UNIVERSITAT
JAUME•I

ESCOLA DE DOCTORAT

PH.D. THESIS IN COMPUTER SCIENCE

Towards a multimodal interface for the specification of intervention tasks in underwater robotics

Author
Juan Carlos García Sánchez

Directors
Prof. Pedro J. Sanz
Prof. Paulo Menezes

Castellón de la Plana, Spain.

September 2021

UNIVERSITAT
JAUME•I

ESCOLA DE DOCTORAT

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

PH.D. THESIS IN COMPUTER SCIENCE

Towards a multimodal interface for the specification of intervention tasks in underwater robotics

Author
Juan Carlos García
Interactive & Robotic Systems Lab
Department of Computer Science and Engineering
Universitat Jaume I, Spain

Director
Prof. Pedro J. Sanz
Interactive & Robotic Systems Lab
Department of Computer Science and Engineering
Universitat Jaume I, Spain

Director
Prof. Paulo Menezes
Institute of Systems and Robotics
Department of Electrical and Computer Engineering
Universidade de Coimbra, Portugal

Juan Carlos G.S.   Paulo Menezes

Castellón de la Plana, Spain.   September 2021

Public funding

This thesis has been carried out thanks to institutions like Universitat Jaume I (UJI), University of Coimbra (UC) or Spanish Ministry of Economy and Competitiveness (under grant DPI2011-27977-C03, TRITON Project).

Acknowledgements

This Ph.D. thesis would not have been possible without the help and support of special people around me. My first grateful words are for my Ph.D. directors, Prof. Pedro J. Sanz and Prof. Paulo Menezes. I could not imagine during the first meeting held in Prof. Sanz's office when he offered me the opportunity to collaborate with the lab testing the RoboNova and participating in the CEABOT contest, that today I would be writing these lines. Since then, I have learned a great deal about grasping, humanoid robots, Graphical User Interfaces and underwater robots. He also gave me the opportunity to make a research stay at the Institute of Systems and Robotics, where I met Prof. Menezes. My relationship with him during these years has been exceptional, in both professional and personal terms. I will never forget some of the big moments and lessons. I would like to publicly thank you for both the support and the suggestions during the development of this Ph.D.

I can not forget my friends and workmates at Robotic Intelligence Lab and Interactive & Robotic Systems Lab, both at Universitat Jaume I, from faculty people to researchers. There are a lot of experiences spanning almost 10 years that I will never forget: the working atmosphere, the collaboration between all of us, the way we work side-by-side... My best wishes to all of you in the future. I would like to mention especially Oscar, one of the most relevant people in my life since our childhood. Your advice, suggestions and support during these years were essential.

There are no words to express my gratitude to my parents Eugenio & Aludia, my sister Alicia and my wife Silvia. Thanks to my parents for the great effort in providing me and my sister with higher education. Thank you to all for your continuous support, words of encouragement, patience and understanding, for being there in each of the difficult moments that I have had since I started this thesis. Without a doubt, this work could not have been done without you.

Resumen

En la actualidad, los robots desempeñan un papel fundamental en nuestra vida cotidiana, realizando tareas tan diversas como mantenimiento, vigilancia, exploración en entornos hostiles u operaciones de búsqueda y rescate. Sin embargo, allí donde más se necesitan es en entornos peligrosos, como zonas de radiación, entornos submarinos, zonas de guerra y misiones espaciales. De todos estos entornos, el submarino es uno de los que más ha aumentado su actividad en los últimos tiempos, debido principalmente a las industrias de gas y petróleo, acciones de búsqueda y recuperación/rescate o la arqueología e investigación oceanográfica.

Respecto a los tipos de robots utilizados en entornos submarinos, nos encontramos tres tipos: robots operados remotamente (Remotely Operated Vehicle (ROV)), robots autónomos (Autonomous Underwater Vehicle (AUV)) y robots híbridos (Hybrid-ROV (HROV)). La decisión de utilizar un tipo de robot u otro suele venir dada por el tipo y la complejidad de la misión a realizar. Sin embargo, existe un problema común a los tres: la interacción hombre-robot presenta diversas deficiencias en cada uno de ellos. Además, no hay que olvidar que el usuario experto sigue jugando un papel central desde el punto de vista de la toma de decisiones.

Basado en el contexto de los últimos proyectos de investigación coordinados por nuestro laboratorio de investigación (Interactive and Robotic Systems Lab, de la Universitat Jaume I), la presente tesis está centrada en la investigación en diferentes aspectos relacionados con la interacción hombre-robot: el uso de algoritmos para ayudar al usuario durante la especificación de la misión (haciendo que la interfaz de usuario sea fácil de usar y sea capaz de asistir al propio usuario), la exploración de una interfaz multimodal (mediante el uso de diferentes dispositivos para interactuar con la interfaz) y la propuesta de una arquitectura de control del robot (permitiendo cambiar el modo operacional, desde autónomo a teleoperado, o viceversa).

Abstract

Today, robots play a fundamental role in our daily lives, performing tasks as diverse as maintenance, surveillance, exploration in harsh environments or Search and Recovery/Rescue (SAR) operations. However, they are most needed in hazardous environments, such as radiation zones, underwater environments, war zones and space missions. In all these environments, the submarine is the one that has increased its activity the most in recent times, mainly due to the oil and gas industries, SAR operations or archaeology and oceanographic research.

Regarding the types of robots used in underwater environment, there are three types: ROV, AUV and HROV. The decision to use one type of robot or another is usually determined by the type and complexity of the mission to be carried out. However, there is a problem common to all three: Human-Robot Interaction (HRI) has several deficiencies in each kind of robot. Furthermore, it should not be forgotten that the expert user continues to play a central role from the point of view of decision-making.

Based on the context of the latest research projects coordinated by our research laboratory (Interactive and Robotic Systems Lab, at the Universitat Jaume I), this thesis is focused on research in different aspects related to HRI: the use of algorithms to assist the user during mission specification (making the user interface easy to use and capable of assisting the user himself), exploring a multimodal interface (by using different devices to interact with the interface) and the proposal of a robot control architecture (allowing the user to change the operational mode, from autonomous to teleoperated, or vice-versa).

Contents

List of Figures

List of Acronyms

AAL Architecture Abstraction Layer

AI Artificial Intelligence

ALIVE Autonomous Light Intervention VEhicle

ASC Autonomous Surface Craft

AUV Autonomous Underwater Vehicle

CSDCS Continuous State Dynamic Control System

DES Discrete Event System

DIY Do It Yourself

DOF Degrees Of Freedom

DVECS Distributed Virtual Environment Collaborative Simulator

DVL Doppler Velocity Log

DP Dynamic Position

FOG Fiber Optical Gyroscope

GUI Graphical User Interface

HCI Human-Computer Interaction

HMD Head-Mounted Display

HMI Human-Machine Interaction

HRI Human-Robot Interaction

HROV Hybrid-ROV

HUD Head-Up Display

I-AUV Autonomous Underwater Vehicles for Intervention

IEO Instituto Español de Oceanografía

IMU Inertial Measurement Unit

IoT Internet of Things

IRSLab Interactive & Robotic Systems Lab

ISR Instituto de Sistemas e Robótica

ITOCA Intelligent Task-Oriented Control Architecture

LIBS Laser Induced Breakdown Spectroscopy

LOA Level Of Autonomy

MBARI Monterey Bay Acquarium Research Institute

MCL Mission Control Language

MCS Mission Control System

ML Machine Learning

MRDS Microsoft Robotics Developer Studio
MRU Motion Reference Unit
NUI Natural User Interface
OGRE Object-Oriented Graphics Rendering Engine
OSG Open Scene Graph
POV Point Of View
PTU Pan/Tilt Unit
RSM Robot Safety Measures
ROS Robot Operating System
ROV Remotely Operated Vehicle
SAR Search and Recovery/Rescue
SDL Simple DirectMedia Layer
SRI Stanford Research Institute
TMS Tether Management System
ToI Target of Interest
UC University of Coimbra
UdG University of Girona
UI User Interface
UIAL User Interface Abstraction Layer
UIB University of Balearic Islands
UJI Universitat Jaume I
URDF Universal Robotic Description Format
USBL Ultra-Short Base Line
UUV Unmanned Underwater Vehicles
UV Underwater Vehicle
UWUI UnderWater User Interface
UWSim UnderWater Simulator
UX User eXperience
VC Virtual Cockpit
VR Virtual Reality
VRPN Virtual Reality Peripheral Network
WHOI Woods Hole Oceanographic Institution

Chapter 1
Introduction

1.1 Introduction

Despite the big evolution of teleoperated robots, which started at the end of the 1940s, there are still some situations where these robots fail during task execution. Many of such failures are not associated only with the lack of technology, e.g. limitations of the robotic platform, but also with human factors.

The Fukushima nuclear power plant disaster in 2011 had a strong impact on the international community and, in particular, on the vision of the ways that robots should operate in this kind of new challenging missions. Inspired by this terrible accident, a lot of new activities have been started, like DARPA Robotics Challenge[1] in USA or the Eurathlon competition[2] in Europe. This and other hostile environments, as well as constraints on robot dimensions, life support unavailability, or other reasons that preclude the presence of humans, are making mandatory a new generation of intervention robotic systems able to performing the missions that, in other conditions, would be carried out by humans.

Another example of a hostile environment, can be found in underwater domains, due to the challenging conditions at certain depth, the water currents and low visibility. Aditionally, we should bear in mind that oceans represent 2/3 of the Earth‘s surface and the importance of the economical impact regarding off-shore industries. In the interest of better understanding the importance of using robotic devices for underwater applications, we list hereafter a few examples that can have real benefits from their use:

- Oil and gas industry: offshore platforms require the inspection and repairing of submerged infrastructures. These tasks need dexterous robots able to open/close a valve, follow pipes to detect leaks, etc.
- Search and Recovery: localization and grasping objects on the seafloor. One example of this application is the flight data recorder search and recovery of a crashed airplane.
- Deep water Archaeology: to document submerged sites, using high resolution 2D/3D seafloor mapping techniques.

[1] http://www.darpa.mil/program/darpa-robotics-challenge

[2] http://www.eurathlon.eu

- Science: periodic maintenance of underwater permanent observatories.

Traditionally, underwater intervention tasks have been made with *manned submersibles*, but the main concerns with this approach is the fact that it may place a human at risk. The robotic research and industries grabbed the concept of ground teleoperated vehicles to use them in underwater domains, appearing the ROV, equipped with one or more robotic arms to carry out the interventions. The control of the robotic arms may be used in a master/slave configuration, enabling the user to continuously control them during the intervention. The main drawbacks of this methodology are the accumulated fatigue of the user during long-term interventions, the high skills required by the users in order to control the arm, and the need of permanent user-arm communication for suitable control.

Concerning autonomous robots, we can distinguish AUV and Autonomous Underwater Vehicles for Intervention (I-AUV), which may be endowed with intervention capabilities based on the use of one or two arms. However, fully autonomous robots are still in their early development stages, mainly due to the difficulties of adapting these systems to changes in their operating environments in real time. In fact, two of the most critical tasks of these systems are to acquire information about the environment and wireless communication skills. Obviously, the technology evolution facilitates the integration of new sensors into the robot, providing more accurate information that can be used to enable the robot to act properly, despite all those changes. Under certain situations and specific tasks, an autonomous robot may operate better than a human user in terms of speed and precision, thus the user can be relieved from some of these tedious tasks, diminishing the induced fatigue.

The most recent innovation into underwater robots are those called HROV, which may operate either as autonomous or teleoperated mode, depending on the task or specific circumstances to be carried out. Nevertheless, some architectures do not allow the user to switch the operational mode from autonomous to teleoperated modes when the pilot deems it appropriate. In fact, most of the times, this conversion needs to take the robot out of the water and modify the structure to add or remove different types of equipment.

To sum up and regarding the aforementioned problems, the main purpose of this thesis is to develop and study new ways to improve the HRI. In order to achieve these purposes, this thesis is focused on two different research lines.

1.2 Aims and scope

As mentioned in the introduction, this thesis has different purposes related to HRI. Although those purposes are quite generic, they can be divided into more specific goals:

- Develop different algorithms, which can be used by the user to identify and select a ToI for the mission being carried out.
- Develop a GUI to be used with a ROV or a HROV, depending on the mission.
- Add multimodal capabilities to this GUI, allowing the user to use the most suitable input device to interact with the system.

- Include VR-related techniques to add immersive support for the user to get first person point of view during the mission execution.
- Define a new architecture to ensure a proper HRI.

Although this thesis is focused on underwater robots, most of the results obtained here can be applied to other domains with similar problems, such as aerial or spatial robots. Although the wireless communications context plays an important role on this robotic area, the focus of this thesis will be concentrated in the HRI problems, leaving communications out of the scope of this research.

1.3 Context

This thesis has been carried out mainly in the Interactive & Robotic Systems Lab (IRSLab) at UJI (Castellón, Spain), taking advantage from a collaboration with the Instituto de Sistemas e Robótica (ISR) at UC (Coimbra, Portugal).

Since 2009, the IRSLab has been coordinating different research projects regarding underwater robotics and this thesis is related to some work packages assumed on these projects. This is a brief summary of these projects:

- RAUVI: *Reconfigurable AUV for Intervention*. Spanish research project (DPI2008-06548-C03). The main goal of the RAUVI project was to develop and improve the necessary technologies for autonomously performing an intervention mission in underwater environments. RAUVI project aims were to design and develop an Underwater Autonomous Robot, able to perceive the environment by means of acoustic sensors, and equipped with a robotic arm in order to autonomously perform simple intervention tasks, dealing with the ”search & recovery” problem. It is noticeable that a new I-AUV was developed (i.e. Girona500) and a new robotic arm ECA CSIP E5, under this coordinated project (UJI, University of Girona (UdG), University of Balearic Islands (UIB))
- TRIDENT: *Marine Robots and Dexterous Manipulation for Enabling Autonomous Underwater Multipurpose Intervention Missions*. EU research project (FP7-ICT-248497). TRIDENT proposed a new methodology for multipurpose underwater intervention tasks with diverse potential applications like underwater archaeology, oceanography and offshore industries. Trident was based on new forms of cooperation between an Autonomous Surface Craft and an Intervention Autonomous Underwater Vehicle. The new methodology allows the user to specify an intervention task to be undertaken with regards to a particular target object, but after that the object is automatically recognized and manipulated by the robot in a completely autonomous way.
- TRITON: *Multisensory Based Underwater Intervention through Cooperative Marine Robots*. Spanish research project (DPI2011-27977-C03). The main goal of TRITON was the use of autonomous vehicles for the execution of complex underwater intervention tasks. The project focused on the use of several vehicles (an AUV and an I-AUV) running in a coordinated manner during the execution of a mission, and on the im-

provement of the manipulation capabilities already developed in RAUVI. The project explored two scenarios that demonstrate the capabilities to be developed: search and recovery of an object, and panel intervention in the context of underwater observatories, both to be developed autonomously.

- MERBOTS: *Multifunctional cooperative marine robots for intervention domains*. Spanish research project (DPI2014-57746-C3). MERBOTS aimed at progressing in the underwater intervention systems development. To that end, an extensive use of multirobot cooperation and multimodal perception systems was planned. The methodology designed uses up to three heterogeneous vehicles cooperating in different configurations at each phase of the mission. A significant achievement through this project has been the commencement of research on wireless communications by the UJI team, culminating now with the first Ph.D. Thesis at UJI in this context.

As aforementioned, the IRSLab has coordinated several Spanish research projects, together with the UdG and UIB. During all these projects, the three universities agreed to use the same framework to develop the code. Thus, most of the parts of code developed in this thesis has been written in C++ and Python, and using Robot Operating System (ROS) (2) as middleware.

1.4 Contributions

Throughout this Ph.D., some problems regarding the HRI to control underwater robots have been faced and the following list shows the most relevant contributions. In the following chapters, more details about them will be given.

- An algorithm focused in a non-expert user assistance has been developed to allow the target identification and mission parameters specification.
- Two different GUI approaches have been proposed, in the context of a Spanish and an European research projects. Both GUIs have been compared with both commercial and research projects GUIs.
- A pilot study was implemented, where some usability test, dealing with Virtual Cockpit (VC) concept and different input devices, were carried out with potentials users.
- Getting a more realistic immersive feeling. The integration of the Inertial Measurement Unit (IMU) data of the Head-Mounted Display (HMD) and the head tracking system data, allow the user to control the camera Point Of View (POV) by simply moving his/her head. A principal advantage of working through the aforementioned immersive device is that it avoids distractions, improving, in the meantime, the necessary concentration for enhancing the human-machine interaction.
- Improving the user feedback. Instead of paying attention to multiple screens, highlighting a lot of parameters, this proposal reduces the information displayed to the user by filtering only the most relevant one in each time. This information is displayed in a visually effective and intuitive in the developed VC.

- A new architecture has been proposed, in order to support the control of a HROV, to manage the level of the autonomy of the robot and to enable the user to remain as a supervisor of the mission. The system allows a smooth transition between the autonomous or teleoperated mode.
- An abstraction layer to manage the input devices (joysticks, 3D mouse, gamepad) has been defined, enabling each device depending on the task to be carried out.
- In order to control a HROV, a module for controlling it as autonomous and another to control the overall mission have been integrated into the proposed architecture.
- The *Robot Safety Measures* algorithm has been created, which takes care of the robot preventing damages when the user teleoperates it.

1.5 Publications

The following list represents all the publications achieved during this thesis.

Book chapters

- Pérez, J., Sales, J., Peñalver, A., Fernández, J. J., Sanz, P. J., **García, J. C.**, Martí, J. V., Marín, R., and Fornas, D., "Robotic Manipulation within the Underwater Mission Planning context: A Use Case for Benchmarking", (2015) In: Carbone G., Gomez-Bravo F. (eds) Motion and Operation Planning of Robotic Systems. Mechanisms and Machine Science, vol 29. Springer, Cham. `http://doi-org-443.webvpn.fjmu.edu.cn/10.1007/978-3-319-14705-5_17`

International journal

- De Novi, G., Melchiorri, C., **García, J. C.**, Sanz, P. J., Ridao, P., and Oliver, G., "New approach for a Reconfigurable Autonomous Underwater Vehicle for Intervention", Aerospace and Electronic Systems Magazine, IEEE, vol. 25, no. 11, pp. 32 -36, 2010
- **García, J. C.**, Prats, M., Sanz, P. J., and Marín, R., "User interface oriented to the specification of underwater robotic interventions", Journal of Maritime Research, vol. 8, no. 2, 2011
- Prats, M., Ribas, D., Palomeras, N., **García, J. C.**, Nannen, V., Wirth, S., Fernández, J. J., Beltrán, J. P., Campos, R., Ridao, P., Sanz, P. J., Oliver, G., Carreras, M., Gracias, N., Marín, R., and Ortíz, A., "Reconfigurable AUV for intervention missions: a case study on underwater object recovery", Intelligent Service Robotics, vol. 5, no. 1, 2012
- Fernández, J. J., Prats, M., Sanz, P. J., **García, J. C.**, Marín, R., Robinson, M., Ribas, D., and Ridao, P., "Grasping for the Seabed: Developing a New Underwater Robot Arm for Shallow-Water Intervention", IEEE Robotics & Automation Magazine, vol. 20, 4 vol., pp. 121 - 130, 2013

- **García, J. C.**, Patrão, B., Almeida, L., Pérez, J., Menezes, P., Dias, J., and Sanz, P. J., "Design and Evaluation of a Natural Interface for Remote Operation of Underwater Robots", IEEE Computer Graphics and Applications, vol. 37, Issue: 1, Jan.-Feb. 2017
- Pérez, J., Sales, J., Peñalver, A., Fornas, D., Fernández, J. J., **García, J. C.**, Sanz, P. J., Marín, R., and Prats, M., "Exploring 3-D Reconstruction Techniques: A Benchmarking Tool for Underwater Robotics", IEEE Robotics & Automation Magazine, vol. 22, no. 3, 2015.
- Peñalver, A., Pérez, J., Fernández, J. J., Sales, J., Sanz, P. J., **García, J. C.**, Fornas, D., and Marín, R., "Visually-guided manipulation techniques for robotic autonomous underwater panel interventions", Annual Reviews in Control, vol. 40, 2015

International conferences

- Palomeras, N., **García, J. C.**, Prats, M., Fernández, J. J., Sanz, P. J., and Ridao, P., "A Distributed Architecture for Enabling Autonomous Underwater Intervention Missions", in 4th Annual IEEE International Systems Conference 2010, San Diego (California, United States), 2010
- **García, J. C.**, Fernández, J. J., Sanz, P. J., and Marín, R., "Increasing Autonomy within Underwater Intervention Scenarios: The User Interface Approach", in 2010 IEEE International Systems Conference, San Diego (California, United States), 2010
- Prats, M., **García, J. C.**, Fernández, J. J., Marín, R. and Sanz, P. J., "Towards specification, planning and sensor-based control of autonomous underwater intervention", in IFAC 2011, Milano, Italy, 2011
- Prats, M., **García, J. C.**, Fernández, J. J., Marín, R., and Sanz, P. J., "Advances in the specification and execution of underwater autonomous manipulation tasks", in IEEE OCEANS 2011, Santander (Spain), 2011
- **García, J. C.**, Prats, M., Sanz, P. J., Ribas, D., Ridao, P., Wirth, S., and Oliver, G., "An integration perspective of the TRIDENT project", in 5th International Conference on Cognitive Systems, Vienna (Austria), 2012
- Fernández, J. J. , Prats, M., **García, J. C.**, Marín, R., and Peñalver, A., "Manipulation in the Seabed: A New Underwater Robot Arm for Shallow Water Intervention", in Conference on Embedded Systems, Computational Intelligence and Telematics in Control, Würzburg (Germany), 2012
- Prats, M., **García, J. C.**, Wirth, S., Ribas, D., Sanz, P. J., Ridao, P., Gracias, N., and Oliver, G., "Multipurpose Autonomous Underwater Intervention: A Systems Integration Perspective", in 20th Mediterranean Conference on Control & Automation (MED 2012), Barcelona (Spain), 2012
- Sales, J., Santos, L., Sanz, P. J., Dias, J., and **García, J. C.**, "Increasing the Autonomy Levels for Underwater Intervention Missions by Using Learning and Probabilistic Techniques", in ROBOT2013: First Iberian Robotics Conference, Madrid (Spain), 2013
- Peñalver, A., Pérez, J., Fernández, J. J., Sales, J., Sanz, P. J., **García, J. C.**, Fornas, D., and Marín, R., "Autonomous Intervention on an Underwater Panel mockup by using

Visually-Guided Manipulation Techniques”, in 19th IFAC World Congress, Cape Town (South Africa), 2014

- **García, J. C.**, Patrão, B., Pérez, J., Seabra, J., Menezes, P., Dias, J., and Sanz, P. J., “Towards an immersive and natural gesture controlled interface for intervention underwater robots”, in MTS/IEEE Oceans’15, Genova (Italy), 2015
- **García, J. C.**, Pérez, J., Menezes, P., and Sanz, P. J., “A control architecture for hybrid underwater intervention systems”, in 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC 2016), Budapest (Hungary), 2016

International workshops

- **García, J. C.**, Prats, M., Sanz, P. J., Marín, R., and Belmonte, O., “Exploring Multimodal Interfaces For Underwater Intervention Systems”, in Proceedings of the ICRA 2010 Workshop on Multimodal Human Robot Interfaces, Anchorage (Alaska), 2010
- **García, J. C.**, Peñalver, A., Prats, M., and Sanz, P. J.,“Recent Progress in HRI for Underwater Robotic Intervention”, in Workshop on Human Robot Interaction (HRI), in IEEE International Conference on Robotics and Automation (ICRA’13), Karlsruhe (Germany), 2013
- **García, J. C.**, Patrão, L., Pérez, J., B., Almeida, Menezes, P., Fornas, D., Dias, J., and Sanz, P. J., “Immersive underwater robot control: HRI benchmarking with UWSim”, in workshop “Towards Standardized Experiments in HRI, IEEE/RSJ International Conference on Intelligent Robots and Systems 2015, Hamburg (Germany), 2015.
- Sanz, P. J., Pérez, J., Sales, J., Peñalver, A., Fernández, J. J., Fornas, D., Marín, R., and **García, J. C.**, “A benchmarking perspective of underwater intervention systems”, in IFAC Workshop on Navigation, Guidance and Control of Underwater Vehicles, Girona (Spain), 2015
- Sanz, P. J., Peñalver, A., Sales, J., Fernández, J. J., Pérez, J., Fornas, D., **García, J. C.**, and Marín, R., “Multipurpose Underwater Manipulation for Archaeological Intervention”, in 6th International Workshop on Marine Technology (MARTECH’15), Cartagena (Spain), 2015

National conferences

- **García, J. C.**, Prats, M., Sanz, P. J., and Marín, R., “Incrementando el nivel de Autonomía en los Sistemas Robóticos de Intervención Submarina: punto de vista de la Interfaz de Usuario”, in XXXI Jornadas de Automática (CEA), Jaén (Spain), 2010
- **García, J. C.**, Fernández, J. J., Prats, M., Marín, R., and Sanz, P. J., “Avances en la especificación y ejecución de tareas autónomas de manipulación submarina”, in XXXII Jornadas de Automática (CEA), Sevilla (Spain), 2011
- Prats, M., **García, J. C.**, and Sanz, P. J., “Una nueva herramienta para la simulación y supervisión 3D de misiones de intervención submarinas”, in III Workshop de robótica: robótica experimental, Sevilla (Spain), 2011

- Fernández, J. J., **García, J. C.**, Prats, M., Sanz, P. J., Ribas, D., Ridao, P., Wirth, S., and Oliver, G., “Proyecto RAUVI: Historia y Resultados”, in XXXIII Jornadas de Automática, Vigo (Spain), 2012
- Sanz, P. J., Fernández, J. J., Pérez, J., Peñalver, A., **García, J. C.**, Fornas, D., Sales, J., Bernabé, J. Antonio, and Marín, R., “GRASPER: Un Proyecto Dirigido a Incrementar la Autonomía de la Manipulación Submarina”, in XXXIV Jornadas de Automática, Terrassa (Spain), 2013
- Peñalver, A., Fernández, J. J., Pérez, J., Sales, J., **García, J. C.**, Fornas, D., Marón, R., and Sanz, P. J., “Avances recientes en manipulación subacuática en el subproyecto GRASPER”, in XXXV Jornadas de Automática, Valencia (Spain), 2014
- Sanz, P. J., Pérez, J., Sales, J., Fernández, J. J., Peñalver, A., Fornas, D., **García, J. C.**, and Marín, R., “MERMANIP: Avances recientes en la manipulación autónoma cooperativa submarina”, in XXXVI Jornadas de Automática, Bilbao (Spain), 2015
- **García, J. C.**, Pérez, J., Fernández, J. J., Menezes, P., and Sanz, P. J.,“Mejoras en la arquitectura de control e interfaz de usuario para el control de un HROV”, in XXXVII Jornadas de Automática, Madrid (Spain), 2016

Other publications Additionally to the previous publications, during this thesis some other work has been done.

- **García, J. C.**, Felip, J., Sanz, P. J., ‘The 2007 Spanish Humanoids Competition: From the Winners Point of View”, in 8th Conference on Autonomous Robot Systems and Competitions, Aveiro (Portugal), 2008
- **García, J. C.**, Sanz, P. J., and Cervera, E., “Using Humanoids for Teaching Robotics and Artificial Intelligence Issues. The UJI Case Study”, in III Workshop de robótica: robótica experimental, Sevilla (Spain), 2011
- **García, J. C.**, Felip, J., Veiga, C., and Sanz, P. J., “Casos de uso y mejoras de un robot humanoide de bajo coste”, in XXXV Jornadas de Automática, Valencia, Spain, 2014
- Cervera, E., **García, J. C.**, and Sanz, P. J., “Toward the Robot Butler: The HUMABOT Challenge”, IEEE Robotics & Automation Magazine, vol. 22, no. 2, 2015

1.6 Outline

This thesis has been structured in following chapters:

Chapter 2: Unmanned Underwater Vehicles: an overview.

This chapter reviews the state-of-the-art concerning unmanned underwater vehicles, paying special attention to the underwater robots technology and the most relevant problems regarding underwater domains. Additionally, a summary with the most relevant research projects will be introduced.

Chapter 3: Issues of Human-Robot Interaction in underwater robotics.

As one of the goals of this thesis concerns the HRI, this chapter summarizes the most common issues regarding a teleoperation process and ergonomic factors.

Chapter 4: Developing a Graphical User Interfaces to control an Underwater Vehicle.

We should consider the GUI is the most critical part concerning the HRI. A poor-designed GUI will represent a huge handicap for the user and will affect the usability and control of the robot. Throughout this thesis, the robot type to be used has evolved from an AUV to a HROV, so this chapter introduces two different underwater intervention approaches developed according to the robot to be used.

Chapter 5: Towards a multimodal and 3D immersive user interface.

In order to explore a multimodal and 3D immersive user interface, this chapter introduces the study and development carried out during a research stay at University of Coimbra. This chapter not only introduces how newer devices such as Leap Motion™ can substitute a traditional joystick to control the robot, but also how the use of the HMD Oculus Rift™ enhances the user experience. Additionally, the usability test carried out in order to validate this approach will be explained.

Chapter 6: An architecture to support interaction with a HROV.

This chapter introduces a new approach to control a HROV, which could operate as both autonomous or teleoperated mode, enabling smooth transitions between them. This approach merges the benefits of both, autonomous and teleoperated systems, while eliminating some limitations of each of them.

Chapter 7: Final remarks.

Finally, the last chapter summarizes all the work developed in the conclusion section, as well as outlining some future work lines, that can be of interest for research as a consequence of the results obtained in this Ph.D. thesis work.

Chapter 2
Unmanned Underwater Vehicles: an overview

2.1 Introduction

The ocean covers about two-thirds of the earth and hosts a large number of invaluable archaeological sites that belong to the historical heritage of nations, as well as relevant marine flora and fauna, which are yet to be discovered or studied. Unfortunately, there is another point of interest related to oceans: ecological disasters (i.e. as a result of oil spills). Thus, these spillages generates the increasing demand for underwater robots more capable than humans, specialized and reliable, which allow the user to interact in these hazardous environments.

One example of utility is the oceanography research, where exploratory dives are traditionally the first step for marine scientists to acquire information on a previously unknown area of scientific interest. Manned submersibles have been the platform of choice for such exploration, as they allow a high level of environmental perception by the scientist on-board and the ability to take informed decisions on what to explore next. However, manned submersibles have extremely high operation costs, provide very limited operational time and represent an elevated risk from the human point of view. In 2010, the Instituto Español de Oceanografía (IEO) presented the Liropus, a ROV capable of maneuvering up to 2000 meters deep and which allows scientist to study a multitude of unexplored areas through non-invasive methods, to better understand underwater habitats, discover new species for science and, in general, improve knowledge about the seabed, which is essential to complete the requirements demanded by different scientific projects and address the new marine conservation challenges. Over a 10-years period, the Liropus carried out 28 oceanographic campaigns in the Atlantic and the Mediterranean. Among its milestones we can mention the first filming in a recently created underwater volcano, El Tagoro on the island of El Hierro; participation in important archaeological campaigns, such as the scientific expedition to the ”Nuestra Señora de Las Mercedes” wreck; or the discovery of ecosystems and habitats of great ecological importance such as rocky reefs with coral, gorgonian forests, sponges and soft corals, among others. Furthermore, the Liropus plays a fundamental role by allowing the acquirement of necessary information to complete the commitments assumed in different national and international projects in which the IEO participates.

Thus, thanks to recent advances in sensors, navigation techniques and robotic systems, the current state of Unmanned Underwater Vehicles (UUV) enables us to perform all those aforementioned kinds of missions.

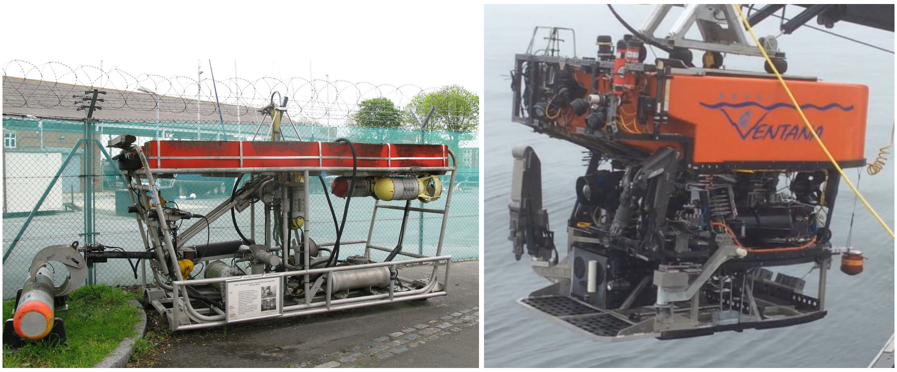
2.2 Underwater robots

The following subsections explain the different kind of unmanned underwater robots used nowadays in underwater environments for intervention.

2.2.1 ROV

ROVs are one of the most used robots for underwater missions, being historically one of the first attempts to explore subsea domains, adapting concepts already known and established in ground-aerial vehicles. The robot is linked to the vessel or a floating platform, where the user is controlling the robot, through a tethering cable. This umbilical cable is an armoured cable that contains a group of electrical conductors and fiber optics that carry electric power, video, and data signals. In the case of very deep missions, this tethering cable is used along with a Tether Management System (TMS). This device contains the ROV during lowering through the splash zone or, on larger work-class ROV, a separate assembly which sits on top of the ROV. The purpose of the TMS is to lengthen and shorten the tether so the effect of cable drag where there are underwater currents is minimized.

The first references to ROVs appears in the 1970s and ’80s, although most of the technology was developed and funded by the U.S. Navy at the end of 1960s. These references concern ”Cutlet”, a ROV used to recover practice torpedoes and mines (see Fig. 2.1(a)). In the 1980s, where the offshore industry required deeper interventions exceeding the reach of human divers, ROVs became essential and the technology needed to achieve these operations evolved considerably. Since then, technological development in the ROV industry has accelerated and today ROVs perform numerous tasks in many fields. The robot design follows mainly two different shapes: *open frame* (see Fig. 2.1) and *torpedo shaped*. Open frame design consisting of an open frame where all the operational sensors, thrusters, and mechanical components are enclosed, but the maneuvering is limited due to its poor hydrodynamic design. On the other hand, torpedo-shaped robots have excellent navigation capabilities, but very low possibilities to be endowed with suitable robotic arms. This constraint prevents their use for manipulation tasks and means they can be operated mainly for data gathering or inspection. Medium and large-size ROVs, which are usually equipped with multiple sensors and robotic arms, are usually used for intervention missions and have been named “work-class ROV”.



(a) Cutlet (b) ROV Ventana used at MBARI

Fig. 2.1 ROV evolution, from one of the first historic versions to a modern example nowadays.

2.2.1.1 Advantages

There are some advantages that make these robots the most common in interventions. One of the most relevant is the operational time. The lack of batteries (the robot is powered directly from the vessel or the surface platform) allows operating with the robot as much as necessary, the ROV pilot being the unique limitation. In fact, during some long operations, different users can work together by simply switching them after a certain period of time.

Working at depth means withstanding higher pressure and it is important to make the robot more resilient to hazardous conditions. This not only increases the weight of the robot due to the reinforcement of the robot structure, but also affects the thrusters used to move the robot and modify some other components like sensors. In any case, this can be done by just adapting commercial solutions or developing custom solutions.

2.2.1.2 Drawbacks

ROVs are normally large and heavy vehicles that need significant logistics for their transportation and handling: they need an expensive oceanographic vessel with a heavy crane and automatic TMS and a Dynamic Position (DP) system. In fact, the need for an umbilical cable introduces additional problems of control (e.g. the communication of real-time data) and sets limits on the workspace (e.g. speed, mobility and spatial range are very limited).

Concerning HRI systems, the complex user interfaces and control methods require expert users for their use. These two facts significantly increase the cost of the applications. Thus, there are limitations regarding the ROV pilot, such as cognitive fatigue and high stress that they normally suffer, mainly due to the master-slave control architectures and the very complex HRI systems used ((3)).

2.2.2 AUV

Some researchers started to think about the natural evolution of the intervention ROV (e.g. work-class ROV), the I-AUV. Without the need for the TMS and the DP, light I-AUVs could theoretically be operated from cheap vessels of opportunity, considerably reducing the cost of operation. Considering the fast development of battery technology, and removing the ROV pilot from the control loop, we can start to think about intervention operations that last for several days, where the user and the vessel are only needed on the first and the last day for launch and recovery, as well as for periodic mission task supervision. But this fascinating scenario, where I-AUVs do the work autonomously, comes at the cost of endowing the robot with the intelligence needed to keep the operator out of the control loop.

Although standard AUVs are also operated without human intervention, they are mainly constrained to survey operations, flying at a safe altitude with respect to the ocean floor while logging data. I-AUVs must be operated in the close proximity of the seabed or artificial structures. They have to be able to identify the objects to be manipulated and the intervention tasks to be undertaken, while safely moving within a cluttered work area. While I-AUVs are the natural way of technological progress, they represent nowadays a research challenge for the robotics community.

2.2.2.1 Advantages

Despite the fact that technology evolves constantly, nowadays AUVs are not used at great depth because of the extra effort to prepare the robot and make it more robust, as was mentioned before. This can be observed as an advantage, because the robot is usually smaller and lighter, which makes it easier to transport in a smaller vessel. The use of batteries makes the TMS and a DP system unnecessary, so the maneuvering improves considerably.

2.2.2.2 Drawbacks

Keeping the user out of the control loop means that all responsibility for the mission falls on the robot and raises some questions like: “is the system intelligent enough to solve all the problems?” or “how can the system react under complex situations?”. Additionally, the lack of connection between the robot and the user/vessel creates an uncertain situation due to the absence of mission information. Both drawbacks are the most relevant concerning HRI systems.

The use of batteries reduces the operational time, so this involves training the mission in order to ensure that it can be carried out within the specific robot configuration and mission parameters. If not, the mission should be modified, adding more steps/phases or endowing the robot with more capable batteries, which represents an extra weight. Obviously, adding more weight to the robot means checking its buoyancy properties.

2.2.3 Hybrid-ROV

Thanks to the advance in robotic systems and the associated technologies over the last few years, now it is possible to consider performing a range of undersea tasks which were considered only for ROVs or AUVs. This new class of hybrid robots are placed between these two types of systems, because they can act as both types depending on the configuration.

2.2.3.1 Advantages

The advantages of a HROV are diverse. For instance, the robot is able to perform some steps of the mission in an autonomous way, positively impacting on the consequent cognitive fatigue and stress inherent with the user, as well as eliminating the failures that accompany human factors. Thus, if there is connection between the robot and the surface, complex operations not supported by the robot autonomy, can be done by the user whenever needed. In some cases, partial user intervention is possible, enabling the user to temporarily take control of the robot and returning the control back to the robot when needed.

2.2.3.2 Drawbacks

Nevertheless, the main problems with the implementation of such a hybrid approach are related to the following aspects: the HRI and the robot system architecture.

Concerning the first one, the system should implement an easy-to-use procedure to move the control from the robot to the user, and vice-versa, as needed. This is challenging, not only from the HRI, but also from the technological point of view: how can we connect the robot and the user: wired, wireless or using a mixed system? If the connection is not wired, how does the latency of the system operate?

On the other hand, the architecture presents a significant complexity, because the system needs to provide a remotely operated mode in conjunction with the ability to control each stage of the task to achieve the desired robot autonomy.

2.2.3.3 Commercial HROVs

As mentioned before, this kind of robots are quite new and there are only few companies developing them. In fact, when searching for a hybrid solution in the website of the most relevant manufacturers, they only mention the ability to switch the power supply, from batteries to tethering cable. Only some of them offer a real HROV solution. In the following text, some of the most relevant robots are presented.

- The Saab‘s Double Eagle SAROV (see Fig. 2.2 left) is a HROV focused on the detection, classification and disposal of maritime explosives. According to the website[1], this HROV can work as AUV/ROV, but there is no mention of the possibility of an operational mode switch. They only mention that the robot can be operated in both modes, but there is no sign on how to do it during the mission.
- Woods Hole Oceanographic Institution (WHOI) developed the Nereus[2] (see Fig. 2.2 right). It can operate as an AUV until the mission finishes. Then, the vehicle’s support team can bring the vehicle back on board the ship and transform it into a ROV tethered to the ship via a micro-thin, fiber-optic cable. Through this tether, Nereus can transmit high-quality, real-time video images and receive commands from skilled users on the ship to collect samples or conduct experiments with a manipulator arm. Unfortunately, Nereus was lost at sea while exploring the Kermadec Trench at ca. 10,000 meters during a May 2014 research cruise.

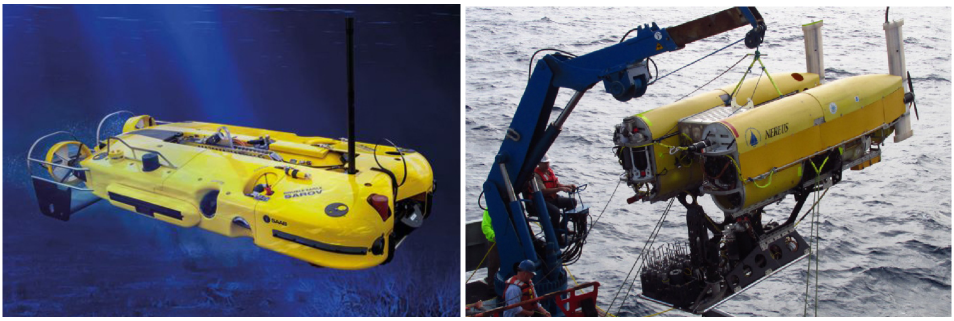


Fig. 2.2 Some commercial HROVs: Saab Double Eagle SAROV (left), WHOI Nereus (right)

So, in most cases, the robot must be recovered and must be modified physically adding or removing parts before the robot switch from an autonomous to a teleoperated vehicle. Fig. 2.3 shows the different aspect of Nereus robot[3], from the early stages of development until the final concept, in ROV and AUV mode.

2.2.4 New approaches

Nowadays, there are new initiatives exploring different concepts and approaches, like OceanOne or Aquanaut. The first one, was conceptualized by King Abdullah University of Science and Technology (KAUST) in Saudi Arabia and it was developed together with Standford Robotics Lab[4]. This project was focused on monitoring deep coral reefs in the

[1] https://saab.com/naval/underwater-systems/autonomous-underwater-vehicles/double_eagle_sarov/

[2] https://www.whoi.edu/main/nereus

[3] https://bit.ly/3bfcXR0

[4] http://cs.stanford.edu/group/manips/ocean-one.html

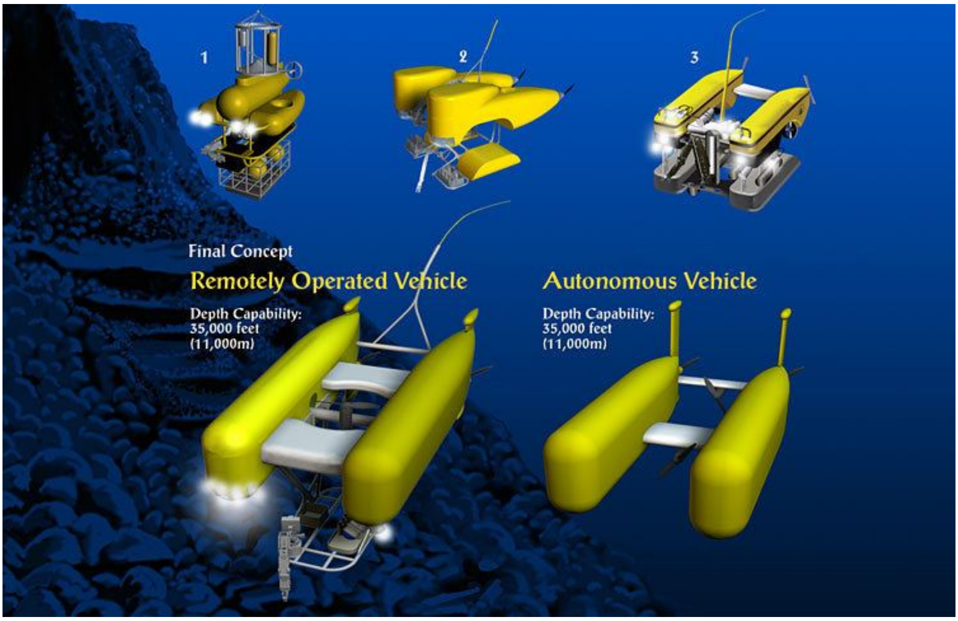
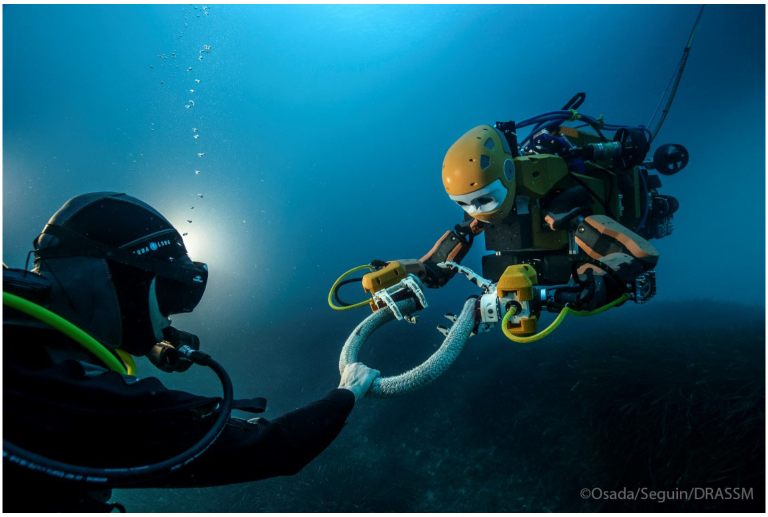


Fig. 2.3 Evolution of Nereus: (up) different concept models, (down) final concept in ROV mode (left) and AUV mode (right).

Red Sea, and pays special attention to the concept of creating an avatar, capable of teleporting a human being to inaccessible places, such as the seabed, and feeling all kinds of sensations from that environment. For instance, the archaeologists do not want an autonomous robot to come down and perform operations on its own, although they want to be the ones who are there in person, but without risking their physical integrity. The robot, which in fact can be classified as a ROV, has been designed like a human, making it easy-to-use for users even without experience. To achieve this goal, there are two clues: use a similar perspective to the user's face in relation to its own arms and the use of haptic feedback, which allows the robotic arms to grip objects without crushing them. This concept is challenging and the robot shape is totally different to previous approaches. Nevertheless, there are still some of the problems aforementioned, like the problem of batteries or the use of the tethering cable.

On the other hand, Aquanaut[5] explores the possibility to endow the robot with the capability to change the operational mode without the need of taking it out of the water. When the robot operates like an AUV (getting the body closed, see Fig. 2.5 (1)), the maneuverability is optimal and reduces the water resistance. Just when the robot achieves the goal or intervention capabilities that are needed, the robot starts the transformation (see Fig. 2.5 (2-3)): the top part of the robot's hull rises up from the base, exposing two massive arms that unfold from either side, and a wedge-shaped head packed full of sensors rotates into place (see Fig. 2.5 (4)). The robot uses a battery and the only way to communicate with Aquanaut is through an acoustic modem. The operator then sends simple commands, such as "Turn the valve at these coordinates 90 degrees clockwise." The robot will autono-

[5] https://bit.ly/3pnF9q1



©Osada/Seguin/DRASSM

Fig. 2.4 New underwater robots: Ocean One, a bi-manual underwater humanoid robot.

mously decide how to grasp the valve and how much force to apply while turning, and it will send back a confirmation when the task is complete. The operator is still directing the robot's actions, but in a way that does not require steering the robot by hand, or a bandwidth-intensive live video feed.

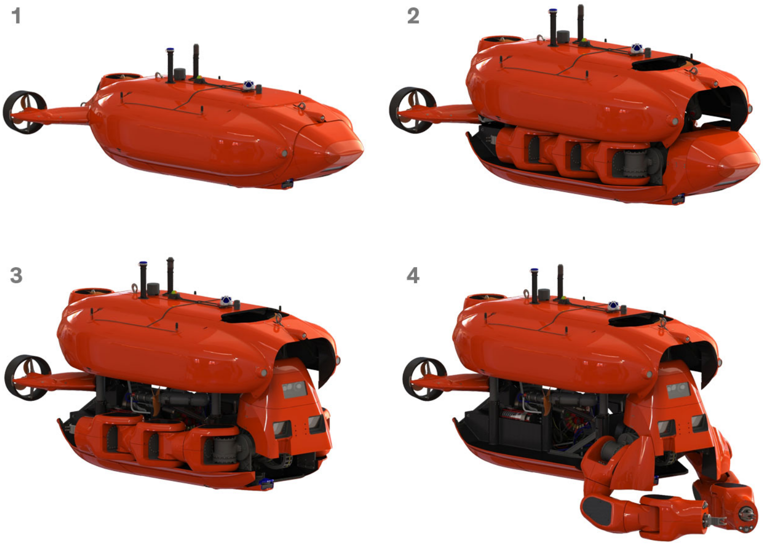


Fig. 2.5 New underwater robots: Aquanaut, a convertible robot able to transform itself from a ROV form (1) to I-AUV form (4).

2.3 Underwater robots applications

Nowadays a relevant number of field operations with UUVs in applications such as those mentioned hereinafter, need intervention capabilities in order to undertake the desired task. Most of the projects can be classified mainly in one of these four categories:

Oil and Gas Industry. The oil and gas industry is one of the principal users of underwater robotics technology. It uses work-class intervention ROVs to routinely inspect and repair the submerged infrastructures (e.g. inspection of subsea structures, pipelines, and platforms). AUVs have recently entered in this market, being already used to undertake geophysical surveys prior to pipe installation and later on, for their inspection. The use of hovering type AUVs for the inspection of 3D infrastructures like submerged oil wells, chains, risers, etc... have started to be considered, although these projects are still in a research stage, since they represent major challenges for contemporary field capabilities. A few simple intervention tasks have also started to be considered by the research community. For instance, proof of concept demonstrations for tasks like acoustic-based homing to a subsea panel, realtime vision-based localisation relative to it, docking and opening a valve or plugging a hot stab have already been demonstrated in simplified mock-up environments.

Search and Recovery. I-AUVs will have a great potential here, doing tasks like search, recognition, and localization of different objects. Some mission examples of this category are: searching for shipwrecks (and hence searching for survivors), the inspection of the remains of a shipwrecks or an airplane that crashed into the water, or recovering specific parts of them (e.g. the flight data recorder).

Deep water Archaeology. During a significant part of human history (one million years), the continental shelf was wider than nowadays. The sea level was about 130 m lower, and these coastal and lowland landscapes were attractive for human settlement. About 16000 to 6000 years ago, after the last Ice Age, the sea level increased until current levels and these territories were flooded, hiding important clues of our historical heritages. Shipwrecks, and in particular deep-sea wrecks (which may not be easily pillaged), are also a very important source of historical information. Underwater archaeologists are primarily interested in documenting submerged sites. High resolution 2D/3D seafloor mapping techniques are of high interest for them. There exist very few precedents of deep underwater excavations, mostly using high cost ad-hoc hardware or expensive ROV operations. But, most of the archaeological institutions are small, having access to small boats not equipped for deep intervention. Small and light HROVs first, and I-AUVs later on, have a great potential to assist archaeologists beyond 50 m depth.

Science. Permanent observatories are infrastructures located on the seabed, which hold instrumentation needing, hence, periodic maintenance. Common tasks include downloading vast amounts of data (for isolated noncabled observatories), connecting/disconnecting a cable, replacing batteries, instrumentation defouling, as well as placing and recovering sensor packages. I-AUVs have a direct application here, since they have the potential to be operated from inexpensive ships of opportunity, drastically reducing the associated costs.

2.4 Brief review of underwater robotics research projects

The first pioneering works in the field of autonomous underwater robotics took place during the early 90s. Nevertheless, the first simple autonomous intervention arrived in the last decade when the systems were capable of demonstrating these features at sea. The following image represents an overview of this evolution.

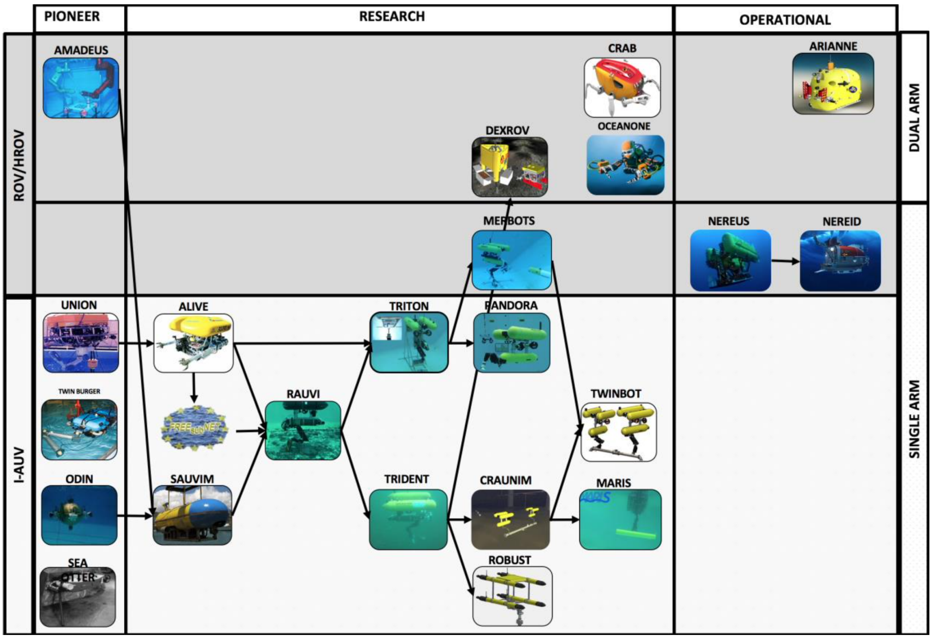


Fig. 2.6 Underwater research projects classified depending on the robot used and the relationship between them.

AMADEUS: 1993/1996/1999 AMADEUS phase I represents the first attempt in developing a dexterous gripper suitable for underwater applications. The 3-fingered gripper was hydraulically actuated and coordinately controlled by mimicking, within each finger, the motions of an artificial elephant trunk. AMADEUS phase II on the other hand, studied the coordinated control of two underwater 7 Degrees Of Freedom (DOF) electro-mechanical arms. Each arm was equipped with an underwater JR3 force/torque wrist sensor. More information in (4).

ODIN: 1994/1995 The research project Omni-Directional Intelligent Navigator (ODIN) was developed at the Autonomous Systems Laboratory (ASL) of the University of Hawaii as an advanced underwater robotic technology test platform. ODIN was a 6 DOF underwater vehicle with dual operational modes (autonomous and tethered) and a single DOF mechanical manipulator. More information in (5).

OTTER: 1995 OTTER (Ocean Technology Testbed for Engineering Research) was an underwater robot designed to be used as a testbed for autonomous technologies. Both

OTTER's hardware and software systems were configured to support simultaneous development and testing of different concepts for underwater robotics by independent researchers. A general control-software framework enabled common access to all subsystems and avoids the duplication of basic robotic functionality jointly required by all projects. Additionally, the autonomous technologies enabled by the results of individual research were mutually compatible and can be easily integrated into a single robotic system. Some examples of technologies demonstrated on the OTTER underwater robot include control from a real-time vision-sensing system, coordinated arm/vehicle control, and control from 3D graphical user interfaces. More information in (6).

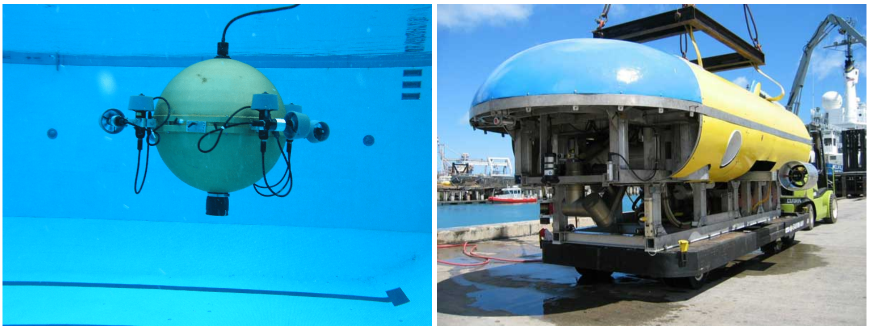


Fig. 2.7 ODIN (left), SAUVIM (right)

UNION: 1996/1999 The main goal of the Union Esprit Basic Research Action was to develop methods for increasing the autonomy and intelligence of ROVs. The project focused mainly on the development of coordinated control and sensing strategies for combined manipulator and vehicle systems. At the end, only experimental validation within simulation conditions was provided. More information in (7).

SAUVIM: 1997/2009 this was a project funded by the Office of Naval Research and carried out at the Autonomous System Laboratory of the University of Hawaii and was conceived as an AUV with accurate navigation and station keeping capabilities to allow for the recovery of seafloor objects. In particular, SAUVIM was supposed to use its passive arm to position itself with respect to the object of interest and use its 7 DOF electromechanical arm to carry out an intervention. SAUVIM was initially designed to recover test missiles from the seabed for the Pacific Missile test Centre in Hawaii. More information in (8).

SWIMMER: 1999/2001 A hybrid AUV/ROV intervention system provided an efficient way to ensure permanent Inspection, Maintenance, and Reparation operations over deep-water oil production facilities. A ROV umbilical was integrated between the surface facility and the subsea site. The SWIMMER AUV transports the ROV to the subsea site and connects the ROV to the umbilical at the subsea location where it can be normally operated from the surface. More information in (9).

ALIVE: 2001/2004 The ALIVE vehicle was equipped with docking and 7 DOF manipulation arms to complete valve override and hot stab connections. It can also be used for the deployment and recovery of acoustic or seismic beacons at sea-bottom. During the

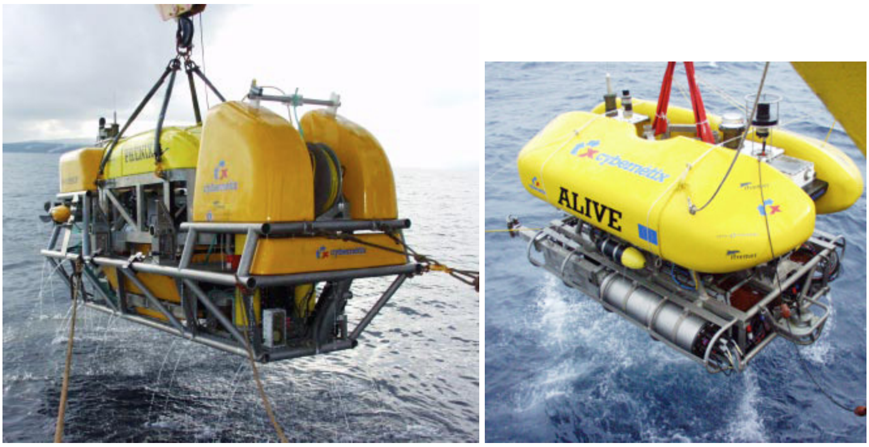


Fig. 2.8 SWIMMER (left), ALIVE (right)

final demo of the project, ALIVE proved its ability to autonomously navigate, dock and operate on an underwater panel similar to those of the oil industry. More information in (10).

RAUVI: 2009/2011 The main goal of the RAUVI project was to develop and improve the necessary technologies for autonomously performing an intervention mission in underwater environments. RAUVI project aimed to design and develop an Underwater Autonomous Robot, able to perceive the environment by means of acoustic and optical sensors, and equipped with a robotic arm in order to autonomously perform simple intervention tasks (e.g. search and recovery problem). More information in (11).

CSURVEY: 2009/2012 The CSurvey project deals with the development and evaluation of a semi-autonomous inspection unit for underwater structures and ship hulls. For the inspection task multimodal sensor data are classified by machine learning methods. The sensor data are acquired by a camera system, a laser projection unit and a multi-beam echo sounder. CSurvey is a flexible system which can be mounted on different host systems and can be adapted to several operation scenarios. It is intended to be the basis for many applications and products in the area of underwater inspection. Furthermore it can also be used in other areas like production, logistics or security. More information in (12).

CUSLAM: 2009/2012 The aim of the CUSLAM project is the development of an algorithm which allows localization and mapping in complex, spatially confined underwater environments. This algorithm enables even small, weakly instrumented vehicles to act and fulfill meaningful tasks in difficult terrain (e.g. underwater production/processing plants). During extensive experiments the results are compared to reference measurements, in order to objectively verify the correctness of the algorithm. More information in (13).

NEREUS: 2009/2014 Nereus was a HROV built by WHOI. Constructed as a research vehicle to operate at depths of up to 11,000 m/36,000 ft, it was designed to explore Challenger Deep, the deepest surveyed point in the global ocean. Nereus began its deep sea voyage to Challenger Deep in May 2009 and reached the bottom on May 31, 2009. On this dive, the Nereus reached a depth of 35,768 feet (10,902 m), making the Nereus the world's second-deepest-diving vehicle in operation at the time, and the first since 1998 to explore the Mariana Trench, the deepest part of the ocean known. On 10 May 2014, Nereus was lost while exploring the Kermadec Trench at a depth of 9,900 metres (32,500 ft). Communications were cut off at around 2 p.m. local time, and debris retrieved later revealed that it imploded due to high pressure. More information in (14).

TRIDENT: 2010/2013 This project proposed a new methodology for multipurpose underwater intervention tasks. A team of two cooperative heterogeneous robots with complementary skills, an Autonomous Surface Craft (ASC) and an I-AUV endowed with a dexterous manipulator and a 3-fingered hand, was used to perform underwater manipulation tasks. TRIDENT concept was based on two phases: Survey and Intervention. During survey, the tandem vehicles map the seafloor. Next, the I-AUV was recovered and a map was built. Next, the user selects an object and a desired intervention task. Then, the I-AUV was launched again to navigate to the target. The vehicles search for the target and perform a multisensory-based intervention through free-floating manipulation. TRIDENT concept has been demonstrated in a harbour environment in an uncoupled way: 1) The capability of both vehicles working in tandem during mapping and 2) the capability of the I-AUV to navigate and intervene over the target. More information in (15).

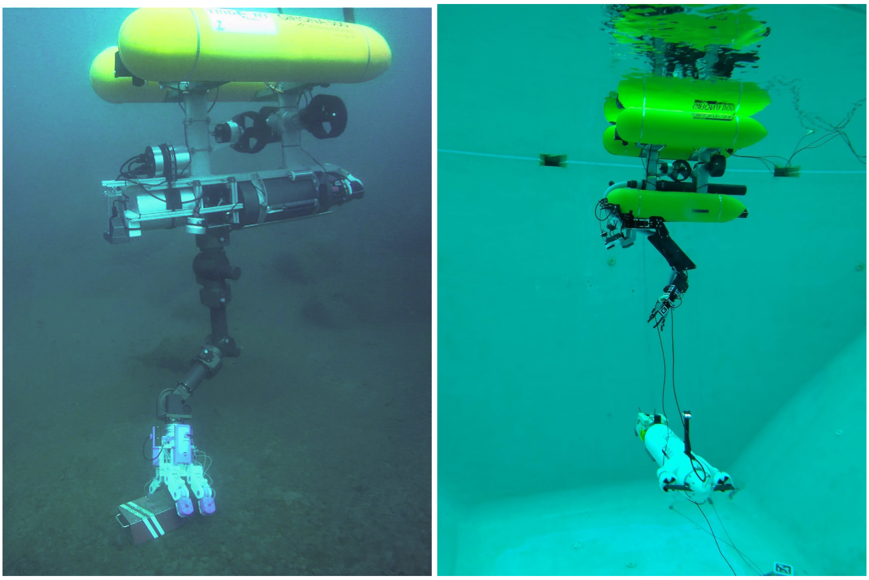


Fig. 2.9 TRIDENT (left), TRITON (right)

TRITON: 2012/2014 The main goal of TRITON was the use of autonomous vehicles for the execution of complex underwater intervention tasks. The project was focused on the use of several vehicles (an AUV and an I-AUV) running in a coordinated manner during the execution of a mission, as well as in increasing the dexterity of a robotic arm (currently under development in the context of the RAUVI project), that will be installed in the I-AUV. The TRITON project proposes two scenarios as a proof of concept to demonstrate the developed capabilities: the search and recovery of an object of interest (e.g. a blackbox) and the intervention of an underwater panel in a permanent observatory. More information in (16).

PANDORA: 2012/2014 PANDORA was aimed at making autonomous robots persistently autonomous, reducing the frequency of assistance requests. The key of this objective was the ability to recognise failure and respond to it, at all levels of abstraction and time constant. The project develops three themes: (1) describing the world for detecting failures in the task execution; (2) directing and adapting intentions by means of planning for responding to failures; and (3) acting robustly, mixing learning and robust control for making actions indifferent to perturbations and uncertainty. The project centered its validation tasks on AUVs acting in an oil field scenario in which the robots perform inspection, cleaning and valve turning. More information in (17).

MORPH: 2012/2016 The MORPH project was aimed at developing efficient methods and tools to map the underwater environment in situations that were not easily addressed by current technology. Namely, the missions that were of interest were those that involve underwater surveying and marine habitat mapping of rugged terrain and structures with full 3D complexity, including vertical cliffs. Potential applications include the study of cold water coral reef communities, ecosystems from underwater canyons, pipeline and harbor monitoring, or the inspection of wind turbine foundations. The project introduced and advanced a novel concept of an underwater robotic system composed of a number of mobile robot modules (nodes), carrying complementary sensors for perception of the environment. Instead of being physically coupled, the modules were connected via communication links that allow a flow of essential information among them. Without rigid links, the so-called MORPH Supra-Vehicle can reconfigure itself and adapt according to the environment and mission goals, responding, for example, to the shape of the terrain, including vertical walls. The flexibility allows for more optimal positioning of each sensor, an increased number of simultaneous viewpoints, and generally high-resolution data collection. More information in (18).

CRAUNIM: After completing the TRIDENT project in 2013, the consortium led by the IRSLab developed the idea of evolving its previous concept into a complete cooperative system involving two cooperative I-AUVs for load transportation and object assembly: the Cooperative Robots for Autonomous Underwater Intervention Missions (CRAUNIM) concept. Although this proposal did not mature at the EU level, it was later developed in the MARIS and TWINBOT (Italian and Spanish national projects respectively).

MARIS: 2013/2016 The general objective of the MARIS project was studying, developing and integrating, technologies and methodologies enabling the development of underwater robotic systems for manipulation and transportation activities; within underwater scenarios which were typical for the off-shore industry, for the underwater search and rescue operations, as well as for the underwater scientific missions. Within such ambitious objectives, the proposing institution also intends to experimentally demonstrate, in the form proof-of-concept, the achievable operational capabilities; by also integrating the research results within real experimental systems. On the basis of the knowledge and experiences gained by the consortium, it was considered viable to coordinate the development of all the necessary technological and methodological aspects. Meanwhile, the consortium continues converging toward their final integration on the experimental, which is to be realized in parallel, starting from the sub-systems and advanced-stage designs made available by some of the proposing institutions. More information in (19).

MERBOTS: 2015/2017 The MERBOTS project, founded by the Spanish Ministry, aims at progressing in the underwater intervention systems development. To that end, an extensive use of multirobot cooperation and multimodal perception systems was planned. The methodology designed uses up to three heterogeneous vehicles cooperating in different configurations at each phase of the mission. A first stage was implemented with an AUV endowed with acoustic and optical sensors and an ASC whose mission was the localization and supervision of the AUV and link it to the remote base. This configuration was used to elaborate, first, an acoustic map on which a second survey of the AUV was planned. Later, the AUV uses the optical sensors and moves closer to specific regions to record detailed information of potential targets. The data from these surveys, georeferenced by the ASC, permit the acquirement of an accurate 3D reconstruction of the area under study to plan the intervention stage. Next, an operator, using an Human-Machine Interaction (HMI) that includes a target recognition system, identifies the target and plans the manipulation stage. Finally, a HROV with a multifunctional system formed by a manipulator and a hoover carries out the supervised intervention task. During this stage, an AUV equipped with cameras stays close and supports the HROV operation providing images from an external viewpoint. More information in (20).

DexROV: 2015/2018 DexROV proposes to implement and evaluate novel operation paradigms with safer, more cost effective and time efficient ROV operations. As a keystone of the proposed approach, manned support will to a large extent be delocalized within an onshore ROV control center, possibly at a large distance from the actual operations, relying on satellite communications. The proposed scheme also makes provision for advanced dexterous manipulation and semi-autonomous capabilities, leveraging human expertise when deemed useful. The outcomes of the project will be integrated and evaluated in a series of tests and evaluation campaigns, culminating with a realistic deep sea (1,300 meters) trial in the Mediterranean sea. The robot was endowed with two right-handed arms. More information in (21).

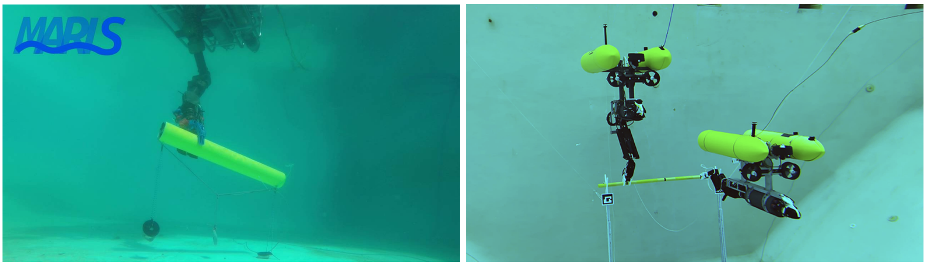


Fig. 2.10 MARIS (left), TWINBOT (right)

ROBUST: 2015/2020 The ROBUST proposal aims at developing seabed in situ material identification through the fusion of two technologies, namely laser-based in-situ element-analyzing capability merged with AUV technologies for sea bed 3D mapping. This will enable resource identification done by robotic control enabled by the synergy between AUV hovering and manipulator capabilities. The underwater robotic laser process was the LIBS, used for identification of materials on the sea bed. The AUV will dive, identify the resources that were targeted for LIBS scanning through 3D real time mapping of the terrain (hydro-acoustically, laser scanners, photogrammetry) and position the LIBS in the required locations of mineral deposits on the ocean floor to autonomously perform qualitative and quantitative analyses. More information in (22).

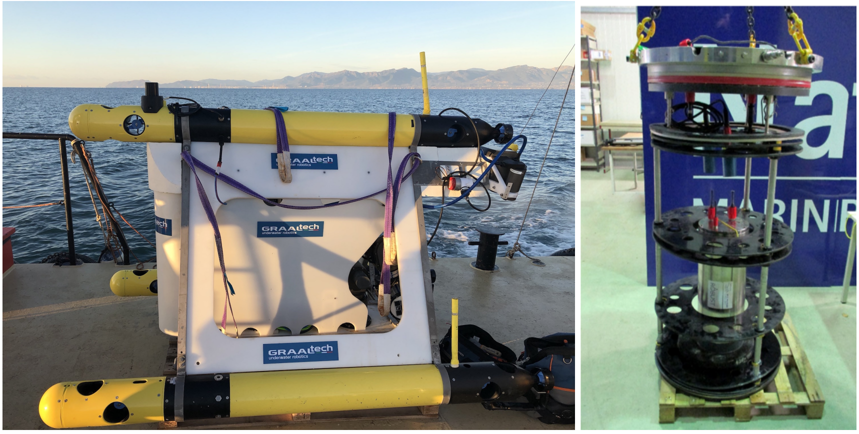


Fig. 2.11 Robust project developments: AUV (left), LIBS (right)

TWINBOT: 2018/2021 TWINBOT project aims to achieve a step forward in underwater intervention using a set of two I-AUV's, which will be able to solve strategic missions devoted to cooperative survey and cooperative manipulation (transport and assembly) in a complex scenario. A multimodal communication (RF/VLC/acoustics) architecture will be used to communicate both vehicles. A laser scanner will provide 3D point clouds of the objects of interest which will be used by an object recognition architecture to identify and locate them for manipulation and semantic SLAM purposes. Potential applications, include oceanography (permanent observatories), offshore (oil and gas) and nuclear industries. TWINBOT pushes the knowledge frontiers by targeting cooperative load transport and cooperative mobile assembly tasks. At the time of writing this thesis, this cooperative load transport was successfully tested in controlled conditions, and the assembly tasks will be tested in the following months.

ARIANE: The hybrid ROV Ariane represents the next generation of remotely operated submersibles which carry their own energy source in the form of lithium-ion batteries, and its only link to the surface was a fibre-optic cable. The hybrid ROV Ariane undertakes dives to 2,500 m for short missions requiring manipulation, sampling, inspection or optical and acoustic mapping. Its propulsion system and navigation sensors enable it to work on any type of seabed, including very uneven ground such as canyons. Fitted with HD video cameras and a pan and tilt digital camera, Ariane can undertake high-quality optical inspections and generate high-resolution photogrammetry (or 3D optical mapping). It can be configured with different payload configurations, such us Exploring and Sampling (remotely operated manipulator arms with 5 and 7 sections, a basket, a pan and tilt digital camera and sampling tools through a suction sampler for fauna) and Mapping (a multibeam echo sounder and a pan and tilt digital camera). More information in (23).

NEREID: HROV Nereid Under-Ice was built to travel up to 40 kilometers (25 miles) laterally underwater, rather than the few hundred feet of a typical ROV, while still receiving control signals and transmitting data, including high-definition video, back to operators located on a ship via a hair-thin fiber optic tether. Instead of receiving power from this tether, as a traditional ROV does, Nereid Under-Ice carries its own battery power on board, which makes the tether much lighter and smaller. In addition, it also carries a full suite of acoustic, chemical, and biological sensors for investigating the underwater environment, as well as a seven-function electro-hydraulic manipulator arm. The vehicle was rated to dive 2,000 meters (6,500 feet) beneath the surface to sample or survey the mid-water or sea floor. More information in (24).

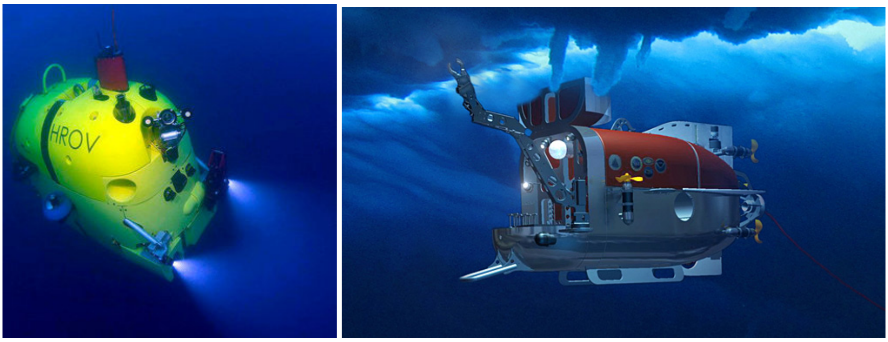


Fig. 2.12 ARIANE (left), NEREID (right)

CRABSTER: It was presented as an alternative to traditional AUVs and ROVs capable of withstanding strong currents thanks to a novel concept inspired by crabs. This robot has six legs that allow it to move around the seabed like a crustacean and, at the same time, generate fewer particles in suspension than with the use of propellers. Regarding manipulation, this robot was equipped with two manipulators capable of picking up objects, although the control was similar to that of an ROV. This robot was designed to work alongside an archaeologist with the aim of examining ships sunk in the 12th century in the Yellow Sea. Developed by the Korean Institute of Ocean Science and Technology (KIOST), the robot was built for the most dangerous deep-sea exploration missions. More information in (25).

Ocean One: It was an anthropomorphic underwater robot, designed to operate in deep aquatic conditions and equipped with an array of sensor modalities. The robot has two 7 DOF arms with electrically driven, torque-controlled joints adapted from the original Meka arm design. The arms were fitted with a series of elastic actuators that provide torque feedback to enhance compliant motion as well as force control, and safety. Additionally, central to the Ocean One concept was a human interface that connects the robot and human operator through haptics and vision. More information in (26).

Chapter 3

Issues of Human-Robot Interaction in underwater robotics

The interaction, and also intervention, between robot and users, started at the end of the 19th century with some primitives approaches to teleoperation. These first attempts, which were based on passive systems based on mechanical systems which transmit forces and efforts to the robot, evolved to the current systems where the robot can operate together with the user and are able to take decisions autonomously. But this evolution includes different aspects, from mechanical capabilities and autonomy, to the interaction between the robot and the user.

This chapter reviews the basic concepts and the evolution of the HMI and the emergence of HRI as a field of research. Then, the relationship between the robot autonomy and user interaction is presented, as well as the main problems regarding the HRI. Finally, an introduction to the problems faced in this Ph.D. will be outlined.

3.1 Basic concepts of Human-Machine Interaction

User Interface (UI) plays an essential role in countless business activities and daily actions. We can find several types of UIs in the modern workplace, however, including HMI, Human-Computer Interaction (HCI) and HRI. Those are designed to provide control mechanisms for one or more human operators, so they can control the device to which the HMI, HCI or HRI is connected. From here, we will use the term of HMI to encompass all the concepts, as we can consider the computer and the robot as a kind of machine, and the term will be used in this broader concept. This Ph.D. is moving between these concepts, as the user faces the computer to control the robot, and additionally part of the work can be considered as a way to control the robot directly.

The interest in improving the interaction between humans-computers and also computers-computers has grown in the last few years. This interest, especially in HMI, not only concerns industry, where the inclusion of different kind of robots are necessary and have been adopted quickly, but also the research community as it joins many scientific areas including artificial intelligence, computer vision, face recognition, motion tracking, etc. In both cases, the main goal is to interact naturally with the user, similar to the way human-human interaction takes place. The reason, in fact, is clear: most sophisticated machines are worthless unless they can be used properly by a user. Why a system is actually designed can ultimately be defined by what the system can do, or in other words, how the functions of a system can help towards the achievement of the purpose of the system. At this point, we can define the concepts of "functionality" and "usability". The functionality of a system is defined by the set of actions or services that it provides to its users. However, the value of functionality is visible only when it becomes possible to be efficiently utilized by the user. Usability of a system with a certain functionality is the range and degree by which the system can be used efficiently and adequately to accomplish certain goals for certain users. The actual effectiveness of a system is achieved when there is a proper balance between the functionality and usability of a system. Therefore, in the design of HMI, the degree of activity that involves a user with a machine should be thoroughly planned. The user activity has three different levels: physical (27), cognitive (28), and affective (29). The physical aspect determines the mechanics of the interaction between humans and computers while the cognitive aspect deals with ways that users can understand the system and interact with it. The affective aspect is a more recent issue and it tries not only to make the interaction a pleasurable experience for the user, but also to affect the user in a way that makes the user continue to use the machine by changing attitudes and emotions of the user.

As mentioned in (30), the existing physical technologies for HMI can be basically categorized by the relative human sense that the device is designed for. These devices are basically relying on three human senses: vision, audition and touch. Input devices that rely on vision are the most used kind and are commonly either pointing devices (31) (32), such as mice, joysticks, touch screen panels, graphic tablets, trackballs and pen-based input; or switch-based, which are any kind of interface that uses buttons like a keyboard (33). The output devices can be any kind of acoustic signals such as beeps, alarms, turn-by-turn navigation commands of a GPS device, visual display, or a printing device (34).

One of the most important factors of a HMI design is its configuration and modality. This concept concerns the number and diversity of inputs and outputs it provides. Thus, the term "multimodal" refers to the combination of multiple input devices that go beyond the traditional keyboard and mouse, like gaze, gesture tracking and speech processing. The exact number of supported input modes, their types and the way in which they work together may vary widely from one system to another. Additionally, multimodal interfaces can offer a number of advantages over traditional interfaces, offering a more natural and user-friendly experience and the ability to provide redundancy to accommodate different people and different circumstances.

So, at this point, we should question which is the main goal in HMI. There is not a single answer, because the main goal differs from one system to another and it is related to the specific purpose of the interaction. Nevertheless, there are some key points that all the systems should achieve:

- Develop usable products, easy to learn and effective to use.
- Provide an enjoyable experience.
- Involve users in the design process, improving the interactions between users and machines. Specifically, it should be concerned with methodologies and processes for

designing interfaces, methods for implementing interfaces and techniques for evaluating and comparing interfaces.

- Developing new interfaces and interaction techniques, including the development of descriptive and predictive models and theories of interaction.
- Increasing innovation in input techniques.

3.2 Evolution of the Human-Machine Interaction

HMI has evolved in pace with the industry and technology, adding more complex interfaces and ways to interact with the machines. The evolution concerns the robot type and its capabilities, the computers by themselves, which are used to control in some cases machines or robots, the architectures, the communication system used and the interaction way.

The first concept of teleoperation appears when Tesla decided to publicly demonstrate his radio system (35). According to Tesla, he had worked on the application of radio waves for remote control since 1893, when he published his basic radio plan. The idea to demonstrate this was by constructing a model boat (see Fig. 3.1(a)) that he operated by remote control, using radio pulses. The boat sailed in a large pool in the great hall of Madison Square Garden in September 1898, during the First Annual Electrical Exhibition. The first modern master-slave teleoperators were mechanical pantographs (see Fig. 3.1(b)), developed by the group working under R. Goertz in the late 1940s at the Argonne National Laboratory [1], where it was developed for the first nuclear reactor. In this case, teleoperation comes into play due to the need to manipulate nuclear material safely. The nuclear material was placed in a ”hot cell” where the operator could manipulate it outside the cell by remote handling. The visual contact with the target was through a protective window and/or a mirror. The mechanical manipulators were soon replaced by electro-mechanical servos. In 1954, Goertz’s team developed the first electro-mechanical manipulator with feedback servo control. After this, the teleoperation of manipulators and vehicles extended rapidly to new branches where the advantages of teleoperation techniques could be utilized.

In 1966, the Artificial Intelligence Center of SRI developed “Shakey the robot” (36), a mobile robot (see Fig. 3.1(c))[2] endowed with cameras and touch sensors and controlled by a large remote computer. This robot was considered the world’s first intelligent mobile robot, due to its ability to reason about its own actions instead of other robots that have to be instructed at each individual step to accomplish the full task.

Computers evolved even faster than machines and robots. In the 1950s, the batch processing era, users specified all details of a task on punch cards and executed the programs

[1] By Argonne National Laboratory - Archival Photographic Files, University of Chicago Library, Special Collections Research Center, CC BY-SA 4.0, `https://commons.wikimedia.org/w/index.php?curid=59580204`

[2] Image from `https://www.computerhistory.org/revolution/artificial-intelligence-robotics/13/289`

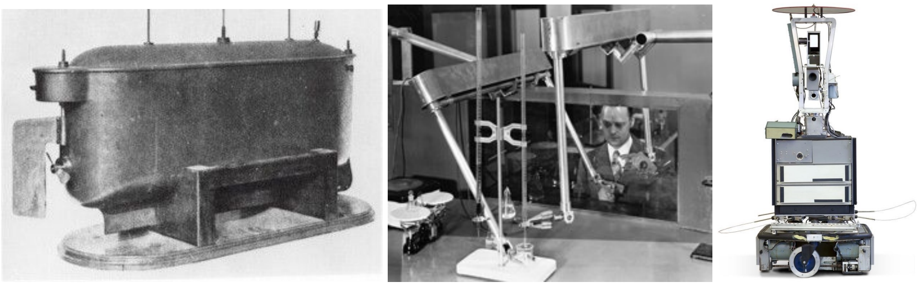


Fig. 3.1 Left: the boat created by Nikola Telsa. Center: R. Goertz during his teleoperation demonstration. Right: “Shakey, the robot” by SRI.

by processing those cards. This process was absolutely tedious and error-prone, not only because of those initial computers but also on account of human errors. With the appearance of command-line interfaces and more powerful computers, the HCI was improved in terms of interaction, due to the possibility of getting the command result in seconds. Despite the improvement, this system did not take full advantage of human perceptual, cognitive and learning abilities until the mid-1960s and early 1970s, when the first GUIs appeared and those abilities were leveraged.

Another breakpoint in robot history appears mid-1980s, related to behaviour-based robotics. These robots broke with the sense-model-plan-act (SMPA), which was the typical procedure of a centralized system (used in the robot “Shakey”), by replacing it with a distributed sense-response procedure to generate appropriate responses to external stimuli. These distributed responses produce behaviours that can generate more sophisticated responses, making the robot able to adapt faster to changes in the environment. Thus, the concept of “robot architecture” can be defined as a software system, which defines the flow of information, how it is processed, how it is transformed and how decisions can be taken. The rise of Artificial Intelligence (AI) opened a new paradigm in HMI relying on the support of multiple operations processed in parallel and continuous sensing-acting activities, instead of the SMPA procedure. Latest architectures are generally referred to as “hybrid” and keep the user in the control circuit, but in a high-level decision-making position and enriching the robot with more autonomy, as will be described in the following section 3.4.

3.3 Human-Robot Interaction as a specific research field

Although, at the beginning of this chapter, the HRI was included in a more general category such as HMI, this became a specific research field in the mid-1990s and early 2000s, as a multi-disciplinary field, joining researchers from disciplines like robotics, cognitive science, human factors or psychologists. According to (37), the HRI is quite different to HCI and HMI, insofar as the systems used have complex and dynamic control systems, exhibit autonomy and cognition, and are able to operate in changing environments (structured or not, and probably close to humans). All these factors involve different types of interactions: the physical structure of the robot, the environment where the robot will work and its ability to adapt itself, the communication system between the robot and the user, and the robot equipment (sensors, cameras, laser...).

Initially, the concept of HRI was associated with the teleoperation of factory robotic platforms. In fact, (38) defines telerobotics as: “direct and continuous human control of the teleoperator” or “a machine that extends a person’s sensing and/or manipulating capability to a location remote from that person”. Nevertheless, this definition is not valid today, as the evolution of technology, robots and their autonomous capabilities makes us go to a broader concept. The current definition not only refers to teleoperating a remote platform, but also to setting commands that will be carried out by the robot autonomously. According to (39), HRI can be divided into four areas of application:

- Human supervisory control of robots in the performance of routine tasks. This area refers to assembly lines, where the robots must follow specific pre-programmed routines or follow instructions set by a human operator, who waits for the robot’s feedback to update those instructions.
- Remote control of space, airborne, terrestrial and undersea vehicles. In most cases, these robots will be located in hazardous environments (as the aforementioned seafloor).
- Automated vehicles in which a human is a passenger. This area applies mainly to rail vehicles and commercial aircraft. Probably, in the short term, commercial cars will also be included here. Manufacturers like Tesla have improved considerably the autonomy of the car, currently at a level 2 of a maximum of 5.
- Human–robot social interaction, including robot devices to provide entertainment, teaching, comfort and assistance for children and elderly, autistic and handicapped persons.

3.4 The relation of autonomy and interaction

The concept of “Autonomy” could be defined as the ability to make their own decisions excluding any interference from others. Perhaps the most strongly human-centered application of this concept is in the notion of Level Of Autonomy (LOA), which describes to what degree the robot can act according to its own will. There are several formal definitions of LOA in the literature, but one of the most widely cited is by (1). Basically, the LOA defines the capabilities of the system to perform a task by itself and the authors propose a continuum line from the entity completely controlled by a human through the entity being completely autonomous. The full list is divided into 10 different categories:

1. Computer offers no assistance; human does it all.
2. Computer offers a complete set of action alternatives.
3. Computer narrows the selection down to a few choices.
4. Computer suggests a single action.

5. Computer executes that action if human approves.
6. Computer allows the human limited time to veto before automatic execution.
7. Computer executes automatically then necessarily informs the human.
8. Computer informs human after automatic execution only if a human asks.
9. Computer informs human after automatic execution only if it decides to.
10. Computer decides everything and acts autonomously, ignoring the human.

This list describes how autonomous a robot is, from a HRI point of view. Nevertheless, another approach was presented in (40) describing to what level the human and robot interact and the degree to which each is capable of autonomy. This new scale (see Fig. 3.2) puts emphasis on a mixed-initiative interaction and fits better with the robots described briefly in the introduction and in depth in section 2.2.

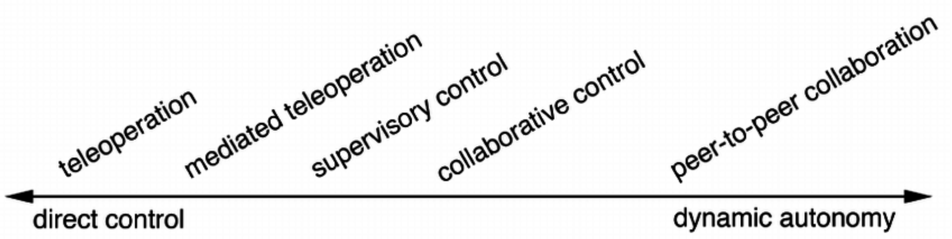


Fig. 3.2 Level of autonomy from the human interaction POV (image from (1)).

According to this scale, the previously introduced underwater robots can be classified:

- AUVs belongs to the supervisory control level. These robots are mostly used for vision-related missions, where they can be deployed under the water and the user waits until the robot finishes (depending on the robot, the user gets info from the robot and supervises the mission status). Nevertheless, only some conceptual robots developed at research centers have been endowed with small arms and have demonstrated the ability to manipulate objects. So, nowadays, these robots are not ready to perform complicated missions or overcome complex situations by themselves in case of a problem during the task execution.
- ROVs fit in with direct control mode (on the left of the classification) and depend completely on the user(s). This kind of robot is the most widely used in underwater environments. The unlimited power supplied from the vessel, keeping the pilot in the control loop controlling all the system and being responsible for all the mission, or the lack of restrictions concerning the robot size or the manipulators, make ROVs a powerful tool. Some of the missions are recovering objects from the seafloor (e.g. the airplane back-box), operating valves from off-shore platforms, archaeological missions or closing the oil tanks of sunken ships. On the other hand, the concentration and cognitive efforts they demand from the pilot, may lead to to substantial fatigue that in turn force them to stop the mission or execute faulty maneuvers due to concentration loss.

- HROVs can be placed in the middle (e.g. between supervisory control and collaborative control), due to the possibility that the robot can be operated as AUV or ROV, depending on the setup applied before the water deployment. Currently, in most of the missions, the robot can perform autonomously, keeping the user as a supervisor, getting task information and data through the tethering cable. Nevertheless, the real steps through collaborative control are some conceptual architectures developed in research projects.

Having said that, the hybrid robot can be presented as the best solution in terms of HRI and LOA, due to the fact that it can take advantage of the best of both AUV/ROV. Some of these advantages are related to this topics:

- In terms of functionality, the HROV can be endowed with more tools and features than usual when operating with only an AUV or ROV.
- The battery autonomy level moves to the maximum, due to the unlimited power supply coming from the vessel.
- If the connection between the robot and the vessel is wireless, the robot will improve its maneuvering.
- The user is still in the control loop, but in this case, the robot capabilities and the architecture will constrain the user responsibilities.

After analyzing the aforementioned benefits, we can observe that the most suitable approach relies on an architecture that also allows an hybrid control. This means that the mission could be performed by both the user and the robot at the same time. We can identify several examples: the mission can be carried out mainly by the robot until the user requests the control for a specific part of the mission or both the user and the robot can collaborate in the mission execution, with the robot and user taking on different responsibilities each of them (e.g. the robot takes control of the navigation system in order to keep the position, whereas the user operates a robotic arm).

3.5 Issues concerning teleoperated robots

All kinds of interactions present, in different ways and scope, problems between the user and the machine, computer or robot. Regarding robots and users, there are problems quite apart from with the autonomy of the robot, as described in 2.2. Teleoperated robots require a physical connection between the robot and the user that introduces specific problems such as communications delays and the user supervisor role generates the well-known cognitive fatigue inherent to master/slave control architectures (41). Autonomous robots still present problems, mainly regarding the control loop. These problems arise if the user needs to be inside it or modify the initial task.

At this point, one of the most relevant questions is: how is it possible to achieve a synergism between the robot and the user? The answer can be found by analyzing the information system and interaction patterns, bearing in mind the concept of “human-centered”. This term assumes that the information system is not only designed with users in mind, but also applies to all the possible techniques that make the system user-friendly (well designed and easy-to-use). (40) describe the following five attributes that affects the interactions between robots and users, and should be considered by the information system designer in order to achieve a successful exchange of information between robots and users:

- Level and behaviour of autonomy.
- Nature of information exchange.
- Structure of the team.
- Adaptation, learning and training of people and the robot.
- Shape of the task.

Going back to the previous question, "synergism" means that both parts should work together, as partners instead of substitutes. The information system should be focused on using robot technology to support and enhance human skills, instead of substituting robot skills for human skills. As it argued in (42), it is more interesting to develop and use robotic technology so that human skills and abilities become more productive and effective, such as removing the user from repetitive or risky tasks. He also suggests that the human-centered design of HRI needs to move forward, in an attempt to avoid common technology issues and consider issues such as task allocations between robots and users, or apply safety measures for the robot and the user. These issues need to be considered in the early stages of the technology designs, otherwise, these issues become secondary and have little impact on design considerations. If robots and users need to work side-by-side, as partners in a "collaborative control" mode, it is mandatory to improve the dialogue between them. This dialogue would be represented in different manners, not only as speech recognition, but also in a way to assign tasks and responsibilities, and establish a confident communication channel. Generally, this "collaborative control" means that the user is placed in a higher hierarchical position, giving new commands/tasks to the robot, but it is the robot that must fulfill the original mission with the new commands/tasks. However, if the user is not available within the time needed, the robot will use default behaviours to react to the situation. In this kind of control, the robot must be endowed with enough sensors and capabilities to adapt itself to the environment and mission changes, and all the information system and control architecture must also be reliable to support this.

3.6 Relevance of the Application Context

The work presented in this thesis has been focused on one specific domain (underwater robotics) and one specific application (remote control) according to the (39) classification, facing specific issues that may not affect other domains or applications. Some of the most relevant issues concerning underwater robots are:

- Robot design. Decisions taken during this phase are relevant because water resistance depends directly on the design and robot shape, the choice of the correct materials impacts on buoyancy properties, and the maximum depth achievable is limited by the materials and construction.

- Robot control system, which also includes issues related to maneuvering, navigation and station keeping during an intervention.
- Intervention capabilities, limited by the arm and gripper/hand type, and also the DOF of them.
- Visual inspection, which not only affects the camera system features or specific algorithms, but also the water properties like transparency, reflections, poor visual conditions, variations in lighting and chromatic distortions.
- Communication system, through tethering cable or wireless system.
- User feedback, needed to provide accurate information about the mission status and also to control the robot in case of problems.
- Power supply. This issue affects the mission duration in case of using a battery or the robot maneuverability in case of using a tethering cable.

A question arises from here: should we consider underwater robots research more difficult than aerial or ground robots? It is not an easy answer, but it is true that work in an underwater environment adds a certain complexity to common or already solved problems in other domains. Some of those aforementioned issues can be considered separately, but others may also be interconnected, as we can see in this example: interaction capabilities will be impaired if the feedback the user receives is not good enough and on time. This feedback is generated through sensorization (acoustic-sensory-video, each of them can introduce its own issues) and must be transmitted in some way to the user, depending on communication capabilities. Finally, in order to make a good decision by the user, a GUI must be used to interpret and assist the user.

Paying attention to specific interaction problems, we can find three types: the GUI, the telepresence feeling and input devices to manage the robot, and a way to control the autonomy of the robot.

There are several GUI approaches and proposals in the literature, but this type includes aspects such as information processing, user assistance, adaptive layout. Most of the robots are endowed with powerful computer systems, several sensors, cameras, and so on. All these components generate a huge amount of data that must be processed in both parts before the robot sends it to the user (improving bandwidth and throughput) and before they are shown to the user (presenting a lot of data to the user generates confusion and cognitive fatigue). Chapter 4 introduces and evaluates our proposal with respect to others GUIs.

Once the mission starts, the interaction between the robot and the user is limited by the robot type and constrictions, and the physical connection (or lack of it). From this moment, the use of third-person views GUI induces mental workload. The use of Virtual Reality (VR) systems, the exploration of the telepresence feeling, and the use of an immersive system benefit the way the user can be placed in a better position to understand how the robot is working and improving the knowledge or the mission status. Another aspect that can be improved is the input device used to manage the robot. The most known and used devices are joysticks, keyboards and mice, but the technology evolves and appears in the market devices that can track and estimate the pose of the human body and user's hands. This enables a new way to manage the robot, using a more natural interaction. Both approaches will be described in chapter 5, as well as explained an experiment carried out with users to compare a traditional setup with a third-person view GUI and the use of a joystick, against the use of an immersive GUI and a hand tracking device.

Finally, the control of the autonomy of the robot. As we classify the robots in section 2.2, we are moving from full teleoperated to autonomous robots, to the hybrid concept which can operate in both modes. Nevertheless, the main problem with these hybrid robots is the way the user delegates the mission to the robot and then can interact during the execution to modify the initial parameters with new ones or just taking total control during a period of time. This feature requires not only specific developments on both sides (the robot itself and the GUI), but also an architecture that enables this autonomy pass. Although the autonomy control seems to be only a feature, it is not because several parts of the architecture must be modified, such as the robot control system, the input device control, or the mission control system. Further details can be found in chapter 6.

Chapter 4
Developing Graphical User Interfaces to control an Underwater Vehicle

The user interface is one of the critical parts of any interactive system. It must be able to communicate the necessary information to the user and provide him/her with the adequate means to issue commands or perform a simple parameter settings as needed. In the particular cases of AUVs and ROVs, it supports the communication of the necessary information about the system/task/mission state for the user to take decisions, and the means for acting on that state for controlling the operation.

This chapter presents two different user interfaces designed and implemented to respond to the particular needs of some of the research projects coordinated by the IRSLab of the UJI. In these specific cases, the user interface must enable the user to perform the specification of underwater intervention tasks, that consist on the encapsulation of certain sets of instructions for controlling the embedded gripper, and perform underwater object manipulation guided through the available visual information. One principle explored here is that the user should be able to establish a dialogue with the robotic system based on high level instructions. In other words, the interface should have the ability to assist the user to perform the visual identification of the target and its gripping and/or manipulation, using actions from a predefined set. In addition, the interface must enable the user to test, in a specially designed simulator, any sub-task of the complete intervention prior to its execution on the real system and thus try to anticipate any difficulties that may arise.

4.1 Introduction

In the last century HMI has evolved from simple knobs and switches to GUIs, endowed with enough capabilities to allow users to fully control the associated systems. The versatility and expandability of such interfaces have brought to the user the possibility of adjusting almost every parameter of the underlying system, enabling thus the setting of optimal parameters for almost any situation in real time based on the contextualised information timely presented to the operator. Nevertheless, the development focus has traditionally been exactly on bringing to the user the possibility of controlling everything, frequently without taking into account ergonomic aspects. This increase in flexibility is typically reflected into more complex operations, which in turn require the operator to understand all the parameters of the system, thus augmenting the possibility of errors. In specific cases, this has lead to the inclusion of multiple operators dividing responsibilities, as commonly happens with the control of underwater robotic devices.

As part of this thesis work, two different concepts of GUI were proposed and explored. The first one, presented in section 4.3, consists in a GUI used during the Spanish research project RAUVI and the EU-FP7 project TRIDENT. The second approach, presented in section 4.4, resulted in a proposal developed for the Spanish research project MERBOTS.

In spite of the differences between the RAUVI and TRIDENT projects, the user plays similar roles in both of them. In both cases the missions are divided in two different phases, whereas each of them is initiated by a specific user intervention to establish the necessary settings. For the first phase (Fig. 4.1(left)) the user started by specifying the survey area, then the robot is deployed (1) and performs the survey task (2), and finally surfaces (3). In the second phase (Fig. 4.1(right)), using the images taken on the survey, the user can identify the ToI and select what kind of intervention should be performed. The robot is then again deployed (4) for searching for the ToI (5), performs the intervention (6) and docks back into the vessel platform (7).

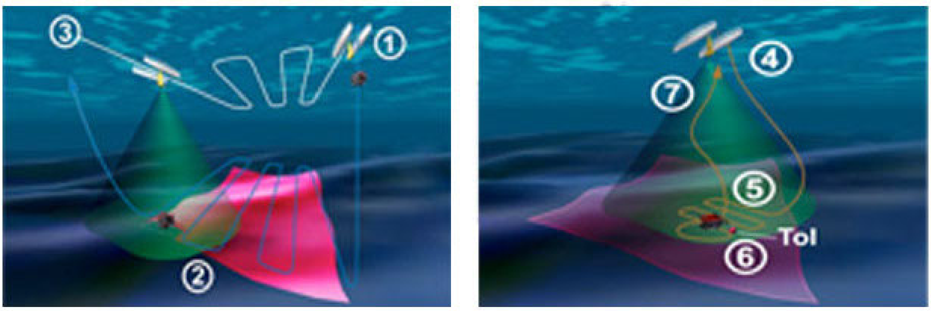
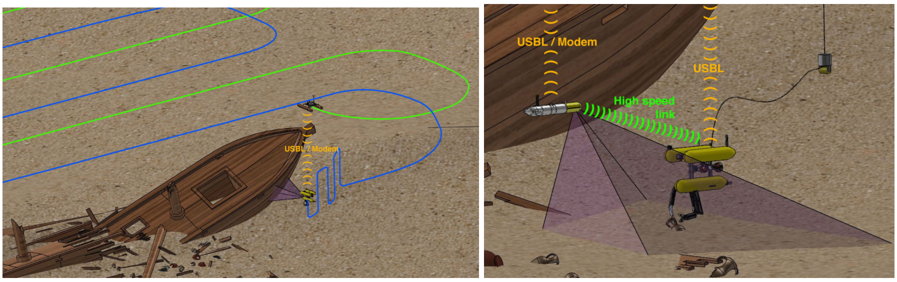


Fig. 4.1 The different phases of the TRIDENT projects and the role of the GUI: (left) phase 1, (right) phase 2

In the case of the MERBOTS project, two different robots were used, where the user could interact with both of them simultaneously, through a specially designed GUI. In this configuration, and depending on the mission type, the user may act as a supervisor (when the robot works as an AUV) or as a teleoperator (in the event of the robot working as a ROV). This way, the user specifies the survey parameters in the case of a "cooperative survey" (Fig. 4.2(a)), or the intervention parameters in the case of a "cooperative intervention" (Fig. 4.2(b)). As there are two robots in operation, with at least one camera on each, the user can choose at any time which camera stream to display for getting the best point of view of the ongoing operations (e.g. manipulation).



(a) Conceptual cooperative survey (b) Conceptual cooperative intervention

Fig. 4.2 Envisioned cooperation missions in the Spanish research project MERBOTS.

4.2 Developed Interfaces and Interaction Support for Underwater Robots

AUVs and ROVs are, both by definition and from the technological point of view, very distinct. Still, the most relevant differences appear on the interfaces they provide to control and manage the robots, and on the employed interaction modes and styles.

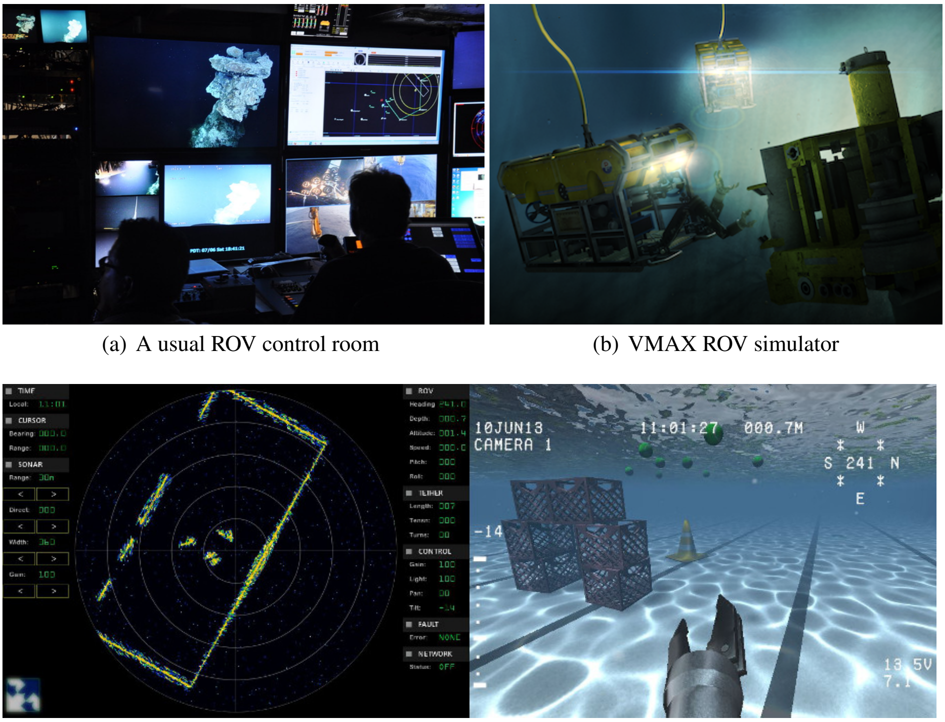
Usually, ROVs are controlled via master/slave architectures by pilots, who are experts both in the particular tasks and in the control of the specific robots. Different UIs have been developed specifically for each of these commercial robots available on the market. Basically, besides the input controls, those interfaces typically include multiple monitors for displaying as much information as possible (e.g.: camera output and sensor data), as we can observe in Fig. 4.3(a).

The interfaces designed for controlling underwater missions are by norm very complex. In addition to that complexity, the characteristics of the data connection between the robot and the pilot typically introduce another set of important constraints on the interaction efficiency and/or operations possibilities.

Although at first glance these aspects may seem to be unrelated, a closer look shows that there are in fact a number of important dependencies, as will be discussed hereafter. In the case of acoustic subsea communications, the underwater environment represents a huge workspace volume and the water properties do not benefit the acoustic transmission either. Thus, the transmission capabilities and the bandwidth used in data transmission is quite reduced. Nevertheless, when the robot uses a tethering cable, the bandwidth increases but the robot manoeuvrability becomes (more) constrained. Given the complexity of the tasks to be performed and the interfaces themselves, pilots need to do extensive training before taking control of real missions, as no errors are allowed or the vehicles my be simply lost. In fact, there exist commercial simulators for the purpose of training pilots before they take control of real missions. Two of the most well-known are ROVsim (Fig. 4.3(c)), from Marine Simulation LLC[1], and VMAX ROV Simulator (Fig. 4.3(b)) from Forum Energy Technologies[2].

[1] https://bit.ly/3B5Wn1v

[2] https://bit.ly/2XQCKw5



(a) A usual ROV control room

(b) VMAX ROV simulator

TIME
Local: 11:01
CURSOR
Bearing: 000.0
Range: 000.0
SONAR
Range: 30m
Direct: 000
Width: 360
Gain: 100

ROV
Heading 241.0
Depth: 000.7
Altitude: 001.4
Speed: 000.0
Pitch: 000
Roll: 000
TETHER
Length: 007
Tension: 000
Turns: 00
CONTROL
Gain: 100
Light: 100
Pan: 00
Tilt: -14
FAULT
Error: NONE
NETWORK
Status: OFF

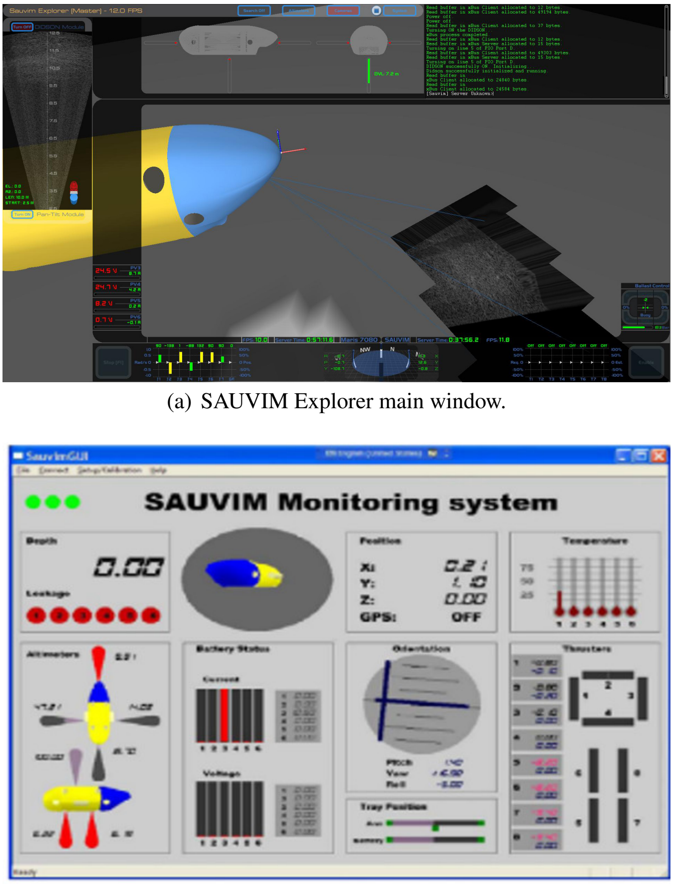
(c) ROVSim Pro simulator

Fig. 4.3 Graphical User Interfaces related to ROVs, from real to simulators.

In regard to AUVs and I-AUVs, which do not dispose of direct connections between robots and pilots, there are several approaches to support the interaction. In fact, the lack of a direct connection limits the interface options that can be used with the robot, but there will still be a GUI to specify the mission parameters and even supervise the mission through a video streaming in most cases. As previously mentioned, SAUVIM project can be considered a pioneer project in the underwater robotics field. In fact, this project focused the research not only on robot design, but also on these major components: adaptive and intelligent motion planning, automatic object detection and measurement, intelligent coordinated motion/force control, and a predictive virtual environment to specify and validate the mission.

Fig. 4.4 shows different screens of the GUI developed during this project. As we can observe in Fig. 4.4(a), the GUI is divided in three different sections: 1) the most relevant robot information (top and bottom of the image), 2) the reconstruction of the environment using different sensors (left), and 3) the simulator to validate the interventions (center) (8). Additionally, there is a section in the upper-right corner to interact with the robot sending commands using its own programming language. Fig. 4.4(b) shows the monitoring system that displays only the most relevant sensor and robot data.

In addition to developing its own interface and architecture for the SAUVIM project, more options were developed such as the Intelligent Task-Oriented Control Architecture



```
Read buffer in xBus Client allocated to 12 bytes.
Read buffer in xBus Client allocated to 49194 bytes.
Pover off.
Pover off.
Read buffer in xBus Client allocated to 37 bytes.
Turning ON the DIDSON...
xBus process completed.
Read buffer in xBus Client allocated to 12 bytes.
Read buffer in xBus Server allocated to 15 bytes.
Turning on line 5 of PIO Port D.
Read buffer in xBus Client allocated to 49303 bytes.
Read buffer in xBus Server allocated to 15 bytes.
Turning on line 5 of PIO Port D.
DIDSON successfully ON. Initializing...
Didson successfully initialized and running.
Read buffer in
xBus Client allocated to 24840 bytes.
Read buffer in
xBus Client allocated to 24584 bytes.
[Sauvim] Server Unknown>
```

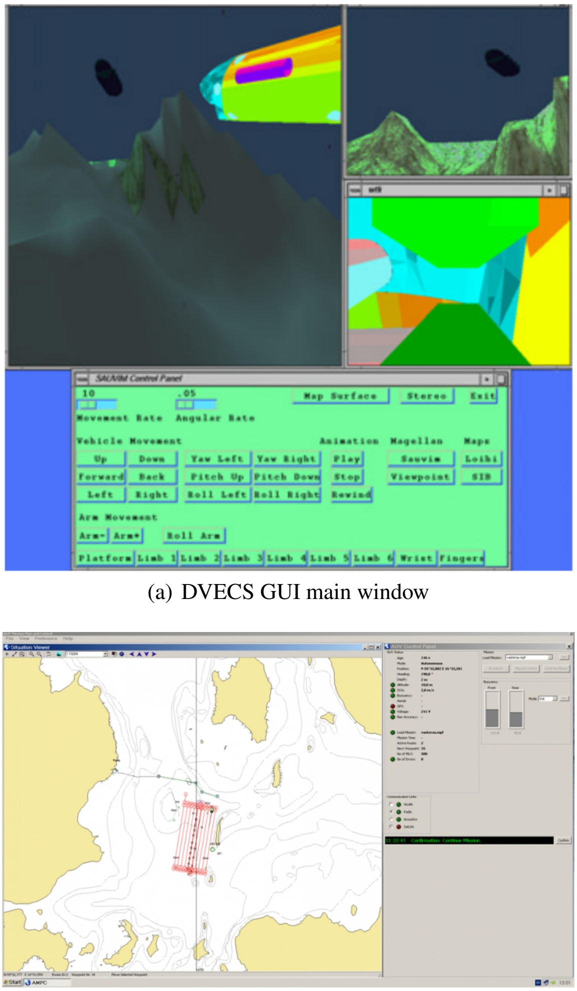
(a) SAUVIM Explorer main window.

(b) SAUVIM monitor system.

Fig. 4.4 SAUVIM graphical user interfaces.

(ITOCA) architecture (43). This hybrid control system, designed specifically to be used in SAUVIM, is organised in three layers: execution, control, and planning. The execution layer groups the sensors and actuators, the control layer contains the drivers of the low level vehicle and handles the non-linear vehicle and arm control, and the planning layer is responsible for high-level control of the vehicle during the mission, as well as being responsible for mission planning, execution, and supervision.

The coupling of simulators to user-interfaces has been growing in recent years, and we may take as a representative example the ‘DVECS’ (44) (45). Its interface was built on different layers and modules, which included the use of OpenGL graphics libraries to generate seafloors, vehicles and obstacles, and Open Inventor 3D Toolkit to recreate the 3D world. As we can observe in Fig. 4.5, the interface is divided into sub-windows, representing the images obtained by the front camera of the AUV, the manipulator camera, the 3D reconstruction and the control panel.



(a) DVECS GUI main window

(b) ALIVE GUI main window

Fig. 4.5 DVECS and ALIVE interfaces.

There were also other projects, such as AMADEUS (4) or ALIVE, but no information related to its interface could be found in the literature or other documentation publicly available. The EU-funded ALIVE project had similar objectives to those of the SAUVIM project, which were to design an AUV capable of performing light (small) tasks. In this case, the vehicle was not physically connected to the base station, using instead an acoustic modem. This supported the communications that enabled the operator to receive images and data from the sensors, that were used to evaluate the mission to be performed by the small manipulator.

There is indeed a common aspect on all the previous referred systems: the robot is controlled by an expert user that has a big responsibility due to the complex environment, the technology involved, and the elevated costs need to carry out an intervention.

4.3 RAUVI and TRIDENT user interface

Section 4.1 already introduced the two first underwater robotic projects coordinated by the IRSLab and the works dealing with the HRI, the arm-robot integration, and the grasping capabilities. As far as interaction and interfaces are concerned, these projects shared the common objective to assist the user during mission specification and robot operation, avoiding the need of learning specific programming languages or using a highly complex GUI to control the robot. Instead, the user should be able to specify the mission following a few steps, as follows:

1. Selection of the area to be surveyed and definition of the survey parameters.
2. Deployment of the robot and initiation of the automatic survey execution.
3. Recovering the robot, getting the images obtained during the survey task and generation of the photomosaic.
4. Selection of the ToI and definition of the grasping parameters.
5. Simulation of the intervention and analysis.
6. Intervention execution, if the previous step has been validated.

With this type of approach the mission becomes less dependent on the expertise level of the pilots, and, although they should still be present due to safety rules, their role becomes more those of observers. This has the clear advantage of simplifying the use of underwater robots and making them accessible to less experienced users, as they will not be put under the typical pressure of the use of the more complex interfaces. In fact part of this pressure comes from the need to control an underwater mission through third person views, or the tracking and interpretation of information scattered along multiple displays, frequently hard to read.

4.3.1 Implementation considerations

In the early stages of the RAUVI project its architecture was implemented based on Linux. There was indeed a change that came from the choice of the simulator to include, which was built and running on Microsoft Windows. This simulator was developed by Gianluca De Novi (46), from the University of Bologna (Italy) and does not use any commercial or opensource engine, such as Object-Oriented Graphics Rendering Engine (OGRE) or Open Scene Graph (OSG), but instead uses one developed by themselves.

Additionally, the Microsoft Robotics Developer Studio (MRDS)[3] was released a few months before the project started and it was considered an advantage to include it as part of the project architecture. The MRDS is a freely available ṄET-based programming environment for building robotics applications. Some of the features included are: a visual

[3] `https://msdn.microsoft.com/en-us/library/bb648760.aspx`

programming tool, Microsoft Visual Programming Language for creating and debugging robot applications, a web-based and windows-based interface, a 3D simulator (including hardware acceleration), easy access to some of the most common robot's sensors and actuators and C# as the primary programming language.

The RAUVI and TRIDENT user interface, which was named *UnderWater User Interface (UnderWater User Interface (UWUI))*, was developed keeping in mind that it should be connected with both aforementioned programs. It was developed on Windows using C# and two different libraries:

- AForge.NET[4], a computer vision and artificial intelligence library, which was mainly used due to the good performance in detecting and labelling blobs[5] into the image, and the ease of integration with the .NET framework.
- EmguCV[6], a wrapper of OpenCV, because it could not be used with C#.

4.3.2 Architecture integration

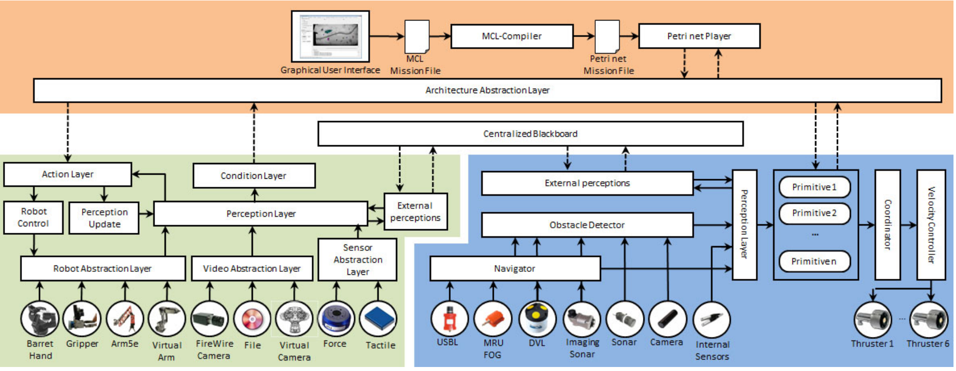


Fig. 4.6 RAUVI general diagram of the software architecture.

The RAUVI project architecture was already defined in (47), where it is explained in details. The architecture integrates two control architectures initially designed and implemented independently, one to control an AUV and the other to control an underwater manipulator (see Fig.4.6). System's integration is achieved by means of a Mission Control System (MCS) allowing the user to program the mission, uploading it into the I-AUV as well as launching its real-time execution. The MCS is the part of a system's architecture that is in charge of coordinating the high level phases to be carried out by the

[4] http://www.aforgenet.com

[5] Blob: it stands for Binary Large OBject and refers to a group of connected pixels (usually as part of the same object) in a binary image. The term *large* indicates that only objects of a certain size are of interest and that *small* binary objects are usually noise.

[6] http://www.emgu.com/wiki/index.php/Main_Page

vehicle and the manipulator in order to fulfil a predefined mission. Each high level mission phase is denominated as a task that can be executed by means of enabling/disabling some vehicle/manipulator primitives. The MCS defines how the mission is divided into a set of tasks and how primitives are combined to fulfil each task. The development of a MCS for an autonomous system lies at the intersection between a Discrete Event System (DES) responsible for enabling and disabling basic primitives when some events are produced and the Continuous State Dynamic Control System (CSDCS) used for every primitive to achieve a specific goal. The DES must ensure the consistency of the resulting controller avoiding driving the autonomous system into a deadlock situation and simultaneously ensuring the reachability of one of the final states described in the mission plan. Thus, the approach to the MCS is based on the Petri net formalism (more details about this can be found in (48)) for the DES representation to model, program and execute missions. When connecting the two independent architectures, additionally to the MCS, a centralized blackboard has been added to connect both systems. Through the blackboard, it is possible to interchange relevant information between the architectures in order to execute their primitives in a more efficient way while retaining the distributed nature of the system's design.

A. Manipulator architecture

Whereas many software architectures have been proposed in the literature in the area of mobile robotics (49), there are only a few contributions in the field of manipulators (50) (51). As pointed out by (52), a control architecture for manipulators should be endowed with: (i) the manipulation ability, in charge of acting on the environment, (ii) the sensory ability, capable of obtaining information from the world, (iii) the data processing ability, which processes data of system activity, and (iv) the intelligent behaviour ability, capable of modifying system behaviour according to external information. This means that a software architecture for manipulation must include sensor information processing modules feeding low-level manipulation controllers and an event-driven layer on top of these controllers providing the robot with behaviour. This layer should be in charge of selecting the most appropriate low-level controllers according to the state of the task.

Taking these goals into account, our manipulation architecture is structured into two layers: a reactive layer containing perception and action modules, and a deliberative layer (the MCS) shared with the vehicle architecture, where the appropriate manipulation actions are chosen according to the intervention mission. An additional layer called Architecture Abstraction Layer (AAL) provides unified interfaces for sending control commands (actions) to the manipulator and for reading conditions (events) generated by the perception layer. In regard to the reactive layer, its modules address specific functionalities, as follows:

Perception module: this refers to both external and internal perception. The external perception includes, for example, the force at the wrist or the robot's perception of the object's pose. Internal perceptions include proprioceptive information and the references, conditions and immediate goals of the robot. Some examples are the joint values, the end-effector position, the maximum force that the robot's fingers can support, the condition that compares the current force at the fingers with respect to the maximum allowed, the position in space where the robot has to immediately move its hand, etc. This module is also responsible by dealing with the detection of specific conditions such as when the force at the fingers is greater than a given threshold, or the hand has reached a target reference, etc. These conditions generate events which are in turn sent to the MCS through the AAL.

Action module: the manipulation architecture distinguishes between two different types of actions: (i) control actions, where data coming from perceptions are used as feedback to execute control algorithms in a reactive manner, and (ii) perceptual actions, that update the robot's internal variables, as for example, modifying a task reference, a target object, etc. A control action takes as input a subset of perceptions and generates a control vector, which is sent to the manipulator. Sensor-based controllers integrating vision, force and tactile information have been implemented as control action modules into this manipulation architecture.

B. Vehicle architecture

The vehicle architecture has the task of guaranteeing the AUV functionality. From the implementation point of view, the real-time POSIX interface, together with the ACE/TAO CORBA-RT ORB, have been extensively used to develop the architecture as a set of distributed objects with soft real-time capabilities. These objects are distributed between two onboard PCs. The architecture is composed by a base system and a set of objects customized for the desired robot. There are classes providing soft real-time capabilities, this is guaranteeing the period of execution of the periodic tasks such as the controllers or the sensors. Another important part of the base system is the loggers. A logger object is used to log data from sensors, actuators or any other object component. Moreover, all the computers in the network are synchronized and hence, all the data coming from different sensors can be time-related. The AUV software system is conceived as a hybrid control architecture in two senses: 1) merging reactive-deliberative control strategies (53) (54) and 2) merging a DES with a CSDCS (55). The vehicle architecture is divided in three modules: Robot interface module, Perception module and Control module.

Robot interface module. This is the unique module that contains software objects that dialogue with the hardware. There are basically two types of components: sensor objects responsible for reading data from sensors and actuator objects responsible for sending commands to the actuators. Sensor objects include drivers for the following sensors: a Doppler Velocity Log (DVL), an Ultra-Short Base Line (USBL), an Imaging Sonar, a Motion Reference Unit (MRU) equipped with a Fiber Optical Gyroscope (FOG) (MRU-FOG), two cameras, a depth sensor and an echo sounder. There are also objects for the safety sensors like the water leakage detectors, internal temperature and pressure sensors that allow for the monitoring of the conditions within the pressure vessels. One actuator object for every thruster is also included. A virtual version of every component allows the transparent connection of the robot control architecture to a real-time graphical simulator allowing performing hardware in the loop simulations (56).

Perception module. This module contains three basic components: the Navigator, the Obstacle Detector and the centralised blackboard. The Navigator object has the goal of estimating the position and velocity of the robot combining the data obtained from the navigation sensors. The Obstacle Detector is used to detect the relative position of the robot with respect to the obstacles detected in the world. The Control module uses these data, keeping the behaviours independent of the physical sensors being used. The centralized blackboard is a shared database with the manipulator's architecture, which is used to share the vehicle and manipulator perceptions. This module writes the main vehicle perception (position, velocity, detected obstacles, etc.) into a centralized database using TCP/IP and reads from this database the main perceptions communicated by the manipulator. Some primitives can use these perceptions in the control module.

Control module. The control module receives sensor inputs from the perception module and sends command outputs to the Actuators residing in the Robot Interface Module. Several behaviours (also called actions or vehicle primitives in the literature) are defined to perform the survey, the approach and the inspection steps of an intervention mission. Behaviours can be enabled and disabled and their parameters can be changed by means of actions sent by the MCS through the AAL. Also, events can be produced by the behaviours to announce that a goal has been reached or a failure has been detected within the vehicle architecture.

C. Mission Control System

The MCS is the part of the control architecture in charge of defining the task execution flow, which means manipulator actions and/or vehicle primitives, to fulfil a mission. Concerning this flow, the mission programmer defines how these actions/primitives are executed and how the tasks are combined, in order to fulfil each task and the whole mission respectively.

The MCS was developed as generically as possible, allowing for an easy tailoring of different control architectures. To achieve this goal, the proposed MCS presents a clear interface with any particular control architecture based on actions and events. Between the MCS and the vehicle/manipulator architectures there is an AAL that adapts these actions and events to the corresponding instances of the target architecture. The AAL depends on the control architecture being used, allowing the MCS to remain architecture-independent. Tasks are the basic building blocks of our system. They are defined using the Petri Nets formalism (57) and communicate with the hardware by means of actions and events sent through the AAL. Actions are associated to transitions in the Petri net and they are executed when these transitions fire. Events communicate conditions detected by the vehicle architecture to the mission Petri net. Every event is associated with one or more places. If a specific event is triggered by the control architecture, all the related places receive a token.

It is possible to compound several tasks using control structure Petri nets and the resulting net will also exhibit the properties defined for the architecture. It is worth noting that it is not necessary to span the whole reachability tree of the compounded Petri net to ensure the deadlock free as well as the state reachability properties. Spanning it, would have a very high computational cost. The Petri net missions, implemented according to these rules, are able to progress from their starting state to their exit state without sticking in a deadlock. It can be proved (58) that the set of this class of Petri nets is closed with respect to the compounding operation. Hence, the result of compounding two Petri nets with these properties is a new Petri net, for which the same properties hold by construction without the need of further verification. Instead of using graphic tools to describe these Petri nets, our approach uses a Mission Control Language (MCL) (48), able to transform a friendly high-level language into a Petri net. A GUI is used to identify the target, using a set of images previously gathered by the I-AUV, to specify its localization, its grasping points and the task to be performed. The GUI also allows the programming of the MCL intervention mission to be uploaded into the I-AUV for execution.

4.3.3 Image input characterization

Underwater environments represent a challenging and hostile scenario for many reasons. Some of them are the light variation of conditions (with respect to the depth), the interaction of fishes and plants, or even the water pressure and corrosion. Additionally, we should bear in mind that some of these problems may appear simultaneously: if the mission can not be achieved in one single stage, the conditions found during the first stage could be significantly different to the next one.

Given the differences in quality and definition between images acquired inside and outside the water, the image characterization was one of the first problems dealt by the GUI development. Hence, a set of real photomosaics and images have been used during the developing phase of the interface. The images used in this phase can be described as follows:

- Figs. 4.7 show photomosaics generated by the UdG after a real survey mission. Even with the technological evolution concerning computer performance, the file size and image resolution of the photomosaics may be subject to compromises. In fact, typically the higher resolution of the images the better, and camera technology has been evolving considerably to support it, but larger resolutions imply increased computational efforts and transmission times on very limited bandwidth communications. Thus, reducing the file sizes benefits the time needed to process and transmit the images, and therefore contributes to the interaction and enhances the user experience managing the GUI.
- Figs. 4.8 show images obtained by the UIB while using different tools (such as a screwdriver and a hammer) over a photomosaic build for the floor of a pool. This set of images was taken with the camera configuration to be used in the robot during the trials. Some of these images were processed by specific algorithms developed by the UIB.
- Images obtained in a swimming pool using an action camera. These images were taken not only focused on the light conditions (lighter than the images from the other sets) but also paying attention to the tools used as an example. These tools were objects presented in the robot development roadmap as tentative ToI.

4.3.4 UWUI overview

Since the very beginning, the design and development of the interface were focused on its use by non-expert users. It includes access to built-in image processing algorithms, which can be used to assist the user to specify mission parameters. Most of these parameters are defined with usable default values or values that are automatically computed from the situation information, leaving the user the option to modify them if needed. Despite the inclusion of several options accessible through the menus, the goal was initially to support the execution of the survey mission, through the specification of the navigation data and the intervention mission, via the specification of the object recovery or a hooking intervention procedure.

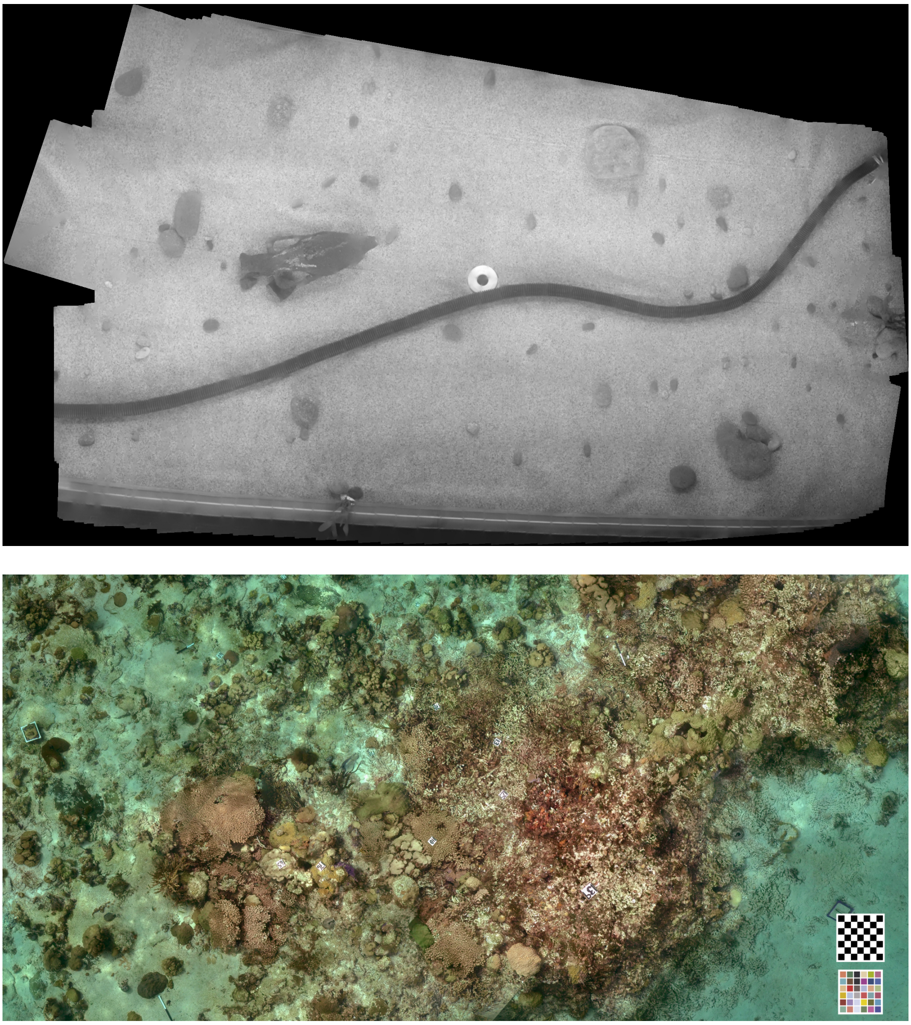


Fig. 4.7 Example of photomosaics generated by the UdG and used during the GUI development.

For the sake of simplicity and not confusing inexperienced users with excessive displayed options, the initial phase (see Fig. 4.9) shows only a few buttons enabled:

- Predefined Intervention Task: this displays the intervention mission types.
- Open an image: the user will select the image or photomosaic to work with.
- Connect to the simulator: sends the intervention mission data to the simulator.

Once the user chooses an option, it enables and displays a new panel (4.10) with several options related to the mission type. Nevertheless, and although the GUI design has been focused on non-expert users, to avoid precluding an experienced user from having control of lower level functionalities, the *Task options* panel offers, in the last tab, several options

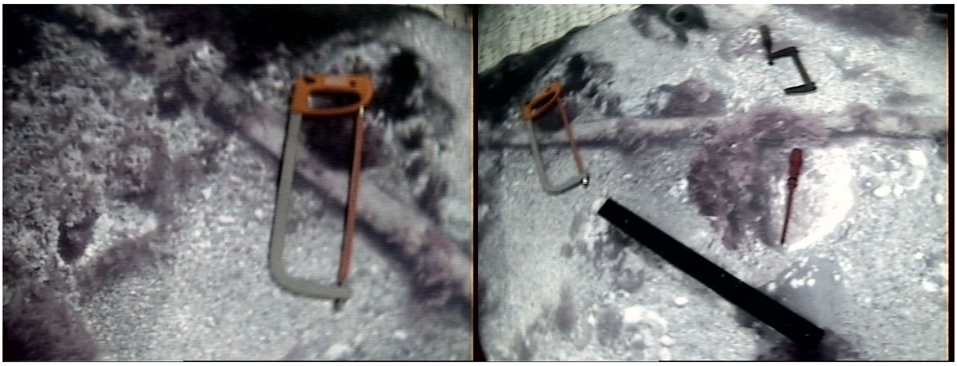


Fig. 4.8 Example of images obtained by the UIB and used during the GUI development.

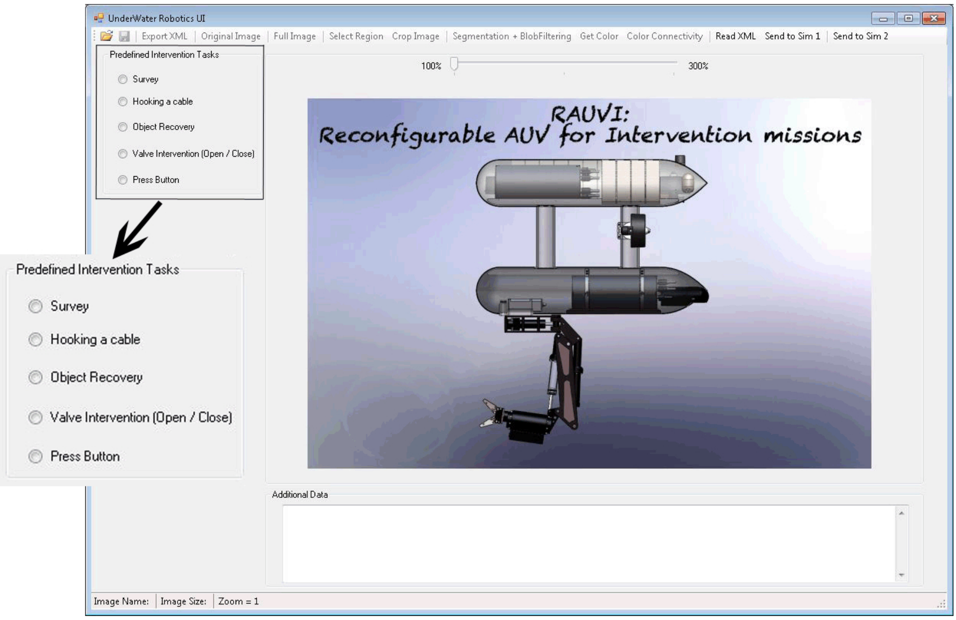
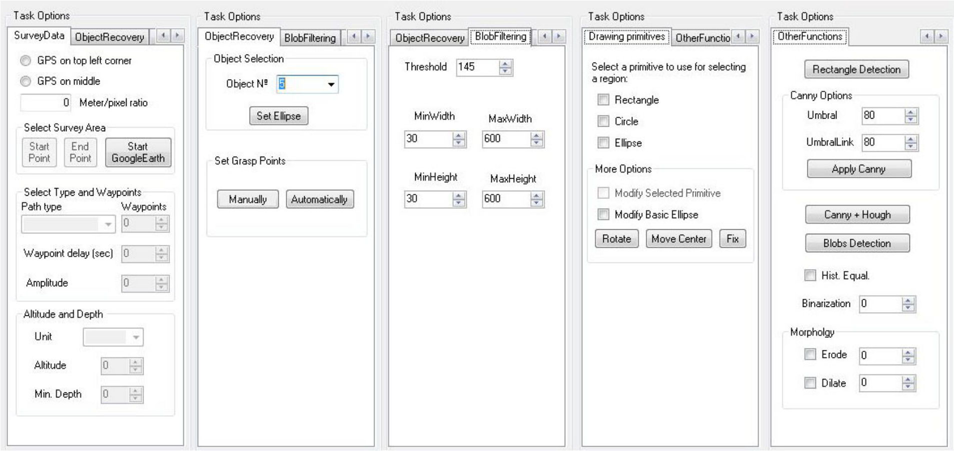


Fig. 4.9 Initial GUI appearance.

to process the image: geometric form detection, algorithms to detect edges, image binarization and histogram equalization, or erode-dilate the image. Applying some of these features, the user can get better results in the target identification process.

Finally, when the mission specification has been finished, the user clicks on the button to generate an XML file, which contains the relevant information for performing the mission. The beginning of this file is the same for all the interventions mission types, containing information related to the image used and intervention type. The rest of the file depends on the intervention type. In a *Survey task*, this section contains a list of way-points and other relevant data (e.g. robot depth, altitude...), meanwhile in an *Intervention task*, this section



Task Options
SurveyData | ObjectRecovery
○ GPS on top left corner
○ GPS on middle
0 Meter/pixel ratio
Select Survey Area: Start Point | End Point | Start GoogleEarth
Select Type and Waypoints: Path type, Waypoints 0
Waypoint delay (sec) 0
Amplitude 0
Altitude and Depth: Unit, Altitude 0, Min. Depth 0

Task Options
ObjectRecovery | BlobFiltering
Object Selection: Object Nº 5
Set Ellipse
Set Grasp Points: Manually | Automatically

Task Options
ObjectRecovery | BlobFiltering
Threshold 145
MinWidth 30
MaxWidth 600
MinHeight 30
MaxHeight 600

Task Options
Drawing primitives | OtherFunctio
Select a primitive to use for selecting a region:
□ Rectangle
□ Circle
□ Ellipse
More Options
□ Modify Selected Primitive
□ Modify Basic Ellipse
Rotate | Move Center | Fix

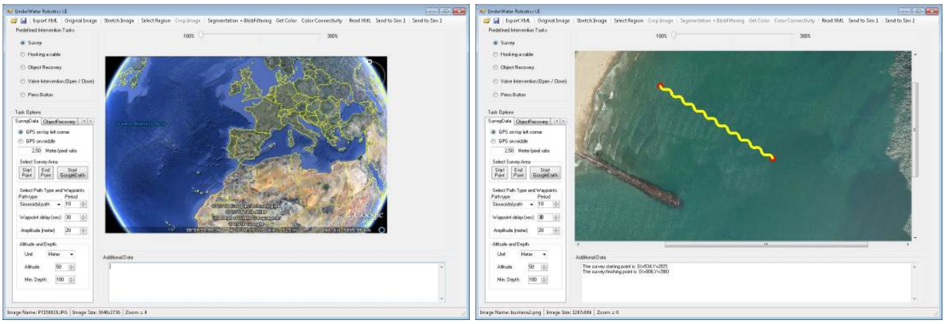
Task Options
OtherFunctions
Rectangle Detection
Canny Options: Umbral 80, UmbralLink 80
Apply Canny
Canny + Hough
Blobs Detection
□ Hist. Equal.
Binarization 0
Morpholgy: □ Erode 0, □ Dilate 0

Fig. 4.10 *Task options* panels, more options for the user.

contains information about the target: size, orientation, position and grasp points/hooking data.

4.3.5 Defining a Survey task

A *Survey task* consists on a navigation task, where the robot must follow a trajectory. In this phase the interface allows the user to define the trajectory and its parameters, either using GoogleEarth (see Fig. 4.11(a)) or loading a geo-referenced map (see Fig. 4.11(b)).



(a) Survey using GoogleEarth

(b) Survey using a georeferenced map

Fig. 4.11 Different ways to specify a *survey task*.

The possibility of using GoogleEarth provides some advantages: the maps are always maintained up-to-date by Google, so there is no work for the user; it is a free service, unless high-detail maps are needed; the user can customise the selected view with more detailed data or add images into the map; the user can store information and maps in order to use it in offline mode. On the other hand, there are also a few drawbacks: an Internet connection is needed and it does not work properly if the area of interest is too small. In a such case, if a custom map is available, it can also be loaded and used in this interface.

In order to specify the survey area using a georeferenced map, the user selects the initial and final point of the survey, just doing a click on the map for each point. Then, in the *Task Options panel*, the user selects all the needed parameters:

- GPS location: the loaded map must contain a georeferenced point and the user needs to specify where it is located (in the top left corner or in the middle of the image).
- Meter/pixel approximate ratio: needed for further calculations. For instance, when the user draws a rectangle on the image, the GUI calculates the position of each waypoint in real life.
- Type of path the robot has to follow: a one-way trip, a round trip or sinusoidal (sweeping) trip. Depending on this selection, the user indicates some specific parameters such as the amplitude of the sinusoidal path.
- Waypoint: the number of georeferenced points, which are part of the path.
- Waypoint delay: number of seconds that the robot needs to be stopped in each waypoint, a parameter needed for the MCS.
- Unit: the user specifies if the units used are miles or meters.
- Altitude: means the minimum distance to the seafloor.
- Depth: maximum distance from the sea level that the robot can reach.

4.3.6 Identifying and selecting the Target of Interest

According to the envisioned project concept, we assume that the robot has finished the first phase (the survey) and the user has generated a photomosaic of the seabed using the acquired images. Once it has been generated, the user may load it in order to select the ToI.

Algorithms that work well with one mosaic, may fail when applied to another photomosaic due to different visibility conditions, backgrounds, etc. For this reason, the main goal of the application is to assist the user in identifying all the possible targets detected in the photomosaic. So, it offers the user three different types of use, where the system autonomy increases as the user responsibility decreases.

According to (59), we can characterize the object using its geometric properties. For global shape characterization, the system uses the centroid, its orientation, and the inertia axes (see Fig. 4.12). These features are calculated using the boundary box and the best fitting ellipse, which are drawn on the object. The reason for using a best fitting ellipse is that it captures the size and shape of the object.

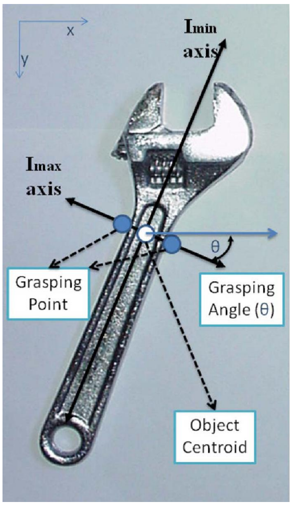


Fig. 4.12 Relationship between the object properties and the information we can get to grasp it.

4.3.6.1 First ToI detection mode

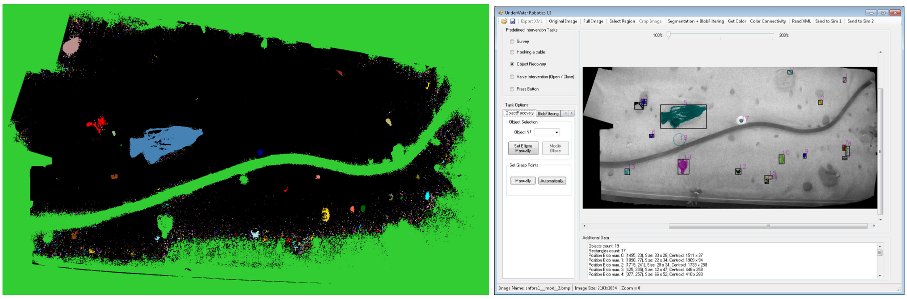
This first interaction mode uses a function to identify the ToI automatically. This function segments the image using a default value calculated after different tests with the aforementioned set of images. If the results are not satisfactory, the user may click on a possible target, achieving the greyscale color value as a new value for the segmentation process.

If the user does not find the ToI during the first iteration, he/she can tune up the segmentation value or the filtering parameters using the settings located in the *Task options* panel. In order to reduce the quantity of image processing and color variations, the user can crop the image around an area. Once the photomosaic has been segmented, a blob search is performed.

Fig. 4.13(a) shows the algorithm result without filtering. As can be seen, there are many objects detected as a blob, although most of them should not be considered. Some examples of these objects are small stones or heaps of sand. Additionally, if the photomosaic has some imperfections, those would also be detected. So, in order to prevent the detection of those false objects, the algorithm will output only the objects that fit inside a predefined range of dimensions, discarding all the objects smaller or bigger than those limits. Fig. 4.13(b) shows the result once the filter has been applied. In case of need, the user can modify the range of this filtering using the *Task options* panel.

Finally, as all the images are labelled with a number, the user can select the ToI identifier from a dropdown menu located in the *Task options* panel. Automatically, the GUI shows the best fitting ellipse mentioned before over the ToI. Although this solution will be suitable for most of the cases, maybe the ellipse is not perfectly orientated or even the user would prefer to move it, in order to modify some of its properties (e.g. center of mass). This can be done using the option placed in the *Task options* panel.

At this moment, the ToI has been identified and selected, storing the position and size.



```
Objects count: 19
Rectangles count: 17
Position Blob num. 0: (1495, 23), Size: 33 x 28, Centroid: 1511 x 37
Position Blob num. 1: (1898, 77), Size: 22 x 34, Centroid: 1909 x 94
Position Blob num. 2: (1719, 241), Size: 28 x 34, Centroid: 1733 x 258
Position Blob num. 3: (425, 235), Size: 42 x 47, Centroid: 446 x 258
Position Blob num. 4: (377, 257), Size: 66 x 52, Centroid: 410 x 283
```

```
Image Name: anfora1__mod__2.bmp | Image Size: 2183x1034 | Zoom = 0
```

(a) Object detection without filtering

(b) Object detection with filtering and labeling

Fig. 4.13 Objects detected automatically using the functions using the first interaction mode.

4.3.6.2 Second ToI detection mode

If the previous mode does not work, the user can use a color connectivity algorithm. Once the user clicks on the image, the algorithm performs a recursive search around the already stated point clicked by the user, searching for pixels with similar colors. The algorithm considers two pixels with a similar color, when the difference of their *RGB* values are less than a given threshold. The result of this algorithm is a grey-scale image, as shown in Fig. 4.14(c), where all the connected pixels are painted with a darker gray. Finally, the algorithm uses this image to get the blob containing all those pixels and the best fitting ellipse is displayed over the ToI. The operation steps are similar to those of the previous mode.

4.3.6.3 Third ToI detection mode

Finally, if neither of the previous modes produce acceptable results, the user can identify and select the ToI manually, just by drawing, and rotating if needed, an ellipse over the ToI (see Fig. 4.15).

4.3.7 Defining the intervention task

Although more possibilities exist, we implemented only the two intervention tasks that were considered in both RAUVI and TRIDENT projects: object recovery and hooking. Let's consider that the *Survey task* has been finished and the photomosaic has been created, as explained before. At this moment, the user needs to select the intervention task between these two options and specify the data needed. The following subsections explains the process, just after the ToI has been identified and selected.

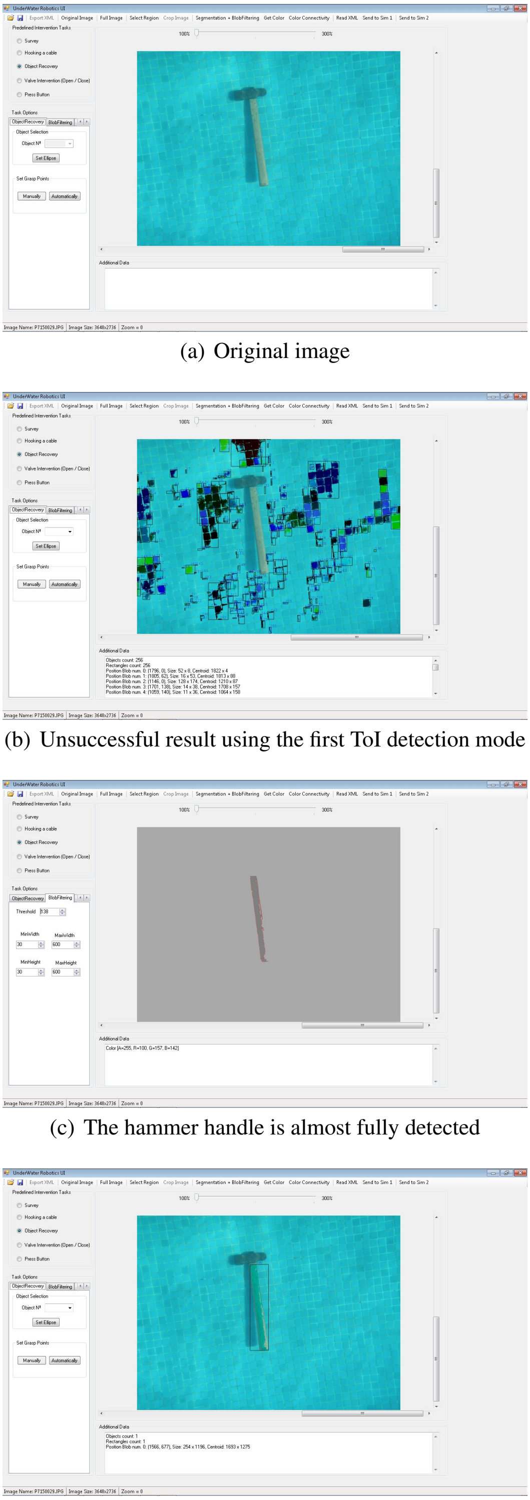


Image Name: P7150029.JPG | Image Size: 3648x2736 | Zoom = 0

(a) Original image

```
Objects count: 256
Rectangles count: 256
Position Blob num. 0: (1796, 0), Size: 52 x 8, Centroid: 1822 x 4
Position Blob num. 1: (1805, 62), Size: 16 x 53, Centroid: 1813 x 88
Position Blob num. 2: (1146, 0), Size: 128 x 174, Centroid: 1210 x 87
Position Blob num. 3: (1701, 138), Size: 14 x 38, Centroid: 1708 x 157
Position Blob num. 4: (1059, 140), Size: 11 x 36, Centroid: 1064 x 158
```

(b) Unsuccessful result using the first ToI detection mode

Threshold: 138
MinWidth: 30
MaxWidth: 600
MinHeight: 30
MaxHeight: 600

```
Color [A=255, R=100, G=157, B=142]
```

(c) The hammer handle is almost fully detected

```
Objects count: 1
Rectangles count: 1
Position Blob num. 0: (1566, 677), Size: 254 x 1196, Centroid: 1693 x 1275
```

(d) A blob is generated with the previous information

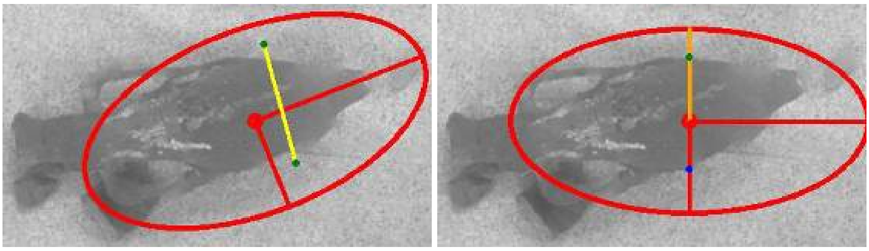
Fig. 4.14 The second ToI detection mode uses a color connectivity to find all the pixels with similar color.

After defining the grasping points or the hooking data, the user can export the *intervention task* into a XML file to finish the specification process.

4.3.7.1 Defining the grasping points for an *Object recovery*

In order to select the grasping points, the user has two possibilities: to execute a function to calculate them or by drawing a line, where the extreme points represent the grasping points (Fig.4.15(a)).

The function used to calculate the grasping points follows the criteria explained in (59). To sum up, the algorithm used in this function searches the first points placed out of the blob, starting on the centroid of the ellipse and following the ellipse minor axis in both directions. When this function detects these points, it stops and marks those points as *grasping points* (see Fig. 4.15(b)).



(a) Grasping points defined manually (b) Grasping points defined automatically

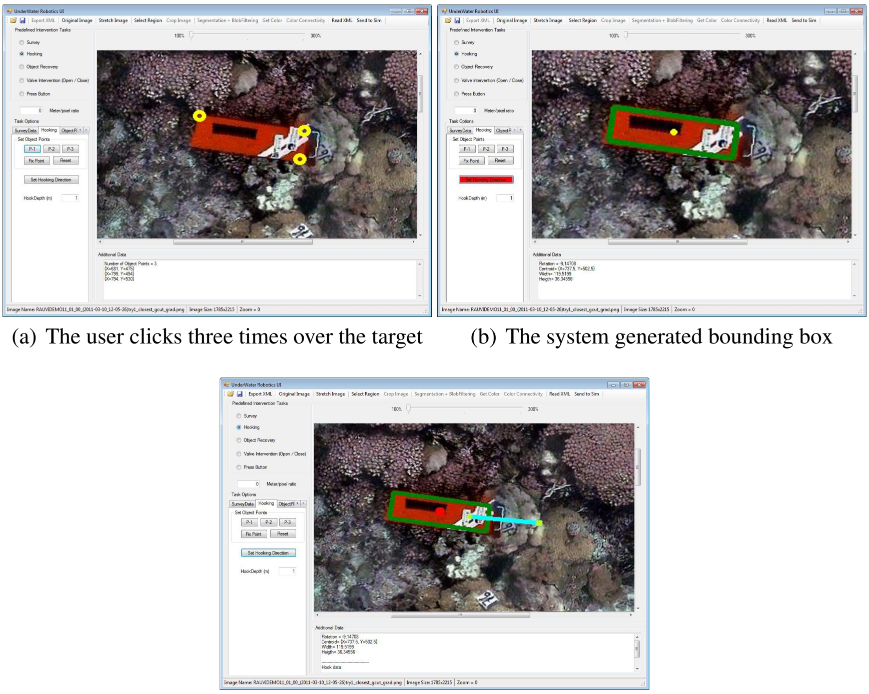
Fig. 4.15 The grasping points can be defined manually (left image) or automatically (right image).

4.3.7.2 Defining the parameters for a *Hooking intervention*

Additionally, and only available in a *Hooking intervention* type, the user can identify the object by clicking three different points 4.16(a). Thus, the application will use these three points in pairs to define the sides of a rectangle that defines the bounding box of the interest object and containing all the information needed. Then, the user needs to specify the direction of the hooking. To do this, the user draws a line (see Fig. 4.16(c)) in the direction that should be taken by the the robot, for this purpose.

4.4 MERBOTS user interface

Unlike RAUVI and TRIDENT research projects, which were oriented to the use of AUVs, the MERBOTS research project was focused on the use of a HROV. As mentioned be-



Panel (a):

```
Number of Object Points = 3
{X=681, Y=475}
{X=799, Y=494}
{X=794, Y=530}
```

Meter/pixel ratio: 0
HookDepth (m): 1

Image Name: RAUVIDEMO11_01_00_(2011-03-10_12-05-26)try1_closest_gcut_grad.png | Image Size: 1785x2215 | Zoom = 0

Panel (b):

```
Rotation = -9,14708
Centroid= {X=737,5, Y=502,5}
Width= 119,5199
Heigth= 36,34556
```

Panel (c):

```
Rotation = -9,14708
Centroid= {X=737,5, Y=502,5}
Width= 119,5199
Heigth= 36,34556

-----------------------------------
Hook data:
```

(a) The user clicks three times over the target

(b) The system generated bounding box

(c) The user draws the line to define the hooking position and direction

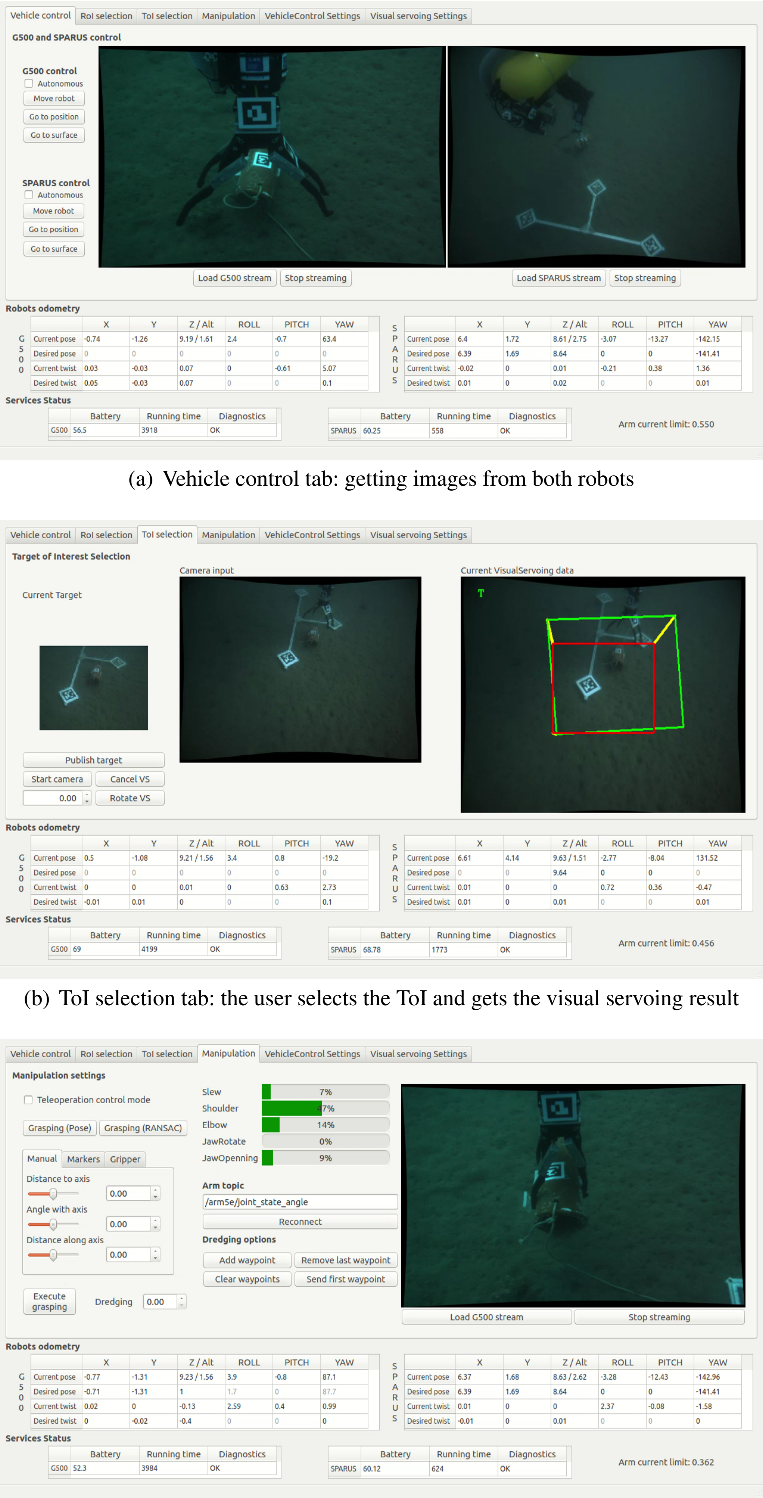
Fig. 4.16 Defining a hooking intervention.

fore, such a robot can operate either as an AUV or as a ROV, and also a data connection between the robot and the vessel could be available. This means that the user can follow the autonomous execution through the GUI or take control of the mission in case of need. Therefore, all the relevant information must be properly available to the user. In order to improve the pilot's perception of the mission, this project added a second robot with the purpose of providing an additional point of view of the ongoing intervention.

So, bearing this in mind, the MERBOTS user interface (see Fig. 4.17) was developed composed of two well defined sections:

- A mission section, which will be used for specifying all the parts of the mission.
- A service-status section, which will be used for displaying the most relevant data (e.g. odometry, battery status...).

Through this implementation, the user is able to supervise the status of both robots (the lower part of the window), and at the same time he/she may control both robots' cameras (see Fig. 4.17(a)), select the ToI and supervise its tracking (see Fig. 4.17(b)) or specify the manipulation options (4.17(c)). Finally, the last two tabs are used to configure some additional parameters.



Robots odometry

G500

	X	Y	Z / Alt	ROLL	PITCH	YAW
Current pose	-0.74	-1.26	9.19 / 1.61	2.4	-0.7	63.4
Desired pose	0	0	0	0	0	0
Current twist	0.03	-0.03	0.07	0	-0.61	5.07
Desired twist	0.05	-0.03	0.07	0	0	0.1

SPARUS

	X	Y	Z / Alt	ROLL	PITCH	YAW
Current pose	6.4	1.72	8.61 / 2.75	-3.07	-13.27	-142.15
Desired pose	6.39	1.69	8.64	0	0	-141.41
Current twist	-0.02	0	0.01	-0.21	0.38	1.36
Desired twist	0.01	0	0.02	0	0	0.01

Services Status

	Battery	Running time	Diagnostics
G500	56.5	3918	OK

	Battery	Running time	Diagnostics
SPARUS	60.25	558	OK

Arm current limit: 0.550

(a) Vehicle control tab: getting images from both robots

0.00

Robots odometry

G500

	X	Y	Z / Alt	ROLL	PITCH	YAW
Current pose	0.5	-1.08	9.21 / 1.56	3.4	0.8	-19.2
Desired pose	0	0	0	0	0	0
Current twist	0	0	0.01	0	0.63	2.73
Desired twist	-0.01	0.01	0	0	0	0.1

SPARUS

	X	Y	Z / Alt	ROLL	PITCH	YAW
Current pose	6.61	4.14	9.63 / 1.51	-2.77	-8.04	131.52
Desired pose	0	0	9.64	0	0	0
Current twist	0.01	0	0	0.72	0.36	-0.47
Desired twist	0.01	0	0.01	0	0	0.01

Services Status

	Battery	Running time	Diagnostics
G500	69	4199	OK

	Battery	Running time	Diagnostics
SPARUS	68.78	1773	OK

Arm current limit: 0.456

(b) ToI selection tab: the user selects the ToI and gets the visual servoing result

□ Teleoperation control mode

Distance to axis 0.00

Angle with axis 0.00

Distance along axis 0.00

Dredging 0.00

Slew	7%
Shoulder	47%
Elbow	14%
JawRotate	0%
JawOpenning	9%

Arm topic

/arm5e/joint_state_angle

Robots odometry

G500

	X	Y	Z / Alt	ROLL	PITCH	YAW
Current pose	-0.77	-1.31	9.23 / 1.56	3.9	-0.8	87.1
Desired pose	-0.71	-1.31	1	1.7	0	87.7
Current twist	0.02	0	-0.13	2.59	0.4	0.99
Desired twist	0	-0.02	-0.4	0	0	0

SPARUS

	X	Y	Z / Alt	ROLL	PITCH	YAW
Current pose	6.37	1.68	8.63 / 2.62	-3.28	-12.43	-142.96
Desired pose	6.39	1.69	8.64	0	0	-141.41
Current twist	0.01	0	0	2.37	-0.08	-1.58
Desired twist	-0.01	0	0.01	0	0	0

Services Status

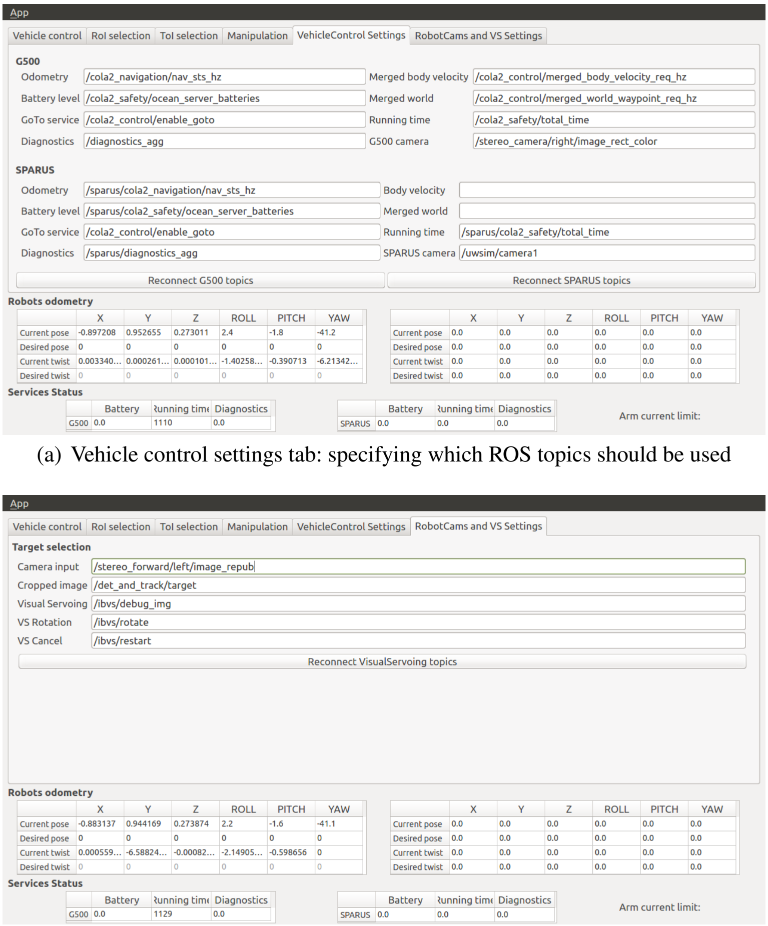
	Battery	Running time	Diagnostics
G500	52.3	3984	OK

	Battery	Running time	Diagnostics
SPARUS	60.12	624	OK

Arm current limit: 0.362

(c) Manipulation tab: controlling and specifying the grasping parameters

Fig. 4.17 MERBOTS GUI overview.



G500

Odometry /cola2_navigation/nav_sts_hz

Battery level /cola2_safety/ocean_server_batteries

GoTo service /cola2_control/enable_goto

Diagnostics /diagnostics_agg

Merged body velocity /cola2_control/merged_body_velocity_req_hz

Merged world /cola2_control/merged_world_waypoint_req_hz

Running time /cola2_safety/total_time

G500 camera /stereo_camera/right/image_rect_color

SPARUS

Odometry /sparus/cola2_navigation/nav_sts_hz

Battery level /sparus/cola2_safety/ocean_server_batteries

GoTo service /cola2_control/enable_goto

Diagnostics /sparus/diagnostics_agg

Body velocity

Merged world

Running time /sparus/cola2_safety/total_time

SPARUS camera /uwsim/camera1

Robots odometry

	X	Y	Z	ROLL	PITCH	YAW
Current pose	-0.897208	0.952655	0.273011	2.4	-1.8	-41.2
Desired pose	0	0	0	0	0	0
Current twist	0.003340…	0.000261…	0.000101…	-1.40258…	-0.390713	-6.21342…
Desired twist	0	0	0	0	0	0

	X	Y	Z	ROLL	PITCH	YAW
Current pose	0.0	0.0	0.0	0.0	0.0	0.0
Desired pose	0.0	0.0	0.0	0.0	0.0	0.0
Current twist	0.0	0.0	0.0	0.0	0.0	0.0
Desired twist	0.0	0.0	0.0	0.0	0.0	0.0

Services Status

	Battery	Running time	Diagnostics
G500	0.0	1110	0.0

	Battery	Running time	Diagnostics
SPARUS	0.0	0.0	0.0

Arm current limit:

(a) Vehicle control settings tab: specifying which ROS topics should be used

Target selection

Camera input /stereo_forward/left/image_repub

Cropped image /det_and_track/target

Visual Servoing /ibvs/debug_img

VS Rotation /ibvs/rotate

VS Cancel /ibvs/restart

Robots odometry

	X	Y	Z	ROLL	PITCH	YAW
Current pose	-0.883137	0.944169	0.273874	2.2	-1.6	-41.1
Desired pose	0	0	0	0	0	0
Current twist	0.000559…	-6.58824…	-0.00082…	-2.14905…	-0.598656	0
Desired twist	0	0	0	0	0	0

	X	Y	Z	ROLL	PITCH	YAW
Current pose	0.0	0.0	0.0	0.0	0.0	0.0
Desired pose	0.0	0.0	0.0	0.0	0.0	0.0
Current twist	0.0	0.0	0.0	0.0	0.0	0.0
Desired twist	0.0	0.0	0.0	0.0	0.0	0.0

Services Status

	Battery	Running time	Diagnostics
G500	0.0	1129	0.0

	Battery	Running time	Diagnostics
SPARUS	0.0	0.0	0.0

Arm current limit:

(b) Visual servoing settings tab: specifying which ROS topics should be used

Fig. 4.18 MERBOTS GUI overview (continuation).

4.4.1 Implementation details

After the development of UWUI, and some research projects related to underwater robotics, the *Interactive & Robotic Systems Lab* has decided to change the development framework to Ubuntu + ROS, in order to keep up with the state of the art in this respect.

So, this interface was developed using the aforementioned framework, in order to integrate into the GUI some functionalities and features already implemented in ROS. Thus, it represents the main advantage of easy integration with the whole project architecture and gain access to all the information provided by ROS. It should be mentioned here that all this development was focused on the control of the Girona500 HROV for the final project experiments, carried out in Sant Feliu de Guíxols (Girona, Spain) on 27-31/March, 2017.

As already mentioned, the interface uses multiple tabs where each of them provides a specific functionality:

- Vehicle control tab (see Fig. 4.17(a)): the user can visualize both cameras, one per robot, and take control of the robot. Thus, the user specifies a new position (e.g. moving the robot some meters in each direction, starting from the current position) or sends the robot to the surface. Additionally, there is a checkbox to send the signal to the robot indicating the user will take control of it. Nevertheless, this feature was not used because both robots implemented this feature as part of its architecture without the need of sending this signal.
- ToI selection tab (see Fig. 4.17(b)): It displays the camera output, while the user draws a rectangle on the ToI. Once it has been selected, the user clicks on *Publish target* button and the algorithm starts to track the target.
- Manipulation tab (see Fig. 4.17(c)): the user may control each arm joint with bars, specifying parameters for the dredging or grasping intervention, or send a signal to teleoperate the arm. Concerning the bars, they use a different background color depending on the joint value, moving from a green color (safe value) to a red color, which indicates the joint reaches its limit.
- Vehicle control settings tab (see Fig. 4.18(a)): this is a control tab, where the user specifies the topics needed for each robot, from odometry information to camera output.
- Visual servoing settings tab (see Fig. 4.18(b)): this is a control tab, where the user specifies the topics related to the visual servoing algorithms, due to fact that this processing was done by a dedicated ROS node.

Concerning the *robots odometry* and *services status* section, they were implemented by using the different ROS topics available in the architecture. Two watchdogs were also added: The first one is in charge of warning when the battery status is over a threshold, painting the cell in red when the value is under that threshold. The second watchdog, parses the *Diagnostics_agg* ROS topic, which includes different service status coming from the robot. Thus, the cell will be displayed in red, when the watchdog detects an “Error” or an “Alarm”.

This video in YouTube[7] shows the GUI running during an experiment of the MERBOTS project.

4.5 GUI evaluation

There are several goals related to the design of a UI, but some of the most relevant is making it easy to use and learning, at the same time it must be reliable against human or system errors. By definition, user-centered UI design is focused on users, whose work is to be supported by an application.

[7] https://www.youtube.com/watch?v=1xECxNb0-dQ

In the following subsections, a comparison between the aforementioned GUI against different commercial and research GUI is carried out.

4.5.1 Comparing RAUVI/TRIDENT GUI vs. MERBOTS GUI

Due to the different roles played by the user in RAUVI/TRIDENT projects (mainly as mission planner) and in MERBOTS project (both as controller & supervisor), the design and purpose of these two interfaces are different. For the first one, the information goes from the interface to the robot, whereas in the second one the information flow is bidirectional.

UWUI offers the user different mission types to define: survey, object recovery or hooking. Additionally, while a specific implemented algorithm assists the user to identify the ToI on the image, there are other ones serving to enhance the acquired images in case of failure of the first. Finally, all the mission data is exported to XML files adequate to be loaded onto the robot or the simulator.

On the other hand and assuming that there is a supporting network connection, the MERBOTS GUI display information about robot real time data, as well as the user can interact with the robot taking its control through the joystick or gamepad.

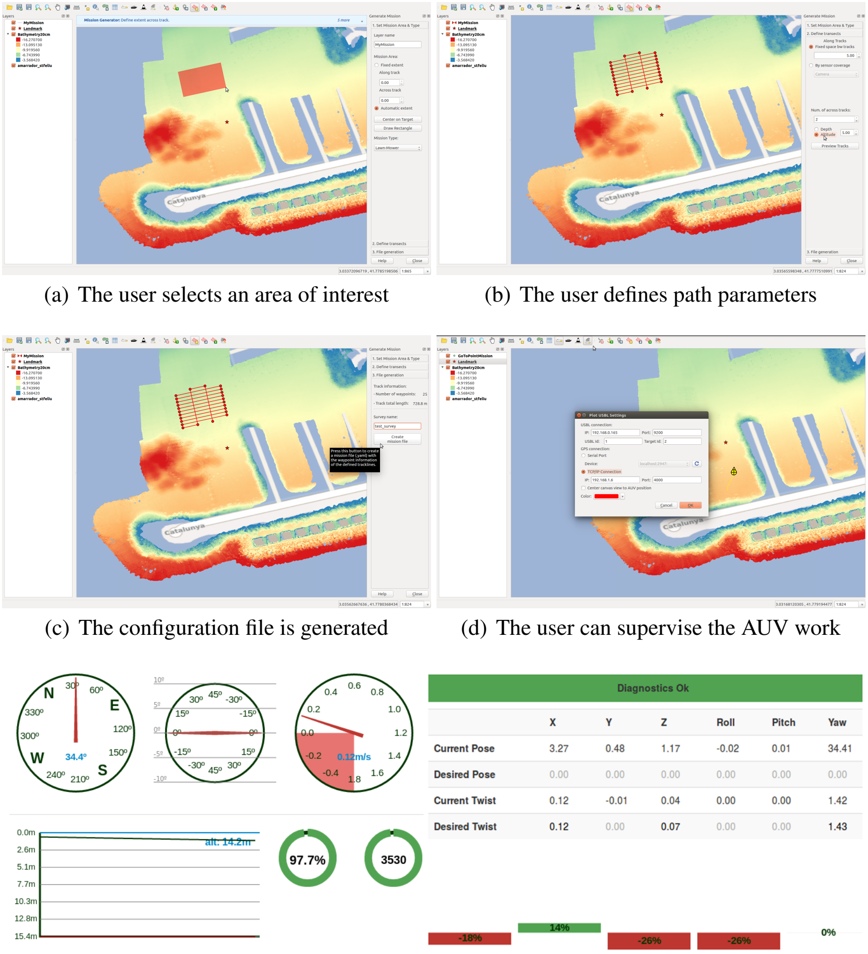
4.5.2 Comparing RAUVI/TRIDENT GUI vs. AUV GUI of research projects

As the current work was developed in the context of multiple ongoing projects coordinated by the IRSLab, where some of the involved partners have specific underwater vehicles, it makes sense to compare our proposals with the GUIs they use. In this case, we compare our work with the GUI used at both Spanish universities: UIB and UdG. The vehicles used in both universities are mainly used for visual-related tasks (e.g. photomosaicing, surveys) and both robots (Sparus-II[8] and Girona500) are mainly used as AUV.

As explained before, the MERBOTS GUI is focused only in the intervention task; no effort was devoted to the development of options to support the specification of the navigation plan. However, the UWUI can be used for the survey specification as we explained before, so we use that to compare it against the aforementioned GUIs at UdG (see Fig. 4.19) and UIB (see Fig. 4.20).

In the case of UdG's GUI, it is divided into two different applications. The first one runs on Linux and is used to specify the mission parameters (see Fig. 4.19(a) - 4.19(d))). The second one (see Fig. 4.19(e)) is accessible with any web browser and displays the most relevant information when the robot is connected to the main station. Concerning the first one, the main difference against our interface is the possibility to track the robot with the GPS information and plot in the map the USBL sensor data.

[8] http://cirs.udg.edu/auvs-technology/auvs/sparus-ii-auv/


(a) The user selects an area of interest

(b) The user defines path parameters

(c) The configuration file is generated

(d) The user can supervise the AUV work

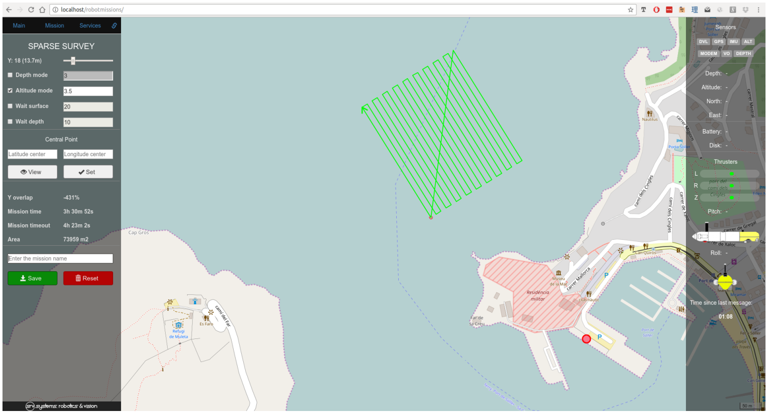
Diagnostics Ok

	X	Y	Z	Roll	Pitch	Yaw
Current Pose	3.27	0.48	1.17	-0.02	0.01	34.41
Desired Pose	0.00	0.00	0.00	0.00	0.00	0.00
Current Twist	0.12	-0.01	0.04	0.00	0.00	1.42
Desired Twist	0.12	0.00	0.07	0.00	0.00	1.43

(e) Web interface to supervise the main data

Fig. 4.19 Two different GUIs to use with the Girona500 AUV.

On the other hand, the GUI used at UIB has been developed using HTML, so the user can access it using almost any device, from a tablet to a computer. Additionally, in the case of data connection with the robot, this GUI displays some information about sensor data, position, battery status and thrusters information. Nevertheless, the overall process to define the survey is quite similar to our approach: the user clicks on the map to select the starting and ending point and specifies depth/altitude. Then, the mission configuration file can be loaded onto the robot.



Y: 18 (13.7m)

□ Depth mode 3

☑ Altitude mode 3.5

□ Wait surface 20

□ Wait depth 10

Latitude center Longitude center

Y overlap	-431%
Mission time	3h 30m 52s
Mission timeout	4h 23m 2s
Area	73959 m2

Enter the mission name

Depth: -

Altitude: -

North: -

East: -

Battery: -

Disk: -

Pitch: -

Roll: -

Time since last message:

01:08

Fig. 4.20 AUV GUI used at University of Balearic Islands

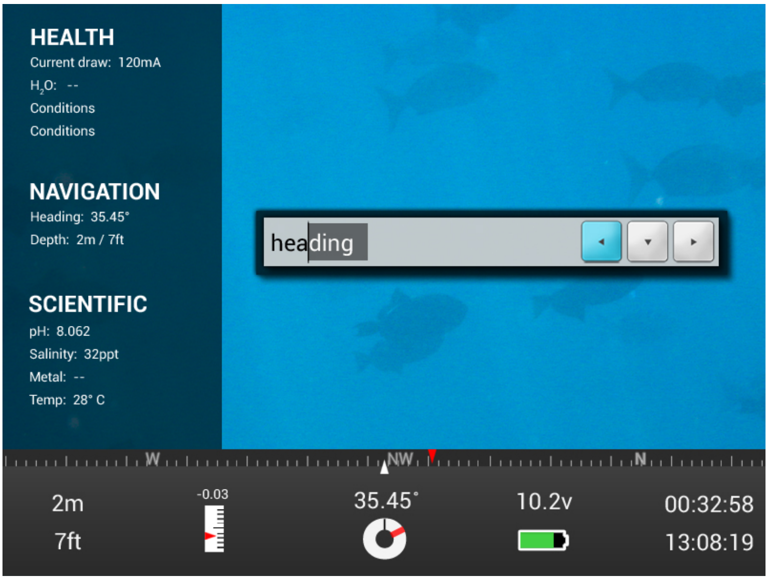
4.5.3 Comparing MERBOTS GUI vs. other open source GUIs

4.5.3.1 OpenROV

Probably, one of the most well-known open source projects related to underwater robots is OpenROV[9]. OpenROV started out as a personal project between David Lang and Eric Stackpole with the aim exploring an underwater cave. Since then, OpenROV has grown into a company and a community of *Do It Yourself (DIY)* volunteers, who are working together to create more accessible and affordable tools for underwater robot exploration. Fig. 4.21 shows the OpenROV GUI, which is based on web technologies and it is hosted inside the robot.

As we can observe in the screenshot of Fig. 4.21, the OpenROV GUI displays the most relevant information concerning the navigation data (bottom of the image), the sensors data (left side of the image) and also streams video from the onboard camera. When comparing with the MERBOTS GUI, this does not display the same sensors' data (mainly because the Girona500 has not been endowed with those sensors). On the other hand, our GUI not only displays the current position and velocity, but also the desired position and velocity as specified by the path planner. Another difference in our favour, is that the error messages sent by the ROV are made available to the user. Concerning the ROV control, the OpenROV is controlled using a joystick and the GUI has no option to send commands to the ROV.

[9] `https://www.openrov.com/`



HEALTH
Current draw: 120mA
H_2O: --
Conditions
Conditions

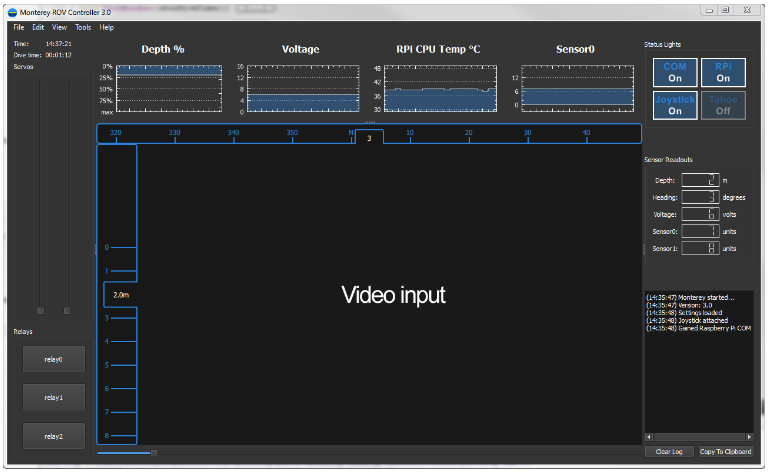
NAVIGATION
Heading: 35.45°
Depth: 2m / 7ft

SCIENTIFIC
pH: 8.062
Salinity: 32ppt
Metal: --
Temp: 28° C

Fig. 4.21 OpenROV GUI

4.5.3.2 Monterey GUI

Monterey[10] is an open source ROV control system developed by Chris Konstad. This multi-platform project has been developed using Qt and the Simple DirectMedia Layer (SDL) library to enable the joystick support. The Monterey architecture ran a small client on the ROV and the main control server on the topside computer.



Time: 14:37:21
Dive time: 00:01:12

Depth: 2 m
Heading: 3 degrees
Voltage: 6 volts
Sensor0: 7 units
Sensor1: 8 units

```
(14:35:47) Monterey started...
(14:35:47) Version: 3.0
(14:35:48) Settings loaded
(14:35:48) Joystick attached
(14:35:48) Gained Raspberry Pi COM
```

Fig. 4.22 Monterey GUI

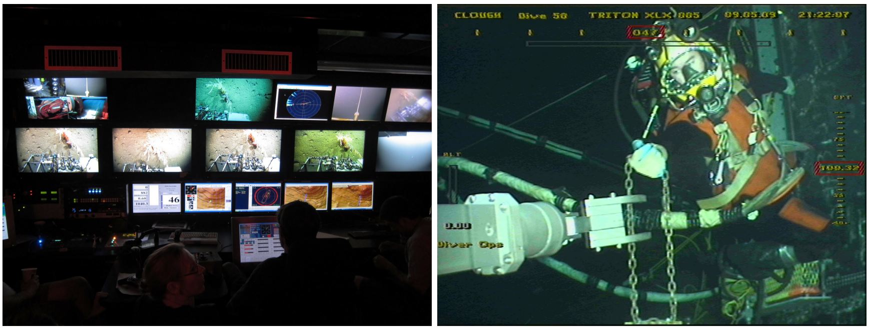
[10] https://github.com/rovsuite/monterey

As we can appreciate in Fig. 4.22, this GUI follows a similar approach to the OpenROV GUI, showing only the basic navigation data (depth and heading degrees), some unspecified sensor data, information about the battery voltage and CPU temperature, and there is a log with information coming from the ROV.

4.5.4 Comparing MERBOTS GUI vs. commercial GUIs

Comparing MERBOTS GUI with commercial solutions is quite difficult, because of the main purpose of each one. As we mentioned before, the typical ROV mission requires a lot of resources, from the vessel to the ROV control room. As we can see in Fig. 4.23(a), the control room for the ISIS ROV consists of several monitors to control the camera outputs, navigation data and sensor data. Additionally, some of these monitors are duplicated, so the ROV pilot pays attention to some of them, meanwhile the rest of the team (e.g. pilot assistant or scientists) pay attention to the duplicated monitors. If we look for a similar solution using the MERBOTS GUI, we should modify the code in order to split the different tabs into different screens, as well as moving one tab in a web browser window into a new window.

Fig. 4.23(b) shows the Triton XLX[11] ROV video output. Paying attention to it, the information received from the user is limited only to depth, vertical/horizontal gain and current time. We consider that the user lacks essential information, such as current and desired robot position, robot services status and diagnostics warnings and errors. Nevertheless, this GUI seems to be more focused on the video streaming, making the use of extra monitors not necessary.



(a) Missions with ROVs requieres a lot of resources.

(b) Triton XLX ROV video output

Fig. 4.23 Example of commercial GUIs for a ROV control.

[11] http://www.dof.no/en-GB/Assets/Subsea-Assets/Triton-XL37

4.6 Closure

This chapter offers an overview of the HRI issues, seen from the GUI point of view. Each one of these has been focused on the use of a specific underwater robot. Focusing on the use of a specific underwater robot, ROV or HROV, two different approaches have been proposed and developed. Although these GUIs were developed in the context of different research projects, they can be used with other projects with a similar approach.

The main motivation behind this research line was to improve the complex and not human-centered GUIs used for piloting an underwater robot. In both approaches, the user does not need to learn a specific language to manage the GUI or the robot and, in addition, long previous experience in the field is not required. We were also focused on the user experience when using the GUI and the reduction of the user cognitive workload and stress.

All these contributions have been published in: (46), (60), (61), (62), (63), (47), (64), (65), (66), (67), (68), (69), (70), (71), (72).

Chapter 5
Towards a multimodal and 3D immersive user interface

In this chapter, we present a solution for the teleoperation problem based on exploring an immersive system. Such a system is used to induce a telepresence feeling so that the operator acts as if he/she was aboard the robot, reducing the mental workload induced by third person views.

The recent introduction on the market of devices like Kinect™ and Leap Motion™, which are able to track and estimate the pose of the human body and hands, seems to create an excellent opportunity to replace the traditional joysticks, keyboards and mice. This motivated the study of their benefits by measuring some parameters related to task performance achieved by a group of users and analyzing their subjective evaluation in terms of usability, perceived task load and immersive feeling.

5.1 Introduction

The traditional setups composed of multiple screens, displays and controls, used in control rooms dedicated for remotely operated vehicles and robots, require one or more specialized trained operators. The need for multiple operators is justified by the multiple systems to control and the need to pay simultaneously attention to multiple variables or sources of information.

In this chapter, our proposal aims at simplifying the remote operation control setup, by exploring the principle of telepresence. Our assumption is that if, by the use of some devices, the operator can experiment the sensation of being inside the robot, disposing of a wide field of view, and then the control task becomes as natural as driving a car. This can be achieved by transforming some of the existent explicit controls into implicit ones, e.g. by controlling the orientation of a camera using head rotation instead of using a joystick or other control for that, reducing both the required dexterity and implied cognitive workload.

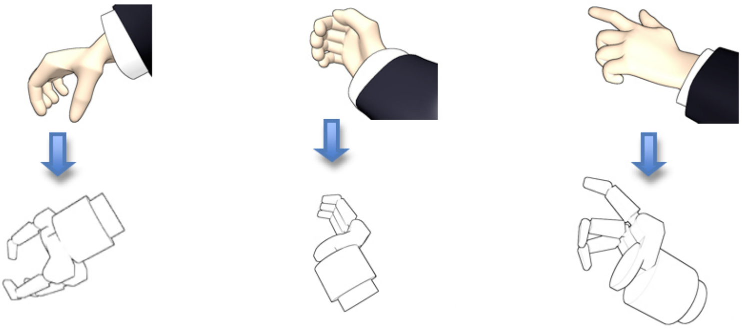


Fig. 5.1 Three different ideal preshapes and their mapping to a Barrett Hand.

5.1.1 Exploring 3D techniques and multimodal capabilities

According to (73), the concept of ideal hand task-oriented hand preshapes are a set of hand preshapes defined for an ideal hand and extended with task-oriented features. The ideal hand is an imaginary hand able to perform all the human hand movements. So, the approach is to plan or define grasps by means of ideal preshapes, and then define hand adaptors as a method for the instantiation of the ideal preshapes on real robotic hands. The main advantage of this approach is that the same grasp specification can be used for different hands, just by defining a suitable mapping between the ideal hand and the real one. This concept is illustrated in Fig. 5.1, which shows three different ideal preshapes and their mapping to a robotic Barrett Hand.

Bearing in mind this concept, we planned to adopt this approach for the grasp specification and execution in the context of our grasp simulator. Thus, the human operator specifies a grasp using its own hand covered with a data glove. The finger joint angles captured by the data glove tracking system will be passed to a standard classifier (e.g. like in (74)), which selects the hand preshape that best suits the human hand posture.

In order to test this concept, we made some tests using a *Essential Reality P5 Data Glove*, where the user's hand is covered with a data glove and the tracking system replicates the human hand motion in the simulated environment (see Fig. 5.2). These tests consist in grasping different objects placed in a virtual environment. Unfortunately, this device was not precise enough, getting different problems, the most relevant being that the tracker loses the hand or the low accuracy to detect finger movements. However, these tests opened a new research line to explore.

Additionally, during the early stages of the UWUI development (see section 4.3), we also explore the immersion and the concept of 3D. In fact, we explore it with two different approaches:

- Concerning the immersive capabilities, the Head-Mounted Display "5DT HMD 800-26 3D" was connected with the simulator developed by Gianluca De Novi (46). Due to this simulator was proprietary and the Interactive and Robotic Systems Lab took the decision to develop a new one, this approach was discarded. However, although the screens of this HMD have a low resolution and it was quite difficult to adapt the

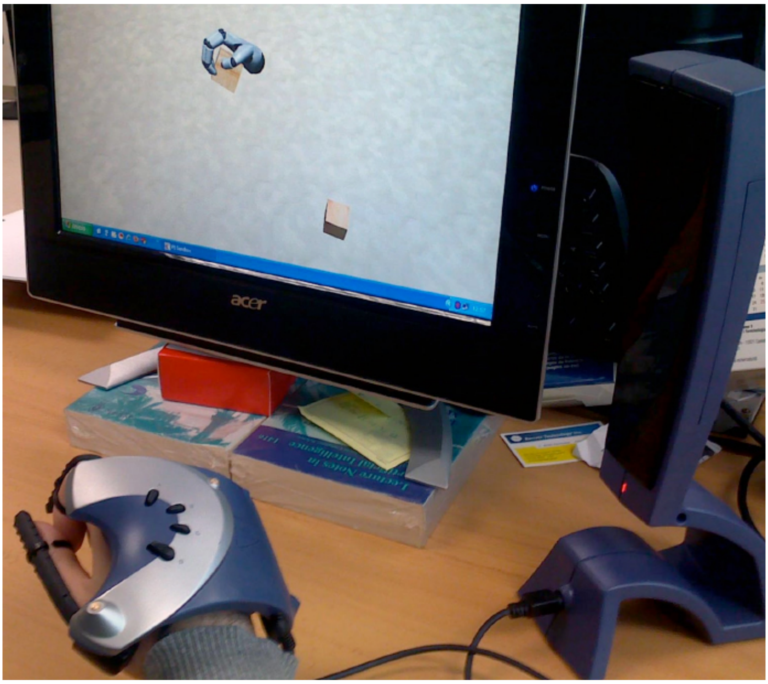
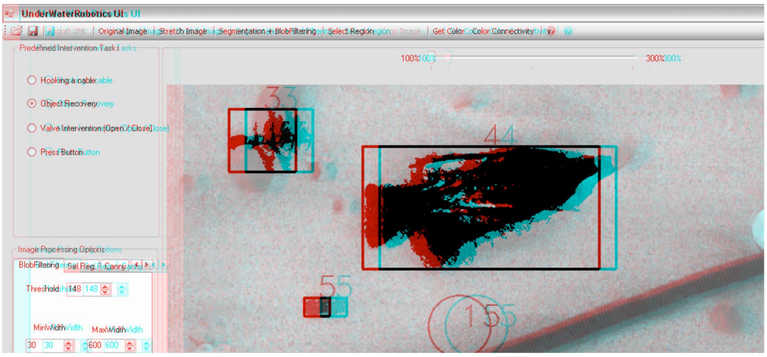


Fig. 5.2 Detail of the P5 data glove during a simple test: ‘grasp a virtual cube”.

simulator to such resolution, this test demonstrated the possibilities of an immersive system. Apart from connecting the HMD, we used the data of a 3D motion tracking “XSENS MTi” placed on the top of HMD, to move the camera POV according to the user head movements. This test was satisfactory, because the user does not need to use the mouse to move the camera POV and this movement became natural and easy-to-adapt.

- Regarding the 3D experience when using the UWUI, we explored the anaglyph effect. This approach was discarded quickly, because the benefits of this technique were not enough to improve the usability and the user experience. One of the main problems concerns the poor text legibility.

So, bearing in mind all these tests, we decide to search for a new framework to develop these features. Due to the fact that some of these features require a graphics and physics engine, and the IRSLab develops an underwater simulator, we decide to use it as a base for this proposal.



Threshold 148

MinWidth 30

MaxWidth 600

Fig. 5.3 Testing the UWUI with the anaglyph effect.

5.2 Designing an immersive teleoperation system

5.2.1 Getting an immersion feeling

The teleoperation of a robot in a remote underwater environment cannot rely on the use of external cameras to produce external views of the robot movements or operation. In most cases the camera has to be placed on the robot itself. This permits the operator to have a ”first person-like” view of a remote environment. The image from this one or more cameras is displayed in one screen with the possibility to commute from one camera to the other, simultaneously with other information related to the mission. In the case of having a single camera on a Pan/Tilt Unit (PTU), this one must be oriented using some additional control device that has to be operated in parallel with piloting the robot.

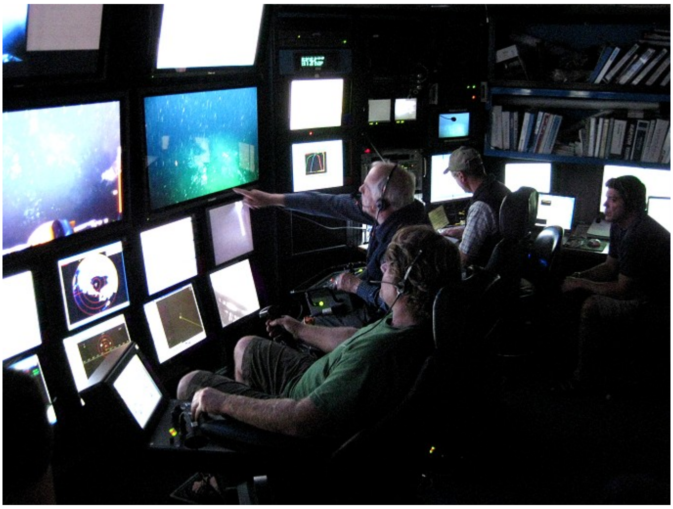


Fig. 5.4 A typical ROV Control Room (courtesy of Monterey Bay Aquarium Research Institute).

To achieve the aforementioned goal of creating a simpler and more natural user interface for teleoperating robots, in particular Underwater Vehicle (UV)s, we have designed a system that takes the user aboard the remote vehicle inside a VC. This should overcome the limitations, of having a single camera view whose orientation is manually controlled, that normally result in higher demands in terms of concentration, attention, etc. This can be achieved by using a HMD, whose orientation is used to control the PTU.

By proportionating a first person wide field of vision, it should induce a sense of presence on the operator, enabling him/her to pilot the UV as if the operator were on board. When immersed, the user gets full attention to perform a specific task. For this, it is important to keep the synchronism between the user head movements and the virtual/real camera movements. Thus, it creates the feeling of being at the remote location providing a better perception of movement and limits of the robot.

To achieve the sense of "being there" several aspects should be taken into account.

- The chosen position on the robot to fixate the PTU and its camera defines the location of the virtual cockpit. This location has to be carefully chosen as it has to be adequate for proportionating the best view for the task to be performed, e.g. navigation and maneuvering the AUV, or controlling a robotic arm.
- The user head movements should be integrated in the system, moving the camera POV. This enables the user to have an egocentric perception of the environment and allows him to naturally explore and navigate, as if he was there. As a result, the user can look around or follow a moving object at the remote location.
- By superimposing virtual elements over the camera view, it is possible to create the perception of a cockpit with its instruments.

Thus, this fulfills our goal of providing the user with a perception similar to that of driving a car, piloting an helicopter, controlling an excavator, etc.

5.2.2 Virtual cockpit: from explicit to implicit controls

As mentioned before, one of the main drawbacks in the already GUI addressed to ROV/AUV systems, is the GUI complexity. As we can see in Fig. 5.5, which represents a robot mission trial related to the TRITON project, the user has to pay attention to different screens, due to UWSim uses a linux command line to show all the information to the user. In this setup (left-right, top-down), screen 1 is used for monitoring the arm, screen 2 is used for the visual tracking system, screen 3 is used for monitoring the vehicle status and screen 4 is used for monitoring the valve opening/closing intervention algorithm.

Thus, when using our proposal of the VC and the use of a HMD, user feedback can be improved and enhanced. This VC has animated elements placed on a virtual sphere that surrounds the user's location, simulating the feeling of being in a real cockpit. The combination of the head tracking and the Oculus Rift™ DK-1 (75), creates a sense of immersion and telepresence that is desirable when conducting a mission with a teleoperated vehicle. So, the virtual windshield is a practical way of presenting the required information

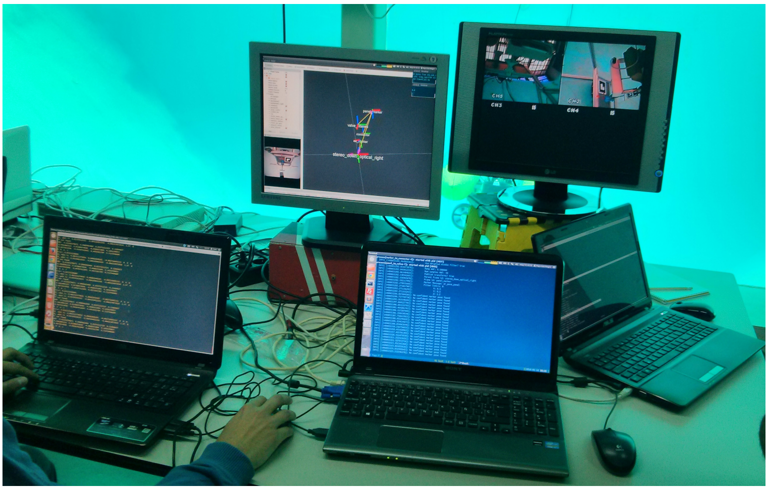


Fig. 5.5 During a trial related to TRITON project, the user should pay attention to different screens.

to the user and it is made to be intuitive, simple and informative without being intrusive, confusing and overwhelming.

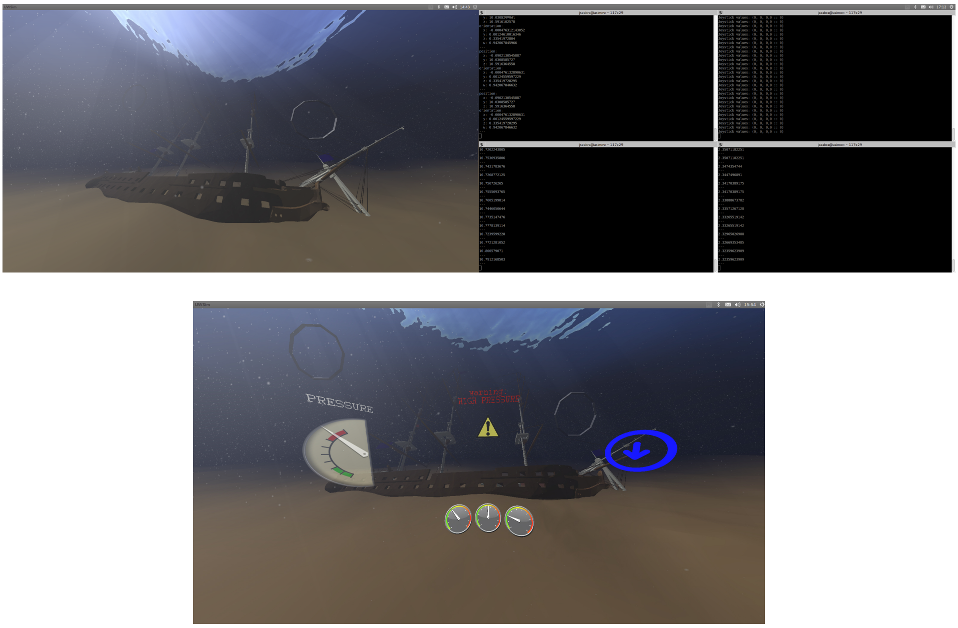
Fig. 5.6 shows the evolution, from the standard setup using UWSim (one screen for the simulator plus one screen for the output data such as navigation information or sensor data) to our proposal, where most of the information is shown graphically and it is filtered depending on its relevance (e.g. a blinking warning signal appears when the robot is close to the seafloor or the surface, a battery indicator appears when the battery charge is down a predefined value).

The possibility of a dynamic interface is not the only advantage of our approach. The feeling of being inside a real vehicle with a dashboard, warning lights and indicators also creates a familiar and immersive experience. With the aforementioned developed head tracking system, the user can even lean on any of the interface components as he/she would do in a real situation. All these little details that sometimes go unnoticed by the user create a sense of presence that is impossible to match just by looking at a screen (or several screens).

5.2.3 A multimodal interface

As it is our intention to understand how to develop a teleoperating system suitable to be used by non-specialised personnel, we have introduced the possibility of controlling the robot using a joystick or a Leap Motion™.

Although the most common input device to control the robot direction and speed is a joystick, we have tested and compared a joystick and a Leap Motion™. We decided to use the Leap Motion™ because we were exploring different approaches:



```
jseabra@asimov: ~ 117x29
  y: 10.0309208041
  z: 10.5916182578
orientation:
  x: -0.000476312143052
  y: 0.00124610816346
  z: 0.33541972804
  w: 0.942067845966
---
position:
  x: -0.0902138545887
  y: 10.0308585727
  z: 10.5916364558
orientation:
  x: -0.000476132890631
  y: 0.00124559597229
  z: 0.335419728295
  w: 0.942067846632
---
position:
  x: -0.0902138545887
  y: 10.0308585727
  z: 10.5916364558
orientation:
  x: -0.000476132890631
  y: 0.00124559597229
  z: 0.335419728295
  w: 0.942067846632
---
```

```
jseabra@asimov: ~ 117x29
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
Joystick values: (0, 0, 0,0 :: 0)
```

```
jseabra@asimov: ~ 117x29
10.7202243805
---
10.7536935806
---
10.7431783676
---
10.7268772125
---
10.756726265
---
10.7555093765
---
10.7605199814
---
10.7446050644
---
10.7735147476
---
10.7778139114
---
10.7239599228
---
10.7721281052
---
10.800579071
---
10.7912168503
---
```

```
jseabra@asimov: ~ 117x29
2.35071182251
---
2.35071182251
---
2.3474354744
---
2.3447496891
---
2.34178389175
---
2.34178389175
---
2.33888673782
---
2.33571267128
---
2.33265519142
---
2.33265519142
---
2.32965826988
---
2.32669353485
---
2.32359623909
---
2.32359623909
---
```

Fig. 5.6 Above: the user needs two screens to get all the information: (above-left) the simulated view and the info, (above-right) navigation data, commands sent the robot, pressure sensor (the user will know the robot depth) and the range sensor (the user will know the distance to the seafloor). Below: the UWSim using our proposal, displaying the same information, but in a graphical way.

- This device is considered close to the Natural User Interface (NUI), which is supposed to be easy to use and easy to understand by a non expert user.
- This device allows a more accurate hand and finger detection and tracker, which will make it easier to map some common user gestures used in smartphones and tablets, with specific robot control functions.

Nevertheless, there are some limitations to solve concerning the user experience. Once the user enables the HMD, the relative position of the hand with respect to the Leap Motion™ is lost, because the user can not see *the real world* and put the hand on the Leap Motion™ workspace. Another aspect is the need for some reference frame for the operator, so that he can perceive in any instant if he is looking in the forward direction of the robot, up, down or elsewhere. For this reason an Augmented Reality approach was taken by adding two virtual elements on a fixed position with respect to the user: a virtual table and a virtual joystick on it. The table acts as the reference object that enables the user to know to where he is looking at. The virtual joystick shows the control that is being applied through the device in use (the real joystick or the Leap Motion™).

To improve the perception of the Leap Motion™ location, a small fan was placed close to this device. With this, the user can sense the air flow and not only perceive its position, but also the vertical distance between the hand and the device by the airflow intensity. A second approach was to enable the user to see himself in the virtual cockpit and perceive the relative position between his hands and the Leap Motion™ device. This was done by using a Microsoft Kinect™, located behind the monitor in an upper position and pointing at the table, to capture a 3D point cloud that represents the user body and introduce it on the virtual environment.

In summary, the proposed teleoperation setup aims at addressing the problem of simplifying the teleoperation of an underwater robot, by taking the user virtually aboard. This setup has the possibility of integrating different interaction modalities and devices always aiming at reducing the number and complexity of controls required for the operation. This justifies for instance the inclusion of the Leap Motion™ as it can detect a large spectrum of hand poses and configurations that can be expected to be mapped to robot controls. As will be shown later, this type of device is not as precise as we could expect and introduces other types of problems.

Considering the combination of Leap Motion™ and the HMD controls, these are summarized in Table 5.1.

User movement	Command	Used Device
Move hand up/down	Robot/Arm moves up/down	*LM*-device
Move hand front/back	Robot/Arm moves front/back	*LM*-device
Move hand right/left	Robot/Arm moves right/left	*LM*-device
Rotate head	Rotate camera	HMD

Table 5.1 Relationship between user's movements and commands to send to the system.

5.3 Implementation details

5.3.1 UWSim, an open source underwater simulator

Simulators have become a very important tool not only for researchers to test new algorithms, but also for professional pilots to train before facing the real mission. The main drawback is that these simulators are closed, and thus, difficult to integrate with custom control architectures. Some of the most well-known ROV simulators are VortexSim (76), ROVsim (77), VROV (78) or DeepWorks (79). Concerning the autonomous vehicles, most of the simulators were developed in research projects (80). They were platform dependent, so access to the source code and collaborative development possibilities are normally very limited.

Thus, the IRSLab started to develop a new open source simulator in 2012: UWSim (81). This new simulator started with the RAUVI and TRIDENT research projects and is currently used in different ongoing projects funded by European Commission (MORPH, PANDORA) to reproduce real missions from the captured logs, for user training, to test algorithms, to monitor the robot or as a 3D simulation tool for benchmarking.

This simulator is multi-platform, has been implemented in C++ and makes use of the OSG (82) and osgOcean (83) libraries. OSG is an open source 3D graphics programming interface used by software developers in fields such as visual simulation, computer games, virtual reality, scientific visualization and modeling. osgOcean is another open source project that implements realistic underwater rendering using OSG and was developed as part of an EU funded research initiative called the VENUS project (84). UWSim uses the above mentioned libraries and adds further functionality for easily adding underwater robots to the scene, simulate sensors and interface with external control programs through the ROS (2).

Fig. 5.7 shows the UWSim architecture. As we can see, the core loads the main scene and robots, and it is connected with the *physics engine*, which manages the contacts between objects; the *Dynamics*, module that simulates hydrodynamics for underwater vehicles; the *osgOcean*, which renders all the surface and underwater effects relative to the ocean; the GUI, which provides support for visualization and the user interaction; and the *interface module*, that offers a ROS interface to communicate with external control software.

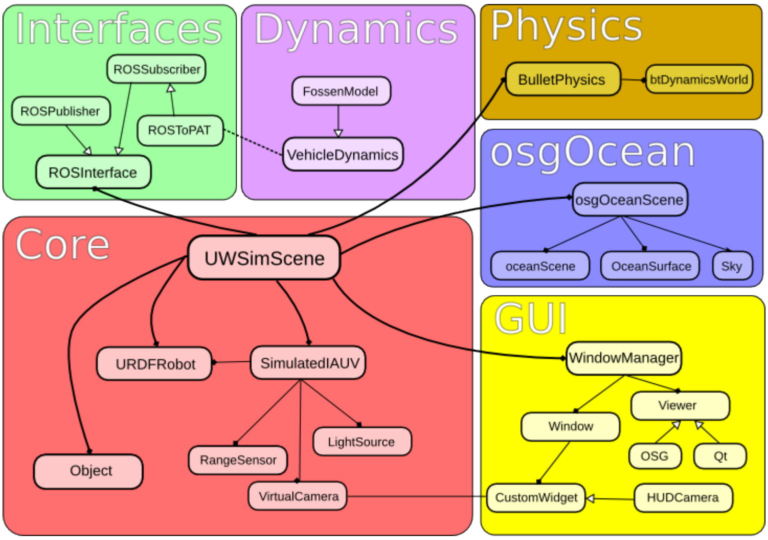


Fig. 5.7 This diagram represents the UWSim architecture.

The main characteristics are:

- Configurable environment: the scene can be configured with any modeling software and described in a XML file. The scene includes underwater visualization effects, such as ocean surface, underwater visibility, water color, silt particles, god rays, etc.
- Multiple robots support: it is provided a default robot (vehicle with arm), but other models can be defined according to the Universal Robotic Description Format (URDF) format. Different robots can be loaded and managed simultaneously in the scene.

- Simulated sensors: twelve sensors are available for vehicles plus default position/velocity sensors available for vehicles and manipulator joints.
- Contact physics: the physics engine Bullet has been integrated into the simulator, so it allows contacts and forces simulations.
- Network interfaces: ROS provides many facilities for communications and distributed computation. Through the network interfaces, it is possible to access/update any vehicle position or velocity, to move arm joints, or to access simulated sensors such as images generated by virtual cameras. However, the core software is independent of any middleware, so it is possible to interface the simulator with other middlewares.
- Widgets: there is support for customizable widgets, which are small windows that can be placed inside the main display in order to show specific data to the user.

5.3.2 Architecture description

The proposed architecture was designed by considering the following: a robotic device has a set of sensors that produce information for different purposes like controlling the trajectory, avoiding obstacles, and other task-related operations. On the other side, it has a set of actuators that are controlled to perform task-specific actions. Depending on which parts of the sensor produced the information and how they used it for generating the commands for the actuators, the robot takes part in certain kinds of task.

Unlike airplanes, cars, and other vehicles, there is still no standard interaction devices for UVs. By consequence researchers need to test and evaluate various combinations of input and output devices. Given that each device has its own characteristics, the replacing of these devices would be a very tough task, as not only each requires specific interfacing, but also the mapping between its controls and the device functions have to be adapted one by one.

To simplify this task we propose a new architecture, which is represented on figure 5.8, and has the characteristic of being highly reconfigurable and adaptable to different types of devices and tasks. This is made possible by the inclusion of the User Interface Abstraction Layer (UIAL), that has the role of enabling several interaction devices to be used for the same purpose. It shares some ideas with Open Tracker (85) in terms of reconfigurability and with Virtual Reality Peripheral Network (VRPN) (86) in terms of device transparency. In fact both can be used to provide a normalized interface for connecting many of the supported input devices. UIAL layer is then responsible for appropriate mappings between the devices and the UWSim and back.

The UIAL provides the following functionalities:

- Receives the information from the robot sensors and robot internal state.
- Transforms the data into the best representation for each visualization device.
- Maps the outputs of the controlling devices in the appropriate commands for the robot actuators.

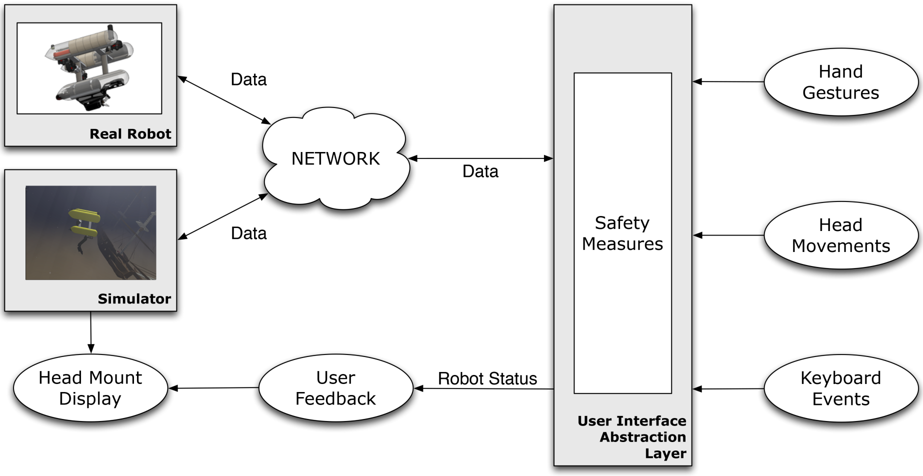


Fig. 5.8 Software architecture showing the role of the UIAL.

- Adapts the previous operations depending on the specificities of each task.
- Requests the simulator for generating some visualizations needed by some output devices.

Thus, the UIAL may reconfigure the use of both the input and output devices according to the mission or the task. It is also responsible for implementing some safety measures to prevent undesirable accidents from user errors. As an example, if the sensors say that the robot is close to the seafloor, any command to take the robot deeper will be ignored.

5.3.3 Low level details

As stated before, the development presented in this chapter uses as a base the simulator UWSim, which as in many current robotic applications, integrates the well established ROS framework. We should also keep in mind, that the current thesis relates to some work packages of the research projects coordinated at IRSLab, so we use the its architecture as a starting point to test our proposal. Nevertheless, the proposal presented here can be exported to other simulators and use other frameworks, e.g. YAML or GeNoM.

There are several benefits of using an architecture based on ROS, the “topics” (asynchronous streaming of data) being one of them. In our specific case, the real robot and UWSim have exactly the same communication interfaces (same topics). This means that the real robot can be controlled by ”publishing” the commands in a named topic, which can also be used for the simulator to mimic the robot behaviour and enable the operator to visualize the expected behaviour of the robot. This, in fact, enables the interface to display any information made available by the sensors or camera streaming, by simply ”subscribing” the corresponding topic and processing the received data.

As one of the purposes of our research is to provide an insight into which are the best interaction styles and modes for use in the teleoperation of remote robots, and as new devices for human interaction appear on the market every day, we realized that each of them has to be adapted for each particular use. As an example, two joysticks with different shapes, or a joystick and a yoke-like input, may require a different mapping between the device ”axes” and the intended commands. This mapping is not only in terms of establishing pairings, but also on the definition of calibration functions that may vary, between two joysticks, due to shape differences. For instance, using a Leap Motion™ the hand movements are tracked and mapped to control the robot using natural gestures: e.g. when the user moves the hand forward/backward/left/right, the robot will move forward/backward/turn left/turn right, and moving the hand up/down will make the robot to go up/down, respectively, etc.

Although VRPN libraries enable us to extend the range of devices that can be used, it is the UIAL that is responsible for providing the correct mappings. In addition it introduces the flexibility in performing online activation or deactivation of interaction devices. As an example, it enables the rapid change of the remote PTU control from a joystick to an HMD’s IMU, or the control of the thrust and direction of the robot from a joystick to a Leap Motion™, or other.

Due to the lack of testing with the real robot, we only use the UWSim, which publishes odometry data and simulated sensors data, like the sensor of range and pressure. The simulated robot can be controlled (navigation movements) by sending data to the odometry topic. So, the UIAL is in charge of getting the data published by the Leap Motion™ controller, which represents the user hand movement. This data is processed and sent to the subscribed odometry topic. In a similar manner, in order to control the camera POV, the UIAL modify this parameter using the Oculus Rift™ IMU or the mouse motion, depending on the current configuration.

5.3.4 Head tracking system & Oculus Rift™ for full immersion

An acceptable immersion level can be achieved through the action-response correlation. Using the right correlations between the user’s actions and the responses within the environment, this immersion can be maintained over time (87). In this state of immersion, it is expected that the user has a better perception of the virtual/remote environment, and consequently, will be more focused when performing any specific task.

To control the onboard camera’s POV, we integrated an head tracking system into UWSim. The developed system combines the information from the IMU of Oculus Rift™ DK-1 and the information obtained from a camera/marker system (Fig. 5.9). The IMU, composed by accelerometer and gyro provides the orientation, while the camera/marker system provides the position. Being the communications supported by cables, with most of the data flowing from the vehicle to the control station, we can expect that no important delay be introduced in the commands sent to the PTU. Concerning the PTU response, commercial PTUs can have very high performance, exhibiting speeds higher than 100 degrees per second. This is, in fact, below the maximum rotational speed that the human head can attain, which can be as high as 365 ± 96 degrees/s (88). Nevertheless, these higher rotational speeds are normally attained in response to frightening events, and not in normal conditions of operating or driving a vehicle like a car. In these cases, neck rotations at speeds of tens of degrees/s are used for browsing the field of vision or visually tracking moving targets. This should enable the user to behave as if he is aboard the remote robot, and attain a better level of control.

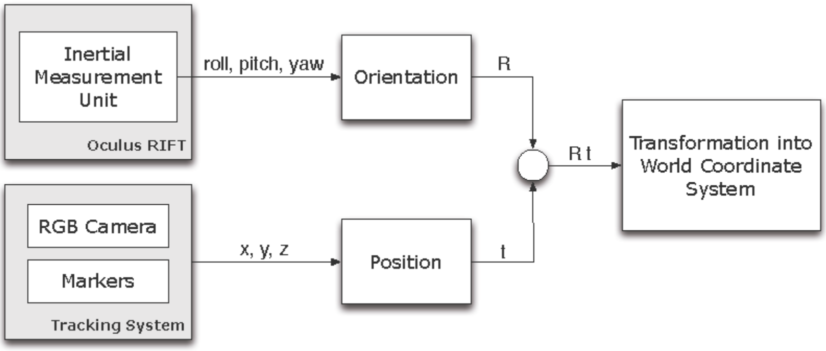


Fig. 5.9 This diagram shows how the information from the HMD and the head tracking system is merged, to transform it into world coordinate system.

The integration of IMU and camera/marker system enables a full mapping of the user's head movements to the virtual world (UWSim), providing a better experience and immersion level. This system is composed by aligning a simple marker to the IMU of a Oculus Rift™ DK-1 HMD, and a camera pointing to the user (Fig. 5.10). From the combination of camera and marker we extract the 3D position (89) and because the marker is physically aligned to the IMU we can apply its orientation on that point. At this stage, we have the transformation (translation and rotation) of the user's head ready to position the virtual camera on UWSim, and consequently, change the camera's POV and translation inside the virtual cockpit.

5.3.5 Robot Safety Measures

Another problem concerning the teleoperation of a robot, is to take care of the robot. There are several risky situations associated with human factors, such as getting the robot stuck, sending the robot deeper than the maximum depth or hitting the robot against the seafloor, to mention only a few. Maybe some of these problems are difficult to solve, but the effects of some others can be minimized or totally removed. Thus, the UIAL is also responsible for the verification of some safety measures. These safety measures are guided by a set of rules which are related to the information provided by some of the onboard sensors, like pressure sensors, proximity sensors, etc.

Some of these rules are:

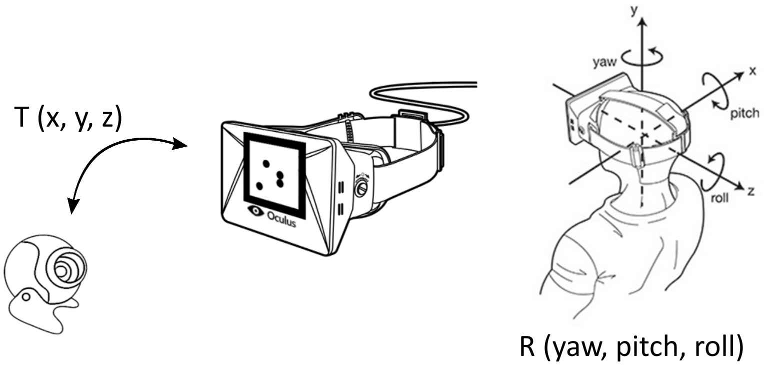


Fig. 5.10 The camera pointing the Oculus Rift™ DK-1 HMD with a marker attached.

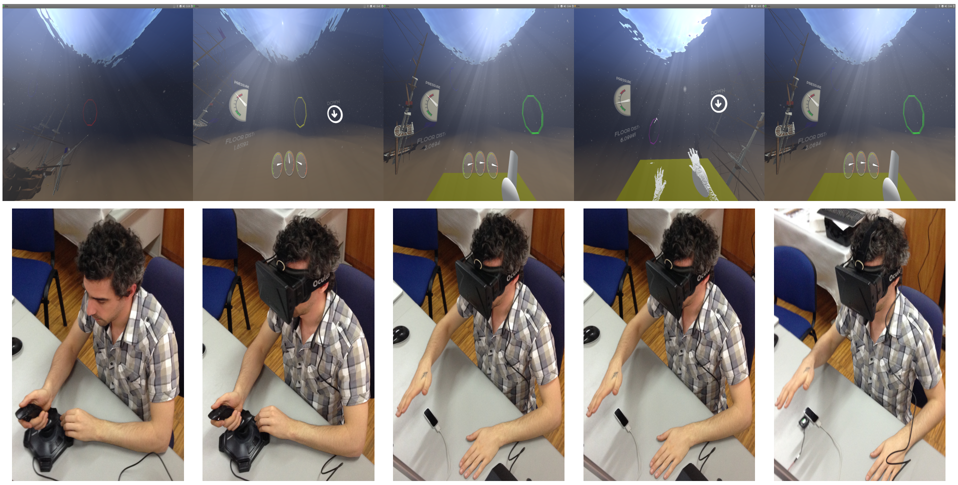
- Sending the robot under maximum depth: the UIAL reads the data from the pressure sensor, so once the maximum depth has been reached, all the commands sending the robot deeper will be discarded.
- Sending the robot too close to the seafloor: due to the fact that robot is endowed with an arm, the robot must keep enough space to allow the arm to work. This free space can be detected using a proximity sensor. Once the minimum distance to the seafloor has been reached, all the commands sending the robot closer to the seafloor will be discarded.
- Sending the robot to the surface: it is possible to know when the robot reaches the surface, by simply reading the pressure sensor data. Once the position has been reached, all the commands sending the robot to the surface will be discarded.

5.4 Evaluating user experience

Although the presented interaction mechanism was developed bearing in mind the control of real robotic platforms, for the sake of safety and given that the interest is in evaluating the interface, all the tests described hereafter were performed using solely the simulator UWSim.

5.4.1 Methodology

To evaluate the benefits of the proposed changes in the interaction mechanisms for teleoperating remote robots, we simulated a teleoperated underwater vehicle performing a simple obstacle navigation task. We compared our proposed immersive teleoperation approach based on a VC with a natural egocentric view, against the traditional teleoperation



Setup 1 Setup 2 Setup 3 Setup 4 Setup 5

Fig. 5.11 Experimental setups: (1) Traditional control; (2) Virtual Cockpit with joystick and virtual joystick; (3) Virtual Cockpit with Leap Motion™ and virtual joystick; (4) Virtual Cockpit with Leap Motion™ and point cloud for arms representation; (5) Virtual Cockpit with Leap Motion™ and airflow haptics.

interfaces that use manual camera control and visualization of mission related information through a set of monitors. Two control devices, joystick and Leap Motion™, were tested in terms of usability. Fig. 5.11 shows a user in different phases of of the test and an example of the scene that is visualized.

The evaluation consisted in analyzing a set of performance related parameters, which were collected during the experiments, and the answers given to a short questionnaire after each trial. The collected parameters, the questionnaire and their analysis are presented in the remaining part of this section.

5.4.2 Evaluation Procedure

For the purpose of evaluating the effect of immersive technologies on the tele-operation of underwater robots, we have designed an evaluation procedure where participants are invited to control a simulated underwater robot with the objective of completing a trajectory. That trajectory includes passing in order through 5 rings that are not collinear and have different orientations, in minimum time and without colliding with the rings or other underwater structures. For each experiment there is a ”warm up” from the starting position until reaching the first ring. The measuring process is started immediately upon passing the first ring. The process is repeated for each of the 5 control setups listed on table 5.2. For each participant, the sequence of the setups is random to avoid the effects of learning the trajectory that normally improves the performance for the latest executions.

In fact, the setups vary in terms of the type of support for visualization of the remote environment, the control of the remote camera orientation, and the robot navigation controls.

Test	**Display**	**Camera Orientation**	**Robot Navigation Control**
1	Traditional 2 Monitors	Fixed	joystick
2	Immersive Virtual Cockpit via HMD	Head orientation from HMD IMU	joystick
3			LeapMotion
4			Leap Motion™ + point cloud
5			Leap Motion™ + air flow

Table 5.2 The five different test setup combinations for the navigation task.

The procedure can be summarized as:

1. Participant is instructed about the task objectives and procedures.
2. Execute trial with 1 of the 5 setups.
3. Fill questionnaire about user experience.
4. Repeat until 5 trials are complete.

5.4.3 Measurements and questionnaires

The usability evaluation was performed in two parts: the objective evaluation of performance related measures and user subjective evaluation through a questionnaire.

Concerning the analysis on performance we measured the following variables directly from the simulator and/or using a third observer to keep records.

- *Time*: navigation time for each of the 4 path segments between rings.
- *Travelled distance*: The length of the executed trajectory for each path segment.
- *Number of collisions*: number of times the robot collided with the elements of the underwater environment, including the rings.
- *Number of steering compensations*: number of issued steering commands for each path segment.

For the subjective evaluation a questionnaire was created, which was inspired on the IBM Computer Usability Satisfaction Questionnaire (90), as well as on Slater, Usoh and Steed (91) (92) presence questions. The participant feedback was given by classifying on 7 point Likert scales subjects like: usability, easiness, control precision, fatigue, realness, tele-presence and embodiment feeling. The 8 questions to answer were divided in two groups as follows:

Usability & Task load questions

Q1: The interface to control the robot was...

(1=Easy to use, 7=Hard to use)

Q2: How tiring was the task?

(1=Felt tired, 7=Didn't feel tired)

Q3:How precise was the robot control?

(1=Not precise, 7=Precise)

Q4: Performing the experiment was:

(1=Frustrating, 7=Rewarding)

Immersion presence questions

Q5: I had the impression of being...

(1=In the lab, 7=Aboard the vehicle)

Q6: How close I felt from the obstacles?

(1=Felt close, 7=Didn't feel close)

Q7: How real was the experience?

(1=Close to real, 7=Far from real)

Q8: The perceived motion sensation was:

(1=I was moving, 7=The scenery was moving)

5.4.4 Participants

The experiments were performed both at the UC and UJI, with 13 participants from UC and 13 from UJI. The participant group included students and researchers in fields such as engineering and computer science, with an overall average age of 30.12 years. All participants reported normal or corrected to normal vision, while 17 had experience with video games. None of them had prior knowledge of the experience or involved technologies. Participation was voluntary, and research ethical principles were respected.

5.4.5 Results

As previously stated, each participant was asked to execute a set of 5 teleoperation experiments (in random order), and performance related values were collected during each of them. At the end of each experiment, the participants were asked to fill in a short questionnaire related to the subjective perception of usability and immersion. We can divide the participants results into two groups: the quantitative results taking into account the performance in each setup, and the qualitative evaluation.

Task performance related measures

The results are summarized in the following plots for the captured parameters, which are: trajectory time (figure 5.12a), traveled distance (figure 5.12b), and number of steering commands or compensations (figure 5.12c). In these figures w_n represents the trajectories between the n and $n+1$ rings, for each of the 5 setups.

Figure 5.12a presents the mean times and variances obtained by the whole set of testers for each path segment (w_n), and for each of the setups. The ANOVA (analysis of variance) test was applied and showed that the results are statistically significant (marked with an asterisk).

($F_{4,95} = 8.57$, $p < 0.0001^*$ on $w1$ time, $F_{4,95} = 2.94$, $p = 0.0242^*$ on $w2$ time, $F_{4,95} = 6.44$, $p = 0.0001^*$ on $w3$ time, $F_{4,95} = 17.79$, $p < 0.0001^*$ on $w4$ time).

Figure 5.12b presents the mean values for the traveled distances between way points, and the significancy analysis shows that only y $w1$ and $w4$ results are statistically significant. ($F_{4,95} = 2.77$, $p = 0.0314^*$ on $w1$ dist., $F_{4,95} = 0.87$, $p = 0.4798$ on $w2$ dist., $F_{4,95} = 0.67$, $p = 0.6141$ on $w3$ dist., $F_{4,95} = 2.82$, $p = 0.0290^*$ on $w4$ dist.).

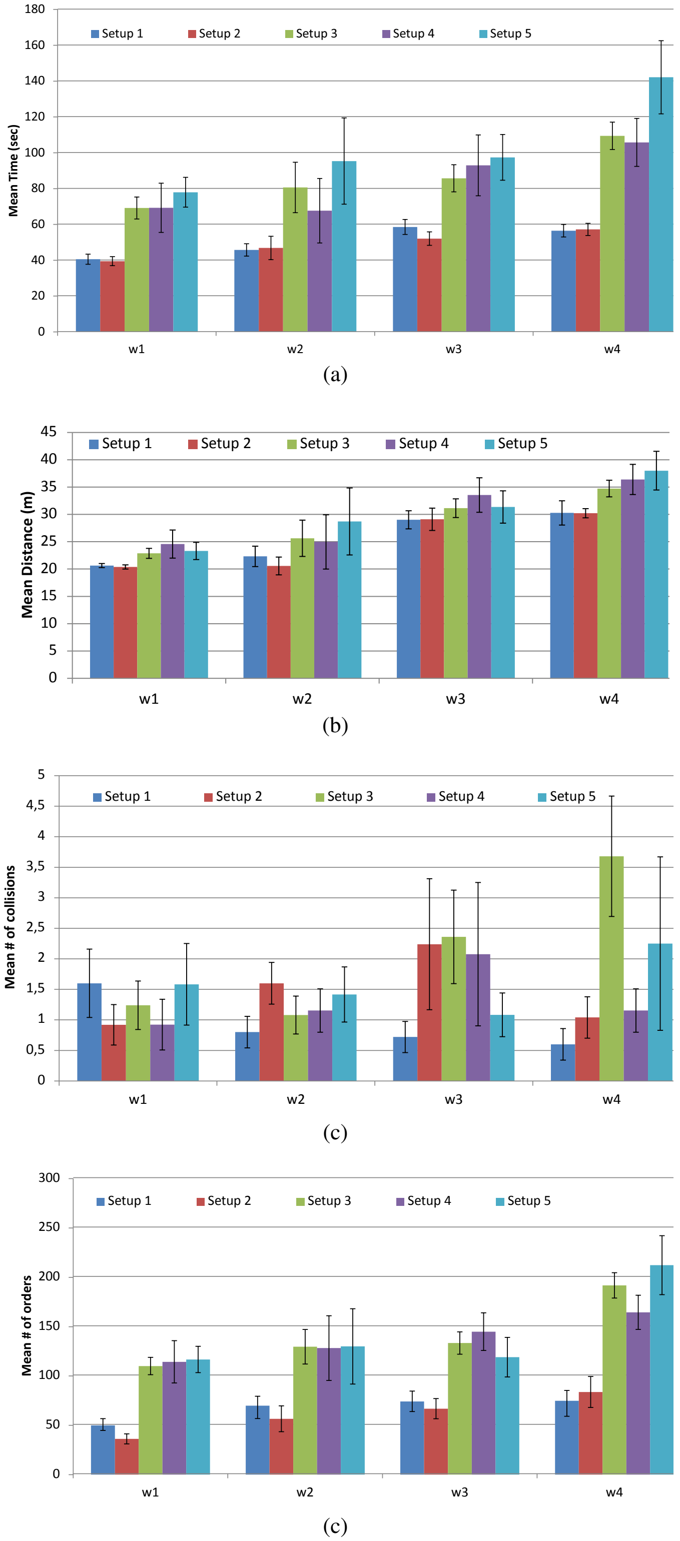
Finally, figure 5.12c presents the mean values of steering commands, and the ANOVA one-way test showed that all the results, except for trajectory segment $w2$, are statistically significant, as follows: ($F_{4,95} = 15.91$, $p < 0.0001^*$ on $w1$ Ord., $F_{4,95} = 3.76$, $p = 0,0068$ on $w2$ Ord., $F_{4,95} = 7.69$, $p < 0.0001^*$ on $w3$ Ord., $F_{4,95} = 15.90$, $p < 0.0001^*$ on $w4$ Ord.).

The plots do not include the number of collisions, as the ANOVA tests show these are not significant from the statistical point of view. ($F_{4,95} = 0.46$, $p = 0.7609$ on $w1$ Col., $F_{4,95} = 0.99$, $p = 0.4168$ on $w2$ Col., $F_{4,95} = 0.87$, $p = 0.4806$ on $w3$ Col., $F_{4,95} = 3.55$, $p = 0.0095$ on $w4$ Col.).

Qualitative evaluation based on user questionnaires

In what concerns the questionnaire presented in section 5.4.3, the results are presented in figure 5.13.

The ANOVA one-way test results are as follows: ($F_{4,95} = 31,59$, $p < 0.0001^*$ on question Q1, $F_{4,95} = 37,97$, $p < 0.0001^*$ on question Q2, $F_{4,95} = 18,45$, $p < 0.0001^*$ on question Q3, $F_{4,95} = 24,57$, $p < 0.0001^*$ on question Q4, $F_{4,95} = 32,19$, $p < 0.0001^*$ on ques-



(a)

(b)

(c)

(c)

Fig. 5.12 Mean values and standard deviation for: (a) trajectory time, (b) trajectory length, (c) number of collisions, and (d) number of steering commands, per trajectory segment and per setup.

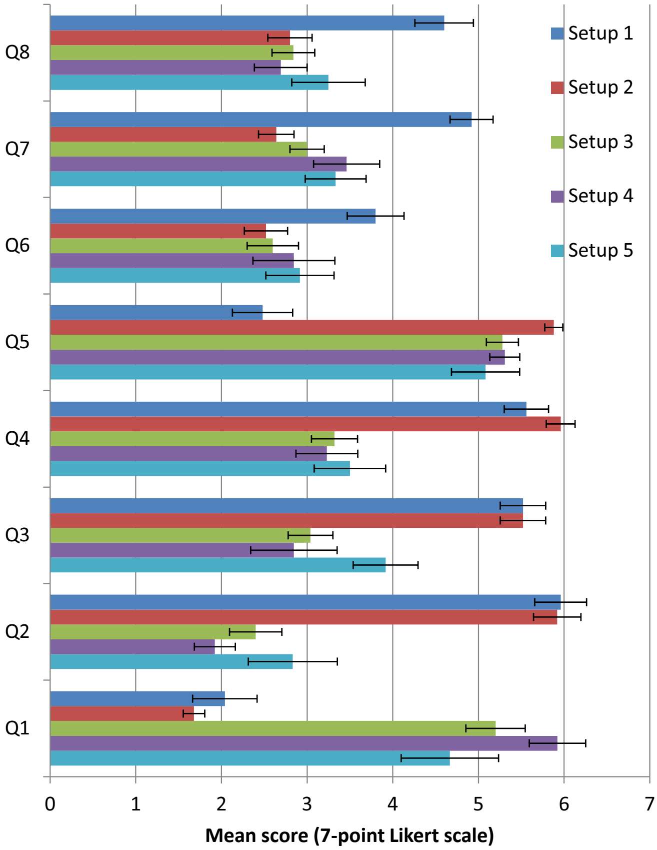


Fig. 5.13 Mean scores for the five setups obtained from the users answers.

tion Q5, $F_{4,95} = 2,86$, $p = 0,0274^*$ on question Q6, $F_{4,95} = 14,06$, $p < 0.0001^*$ on question Q7, $F_{4,95} = 7,46$, $p < 0.0001^*$ on question Q8).

The navigation task performance measures, correspond to the execution of a trajectory, divided in 4 segments ($w_k, k = 1..4$), for each of the 5 setups presented. The task was to drive the robot along, passing through each ring that separates a path segment from the next. The user could adapt to the commands during the first part of the trajectory, i.e. till passing the first ring, and then the all the measures were started for w_1, then after the second ring for w_2, etc.

Each of the trajectory segments had its own particularities and implied complexities, as follows: w_1 - straight forward, w_2 - simple curve, w_3 - hard curve, and w_4 - variations in altitude ending with a curve.

To sum up, these are the devices to use in each setup:

Setup 1 - Conventional teleoperation setup using 2 monitors and joystick, with no camera control.

Setup 2 - VC using an HMD for visualization and controlling camera orientation, and joystick.

Setup 3 - Same as previous, replacing joystick by Leap Motion™.

Setup 4 - Same as previous, with representations of the user and Leap Motion™ inside the VC.

Setup 5 - Same as previous, replacing point cloud by air flow-based haptic for Leap Motion™ localization.

The best mean times, on almost all path ways resulted from using Setup 2, except for *w*4 where operators presented less time using Setup 1. Using Leap Motion™ had generally a negative effect on time performance. Its combination with the air flow solution (Setup 5) showed good results for controlling rotation on a plane, e.g. w_3. For more complex cases of changes on orientation and altitudes (ex: w_4), Setup 4 presented better results. In these cases, operators did not sense air flow changes when moving their hands up and down, but could visualize their hand representation.

The lower distances traveled, on w_1 and w_4 pathways resulted from using both Setups 1 and 2. Leap Motion™ based interfaces led to bigger traveled distances, as users also reported that it is less precise.

The smallest number of steering order is associated with Setup 2 on w_1 and w_3 pathways. When dealing with changes in altitude changes (*w*4), Setup 1 performed better. Setups using Leap Motion™ led to higher number of steering orders.

Operators using Setup 2, the immersive virtual cockpit with joystick, presented better quantitative measures for task time performance, traveled distance and number of steering orders. Setup 1, the traditional approach, played better on challenges involving orientation and altitude complexities, although the operators did not change the camera point of view leading to an increase in the dexterity workloads.

Analyzing answers to questions 1 to 4 we can conclude that: (1) the immersive approaches, with the POV based on head orientation, are both easy to use and intuitive; (2) the users consider it is important to use a precise device for controlling robot navigation, like the joystick; besides precision, (3) perceiving the range limit of the device and sensing some mechanical feedback from the device is important for the users. This, and the fatigue induced by Leap Motion™ use made users tend to choose the joystick as the preferred control device.

From questions 5 to 8 we can conclude that the VC solution is clearly a contribution for the immersion feeling as demonstrated by sensed presence question (Q5). Questions related to tele-embodiment (Q6 and Q8, i.e. virtual contact and self motion) also show a trend to higher immersion rates. Realism question (Q7) presents a moderate trend while the operator perceives the simulator as a game and not as a real environment. The perceived realism of the simulated environment exhibits a correlation with the reported ease of use of the control device. This suggests that the simpler and natural is the interaction, the more immersive becomes the experience.

5.5 Exploring new technologies

There are some new research lines to explore after running the experiments. The ideas of representing the user hands into the virtual world or placing a little fan close to the Leap Motion™, improved the results in terms of user experience. Nevertheless, these solutions can be enhanced to get better results.

Concerning the virtual hands, one of the most RGB-D device used up to now, is Microsoft Kinect™. The main problem when using Microsoft Kinect™, is the effective workspace: the sensor should be placed between 80cm and 4m, the lower limit being the biggest constraint. Fig. 5.14 shows Microsoft Kinect™ location during the setup of the experiments.

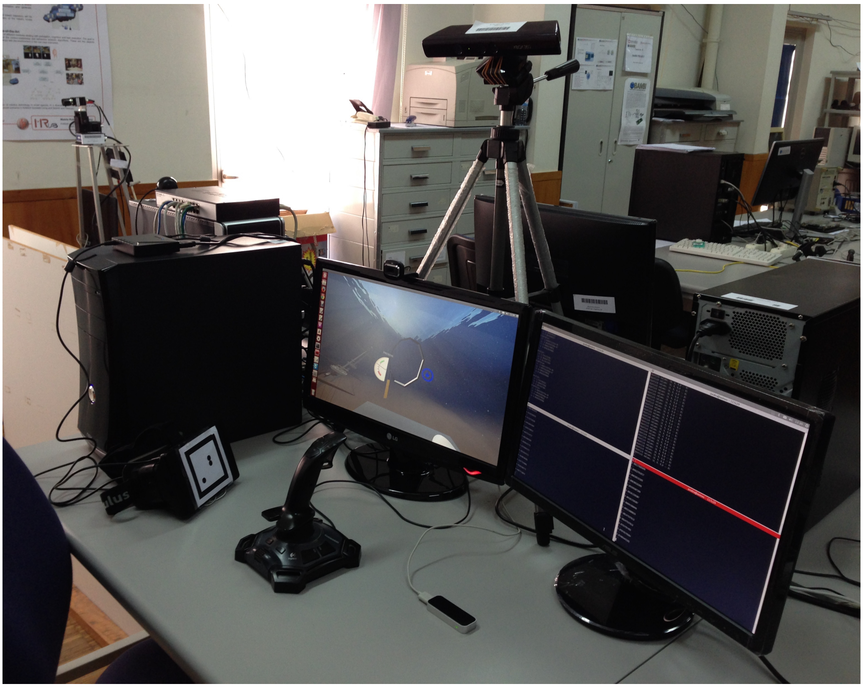


Fig. 5.14 The Microsoft Kinect™ should be place behind the monitors in order to keep the user inside the workspace.

Nevertheless, there are newer devices with similar capabilities, but with a smaller size. Although the previously mentioned Leap Motion™ has been used to track the user hand, this device can not be used to represent the real user hands. It can be used to modify virtual hands generated by the simulator, but the main drawback of this approach, is the computational cost and the lower reliability between the user and their feelings. We tried to put the device on the Oculus Ritf™ and get the camera images, but they were badly distorted. However, depending on the situation, maybe it would be better a user hand representation with low quality, than a better one with a complex setup. An analysis of the precision and reliability of this device can be found in (93).

Another device that can be used for real hand representation, is the Structure™ sensor (94). This sensor is a little bit bigger than Leap Motion™, but its features are better: the precision is better and the work space is also bigger. showing the user hand representation using the OpenCV library. As both Microsoft Kinect™ and Structure™ sensor rely on the same library, we can assume that we can use the Point Cloud Library (PCL) (95) to get a better representation with lower computational requisites and even better integration with UWSim. The PCL is a standalone, large scale, open project for 2D/3D image and point cloud processing.

Finally, once the user hands are reconstructed and included into the virtual cockpit, we can use them not only to interact moving the virtual joystick to control the robot navigation, but also to interact with other virtual elements located into the scene. Thus, we can add some virtual buttons to the virtual table, mapping those buttons with specific functions. Another example of use is to employ the hand representation to define the grasp parameters, following the approach "learning by demonstration".

5.6 Closure

This chapter presented the principle that virtual reality-related immersive systems can be used to induce the telepresence feeling in remote operation of underwater robots and that this can be used to improve the performance task execution. To this end, a system was developed with the objective of virtually placing the operator aboard the remote robot and letting him/her do the driving tasks from there.

To sum up, the combination of the immersive virtual cockpit, with implicit control of the remote camera orientation from the user head orientation, and joystick, has been shown to produce the best results in terms of performance and is preferred by the users.

All these contributions have been published in: (96), (97), (98), (99).

Chapter 6

An architecture to support the interaction with a HROV

Following the two previous chapters where two different approaches for interacting with underwater vehicles were presented, this chapter introduces a new concept both in terms of interaction and operation of these devices.

To better understand the concept, we may recall that chapter 4 proposed specific interfaces for specifying missions for autonomous robots. However there are still substantial limitations in their capacity to take decisions in the presence of incomplete data. This is indeed one of the main problems that hinder their generalised use. Chapter 5 focuses on teleoperation and on how to improve the system and the concept of telepresence was used to simplify its use. Although the proposed approach has shown substantial advantages over the traditional forms of teleoperating a robot, these operations still induce a substantial amount of cognitive fatigue, which in consequence may reduce the attention and increase the possibility of human errors occurring during missions.

This new proposal which aims at leveraging the best of the two previous approaches with a system that operates in an autonomous way (as traditional AUVs) but supporting interventions of the user as needed. These interventions can range from simple adjustments to the mission execution to fully teleoperated actions with smooth transitions between the modes. The next sections introduce the underlying concepts and architectural changes that support it.

6.1 Introduction

As presented in section 2.2, there are basically three different types of robots for underwater environments.

Teleoperated robots or ROV, are the most common robotic devices used in deep intervention missions. They use a tethering cable used to power the robot and support communications, which offers a great flexibility for missions directly operated by human pilots. The existence of the cable, besides imposing restrictions on the manoeuvrability and range of operation, enables the use of wideband communications, and basically removes the power limitations, therefore extending the possible duration of the operations. This may give rise to other points of concern, as long term operations are typically demanding for the operator in terms of attention and cognitive efforts. For the case of autonomous robots, or AUV, there are several commercial models but, as their autonomy is typically reduced as it depends on battery capacity, and the on-board reasoning/decision capabilities are quite limited, they cannot be used in complex missions. However, the lack of physical connection increases the robot manoeuvrability and frees the user from the responsibility of mission execution. Finally, hybrid vehicles, or HROV exhibit both modes and can operate either autonomously or teleoperated, depending on the type of mission. In this group, a few commercial models exist that need to be physically modified prior to being used in each operational mode, while the others allow the switch directly from the mission control room.

The choice of the type to be used depends either on the mission particularities or on the user preferences or characteristics. If the mission concerns a seafloor mapping or a survey, the use of an AUV could be the best option. In this case, once the user has defined the area to explore, the robot can be deployed to execute the mission and emerges when it finishes. Typically, the limitations on the operational time come from the batteries capacity, so if the area to explore is large, a ROV could be more interesting as the power is supplied directly from the vessel. On the other hand, if the mission concerns an off-shore intervention mission, where for example the robot has to open or close a valve, a ROV should be used, because the robot can be endowed with a robotic arm and hand, allowing the pilot to grasp the valve and turn it. The use of HROV could be interesting in a mission, which can be divided in different phases. and where each phase requires robots with different capabilities. For instance, in a mission where seafloor samples are to be collected, the robot may autonomously survey the area and choose the most relevant zone, then it can be used in teleoperated mode to collect the samples.

While an HROV becomes in fact an interesting option, as it can either execute at least part of the mission autonomously (with no user interaction) or being teleoperated (full user interaction), the user support and robot use should be enhanced and improved due to the main disadvantages explained in section 2.2. This chapter proposes a new concept of architecture, derived from the previous one (section 5.3.2) and deals with aspects such as how the user modifies the operational mode, or how the robot may assist the user during teleoperation.

6.1.1 Assisting the human in the loop

Although there is a tremendous increase in pattern recognition and automated control, due to the recent developments on deep neural network-based architectures, humans still play a vital role as pilots in unstructured environments. However, in spite of the fantastic abilities to deal with and take decisions in the presence of incomplete or noisy data, human performance typically tends to degrade through time due to a number of factors. One of the factors affecting performance, is the physical and cognitive fatigue that accumulates during prolonged and intense activities. This is why the development of new cars, planes or other transport means have been including more and more assistive elements,to make the driver or pilot life easier and thus less prone to errors.

The other way is also true, automatic systems are still far from being able to deal with most of the uncertainty levels of real world conditions. The variability of the problems to solve leaves little space for rule based automated systems. However, there is insufficient data about common tasks to use in machine learning-based solutions to draw conclusions in this respect. Hence, having human operators is still the only viable approach for many of the currently performed interventions tasks. The operator therefore becomes an important element of the decision and control loop. Thus, not only should the GUI be developed using a user-centred approach as it was mentioned in chapter 4, but also the architecture must consider the user as a key part of the system. Given the complexity of the missions and associated cognitive efforts, both the GUI and the architecture must be designed to assist the user during the mission.

6.1.2 Existing Architectures for HROVs

HROVs are quite recent and there is not much literature describing their architectures. There are however a few research projects, already mentioned on section 2.4, that produced some documentation and publications publicly available.

Among these projects, we may refer to possibly one of the most relevant research projects that preceded our work, which introduced "HROV-Arch, Control Architecture for an Hybrid-ROV[1]", developed at the DFKI Robotics Innovation Center and MARUM Center for Marine Environmental Sciences (University of Bremen). This architecture was developed via a cooperation between DFKI Robotics Innovation Center and MARUM Center for Marine Environmental Sciences (University of Bremen), and it is focused on controlling a HROV paying special attention to the commutation between AUV/ROV operation modes, and on applying an adaptive fault detection and fault response system. It is based on the integration of the following software packages:

- Open RObot COntrol Software (OROCOS), a general-purpose and open robot control software package, whose development project started in September 2001, and which is divided into three main componentes: OROCOS Toolchain, Kinematics and Dynamics Library, and Bayesian Filtering Library. More information about this project can be found in (100).
- Robot construction Kit (Rock) is a software framework for the development of robotic systems. It contains a rich collection of ready-to-use drivers and modules for use in your own system, and can easily be extended by adding new components. More information about this project can be found in (101).
- Syskit, the Rock system integration and management layer. More information about this project can be found in (102).

According to the description in (103), this approach to control a HROV enables the user to remain in the control loop or allows him/her to be assisted during the intervention phases. Among the features it provides we may refer to:

[1] https://robotik.dfki-bremen.de/en/research/projects/hrov-arch.html

- Aided teleoperation: the user can enable some autonomous services to aid the teleoperation. In fact, only a reduced subset of services (depending on the current state of the mission) are shown. Thus, if the user tries to enable a service, which is not available, the system notifies the user with a message similar to “if you want to start X, then you have to stop Y and Z”.
- Making fault-information human-readable: instead of reducing the communication with the user using icons; this approach shows full-text messages (e.g.: “autonomous trajectory following failed because of DVL loss lock and MotionModel does not provide the necessary precision”).
- Fault response: there are different services that must be running to preserve the system. For instance, if the communication robot-vessel is lost, the robot must detect this failure and must return to the vessel by itself in a safe manner.

Our approach was inspired by these ideas and it will be described in the following sections.

6.2 A Selected Use-Case: Search & Recovery Intervention

As already mentioned in the introductory chapter, there are several applications where underwater robots are useful, the *Search & Recovery* intervention being one of the most common and relevant applications. Typically, this type of intervention can be divided into the following four phases:

- Survey mission: the robot navigates around the area specified previously by the user, taking geo-referenced images of the seabed. These images will be used later to build a seafloor photomosaic.
- Identify ToI: once the seafloor photomosaic has been built, the user identifies and selects the ToI.
- Recovery intervention (see Fig. 6.1): although we know the ToI location and given the nature of the environment, as the seabed conditions can vary from the first immersion to the last one or even the object could be partially hidden or moved, the robot will perform a new short survey to confirm the ToI localization. Once the ToI location is confirmed, the object recovery will be carried out.
- Navigate to the surface: once the object has been recovered, the robot navigates to the surface.

Although this kind of applications are typically carried out with ROVs, some research projects like SAUVIM, and TRIDENT have tested AUVs for the missions. The preliminary results obtained from the use of AUVs are quite interesting, but there are still some problems to solve, such us the need for fully autonomous capabilities to react under certain situations.

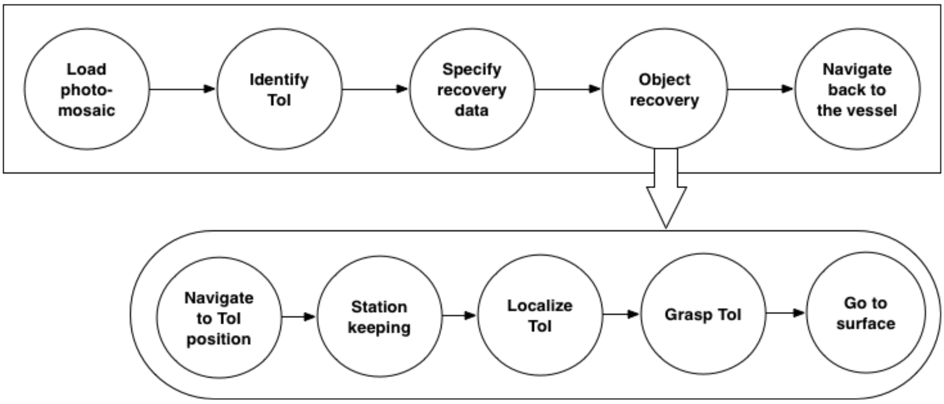


Fig. 6.1 Recovery intervention phases (up), subphases for the "object recovery" phase (down).

6.2.1 Limitations of AUV and ROV for the given examples

By doing a simple analysis relating the above examples to what can be done with an AUV (or ROV), some limitations become immediately evident, as follows.

When using an AUV, the user cannot modify either the parameters or the conditions of the task after it starts the mission. This represents a very important limitation, because it means that the user has to wait until the robot rises to the surface. In a problem similar to the first given example, the robot could be trapped by some undetectable obstacle during the navigation phase, resulting on the loss of the robot. During the grasping phase, the problem is similar: the robot will try to grasp the object several times until the system generates a failure system and the robot rises to the surface without the object.

On another hand, if a ROV is used some of those problems will not appear, insofar as the user has all the responsibility and must take care of the robot and the mission. Concerning the navigation problem, if the robot has been endowed with the appropriate sensors, the user will be notified and would be able to avoid the obstacle. The case of the grasping example can be quite challenging, as the user must simultaneously control the vehicle position, and at the same time control the arm to grasp the object. This can be a complicated task in the presence of underwater currents.

Bearing all this in mind, and considering the above HROV application examples, it becomes obvious that missions will either be constrained due to well known limitations of AUVs, or impose significant stress and cognitive fatigue on the users piloting ROVs.

6.2.2 Two use case examples

In order to understand better how missions could be carried out using HROVs, let's consider the following two examples.

Example 1: The user defines the task and the robot starts the navigation, keeping the user in a supervisor mode. During the navigation, the robot will move towards the goal position, but eventually the sensor equipment (e.g. navigation sensor, pressure sensor, range sensor) does not allow the robot to identify and notice an obstacle. At this moment, when the robot is close to the obstacle, the user can take control of the mission, by just using the input device (e.g. joystick) and modify the trajectory. Then, after the user has avoided the obstacle and has released the joystick, the robot will take control of the task and will try to reach the previously established goal position.

Example 2: In addition to the previous example description, we may consider that during the grasping phase, the user would only need to control the arm/gripper in case of need or failure, as the vehicle can autonomously keep the position stable.

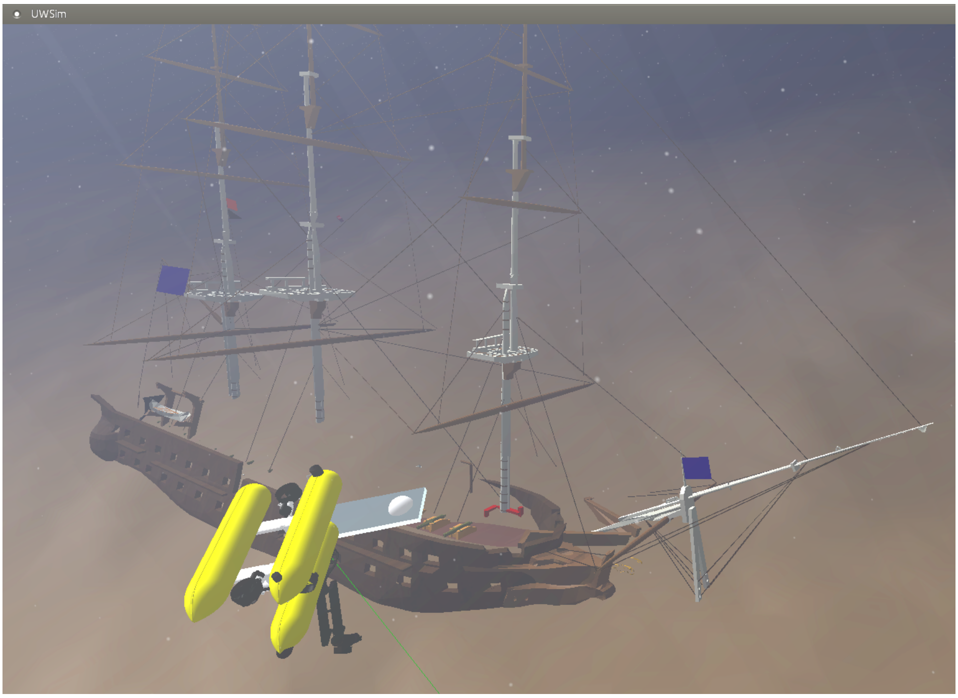


Fig. 6.2 Scene from the UWSim. The robot navigates to a position defined by the user. The shipwreck represents an unavoidable obstacle, because the ropes are hardly detectable by the robot sensors.

Thus, comparing our approach using a HROV with the typical intervention using an AUV or ROV, the user defines the mission and remains as a supervisor until its intervention is needed, reducing the user responsibility and other drawbacks mentioned previously.

6.3 Leveraging the level of the autonomy of the robot

One of the most critical parts when controlling a HROV is the control system of the autonomy of the robot, as it should allow a smooth transition between autonomous and teleoperated modes, without high requirements either from the user point of view or complicated hardware. To attain this goal, the proposed approach, which will be described hereafter, not only allows the control of these transitions, but also supports the user interaction on demand. This enables the user to remain as a mission supervisor after specifying the mission settings, and modify those parameters or take robot control whenever he/she decides to do so, or when the system detects a failure that requires user intervention. Thus, the possibility of having a human controlling the robot will be maintained, therefore taking advantage of the irreplaceable human ability to interpret complex information from different sources and take appropriate action in short-term decisions. In fact this approach also contributes to a more frequent use of autonomous modes as the possibility for the user to take control as needed will solve one of the major limitations of the AUVs, which is their reliability of operation.

Such system will be based on two different controllers, one per each working mode: one autonomous controller, that should allow the user to specify the mission and provide feedback to the system about how the mission is going on, and a teleoperation-supporting controller that should allow an easy way to take control of the robot, by simply using any of the input devices connected.

In order to enable this kind of shared control, we may consider different situations:

- Best case: the robot achieves successfully the goal position without the user intervention. This is the most desired one, because the user specifies the mission and remains as a supervisor while the robot performs the mission autonomously.
- Intermediate case: the robot is carrying out the task, but the user detects an obstacle or decides to modify the task. In this case, the user teleoperates the robot to solve the problem and then releases the input device, letting the robot become autonomous again.
- Worst case: the robot fails in its attempt to achieve the goal position and therefore raising an alarm to be received by the user, who takes the input device to teleoperate the robot and finish the mission.

It should be noted, as previously stated, that the communication problems are beyond the scope of this thesis and we assume that the involved delays and bandwidth are sufficient to enable the control of the robot in real-time.

6.4 Defining a new architecture concept

This section presents a new proposal for changes in the architecture already discussed in section 5.3.2, which has been improved in order to enhance the capabilities and address the problems that appear when working with a HROV. As can be expected, some parts of the architecture will be kept without changes, while others are adapted to fulfil the new requirements. This new architecture is depicted in figure 6.3, and comprises the following four modules:

- *UIAL*, which has been explained in section 5.3.2. Although the general aim is similar, there are some differences when comparing with the previous one.
- *AUV control*, which tries to achieve the goal autonomously, to ensure a successful task execution (e.g. navigate or keep a certain robot position) and sends the mission status result to *HROV control*.
- *HROV control*, which enables the transfer of the mission control to the user or the robot, depending on the situation. Initially, the user selects the task to be carried out by the robot autonomously, although the user will take over the task if the robot fails or the user requests control.
- *An immersive interface*, which allows the user to interact with the robot, following the principles explained in the previous chapter.

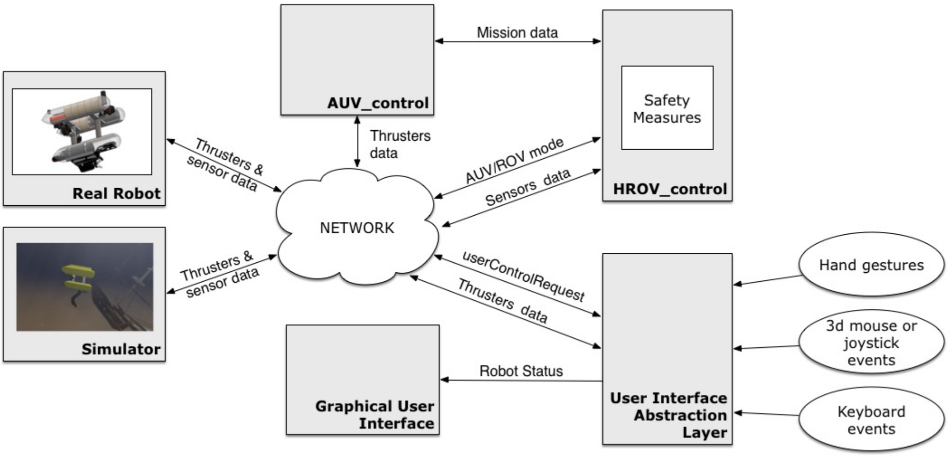


Fig. 6.3 Our shared control approach is based on the architecture introduced previously, but enhanced with new modules.

6.4.1 User Interface Abstraction Layer

From the HRI and usability point of view, AUVs and ROVs have several drawbacks. To control a ROV, the user needs to use multiple joysticks with several buttons, as well as paying attention to multiple screens (e.g. robot sensor information or video-camera images). In the case of AUVs, the user must specify the task to be carried out by the robot using a GUI typically designed to be used by an expert user.

As we explained in section 5.3.2, the main features of the UIAL concerns the HRI, not only to control the input devices, but also to give feedback to the user. As detailed hereinafter, the UIAL is in charge of:

- Receiving the real or virtual robot sensors information.
- Handling input devices, selecting among the existing ones those that are to be used in the ongoing task.
- Filtering information by selecting the most relevant parts for the mission and displaying them to the user.

The architecture proposed here is compatible with the use of both the real robot and the simulator. The UIAL reads the information related to the aforementioned items (sensors, input devices and specific mission parameters, making it available for other modules of the architecture after processing it and generating new data.

Concerning the input devices, a joystick, a 3D mouse (SpaceNavigator™) and a hand tracker (Leap Motion™) have been configured and tested, representing also different interaction approaches. The joystick is one of the most common input devices used with simulators and to control robots, but its main downside is that the user needs to practice and learn its configuration in particular in the case of complex joysticks with several buttons and functionalities. In the case of using a SpaceNavigator™ or a Leap Motion™, the user tends to feel more comfortable and confident, as both devices are intuitive (e.g. when the user opens/closes the hand, the robot gripper will be opened/closed). In practice, all these devices can be connected at the same time with predefined specific roles for each of them, or can be configured to be used independently depending on the user preferences or expertise.

Usually, there is a large amount of information generated during a mission, typically spread across multiple displays, that the user has to pay attention to. In fact, during some ROVs interventions, there is commonly another user in charge of paying attention to those displays. In order to reduce the user cognitive fatigue, the UIAL reduces the amount of information to be shown to the user, filtering in only the most relevant data depending on the mission phase. It also shows the information in different ways: some messages will be displayed as text and other information will be displayed as icons. For instance, during a grasping task (e.g. the robot should grasp an amphora or an object located on the seafloor), the UIAL would show up information related to the arm or the gripper, not about the vehicle. Of course, if the system detects a problem concerning the vehicle (e.g. the underwater currents move the robot to the seafloor and the robot gets a position too close to the seafloor), the UIAL will show up a warning message or icon.

There are small differences between what it was presented in section 5.3.2) and this new approach, in which some features were modified and new ones were added to fulfil the requirements. For instance, the Robot Safety Measures (RSM) (explained in section 5.3.5), which was in charge of taking care of the robot, has been moved from this module to the one with specific interaction between the robot controller and thrusters (the *HROV control*). More details about this can be found in section 6.4.3.

6.4.2 AUV control

The *AUV control* module is in charge of carrying out the task autonomously. It has been designed only to control the navigation of the vehicle, so currently only controls the vehicle navigation to the desired position. In the future, this module can be improved with additional features, so it can be used to control some operations with the robotic arm, such as grasping an object or opening/closing a valve, to mention only some examples.

As explained in the previous chapter, the system has been tested in simulation using the UWSim and the same configuration was used to prepare the real test in MERBOTS project. In order to use this module with other robots or mission types, some details, such as thruster configurations, or the sensors available must be taken into account.

This module consists of two cascaded controllers, as shown in Figure 6.4. The first is responsible for obtaining a reference speed according to the position of the vehicle and the second tries to achieve this speed by sending effort to the thrusters, depending on the speed at which the vehicle is currently moving.

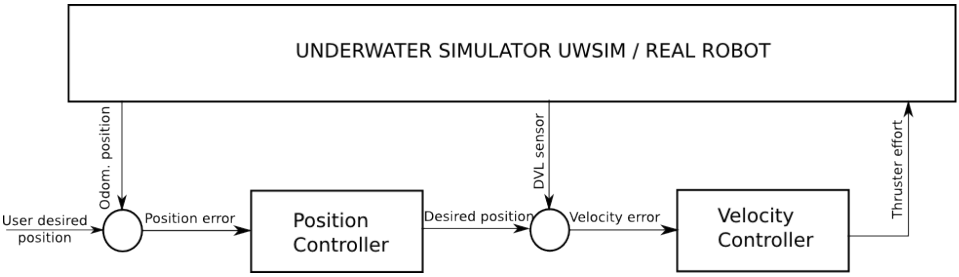


Fig. 6.4 Robot controller layout.

The position controller uses the estimated vehicle odometry and desired position to measure the error. Using this error reference, the speed can be determined so the vehicle can reach the target position. It should be noted that we are trusting odometry which may be prone to drifting, so an absolute measure is recommended to avoid it. A simple proportional controller (reference speed is proportional to the error) with a maximum speed has been used.

The speed controller is similar, but instead of using odometry estimation, a DVL sensor has been used. It is also necessary to map the output of the vehicle thrusters configuration as efficiently as possible. In this case, the vehicle has two engines facing forward, two facing the surface and one that allows lateral movement. Although there are other possibilities, the frontal thrusters have been used for displacements and rotations on the surface plane (yaw), thrusters facing surface to control the depth of the vehicle and the lateral displacement thruster to move on that axis. Other possible rotations are not considered (roll and pitch). This speed controller is also used when the user control loop sends signals directly to the vehicle, as it is extremely difficult to control the thrusters directly and requires prior knowledge about their configuration. The main goal with this approach is to abstract the user from the specific vehicle configuration, which may change during the mission.

Thus, when the user takes control of the vehicle, he/she sends Cartesian velocities to the controller and he/she can easily move the vehicle.

Although more advanced control techniques can be employed, this simple strategy tested here has been demonstrated to be sufficient for the current purposes.

6.4.3 HROV control

This module can be considered the most relevant of our proposal, because it is in charge of controlling the robot autonomy, has access to the robot data, and interacts with it both the robot and the user. The main features of this module are:

- Control the different phases of each mission type.
- Control when the robot should operate as an AUV or a ROV.
- Implement specific functions to ensure the safety of the robot (RSM).

As stated in Fig. 6.1, each mission can be divided into different phases allowing the system to control if each of them has been finished successfully (the system starts the next one) or not (the system requests the user to take control of the robot). We consider a mission failure when the robot gets stuck (not only in a navigation task, but also if the robot generates error messages during a task execution) or there is an RSM alarm. This information will be processed by the *HROV control* to generate the appropriate feedback to the user.

In the proposal here introduced, it is up to the user to determine the robot operational mode, according to the functionalities endowed to the system. Regarding the MERBOTS project, there were some functionalities that the robot can accomplish autonomously, such as the navigation between waypoints, grasping the ToI or opening/closing a valve in an intervention panel. Fig. 6.6 shows some of these options. Although the robot can perform these tasks autonomously, the user can take control of the task by simply interacting with the input device at any time. At that moment, all the algorithm-generated instructions will be ignored and the system will react according to the user inputs. Only when the user releases the input device, will the system evaluate the current status to determine if the algorithm should continue or the task should be checked as finished and move forward to the next phase.

As stated before, the user may introduce errors involuntarily in the system. The RSM, already introduced in section 5.3.5, is an algorithm focused on preventing robot damages and assisting the user during the teleoperation. For instance, the robot will not go deeper when the maximum depth has been reached or if the depth will reduce the space of work of the arm, even if the user sends the commands to go deeper. When some of these situations occur, the system will warn the user by showing up a message or icon depending on the relevance. In this new version, some extra features have been added, such as navigate to the surface autonomously if the robot loses the communication with the pilot during a teleoperated phase or improvements concerning the *station keeping* functionality (this means keep the position at certain position) while the arm/gripper is working.

6.4.4 The immersive interface

Once the architecture is presented, the next key aspect within the proposed architecture is the user interface. Following the proposal presented in 5.2, the evolution presented here emphasizes how the user interacts with the system through the multi-level menu.

The sensation of immersion is achieved thanks to the 3D engine that supports the use of an HMD, such as Oculus Rift™.

The benefits of using a HMD like this are: abstraction of the environment to avoid distractions, sense of realism or immunity to sunlight (compared to using an outdoor screen). Despite these benefits the loss of spatial references to the environment can be considered as a constraint in terms of usability. In other words, once the HMD is being used, it is difficult to use input devices such as mouse-keyboard or Leap Motion™, since the reference to its location is lost.

The user interface acts as a new visual layer on the UWSim simulator and consists of two parts: the user menu and a VC.

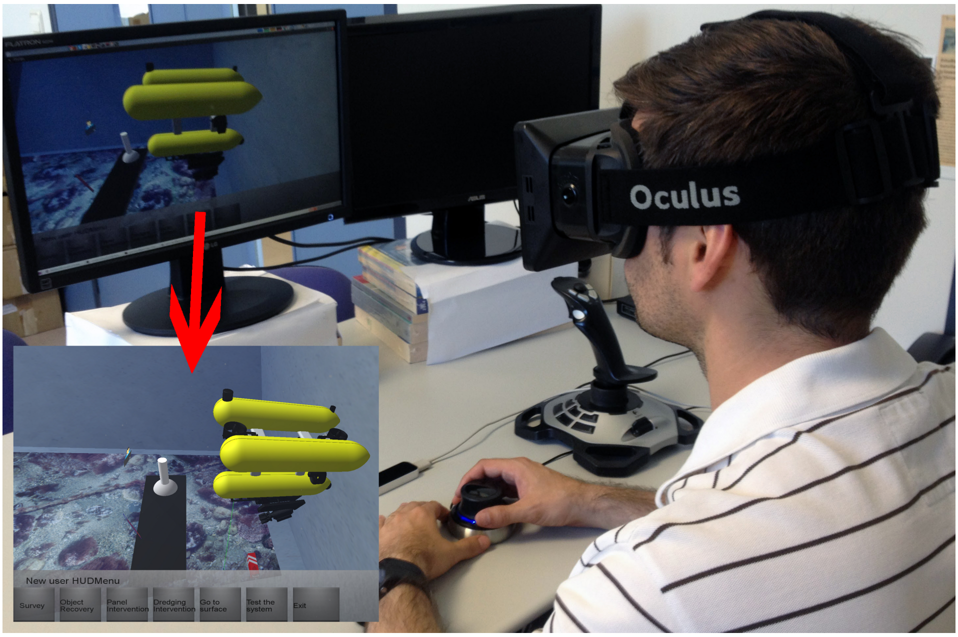


Fig. 6.5 Example of the user menu within a UWSim scene.

Since the UIAL module supports different input devices, it is possible to use any of these devices (in our case a SpaceNavigator™) to interact with the Head-Up Display (HUD) menu. This allows the user to show, hide and navigate through the menu and to select the desired menu options. Fig. 6.5 shows an example of the main menu, which appears and disappears on demand by the user. The menu is compatible and integrated with UWSim, having access to all its functions as a new visual layer. The options in this menu refer to high level instructions (e.g. survey specification, panel intervention or object recovery). Once the user has selected the desired option, a new menu will appear with the options available for that selection.

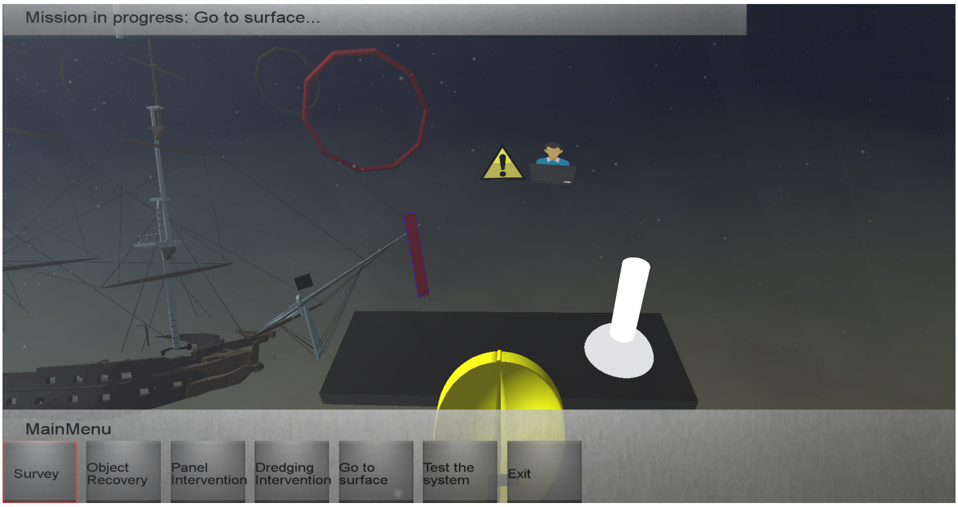


Fig. 6.6 HROV GUI: user feedback at top, user menu at bottom.

Regarding VC, the main idea is to reduce the amount of visual information that the user should pay attention to. Instead of the traditional teleoperation interfaces that use manual control of the camera and where the visualization of information related to the mission is achieved through a set of monitors, our immersive approach enables a natural egocentric view of the underwater environment and robot controls. For this, the VC includes:

- **Virtual Cockpit**. Following the approach introduce in section 5.3.4, the user is located inside the vehicle and uses the head to move the virtual camera. This generates realism and places the user in the scene.
- **A virtual visualisation of a physical table**. When the user wears a HMD, he/she looses spatial references. If the input devices are located on a physical table not represented in the user's view, it would simultaneously be awkward to touch it with seeing it and would make it difficult to locate and reach the vehicle control devices. The virtual table serves to avoid such disassociation, and at the same time, supports the representation of other input and status display devices on the same workspace.
- **Virtual joystick**. The system must provide feedback to the user in order to enhance its usability; at the same time it adds visual feedback regarding the vehicle movement. This virtual joystick represents the order sent by the user to the robot. When the user manages a joystick, the mapping between the movement and the input commands become easier to understand. Nevertheless, in the case of using a hand tracker as input, this real-virtual relation is even more complicated and this representation solution intends to compensate for the lack of haptic sensory information. In addition, as already mentioned, when the user wears the HMD, it helps the user to properly position the hand over the device.

- **User intervention request indicator**. It is activated when the *HROV control* requests the intervention of the user or when the user him-/herself requests the control of the robot.
- **Alarm signal**. It is triggered when there is an alarm coming from RSM requesting user interaction
- **Status indicator**. Instead of showing the user numeric values of depth or distance to the seafloor, it is easier to get this information from a safety status indicator.

To sum up, the use of this immersive interface is focused mainly on two aspects: improve the usability of the robot teleoperation and try to solve the lack of feedback given to the user by traditional interfaces. The first issue was solved using a properly integrated menu displayed in a HUD representation integrated in UWSim, and the second one was covered by using visual graphical communication (icons) and by reducing the amount of information to prevent cognitive fatigue.

6.5 Closure

This chapter introduced a new approach to control a (HROV), which may operate in either autonomous or teleoperated mode. This approach tries to merge the benefits of both, autonomous and teleoperated systems, while eliminating some of the main disadvantages of each of them, while enabling the possibility of having smooth transitions between these modes. Through this approach, the user specifies the task to be carried out autonomously by the robot and continues to play the role of supervisor of the mission. Nevertheless, the user will be able to teleoperate the robot whenever he/she decides it is necessary, or in the event that the system should detect a problem and requests the user to intervene. Once the user finishes the teleoperation, the robot becomes autonomous again without the need to perform any special reconfiguration.

To support this hybrid mode of operation, an architecture has been introduced. This architecture is made up of four modules, which control the principal parts of the system: one module controls the HRI, another module controls the task to be carried out autonomously, a further module is responsible for deciding when the robot should operate in autonomous or teleoperated mode, and the last one is an immersive interface in charge of providing a proper interaction between the robot and the user.

All these contributions have been published in: (104), (105), (106).

Chapter 7
Final remarks

This final chapter concludes and summarizes the work presented previously, summing up this thesis by reviewing the content described in each chapter. Then, the chapter concludes with related future lines that can be addressed as new research lines.

7.1 Conclusions

This thesis was developed in the context of the latest research projects coordinated by our research lab (IRSLab) and it is focused on research in different aspects related to HRI: the use of algorithms to assist the user during mission specification (making the user interface easy to use and capable of assisting the user himself), exploring a multimodal interface (by using different devices to interact with the interface) and the proposal of a robot control architecture (allowing the user to change the operational mode, from autonomous to teleoperated, or vice-versa).

Aiming at providing an adequate user interaction support, two different GUIs have been proposed and developed, focused on the use of a specific underwater robot (AUV and HROV). The first one, the UWUI was related to the Spanish and European research projects RAUVI and TRIDENT, and was used with an AUV. With it, the user was able to load seabed images and it also provides automatic methods for object detection and task specification. Thus, this is a way to assist the user, reducing the previous expertise background requirement and reducing also the stress attached to very complex interfaces. More information about this development and some videos are available here[1]. The second one, related to the Spanish research project MERBOTS, was developed to interact with a HROV. The main features rely on the possibility to get feedback from the robot and supervise the mission by reading the mission data in a user-friendly GUI. This research line was driven by the motivation of improving the complex and difficult-to-use GUIs used for piloting an underwater robot, improving and enhancing the User eXperience (UX), the reduction of the user cognitive workload and stress, and enabling the use of the GUI without the need to learn a specific language to manage it or the robot.

[1] http://www.irs.uji.es/info/uwui-underwater-robot-user-interface

Following the previous research line, it makes sense to explore new ways regarding HRI. After some time using these GUIs, some questions arise easily: what if this interface or features could be inside the simulator used to test and verify the mission? Undoubtedly, this will enhance the telepresence and embodiment feeling. Due to the fact that the IRSLab was in charge of the UWSim development, a new layer was developed and merged into the main branch. To this end, a system was developed with the objective of virtually placing the operator aboard the remote robot and allowing him/her to do the driving tasks from there. The immersive system combines the images, obtained from an adjustable camera on the robot, with virtual instruments. This combination is displayed to the user through the HMD, which tracks the user's head movements to modify the POV camera. Additionally, the system was improved by adding to the scene the user's own representation. The first experiments carried out to test this approach encourage us to continue this research line, but with a new focus on input devices which reduce the interaction complexity and are classified under the NUI approach. This task was accomplished using devices such as a 3D-mouse or a hand tracker device, a device which localizes the user's hand. Nevertheless, the addition of these devices involved the modification of the system architecture, generating a new research line. The goals achieved with this work were evaluated through a test, not only controlling the robot with the simulator, but also with a questionnaire. The evaluation results showed that the immersive system was preferred by the users. Furthermore, when compared with the traditional interfaces, the use of this immersive system has a positive effect on teleoperation performance. Concerning the input devices, the results indicated that the lack of physical interaction with the device (in case of using the hand tracker) has a negative effect on the observed user performance and an arm fatigue appears, which results from "keeping the hand in the air". Nevertheless, in terms of interaction and controlling the robot, non-expert users feel more comfortable than using traditional devices. To sum up, the combination of the immersive virtual cockpit, with implicit control of the remote camera orientation from the user head orientation and joystick, has been shown to produce the best results in terms of performance and is preferred by the users.

At this point, the aforementioned research lines are focused on how we present the information (the GUI) and how the user interacts with the system. In both cases, using the approach to place the user in the control loop (but not delegating all the responsibility) and the approach of "user assistance". Concerning the latter approach, we can find several definitions in the literature. Nevertheless, the one which fits better in this case is "any interface element that helps the user to use the software correctly, whether this element is a button or algorithms used inside the interface". In our approach, algorithms gain special relevance bearing in mind the problems mentioned in section 2.2, and the main contribution presented in section 6.4 addresses the development of an architecture for controlling a HROV (an underwater robot which can work as autonomous or teleoperated). This architecture, comprised of four modules, improves the early development (see section 5.3.2) adding new features. The main features can be grouped in the following lines: enabling the smooth transition from autonomous to teleoperated robot based on the pilot's needs, where the pilot can modify the robot task execution and get back to autonomous mode; reducing the information which must be presented to the pilot through the VC, which means an adaptative interface focused in reducing the cognitive fatigue; enhancing the feedback to the user during the mission, getting the pilot information about the task when the robot operates as autonomous and there is a connection between them, the pilot and the user; adding specific features to the architecture, which take care of the robot (and hence removing this responsibility from the user); and enabling the pilot the use of different input devices at the same time.

To sum up, this thesis covers along with chapters 4, 5 and 6, the five points defined in section 1.2 and explores new ideas, that appeared during the research. Part of the work here presented (e.g. GUIs) has been used in different research projects led by the IRSLab: RAUVI, TRIDENT, MERBOTS; and other parts of the work were used to explore new features for the UWSim. Additionally, this work has been presented at national and international conferences, and it has been published in proceedings, journals and in a book chapter.

7.2 Future work lines

This thesis works on different research lines focused on underwater robot missions. As mentioned in chapter 1, the thesis has produced several contributions (section 1.4) and publications (section 1.5) related to those research lines, trying to tackle some of the common problems that concern the HRI with underwater robots. However, each of the contributions presented in this thesis can be improved. Beyond these improvements, the work presented opens several opportunities for future work and proposes several open questions that would enhance and expand the research conducted throughout this thesis.

Graphical User Interface

The concept of GUI has evolved considerably during the last years, paying additional attention to concepts like UX, which includes the UI and the usability. In chapter 4 this thesis explores different GUIs, focused on the context of the research projects coordinated in our research lab. Although those developments must fit the needs and be compliant with the requisites of both the project and the architecture used, diverse techniques to improve the UX were evaluated and implemented: “which information must be shown to the user?”, “how this information should be shown to the user?” and “how specific algorithms can assist the user?”. Nevertheless, these questions do not cover all the areas defined by the UX and must be considered closer to the UI design.

The concept of usability is defined as a measure of how well a specific user in a specific context can use a product/design to achieve a defined goal effectively, efficiently and satisfactorily. The usability improvement can be reached from multiple approaches such as interaction design, visual design and information architecture. The interaction design is focused on developing interfaces that are able to get and maintain the attention of the user, and understanding how users and technology communicate with each other. This understanding between users and technology allows the system, just to mention the most relevant benefits, to anticipate user wishes or to adapt the information structure. On the other hand, visual design is used to focus on the aesthetics and the selection of images, colors, fonts, and other elements. A successful visual design enhances the user’s engagement and helps to build a relationship of trust and confidence in the system. Finally, information architecture concerns organizing, structuring, and labeling content in an effective and sustainable way, and the main goal is to help users find and use the information successfully.

Since these proposed GUIs were developed until the time of writing this manuscript, technological evolution has allowed us to run more complex applications on mobile devices such as tablets, power laptops with a performance equivalent to traditional ones or use the cloud to process the most complex algorithms acting the physical device as a visualizer. From a software point of view, there have also been interesting movements, such as the paradigm shift from proprietary to multi-platform libraries (e.g. .NET) or the new libraries focused specifically on UI design.

Multimodal and 3D immersive user interface

Just to continue with the last point mentioned before, the hardware evolution over the last few years has been astonishing. Fields like VR have explored considerable growth from early devices to the final consumer version with enhanced capabilities. In section 5.2, the Oculus Rift™ DK1 (Development Kit 1) was used to explore the immersive teleoperation system. The latest version of this device (and other similar like Sony PlayStation™ VR or HTC Vive™) has several improvements: high-resolution screens with high refresh rate, controls for both hands to interact with the virtual environment or headphones included in the headset. In the case of Oculus Rift™ S, there are two tiny cameras included, so you can enable the vision of the real environment on-demand, which concerns the proposal of the virtual table explained in section 5.2.3.

Obtaining an immersive UI in 3D is not only about using a VR viewer, but also about ensuring that the user is completely immersed in the virtual environment and is able to interact in it. The aforementioned devices enhance the sensation of immersion thanks to precise tracking of the user's entire body, the use of dedicated controllers equipped with buttons, which capture the position and movements of the hand, and the inclusion of audio. On the other hand, new ways to track the user's movements (not just the head as shown in chapter 5) unite the sensors included in the headset, making the control of the camera point of view more natural and realistic in the interaction.

The combination of all these options opens new research lines like improving the embodiment or the teleoperation. Despite the problems related to ergonomics/human factors when the user is standing (as mentioned in the appendix A), the chance to track all the user's full-body and not only some of his/her parts enables and improves the immersive capabilities.

Improving the robot capabilities, communication and architecture

Without doubt, the following years will become more interesting and probably, some of the current problems that have been omitted in this Ph.D. will be solved. Projects like Ocean One (see more information in section 2.4) and other cutting-edge technology projects that are arising at this moment, will push the technology to new limits. The robot itself generates several research lines like exploring different shapes (nowadays most of the underwater robots have a similar shape like a torpedo or squared), improving the battery autonomy (this will increase the operational time underwater), enabling new features and capabilities (there are several sensors used in-ground or aerial vehicles that can be interested in underwater environments), and so on.

Nevertheless, one of the main constraints for AUVs and HROVs concerns the lack of feedback to the user. There are some approaches to communicate the robot with the surface without a tethering cable, but this field has not been achieved successfully. Solving this problem will help to enhance the development and use of those robots in more tasks/missions, gathering both robot and mission information in real-time. Solving this issue benefits not only usability but also in carrying out more secure missions.

Although there are several architectures proposed in the literature for almost all kinds of robots and purposes, chapter 6 introduces a new concept of architecture focused on HRI and the use of HROV. However, this architecture can represent a starting point for new research exploring cloud computing and AI/Machine Learning (ML) algorithms. The use of cloud computing would reduce even more the hardware used in both the robot and the vessel, as was explained before. On the other hand, the use of AI/ML algorithms will represent a new paradigm in HRI, predicting user needs and enhancing the GUI capabilities.

Cybersecurity issues

Cybersecurity is one of the most common topics in news almost every day, regardless of the company size and market. At the moment, cyber-delinquent are more focused on tech companies, medical, insurance companies and banks, but other companies focused on the industry will be targeted soon. We can face several questions regarding security in industrial environments: what will it happen if a company suffers an attack modifying the parameters used in the robot configuration? If they discover the attack soon (most of the attacks are found out more than 60 days later), the company can react soon and restore the original robot configuration. The main question here is: what has happened during this period? In the case of underwater robots, the risk could be the modification of some sensors configuration, damaging the robot or getting the robot lost.

Robots can be considered close to the concept of Internet of Things (IoT), where interaction with the environment is more relevant, leading to easy integration and communication, than secure each of these features. It is important to bear in mind that some of these robots are currently (or will be in the near future) dealing with confidential data (e.g. medical surgeries), so cybersecurity should be considered by default and by design.

The research lines that appear in this topic can be addressed to improve the security in how the information is managed, secure communications, improve the concepts of integrity and authentication, and leave obsolete protocols used mainly in industrial environments. Of course, we can excuse or avoid cybersecurity simply because the middleware or the architecture does not fit them. It is the responsibility of each developer to apply the best during each development. For instance, ROS has evolved considerably and the latest version includes cybersecurity measures[2].

[2] https://ubuntu.com/blog/what-is-sros-2

Appendix A

Some considerations on human factors for new input devices

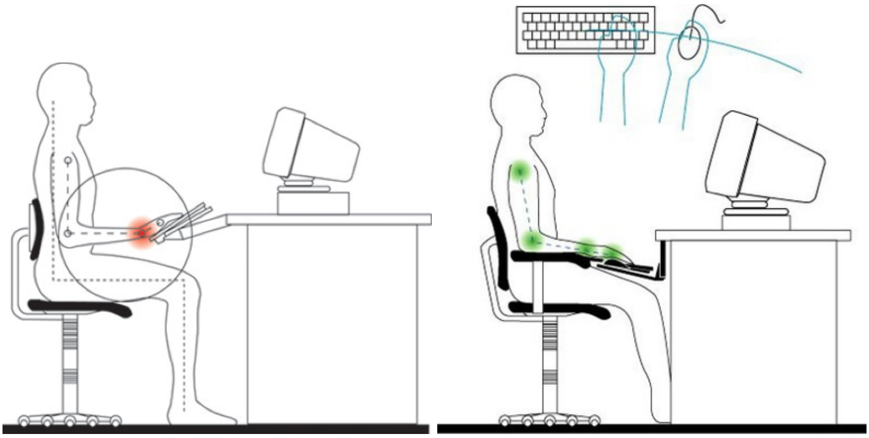
During the experiments carried out to test the different setups shown in section 5.4, the concept of human factors represents a main concern. Some users commented out how the new input devices appear to be easy-to-use and comfortable, but in some case only for a short period of time. Due to human factors can be considered as a new research line, this annex could be considered as a starting point in this direction.

According to the International Ergonomics Association[1], "human factors is the scientific discipline concerned with the understanding of interactions among humans and other elements of a system, and the profession that applies theory, principles, data and methods to design in order to optimize human well-being and overall system performance".

Films like "Minority Report", where the actor interacts with a computer using nothing else than his hands, influence the way we think we could interact with computers in the future. Nevertheless, the technology is evolving everyday quickly and appear new devices such as depth sensors, which provides depth image data, enhancing the way the computers are used. Keeping in mind the human factors, these devices are so new in the market, that most of them need to be studied. If fact, most of the times the user should know which is the best position and how use it depending on the task to be done, in order to avoid problems and get comfortable. Fortunately, some manufacturers are aware of humans factors and suggest some tips to use their products giving some suggestion about how to use and how not to use.

Despite of the technology is evolving everyday and there are several researches (107) to help us to find the best position using a computer, new devices such as depth sensors (e.g. Leap Motion™ or Microsoft Kinect™) are still under research. Due to the different possibilities to use them, the user should know which is the best position and how use it depending on the task to be done. For instance, in the Leap Motion™ website, can be found some suggestion about how to use and how not to use. Fig. A.2 shows the correct way we should use it (up) and how we should not use it (down). In our opinion and after testing these sensors in different positions, we found that:

[1] `http://www.iea.cc/whats/index.html`



(a) Negative use of the keyboard with a positive angle inclination force the wrist position

(b) Keyboard and mouse should be place together, making it accessible just rotating the arm from the elbow.

Fig. A.1 Some tips about the use of a computer, concerning the ergonomics point-of-view.

- If we can work in glass table, we can create a support to place the Leap Motion™ under the glass. Thus, the interaction range is lower and the user does not need to raise the hands until an uncomfortable position.
- The are two locations where to put the Leap Motion™: in front of the keyboard to interact more confortable with both hands or placing it in the mouse position if we are using it as a mouse.
- When using these sensors more than an hour, the user can feel uncomfortable and gets the arms tired.

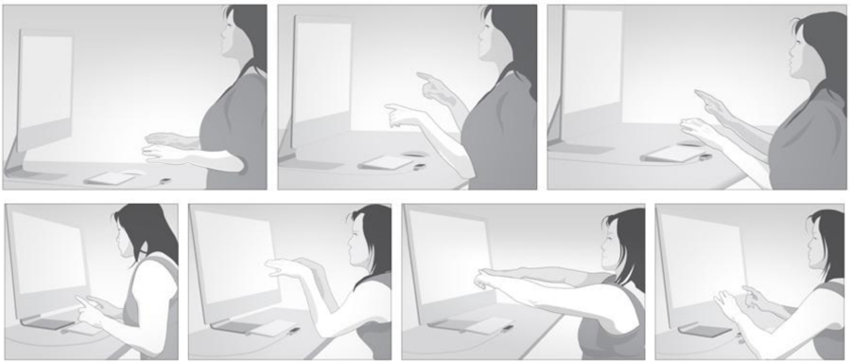


Fig. A.2 Some tips about how to use the Leap Motion™ sensor, according the manufacturer.

In addition, from the software point of view, only few applications could make use of these sensors and implement specific gestures to interact with it. Thus, the user needs to learn how to interact with each sensor and each application, and this represents another problem.

Keeping in mind all these factors, the proposal we presented in section 5.2 is focused on how to connect all the different devices and the way the user interacts with them, paying our attention in two aspects:

- The user experience. The immersive perception produces realism but at the same time, reduce the possibilities to get feedback from multiple monitors or even multiple windows inside the simulator. So, in our proposal, the user gets only the most relevant information to achieve successfully the mission.
- The intuitive gestures. In our proposal, the user interacts with the system using intuitive gestures: moving the hand upper, means the robot will navigate towards the surface.

References

[1] T. B. Sheridan, W. L. Verplank, Human and computer control of undersea teleoperators (1978). (Cited on pages xiii, 33, and 34.)

[2] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, A. Ng, Ros: an open-source robot operating system, in: Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA) Workshop on Open Source Robotics, Kobe, Japan, 2009. (Cited on pages 4 and 77.)

[3] B. Boyle, R. McMaster, J. Mixon, Teleoperation of an underwater robotic repair system using graphical simulation, in: IEE Colloquium on Control of Remotely Operated Systems: Teleassistance and Telepresence, 1995, pp. 2/1–2/4. doi:10.1049/ic:19950650. (Cited on page 13.)

[4] D. M. Lane, J. B. C. Davies, G. Casalino, G. Bartolini, G. Cannata, G. Veruggio, M. Canals, C. Smith, D. J. O'Brien, M. Pickett, et al., Amadeus: advanced manipulation for deep underwater sampling, IEEE Robotics & Automation Magazine 4 (4) (1997) 34–45. (Cited on pages 20 and 44.)

[5] S. Choi, G. Takashige, J. Yuh, Experimental study on an underwater robotic vehicle: Odin, in: Proceedings of IEEE Symposium on Autonomous Underwater Vehicle Technology (AUV'94), IEEE, 1994, pp. 79–84. (Cited on page 20.)

[6] H. H. Wang, S. M. Rock, M. Lees, Experiments in automatic retrieval of underwater objects with an auv, in: 'Challenges of Our Changing Global Environment'. Conference Proceedings. OCEANS'95 MTS/IEEE, Vol. 1, IEEE, 1995, pp. 366–373. (Cited on page 21.)

[7] V. Rigaud, E. Coste-Maniere, M.-J. Aldon, P. Probert, M. Perrier, P. Rives, D. Simon, D. Lang, J. Kiener, A. Casal, et al., Union: underwater intelligent operation and navigation, IEEE Robotics & Automation Magazine 5 (1) (1998) 25–35. (Cited on page 21.)

[8] G. Marani, S. K. Choi, J. Yuh, Underwater autonomous manipulation for intervention missions auvs, Ocean Engineering 36 (1) (2009) 15–23. (Cited on pages 21

and 42.)

[9] J. Evans, K. Keller, J. Smith, P. Marty, O. Rigaud, Docking techniques and evaluation trials of the swimmer auv: an autonomous deployment auv for work-class rovs, in: MTS/IEEE Oceans 2001. An Ocean Odyssey. Conference Proceedings (IEEE Cat. No. 01CH37295), Vol. 1, IEEE, 2001, pp. 520–528. (Cited on page 21.)

[10] J. Evans, P. Redmond, C. Plakas, K. Hamilton, D. Lane, Autonomous docking for intervention-auvs using sonar and video-based real-time 3d pose estimation, in: Oceans 2003. Celebrating the Past... Teaming Toward the Future (IEEE Cat. No. 03CH37492), Vol. 4, IEEE, 2003, pp. 2201–2210. (Cited on page 22.)

[11] P. J. Sanz, M. Prats, P. Ridao, D. Ribas, G. Oliver, A. Ortiz, Recent progress in the rauvi project: A reconfigurable autonomous underwater vehicle for intervention, in: Elmar, 2010 Proceedings, IEEE, 2010, pp. 471–474. (Cited on page 22.)

[12] J. Albiez, A. Duda, M. Fritsche, F. Rehrmann, F. Kirchner, Csurvey—an autonomous optical inspection head for auvs, Robotics and Autonomous Systems 67 (2015) 72–79. (Cited on page 22.)

[13] M. Wirtz, M. Hildebrandt, C. Gaudig, Design and test of a robust docking system for hovering auvs, in: 2012 Oceans, IEEE, 2012, pp. 1–6. (Cited on page 22.)

[14] A. D. Bowen, D. R. Yoerger, C. Taylor, R. McCabe, J. Howland, D. Gomez-Ibanez, J. C. Kinsey, M. Heintz, G. McDonald, D. B. Peters, et al., The nereus hybrid underwater robotic vehicle for global ocean science operations to 11,000 m depth, in: OCEANS 2008, IEEE, 2008, pp. 1–10. (Cited on page 23.)

[15] P. J. Sanz, P. Ridao, G. Oliver, G. Casalino, Y. Petillot, C. Silvestre, C. Melchiorri, A. Turetta, Trident an european project targeted to increase the autonomy levels for underwater intervention missions, in: Oceans-San Diego, 2013, IEEE, 2013, pp. 1–10. (Cited on page 23.)

[16] N. Palomeras, A. Peñalver, M. Massot-Campos, P. L. Negre, J. J. Fernández, P. Ridao, P. J. Sanz, G. Oliver-Codina, I-auv docking and panel intervention at sea, Sensors 16 (10). doi:10.3390/s16101673.
URL https://www.mdpi.com/1424-8220/16/10/1673 (Cited on page 24.)

[17] D. M. Lane, F. Maurelli, T. Larkworthy, D. Caldwell, J. Salvi, M. Fox, K. Kyriakopoulos, Pandora: Persistent autonomy through learning, adaptation, observation and re-planning, IFAC Proceedings Volumes 45 (5) (2012) 367–372. (Cited on page 24.)

[18] J. Kalwa, D. Tietjen, M. Carreiro-Silva, J. Fontes, L. Brignone, N. Gracias, P. Ridao, M. Pfingsthorn, A. Birk, T. Glotzbach, et al., The european project morph: Distributed uuv systems for multimodal, 3d underwater surveys, Marine Technology Society Journal 50 (4) (2016) 26–41. (Cited on page 24.)

[19] G. Casalino, M. Caccia, S. Caselli, C. Melchiorri, G. Antonelli, A. Caiti, G. Indiveri, G. Cannata, E. Simetti, S. Torelli, et al., Underwater intervention robotics: An outline of the italian national project maris, Marine Technology Society Journal 50 (4) (2016) 98–107. (Cited on page 24.)

[20] P. Sanz, A. Peñalver, J. Sales, J. J. Fernández, J. Pérez, D. Fomas, J. García, Multipurpose underwater manipulation for archaeological intervention, Instrumentation Viewpoint (18) (2015) 50–51. (Cited on page 25.)

[21] J. Gancet, D. Urbina, P. Letier, M. Ilzkovitz, P. Weiss, F. Gauch, B. Chemisky, G. Antonelli, G. Casalino, G. Indiveri, A. Birk, M. F. Pfingsthorn, S. Calinon, A. Turetta, C. Walen, L. Guilpain, Dexrov: Enabling effective dexterous rov operations in presence of communication latency, in: OCEANS 2015 - Genova, 2015, pp. 1–6. `doi:10.1109/OCEANS-Genova.2015.7271691`. (Cited on page 25.)

[22] E. Simetti, F. Wanderlingh, G. Casalino, G. Indiveri, G. Antonelli, Robust project: Control framework for deep sea mining exploration, in: OCEANS 2017 - Anchorage, 2017, pp. 1–5. (Cited on page 26.)

[23] E. Raugel, J. Opderbecke, F. Marie-claire, L. Brignone, V. Rigaud, Operational and scientific capabilities of ariane, ifremer's hybrid rov, 2019, pp. 1–7. `doi:10.1109/OCEANSE.2019.8867102`. (Cited on page 26.)

[24] L. L. Whitcomb, A. Bowen, D. R. Yoerger, C. R. German, J. C. Kinsey, L. A. Mayer, M. Jakuba, D. Gomez-Ibanez, C. L. Taylor, C. Machado, et al., Development of nereid-ui: A remotely operated underwater vehicle for oceanographic access under ice, in: AGU Fall Meeting Abstracts, Vol. 2012, 2012, pp. C13E–0654. (Cited on page 27.)

[25] J.-Y. Kim, B.-H. Jun, Design of six-legged walking robot, little crabster for underwater walking and operation, Advanced Robotics 28 (2) (2014) 77–89. (Cited on page 27.)

[26] O. Khatib, X. Yeh, G. Brantner, B. Soe, B. Kim, S. Ganguly, H. Stuart, S. Wang, M. Cutkosky, A. Edsinger, et al., Ocean one: A robotic avatar for oceanic discovery, IEEE Robotics & Automation Magazine 23 (4) (2016) 20–29. (Cited on page 27.)

[27] A. Chapanis, Man-machine engineering. (Cited on page 30.)

[28] D. Norman, User centered system design, New perspectives on human-computer interaction. (Cited on page 30.)

[29] R. W. Picard, Affective computing: challenges, International Journal of Human-Computer Studies 59 (1-2) (2003) 55–64. (Cited on page 30.)

[30] F. Karray, M. Alemzadeh, J. Abou Saleh, M. N. Arab, Human-computer interaction: Overview on state of the art. (Cited on page 30.)

[31] M. G. Helander, Handbook of human-computer interaction, Elsevier, 2014. (Cited on page 30.)

[32] B. A. Myers, A brief history of human-computer interaction technology, interactions 5 (2) (1998) 44–54. (Cited on page 30.)

[33] B. Shneiderman, C. Plaisant, M. S. Cohen, S. Jacobs, N. Elmqvist, N. Diakopoulos, Designing the user interface: strategies for effective human-computer interaction, Pearson, 2016. (Cited on page 30.)

[34] J. Nielsen, Usability engineering, Morgan Kaufmann, 1994. (Cited on page 30.)

[35] A. Marincic, D. Budimir, Tesla's multi-frequency wireless radio controlled vessel, in: 2008 IEEE History of Telecommunications Conference, IEEE, 2008, pp. 24–27. (Cited on page 31.)

[36] N. J. Nilsson, et al., Shakey the robot. (Cited on page 31.)

[37] T. Fong, C. Thorpe, C. Baur, Collaboration, dialogue, human-robot interaction, in: Robotics Research, Springer, 2003, pp. 255–266. (Cited on page 32.)

[38] T. B. Sheridan, Telerobotics, automation, and human supervisory control, MIT press, 1992. (Cited on page 33.)

[39] T. B. Sheridan, Human–robot interaction: status and challenges, Human factors 58 (4) (2016) 525–532. (Cited on pages 33 and 36.)

[40] M. A. Goodrich, A. C. Schultz, Human-robot interaction: a survey, Now Publishers Inc, 2008. (Cited on pages 34 and 36.)

[41] T. Sheridan, et al., Automation and human supervisory control, Chapter 1 (1992) 13–97. (Cited on page 35.)

[42] P. T. Kidd, M. Rahimi, W. Karwowski, Design of human-centered robotic systems, in: human robot interaction, London, UK: Taylor and Francis, 1992, pp. 225–241. (Cited on page 36.)

[43] P. Ridao, J. Yuh, J. Batlle, K. Sugihara, On auv control architecture, in: Intelligent Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on, Vol. 2, IEEE, 2000, pp. 855–860. (Cited on page 43.)

[44] S. K. Choi, S. Menor, J. Yuh, Distributed virtual environment collaborative simulator for underwater robots, in: Intelligent Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on, Vol. 2, IEEE, 2000, pp. 861–866. (Cited on page 43.)

[45] S. K. Choi, O. Easterday, An underwater vehicle monitoring system and its sensors, Experimental Robotics VII (2001) 551–560. (Cited on page 43.)

[46] G. De Novi, C. Melchiorri, J. García, P. Sanz, P. Ridao, G. Oliver, New approach for a reconfigurable autonomous underwater vehicle for intervention, IEEE Aerospace and Electronic Systems Magazine 25 (11) (2010) 32–36. (Cited on pages 45, 68, and 70.)

[47] N. Palomeras, J. Garcia, M. Prats, J. Fernández, P. Sanz, P. Ridao, A distributed architecture for enabling autonomous underwater intervention missions, in: 2010 IEEE International Systems Conference, IEEE, 2010, pp. 159–164. (Cited on pages 46 and 68.)

[48] N. Palomeras, P. Ridao, M. Carreras, C. Silvestre, Using petri nets to specify and execute missions for autonomous underwater vehicles, in: Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on, IEEE, 2009, pp. 4439–4444. (Cited on pages 47 and 49.)

[49] G. A. Bekey, Autonomous robots: from biological inspiration to implementation and control, MIT press, 2005. (Cited on page 47.)

[50] B. Finkemeyer, U. Thomas, F. M. Wahl, Compliant motion programming: The task frame formalism revisited, in: In Mechatronics & Robotics, 2004. (Cited on page 47.)

[51] L. Petersson, M. Egerstedt, H. Christensen, A hybrid control architecture for mobile manipulation, 1999, pp. 1285 – 1291 vol.3. doi:10.1109/IROS.1999.811657. (Cited on page 47.)

[52] B. Siciliano, L. Villani, Robot Force Control, 1st Edition, Kluwer Academic Publishers, Norwell, MA, USA, 2000. (Cited on page 47.)

[53] P. Ridao, J. Batlle, J. Amat, G. Roberts, Recent trends in control architectures for autonomous underwater vehicles, International Journal of Systems Science 30 (9) (1999) 1033–1056. (Cited on page 48.)

[54] K. P. Valavanis, D. Gracanin, M. Matijasevic, R. Kolluru, G. A. Demetriou, Control architectures for autonomous underwater vehicles, IEEE Control Systems Magazine 17 (6) (1997) 48–64. (Cited on page 48.)

[55] E. Coste-Maniere, H. H. Wang, A. Peuch, Control architectures: what's going on?, in: Proc. US/Portugal Workshop on Undersea Robotics and Intelligent Control, 1995, pp. 54–60. (Cited on page 48.)

[56] P. Ridao, E. Batlle, D. Romagós, M. Carreras, Neptune: a hil simulator for multiple uuvs, Vol. 1, 2004, pp. 524 – 531 Vol.1. doi:10.1109/OCEANS.2004.1402970. (Cited on page 48.)

[57] T. Murata, Petri nets: Properties, analysis and applications, Proceedings of the IEEE 77 (4) (1989) 541–580. (Cited on page 49.)

[58] N. Palomeras, P. Ridao, M. Carreras, C. Silvestre, Towards a mission control language for auvs, IFAC Proceedings Volumes 41 (2) (2008) 15028–15033. (Cited on page 49.)

[59] P. J. Sanz, A. Requena, J. M. Inesta, A. P. Del Pobil, Grasping the not-so-obvious: vision-based object handling for industrial applications, IEEE Robotics & Automation Magazine 12 (3) (2005) 44–52. (Cited on pages 54 and 58.)

[60] J. Pérez, J. Sales, A. Peñalver, J. J. Fernández, P. J. Sanz, J. C. García, J. V. Martí, R. Marín, D. Fornas, Robotic manipulation within the underwater mission planning context, in: Motion and Operation Planning of Robotic Systems, Springer, 2015, pp. 495–522. (Cited on page 68.)

[61] J. García, M. Prats, R. Marin, P. Sanz, User interface oriented to the specification of underwater robotic interventions, Journal of Maritime Research 8 (2) (2011) 47–62. (Cited on page 68.)

[62] M. Prats, D. Ribas, N. Palomeras, J. C. García, V. Nannen, S. Wirth, J. J. Fernández, J. P. Beltrán, R. Campos, P. Ridao, et al., Reconfigurable auv for intervention missions: a case study on underwater object recovery, Intelligent Service Robotics 5 (1) (2012) 19–31. (Cited on page 68.)

[63] J. J. Fernández, M. Prats, P. J. Sanz, J. C. García, R. Marin, M. Robinson, D. Ribas, P. Ridao, Grasping for the seabed: Developing a new underwater robot arm for shallow-water intervention, IEEE Robotics & Automation Magazine 20 (4) (2013) 121–130. (Cited on page 68.)

[64] J. García, J. Fernández, P. Sanz, R. Marín, Increasing autonomy within underwater intervention scenarios: The user interface approach, in: 2010 IEEE international systems conference, IEEE, 2010, pp. 71–75. (Cited on page 68.)

[65] J. C. García, J. J. Fernández, R. M. P. J. Sanz, M. Prats, Towards specification, planning and sensor-based control of autonomous underwater intervention, IFAC Proceedings Volumes 44 (1) (2011) 10361–10366. (Cited on page 68.)

[66] M. Prats, J. C. Garcia, J. J. Fernandez, R. Marin, P. J. Sanz, Advances in the specification and execution of underwater autonomous manipulation tasks, in: OCEANS 2011 IEEE-Spain, IEEE, 2011, pp. 1–5. (Cited on page 68.)

[67] J. Garcia, M. Prats, P. Sanz, R. Marin, O. Belmonte, Exploring multimodal interfaces for underwater intervention systems, in: Proceedings of the IEEE ICRA 2010 Workshop on Multimodal Human-Robot Interfaces, 2010. (Cited on page 68.)

[68] J. García, A. Peñalver, M. Prats, P. Sanz, Recent progress in hri for underwater robotic intervention, in: 2013 IEEE International Conference on Robotics and Automation (ICRA 2013), Karlsruhe, Germany (May 2013), 2013. (Cited on page 68.)

[69] J. C. García, M. Prats, P. J. Sanz, R. Marín, Incrementando el nivel de autonomía en los sistemas robóticos de intervención submarina: punto de vista de la interfaz de usuario, in: XXXI Jornadas de Automática (CEA), Jaén (Spain), 2010. (Cited on page 68.)

[70] J. C. García, J. J. Fernández, M. Prats, R. Marín, P. J. Sanz, Avances en la especificación y ejecución de tareas autónomas de manipulación submarina, in: XXXII Jornadas de Automática (CEA), Sevilla (Spain), 2011. (Cited on page 68.)

[71] J. J. Fernández, J. C. García, M. Prats, P. J. Sanz, D. Ribas, P. Ridao, S. Wirth, G. Oliver, Proyecto rauvi: Historia y resultados, in: XXXIII Jornadas de Automática, Vigo, Spain, 2012. (Cited on page 68.)

[72] P. J. Sanz, J. J. Fernández, J. Pérez, A. Peñalver, J. C. García, D. Fornas, J. Sales, J. A. Bernabé, R. Marín, Grasper: Un proyecto dirigido a incrementar la autonomi?a de la manipulacio?n submarina, in: XXXIV Jornadas de Automa?tica, CEA, CEA, Terrassa, Spain, 2013. (Cited on page 68.)

[73] M. Prats Sánchez, P. J. Sanz Valero, Á. Pascual del Pobil Ferré, A framework for compliant physical interaction: the grasp meets the task. (Cited on page 70.)

[74] S. Ekvall, D. Kragic, Grasp recognition for programming by demonstration, in: Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on, IEEE, 2005, pp. 748–753. (Cited on page 70.)

[75] OculusRift, Oculusrift, `https://www.oculus.com/rift/`. Accessed: 2017-06-15 (2017). (Cited on page 73.)

[76] CMLabs, Vortexsim, `https://www.cm-labs.com/simulation-products/custom-simulator-design/`. Accessed: 2017-06-15 (2017). (Cited on page 76.)

[77] M. Simulation, Rovsim, `http://marinesimulation.com`. Accessed: 2017-06-15 (2017). (Cited on page 76.)

[78] G. simulations, Vrov, `http://www.grisim.com/products/vrov-virtual-remotely-operated-vehicle/`. Accessed: 2017-06-15 (2017). (Cited on page 76.)

[79] F. G. R. Ltd, Deepworks, `http://www.fugro.com/our-expertise/our-services/subsea/live-operations-monitoring-visualisation`. Accessed: 2017-06-15 (2017). (Cited on page 76.)

[80] O. Matsebe, C. Kumile, N. Tlale, A review of virtual simulators for autonomous underwater vehicles (auvs), IFAC Proceedings Volumes 41 (1) (2008) 31–37. (Cited on page 76.)

[81] M. Prats, J. Pérez, J. J. Fernández, P. J. Sanz, An open source tool for simulation and supervision of underwater intervention missions, in: Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on, IEEE, 2012, pp. 2577–2582. (Cited on page 76.)

[82] OpenSceneGraph, Openscenegraph, `http://www.openscenegraph.org`. Accessed: 2017-06-15 (2017). (Cited on page 77.)

[83] osgOcean, osgocean, `https://github.com/kbale/osgocean`. Accessed: 2017-06-15 (2017). (Cited on page 77.)

[84] F. Alcala, A. Alcocer, F. Alves, K. Bale, J. Bateman, A. Caiti, M. Casenove, J.-C. Chambelland, G. Chapman, O. Curé, et al., Venus (virtual exploration of underwater sites) two years of interdisciplinary collaboration. (Cited on page 77.)

[85] OpenTracker, An open software framework for virtual reality input, `https://www.ims.tuwien.ac.at/projects/opentracker2`. Accessed: 2017-06-15 (2017). (Cited on page 78.)

[86] V.-R. P. Network, Virtual-reality peripheral network, `https://github.com/vrpn/vrpn/wiki`. Accessed: 2017-06-15 (2017). (Cited on page 78.)

[87] M. Slater, B. Lotto, M. M. Arnold, M. V. Sanchez-Vives, How we experience immersive virtual environments: the concept of presence and its measurement, Anuario de psicología/The UB Journal of psychology 40 (2) (2009) 193–210. (Cited on page 80.)

[88] U. Röijezon, M. Djupsjöbacka, M. Björklund, C. Häger-Ross, H. Grip, D. G. Liebermann, Kinematics of fast cervical rotations in persons with chronic neck pain: a cross-sectional and reliability study, BMC musculoskeletal disorders 11 (1) (2010) 222. (Cited on page 81.)

[89] B. Patrão, Biblioteca para desenvolvimento de aplicações de realidade aumentada em marcadores binários [augmented reality library using binary markers](university of coimbra master thesis), DEEC–FCT, University of Coimbra. (Cited on page 81.)

[90] J. R. Lewis, Ibm computer usability satisfaction questionnaires: psychometric evaluation and instructions for use, International Journal of Human-Computer Interaction 7 (1) (1995) 57–78. (Cited on page 84.)

[91] M. Slater, M. Usoh, A. Steed, Depth of presence in virtual environments, Presence: Teleoperators & Virtual Environments 3 (2) (1994) 130–144. (Cited on page 84.)

[92] M. Usoh, E. Catena, S. Arman, M. Slater, Using presence questionnaires in reality, Presence: Teleoperators and Virtual Environments 9 (5) (2000) 497–503. (Cited on page 84.)

[93] J. Guna, G. Jakus, M. Pogačnik, S. Tomažič, J. Sodnik, An analysis of the precision and reliability of the leap motion sensor and its suitability for static and dynamic tracking, Sensors 14 (2) (2014) 3702–3720. (Cited on page 90.)

[94] StructureSensor, Structuresensor, `https://structure.io/`. Accessed: 2017-06-15 (2017). (Cited on page 91.)

[95] PCL, Point cloud library, `http://pointclouds.org/`. Accessed: 2017-06-15 (2017). (Cited on page 91.)

[96] J. C. García Sánchez, B. Patrao, L. Almeida, J. Pérez Soler, P. Menezes, J. Dias, P. J. Sanz Valero, Design and evaluation of a natural interface for remote operation of underwater roter. (Cited on page 91.)

[97] J. Perez, J. Sales, A. Penalver, D. Fornas, J. J. Fernandez, J. C. Garcia, P. J. Sanz, R. Marin, M. Prats, Exploring 3-d reconstruction techniques: A benchmarking tool for underwater robotics, IEEE Robotics & Automation Magazine 22 (3) (2015) 85–95. (Cited on page 91.)

[98] J. C. García, B. Patrao, J. Perez, J. Seabra, P. Menezes, J. Dias, P. J. Sanz, Towards an immersive and natural gesture controlled interface for intervention underwater robots, in: OCEANS 2015-Genova, IEEE, 2015, pp. 1–5. (Cited on page 91.)

[99] A. Peñalver, J. J. Fernández, J. Pérez, J. Sales, J. C. García, D. Fornas, R. Marín, P. J. Sanz, Avances recientes en manipulación subacuática en el subproyecto grasper, in: XXXV Jornadas de Automática, Valencia, Spain, 2014. (Cited on page 91.)

[100] H. Bruyninckx, Open robot control software: the orocos project, in: Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on, Vol. 3, IEEE, 2001, pp. 2523–2528. (Cited on page 95.)

[101] Rock, Robot construction kit, `http://rock-robotics.org`. Accessed: 2017-06-15 (2017). (Cited on page 95.)

[102] S. Joyeux, J. Albiez, Robot development: from components to systems, in: 6th National Conference on Control Architectures of Robots, 2011, pp. 15–p. (Cited on page 95.)

[103] G. Meinecke, J. Albiez, S. Joyeux, V. Ratmeyer, J. Renken, Orocos based control software of the new developed marum hybrid-rov for under-ice applications, in: Oceans-San Diego, 2013, IEEE, 2013, pp. 1–6. (Cited on page 95.)

[104] J. García, J. Pérez, P. Menezes, P. J. Sanz, A control architecture for hybrid underwater intervention systems, in: 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), IEEE, 2016, pp. 001147–001152. (Cited on page 106.)

[105] J. C. García, J. Pérez, J. J. Fernández, P. Menezes, P. J. Sanz, Mejoras en la arquitectura de control e interfaz de usuario para el control de un hrov, in: XXXVII Jornadas de Automática, Madrid (Spain), 2016. (Cited on page 106.)

[106] P. J. Sanz, J. Pérez, J. Sales, J. J. Fernández, A. Peñalver, D. Fornas, J. C. García, R. Marín, Mermanip: Avances recientes en la manipulación autónoma cooperativa submarina, in: XXXVI Jornadas de Automática, Bilbao (Spain), 2015. (Cited on page 106.)

[107] M. K. Nagy, V. G. Foris, Ergonomically designed computer workstation adjustable to various sitting and standing positions, uS Patent 5,174,223 (Dec. 29 1992). (Cited on page 113.)