Passive Floating Probe

Authors: *

Vid Keršič
University of Maribor
Faculty of Electrical
Engineering and Computer
Science
Koroška cesta 46, Maribor
vid.kersic@
student.um.si

Urban Knupleš
University of Maribor
Faculty of Electrical
Engineering and Computer
Science
Koroška cesta 46, Maribor
urban.knuples@
student.um.si

Michele Perrone
University of Maribor
Faculty of Electrical
Engineering and Computer
Science
Koroška cesta 46, Maribor
michele.perrone@
student.um.si

Tadej Šinko
University of Maribor
Faculty of Electrical
Engineering and Computer
Science
Koroška cesta 46, Maribor
tadej.sinko@
student.um.si

Mitja Žalik
University of Maribor
Faculty of Electrical
Engineering and Computer
Science
Koroška cesta 46, Maribor
mitja.zalik@
student.um.si

ABSTRACT

This paper illustrates a student project design of an autonomous passive floating probe, which gathers data from its on-board sensors. The data are subsequently transmitted via the Iridium satellite network to a dedicated server which then displays the information on the web. Once assembled, the probe is planned to be deployed into the Atlantic Ocean. The main features of the probe are a solar panel with a rechargeable battery, a GPS module, an Iridium satellite modem, a vast set of sensors and an ARM-based microcontroller running a custom firmware based on FreeRTOS.

Keywords

Remote sensing, data gathering, real-time systems

1. INTRODUCTION

Remote sensing and data retrieval is a challenging task because physical access to the equipment is limited or impossible; more so, if executed in a harsh environment such as the oceans. Operating in such an environment brings additional challenges concerning protection from water, salt corrosion, plaque, or waste build-up. Furthermore, unpredictable weather can affect the energy autonomy of the de-

vice or the accuracy of sensor data. To overcome these problems, we propose a design for a passive floating probe, which is accessible due to its consumer-available components.

The proposed design follows a modular architecture, which allows faster prototyping and more advanced future iterations based on this outline.

The main difference between a passive and an active floating probe is in the way it interacts with its surrounding environment. A passive probe only listens and gather information through its sensors and move along with the ocean currents, while an active probe is equipped with a propulsion system that would allow it to change its course by itself based on sensor data and parameters or remotely by an operator.

The goal of this project is the acquisition of real-world measurements for analysis and distribution. A dedicated website [1] is planned to show project progress, visualize received data once deployed and enable distribution of collected data to interested parties.

2. RELATED WORK

Before discussing the proposed design, this paper outlines a handful of projects which had been the primary source of inspiration for the authors. The main inspiration comes from the Maker Buoy project [2]. Its features are a modular building design powered by solar energy, communication via the Iridium satellite network and the use of a floating exterior design, which is the basis of this paper's proposed design (discussed in Sec. 3). A project called OMNI (Ocean Monitoring Network Initiative) uses a similar hardware approach to the Maker Buoy [3].

The ZL1SIX Ocean Floater is a floating buoy powered by a battery pack which uses an Amateur radio transmitter to

*Listed in alphabetical order

transmit data on a 10 MHz Amateur Radio band [4].

The international project Argo is an initiative that uses thousands of battery-powered floats around the world to measure the temperature, salinity, and velocity of the ocean currents. The floats collect data by staying submerged underwater and transmit data by coming up to the surface and sending it via a satellite antenna. As of this writing, the project has 3869 floats active across all the oceans [5].

The main difference between the outlined design and other similar projects is the presence of a Geiger counter, a gas sensor, a microphone ¹, a camera ², and the use of a more capable additional microcomputer system in the form of Raspberry Pi for processing extensive data and ease of development of such processing software.

3. PROPOSED DESIGN

This section outlines the design of the probe, the software, and the communication between them. Because the proposed design is not necessarily final, minor changes can happen to individual modules while components are sourced.

3.1 Hardware design

For its operation, the probe must accomplish different tasks. Modules that contribute to the same task are classified into a subsystem. All proposed subsystems are shown in Fig. 1. Modules inside these subsystems will be presented later.

The core of the probe will be a SensiBLE SIMBA-PRO development board. Besides various sensors, which are included in the inner sensors subsystem, it will feature an STM32L476xx microcontroller with an ARM Cortex M-4 m processor. The microcontroller will retrieve data from the sensors and send them to the data storage subsystem. The most critical data will be sent to our server daily using the communication subsystem. When the system has access to enough power, advanced operations, such as image capture, sound recording, and advanced data processing will be executed on an external Raspberry Pi module (which is part of the image capture and processing subsystem).

The data storage subsystem shown in Fig. 2 consists of a micro SD card that will store the raw collected data and an adapter for the storage device. Since satellite communication is expensive, not all data will be sent to the server (e.g., only average or extreme measurements during the day). Nevertheless, every measurement will be saved to the data storage device. This way, if the probe is ever recovered, much more data could be analyzed. Even with low chances for recovery event, it was decided to include this system due to its low cost.

The inner sensors subsystem shown in Fig. 3 consists of the sensors which operate from inside the probe. Some of them gather engineering information about the environment inside the casing and can, therefore, reveal some problems with the probe (e.g., increased temperature of the processors, change in inner pressure or humidity which could in-

¹possible detection of nearby ships or wildlife for imaging purposes

²in order to visually detect plastic debris

Figure 1: Subsystems are shown in rectangles. The microcontroller synchronizes the work of all the sensors. The case is not part of any subsystem, since it is a passive element and therefore not controlled by the microcontroller.

Figure 2: Modules in the data subsystem

dicating that the watertight case is damaged). Other sensors can detect and measure outside sources (GPS signal, Beta and Gamma radioactive rays, sound, vibration, and light – since the case is transparent).

The outer sensor subsystem shown in Fig. 4 consists of sensors that cannot perceive outside environmental changes from the interior of the airtight probe. As such they are placed

Figure 3: Modules in the communication subsystem

Figure 4: Modules in the outer sensors subsystem

outside of the probe casing. To protect the sensors, when they are not in use, an additional standalone module with a hatch will be used. Connection to the probe will be through waterproof connectors. The primary purpose of the outer sensors is the detection of air pollution.

The image capture and processing system shown in Fig. 5 will contain a processing unit (a Raspberry Pi 4) separated from the microcontroller, which is capable of various image processing methods (e.g., pattern recognition, image compression). Since running the processing unit uses more power, it will be working only in short time intervals in order to process and transfer the needed data to the main microcon-

Figure 5: Modules in the image processing subsystem

troller. For capturing an image, the module will also feature a camera and 360° lens assembly, which makes it possible to take panoramic photos.

3.2 Firmware design

The software on the probe is separated into three standalone modules. Their physical execution flow and physical location in microcontroller flash is shown in Fig. 6. This design will enable data gathering, over the air updates and possible recovery of essential functions in the event of an unforeseen error in programming or damage to the microcontroller flash.

Figure 6: The outline of the firmware.

The bootloader module will be responsible for deciding and booting the appropriate firmware. This feature will enable the probe to turn itself off due to the lack of power. An additional feature will be the ability to flash prepared firmware from an external storage, enabling it to restore a previous firmware or to perform firmware updates prepared by the full firmware. In case of a fatal error during the booting of the full firmware, the failsafe version of firmware will be loaded.

The failsafe firmware will support only essential functions such as communicating with the GPS sensor, the Iridium modem, and the external storage. Its purpose is the continuation of the highest priority task and a last resort for recovering from critical errors. When booted, it will periodically send a package with GPS coordinates, requested diagnostic messages and possible direct flash commands. This

operation will allow it to try and recover the functionality of the probe or at least receive the most critical data.

The full firmware will feature FreeRTOS with tasks to read data from all of the sensors, process and save the data and communicate through the Iridium satellite network. It will be able to receive new settings (e.g., a new task schedule) or differential updates, which will be processed and a new version of the firmware prepared on the external storage device.

This modular firmware design allows for the flash memory to be partitioned into sections. The bootloader and failsafe sections will be read-only, so it will not be possible to change their content. Instead, they will be less complicated and extensively tested. There will be a section after that which can be changed by the first two modules. Also, the preparation of the full firmware allows the tasks to be compiled and linked into predetermined sized blocks, which will allow for more efficient and partial upgrades since a change in one task should not require to change or move compiled code from every other task.

3.3 Software design

The mediator of the communication between the users and the probe has the most significant role in supporting the proposed architecture. For this purpose, custom software is implemented. To accommodate the need for adaptability and accessibility of the software being developed, a dedicated server is necessary.

As of the writing of this paper, the server has an Intel® Core™ i7 930 CPU with a base frequency of 2.80 GHz and four physical cores, 12 GB random access memory (RAM), two 1 TB hard disk drives (HDDs) and a 128 GB solid-state drive (SSD). For the purpose of data redundancy and reliability, a Redundant Arrays of Inexpensive Disks (RAID) 1 is set up on the two used HDDs [6]. The server runs a Linux operating system Debian Gnu/Linux 9 (stretch).

The tasks, running on the server, are divided into three main groups: two-way communication with the probe, data handling, and hosting services and websites.

As already mentioned, the probe will send data using expensive satellite communication. The cost is calculated based on the number of sent messages [7], therefore the data are compressed to decrease the number of messages. The packages in the Iridium satellite network will be transmitted from the sending device to the predefined web service, which will run on the described server. The received data must be decompressed before it can be used. Communication can also occur in the other direction – from the server to the probe. In this case, only the critical updates are sent (e.g., firmware updates).

Data handling refers to storing, analyzing, and visualizing received data. The received data are planned to be stored in the InfluxDB database, which is designed for the time-series data. This way, data can be efficiently queried for analysis and visualization. Data analysis is necessary for controlling the probe and research. Based on the data received from the core of the probe, we can inspect its state and act upon

unexpected failures with firmware updates. The data from the additional sensors will be used for the analysis of the surroundings of the probe (e.g., water pollution). For data visualization, Grafana will be used due to its supported integration with InfluxDB. Different visualization techniques will help the authors in monitoring the probe and improve the research. The dataflow is presented in Figure 7.

Figure 7: A high-level depiction of the data flow in the communication between the probe and the server.

The analyzed and visualized data will be available to the authors on the web-based developer dashboard, hosted on the server. The server also hosts a custom-made public website, which shows only data relevant to the general public [1]. For both websites, Nginx, Node.js and MongoDB are used. Because of the growing scale of the project and its codebase, the server also features GitLab for source code management and OpenProject for project management. All services run in separate Docker containers for easier management. Daily backups are implemented to prevent unexpected loss of data.

4. CONCLUSION

The release of the probe into the ocean is planned for late February 2020.

The main project’s goal is to gather various types of information about the ocean, such as temperature and currents.

Based on other explored projects, the expected battery life of the probe is around 300 to 400 days, but the authors will try to extend that life by utilizing its smart battery management.

The collected data may prove useful to large scale initiatives such as the Ocean Cleanup [8] and Argo [5], therefore giving a further incentive to the support of the project.

Technical difficulties related to different electrical and mechanical aspects of the probe are to be expected, such as its water tightness, the lifespan of its various sensors and the battery, and electrical failures due to increased humidity and temperature. As far as the software side of the project is concerned, complications related to the maintenance and setup of the services running on the server may arise, such as data handling, data visualization, security and so forth.

5. ACKNOWLEDGMENTS

The authors acknowledge the financial support from the Institute of Computer Science of the Faculty of Electrical Engineering and Computer Science and would like to thank mag. Jernej Kranjec for his guidance and assistance.

6. REFERENCES

- [1] ZZZ. Our homepage. <http://zzz.feri.um.si/>, July 2019. Accessed on 2019-5-9.
- [2] M. Buoy. Homepage. <https://www.makerbuoy.com/>, May 2019. Accessed on 2019-5-8.
- [3] designlab.ac. Omni - ocean monitoring network initiative. <https://hackaday.io/project/165963-omni-ocean-monitoring-network-initiative>, July 2019. Accessed on 2019-5-9.
- [4] B. Sutton. Zllsix ocean floater build and voyage archive page. <https://www.qsl.net/zllrs/oceanfloater1.html>, July 2016. Accessed on 2019-5-9.
- [5] Argo. Homepage. <http://www.argo.ucsd.edu/>, July 2019. Accessed on 2019-5-9.
- [6] P. M. Chen, E. K. Lee, G. A. Gibson, R. H. Katz, and D. A. Patterson. Raid: High-performance, reliable secondary storage. *ACM Comput. Surv.*, 26(2):145–185, June 1994.
- [7] R. Seven. Rock seven | truly global gps tracking and messaging systems using iridium satellite | rockblock. <https://www.rock7.com/products-rockblock>, 2014. Accessed on 2019-7-28.
- [8] T. O. Cleanup. Homepage. <https://theoceancleanup.com/>, July 2019. Accessed on 2019-5-9.