Learning in Vision Processing for CUAUV
Kenneth Bongort (kjb79)

## ABSTRACT

One of the problems that the Cornell University Autonomous Underwater Vehicle Team (CUAUV) faces is dealing with variable lighting conditions for vision tasks. At Cornell, we test in the Teagle pool, so lighting is very consistent. However, the competition takes place in an outdoor pool, so the lighting conditions can change drastically, even in the middle of an autonomous run. Depending on whether we have cloud cover, the bottom of the pool varies from consistently dim to very bright, with dappling from the surface of the water. We attempt to deal with these changes by thresholding the input images, and adjusting the threshold parameters according to the environment. With effective use of learning techniques, it should be possible to create an algorithm that does not depend on tuning parameters. This paper develops a method for categorizing the shapes detected by the downward camera in any lighting condition that we encounter.

## INTRODUCTION

CUAUV competes in an annual competition sponsored by the Association for Unmanned Vehicle Systems International (AUVSI). The competition consists of an underwater obstacle course that our vehicle must navigate entirely on its own. One task in the annual AUVSI AUV competition is to drop markers into two of four bins, based on the different shapes painted in the bins. The shapes are painted in red against the black bottom of the bin, and the bins are surrounded by white frames.

CUAUV's old shape classification algorithm is based on finding the contour around the red area, and calculating the Hu moments for the shape defined by the contour. Hu moments are a special set of image moments that are invariant under rotation, scaling, and translation. We calculate the distributions of the Hu moments for a set of known images, and record the parameters for each distribution. When the vehicle sees an unknown shape, it attempts to categorize it by comparing its Hu moments to the distributions we calculated. The formula used for categorization was essentially made-up, with no basis in probability.

The disadvantage to this approach is that the contour-finding algorithm relies on thresholding the hue, saturation, and value of each image. We need to hand-tune the threshold boundaries for different operating environments and different lighting conditions, or the contours will be badly miscalculated.

This paper presents a new algorithm that works well in any environment without manual adjustment. My approach uses a different pre-processing method that uses no pre-determined thresholding values. It classifies shapes the "right" way, using Bayesian inference on the first two Hu moments. The algorithm uses training system to collect

initial Hu moment statistics, allows for correcting errors once the data has mostly stabilized.

## THE ROBOT

This work was done on image sets recorded by Nova, CUAUV's 2009 competition vehicle. Since this work involves vision processing, and not control, it is independent of the robot it is deployed on.

## THE ALGORITHM

### Bin-Finding

The input to this algorithm is an image recorded by the downward-facing camera of the vehicle. These images are recorded by a high-quality underwater video camera. Figure 1 gives two examples of the images we are working with.

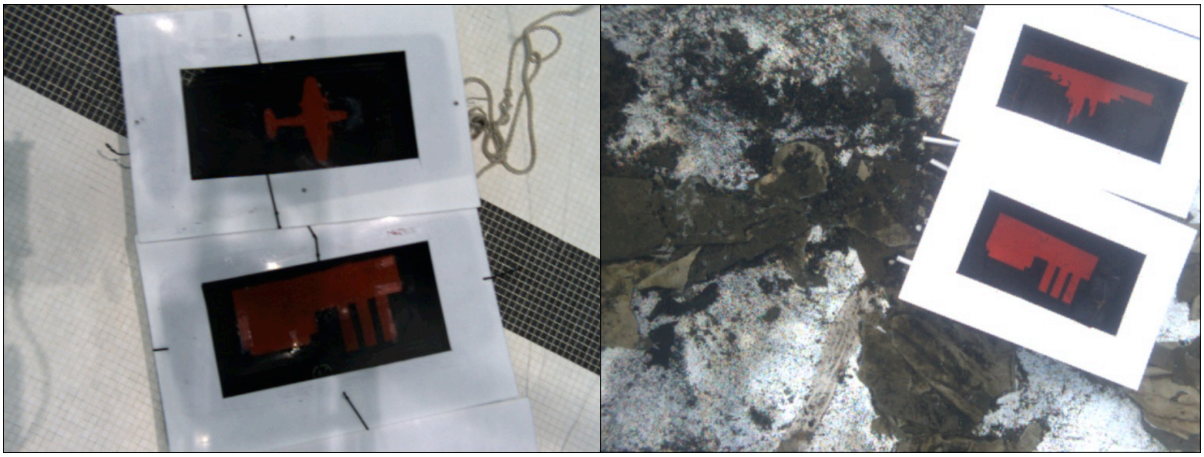

Figure 1. Left: Bins in Teagle Hall pool. Right: Bins in TRANSDEC, San Diego.

Before I can classify the shape, I must find it in the image. I developed a bin-finding algorithm that finds the center-point, size, and angle off the horizontal for the inside (the black region) of each bin the the image. This algorithm uses bi-lateral filtering to reduce surrounding noise and insignificant edges, then applies Canny edge detection to get the outline of the bins. After this step, I have an image that includes the outlines of the inside and outside of each bin, plus noise from surrounding objects (Figure 2).

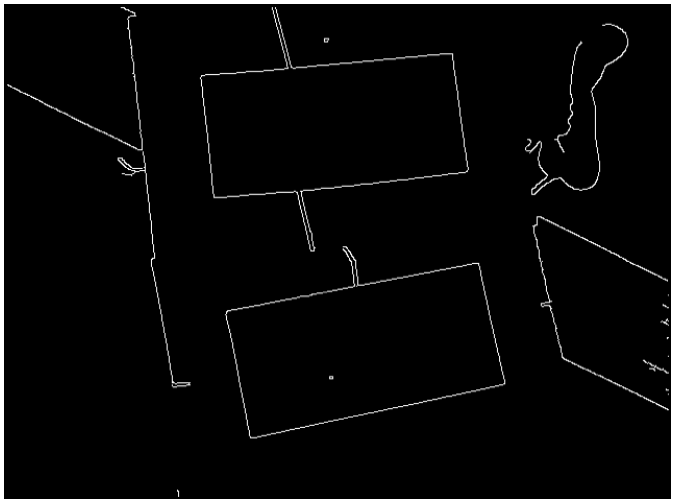

Figure 2. Canny edge detection applied to the left hand image of Figure 1.

I use probabilistic Hough line detection to find straight lines in the image. If two lines have nearby endpoints, and form approximately a 90-degree angle, I consider the shared endpoint to be a corner. Once I have found all the corners in an image, I look for pairs of corners that have the correct aspect ratio to form a rectangle shaped like the inside of a bin. The corners on the outsides of the bins also have the right aspect ratio, so when more than one rectangle intersect, I keep the smallest. Figure 3 Once I have found the inside of a bin, I crop it out, correct the angle, and send it to the shape detection algorithm (Figure 4).

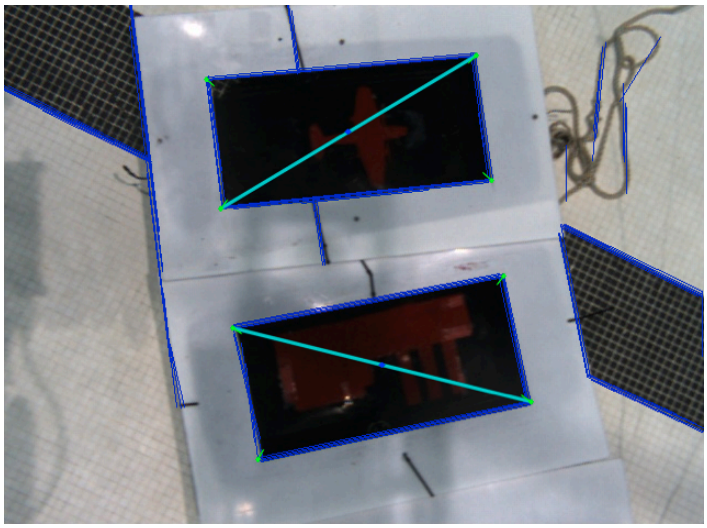

Figure 3. Image annotated with lines detected by Hough method in blue, corners in green, and diagonals of bin interiors in teal.

Figure 4. The insides of the bins, cropped out and corrected.

It is important to make the shape of interest as distinctive as possible before calculating the Hu moments of the image. Otherwise there is high overlap in the distributions of the Hu moments for different shapes. To separate the shape from the background, I first separate the image's saturation and value channels (Figure 5).

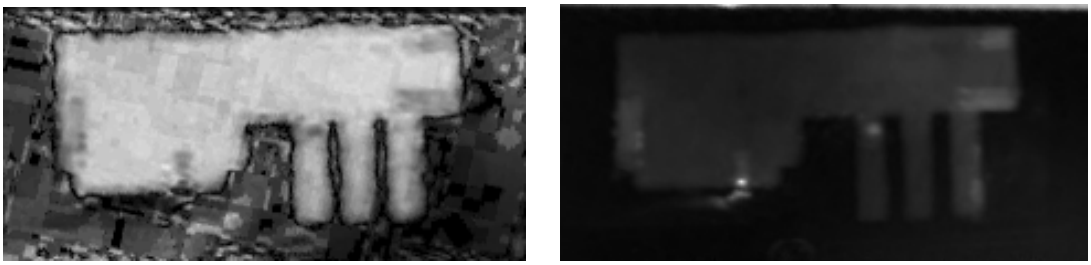

Figure 5. Saturation (left) and value (right) for the factory.

The shape is very bright in the saturation channel, but the dark black surround causes the channel to be very noisy. Multiplying it by the value channel reduces that noise, but retains the bright shape. I also multiply by a constant 5, to better separate the shape from the background. Figure 6 shows the resulting image.

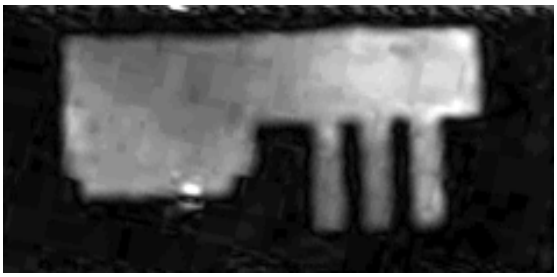

Figure 6. The product of the saturation of the value, multiplied by 5.

With the shape fairly well-separated, I threshold the image with Otsu's method. This is a method that computes the ideal thresholding value for separating foreground and background. Thresholding produces a black-and-white picture ideal for Hu moment calculation (Figure 7).

Figure 7. Image after thresholding with Otsu's method.

The algorithm employs Bayesian inference to calculate the probability that the shape in the bin matches each of the four shapes, given the first two Hu moments for the image. The algorithm outputs the shape whose probability is highest.

## TRAINING

The shape classification algorithm requires the distributions of the Hu moments for each shape. Initially, it has no information about these distributions, so it runs in a training mode. When it detects a shape, it asks the trainer to identify it. The trainer must enter a number, 1 through 4, to tell the algorithm what the shape is. Once the algorithm has some data, it will start guessing what the shape is. The trainer can either confirm the guess, or correct a mistake (Figure 8).

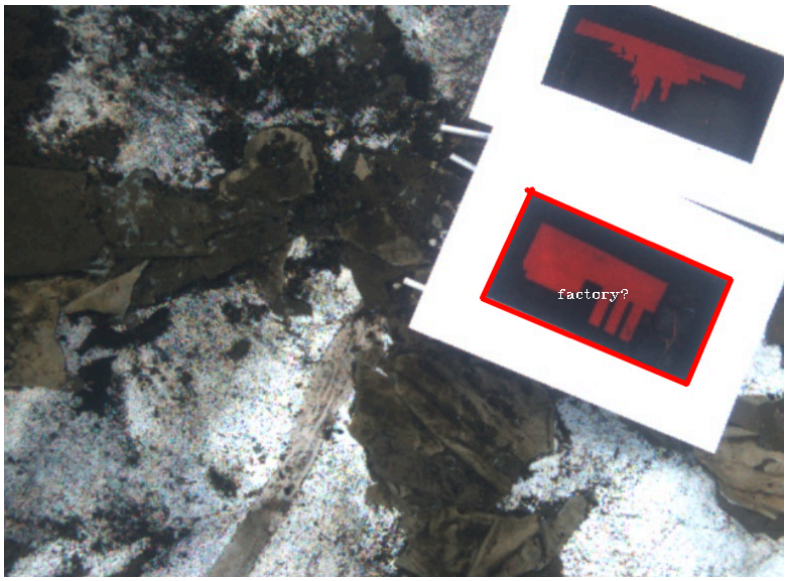


Figure 8. The training system asks for a confirmation of its guess.

After one or two training sessions, the distributions are well-enough stabilized that you can let the algorithm run. As it goes, it will save the image of each shape to a directory for that shape. Once it's finished you can look through the directory for misplaced images, move them to the right directory, and then compute the distributions with the corrected assignment.

## EVALUATION

I tested my algorithm on six sets of images, from three pool tests in Teagle Hall and three runs in TRANSDEC. The images from pool tests tend to have dim, even lighting, and consistent color. The images from TRANSDEC have lighting that varies from dim to very bright, and sometimes exhibit sun-dappling. The murkier water also affects the image color. The number of shape samples in each of these sets are given in Table 1.

| Image Set | Shape Counts | | | | |
|---|---|---|---|---|---|
| | Ship | Factory | Plane | Tank | Total |
| pooltest01 | 34 | 52 | 39 | 23 | 148 |
| pooltest02 | 130 | 136 | 23 | 216 | 505 |
| pooltest03 | 262 | 82 | 103 | 149 | 596 |
| transdec01 | 187 | 197 | 133 | 164 | 681 |
| transdec02 | 263 | 138 | 89 | 240 | 730 |
| transdec03 | 655 | 155 | 393 | 515 | 1718 |
| | **1531** | **760** | **780** | **1307** | **4378** |

Table 1. Shape

To test the algorithm, I collected statistics based on some selection of image sets, and then tried classifying the shapes in another image set. For example, I would use the learned data from the pooltests 1 and 2 to classify the shapes from pooltest 3. Table 2 shows the result of these tests.

| Image Set | Accuracy (Percent) | | |
|---|---|---|---|
| | Other Pooltests | All Transdec | All Others |
| pooltest01 | 100 | 100 | 100 |
| pooltest02 | 96.63 | 96.44 | 96.63 |
| pooltest03 | 89.26 | 89.77 | 90.77 |
| All Pooltests | - | 93.67 | |
| | Other Transdec | All Pooltests | All Others |
| transdec01 | 89.87 | 95.01 | 92.22 |
| transdec02 | 95.34 | 98.49 | 98.22 |
| transdec03 | 90.92 | 95.01 | 93.25 |
| All Transdec | - | 95.21 | |

Table 2. Classification accuracy. The algorithm was trained on the sample sets in the columns, and used to classify the sample sets in the rows.

This testing shows that the algorithm has an overall accuracy over 90%, but its performance depends on the sample set it is classifying. It works particularly well on

pooltest01, but poorly on pooltest03 and transdec01. Since the distributions of the first two Hu moments have some overlap (Figure 9), some amount of error is inevitable.

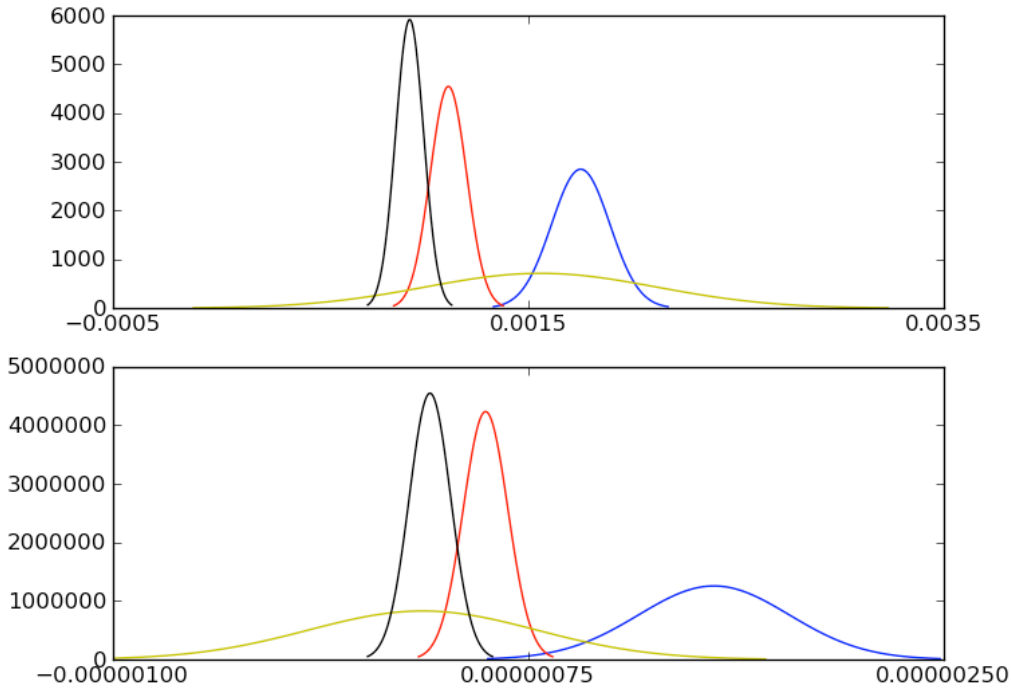


Figure 9. Distributions of the first two Hu moments, over all samples. Tanks are black, factories are red, ships are blue, and planes are yellow.

The small difference in accuracy with different training sets indicates that the distribution data stabilizes fairly quickly, and is similar for images from Teagle Hall and from TRANSDEC.

## CONCLUSION

The evaluation results show that this algorithm performs quite well. It successfully identifies shapes over 90% of the time, without any need for hand-tuning for different operating conditions. A training system brings Hu moment statistics to stable values quickly, so that the estimates are as good as possible given the variance in the samples.

It is certainly possible to achieve better performance by enhancing this method. Since shapes are isolated and rotated before classification, it is not important to use statistics that are rotationally invariant. We could use additional techniques, like Legendre moments or histogram correlation to generate more, and perhaps better predictors of image shape. With better preprocessing, the Hu moment distributions could be improved so that different shapes are more distinct.