Design Aspects Of An Open Platform For Underwater Robotics Experimental Research

Carlos R. Rocha*, Rogério M. Branco†, Lais A. da Cruz‡, Marcos V. Scholl§,
Matheus M. Cezar§ and Felipe D. Bicca§

Federal Institute of Education, Science and
Technology of Rio Grande do Sul - Rio Grande Campus
Rio Grande(RS), Brazil
* Email: carlos.rocha@riogrande.ifrs.edu.br
† Email: rogerio.branco@riogrande.ifrs.edu.br
*‡ Telephone: +55–53–3233–8708

§IFRS - Rio Grande
Federal University of Rio Grande
Rio Grande(RS), Brazil

***Abstract*—The early results in the development of a platform for experimental underwater robotics research are presented in this work. This platform consists of an Unmanned Underwater Vehicle (UUV) and a Mission Monitoring/Control System (MMCS) unit to teleoperate and to supervise the underwater vehicle. Following the initial phases of a product design methodology, a conceptual model for the platform was obtained. This model and the design process are analyzed here. The first results in the implementation of a prototype are also presented.**

I. Introduction

Underwater Unmanned Vehicles (UUV) are fundamental for submersed operations. This stimulates research in several different areas such as energy, communications, control, sensors, localization, motion planning and hydrodynamics. Autonomous operation of such vehicles, in particular, is considered a major challenge for underwater robotics [1]. However, many research results are theoretical or verified by simulation. So, experimental validation is needed [2]. There are many difficulties to execute experimental research in this area, due to the complexity of operation in a real environment and the costs involved. The robotic platform has high acquisition costs, especially in Brazil, where this research area is mainly motivated by the need of national technology development for the offshore gas/oil exploration industry [3], [4]. Also, several commercial plaftorms have closed, proprietary architectures which difficult access to the hardware/software in order to implement the experiments.

In order to stimulate and to increase experimental research/validation in this area, a project to develop an open source, relative low-cost hardware/software plaform for underwater robotics was proposed. This work presents the initial results of this project, consisting in the design phase aspects and initial software development of the platform. This platform is composed by an Unmanned Underwater Vehicle and a Mission Monitoring/Control System (MMCS) unit to teleoperate/supervise the vehicle. Thus, the complete platform will allow researchers to conduct different types of experiments both in teleoperated and autonomous modes, when completed.

*Corresponding author

Although the several research challenges in underwater robotics are sufficient to justify such a project, the initial motivation was the need to experimentally validate the work of Rocha [5], where a methodology to develop motion planning strategies for UUV was proposed. The methodology was based on Screw Theory and related tools [6], concerning different UUV configurations and possible cooperation between them. To sistematically implement the motion planning strategies, computational frameworks were developed [7]. These frameworks formed the KAST library (Kinematic Analysis by Screw Theory). It was tested in computer simulations, where its functionality and flexibility were verified. However, the lack of an experimental platform restrained the analysis of the KAST library performance in a real scenario.

A design process was made to identify the requirements and feasible conceptual models for an UUV aimed for experimentation in underwater robotics [8]. Further analysis indicated that a supervisory/operation interface to interact with the UUV would be useful to test the vehicle in teleoperation and to gradually implement autonomous features, including motion planning strategies. A study of a human-machine interface was then conducted, resulting in the implementation of a software prototype, which would be used in an UUV to be build [9].

These previous work led to the present project, where a broader vision for an experimental platform was devised, concerning a MMCS with open software architecture and a basic model of an UUV. This could be evolved into more complex models, but always maintaning its open hardware/software architecture characteristic. Cooperation between UUV was of interest, so the vehicle should be of relative easy construction and low cost.

A workgroup was created, consisting of two researchers from the Automation and Systems Research Group of the Federal Institute of Education, Science and Technology of Rio Grande do Sul - Rio Grande Campus (IFRS-Rio Grande), undergraduate students from the Systems Analysis and Development course of the Federal University of Rio Grande (FURG) and high school (in technical courses) students from the Industrial Automation course of IFRS-Rio Grande.

The platform design followed a product development pro-

cess model. This will be introduced in the next section. The results of the design process initial phases will be analyzed afterwards. The following section will present the conceptual model adopted for the platform. Finally, some of the early results in the implementation of a prototype will be described, and perspectives of future work will be addressed in conclusion.

II. The Product Design Methodology

In order to sistematize the design process, the Integrated Process of Product Development (PRODIP) model was adopted. It was proposed by the Nucleus of Integrated Product Development of Federal University of Santa Catarina (NEDIP-UFSC). It combines the results of several research conducted by NEDIP in a reference model to systematize and to formalize the product development process [10]. The model was formalized and described initially in [11] and its structure is depicted in Fig. 1.

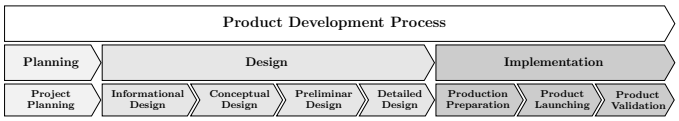

Figure 1. Graphic representation of the PRODIP model (adapted from [11])

The three macrophases in the PRODIP model are *Planning*, where the project plan is defined; *Design*, where a technical solution for the product and its manufacture is developed; and *Implementation*, where the design solution is produced and validated.

Since the platform is not to be manufactured in an industrial production line, the *Design* macrophase was the main concern in the design process. It is composed by four phases: *informational design*, *conceptual design*, *preliminar design* and *detailed design*.

Informational design is the initial phase of the design process, where the product features must be identified and understood. It is formed by a series of activities which aim to produce design specifications from the user needs and desires for the product [12].

Conceptual design uses the informational design results to create product conceptions and to select the one that best complies with the project demands. To do so, the product functional structure is detailed and from this, solution principles are generated. The possible combinations of these solutions generate the conceptions that are compared to determine the one that best attends the system requirements and other desired objectives [10].

Preliminar design is where the final product layout is produced from the results produced in the conceptual design. This layout comprises detailing of form, dimensions, specifications and behaviors. Eventually, virtual models and propotypes are used to this end. The results of these activities are analyzed, in order to optimize the proposed solution [10].

Detailed design is the final design phase, where the product solution is revised and further detailed. Usually, the final product documentation is the result of this phase. When considering industrial production, this also comprises financial revisions, production manuals and other documentation necessary to implement the production line [11].

The next sections describe the results of the informational and conceptual design phases, which are currently completed, in order to explain the platform conceptual model adopted in this project.

III. Informational Design Phase

To start the design process, a planning activity was executed to clearly define the design problem. It consisted in a survey of the state of the art of underwater systems used in industry and research. Applications and usual features were identified from the industrial systems survey. The research platform survey provided insights in the construction process, possible architectures, desired features and enviroments where experimentation occurs.

Among the commercial/industrial systems, the Cybernetix ALIVE [13] autonomous intervention system, the Seabotix [14] and the Videoray [15] mini inspection Remotely Operated Vehicles (ROV) were studied.

In the research/academic platform survey, different UUV configurations were identified, according to the type of research. The MIT SeaPerch [16] was created as a introductory platform for ROV building, where the low-cost and assembly simplicity were the primary concern. The Cornell University AUV (CUAUV) [17] is an autonomous vehicle used in academic competitions, where several different sensors are used to navigation in structured environments as tanks or pools, being a relatively complex example of academic platform. The SAUVIM [18] and the RAUVI [19] intervention systems have manipulators attached to the vehicle in order to execute intervention tasks, both being used to develop autonomy features to this kind of UUV, which is usually teleoperated due to the complexity of interaction between the UUV/manipulators and the environment [20]. Regarding brazilian projects, the pioneering works of Hsu et al. [21] and Barros/Soares [22] were considered, and recent projects such as the Pirajuba Autonomous Underwater Vehicle (AUV) [23] and the ROVFURG [24], [25] were analyzed. The Pirajuba is used to study control and autonomy of survey systems, while the ROVFURG is an UUV which is gradually evolving from a pure teleoperated device to an autonomous vehicle, using an open architecture architecture.

The survey led to the design problem definition: An UUV system similar to the ROV (Remotely Operated Vehicles) used in the offshore oil industry, composed by a vehicle and a base station. The vehicle should operate in low depths (max. 20m) in low current environments (such as tanks or ponds). A prototype should be built to study the underwater vehicle construction process and its inherent problems. Initially, the vehicle should be teleoperated, in order to study its kinematic/dynamic behavior. As the project evolves, and experience is acquired, autonomy features will be gradually implemented, using the theoretical results previously obtained.

Underwater robotics, control and automation researchers were considered the primary potential users of the platform. Since the platform would initially be used for a small group,

it was assumed that the production would be limited. So, industrial production, marketing and commercial aspects were not considered. Assembly, maintenance, transportability, usability and storage were the aspects taken into account. Also, user requirements were produced from the needs identified in literature for such systems.

Design tools such as sort matrices and quality function deployment method were used to combine and to ponder these information [8]. This resulted in the following design specifications:

- *Relative low cost*: despite similarities with industrial systems, the platform should be affordable to research groups with limited funds. Cooperation between UUV is of interest, so more than one vehicle could be built, which means that costs should be as low as possible. This specification takes precedence over the others, even if it means simplification or exclusion of some desired features;
- *Open/extensible architecture*: hardware and software features must be fully accessible to researchers. This means that they should be able to adapt/modify the platform to comply with their experiments. New sensors and actuators could be added to the existing hardware, as manipulators, for instance. Software should be easily extended to deal with the hardware modifications and also modifiable to implement new features to be experimented. A communication interface could be used to interact with other software;
- *Basic sensor set*: unlike other low cost platforms, the vehicle should have a basic set of sensors to estimate its attitude and depth. Also, it should have enough processing power to deal with sensor data and images provided by a camera. This would be useful to implement experiments in control (such as automatic attitude/depth) and localization using pattern recognition;
- *Full mobility*: independent motion in six degrees-of-freedom (DOF) is a commmon feature of industrial systems (usually ROV). So, the vehicle should be able to translate and rotate in the three spatial directions, as shown in Fig. 2. Roll and pitch are usually limited, being mostly used to level the UUV when it is not balanced. This could happen when manipulators are used in intervention systems [26];
- *Portable/Configurable MMCS*: The monitoring/control system should be easily transported into the experimentation sites. So, a portable hardware with enough power source is required. The MMCS software could be executed in a laptop or some dedicated station, customized to teleoperation. Regarding this software, it should be portable, extensible, and its GUI (Graphical User Interface) must have easy configurability to be adapted for the experiment needs.

It must be observed that the six DOF motion for the platform was adopted in order to establish similarity to industrial intervention/inspection systems, where autonomy is considered an open problem [20]. This differs from the commonly format of survey/cruise UUV, which are designed to be propelled in one direction, while motion in other directions are obtained by the use of surface controls (rudders or fins). This kind of UUV is beyond the scope of this project, although it is the configuration of most AUV (Autonomous Underwater Vehicle) [26].

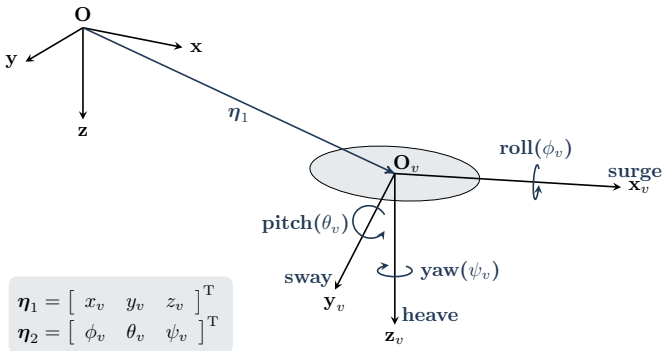

Figure 2. UUV motions in six DOF (adapted from [5])

IV. Conceptual Design Phase

Following the informational design phase conclusion, the conceptual design activities aimed to produce a conceptual model that satisfied the design specifications previously obtained. This design phase was composed of analysis and synthesis activities. In the first, the design problem was divided into smaller ones and partial possible solutions for each subproblem were enumerated. In the latter, these partial solutions were combined to produce different design conceptions, and one was chosen by evaluating their compliance to the specifications and other aspects such as assembly, maintenance and knowledge base availability.

In the analysis activity, the design problem was decomposed into the UUV systems and subsystems analyzed by Rocha [5], as depicted in Fig. 3. For each subsystem, different solutions were listed, and the most viable ones were maintaned. Among the criteria used to filter the viable partial solutions were cost, avaliability and knowledge/experience. In some cases, these partial solutions were tested by the research group. In other cases, it was observed the development of similar projects.

To illustrate this activity, for the vehicle processing hardware (in Systems Architecture subsystem) were considered PC compatible motherboards (netbook and industrial ones), Android tablet, Arduino [27], Raspberry Pi [28] and Beagleboard [29]. From these, the Android tablet (due to the difficulty of interfacing with another hardware) and the Arduino (because of its very low processing power) were discarted.

A set of product conceptions was produced in the synthesis activity, combining the different partial solutions for each subsystem. Using a Pugh matrix [8], these conceptions were evaluated against each other considering several comparison criteria, weighted according to their relevance to the design specifications. From this process, it was selected the conception that was most adequated to the design specifications and the user requirements. The conception choices are briefly presented in the following.

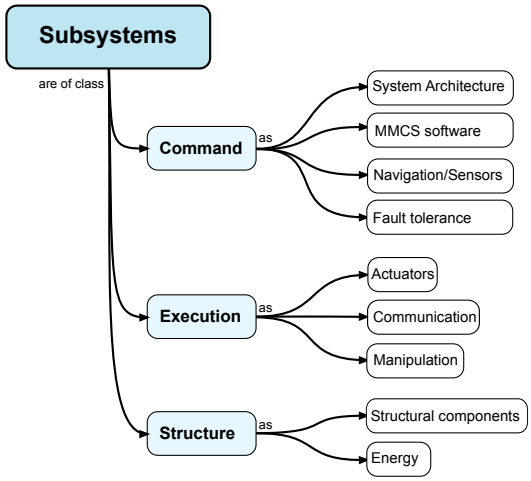

Figure 3. UUV subsystems (adapted from [5])

- For the *structural components* subsystem, composed by the structural frame and vessels to hold the vehicle electronics, PVC pipes and joints were the selected materials. An acrylic glass dome is to be used for the camera part in the electronics holdings. These can be easily found in construction material stores and have low cost compared to other candidates such as aluminum frames;
- Energy is to be provided by an external power source (a computer or surveillance systems one). A shielded flexible cable will be used to supply energy (12VDC) to the UUV;
- The propulsion system will be composed by bilge pumps adapted as thrusters (with helices instead of the pump system). They are designed to operate underwater and have low cost. Also, they are easy to install and to vary its speed using PWM (Pulse Width Modulation) drivers;
- Raspberry Pi will be used as processing hardware for the vehicle. These boards have excellent benefit-cost ratio and are open source, which is of essence to the project open architecture characteristic. Arduino boards will be used to acquire sensor information and to command actuators;
- Communication will use Ethernet cables. Since the processing hardware support TCP/IP networking, this protocol will be used to connect the vehicle to the MMCS station. So, the tether (or umbilical cable) will be composed of an shielded power cable and an Ethernet network cable tied together;
- The navigation subsystem will use MEMS (Micro-Electronic-Mechanical System) sensors. An IMU (Inertial Measuring System) will provide attitude information, while an absolute pressure sensor will determine the vehicle depth. A 5MPixel CCD camera designed for the Raspberry Pi will be used to capture images;
- For the fault tolerance system, internal sensors will detect humidity/leakages and measure internal temperature. In case of abnormal functioning, the vehicle can be automatically deactivated. Since it will have slightly positive bouyancy, it is expected that it will surface in order to be recovered;
- Python [30] was chosen as the development platform for the MMCS and the vehicle software. This was due to its open source/object-oriented/scientific nature. Also, Python has a large knowledge base composed by literature, web pages and user groups. Finally, it has several different libraries useful for scientific computation and is multiplataform. C++ will be also used for Arduino programming.

V. The Platform Conceptual Model

A general view of the platform hardware structure is depicted in Fig. 4. It is formed by two distinct units, the vehicle and the MMCS, which are connected by an Ethernet network cable. Both units are detailed in the following.

A. The Unmanned Underwater Vehicle

The UUV hardware is composed by the *actuator*, the *sensor* and the *processing* blocks.

The actuator block is formed by thrusters and their drivers. As previously stated, bilge pumps are modified by replacing their pump parts for helices to serve as thrusters. These are driven by an Arduino Nano board [27] connected to H-bridges. Up to six thrusters can be driven by the board, having their speed defined by the Arduino Nano PWM features. L298N integrated circuits are used as H-Bridges. The Arduino board also reads leakage and temperature sensors in order to monitor the electronics operational condition.

The sensor block uses another Arduino Nano board to process sensor information. A MEMS IMU composed by three-axis accelerometer, gyroscope and compass generate data that is processed by a sensor fusion software running in the Arduino (such as the FreeIMU [31]) to produce attitude and heading information. An absolute pressure MEMS sensor is used to determine depth, since there is a linear relationship between this information and pressure (it increases 1atm for each 10m of depth). The sensor maximum reading is 700 KPa, which is far beyond the pressure in the UUV maximum operation depth.

A Raspberry Pi [28] is the processing block CPU. It is connected to both Arduino boards by USB cables. This unit sends commands to the driver board to activate the thrusters in the way that the desired motion is produced. It also receives information from the sensor block preprocessed by its Arduino board, which is used to determine its attitude/heading/depth. The processing block communicates with the MMCS unit by the Ethernet network embedded in the Raspberry Pi. Sensor information are sent and high-level commands are received by this block. Also, there is enough processing power to implement further features, such as automatic depth or attitude control, following objects, etc.

Although is could be considered part of the sensor block, the 5MPx camera is directly connected to the Raspberry Pi GPU (Graphical Processing Unit), which provides fast

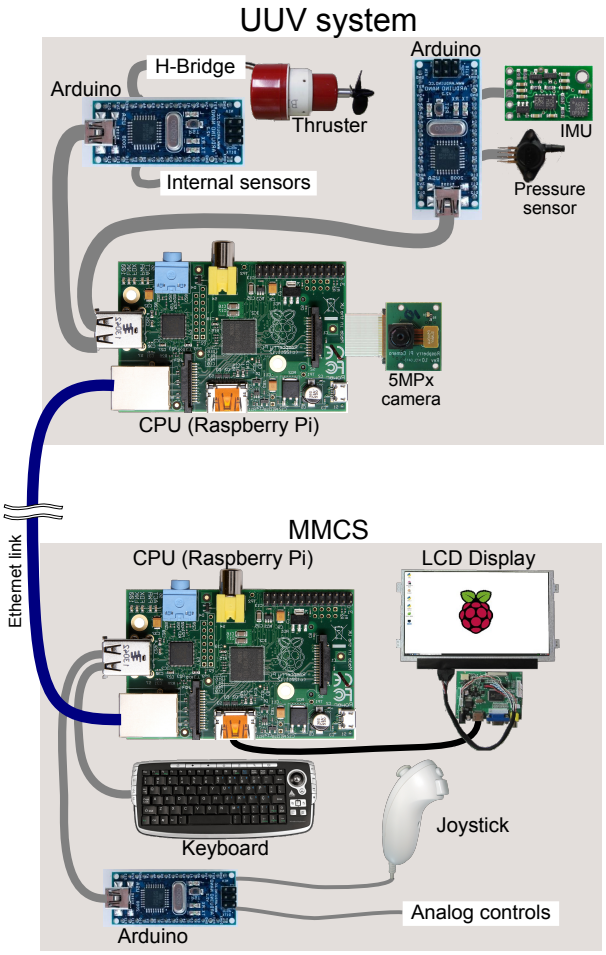

Figure 4. General view of the platform hardware components

video processing (up to 30 fps - frames per second) in 720p resolution.

The sensor and processing blocks will be housed into a 110mm diameter PVC pipe, where one end will have an acrylic glass dome for camera visualization. The actuator block will be housed into another 110mm diameter PVC pipe, in order to better distribute the load and to minimize electrical noise to the processing block.

The frame structure is still being analyzed. It should have avaliable space to future additions to the platform, such as a manipulator (which is a feature essential to intervention task research), and to hold some cargo. Also, there is the concern to the thruster disposition to be taken into account. To this moment, the best suited geometric arrangement to the platform characteristics is similar to the one shown in Fig. 5.

B. The Mission Monitoring/Control System

The MMCS is essentially a processing unit where the monitoring/control software will run. Since it usually will be used near the experiment sites, it would be of interest to use a portable device, such as a laptop running Linux, Windows or MacOS operating systems.

To increase the operator interaction with the platform, an *analog module* was devised to complement the key-board/mouse interface. It consists of an Arduino Nano board which processes information from a joystick and other analog components, as potentiometers. The joystick has a three axis accelerometer in order to define its attitude, along with a 2-axis analog command and 2 digital buttons. The module also has status leds to indicate operational conditions/faults.

Although this module could be easily attached to a computer by its USB interface, it was noted that a portable dedicated unit would be easier to handle. So, the MMCS portable unit shown in Fig. 4 was created. The CPU used is a Raspberry Pi, where a integrated keyboad/trackball and the analog module are connected using the USB interface. A 10-inch, 1200x800 pixel LCD display is connected by the HDMI port, providing high-quality visualization. This portable unit is encased in an adapted rigid plastic toolbox. The Raspberry Pi network interface will provide the data connection with the UUV.

C. The Platform Software

As previously stated, Python was the software development platform chosen. Due to its multiplataform and open source characteristics, it best suited the platform needs. Also, its large knowledge base facilitates its learning and use.

In order to meet the platform requirements of modifiability, extensibility and adaptation to the experiments, an object-oriented approach was adopted to the software modeling process. Since Python is a object-oriented language, implementing the modelled software would be a straightforward process. Several interfaces were defined, and concrete classes were also defined to implement the functionalities according to specific hardware components behaviors. The class diagram in Fig. 6 shows the basic structure from where the software must be developed and evolved.

Since a TCP/IP network stack was established as the communication medium between the UUV and the MMCS, a socket interface was chosen to exchange messages between the components of the two softwares. These will be most textual, although camera images will also be transmitted this way. This communication complexity will be encapsulated into the data provider components represented by the InputDevice and OutputDevice interfaces present in the class diagram. Sensors, actuators and camera access classes descend from these interfaces.

For the graphical user interface, different kind of widgets were also included in the software modeling. These graphical components will be linked to the data providers in order to

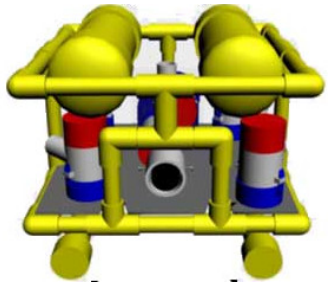

Figure 5. Seafox ROV frame [32]

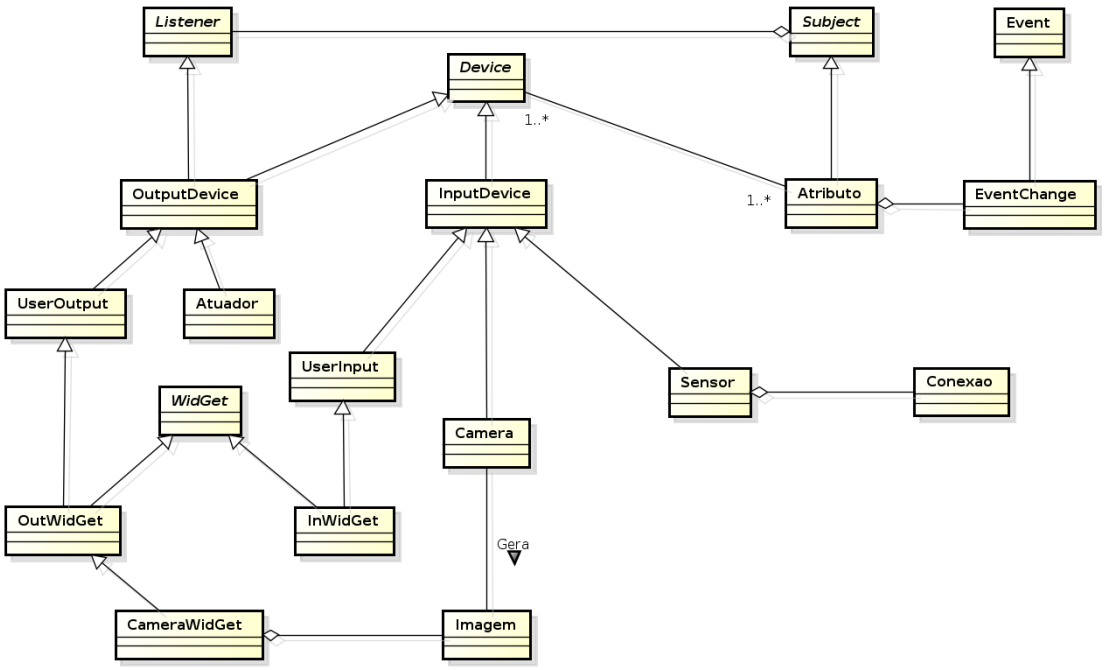

Figure 6. Platform software class diagram

display information in an intuitive way to the UUV operator. The Pyside library is the base for the GUI development, which rely on the Qt cross-platform application framework [33]. This complies with the multiplataform feature desired for the software. Also, the Qt already implemented widgets attend several user interface needs and turn easier to implement new widgets from them.

VI. Early Implementation Results

To this moment, the prototype construction is in its initial stage, and the research team is focusing in the development of the UUV actuator block, the frame construction and in the implementation of the MMCS software and the portable unit.

Multiplatform tests of the user interface are being conducted, in Windows and Linux computers and in a Raspberry Pi running the Raspbian Linux distribution. From the initial implementations, it is observed that it is possible to run the MMCS software without modification in all tested hardware. The initial GUI implementation is illustrated in Fig. 7. This is a screenshot of the tests in the Raspberry Pi hardware.

In this figure, it is possible to observe some of the widgets implemented, such as textual displays, gauges, status indicators, compass and attitude indicator (or artificial horizon). Most of them were implemented based on the QWT library [34], which is being used to complement Pyside.

A camera visualization widget, not shown in the figure is being finalized. The Raspberry Pi of the vehicle uses the TCP/IP network to stream the camera images to the MMCS processing unit. The main problem of this streaming is to obtain a balance between the image quality and the lowest latency possible. Also, the OpenCV library is being tested to process the camera images to identify objects in the environment [35].

The Pyro library [36] is being used in to implement the communication between the MMCS and the vehicle. This library uses a remote objects interface to encapsulate the network communication complexity. This allows to use the data-aware classes created in the KAST framework to deal with the sensor data(in the UUV) - widgets(in the MMCS) relationship. These classes implement the listener/subject design pattern to automatically notify the software components(widgets) when an event occurs in an object of interest(sensor data providers), which could be as simple as a change of value in an analog sensor. So, the display components can be automatically updated.

Several isolated implementations are being made to create and to use the different sensor/actuators in the platform. The analog command module to be used in the MMCS is functional, although it was tested only in computers, for the moment. The IMU and the pressure sensors are being tested and calibrated, along with the implementation of the components to encapsulate their use to the UUV software.

VII. Conclusion

The present paper presented the design aspects of a platform for experimentation in underwater robotics, which is composed by an unmanned underwater vehicle and a mission monitoring/control system unit. The main requisites of the project identified were the relative low cost and an open software/hardware architecture. In the conceptual model, this was achieved by adopting open hardware/software and materials found in any construction shop. In particular, the software development platform chosen was mainly Python and its scientific libraries, although C++ was also of use due to the hardware design choices.

This project is a work in progress. Following the conceptual design phase, a prototype is being constructed. Initial tests of

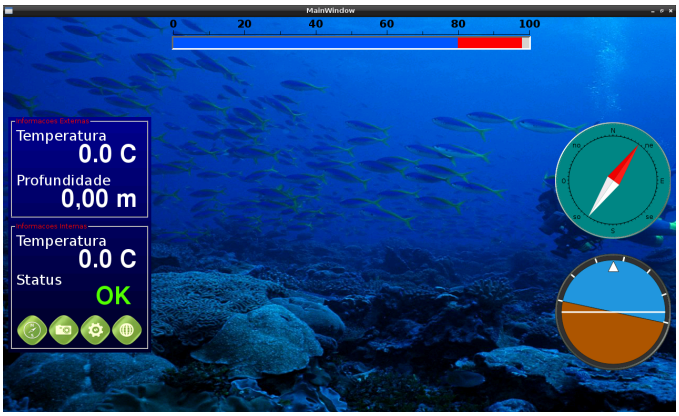

Figure 7. MMCS graphical user interface

the vehicle and its telecommanded operation are planned for the first semester of 2014. From these tests, the conceptual model and the prototype behavior will be analyzed for further improvements. When finished, the main contribution of this project will be to provide an platform for experimental underwater robotics research which, although being a low cost one, has the essential features that are present in its production counterparts. Also, the open source characteristic of the project will ease the implementation of the experimenst and further development of the plaftorm.

Several future work are devised from this initial platform. These can be of two types. The first one is to add features similar to current production systems, such as an underwater robotic manipulator, adoption of different sensors (e.g. Doppler Velocity Log) and propulsion systems. The second type concerns the gradual implementation of autonomy features, where a kinematics/dynamics model is needed (using identification methods), along with automatic attitude/depth control and localization using sensor fusion/computational vision.

This platform will also be of use to experimentally validate the screw-based motion planning methodology, which was a major motivation for the development of this project. In this case, two or more UUV may be used to cooperative task execution.

ACKNOWLEDGMENT

The authors thank the partial support granted by the Federal Institute of Education, Science and Technology of Rio Grande do Sul.

REFERENCES

[1] R. W. Button, J. Kamp, T. Curtin, and J. Dryden, *A Survey of Missions for Unmanned Undersea Vehicles*. Arlington: RAND Corporation, 2009.

[2] L. L. Whitcomb, "Underwater Robotics: Out of The Research Laboratory and Into The Field," in *Proceedings of ICRA '00*, vol. 1. San Francisco: IEEE, Abril 2000, pp. 709–716.

[3] Petrobras. (2011) Cada Vez Mais Fundo. [Online]. Available: http://www.petrobras.com.br/minisite/presal/pt/cada-vez-mais-fundo

[4] ——, "Inovar Para Crescer: Entrevista com Gerente Executivo do CENPES, Carlos Tadeu Fraga," *Petrobras - Fatos e Dados*, vol. Online, 2011. [Online]. Available: http://fatosedados.blogspetrobras.com.br/2011/05/06/inovar-para-crescer-entrevista-com-gerente-executivo-do-. cenpes-carlos-tadeu-fraga

[5] C. R. Rocha, "Motion planning of underwater intervention robotic systems based on screw theory - planejamento de movimento de sistemas robóticos de intervenção subaquática baseado na teoria de helicoides," PhD Thesis, Federal University of Santa Catarina, Florianópolis(SC), Brazil, 2012.

[6] C. R. Rocha, C. P. Tonetto, and A. Dias, "Kinematic analysis of cooperative underwater intervention systems based on screw theory," in *Proceedings of the 13th Mechatronics Forum International Conference*, vol. 3, Johannes Kepler University. Linz: Trauner-Verlag, 2012, pp. 707–713.

[7] C. R. Rocha, C. Tonetto, and A. Dias, "A framework for kinematic modeling of cooperative robotic systems based on screw theory," in *Proceedings of the 21th International Congress of Mechanical Engineering - COBEM 2011*. Natal: Universidade Federal do Rio Grande do Norte, 2011.

[8] B. L. Floriani, A. Dias, and C. R. Rocha, "Methodological aspects in the informational and conceptual design of a remotely operated underwater vehicle - aspectos metodológicos no projeto informacional e conceitual de um veículo remotamente operado subaquático," in *Anais do VI Congresso Luso-Moçambicano de Engenharia - CLME 2011*, IMechE. Maputo: Edições Inegi, 2011.

[9] D. A. Santos, "Human-machine interface for supervision and operation of remotely operated underwater vehicles - interface homem-máquina para supervisão e operação de veículos subaquáticos remotamente operados," Monography, Universidade Federal do Rio Grande, Rio Grande(RS), Brazil, 2013.

[10] N. Back, A. Ogliari, A. Dias, and J. C. Silva, *Integrated Design of Products: Planning, Conception and Modeling - Projeto Integrado de Produtos: Planejamento, Concepção e Modelagem*. São Paulo: Manole, 2008.

[11] L. N. Romano, "Reference model for the development process of agricultural machinery - modelo de referência para o processo de desenvolvimento de máquinas agrícolas," PhD Thesis, Federal University of Santa Catarina, Florianópolis, 2003.

[12] A. J. H. Fonseca, "Systematization of the design specification process of industrial products and its computational implementation - sistematização do processo de obtenção das especificações de projeto de produtos industriais e sua implementação computacional," PhD Thesis, Federal University of Santa Catarina, Florianópolis, 2000.

[13] Cybernetix. (2008) Présentation - ingénirie des systèmes robotiques - Cybernetix. [Online]. Available: http://www.cybernetix.fr/Presentation,90

[14] Seabotix. (2013) Seabotix inc. [Online]. Available: http://www.seabotix.com

[15] Videoray. (2013) Videoray products. [Online]. Available: http://www.videoray.com/homepage.html

[16] Sea Perch. (2011) Sea perch program. [Online]. Available: http://seaperch.mit.edu

[17] CUAUV team. (2010) Cornell university autonomous underwater vehicle. [Online]. Available: http://cuauv.ece.cornell.edu/

[18] G. Marani, S. Choi, and J. Yuh, "Underwater autonomous manipulation for intervention missions auvs," *Ocean Engineering*, vol. 36, no. 1, pp. 15–23, 2009.

[19] P. Ridao, P. J. Sanz, and J. Oliver. (2011) Reconfigurable auv for intervention. [Online]. Available: http://www.irs.uji.es/rauvi/abouttheproject.html

[20] G. Antonelli, *Underwater Robots: Motion and Force Control of Vehicle-manipulator Systems*, 2nd ed., ser. Springer Tracts in Advanced Robotics, B. Siciliano, Ed. Springer, 2006, vol. 2.

[21] L. Hsu, R. Costa, F. Lizarralde, and J. da Cunha, "Avaliação experimental da modelagem e simulação da dinâmica de um veículo submarino de operação remota," *Revista Controle e Automação*, vol. 11, no. 2, pp. 82–93, 2000.

[22] E. A. Barros and F. J. A. Soares, "Desenvolvimento de um robô submarino de baixo custo," in *Proceedings of the 2002 Brazilian Congress of Automatica*. Natal: Federal University of Rio Grande do Norte, 2002.

[23] J. L. D. Dantas, W. d. S. Caetano, R. T. d. S. Vale, L. M. de Oliveira, A. R. M. Pinheiro, and E. A. de Barros, "Experimental research on underwater vehicle manoeuvrability using the auv pirajuba," in *Pro-*

ceedings of the 22th International Congress of Mechanical Engineering - COBEM 2013. Ribeirão Preto: Universidade do Estado de São Paulo, 2013.

[24] M. L. Centeno, "Rovfurg-II: design and construction of a low-cost unmanned underwater vehicle - Rovfurg-II: projeto e construção de um veículo subaquático não tripulado de baixo custo," Master Thesis, Federal University of Rio Grande, Rio Grande, 2007.

[25] V. Kuhn, "Automatic control of an underwater inspection vehicle using low-cost sensing - controle automático de um veículo de inspeção subaquática utilizando sensoriamento de baixo custo," Master Thesis, Federal University of Rio Grande, Rio Grande, 2011.

[26] G. Antonelli, T. Fossen, and D. Yoerger, "Underwater robotics," in *Springer Handbook of Robotics*, ser. Springer Tracts in Advanced Robotics, B. Siciliano and O. Khatib, Eds. Heidelberg: Springer, 2008, ch. 43, pp. 987–1008.

[27] Arduino team. (2013) Arduino homepage. [Online]. Available: http://arduino.cc

[28] Raspberry Pi Foundation. (2013) Raspberry pi - an arm gnu/linux box for $25. [Online]. Available: http://www.raspberrypi.org

[29] Beagleboard Foundation. (2013) Beagleboard.org - community open source hardware computers. [Online]. Available: http://beagleboard.org

[30] Python.org. (2011) Python programming language - official website. Python Software Foundation. [Online]. Available: http://www.python.org

[31] F. Varesano, "Freeimu: An open hardware framework for orientation and motion sensing," *arXiv preprint arXiv:1303.4949*, 2013.

[32] S. Thone. (2009) Homebuilt ROVs. [Online]. Available: http://www.homebuiltrovs.com/

[33] Digia. (2013) Language bindings: Pyside - qt wiki - qt project. Digia Plc. [Online]. Available: http://qt-project.org/wiki/PySide

[34] U. Rathman and J. Wilgen. (2013) Qwt - qt widgets for technical applications. [Online]. Available: http://qwt.sourceforge.net

[35] J. Howse, *OpenCV Computer Vision with Python*. Packt Publishing, 2013.

[36] I. de Jong. (2014) Pyro - python remote objects. [Online]. Available: http://pythonhosted.org/Pyro4