Modular Controls and Instrumentation Software for Icefin ROV

Charles Ramey[1,2], Matthew Meister[1], Anthony Spears[1], Josh Lutz[1], Daniel Dichek[1], Ben Hurwitz[1], Justin Lawrence[1], Jade Lawrence[1], Margaret Philleo[1], & Britney E. Schmidt[1].

[1]Earth and Atmospheric Science Department, Georgia Institute of Technology, [2]College of Computing Georgia Institute of Technology, Atlanta, Georgia, cramey7@gatech.edu

***Abstract*— Icefin is a hybrid ROV/AUV, designed to function as a robotic oceanographer in the Antarctic under-ice waters. Autonomous command, control, and navigation of such robotic platforms in remote sub-ice environments is extremely challenging, but also niche enough of an application to not benefit from the foundation of a wide community of open-source research. We will present here our approach and lessons learned in addressing this challenge through the integration of commercial off-the-shelf and open-source tool sets while working inside a framework that supports rapid, modular expansion of the vehicle's command and control software.**

Keywords—AUV, ROV, UUV, LCM, Software, Controls, Antarctica

I. INTRODUCTION

The Icefin vehicle is a modular, battery-powered, underwater vehicle with five thrusters that enable full control of XY position, depth, yaw, and pitch. Most importantly, the vehicle serves as a platform for an extensive suite of scientific instruments. A fully functioning Icefin vehicle was deployed to Antarctica for the austral summer 2017, where it collected high definition imagery, bathymetric, chemical, and sonographic data in never-before-observed environments. This vehicle leveraged lessons learned from the deployment of the prototype Icefin vehicle in 2014. Future deployments to Antarctica will involve surveying beneath the Ross Ice Shelf and the Ross Ice Shelf grounding line. The eventual goal for this project will be the development of an AUV capable of searching for life in Europa's ocean [1]. For more complete details, see the companion paper for this meeting, Meister et al 2018 (OCEANS 2018)[2].

The main software component of the Icefin system is composed of a customized version of Greensea's Balefire software. This software provides a framework for streaming sensor data and vehicle status to a control station, calculating inertial navigation estimates onboard the vehicle in real time, controlling the vehicle actuators in real time, and transferring high-level control signals from the surface control station to the vehicle during mission operation. The control software provides the capability for both real-time human operator control as well as autonomous control of the vehicle. Vehicle state and estimated position information is presented through a graphical interface to the operator at the surface control station. The actuator control architecture consists of a low-level front seat driver application running on the vehicle's onboard computer, whereas a higher-level back seat driver application is run at the surface control station. The front seat driver application controls the actuators directly through low-level hardware commands. The back seat driver application uses input from the autopilot or human operator to generate set points, which guide control signal mixing and generation with a set of PID controllers. Video streams from both vehicle cameras, and a visual representation of the sonar data is also presented at the control station in real time using an external host computer. Because recovery of the Icefin vehicle through the thick Antarctic ice shelves requires the use of a tether, the current configuration of the vehicle is designed to remain tethered with constantcommunication and control from the surface. Autonomy capabilities were not used during the 2014 Antarctic missions due to the early nature of the deployments, but they are fully integrated into the Balefire system and will be used in the 2018 season, enabled by custom enhancements to the software interface.

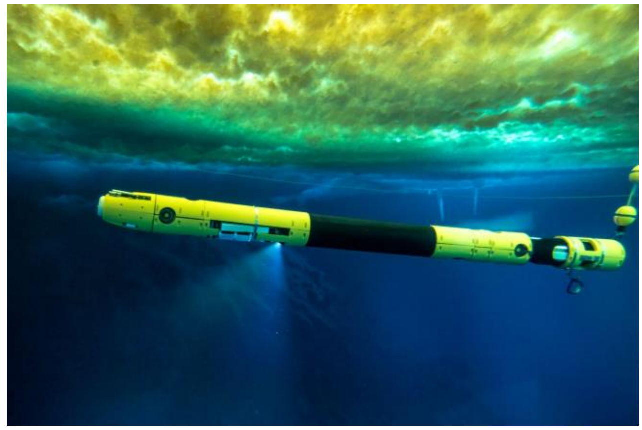

Fig. 1. The Icefin vehilce deployed under the ice in McMurdo Sound, Antarctica.

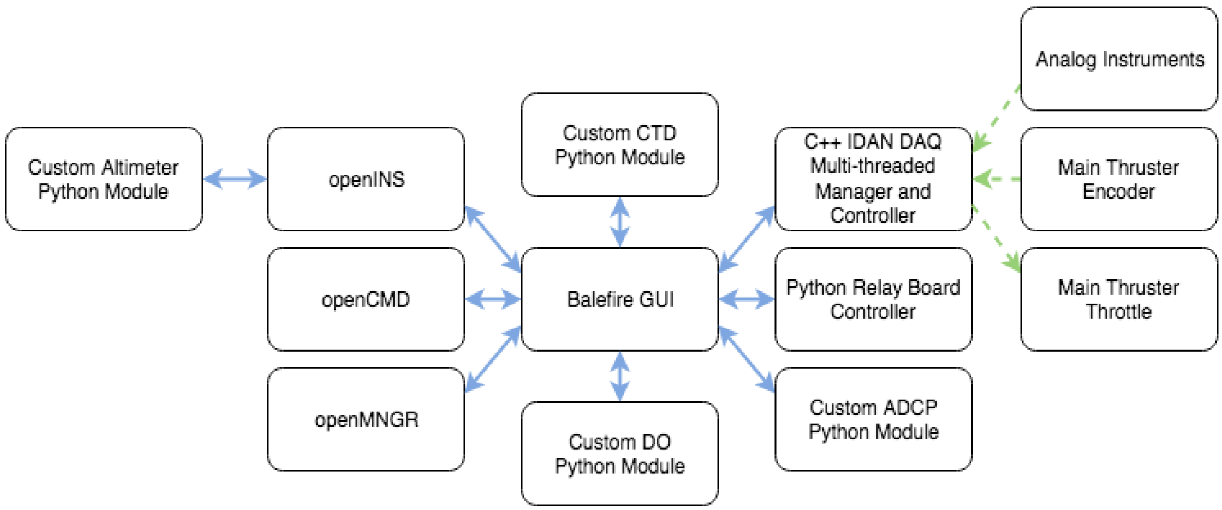

Fig. 2. The Icefin software architecture. Blue (solid arrows) represent LCM traffice. Green (dashed arrows) represent ananlog and discrete digital I/O signals.

II. MODULAR COMMAND AND CONTROLS

The two latest design iterations of Icefin rely on an RTD Intelligent Data Acquisition Node (IDAN) onboard computer, which is installed inside of the vehicle's main pressure vessel. The IDAN is a PC104 form-factor computer comprised of an Intel i7 powered single board computer, a serial communications module, an 18-bit A/D simultaneous sampling data acquisition module, and an ATX power supply unit. A fiber optic tether passes serial data, imagery data, and Ethernet packets between the IDAN inside the vehicle and the topside base station by way of MOOG fiber optic multiplexer boards.

Software for the IDAN is deployed on the Ubuntu 14.04 Linux operating system, which runs on the single board computer and provides a full development stack. Topside command and controls are hosted on a portable base station distributed across several laptops. The Lightweight Communications and Marshaling (LCM) protocol is used for intercommunication between software modules [3]. Multicast LCM messages are used both locally on the vehicle as well as across the fiber optic tether to the topside base station.

Balefire is built around LCM, which facilitates custom and experimental software modules developed outside of the Greensea Systems software ecosystem. The current Icefin software architecture extends the Balefire system's capabilities with a series of modules written in Python and C++ (Figure 2). For high-level tasks such as simultaneously parsing asynchronous serial inputs, Python modules were written which receive and parse each system or sensor's serial messages, encode the information as LCM messages, and then broadcast the messages for other scientific and control modules to utilize. To interface with low-level system components such as the IO board for controlling the digital to analog convertors (DACs) and analog I/O interfaces, several modules were written in C++ to optimize for embedded real-time processing and ease of working with vendor-supplied drivers and C libraries. Similar to the Python modules, each C++ module accepts data through an I/O or serial interface, encodes the data into an LCM message, and then broadcasts the messages over UDP multicast. Each software module that has developed in house for the vehicle is listed and described below in Table 1.

The vehicle also relies on several embedded computers to interface between vehicle subsystems and the IDAN control computer. For the 2017 field season, a Teensy microcontroller running the Arduino bootloader was utilized to control the vehicle's master power control board and main thruster power control board. The Teensy power controller was programmed to receive commands over serial and then execute functions such as monitoring voltage levels on the power control boards, monitoring temperatures on the power control boards, and switching power control boards on and off.

TABLE I. CUSTOM ICEFIN SOFTWARE MODULES

Custom Software Modules	
Module	***Module Functionality***
ADCP parser	This Python module scrapes acoustic Doppler current profiler measurements from the serial output of the Doppler velocity log, encodes the data as an LCM message, and publishes the LCM message.
DO parser	This Python module parses the serial output from the dissolved oxygen sensor, encodes the data as an LCM message, and publishes the LCM message.
CTD parser	This Python module parses the serial output from the conductivity and temperature sensor, encodes the data as an LCM message, and publishes the LCM message.
ALT parser	This Python module parses the serial output from the altimeter, encodes the data as an LCM message, and

Custom Software Modules	
Module	***Module Functionality***
	publishes the LCM message.
Analog instruments manager	This Python module sends analog read commands to the DAQ controller, parses the data returned from the analog to digital convertor, encodes the data as an LCM message, and publishes the LCM message.
Relay board controller	This C++ module accepts relay control LCM commands from the Balefire GUI and switches the USB relay board relays accordingly.
Relay status manager	This Python module monitors LCM relay control commands and sends LCM relay status commands to update the Balefire GUI relay status states.
DAQ board controller	This C++ module receives LCM commands for the analog to digital convertors and digital to analog convertors, handles all low-level device configuration, executes analog and discrete digital I/O commands, and sends LCM response messages with result data.
Battery logger	This Python module parses the serial output from the battery management system, encodes the data as an LCM message, and publishes the LCM message.
Main thruster encoder	This Python module sends LCM analog read commands to the DAQ controller to monitor the output of the main thruster's rotation encoder, receives the ADC read result LCM message from the DAQ controller, encodes the data as an LCM message, and publishes the LCM message.
Power Board Controller	This Arduino program receives control commands over serial, monitors power control board status, and controls the vehicle's master power switching board.
T200 thruster controller	This Arduino program receives thruster commands over serial and outputs PWM signals to the vehicle's T200 thrusters.

III. Results

The Icefin vehicle was deployed 12 times during the 2017 Antarctic summer field season. Over the course of the dives several minor software upgrades were necessitated. In order to monitor the vehicle's power system status and charge level during deployments, a Python module was written to parse the battery management system status outputs. The battery-monitoring module reports battery charge levels, battery current outputs, and battery cell statuses to the Balefire GUI through LCM messages. Additionally, exception handling was added to all Python scripts to recover and reconfigure in the event of sensor power loss or system restart.

The usage of a Teensy microcontroller as the control interface for the master power control board and the thruster power controller board proved to be a success during the 2017 Antarctic field season. The Teensy board was easy to prototype with, provided reliable operations as the power controller, and was inexpensive to replace in the event of a system failure. After the field season, a decision was made to utilize additional Teensy microcontrollers in the vehicle. A second Teensy has been added to the vehicle for the purpose of controlling a set of four Blue Robotics T200 thrusters via pulse width modulation (PWM). The Teensy has been programmed to receive serial commands from the IDAN control computer and then output PWM signals to control each individual thruster.

Throughout the development and deployments of Icefin, the LCM protocol and its associated suite of tools has proven effective at logging and replaying vehicle and sensor data. Over the course of the 2017 field season, Python modules for publishing LCM messages were written for several vehicle subsystems. Each module allowed further information about vehicle engineering operations and scientific instrumentation to be captured in a format that provided for easy visualization and storage. As Icefin engineering efforts continue, LCM modules will be written for every new sensor and engineering control system as they are integrated into the vehicle.

Since the 2017 Antarctic field season, additional full vehicle testing and tuning was performed at Georgia Institute of Technology, Lake Allatoona, and the Florida State University Marine and Coastal Laboratory. During these testing deployments PID control loops were tuned within the Balefire Open Command system to control the vehicle across its full 5 degrees of freedom (X, Y, Z, Yaw, Pitch). With the

Fig. 3. Balefire waypoint following behavior testing in the Gulf of Mexico.

PID control loops operational, waypoint following and station keeping operations can now be exercised during deployments. Waypoint following has been utilized in Lake Allatoona and the Gulf of Mexico to conduct autonomous surveys with the vehicle. Using DVL based positioning data, waypoint following will enable autonomous under-ice surveys during the 2018 austral summer Antarctic field season.

IV. Future Work

In preparation for the 2018 field deployment, the Icefin software team is currently developing a Doppler velocity logger (DVL) sensing module which will enable acoustic Doppler current (ADCP) profiling and will supplement the Balefire openINS (open Inertial Navigation System). In addition to developing custom supplemental software modules, the team is also experimenting with structure from motion software to create 3D models from vehicle imagery data. During the 2018 field deployment an NVIDIA Jetson TX2 embedded computer will be used to prototype real-time computer vision pipelines for feature recognition and simultaneous localization and mapping (SLAM). Real-time feature recognition and SLAM will greatly advance under-ice autonomy capabilities.

The establishment of bridges between the Balefire system and the custom software modules of Icefin and the MOOS [4] and ROS [5] simulation, planning, and controls software will further extend both the autonomous and human controlled capabilities of the vehicle. This will further incorporate open source command and control software into the Icefin vehicle. LCM bridges will allow units from each platform to communicate and collaborate with each other. The bridge architecture will allow for Balefire to run as normal, which will maintain the existing level of software systems reliability.

Additional future software developments will be focused on the in-house development of an inertial navigation system (INS), an open source sensor sampling and logging suite, and advanced autonomy utilities for navigation and scientific discovery. The 2018 field deployment will see the most capable version of this vehicle to date. In presenting our development efforts and field deployment experiences, we hope to support similar research in the community and provide a catalyst for future collaborative discussion around such unique applications.

Acknowledgment

This work was supported by NASA PSTAR RISE-UP program grant NNX16AL07G, PI B.E. Schmidt as well as funds from the EAS department and Georgia Tech. Field work in Antarctica was supported by NASA and NSF under USAP project number B-041-M. The authors thank the USAP McMurdo Station staff for their assistance in the success of this program.

References

[1] A. Spears, M. West, M. Meister, J. Buffo, C. Walker, T. Collins, A. Howard, B. Schmidt, "Under Ice in Antarctica: The Icefin Unmanned Underwater Vehicle Development and Deployment," in *IEEE Robotics & Automation Magazine*, vol. 23, no. 4, pp. 30-41, Dec. 2016. doi: 10.1109/MRA.2016.2578858

[2] M. Meister, D. Dichek, A. Spears, B. Hurwitz, C. Ramey, J. D. Lawrence, M. Philleo, J. Lutz, J. Lawrence, B. E. Schmidt. “Icefin: Redesign and 2017 Antarctic Field Deployment”. Oceans 2018. Charleston, South Carolina. 22-25 October 2018.

[3] A.S. Huang, E. Olson, and D.C. Moore, 2010, October. LCM: Lightweight communications and marshalling. In Intelligent robots and systems (IROS), 2010 IEEE/RSJ international conference on (pp. 4057-4062). IEEE.

[4] M.R. Benjamin, H. Schmidt, P.M. Newman, and J.J. Leonard, 2010. Nested autonomy for unmanned marine vehicles with MOOS-IvP. Journal of Field Robotics, 27(6), pp.834-875.

[5] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A.Y. Ng, 2009, May. ROS: an open-source Robot Operating System. In ICRA workshop on open source software (Vol. 3, No. 3.2, p. 5)