Eric Perko and Chad Rockey

Michael Lewicki

EECS 600

Final Project

5/6/2010

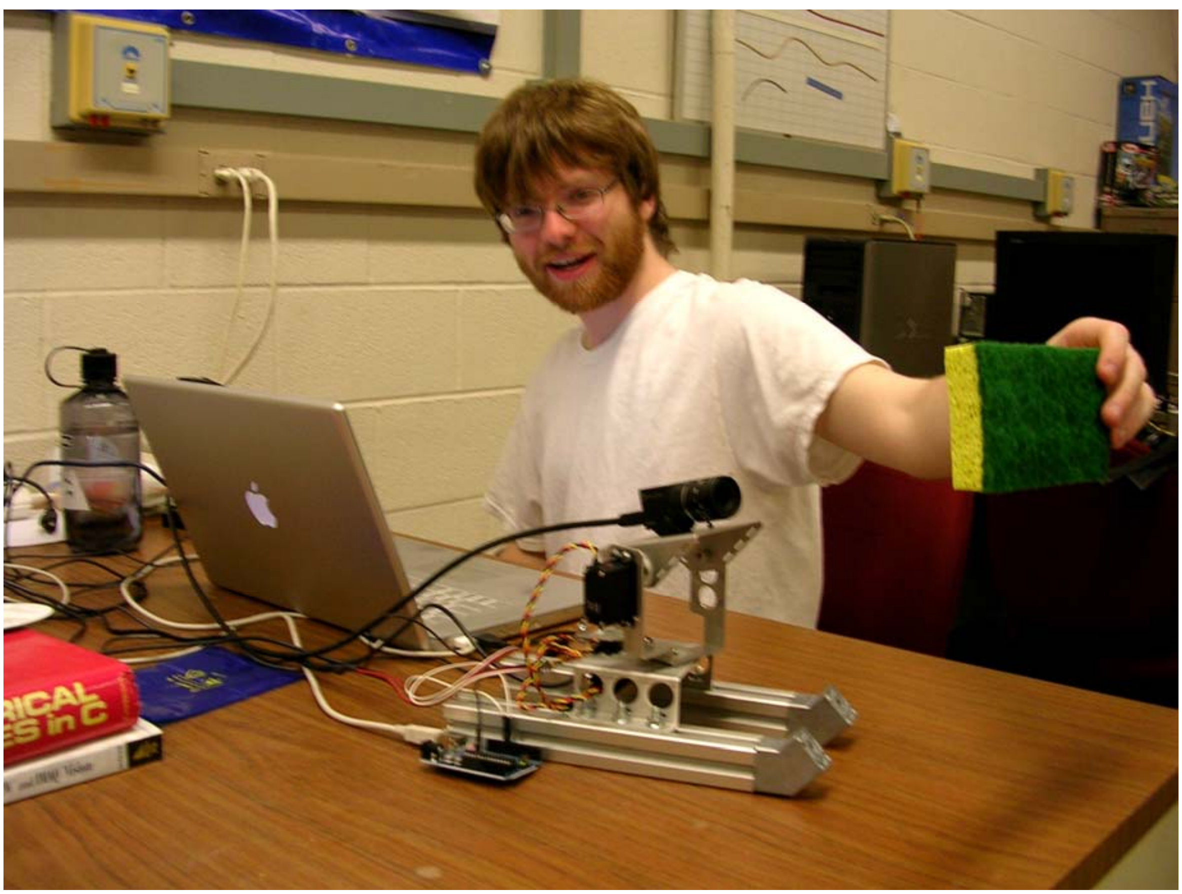
Tracking Moving Targets Using a Two Degrees of Freedom Camera

Problem Description

Our goal is to track a specific target that is moving in three dimensional space using our two degree of freedom camera. This can be very useful in many applications, for example target tracking for a projectile weapon or continuously viewing a desired event without the use of a camera operator. Our goal was then to track a specific article or motion and keep that object in the center of the frame.

We also wanted to try to use stereo vision to track the true motion in three dimensions and keep a representation of actual movement.

Figure 1: Image of Camera Mount Setup



Previous Research

We have seen target tracking of bright colored objects done before at CMU, for example, the CMUCAM or various Kalman Filter tracking of orange objects on YouTube. These trackers all used color segmentation to find the object within the frame. We decided to start here and then see if we could extend the algorithm to track general motion instead of simply a specific color.

Color Tracking I – RGB Segmentation

Our first effort to track an object involved color segmentation. We decided to track a specific shade of green that is common on a cleaning sponge but also appears on Mountain Dew bottles. Our algorithm took the following steps:

1. Color Threshold to find areas of color within a threshold of target color.
2. Erode these regions to reduce noise.
3. Dilate these regions to produce blobs.
4. Find the largest blob currently in the image.
5. Find the coordinates of the center of the largest blob.
6. Calculate the offset between this coordinate and the center of the image frame. This represents the cameras current error in tracking.
7. Run this value through a PID control scheme to minimize the offset between the center of the object and the center of the frame.

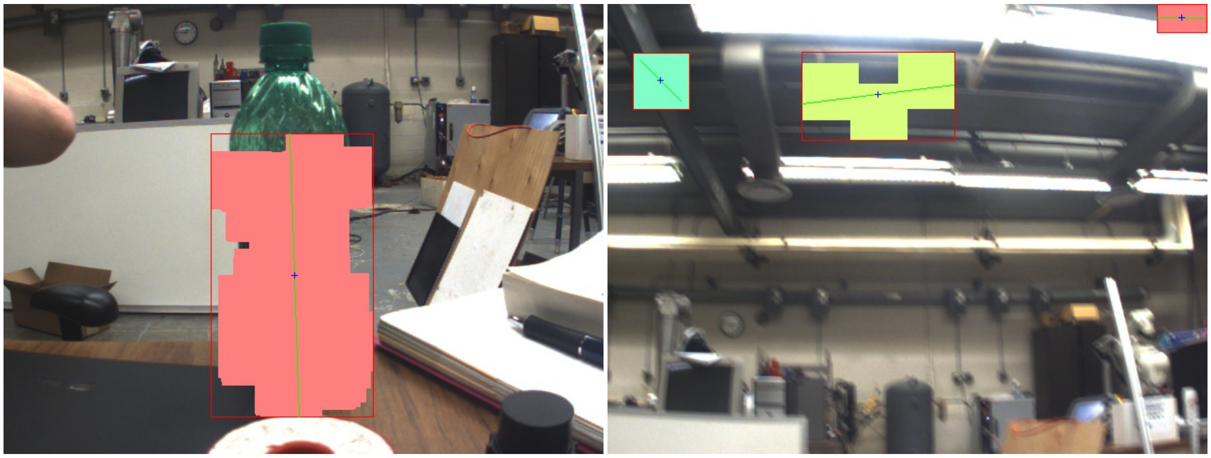
Analysis

This algorithm worked sufficiently well to keep the object in the center of the camera frame. The camera could easily follow the specified object, but there were a few issues.

The first issue is that color thresholding is dependent on the average intensity of the image. If the room was too dark or too light, then our colors would change and fall outside of the range of allowed values, thus losing the obstacle. It was also possible to have objects in the room fall within the ranges needed and “distract” the camera from its true target. This occurred when the camera faced the lights and the dark grey ceiling appeared to be green from the camera’s perspective.

Another issue was the original use of a telephoto lens. This lens focused too far away from the camera, resulting in a very narrow field of view. When we switched to a wide angle lens, we were able to get much better results from a wider field of view.

Figure 2: Object Detection and False Positive Detection of the Ceiling



Another problem was the speed of the camera. If we set the proportional gains too high, then the camera would oscillate around the center and not converge on the target. If we set the gains too low, then the camera would easily lose faster targets. We eventually settled on a compromised gain that tended to always track slow objects without oscillation but would lag too far behind moderately fast objects.

Color Tracking II – HSV Segmentation

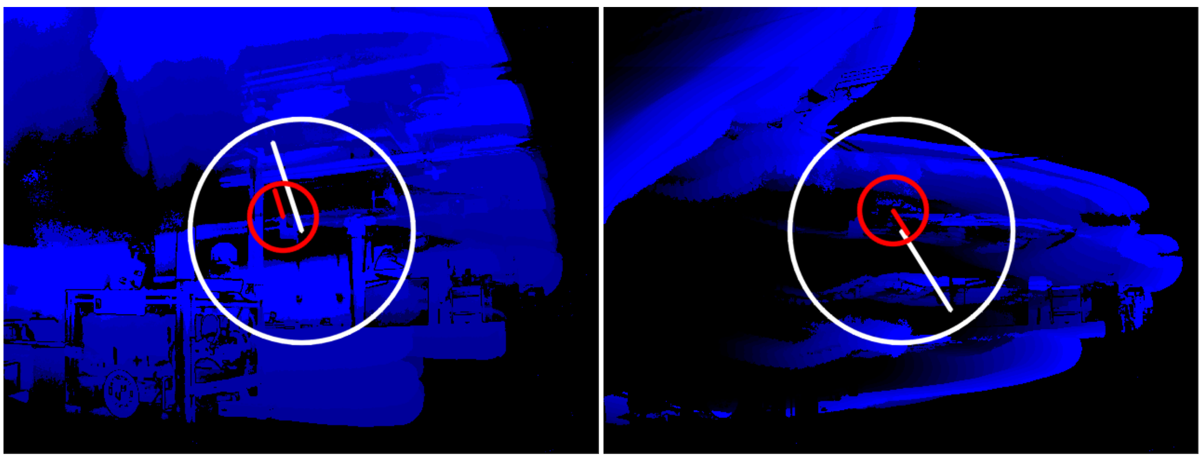
We made an attempt to overcome the noisiness and false positive detection of the RGB thresholding. To do so, we first converted our RGB images into HSV color space. HSV represents the colors as a cylindrical angle going through the color spectrum from red to orange to yellow, etc, from 0 to 360 degrees. The intensity of the color and brightness of the light are determined by the saturation and value components. Since we are interested in tracking green objects, we now segmented only the hue (color) channel around green. By doing so, we were able to track our objects in a much wider variety of lighting conditions while avoiding the false positives. In this way, we were able to improve upon our first algorithm, but we still had many of the negative effects such as the need to tune the tracking PID.

Figure 3: HSV Threshold (Pre Dilation) and No False Positives Ceiling

Motion Tracking I – Motion Gradient Equation

Our next desired improvement was to have no need for a priori information on the object to track. We wanted to track general motion within a frame. With a stationary camera, the easiest way to determine where motion has occurred is through subtraction of two grayscale frames in time with a threshold at some low value. Any pixels above this value will be declared motion and any pixels below will be declared noise. This is a simple approximation, but we wanted to track all objects, so we implemented a motion gradient equation algorithm.

Figure 4: Motion Gradient images



This algorithm attempted to gauge both the magnitude and direction of motion and feed it into our camera. We were able to get to this stage successfully. However, we had an oversight when we started moving our camera. In the ideal case, the motion of the center (ie, the object being tracked) would be zero and the motion of the outside of the frame would be opposite the motion of the camera. However, when first tracking an object, the camera would need to seek the largest area of motion instead of the area with the least motion. I think this is unclear... perhaps you should point out that this is the total opposite of the case where the camera is stationary and the object is moving (ie when it enters the scene) It would be possible to set up an experiment where the camera had different behaviors based on applied commands. First, it would seek the largest motion while still. Then it would somehow mix into assuming the same motion it had plus a correction from the center of the image if it were not quite zero. We did not have time to get this extension to the algorithm working.

Stereo Vision Complexity

One area of interest we intended to study but could not get working was stereo vision object tracking. We ran into two issues with stereo vision. The first is that the cameras and lenses need to be exactly the same. In order to easily do calibration and disparity image generation, the only pair of lenses we were able to find for our cameras were telephoto lenses that focused on objects roughly 10 to 15 feet from the camera. This lead to very hard estimation and calibration of stereo depth as a slight angle error between the two cameras leads to very large offsets that could overpower the calibration using the distance between the two cameras. The second issue was the need to have synchronized images.

We discovered that our cameras do sync up if placed on the same firewire hub, but we could not find a working firewire hub or a computer with synchronized firewire ports. We did not have the time and microcontrollers available to do manual synchronization of the cameras, though this is possible with the cameras we used in this project. This lead to the timing differences between the two frames becoming nondeterministic and thus invalidating the stereo processing. While these problems could easily be overcome with other equipment or some additional timing, since stereo vision was an "extra" for our project, we decided to focus on the monocular camera tracking aspects.

Conclusion

While tracking of a specific target can be done easily, tracking of an unknown object becomes much more difficult. Using simple image signal processing techniques on images to track objects yields good results for a priori objects. Our application of motion gradient equation failed in this instance due to a complicated array of assumptions that the motion gradient equation makes. The motion of the camera added to the motion of the object was too much to cover in this assignment but is probably possible. Any further research on this topic should have a more thorough literature review.

Some possible directions for future testing could include discovering the target's approximate shape or color through motion gradient by finding the location of movement and then applying this information to the simpler algorithms. It could also include using multiple cameras, including the combination of static and moving cameras to use the combination of algorithms to yield better results.