Autonomous Surface Vehicle For Surveillance-like Activities

Gabriel González, Pedro Fonseca, Juan Carlos Aguilera, Ricardo Plata, Alejandro González, Pedro Sánchez, Alejandra Monsiváis, Mauricio Méndez, Kevin Kwan, Michelle Wong, Ricardo Salgado, and Leonardo Garrido

***Abstract*—Roboboat is an international competition where teams design autonomous boats to overcome challenges with the least possible error. The challenges evaluate the boat's ability to navigate autonomously, recognize objects, avoid obstacles and find paths to a destination. This paper documents the work done by the VANT Tec team during the development of the boat for roboboat 2017.**

***Index Terms*—roboboat, autonomous vehicle, convolutional neural networks, computer vision, boat design, pathplanning.**

I. Introduction

ROBOBOAT is an international competition that promotes the research and development of autonomous unmanned surface vehicles through a series of challenges. Although the teams have the liberty of deciding how to overcome the challenges, they have to comply with some restrictions, such as the size and weight of the boat being under certain measurements. The trials to overcome are change every year. In 2016, the vehicle had to deploy a submarine for an underwater trial. This year, an *Unmanned Aerial Vehicle* (UAV) has to take off from the boat. The main tests of this year are:

- Autonomous Navigation
- Speed Challenge
- Automated Docking
- Pathfinding
- Dynamic Target Identification

The team is made up of students from the careers of digital systems and robotics (ISD), mechatronics (IMT), electronics (ITE), industrial design (LDI) and computer science (ITC). In addition, there are two students from the master in science in intelligent systems.

II. Design Strategy

The principal objective of the team is completing the competition's course. Once that the boat is able to complete the course, the remaining time will be used to minimize errors for increasing reliability to finish without errors. Since it is the first time VANT Tec participates in the Roboboat competion, the team focused in building the boat and acquiring all the sensors needed as soon as possible. The team was divided into subteams so that each one would address a certain problem. A subteam leader was elected for each of them for easier coordination. The team was divided in the following subteams:

- Design (α)
- Computer Vision (γ)
- Electronics (β)
- Mechanics (Ω)
- Drone (Δ)
- Programming (ϵ)

Each subteam would investigate about their field and there would be a weekly meeting with all the subteam leaders, and all available members, for deciding what to do about each topic. The journals from roboboat 2016 were very helpful for having an idea of what could be achieved with the components that each team used. If there were cheaper alternatives, the cheaper were chosen for not going over the budget.

III. Vehicle Design

The design of the vehicle can be divided into the physical design of the boat, the hardware used for sensing and modifying the state of the environment and the software that enables the boat to be autonomous.

A. Boat design

The *design team* was in charge of designing the boat. They decided to go for a catamaran design, which has two parallel hulls, for more stability. The boat also had to have a helipad for the drone to take off and land. The first designs had totally planar deck for having a bigger for the parrot to land after the interoperability challenge. However, the design changed to a deck with a small platform for the RPLIDAR and a bigger one for the helipad. The RPLIDAR is able to map in 360 degrees, but only maps in two dimensions. There was the need of a mobile platform that could change the inclination of the RPLIDAR since the sensor would be in a higher position than most obstacles. A pole was also placed in the design so that the camera could have a wider view of the surroundings (figure 1).

While the design of the deck was not ready, the design team made the mold of the catamaran. The mold was made by covering a wood skeleton (figure 2) with plaster (figure 3).

The boat was to be made of plastic using thermoforming. However, the mold was made hollow and the suction of the thermoforming machine would destroy the mold before making an usable hull. Since the mold could not be used for thermoforming, it was decided that the boat had to be made of fiberglass.

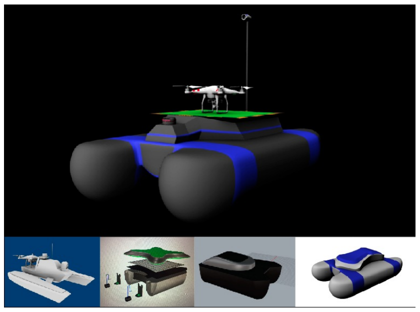


Fig. 1. Evolution of the boat's design

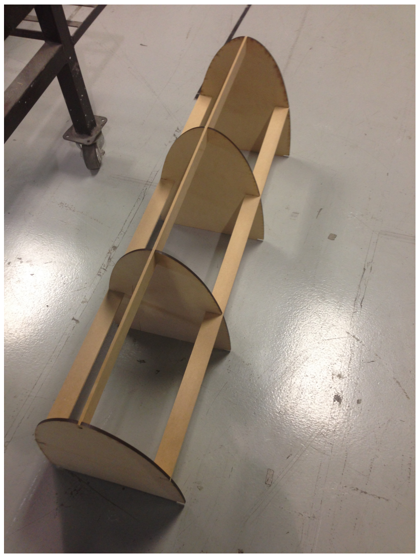


Fig. 2. Wood skeleton of the mold.

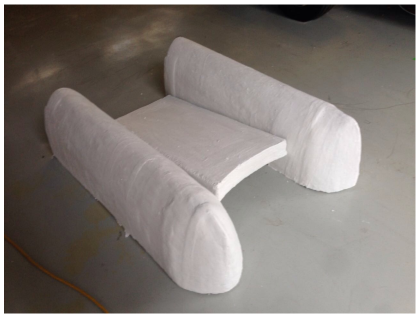


Fig. 3. Finished mold.

B. Hardware

The first step was to select the thrusters and the batteries that could feed the thrusters the enough voltage and current to move the boat. They were the most fundamental parts of the boat because the maneuverability of the boat had to be tested for the possibility of making a new hull if the performance was not as expected.

The components used in the boat are:

- Logitech c525 webcam
- RPLIDAR 360 Laser Scanner
- Inertial Navigation System VN-200 SMD from Vector-Nav Technologies (IMU)
- 2 T200 Thrusters from Blue Robotics
- 1 Lithium-ion Battery (14.8V, 18Ah)
- Taranis FrSky X9D
- X8R. FrSky 8/16ch telemetry receiver
- Arduino
- Laptop with NVIDIA GPU
- 1 Hydrophone TC4013-1
- Benthos ALP-365 Pinger

The first sensors to be bought were the webcam, the *Inertial Measurement Unit* (IMU) and the *hydrophone*. This elements were chosen first because the challenges of the competition require object detection (webcam), localization (IMU) and underwater acoustic detection (hydrphone). After some consideration, the RPLIDAR was acquired for improving the pathplanning since the burden of the vision system for obstacle detection was too heavy.

The laptop acts as the brain of the system and delegates the control of the thrusters to the arduino. The arduino receives orders from the laptop and from the Taranis FrSky X9D transmitter. The transmitter has higher priority than the laptop so that if there is any problem with the decision-making of the laptop, the boat can be controlled back to safety.

Two BlueRobotics T200 thrusters were used for propulsion. Each of them needs a *BlueRobotics Basic Electronic Speed Controller* (ESC), which controls the speed of the thrusters. The ESCs are connected to BlueRobotics Lithium-ion batteries on one side, and to an Arduino through the other side.

The remote control is Taranis FrSky X9D with a X8R receiver. The receiver uses channel 2 and 4 to communicate with the Arduino. The algorithm for power control states a stop range to avoid noise. Outside that range, the equation for turning left is:

$$y = V_2 - (V_2 - 1500)(\frac{(1500 - V_4}{512}) \tag{1}$$

and for turning right:

$$y = V_2 - (1500 - V_2)(\frac{(V_4 - 1500}{512}) \tag{2}$$

where V_2 and V_4 are the input channels and the output gives the signal for the opposite side thruster.

C. Software

1) Computer vision: One of the fundamental requirements for the competition is object recognition. The boat has to be able to recognize buoys and numbers to complete the course with the highest possible marks. Two algorithms were used for object recognition. One for its simplicity and the other for its robustness.

The first algorithm uses *Hu moments* [1] to detect objects by comparing the values of contours in an image with the saved values of the objects that want to be detected. Hu moments are useful because are invariant to scale and rotation. This simple approach works, but can be affected by the illumination in the environment. In addition, if the object is seen from another point of view, the object would not be recognized. First, *moments* are calculated for every object:

$$m_{ij} = \sum_{x,y} (image(x,y) \cdot x^i \cdot y^j) \tag{3}$$

and used for getting the *centroid*:

$$\bar{x} = \frac{m_{10}}{m_{00}} \qquad \bar{y} = \frac{m_{01}}{m_{00}} \tag{4}$$

Second, using the previous values for getting the *central moments*:

$$\mu_{ij} = \sum_{x,y}(image(x,y) \cdot (x-\bar{x})^i \cdot (y-\bar{y})^j) \tag{5}$$

Third, the *normalized central moments* are calculated:

$$\eta_{ij} = \frac{\mu_{ij}}{m_{00}^{(i+j)/(2+1)}} \tag{6}$$

Finally, seven Hu moments are obtained:

$$Hu_0 = \eta_{20} + \eta_{02} \tag{7}$$

$$Hu_1 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \tag{8}$$

$$Hu_2 = (\eta_{30} - \eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \tag{9}$$

$$Hu_3 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \tag{10}$$

$$\begin{aligned} Hu_4 = {} & (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ & +(3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \end{aligned} \tag{11}$$

$$\begin{aligned} Hu_5 = {} & (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\ & +4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03)} \end{aligned} \tag{12}$$

$$\begin{aligned} Hu_6 = {} & (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})] \\ & -(\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \end{aligned} \tag{13}$$

The second algorithm is a *Convolutional Neural Network* (CNN). This method is better for object recognition since it is not affected as much by light as method one. Besides, it is able to recognize objects even if seen from different angles. The downside is that the network requires a large quantity of photos from many different angles. In addition, the training can take up several days because of the quantity of information to process. The CNN that is being used is a Faster R-CNN [2]. The difference between a common CNN and the Faster R-CNN is that the latter gives a bounding box that surrounds detected objects unlike the former which only says if an object is present or not (figure 4). There are other CNNs that give bounding boxes, such as *Single Shot Multi Box* (SSD) [3] and *You Only Look Once* [4] (YOLO). Faster R-CNN was chosen because Yan Henon's implementation [5] provides a parser for easily using a custom training set. Training is done by feeding the program a text file with all the training instances. Each line of the file must have the location of the photo, two points (opposite corners of a bounding box) and the class of the object.

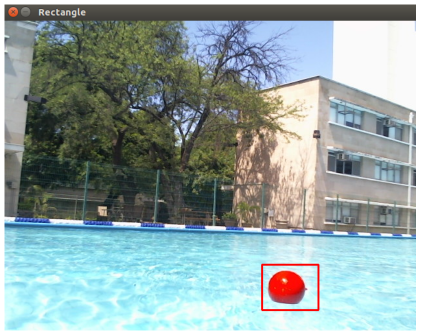


Fig. 4. Example of a bounding box on a buoy

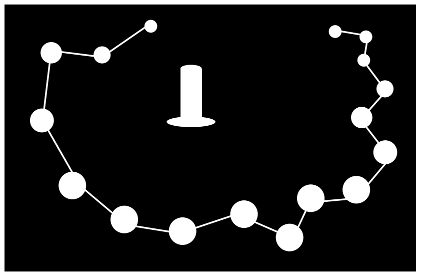


Fig. 5. Resulting image of closing the spaces where the boat cannot pass through.

D. Pathfinding

The boat will use the VN-200 to keep track of its position. The VN-200 has incorporated:

- 3-axis accelerometer
- 3-axis gyroscope
- 3-axis magnetometer
- barometric pressure
- GPS

The position is used to determine if the boat is on track to its destination and if it has reached the destination. The RPLIDAR and the logitech c525 webcam are used to generate a 2D map. The 2D map generated by the RPLIDAR will be updated with any obstacle detected by the webcam in case that the RPLIDAR misses to detect an obstacle. The map will be used for potential field pathplanning in most of the track. For the *find the path* challenge, the boat algorithm is as follows:

1) Take a photo of the the area.
2) Filter the image.
3) Use watershed algorithm for image segmentation.
4) Considering objects as clusters, calculate real euclidean distance between objects using single-link for distance between two objects. The real distance can be obtained using the relationship between the buoy's real size and its pixel size.
5) Use an ϵ equal to the boat's diameter *d*, which will have a similar use as in DBScan (Density-based clustering). Unite clusters/objects by a line when $\epsilon >=$ the distance between them (figure 5).
6) Apply A* to the resulting image to find a path.

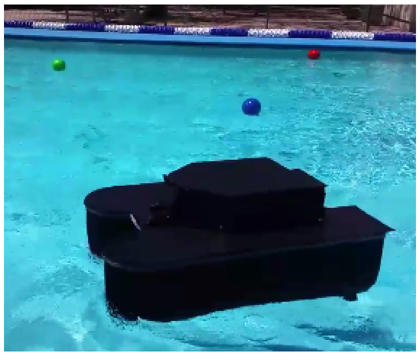
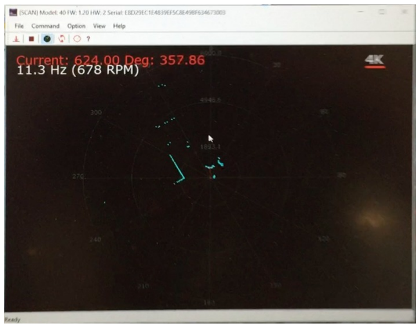


Fig. 6. Floatability test.



```
Current: 624.00 Deg: 357.86
11.3 Hz (678 RPM)
```

Fig. 7. 2D map generated with the RPLIDAR.

IV. Experimental Results

A. Boat Design

After testing the floatability of the boat, the T200 thrusters were added for checking the speed and maneuverability of the boat using an arduino and the Taranis FrSky X9D transmitter (figure 6). The maximum speed of the boat was around 2 m/s.

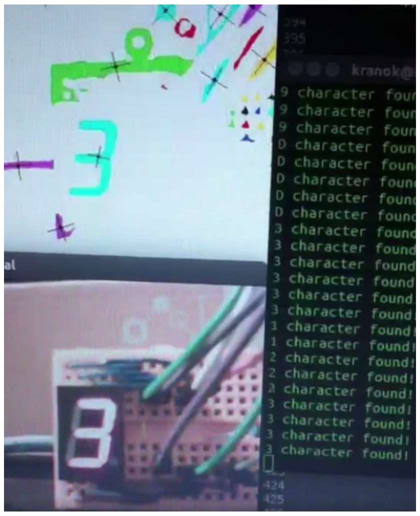
B. Sensors

The data from the RPLIDAR A2 was successfully retrieved and displayed (figure 7). The data can be easily used to avoid obstacles in a 6m range.

A digital circuit with a 7-segment display was used to test the recognition of numbers. The display showed the numbers from 0 to 9 and was able to detect them correctly (figure 8).

The faster R-CNN was trained with PASCAL VOC dataset [6] to test the implementation of the network (figure 9). It could not be trained with the buoys because the dataset is still being labeled. Currently, there are 100 labeled instances. Many instances can be made with the same picture if the number of buoys is more than one. Around 850 photos were taken to generate the dataset. Data augmentation can be used by rotating and flipping the images.

A circuit for simulating a wave emitter at different frequencies was implemented. This circuit will be used to simulate the pinger detection in the automated docking challenge (figure 10). The integrated circuit NE555 is used as a multivibrator to generate a signal that oscillates between the frequency range stipulated by the rules of the competition.



```
9 character four
9 character four
9 character foun
D character foun
D character foun
D character foun
D character foun
D character founc
3 character found
3 character found
3 character found
3 character found
3 character found
3 character found
1 character found
1 character found!
2 character found!
2 character found!
3 character found!
3 character found!
3 character found!
3 character found!
3 character found!
```

Fig. 8. Testing of number detection using Hu moments.

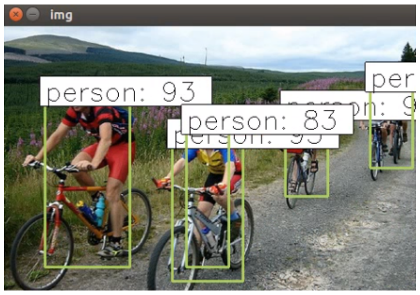


Fig. 9. Testing keras-frcnn after training with PASCAL POV.

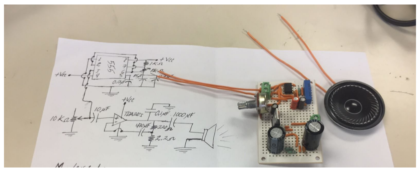


Fig. 10. Circuit to simulate the pinger.

V. Conclusion

In conclusion, it is possible to overcome all challenges by tackling them with several approaches in order to find the proper one. It might be difficult to have a perfect performance in the competition without previous experience in roboboat. Nonetheless, the multidisciplinary team has the tools and the knowledge to tackle any problem. The results achieved so far point out that it is possible to have a good performance.

References

[1] M.-K. Hu, “Visual pattern recognition by moment invariants,” *IRE Transactions on Information Theory*, vol. 8, no. 2, pp. 179–187, February 1962.

[2] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, June 2017.

[3] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, *SSD: Single Shot MultiBox Detector*. Cham: Springer International Publishing, 2016, pp. 21–37. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-46448-0_2

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 779–788.

[5] Y. Henon, “keras-frcnn,” https://github.com/yhenon/keras-frcnn, 2015.

[6] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, Jun. 2010.