Contact Detection and Analysis System (CDAS)

Maritime Autonomy and Perception

Helena Rhee

Consortium for Undergraduate Research Experience (CURE) at Los Angeles City College (LACC)

Work supported by National Science Foundation (NSF) Grant #1460538

Using image chips cropped from the full image as specified in the annotation files, we are able to produce datasets with the purpose of reclassifying certain vessels to retrain the system by relabeling the labels associated with the chips.

I. Introduction

JPL Maritime Robotics has been tasked with the system development that allows Unmanned Surface Vehicles (USV) to navigate safely among other vessels and identify potential hazards on water while complying with the International Regulations for Preventing Collision at Sea (COLREGS). This capability will eventually allow for mission operations that require automated target recognition (ATR) and intelligence, surveillance, and reconnaissance (ISR).

The perception aspect of the system is called the Contact Detection and Analysis System (CDAS). This is where visible and IR spectra images received by unmanned surface vehicles (USV) are processed for 360-degree maritime situational awareness. CDAS captures the images obtained from cameras mounted on vessels such as the Hammerhead system shown below. The cameras are based on a single bar and faced forward to capture wide range images.

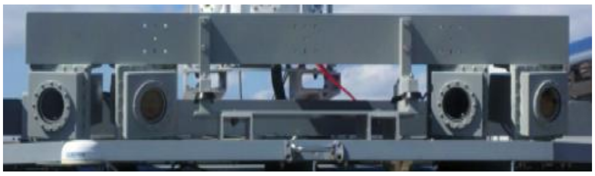


Fig 1: Cameras on Hammerhead System

After the images have been processed by CDAS, they are stored to a disk. Then the disk is sent to JPL for further processing.

II. Annotations

Using algorithms based on machine learning, the USV can learn how to react in situations and identify objects on water depending on the datasets previously annotated. An annotated dataset is where any objects, if not all, in the image are processed by bounding the object with a rectangle box and labeling it. The following demonstrates a full image before it's processed:

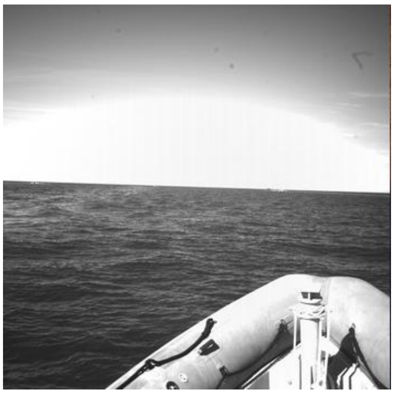


Fig 2: Full Image

The full image, figure 2 was taken by the camera on board the vessel. There are three objects about the horizon in the image that were annotated. Each of the annotated objects are resized and is shown with its respective labels below:

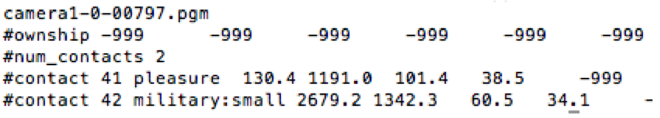
Fig 3: pleasure

Fig 4: military:small

Fig 5: military:small

Each annotated object is considered to be a single image chip. And a collection of annotated objects is referred as a dataset of image chips. The term dataset signifies either a pool of image chips and/or a collection annotated objects. The datasets and the machine learning algorithms together permit the system to identify and classify objects as a certain type of vessel or an anchored object without using a hierarchical ordering. If a hierarchy classification was used instead, the system would have difficulty identifying the object or even possibly fail to when they possess multiple characteristics of different vessels.

Each full image, when processed, results in an annotation file. A single annotation file, a text file, would contain meta data to each image chip and its Cartesian coordinates of the bounded box with respect to the full image. An example of an annotation file is shown below:



```
camera1-0-00797.pgm
#ownship -999      -999      -999      -999      -999      -999
#num_contacts 2
#contact 41 pleasure  130.4 1191.0  101.4   38.5      -999
#contact 42 military:small 2679.2 1342.3   60.5   34.1      -
```

Fig 6: Annotation File

The first line of every file displays the name of the full image, camera1-0-00797.pgm, as shown in figure 6. In this annotation file, there are two annotated objects labeled pleasure and military:small respectively. The first two numbers following the labels are the Cartesian coordinates of the top-left corner of the bounding box. The subsequent numbers represent the width and the height of the bounding box. These values determine the position and location of the bounding box relative to the full image.

III. Retraining

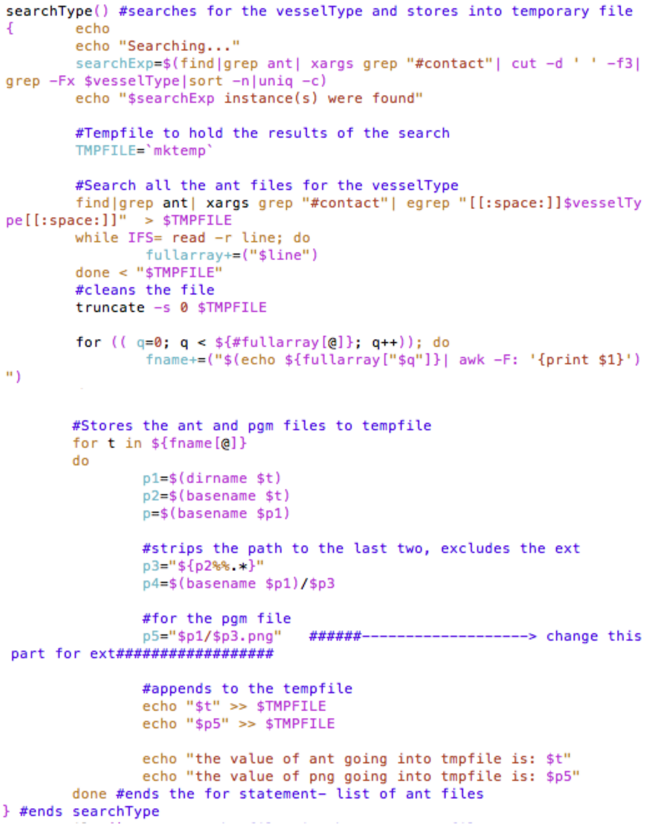
The task at hand is to retrain the system to identify a particular type of vessel called fountain amongst other cigarette boats and rigid-hulled inflatable boats (RHIBs). Cigarette boats are small and fast vessels with two seats to carry passengers in lieu of a cabin; and a fountain is a "notional identity of a particular cigarette boat with a white paint job (D. Levine, personal communication, July 18, 2016)." In a nutshell, a fountain is a cigarette boat with a white paint job. A RHIB, on the other hand, is a vessel with an inflatable hull and usually used by military for patrols and transports.

As it stands, the system is identifying fountain, cigarette boats, and RHIBs as pleasure, small, and military:small vessel types. Hence, these vessels should be labeled differently since their build and roles on sea varies and should not be grouped to the same type.

IV. Methods

To retrain the system, it requires datasets containing only its respective vessel type image chips to avoid contamination which leads to misclassification of vessels. A command line utility tool, written in bash, was used to search through all the directories on the server to find instances of the specified vessel type in annotation files and produce the image chips accordingly.

The bash script can accept up to six parameters but requires a minimum of three which are the full images' navigation path, output navigation path, and the vessel type. The full images' navigation path routes the script to the full images location in the server while the output navigation path routes it to where the image chips should be placed in. The vessel type determines which labels' bounding box values gets pulled from the annotation files. The script validates data entry, routes to the full images' location, searches for the specific vessel type in the annotation files, and processes the image files using four functions. A snippet of the bash script is shown in figure 7.



```
searchType() #searches for the vesselType and stores into temporary file
{        echo
         echo "Searching..."
         searchExp=$(find|grep ant| xargs grep "#contact"| cut -d ' ' -f3|
grep -Fx $vesselType|sort -n|uniq -c)
         echo "$searchExp instance(s) were found"

         #Tempfile to hold the results of the search
         TMPFILE=`mktemp`

         #Search all the ant files for the vesselType
         find|grep ant| xargs grep "#contact"| egrep "[[:space:]]$vesselTy
pe[[:space:]]"  > $TMPFILE
         while IFS= read -r line; do
                 fullarray+=("$line")
         done < "$TMPFILE"
         #cleans the file
         truncate -s 0 $TMPFILE

         for (( q=0; q < ${#fullarray[@]}; q++)); do
                 fname+=("$(echo ${fullarray["$q"]}| awk -F: '{print $1}')
")


        #Stores the ant and pgm files to tempfile
        for t in ${fname[@]}
        do
                p1=$(dirname $t)
                p2=$(basename $t)
                p=$(basename $p1)

                #strips the path to the last two, excludes the ext
                p3="${p2%%.*}"
                p4=$(basename $p1)/$p3

                #for the pgm file
                p5="$p1/$p3.png"   ######------------------> change this
 part for ext##################

                #appends to the tempfile
                echo "$t" >> $TMPFILE
                echo "$p5" >> $TMPFILE

                echo "the value of ant going into tmpfile is: $t"
                echo "the value of png going into tmpfile is: $p5"
        done #ends the for statement- list of ant files
} #ends searchType
```

Fig 7: searchType function in bash

A searchType function, one of four functions written in the bash script, searches for any instances of the vessel type in all the annotation files within a directory and its subdirectories and store its filename in a temporary file.

An image is usually layered on a canvas which resides in the window when displayed on the screen. The additional parameters received by the script manipulates the values that determine the size of the canvas and the actual image. The script can produce a padding between the image chip and the canvas or extend the image chip by additional pixels.

The feature to extend the image presented an issue when the bounding box values started at the corners of the full image. In this specific case, there were no room to extend further outward. This would eventually cause an error when using the Imagemagick cropping tool in the bash script. Under Dr. Dan Levine's advisement, it was decided that another function would be required in the utility tool which would compare values of space between the edge of the original image and the bounding box against the wanted extended amount. Then it would take the minimum of these values and extend the image.

Another unforeseen challenge was presented when applying the script to different directories. Initially, it was assumed that all annotation and image files were in the same format as shown in figure 6. However, the script failed to execute when applied to certain directories. In truth, the annotation and image files were not in the same format. It was discovered that few of the annotation files did not have a new line at the end of the file while most of the files did. And the image files were in png format instead of pgm. This caused an issue when reading the filename and finding its corresponding image file along with opening the annotation file in the bash script.

Once we were able to complete the additional features to the script and produce the image chips, we proceeded to reclassifying the labels associated with each chip manually. Relabeling the image chips presented its own challenges. Generally, "labeling forces us to clearly think about naming and categorization issues, how to represent occluded objects, how to deal with parts of the image that are unrecognizable, when context becomes important for recognition, what is the effect of our prior knowledge and expertise about a particular visual domain, and how do we deal with clearly visible objects whose name or function is unknown to us." [2] This was particular true when using the Quick Annotator tool developed by JPL's Visualization Software Engineer, Viet Nguyen, to manually scrutinize and relabel each image chip. Without the extensive background and knowledge on the different types of vessels, it was difficult to differentiate them at certain angles. In these situations, by viewing images that came before and after assisted in the identification of the vessel. Additionally, there were times where we had to look at the context of the full image to make a determination of what the image chip contained. The following images demonstrate these scenarios:

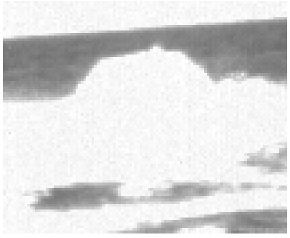


Fig 8: Fountain 1

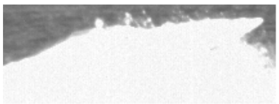


Fig 9: Fountain 2

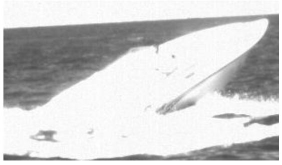


Fig 10: Fountain 3

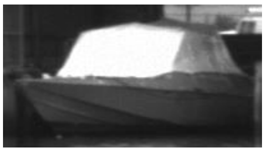


Fig 11: Image chip

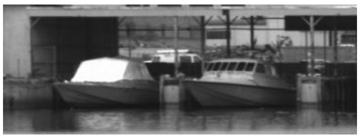


Fig 12: Context image

If one looked at figure 8 alone, it has an outline of a vessel with a driver but there's no clear indication that it's a fountain. However, viewing the image chips that came before and after it, we can determine that it is indeed a fountain. As for figure 11, it is unclear as to whether the vessel under the tarp is a fountain or not. However, by comparing the image chip to the vessel stationed next to it in the context image, figure 12, we can determine by comparison that the bounded image is in fact not a fountain.

V. Conclusion

After producing about 15,000 to 20,000 images chips and annotating them for reclassification, it became apparent how difficult the task can be to identify objects accurately, especially at far distances without human intervention. Furthermore, this process must be completed manually since we are creating a dataset that would retrain the system. There's no method to automate this process. With certain image chips, the objects became identifiable only by either scrutinizing other images that came before and after it or looking at them contextually. This indicates a system must be able to learn how to process images as opposed to systematically categorize them according to how they are defined. Also provides the support why a hierarchical ordering or categorization would fail since vessels with multiple characteristics can fall under different categories.

VI. References

[1] H. Agharzarian, A. Howard, T. Huntsberger, and D. Trotz, "Stereo Vision Based Navigation for Autonomous Surface Vessels," Jet Propulsion Laboratory, California Institute of Technology, September 2015.

[2] A. Barriuso, and A. Torralba, "Notes on Image Annotation," Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, pp. 1-15, October 2012.

VII. Acknowledgments

I would like to thank Professor Paul McCudden for his guidance and attention in the Cure program, Dr. Michael Wolf for the opportunity at JPL, Dr. Dan Levine for the conceptual guidance in computer vision and its related fields, Viet Nguyen for providing technical materials and responding to questions, and lastly Dr. Jayesh Bhakta for recognizing the greatness in me.

This work is supported by National Science Foundation (NSF) Grant #1460538 to Los Angeles City College, CURE program.