The University of Maryland, College Park

**Abstract**

TortugaV is an autonomous underwater vehicle (AUV) designed and constructed by students from the University of Maryland, College Park as part of a student government sponsored club. It is designed primarily to compete at AUVSI and ONR's RoboSub competition, however, real world applications for AUVs such as sea floor mapping, mine detection, and underwater scientific research continuously motivate further development of similar vehicles. TortugaV is capable of serving as a development and testing platform for underwater actuation, underwater machine vision, sensor fusion algorithms, and six degrees of freedom control algorithms. TortugaV retains many of the previous design concepts of TortugaIV while improving significantly on the dynamics, sensing capabilities, and reliability. Advances in the software system provide expanded modularity, more sophisticated estimation algorithms, a greatly improved AI and invaluable testing and tuning software.

**Authors:** Christopher Carlsen, Nicholas Limparis, Donald Gregorich, Eliot Rudnick-Cohen, Stephen Christian, Katherine McBryan, Johnny Xian Yi Mao, Justin Kanga, Joshua Pugh, Rajath Shetty

Faculty Advisors: Dr. David L. Akin, Dr. Nuno Martins, Ph.D.

# Introduction

Tortuga V is an Autonomous Underwater Vehicle (AUV) designed and built by Robotics@Maryland (R@M) an undergraduate student organization based at the University of Maryland, College Park. The original Tortuga was created in 2006 to compete in the Tenth Annual International AUV Competition hosted by AUVSI and ONR. The purpose of the competition is to develop AUV technology by challenging students to construct submersibles capable of completing underwater tasks that simulate possible missions. Since it's inception, the competition has gotten more difficult; introducing manipulation tasks, sonar, and more complex vision tasks. Just as the competition has improved, so has Tortuga. In the current configuration, Tortuga V is now capable of navigating relative to active sonar pingers, complex visual assessment, controlled underwater motion, and the manipulation of objects.

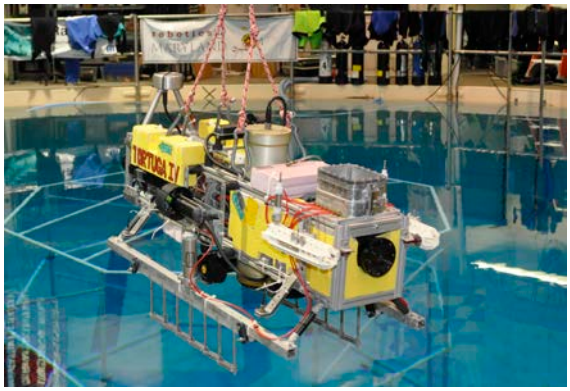

Figure 1: Tortuga V

## Team Overview

Building an autonomous underwater vehicle is a very complex task and requires students from all disciplines to work together. Sub-teams allow the very large complex task of making an autonomous vehicle to be broken down into simpler and more

manageable components. Theses sub-teams, based on disciplines, are: mechanical, electrical, AI, vision, and controls. A core leadership, made up of the more experienced and dedicated students, lead the sub-teams and discuss system issues. A systems engineer and program manager help to keep the project on schedule and manage the financing.

One of they key parts of each sub-team is to train new recruits. At the start of the new year, recruiting begins and a new batch of interested students are introduced to Tortuga. While many students are very interested, most lack the required technical knowledge that can only come from working on an autonomous underwater vehicle. Each sub-team has a series of introductory projects which teach the students about the discipline and the specifics of Tortuga. These introductory projects are designed to allow new students to go from the basics to working on Tortuga at their own pace.

Figure 2: R@M at Transdec 2013

## TortugaV Overview

Tortuga has one main electronics hull and a number of smaller pressure hulls. The main hull is made of acrylic which allows the team to see inside the vehicle. This is important as it allows all the indicators on all the electronics boards to be read and used for testing. The acrylic hull allows the vehicle to operate at a maximum depth of 30 feet. At this depth, Tortuga can maintain position and orientation for over two hours without needing to recharge. This allows the vehicle to complete an entire day at competition without needing to charge the batteries. The batteries are located in the main acrylic hull along with the computer, a 2009 mac mini, a stack of custom boards and sensors. The other pressure hulls include: a pneumatics housing to control all the mechanisms on board, a magnetometer boom to move the sensors away from the computer and thrusters, and a Doppler Velocity Log (DVL), and two camera housings.

| Sensor | Quantity |
|---|---|
| 9DOF Memsense IMU | 2 |
| Depth Sensor | 1 |
| Digital Compass | 1 |
| Doppler Velocity Log (DVL) | 1 |
| Firewire Camera | 2 |

Table 1: Navigational Sensors

# 1 Mechanical System

The mechanical design of Tortugas frame values modularity and ease of access. Constructed out of 80/20 Inc. T-slot aluminum beams the relatively simple design allows for a variety of actuators and hulls to be added and removed as required for mission completion. In addition the accessibility of the design allows easy access to all of the major components.

## 1.1 Main Hull

The main hull houses the processor and power supply for the robot. It consists of an acrylic tube sealed with 2 aluminum end caps. The acrylic tubing allows for visual inspection of the o-ring seals and debug displays on the electronics. One of the aluminum end caps acts as the mounting surface for the various SubConn wet-mate electrical connectors. The other end cap has been machined to give the maximum exposed surface area for heat dissipation. The heat dissipating end cap is mounted on 80/20 frame where it is latched onto the robot. These latches allow us to quickly remove the hull from the robot if necessary. Finally, to allow activation of the robot when sealed there are a number of magnetically activated switches

positioned close to the inside surface of the hull for easy activation and deactivation of the robot.

## 1.2 Camera Hulls

For completion of visual tasks the robot uses two cameras; one downward facing and one forward facing. These 2 cameras are in individual custom hulls designed by the team which allow them to be adjusted independently.

Figure 3: Camera Housing

## 1.3 Pneumatics

For the actuation of manipulators Tortuga utilizes a CO2 pneumatic system. The core of this system is the pneumatics hull. This hull can support up to twelve dual action pistons. The system is very robust and reliable. In addition, the CO2 cartridges are easily accessible allowing the gas supply to be quickly refreshed.

## 1.4 Downward Manipulator

The bottom manipulator consists of piston actuated tines. When the robot descends onto open frame structures the tines allow the structures crosspieces to passively catch within spring tensioned latches which sit between the tines. When the robot needs to release the structure the piston extends flexing the tines causing them to release the latching mechanisms. The passive grabbing system has high reliability due to its simplicity.


Figure 4: Tortuga with downward grabbers

## 1.5 Bin Droppers

The markers consist of illuminated cylinders with ferromagnetic bases. These are suspended against Delrin stoppers with magnets actuated by pistons. When the robot needs to release a marker a pneumatic piston pulls the magnet away allowing the marker to fall straight down. The actuators are adjacent to the downward facing camera. This allows increased accuracy with the droppers when the camera is centered over the bins.

# 2 Electronics

As in previous years, the electrical system on this update to Tortuga IV is a modular system with a radial control structure chosen for its logical separation of functions, expandability, and ease of servicing. In this design, a central board links plug-in cards, each card with a specific set of functions. Our electronics stack consists of a backplane board which holds six other modules: the power regulator (DC/DC), battery load balancing, power distribution, actuator control, sensor, and sonar processing boards.

## 2.1 Power

Power within Tortuga V is handled primarily by three of the main six boards, namely the DC/DC, Battery Balancing, and Distribution boards, with additional device-specific regulation by two peripheral UVQ boards. Power for the system is drawn from a 5-pack block of 25.9 volt, lithium-ion batteries. In order to supply sufficient

current for the robot, these packs must be connected in parallel, which necessitates battery balancing to keep the draw as even as possible from each pack. The even draw mechanism disconnects batteries which have a lower nominal voltage, preventing the batteries from wasting power, and ensuring a long operational period.

Next, the power is distributed throughout the robot. This task is handled primarily by the dedicated distribution board, which serves the various devices requiring high currents, such as the motors, the Doppler Velocity Log (DVL), and the computer, while controlling which devices receive power. The remaining distribution is handled by the backplane itself, which shares power between all of the plug-in boards. The distribution board gives Tortuga control over which devices are currently receiving power. This manner of power distribution does not require all devices to be active, which allows for quick diagnosis of problems and power conservation when the robot is not in the water. There is also a safety mechanism which can terminate power to any device if the software detects unsafe conditions.

The final step in the power subsystem is to regulate the voltage from the input batteries to useful, common voltages for the lower power devices. This is handled by three separate boards. The first is the regulator board, which handles power to the backplane, hydrophones, inertial measurement units, and the router. The voltages produced by this board are 20 volts, 12 volts, 9 volts, and 5 volts.

The remaining regulation is performed by a pair of UVQ boards, so named for manufactures designation for isolating DC-DC converters. The UVQ boards provide the power needed for the DVL and the computer (2009 Mac Mini). Currently, power into and out of the UVQ boards requires filtering to prevent excessive electrical and magnetic interference (EMI) from interfering with other systems. This is accomplished using an inductor/capacitor pair on the input and output wires.

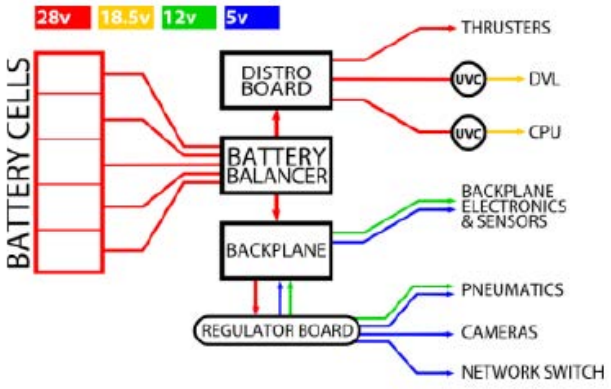


Figure 5: Power Distribution Layout

## 2.2 A Software, Firmware, and Sensor Communication

Communication between the main computer and the various actuators and sensors happens through a serial communications bus to a single integrated circuit (IC). This Microchip dsPIC (the master IC), distributes commands to all of the other microcontrollers, and allows any microcontroller with enough pins to interface with the current electrical system. Any microcontroller capable of a few simple protocols ($I^2C$, rs-232, and SPI) could easily replace the current PIC architecture, but Microchip dsPICs were chosen over Atmel or ARM microcontrollers because the electrical team was already familiar with their use.

Certain high-bandwidth sensors, such as the Doppler velocity log (DVL), inertial measurement units (IMU), and cameras, are interfaced directly to the computer rather than through the master IC interface. The DVL and IMUs are connected directly through USB to serial converters, while the cameras connect via firewire. This allows the computer to more quickly pull data from the sensors, rather than relaying request commands through the IC network.

## 2.3 Actuator Control and Communication

The current actuation system is comprised of six thrusters and various pneumatic pistons. The thrusters are controlled through a simple $I^2C$

interface. Although the $I^2C$ protocol allows for all the thrusters to communicate on the same 2-wire bus, we have isolated each thruster so that a failure or power outage in any individual thruster does not affect our ability to communicate with any other thruster. The pneumatic system is run by a single dsPIC connected through serial lines to the main hull. To actuate the pneumatic system, the small pneumatic relays are opened and closed to control the flow of pressure to and from the piston chambers.

# 3 Control and Estimation

Tortuga V is designed with preference of maneuverability over speed, having control over all six degrees of freedom. In order to control all six degrees of freedom, we have chosen to incorporate six thrusters into the design rather than dynamically vectoring the thrust. This system allows for simple control that can perform complex motions requiring the use of multiple degrees of freedom at the same time. The six SeaBotix SBT153 thrusters are placed near the vehicle's center of mass in order to maximize control effort. These thrusters include integrated motor controllers and encoders which allow precise control over the thruster output.

Designing an effective control system requires the careful consideration of nearly every aspect of the AUV's design. Starting with the careful attention to vehicle dynamics, the selection of accurate sensors and actuators, and the design of communication hardware to allow low-latency transmission of measurements and thruster commands, the foundation for a well-behaved vehicle is laid before any control algorithms are written. The control subsystem software architecture has been designed with the primary goal of creating a modular object-oriented system which can accommodate multiple control algorithms, allowing for quick switching between algorithms without recompilation. An object-oriented design significantly reduces the amount of logic required, converting semantic errors to syntax errors, which are much easier to identify. This makes development and testing of new control algorithms significantly easier and less error prone. This design also separates code into reasonably sized components, drastically improving the readability and making the understanding of logic flow much easier.

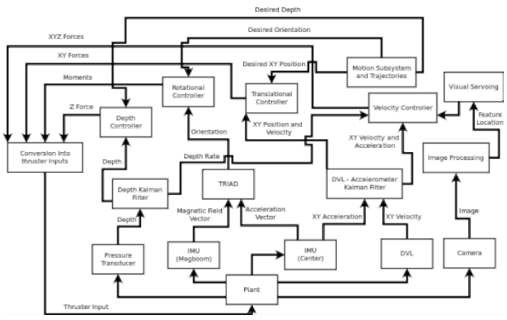


Figure 6: Controls Diagram

## 3.1 Depth Control

Tortuga V primarily holds a fixed depth, so depth control has been implemented as a regulator as opposed to a tracking controller. The vehicle is positively buoyant which can cause a non-linearity. However, due to the buoyancy design of the vehicle, this non-linearity can be treated as a constant disturbance. Thus, any linear controller can be used as long as it is augmented with an integrator for constant disturbance rejection. Several depth controllers have been implemented on the vehicle. PID, LQG, and model based observer controllers [2] are currently available to be selected at run time. Since the vehicle's buoyancy and trim characteristics are often altered for new actuators or adjustment of ballast, the robustness of the control algorithms is as important as controller performance. After comparing the various depth-control algorithms on the vehicle, a PID implementation was chosen.

## 3.2 Rotational Control

The rotational control algorithm is based off [3], a full three axis non-linear PD controller. The controller uses quaternions, making it singularity free (i.e. immune to 'gimbal lock'). This rotational controller is a tracking controller and subsequently can be commanded to rotate the vehicle in any direction by either specifying a desired orientation

or specifying a desired angular rate and integrating the desired orientation. A more advanced rotational control algorithm based on Egeland and Godhavn's adaptive attitude control algorithm [1] has also been implemented. As an adaptive controller, it can 'learn' Tortuga V's inertia, drag, and buoyant moment properties while running to provide a significant improvement to the vehicle's attitude control.

## 3.3 Motions

The ultimate goal of the control system is to allow the execution of complex motions. To achieve this an artificial planning subsystem generates a trajectory for the controller to follow, which allows the controller to reach our desired state more quickly. This also allows the planning subsystem to keep track of when the controller has finished a motion and can then begin the next motion.

## 3.4 Visual Servoing

A visual servoing controller has been implemented for use in aligning with various vision objectives in the competition. The controller sets the Cartesian velocities of the robot based off a proportional control scheme. The velocities are then maintained using a PID controller. This controller allows the vehicle to quickly and accurately position itself around the objectives for the course.

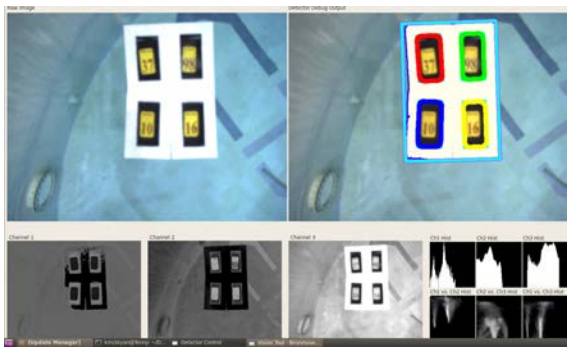

Figure 7: Bins positions are identified by the vision system

# 4 Software

The software architecture on TortugaV is designed around a few core concepts: modularity, extensibility, and configurability. Focusing on these tenets has helped reduce development and testing time while improving software quality. C++ is the core language because of its object-oriented facilities, good performance, and easy integration with other languages. All performance sensitive code is written in C++, aside from the drivers, all of which are written in ANSI C. Code that is not performance-sensitive is written in Python because it is more expressive than C++, leading to shorter development times, and does not require compilation after modifications are made.

## 4.1 Modularity

Modularity comes about naturally from the use of object-oriented design. The software is divided into subsystems, each of which represents an abstraction around a particular task. Each subsystem makes use of abstract classes, each of which serves as an interface to a different physical device or algorithm, reducing both the amount of code that must be written and the number of locations where modifications must take place to integrate new devices or algorithms.

## 4.2 Extensibility

Extensibility is a measure of the amount of work required to implement enhancements. Even a modular system is not necessarily extensible. Extensibility requires foresight of what future enhancements will need to be implemented. Since the software on Tortuga V is under continuous development and new algorithms are frequently being added, careful attention has been paid to make the design general enough to incorporate algorithms that require slightly different input. This extensibility is partly attained through an event-driven architecture. An event-driven architecture reduces the coupling between subsystems as the location in the code where an event is published can be changed without requiring changes to the code of the subscriber.

## 4.3 Configurability

Configurability is an extremely important design aspect. The configuration system allows a selection of different algorithms and the ability to tune parameters without the need for recompilation. Compilation is one of the most processing intensive operations and consequently requires a significant amount of battery power. By reducing or eliminating the need for compilation while testing and tuning, the amount of time that Tortuga can be operating in the water without the need to change or recharge batteries has been greatly extended. YAML was selected as the configuration language because it is human readable and writable, allowing new members to quickly learn the system. It also integrates very well into Python and C++.

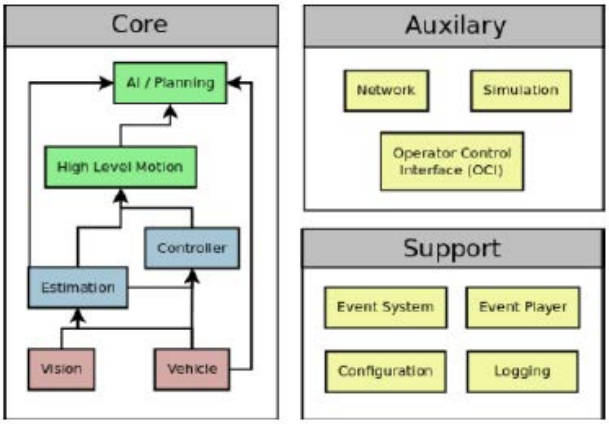


Figure 8: Tortuga V Software System Diagram

## 4.4 Development Practices

In order to facilitate development in a team environment, Github is used to host all of the code, and CMake as a build system. In addition, Trac is used to assign tickets and monitor progress. These tools reduce development times, allow multiple developers to work simultaneously on the same code, and provide a system in which to solve bugs systematically while providing documentation of the solution. The build system generator allows different environments to easily build the code with minimal work required from the developer.

## 4.5 Debugging Tools

There are several debugging and tuning tools which help speed development and analysis of the software. The first is the event player which reads logs of received events during a dive operation and can replay them so developers can more closely examine what occurred during a dive. This system enables developers to more easily debug errors. Another tool is the real-time state visualizer, which enables any networked computer to plot time series of any aspect of Tortuga's sensor suite. This immediate feedback is invaluable for tuning estimation algorithms and controllers.

## 4.6 Operator Control Interface

Tortuga Vs dive operations are supplemented by an Operator Control Interface (OCI) program designed to allow simple interaction with all of the subsystems while displaying telemetry data in a modular fashion. The OCI is composed of multiple panels, each displaying one category of telemetry, such as orientation or artificial intelligence state. The layout of these panels can be reconfigured during runtime to display as much or as little information as the user desires. Individual panels also reconfigure themselves based upon the number and types of devices present on the vehicle. Interaction with the subsystems is accomplished via an integrated Python interpreter.

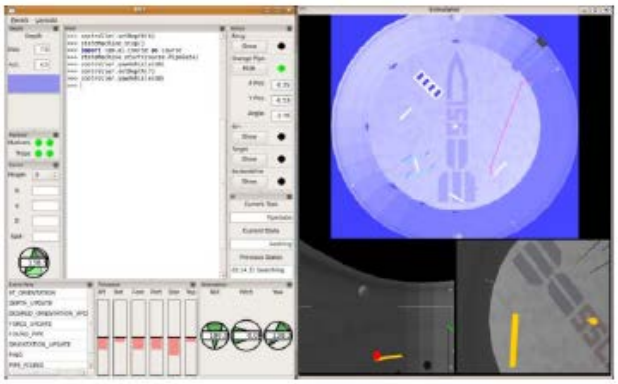

Figure 9: Operator Control Interface

## 4.7 Simulation

The OCI can also be run concurrently with the Simulator. This program simulates Tortuga Vs hardware as well as the competition environment, completely replicating a testing environment. Additionally, the simulator is able to display the view of the on-board cameras, thus enabling integration testing of almost all of the software systems without time-consuming physical dive operations. The simulator, through the use of these mock subsystems, enables concurrent development of the AI and subsystems it is dependent on.

## 4.8 AI

The main focus for the 2014 competition was to redesign the AI system, to be easier to use and to pave the way forward for a new software system in the future. The new AI state machine is designed to solve these problems by putting a strong focus on simplicity and reusability, giving developers a way to mix and match AI states to form a robust and stable routine. One of the key features of this new state machine is the ability to run nested state machines inside individual states, allowing a single state to represent a complex behavior, such as a search pattern or visual tracking. This way, one can build a repertoire of useful actions that can be easily reused and extended.

# 5 Vision System

The vision system serves to give Tortuga V's software information about the locations of visual objectives in the view of the forward and downward facing cameras. The vision system input consists of two Toshiba Teli FireDragon cameras which stream video over FireWire to the Mac Mini. The two cameras are stored in seperate housings from the rest of the vehicle's electronics. The external housings provide each camera with a flat optical viewing surface.

The vision system is split into a series of detectors, corresponding closely with the set of vision objectives. This allows the AI system to select which detector to use for a task and keeps the vision system simple. The use of YAML files allow the detector parameters to be modified without needing to recompile. Open source algorithms, such as OpenCV, allow for quicker implementation time and therefore more testing. In addition, open source algorithms are well documented and easy to understand. This greatly decreases the learning curve for new users.

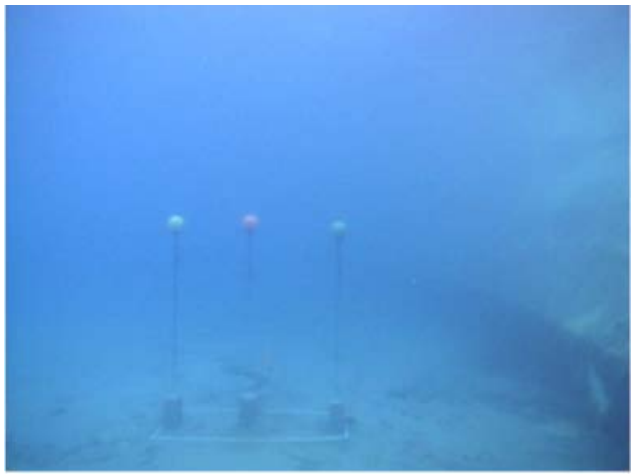

Figure 10: Original image from cameras

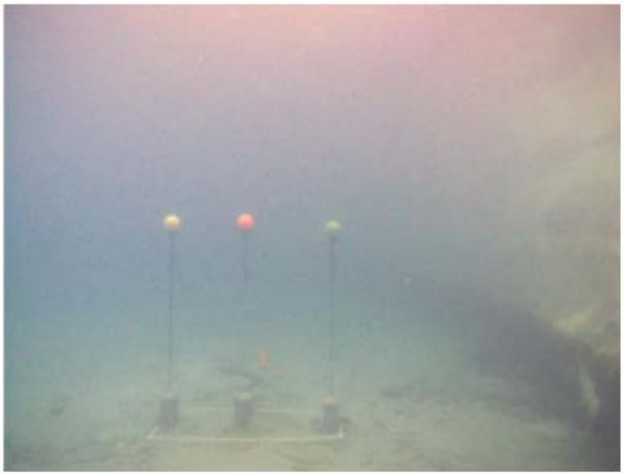

Figure 11: Image with white balancing applied

The detection of objects is heavily dependent on having an accurate color filter. An ideal filter is one that only detects the desired color but can also handle changes in lighting. Lighting changes are particularly important when operating outdoors

where clouds, weather, and time of day all affect what the camera sees. Two methods are used to improve the color filter: the first is the use of HSV color space rather than RGB. HSV (Hue, Saturation, and Value) is a color space which separates the color hue and the 'brightness' of the color. In this sense it is less susceptible to lighting changes. A second benefit of using HSV space it that only one channel, the hue channel, requires filtering. This reduces the computational complexity by 2/3 compared to RGB.

The second method to improve the color filter is the use of a white balance. Images taken underwater have a green or blue tint to them. This greatly increases the difficulty in differentiating some colors, such as green and yellow. A white balance corrects this tint and allows the ranges for the different colors to be increased without resulting in false positives.

No vision algorithm is perfect, particularly if it needs to run fast. False positives are an issue, as they can greatly confuse the AI. In order to reduce false positives, an object must remain in multiple frames before it is declared 'found'. The center of the object should be relatively close between the frames. If it cannot find a match between the previous two frames, or even one frame, it is declared a false positive. This helps reduce the effect of noise in the image.

# 6 Acknowledgements

Robotics@Maryland would like to thank its sponsors, without whom TortugaV would have never left the drawing board. Within the University of Maryland we would like to thank the Institute for Systems Research, the A. James Clark School of Engineering, the Space Systems Laboratory, the Student Government Association, the Maryland Robotics Center, and the Departments of Computer & Electrical Engineering, Aerospace Engineering, and Computer Science. We would also like to thank L-3 Communications, SAIC, MEMSense, Sidus Solutions, TC technologies, SubConn, Advanced Circuits, Teledyne RD Instruments, the NSF ECCS division, Northrop Grumman, USBFireWire, BAE Systems, and Lockheed Martin.

# References

[1] O. Egeland and J.M. Godhavn. Passivity-based adaptive attitude control of a rigid spacecraft. *IEEE Transactions on Automatic Control*, 1994.

[2] S. Skogestad and I. Postlethwaite. *Multivariable feedback control: analysis and design*. Wiley, Chichester, 1996.

[3] B. Wie and P.M. Barba. Quaternion feedback for spacecraft large angle maneuvers. *Journal of Guidance, Control, and Dynamics*, 1985.