of the more linear edges, such as the sides of the rover deck against the background. It is also noted, however, that in this case the three targets visible at the bottom of the deck in figure 1 that are obscured by the shadow of the mast are not detected because the contrast between the foreground targets and background deck is not high enough. This is an acceptable limitation for the following reasons.

First, there are a large number of targets on the rover deck, so missing only a few should not completely prevent successful detection of the rover position and orientation. Second, the frame rate at which the images is taken is high enough compared to how much the rover is able to move in between images that occasional failures to track are OK. With these factors in mind, edge detection via the Sobel operator is an adequate first step in the processing chain.

B. Circle Detection via the Circular Hough Transform

Although Sobel edge detection can successfully extract most of the targets from the image, as is evident by Figure 3, there is still plenty of clutter that must be dealt with. Because the targets are practically the only circular shapes in the images, we must now focus on extracting only the circular shapes, and their positions, from the edge-detected image.

The proposed method for this is the Circular Hough Transform [2]. The Circular Hough Transform is useful for detecting circles of known radius in a black-and-white level image, but can be iterated to detect circles of various radii. The method to find a circle of radius R is as follows.

1) Create an accumulator matrix the size of the original image to be processed. This accumulator matrix will be used to store data that is used in the peak-finding step.

2) For every bright-level pixel position in the image, draw a circle of radius R in the accumulator matrix. This has the effect of incrementing the value of the accumulator by one every time a drawn circle intersects a pixel in the accumulator matrix.

3) After all circles have been drawn, search for peaks in the accumulator matrix. Peaks in the accumulator matrix represent a circle in the original image being cenetered at this pixel. The reasoning of this is as follows:

Consider a circle in an image:

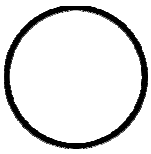

Figure 4: A circle in the original image.

For every pixel in this circle, additional circles are drawn in the accumulator matrix, as illustrated in figure 5. As more and more of these circles are drawn, they all have pixels that intersect at the center of the circle in the original image, as illustrated in figure 6.

This result of the Circular Hough Transform is how we will find the centers of the targets in the edge detected image. As the accumulator matrix is created, we are left with peaks at the

center of each circle in the original image. We now are left with the simple task of searching for peaks in this accumulator matrix.

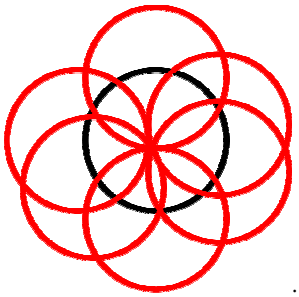

Figure 5: Circles centered at each pixel of the original circle are drawn in the accumulator matrix.

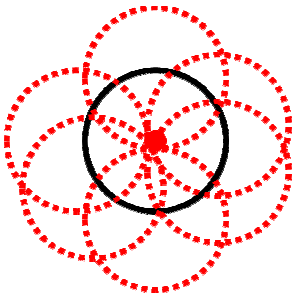

Figure 6: All of the circles intersect at the center of the circle.

An edge detected image processed through the Circular Hough Transform is shown in figure 7, with a close up of a single target shown in figure 8. As is pictured, the targets definitely stand out more than any other feature of the rover, exactly what we wanted.

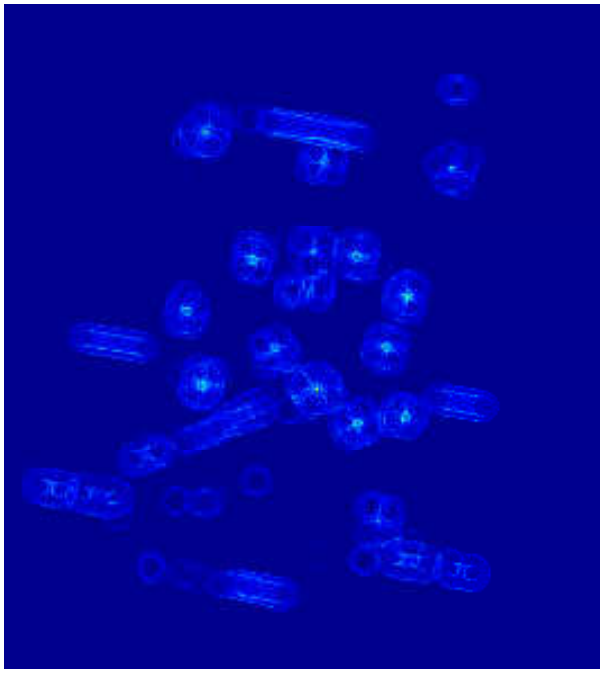

Figure 7: An image of the accumulator matrix values.

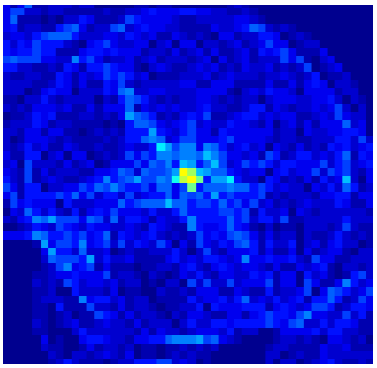

Figure 8: A close-up of a single target. Notice the strong peak in the center of the circle, what we are looking for.

Peak-finding in 2-dimensions can be done quickly and simply by just comparing the value of each pixel with the values of the eight pixels that border it. This method has shown itself to be fast enough for our purposes, and was selected.

However, peak-finding alone cannot be used. This is because the Circular Hough Transform really only detects perfect circles. As is evident by figure 3, the circles resulting from the edge detection stage are not perfect. The side-effect of this is the Circular Hough Transform produces a set of peaks near the center of the circle. Similarly, if the circle is not the exact radius of that being searched for, we will get a cluster of smaller peaks near the center of the target.

To resolve this issue, after peak-finding we find other peaks that are within an R radius distance from the peak being analyzed. For all peaks within an R radius of each other, we collapse them into a single peak located at the mean position of all the peaks. This process is illustrated in figure 9.

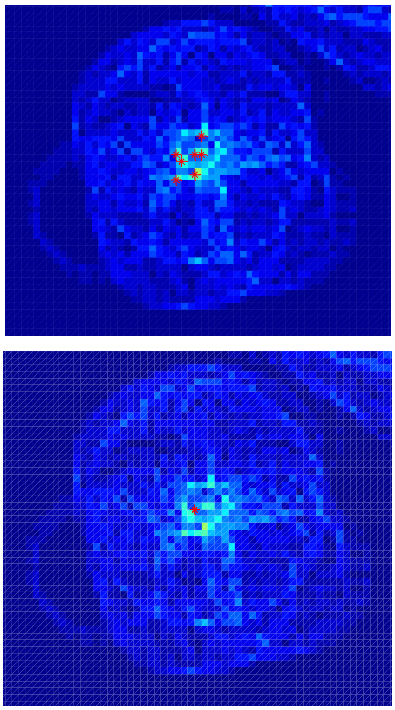

Figure 9: Multiple peaks (top) within a circle collapsed into a single peak more centered at the middle of the target.

After the processing techniques mentioned, we can successfully identify the locations of targets in the original image, as shown in figure 10. As is illustrated, this method is highly successful at extracting the locations of many of the targets in the images.

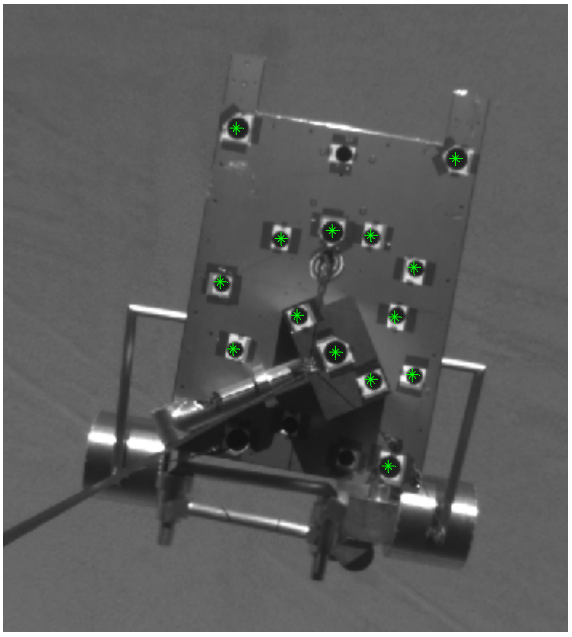

Figure 10: The green stars represent the locations of the filtered peaks form the accumulator matrix.

C. Target Alignment via Procrustes Method

Now that we have some (or possibly all) of the targets identified in the image, the final step is to align them with the known actual targets. The proposes method to accomplish this task is an iterative Procrustes Alignment Method [3].

Procrustes Alignment is a method developed to compare the relative shape of objects in multiple images that differ by means of translation, scale, and rotation, which are the transformations we have assumed our rover to have been subjected to in each image. The detailed method chosen is as follows.

1) Pre-process the known target locations by translating them along the X and Y axis such that the mean positions along both dimenions is zero. This effectively centers the image to be detected in the reference frame. If we have n known targets, for each (x_1, y_1) ... (x_n, y_n) points, we perform the operation:

$$(x_k', y_k') = (x_k - \bar{x}. y_k - \bar{y}) \qquad (3)$$

Where $\bar{x}$ and $\bar{y}$ are the mean known target positions.

2) After translating the target positions, scale them in magnitude such that the RMS distance of all points to the origin is one. This operation makes the method scale invariant. We calculate a scaling factor:

$$\alpha = \sqrt{\frac{\sum_{k=1}^{n}(x_k')^2 + \sum_{k=1}^{n}(y_k')^2}{n}} \qquad (4)$$

Each translated point is divided by this scale factor, giving us the desired unity RMS distance of all points to the origin.

3) For each image to be processed, perform steps 1 and 2 to the detected target pixel locations. This puts the detected targets to be on the same scale as the known targets.

4) After the detected targets are translated and scaled, perform an iterative rotation alignment. Sweep over a range of angles from 0° to 360°. In each iteration, for each detected target point, calculate the distance to all of the known target points. Choose as its pair the closest known target. Calculate as a cost function the sum of the square of these distances for all of the detected targets. The rotation angle to select is that which has the smallest cost function.

We now should have a relatively close match of the detected targets to the known targets. However, because not all of the targets will have been detected, the translation and scaling will not have put the detected points at the same reference frame as the known targets. Therefore, we perform another series of iterations on the translated, scaled, and rotated points.

The first iteration is to sweep over a small range of additional X and Y translations, re-calculating the cost function at each iteration and picking the additional translation that minimizes it. Next, we iterate over a small range of additional scale factors, calculating the cost function again at each iteration and again picking the additional scale factor that minimizes it.

At this point, we will have good agreement between the detected and known targets. To match a detected target with a known target, we simply choose the known target that is the smallest distance away, and perform filtering by rejecting points that are too far away from a known point (indicative of the detected point being clutter and not an actual target) or rejecting points that share the same known target (pick only the target that is closest, and reject the rest). We are now able to associate some (or possibly all) of the known targets with a pixel number in the image, which can be used for further pose estimation to get exact measurements. Pose estimation will not be discussed here.

IV. Results and Conclusion

An example result of this method is shown in figures 11 and 12. In this example, we are able to successfully match all of the detected targets with a known target. However, because the edge detection and Circular Hough Transform stages are not perfect, some targets are not associated. As previously mentioned, this is acceptable because of the large number of targets. If desired, the locations of the missing targets can be interpolated to complete pose estimation.

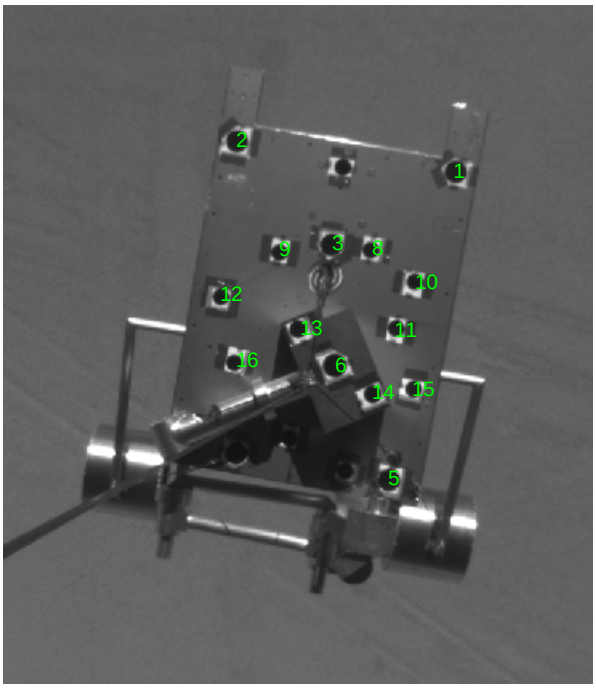

Figure 11: The detected targets matched with known targets

Figure 11 shows each detected target assigned a numerical reference to known targets. In this case, of the detected targets, 100% of them are correctly associated with known targets, although four of them were not detected prior to Procrustes alignment.

Figure 12 shows the how the target positions are aligned. Their relative positions after Procrustes alignment are in very good agreement. Small errors do not prevent us from associating a known target with a pixel number in the image.

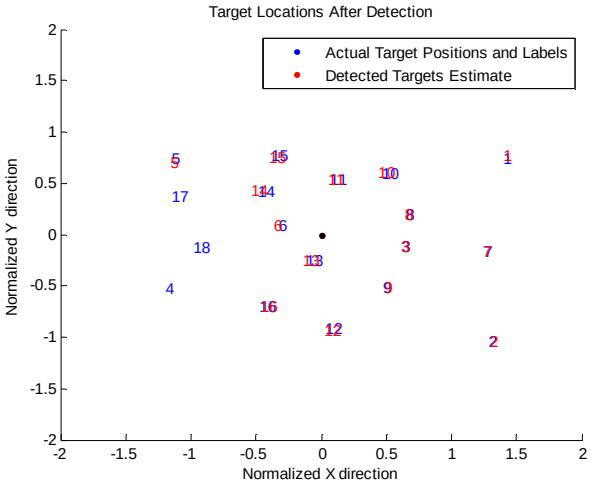

Figure 12: The known and detected target locations after translation, scaling, rotation, and alignment via the Procrustes Method.

Statistics were calculated for a small sample of 50 images run through this method. An average of 74.4% of the targets across the images were successfully aligned with a known target, plenty to later develop a good estimation of the rover position. Image quality differences, due to previously discussed effects of sun glare, shadowing, as well as excessive scale or translation cutting off some targets, does negatively impact performance. Some statistical plots are shown below.

To conclude, this method has proven itself quite capable of performing the task at hand. Images are processed in the sub-second time-frame with little optimization. The method is also fairly free of continual "fiddling" with parameters to get it to work, which is desirable for large sets of images.

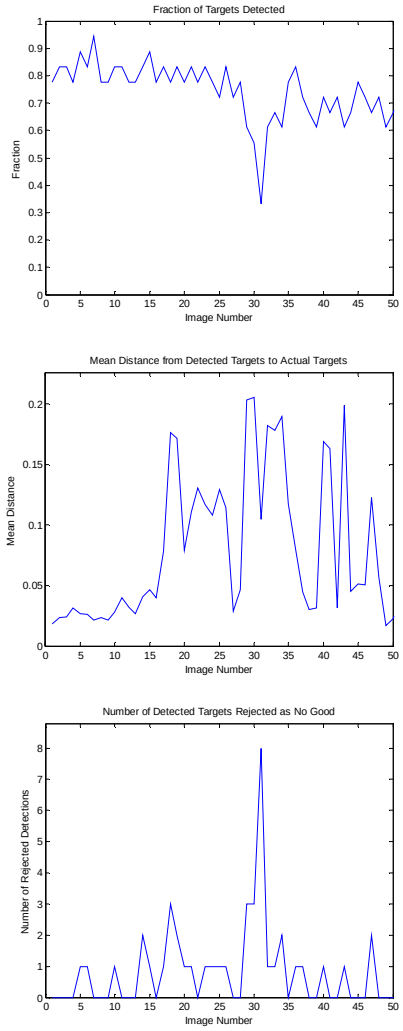

Figure 13: Statistical results of alignment.

REFERENCES

[1] R. Gonzalez and R. Woods *Digital Image Processing*, Addison Wesley, 1992, pp 414 - 428.

[2] Simon Just Kjeldgaard Pedersen. *Circular Hough Transform.* Aalborg University, Vision, Graphics, and Interactive Systems,November 2007

[3] B. Luo and E.R. Hancock. *Iterative Procrustes alignment with the EM Algorithm.* Image and Vision Computing 20 (2002), 377-396. December 2001.