DESIGN AND DEVELOPMENT OF THE YELLOWFIN UUV FOR HOMOGENEOUS COLLABORATIVE MISSIONS

Michael E. West,* Michael Novitzky, Jesse P. Varnell, Andrew Melim, Evan Seguin, Tedd C. Toler, Tomas R. Collins, John R. Bogle, Matthew P. Bradley, and Andrew M. Henshaw

Georgia Tech Research Institute (GTRI) has developed the Yellowfin, a small man-portable Unmanned Underwater Vehicle (UUV). The mission for Yellowfin is to conduct autonomous collaborative operations. The multi-UUV design allows for a much wider swath of the ocean to be observed and monitored, while collaborative operations allow multiple aspects of a mission to be tackled with distributed systems. Both oceanographic and military missions are aided tremendously by the use of such a UUV network. This paper introduces the modular and flexible design of the Yellowfin system and describes some of the technologies integrated within the system construct. The system and software architectures of Yellowfin leverage COTS technologies, including software whose foundation is MOOS-IvP, expanded to include several aspects of autonomy, communication with the WHOI acoustics modem utilizing the JAUS message standard, mission planning using MissionLab, mission execution via FalconView™ , front seat control with a microcontroller, and visualization with the Blender open-source, cross-platform suite of tools for 3D graphics.

INTRODUCTION

Unmanned Underwater Vehicles (UUVs) have become indispensable tools for undersea scientific, military, and commercial applications. Using UUVs for Intelligence Surveillance and Reconnaissance (ISR), Mine countermeasures (MCM) and Anti-Submarine Warfare (ASW) have shown great potential. [1] UUVs have shown to be invaluable tools for understanding ocean behaviors such as Harmful Algal Blooms, Chemical transport (Nitrates and Hydrates) and the impact of CO2 absorption. The current paradigm for the use of UUVs is a platform centric sensing system. Our research looks at the use of multiple vehicles which would allow for a net centric distributed sensing system and interrogate a much wider swath of the ocean. Currently, the US Navy's planned deployment of UUVs emphasizes single-vehicle operation through at least 2015 with uncertain capabilities for cooperating vehicles beyond that. [2] [3] However, to reduce the overall time and cost of acquiring data over large unstructured areas, multiple vehicles must be used. To address this need, the Georgia Tech Research Institute (GTRI) has developed the Yellowfin UUV. See Figure 1.

* Senior Research Engineer, Electronic Systems Laboratory, Georgia Tech Research Institute, 400 W. 10th St. Rm. 272 Atlanta, GA 30332-0829.

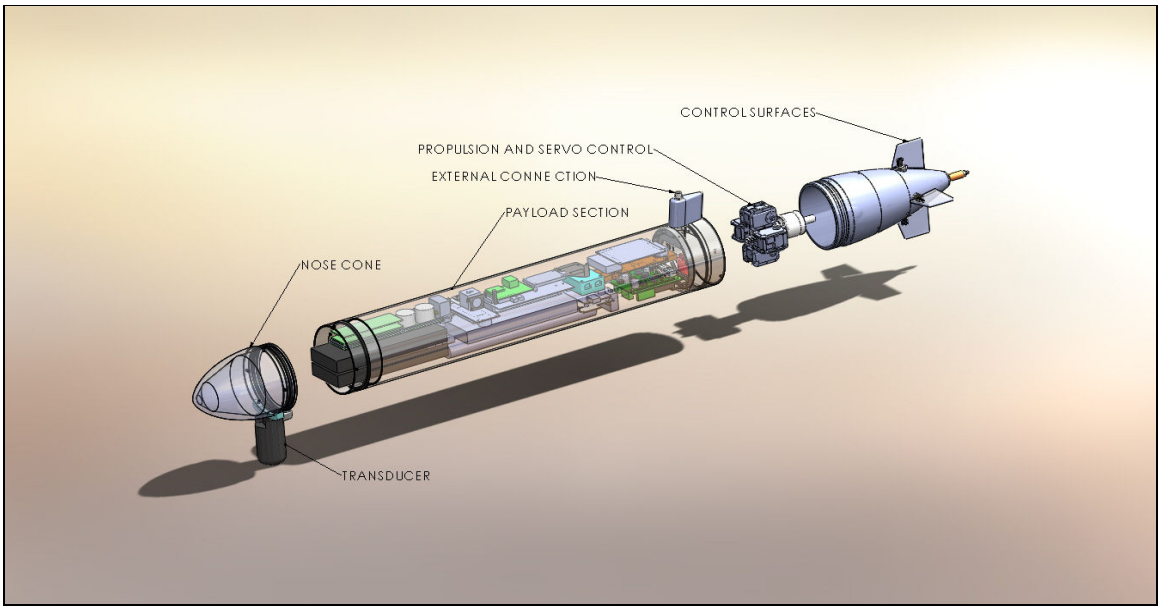


Figure 1: Yellowfin CAD Drawing

The Yellowfin is a man-portable UUV that weighs less than 17 lbs and was designed to support autonomous homogeneous collaborative operations. Advances in multi-agent collaborative control have been integrated into the system to autonomously control and coordinate multiple UUVs. The behavior modules were developed and implemented using the open-source Mission Oriented Operating Suite (MOOS) architecture along with the Interval Programming (IvP) Helm software module. The MOOS architecture was coupled with MissionLab open-source software tools developed by Georgia Tech. This suite of software is then used to develop and execute behaviors for both single UUV autonomy and for a team of autonomous collaborating UUVs. This coupling of software allowed for the development of highly capable UUVs using behavior-based autonomy. By using open source and standards-based development in the design of the Yellowfin, we have attained a modular and adaptable suite of platforms that support flexibility in mission execution. The design also decouples the vehicle autonomy logic from the mechanical control of the vehicle hardware, so that upgrades and adaptations are readily incorporated. Yellowfin is also designed to seamlessly incorporate different sensors and payloads, so that collaborative behaviors enable coordination among the different vehicle sensors and configurations. The result is a standards-based system capable of supporting intelligent autonomy, from perception to situational understanding to automated responses. The paper presents the foundations of the Yellowfin design.

YELLOWFIN DESIGN

A number of design challenges exist for the development of man-portable UUVs. Because space is at a premium, all of the electronics for power, communications, sensors, computation, and actuation, along with their packaging, must be carefully considered. A constant tradeoff between cost and development time must be mitigated, and the use of commercially available off-the-shelf components must be leveraged when possible. Figure 2 presents the high-level design process.

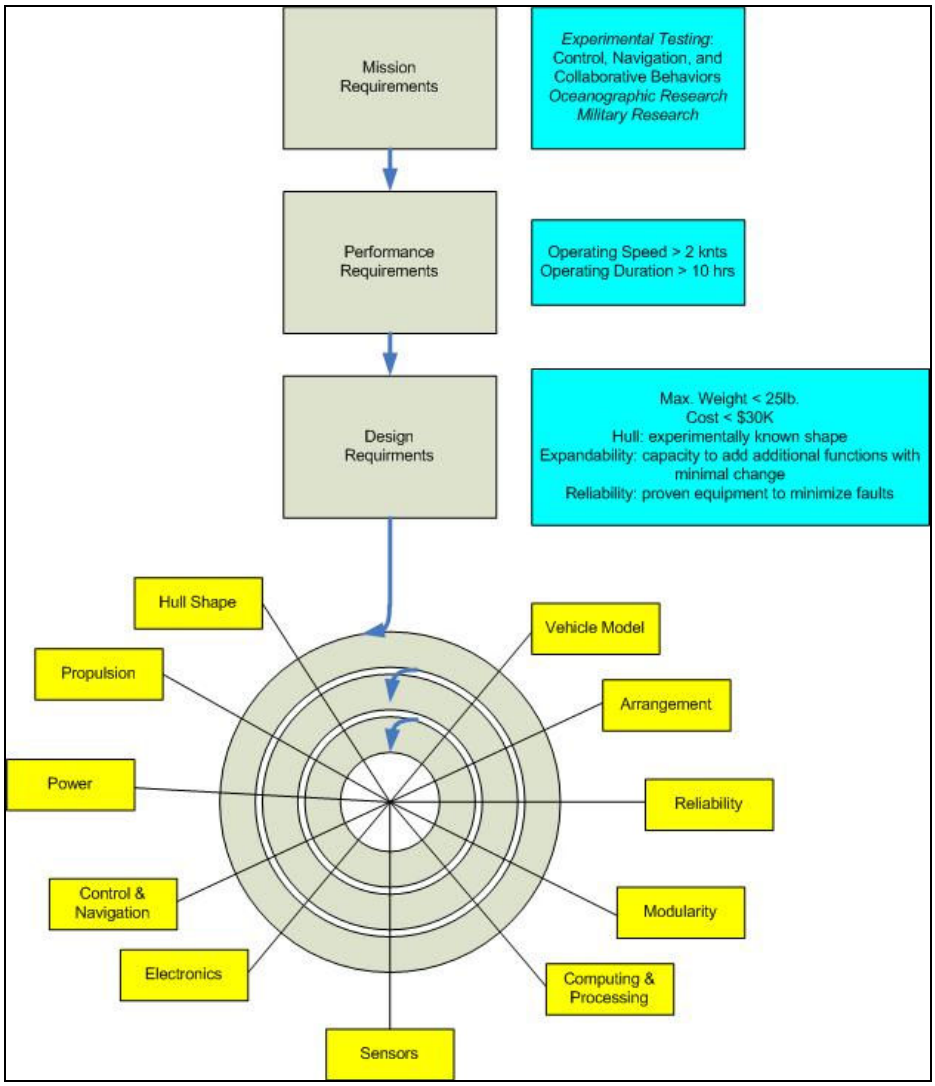


Figure 2: Yellowfin Design Approach

Design Approach

Yellowfin was designed to rapidly provide a solid foundation of integrated baseline functionality while still being adaptable to a wide range of mission-specific requirements. To keep cost and development time as low as possible while maintaining the quantity and quality of vehicle features along with system interoperability, the design of Yellowfin adopted COTS (Commercial, off-the-shelf) components, open-source software, and industry standards whenever feasible. The vehicle itself was designed through an iterative process where one prototype was designed, built, and tested followed by several additional vehicles. The software design also progressed in stages with the results of the implementation motivating additional cycles of design work.

Design Goals

The initial design goals generated requirements that were categorized into two primary areas: mission requirements and vehicle requirements [4]

Mission requirements

These are specific to the mission that Yellowfin will perform. While Yellowfin is capable of performing a diverse category of missions, there is a core set of features that is common to the requirements of most UUV missions. These include

- Control, navigation, and collaborative behaviors
- Validation through real and simulated testing
- Suitability for multiple applications, including oceanographic and military research

Vehicle requirements

These are specific to the Yellowfin vehicle. Particular values for performance specifications are determined by examining the requirements for various applications and were chosen to meet or exceed those requirements.

- Operating speed > 2 knots
- Operating duration > 10 hours
- Max weight < 17 lbs
- Cost < $30K
- Ability to accommodate various sensor and payloads

Resulting design

Coupling the mission and vehicle requirements resulted in a vehicle design with the following features. Figure 3 shows a fully assembled Yellowfin.

- Length: 889mm
- Diameter: 123.8mm
- Weight: 7.7kg
- Front Seat/Back Seat Driver Software Paradigm
- Modular, Behavior-Based Autonomy
- Mission Oriented Operating Suite (MOOS)
- JAUS Interfaces

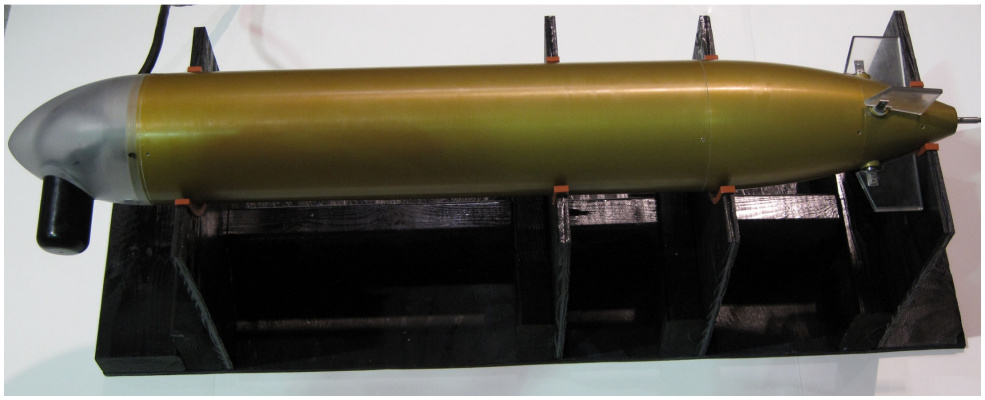
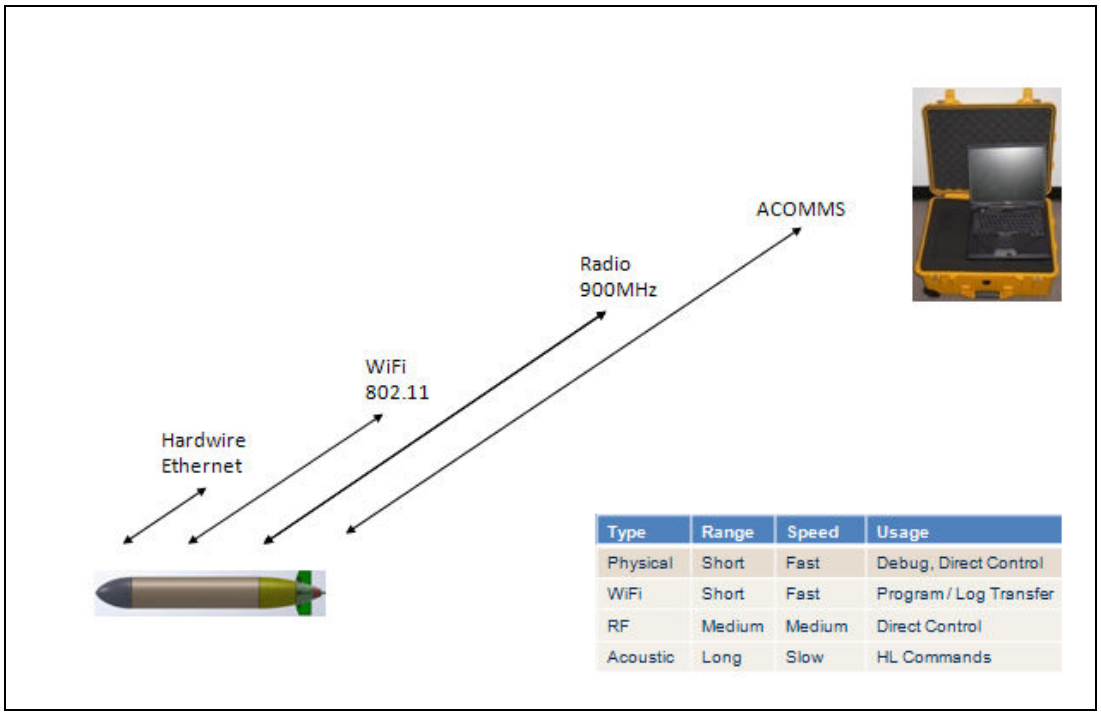


Figure 3: Fully Assembled Yellowfin

SUBSYSTEM OVERVIEW

Communications

Acoustic communications are enabled through a WHOI micro-modem, which is used throughout the UUV community and allows the Yellowfin to communicate underwater over fairly long distances at very low bandwidth. Wi-Fi provides a high bandwidth data link when the Yellowfin is at the surface of the water, but does not work underwater. Radio frequency (RF) communications fall in-between the acoustic and Wi-Fi in terms of bandwidth and works both above and below water (moderately well). Yellowfin is also equipped with an optional Ethernet tether that supports in-water drive testing, evaluation of new capabilities, and high-bandwidth connections when the Yellowfin is docked. Yellowfin uses an open-source implementation of the Joint Architecture for Unmanned Systems (JAUS) message protocol called OpenJAUS.* OpenJAUS provides a library of message routines that code and decode a variety of JAUS message types including system health, UUV pose, mission directives, and sensor data communication. The Yellowfin software also includes a library of routines that encode and decode the JAUS messages into and out of the NMEA 0183 standard protocol utilized by the WHOI acoustic modems.



Type	Range	Speed	Usage
Physical	Short	Fast	Debug, Direct Control
WiFi	Short	Fast	Program / Log Transfer
RF	Medium	Medium	Direct Control
Acoustic	Long	Slow	HL Commands

Figure 4: Yellowfin Communications

* http://www.openjaus.com

Sensors

Yellowfin has many of the most common navigational sensors, including GPS, IMU, compass, and pressure sensor. These are used for low-level motion control and for high-level localization. Yellowfin also includes moisture sensors for leak detection and a BlueView forward-imaging sonar. In contrast with traditional sonars where a single beam is mechanically rotated, imaging sonars implement multi-beam sensors that form several small acoustical beams at once. Imaging sonar is effective on moving platforms, whereas movement of traditional sonars during vehicle operation can cause data errors. Yellowfin uses the sonar in several ways, but this sonar specifically enables simultaneous localization and mapping (SLAM). See Figure 4. Other sensors and payloads can be easily incorporated into various-sized nose cones for Yellowfin.

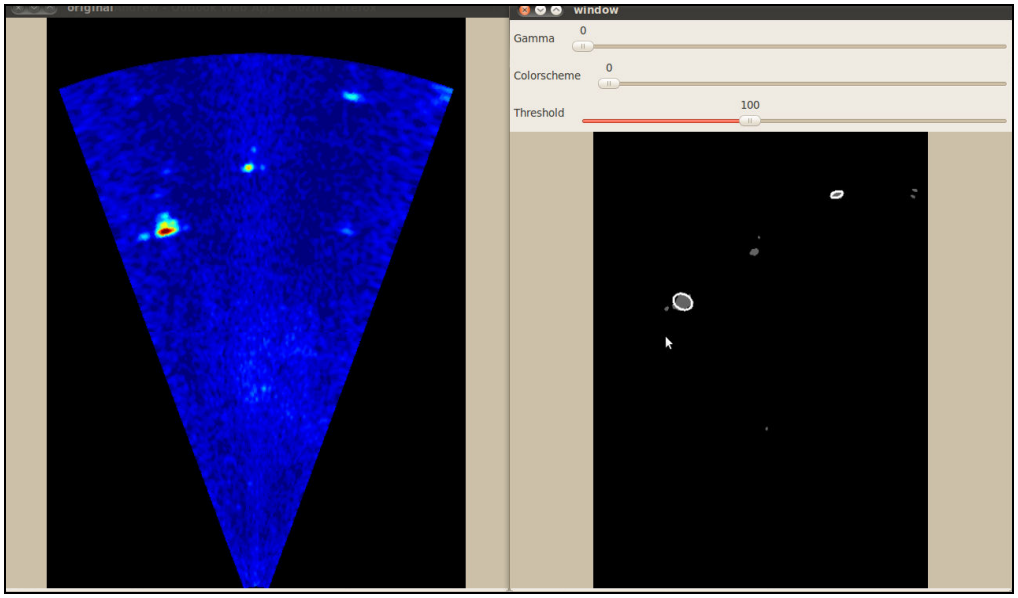


Figure 4: Yellowfin's Sonar GUI Tracking Buoys

Computation

Processing on the Yellowfin is split into two sections: a low-level processor, which manages all hardware and software integral to the sensors and actuators, and a high-level processor, which manages the autonomy and collaborative behaviors. The so-called "Backseat-driver" paradigm allows for the decoupling between high-level and low-level control.[5] This dual architecture was created to reduce the difficulty of modifying the more complex high-level software and to make the software platform-independent. It also reduces the risk of the less stable high-level software interfering with safety-critical low-level software. The low-level and high-level processors are able to communicate in a reasonably fast manner (high bandwidth, low latency) to exchange data.

Software

The Yellowfin has a complete software package from pre-mission planning to mission execution. See Figure 5. The pre-mission planning is performed by Mission Lab by organizing available behaviors to generate mission behaviors. Mission execution on the vehicle is performed using the MOOS-IvP suite of applications for high-level autonomy and an XMOS low level controller. Command and control is performed at a base station, using FalconView. A mission can be ex-

ecuted in the Yellowfin simulator or on the actual vehicles.

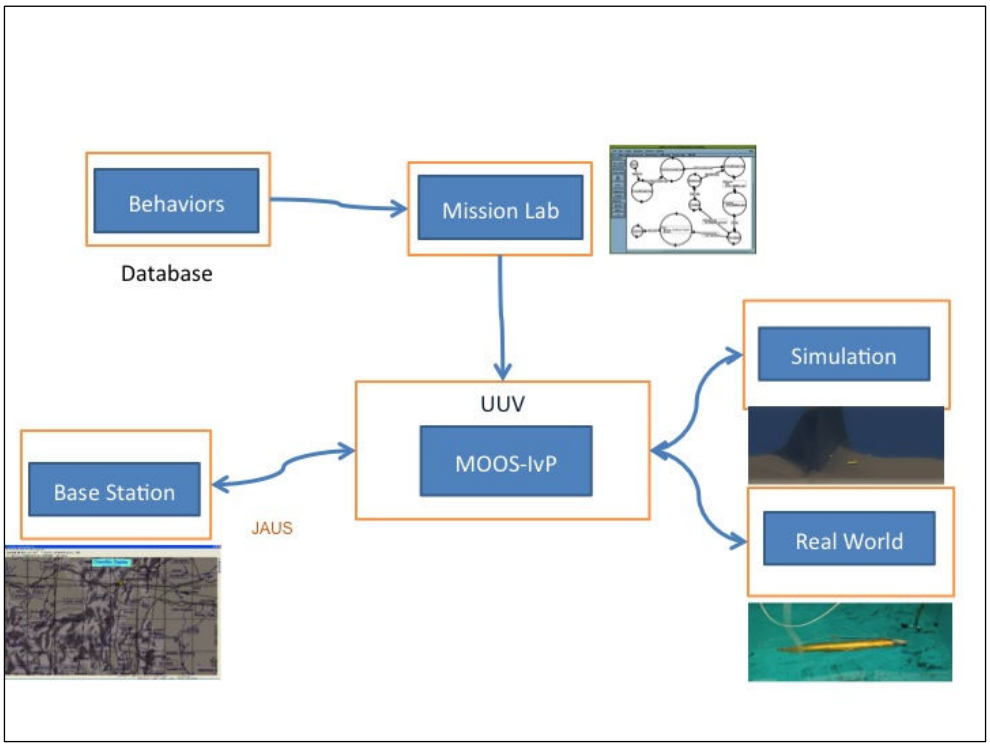


Figure 5: Yellowfin's Software For Mission Execution

Mission Planning

Yellowfin uses MissionLab for pre-mission planning. MissionLab was created at the Georgia Institute of Technology for the purposes of organizing and executing behavior-based architectures.[6] In particular, Yellowfin has a database of behaviors that can be organized for a particular mission based on requirements through the use of a GUI application. See Figure 6. This GUI translates the organized behaviors into mission files which Yellowfin's autonomy can execute during mission deployment.

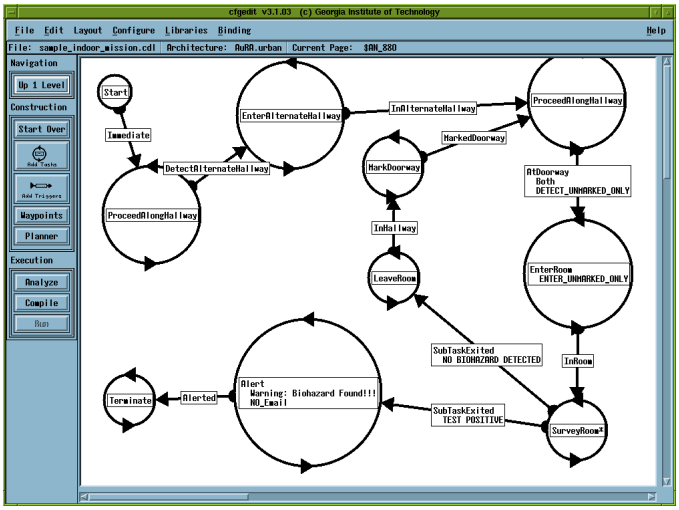


Figure 6: Mission Lab's Behavior-based GUI

Command and control of multiple UUVs is performed through a base station using the open source FalconView™* Software. FalconView™ is widely used by the United States Department of Defense for its aircraft mission planning and mapping capabilities and has over 40,000 users. FalconView™ provides for application extensions through a plug-in framework. The UUVs in this system can communicate to a base station server when or as needed through the Yellowfin communication system, and the FalconView™ application plug-in displays the vehicles' positions and telemetry information in real time using JAUS messages. The base station can also be used to send JAUS messages to the vehicles, such as waypoint and mission-based commands.

* http://www.falconview.org

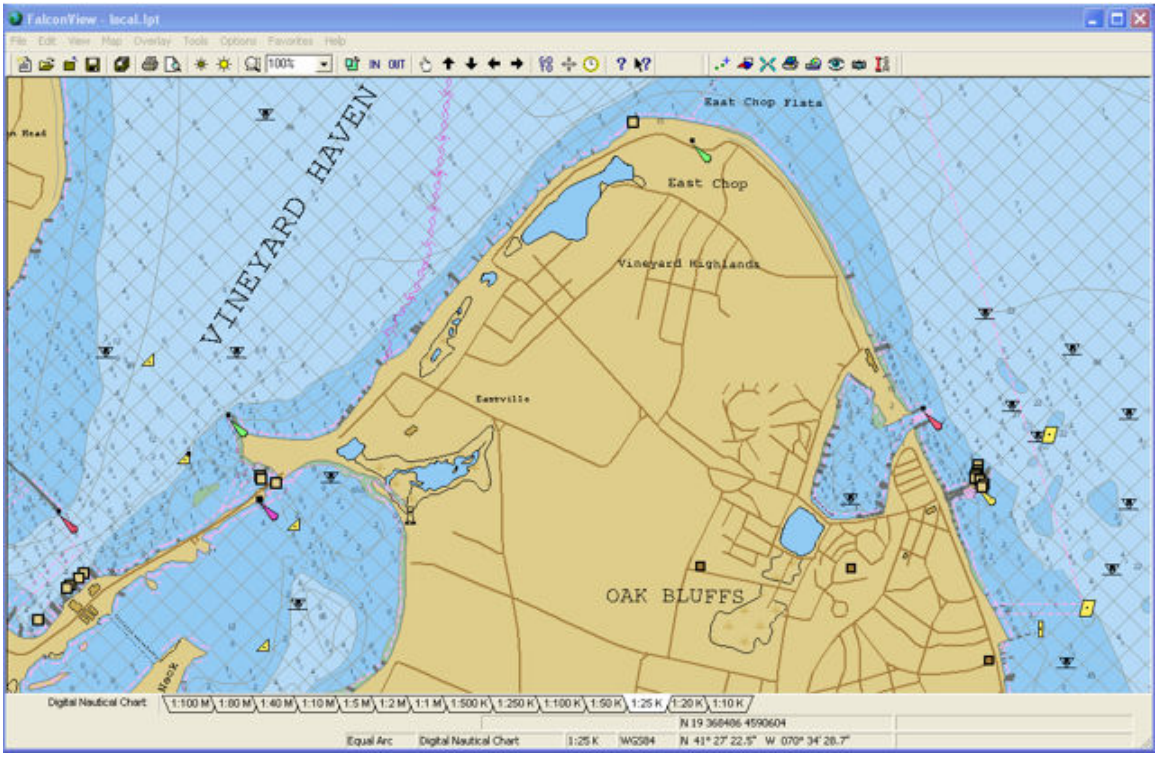


Figure 7: FalconView™ Mission Planning Software

Collaborative Autonomy

The high-level software architecture and autonomy for Yellowfin is provided by MOOS and MOOS-IvP, respectively. [7] MOOS is C++ cross-platform middleware, created and maintained by the Oxford Mobile Robotics Group for robotics research. It enables interprocess communication through a central database using a publish-subscribe architecture. MOOS-IvP, which is maintained by MIT's Laboratory for Autonomous Marine Sensing Systems (LAMSS), is a collection of MOOS-based applications designed for maritime autonomy. MOOS-IvP includes applications for communication, simulation, data acquisition, pre-mission planning, and post-mission analysis. IvP Helm is a behavior-based architecture that uses multi-objective optimization to support coordination of multiple competing behaviors. Included with IvP Helm are 17 behaviors and the ability to create new ones. Behaviors can be clustered according to different mission modes, making the entire cooperative system flexible to changing mission dynamics.

The software system has been designed to optimize the execution of a course of action to carry out a specific mission, given the situational awareness derived by the sensors. The mission represents both the stuational information and operational priorities. It includes a set of rules which control the execution of a set of behaviors that are not completely known in advance and occur during the execution of that mission. Thus, this enables

- The ability to react to unforeseen situations (no scripted cases), and
- Autonomous, on-the-fly planning and replanning

Example behaviors needed for collaborative autonomy include Navigation, Avoidance, Search, Investigate, Attack (independent and assisted), Assist, Rendevous/Loiter, Communicate,

and Negotiate. Each of these operations is implemented as an independent behavior that operates autonomously within its scope; each conducts real-time planning and analysis of the situation relative to mission execution, and each responds appropriately to the results of that analysis.

- **Search Area Collaboration.** When a vehicle completes searching an area, it communicates with its partners informing them that it is available for to assist them. A partner Yellowfin can assist in several ways. One form of assistance is to help another Yellowfin complete its mission (e.g., re-partition a large search area for one Yellowfin into two smaller search areas for two Yellowfins). Another form of assistance is if a partner does not respond (e.g., it may have been removed from the group before completing its mission), then the assisting Yellowfin will have know the first Yellowfin's mission and can assist in completing that mission based on the last communications that occurred between partners.

- **Negotiations.** Negotiation can occur between Yellowfins in support of a mission for any situation. Two behaviors (Negotiate and Assistor Negotiate) are designed for coordinating actions amongst many Yellowfins. A single Yellowfin using Negotiate can initiate negotiations with multiple vehicles in a single instance. Multiple vehicles use Assistor Negotiate to indicate they are available to assist in the coordinated operation. These two behaviors form the foundation for a generic communication between two, where one requests help and determines which partner is best to assist in the task.

- **Partitioning / Assigning Search Areas.** A capability is being incorporated that allows a Yellowfin that is designated as the supervisor to receive an entire mission and then delegate individual sub-missions to the group of UUVs under its control, including itself. Based on the mission, it decides whether to partition one large search area into multiple smaller areas, assign a single search area to each Yellowfin, or plan specific flight paths for each vehicle.

- **Re-Partitioning Search Areas / Supporting a Partner in Completing a Search Area.** Mission replanning can occur after each Yellowfin completes a given mission. This is in contrast to when replanning occurs only at rendezvous points. Also being incorporated is the ability to assist a partner in searching a designated area (e.g., re-partitioning a large area into two small areas where a second can assist). This includes completing the search mission of Yellowfins that are out of communication.

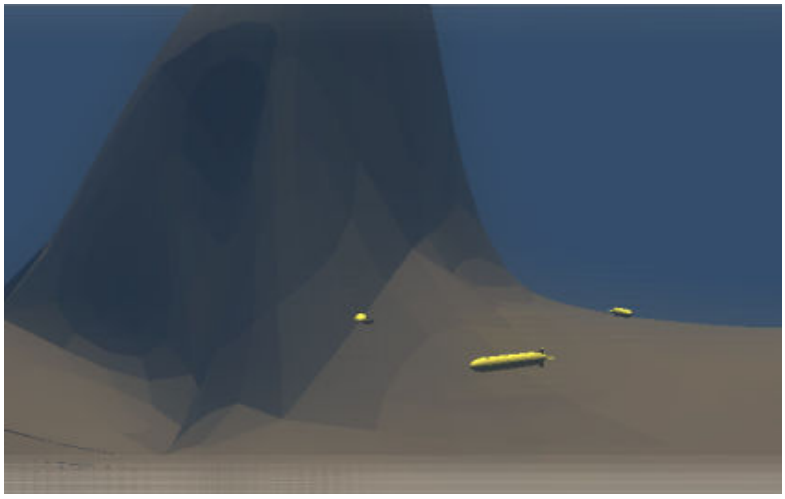


Figure 7: Yellowfin Simulator

YELLOWFIN SIMULATOR DESIGN

The Yellowfin simulator was designed for the purpose of developing autonomous behaviors and functionality in a fully realized 3D environment, as seen in Figure 7. A key aspect of this is the ability to model the Yellowfin vehicle within the simulator and to allow the Yellowfin software package to operate in a manner similar to that in which the vehicle will be deployed. Developing a simulator also provides the ability to develop multi-vehicle behaviors without requiring the cost of deploying multiple vehicles.

Visualization Software

The main visualization technology driving Yellowfin's simulator is the use of Blender, an open source cross-platform toolset for creating 3D worlds.[*] Blender provides a programming interface using the Python language as well as several graphical software development tools. A 3D model of the Yellowfin vehicle was imported into the simulation using Blender's 3D modeling tools.

Bathymetry Data

In order to simulate true deployment conditions, an underwater Bathometry can be imported into Yellowfin's simulator using GeoTIFF files of any location of interest. GeoTIFF is a non-proprietary standard for TIFF image files that contain embedded geographical or georeferencing data for use in constructing real world geographical images. These GeoTIFF files can be constructed from various sources including satellite imagery or elevation models.

Sensor Data Simulation

Yellowfin's simulator has the ability to simulate sensor data from the 3D environment. Ray tracing is utilized to create a sonar image of the simulated sonar's field of view. This allows for the testing of Yellowfin's target-tracking and simultaneous localization and mapping (SLAM) capabilities. These are two active research areas and contribute to Yellowfin's overall autonomy.

---

[*] http://www.blender.org

Simulator to Vehicle Interface

A MOOS database interface has been created for the Blender-based simulation in order to provide a gateway for the testing of Yellowfin's autonomy software. MOOS-IvP provides simulation tools, such as iMarineSim, to exercise a vehicle's autonomy. However, the provided simulator does not include an attractive 3D visualization with perception sensor data such as sonar data from the simulated sea floor as provided by bathymetry data. Connecting Yellowfin's simulator to the MOOS Database allows the Yellowfin autonomy software to receive the same input as the physical robot would receive during a mission and respond accordingly. This provides the ability to develop working autonomous behaviors in the simulator before actual deployment.

CONCLUSION

Yellowfin is a man-portable unmanned underwater vehicle. It has been designed to be deployed for both scientific and military missions, and its payload is adaptable to mission requirements. The design process emphasized low cost, high functionality and interoperability by utilizing off-the-shelf parts, open-source software, and industry standards. The software of Yellowfin allows for pre-mission planning through mission execution of a collaborative team of vehicles with robustness for different missions and real-time situational awareness. The Yellowfin simulator enables testing of the autonomy software in various life-like situations. Yellowfin's small size and robust software make it ideal for littoral missions and research into homogeneous collaborative operations.

ACKNOWLEDGMENTS

This work was supported by Georgia Tech Research Institute as part of a thrust to development of enabling technologies in unmanned underwater vehicles.

REFERENCES

[1] T. Bean, G. Beidler, J Canning, D. Odell, R. Wall, M. O'Rourke, M. Anderson and D. Edwards, "Language and Logic to Enable Collaborative Behavior among Multiple Autonomous Underwater Vehicles." *International Journal of Intelligent Control and Systems*. Vol. 13, No. 1, 2008, pp. 67–80.

[2] Navy UUV Masterplan 200, http://www.navy.mil/navydata/technology/uuvmp.pdf.

[3] Navy UUV Master Plan Update 2004.

[4] D. Furey, et al, "AUVSI/ONR Engineering Primer Document for the Autonomous Underwater Vehicle (AUV) Team Competition".

[5] M. Benahin, "White Paper – Software Architecure and Strategic Plans for Undersea Cooperative Cureing and Intervention", 2007.

[6] R.C. Arkin and T. Balch, "AuRA: Principles and practice in review." *Journal of Experimental & Theoretical Artificial Intelligence*, Vol. 2, No. 2, 1997, pp. 175-189.

[7] M. Benjamin, J. J. Leonard, H. Schmidt, and P.M. Newman, "An overview of moos-ivp and a brief users guide to the ivp helm autonomy software." MIT, Tech. Rep. MIT_CSAIL-TR-2009-028, 2009.