RoboBoat 2022: VantTec Technical Design Report VTec S-III

Rogelio Salais, Ian de la Garza, Sebastian Martinez, Abiel Fernandez, Mauricio Sánchez,
Max Pacheco, Alexa Arreola, Anel Alvarado, Alberto Tamez, Fernando Arizpe,
Isaac Gutiérrez, Javier Prieto, Kenia Sanchez, Rodrigo Garza, Sofia Martinez,
Jesus Arteaga, Andres Sanchez, Nadia Garcia, Andrea Rangel, Adrian Leon, Aleksandra Stupiec,
David Martínez, Alejandro Gonzalez, and Leonardo Garrido
Monterrey, Mexico
vanttec@servicios.tec.mx

***Abstract*—In this report, the competition strategy, the design creativity over new implementations, improvements on the VTEC S-III's systems, and experimental results for the RoboBoat 2022 competition are presented. The strategy relied on validating the base algorithms of the *Road to the Show*, *Snack Run*, and *Avoid the Crowds* through physical testing on campus facilities. The electronics system was improved, with the creation and implementation of three PCBs, corresponding to motor control, power distribution, and LED status control. New challenges also arised from the return to normal activities after almost two years of online activities due to the COVID-19 pandemic.**

***Index Terms*—Unmanned surface vehicle, robotics, autonomy, GNC system, computer vision, artificial intelligence**

I. Competition Strategy

It was not until late 2021 that our university, Tecnológico de Monterrey, started opening its doors for the return of students. Our return to a physical setting came with several challenges. Even though facilities were open again, most of our team members couldn't go to the campus because they lived in other states or countries, stunting physical progress. It wasn't after February of 2022 when our classes began from being hybrid to fully on-site, that our workspaces were finally available and team members initiated physical work.

During the second half of 2021, we started development of a quadcopter to overcome complications we had with the non-friendly API of the drone (DJI Phantom 4 Pro), utilizing a custom-build drone with a Pixhawk. This project became independent after the *Object Delivery Challenge* was removed from the RoboBoat 2022 edition's challenges.

After almost two years of working online, many of our senior members graduated and several new members were added to the project, resulting in an overall low-familiarity with the VTEC S-III platform. This was quite the challenge as the algorithms made for the challenges during the online competition haven't been tested on the VTEC S-III platform. An overall focus on gaining experience and preparing these new team members was prioritized, lowering the speed of current developments, but with the expectation to achieve greater performance on next editions.

A. Course Approach

Our current course approach is a modification of our previous approach to account for the new challenges and the disruptive lack of starting coordinates of each competition task in the 2022 edition. Our increased reliability in solving the *Mandatory Navigation challenge* of previous editions translates to confidence in the 2022's edition's *Avoid the Crowd*, *Snack Run* and *Return to Home challenges*, making them our top priority. Our next prioritized challenge is the *Find a Seat in the Show* , where the docking location is now vision-based. A classic-vision approach is sufficient to distinguish the required seat. An approach and mechanism for the *Skeeball Game* and *Water Blast* challenges was planned but development didn't reach the required level of confidence, making them our lowest-priority challenges. A registry system was implemented to make our VTEC S-III platform capable of registering the global positions of obstacles within the course. A

challenge detection algorithm takes this registry and produces waypoints that directly correspond to each of the challenges, solving the lack of coordinates issue and also to initiate our state-machine based solutions when approaching each challenge.

Fig. 1. VTec S-III USV.

II. Design Creativity

A. Software Architecture

The software architecture for the VTec S-III relies on the Robot Operating System (ROS) as the backbone of software development. The main difference compared to last year's approach is an upgrade from ROS Kinetic to ROS Melodic, as Ubuntu 16.04 is no longer supported.

A new package, DARKNET_ROS_ZED was included in the architecture (Fig. 2). This is an open source package that provides an implementation of YOLOv3 and tiny-YOLO on GPU and CPU on ROS, and also provides depth estimation for the object detections based on a ZED Camera.

The USV_PERCEPTION package was improved as a new node that registers objects that persist on memory was included. This new node considers both LiDAR point cloud detection from last year's approach and the ZED distance estimation from the DARKNET_ROS_ZED package.

B. Simulation Environment

As complete on-campus activities resumed on February of the current year, a simulation environment was still required to further develop our algorithms.

The simulation environment (Fig. 3) employed is the same one presented for last year's competition,

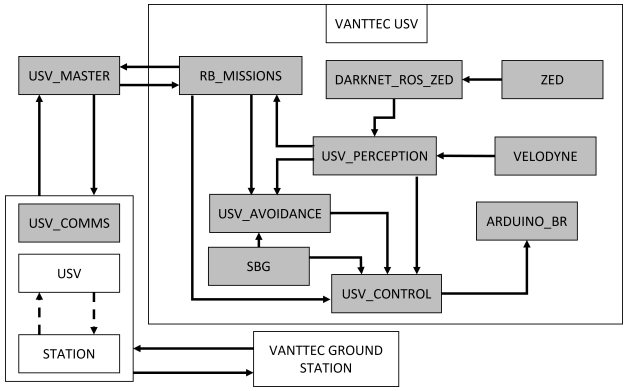

Fig. 2. USV software architecture

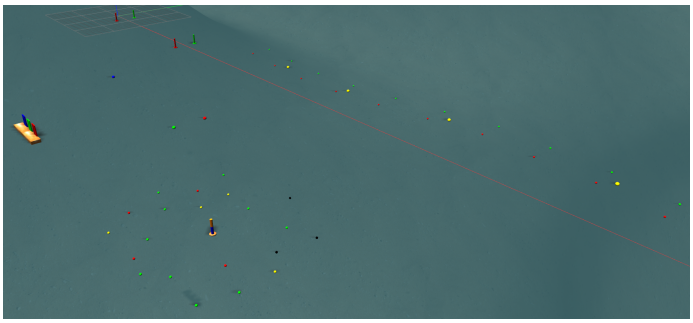

Fig. 3. Gazebo simulation environment.

relying on Gazebo, a dynamic model of the USV to simulate the vehicle pose and velocity; a 3D model of the boat; a LiDAR sensor [4]; a stereo camera [5]; a node to interface the USV repository with Gazebo; and a basic lake scenario [3] and custom props.

C. Guidance and Control

A cascaded scheme is used for the guidance and control for the USV. The guidance law is based on a Line-Of-Sight (LOS) algorithm developed for RoboBoat 2020 [2], and the control law is an Adaptive Sliding Mode Controller strategy presented also in RoboBoat 2020. It was proven in [6] that the controller is robust against uncertainties as shown in the physical experimental results.

D. Perception

A problem present since RoboBoat 2019 was the YOLO-based object detection speed, as in the best case scenario, the detections never surpassed the frame rate of five fps. This problem made the object detection system based on the ZED

and YOLOv3 unreliable, as the images took to much time to process with YOLO. Until this year we focused on detecting objects with the LiDAR pointclouds, but we decided to improve the YOLO-based detections with the ZED images to obtain full information on the environment obstacles. We used the package DARKNET_ROS_ZED [1] that provides an implementation on ROS of YOLOv3 with GPU. The use of the aforementioned package increased the frame rate up to 20 frames, which is a major improvement. On top of the increased detection speed, the package already provided a distance estimation for each detected class, based on the depth map of the ZED camera. With the given depth estimation, trigonometry was used to estimate the 3D position of each detected object.

1) Object registry: An obstacle registry algorithm was implemented, which takes as inputs the detections from our YOLOv3-ZED and our LiDAR detection systems. This registry system stores all detected obstacles' global position, id and type, being either 'buoy', 'marker' or 'dock'. The obstacle registry system ultimately acts as a filter for verifying obstacles, further enhancing our perception capabilities.

A status is attached to each object in the obstacle registry. By default, the status is 'registered'. When detection is consistent, meaning an object is detected in the same range of XY coordinates a number of times specified in the code, the status is changed to 'persistent'. This change in status filters out detections caused by noise in a real-time feed of the LiDAR dependent system. Finally, when a registered object shares coordinates with a YOLOv3-ZED based detection, it changes the status to 'verified' and logs the color attribute on the object.

Depending if both systems are implemented on our VTEC S-III platform, either registered objects with 'persistent' or 'verified' status are used on our waypoint-creating or collision-avoidance algorithms. This implementation solves an issue presented during real-time testing, which is non-consistent detection of obstacles due to blind spots in sensor positioning.

2) Challenge detection: A challenge detection algorithm was developed as direct solution to the lack of provided coordinates for challenges, using the obstacle registry system's stored information as input. Buoys are paired using approximate distances give by handbook guidelines, creating gates. This detection of gates permit us to map the *Avoid the Crowd* and *Snack Run* challenges, creating the corresponding waypoints. Other non buoy-based challenges, such as *Water Blast*, *Skeeball Challenge* and *Find a Seat in the Show* challenges are first distinguished by the prominent dock shape.

E. Collision Avoidance

The MPC-based collision avoidance method developed during the first half of 2021 is still utilized for this competition [10]. When the collision avoidance is used, the NMPC replaces the LOS guidance law, sending the desired surge speed and yaw references to the ASMC controller. As of now the NMPC is intended to be used in *Avoid the Crowd* and *Return to Home* challenges, but is known to work as demonstrated in last year report.

F. Electronics and Embedded Systems

As an evolution from last year's design, we designed and manufactured two new PCBs. One of these, is the motor controller PCB as seen in Fig. 4. This PCB receives motor commands from the Jetson TX2 via CAN bus. We use a STM32 [7] microcontroller, which processes sensor data, and generates PWM signals for the thruster motor ESCs and mechanism servos. By utilizing FreeRTOS [8], a real-time operating system and our CAN bus protocol, we can ensure that we have real-time robust control over our motors and mechanisms. The second PCB we designed is the power distribution PCB (Fig. 5), that uses relays to kill-switch the devices with a digital signal or through the emergency stop button. It also distributes power and provides overcurrent protection for each output channel. Finally, for this season we developed a new status indicator PCB (Fig. 6), which controls our LED indicator, this is used to show the operational status of the autonomous vehicle by turning on one of the three LED lights located on the outside of the boat.

III. Experimental Results

A capture of the object registry algorithm and the functioning of both the YOLOv3-ZED detection system and LiDAR detection system can be seen in

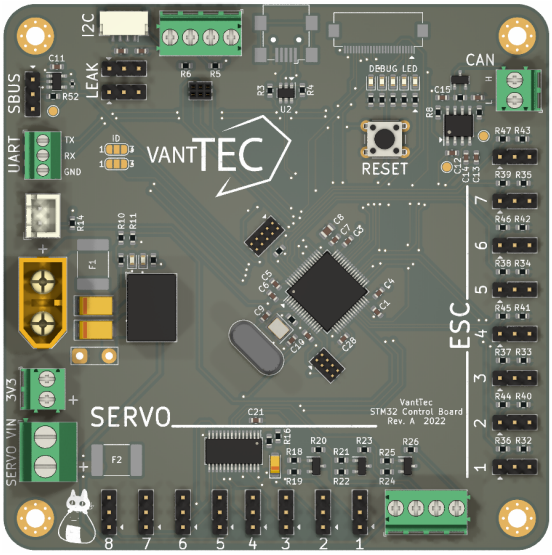

Fig. 4. Motor controller PCB.

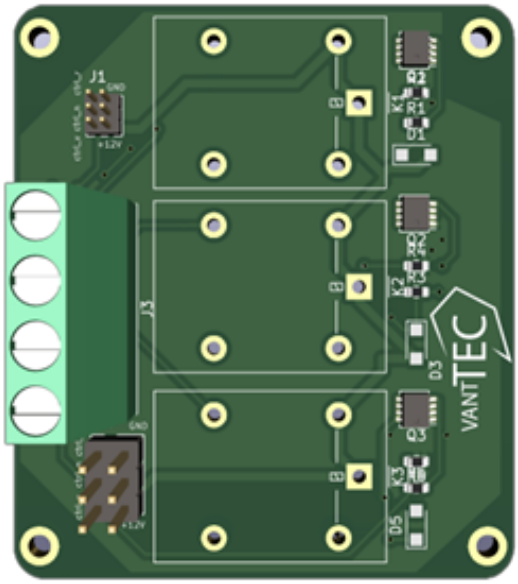

Fig. 6. Status indicator PCB.

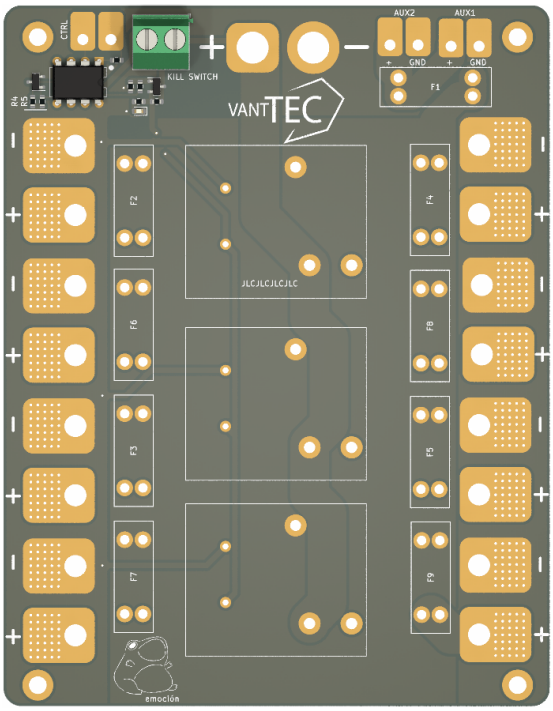

Fig. 5. Power Distribution PCB.

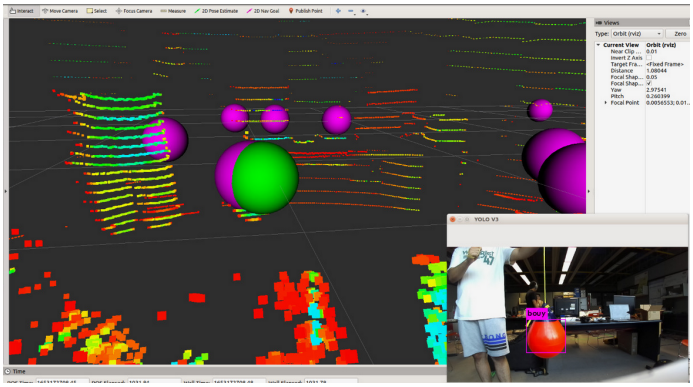

Fig. 7. LiDAR (purple) and ZED (green) 3D object detection with YOLO (bottom right corner) bounding box

Fig. 7. In the capture, the purple spheres correspond to current detections on the LiDAR detection system, and green spheres correspond to detections on the YOLOv3-ZED detection system. Several fake detections on the LiDAR detection systems are due to the test being performed in a cluttered, indoor environment. The obstacle registry systems logs the position of each detection, and when the detection of both systems correspond to the same coordinates, as seen with the buoy in Fig 7, it changes the state to 'verified'.

An additional capture done in our Gazebo-simulated environment is included in Fig. 8, where the detections of the LiDAR detection system is represented with purple spheres and the green spheres represent objects in the object registry system with 'persistent' status. These coordinates are logged in a NED reference frame, with the origin being an initial reference point when the boat is spawned. When visualizing these objects, a transformation matrix transforms these NED coordinates to a 'body' reference frame, in which the center of mass of the

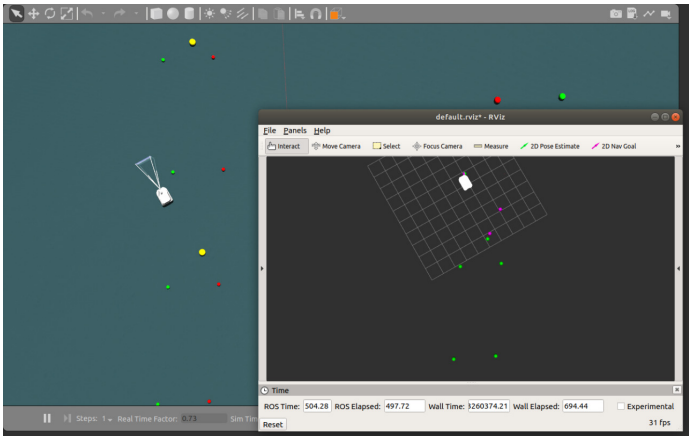

Fig. 8. Object registry in Gazebo

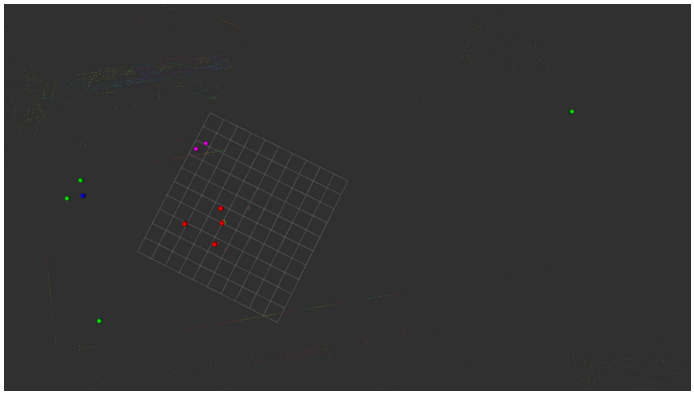

Fig. 9. Pool filtering using rosbags

VTEC S-III acts as the origin.

A noteworthy implementation is the addition of a coordinate filtering script. This script was made due to an issue with our main physical testing environment being an outdoor pool. Objects outside the area the pool, with similar dimensions as markers, buoys or docks, were incorrectly detected and registered. The coordinate filtering script directly calculates the limits of the pool using the initial reference point used by the NED reference frame, and filters out objects outside the desired range. Although its implementation was tested using *rosbags*, which is logged sensor data from previous tests, true physical testing is still required. It is expected that this coordinate filtering script can also be applied in the physical setting of the competition, eliminating the risk of the VTEC S-III platform invading other course areas when detecting challenges.

In Fig. 9 the blue sphere represents the initial reference point used by the NED reference frame and the green spheres correspond to the calculated limits of the pool. Additional markers were added to represent the filtered and non-filtered detections, with the purple spheres representing detections outside the delimited area and the red spheres representing detections within the area.

Additional physical experiments with the VTEC S-III platform were not realized due to hardware failures involving the Inertial Navigation System, unable to be resolved before the deliverable deadline.

IV. Conclusions

The presented RoboBoat 2022 strategy shows our advancements. This edition's progress can be mostly observed in advances on the electronics and embedded system, and the addition of new implementations on the perception system, further refining the data used for path-making and obstacle avoidance.

After 2 years of working from home, getting familiarized with the VTEC S-III platform and its development became a challenge. By testing our perception algorithms on the VTEC S-III platform physically, we realized that relying solely on the Velodyne VLP-16 LiDAR wasn't enough due to noise, so we found necessary to combine the YOLOv3-ZED detection with the LiDAR detection. This improvement will help us to be able to find gates, and being able to identify the challenges, though most of our test has been made in rosbags, more tests need to be made.

Further improvements in our electronics were made with a new PCB with an embedded STM32, this changes were made to avoid the incident of 2019 RoboBoat competition in which at the finals, an Arduino nano used for the motor speed controllers burned out.

The areas of opportunity to tackle are the perdurance of knowledge inside the team, a better way to teach new members about the VTEC S-III platform and a better way to manage time. As for next year we hope to create a read the docs (https://readthedocs.org/) for our USV code documentation so that new members find it easier to

adapt and start developing for the platform, we also hope to test more our algorithms physically and to start implementing the mechanisms for the skeeball and waterblast challenges in our boat.

ACKNOWLEDGMENTS

This work was supported by: Techmake, SBG Systems, Google, IFM efector, RoboNation, Velodyne LiDAR, NVIDIA, Akky, ZF Group, Güntner, and Siemens. Finally, VantTec appreciates the support from the university, Tecnologico de Monterrey and to our advisors.

REFERENCES

[1] Bjelonic, M., YOLO ROS: Real-Time Object Detection for ROS, Github repository, 2018. Available: https://github.com/leggedrobotics/darknet_ros

[2] A. Gonzalez, et al., "VantTec ts and Systems (IROS), pp. 2577-2582, 7-12 Oct. 2012.

[3] Musa, M.M.M., Scherer, S.A., Voss, M., et al., "UUV Simulator: A Gazebo-based package for underwater intervention and multi-robot simulation", IEEE OCEANS, 2016.

[4] Lovro, M., Velodyne Simulator, Github repository, 2021. Available: https://github.com/lmark1/velodyne_simulator

[5] Guilherme C., Librealsense, GitHub repository, 2016. Available: https://github.com/guiccbr/librealsense

[6] A. Gonzalez-Garcia and H. Castañeda, "Guidance and Control Based on Adaptive Sliding Mode Strategy for a USV Subject to Uncertainties," in IEEE Journal of Oceanic Engineering, doi: 10.1109/JOE.2021.3059210. https://github.com/morriswmz/doa-tools

[7] "STM32F405XX Datasheet " www.stm.com [Online]. Available: https://www.st.com/resource/en/datasheet/stm32f405rg.pdf. [Accessed: 05/23/2021].

[8] "What is An RTOS? " www.freertos.org [Online]. Available: https://www.freertos.org/about-RTOS.html [Accessed: 05/23/2021].

[9] Robert, B., "CAN Specification, Version 2.0" [Online], 1991. Available: http://esd.cs.ucr.edu/webres/can20.pdf. [Accessed: 23-May-2021].

[10] A. Gonzalez-Garcia and H. Castañeda, "A Real-Time NMPC Guidance Law and Robust Control for an Autonomous Surface Vehicle" in IFAC Conference on Control Application in Marine Systems Robotics, and Vehicles CAMS 2021, doi: 10.1016/j.ifacol.2021.10.101.

APPENDIX A: COMPONENT SPECIFICATIONS

See Table I.

TABLE I
COMPONENT SPECIFICATIONS

Component	**Vendor**	**Model/Type**	**Specs**	**Cost**
ASV Hull	VantTec	VTec S-III	Fiberglass	NN
Propulsion	Blue Robotics	T200	http://docs.bluerobotics.com/thrusters/t200/	NN
Power System	Blue Robotics	Lithium-Ion Battery	http://docs.bluerobotics.com/batteries/	NN
Motor Controller	Blue Robotics	Basic ESC R2	https://www.bluerobotics.com/store/retired/besc30-r2/	NN
SBC	NVIDIA	Jetson TX2	https://developer.nvidia.com/embedded/buy/jetson-tx2	NN
MCU	STMicroelectronics	STM32F405RG	https://www.st.com/en/microcontrollers-microprocessors/stm32f405rg	NN
Teleoperation	FrSky	Taranis X9D Plus	https://www.frsky-rc.com/product/taranis-x9d-plus-2/	NN
Teleoperation	FrSky	X8R	https://www.frsky-rc.com/product/x8r/	NN
IMU	SBG Systems	SBG Ellipse2-D	https://www.sbg-systems.com/products/ellipse-series	NN
Camera	Stereolabs	ZED Camera	https://www.stereolabs.com/zed/	NN
Hydrophone	Telodyne	TC4013	http://www.teledynemarine.com/reson-tc4013	NN
Hydrophone	Aquarian	H1C	https://www.aquarianaudio.com/h1c-hydrophone.html	NN
CAN transceiver	Waveshare	SN65HVD230	https://www.waveshare.com/sn65hvd230-can-board	NN
RF Modules	Digi	XTend	https://www.digi.com/products/networking/gateways/xtend-900mhz-rf-modems	NN
LiDAR	Velodyne Lidar	VLP-16	https://velodynelidar.com/vlp-16.html	NN

Algorithms	Internal development. Adaptive sliding mode based control, line-of-sight based guidance, model predictive control based collision avoidance
Vision	Point Cloud Library, OpenCV
Localization and Mapping	Internal Development. Based on reference frames and 3D computer vision.
Team Size	23 members
Expertise Ratio	1:1
Testing time: simulation	9 months
Testing time: in water	0 months
Inter-vehicle communication	NN
Programming	ROS, Python 2.7, C++ and MATLAB/Simulink