Preprints of the 2012 20th Mediterranean Conference on Control & Automation (MED), Barcelona, Spain, July 3-6, 2012

Multipurpose Autonomous Underwater Intervention: A Systems Integration Perspective

M. Prats, J.C. García, S. Wirth, D. Ribas, P.J. Sanz, P. Ridao, N. Gracias and G. Oliver

***Abstract*— Nowadays, autonomous intervention is getting more attention in the underwater robotics community. Few research projects on this matter are currently under development. In this context, and after a first successful experience in the RAUVI Spanish project (2009-2011), the authors are currently involved in the TRIDENT project (2010-2013), funded by the European Commission. To succeed in autonomous intervention, an AUV endowed with a manipulator and with a high degree of autonomy is essential. The complexity of the required robotic system is very high and the system integration process becomes critical. This paper presents the problems being solved in TRIDENT, from a systems integration perspective. As a case study, some results, achieved during the last experiments carried out in the Roses harbor (Girona) in October 2011 will be presented, to demonstrate the capabilities exhibited by the AUV for Intervention under development. The experiments were focused on the problem of autonomously searching and recovering a black-box mock-up that was previously thrown to an unknown position. This paper presents the hardware and software integration aspects that were necessary in order to address such a challenging problem.**

I. INTRODUCTION

The interest in underwater intervention is significantly increasing in the last years and the autonomous factor is becoming each time more important. Currently, most underwater intervention missions are carried out with work class ROVs (i.e. Remote Operated Vehicles), which represent an expensive solution, mainly due to all the infrastructure needed for deploying such heavy and tele-operated systems. Furthermore, the need of an umbilical cable introduces workspace limits and control issues. One of the goals of our RAUVI and TRIDENT projects is to increase the levels of autonomy in underwater survey and intervention tasks by means of a light-weight Intervention Autonomous Underwater Vehicle (I-AUV).

Our approach is to adopt a two-step strategy. In the first step, the I-AUV is deployed for surveying a given region of interest on the seabed and build a image photo-mosaic. The target of interest is then identified on the mosaic and the manipulation action is specified by means of a suitable user interface. After that, during the second step, the I-AUV navigates close to the identified target, localizes it and executes the intervention mission in an autonomous manner.

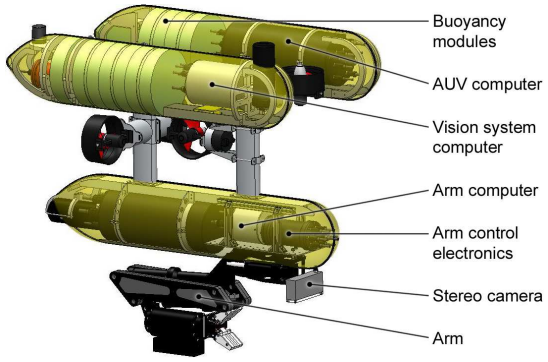

Fig. 1. Distribution of the integrated systems.

This paper shows a proof-of-concept of our approach, carried out at Roses Harbour (Girona, Spain). An autonomous object recovery experiment is described from a hardware and software integration perspective. In section II we introduce the mechatronics integration, with a brief description of the AUV, the vision system and the light-weight robotic manipulator used in the experiments. Section III covers the software integration, explaining the main algorithms used for AUV control, target identification, tracking and manipulation. Finally, section IV explains with more details the experiments and section V concludes the paper.

It is worth mentioning that few autonomous intervention missions have been carried out in non-controlled environments. To the best of the authors' knowledge, only the SAUVIM project [1] has exhibited autonomous intervention capabilities similar to those described here, although with a much more heavy vehicle.

II. MECHATRONICS INTEGRATION

The complete mechatronics system is composed of three elements: the Girona 500 autonomous underwater vehicle (AUV), from the Girona University, a light-weight robotic manipulator from the University of Jaume-I and a stereo vision system from the Balearic Islands University. All the systems and their integration are described in the following sections.

A. Girona 500 AUV

The Girona 500 is a compact-size AUV designed for a maximum operating depth of 500 m [2]. The vehicle

M. Prats, J.C. García and P.J. Sanz are with University of Jaume-I, Spain. S. Wirth and G. Oliver are with University of Balearic Islands. D. Ribas, P. Ridao and N. Gracias are with University of Girona. Contact: mprats@uji.es

This research was partly supported by the European Commission Seventh Framework Programme FP7/2007–2013 under grant agreement 248497 (TRIDENT Project), by Spanish Ministry of Research and Innovation DPI2011–27977–C03 (TRITON Project) and DPI2008–06548–C03 (RAUVI Project), by Foundation Caixa Castelló-Bancaixa PI.1B2011–17, by Generalitat Valenciana ACOMP/2012/252, and the Ramon y Cajal program to N. Gracias.

is built around an aluminum frame which supports three torpedo-shaped hulls as well as other elements like the thrusters. The overall dimensions of the vehicle are 1 m in height, 1 m in width, 1.5 m in length and a weight (on its basic configuration) of about 140 Kg. The two upper hulls, which contain the flotation foam and the electronics housing, are positively buoyant, while the lower one contains the heavier elements such as the batteries and the payload. This particular arrangement of the components provides the vehicle with passive stability in pitch and roll, making it suitable for tasks requiring a stable platform such as video surveying or intervention. The most remarkable characteristic of the Girona 500 is its capacity to reconfigure for different tasks. On its basic configuration, the vehicle is equipped with typical navigation sensors (DVL – Doppler Velocity Log, AHRS – Attitude and Heading Reference System, pressure gauge and USBL – Ultra Short Base Line) and a basic survey equipment (profiler sonar, side scan sonar, video camera and sound velocity sensor). In addition to these sensors, almost half the volume of the lower hull is reserved for mission-specific payload, which makes possible to modify its sensing and actuation capabilities as required. A similar philosophy has been applied to the propulsion system which can be set to operate with a different number of thrusters, ranging from 3 to 8, to actuate the necessary degrees of freedom and provide, if required, some degree of redundancy.

The Girona 500 is equipped with a 2.2kWh battery system. The operational range of the vehicle will depend largely on the configuration for each particular mission (number of thrusters, type of payload, arm, lights, etc.) and also on the velocity. On general terms (standard survey sensors, 5 thrusters and a velocity of about 0.5 m/s) it will last about 6-7hr.

B. Light-weight ARM 5 E

The Light-Weight ARM 5 E is an underwater robotic manipulator powered by 24V brushless DC motors. It is composed of four rotation joints, and can reach distances up to one meter. An actuated robot gripper allows small objects to be grasped, and its T-shaped grooves also permit the handling of special tools. The arm is made of aluminum alloy partially covered with foam material in order to guarantee suitable buoyancy. The total weight in the air is about 29 kg (including the electronics cylinder), whereas in fresh water it decreases to 12 kg approximately.

The arm low-level control and power electronics are placed in a housing cylinder, and the control PC is placed on another cylinder. Therefore, the manipulation system is composed of the arm, and two containers that include the arm electronics and the control PC. The electronics housing uses a PIC microcontroller in order to (i) send/receive RS232 data packages to/from the control PC, and (ii) communicate with each motor microcontroller through a CAN bus. The RS232 communication protocol includes fixed-length motor command and sensor messages. Motor command messages are sent from the PC to the arm, and can be either a control demand in terms of position, speed or voltage, or a PID setting message. When the arm microcontroller receives a motor command message, it performs the corresponding control action and sends back to the PC a sensor message including position, speed, current and temperature of each motor as measured by the internal sensors.

C. Vision System

The vision system consists of two high resolution stereo cameras and a PC for image processing. The cameras and the PC have separated watertight housings. This layout allows for a much easier reconfiguration of the cameras within the vehicle, in contrast to having all components inside a single transparent cylinder. The cameras have different focal lengths, which enables selecting different fields of view. In the current configuration for seafloor mapping and object detection we use the camera with the wider field of view mounted looking downwards (see Figure 1). With this setting, at two meters distance from the ground one pixel corresponds to an area of $1.5 \times 1.5 mm^2$.

D. The integrated I-AUV

Payload systems, like those that can be integrated in the Girona 500 AUV, generally rely on the vehicles computer and power system to operate. This makes possible a more efficient management of the energy and reduces the vehicle weight. The approach taken for the I-AUV was different. Each payload sub-system was designed to operate independently, with their own computers and power adapters. Although not optimal, this strategy made possible for each research group to work separately in their systems software. Moreover, during the integration process, this also simplified debugging since it made possible to clearly isolate the origin of any malfunction. On the other hand, the main drawback of this approach was that the resulting payload is over-sized. The limitations in the space reserved for payload in the Girona 500 made necessary to install components in other parts of the vehicle. The pressure hull containing the vision systems computer, which is one of the most buoyant parts of the whole system, was conveniently re-allocated in the top part of the vehicle (see Fig. 1); while the pressure hulls for the arms computer and control electronics were placed close to the arm, as well as the stereo camera, which requires a clear view of the manipulator operation area. To make room for the vision system pressure hull, some flotation modules had to be removed. This, together with the heavy weight of the complete payload (about 46 kg), made necessary to replace all the flotation modules with other modules made with a more lightweight foam. However, this was at the cost of reducing the maximum operative depth from 500 to only 100 m. The resulting platform was very stable and required only small pieces of lead to adjust the balance and buoyancy of the vehicle.

The electrical integration of the vehicle and the payload was more straightforward (see Fig. 2). The payload area of the Girona 500 is equipped with an Ethernet connection and a direct power source from the battery system. The vehicle computer is connected via Ethernet to a switch placed inside

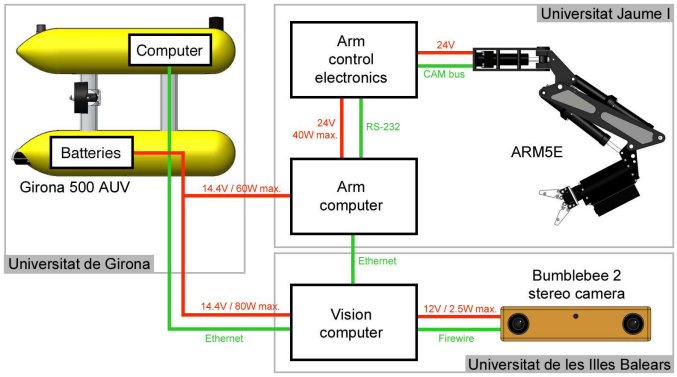

Fig. 2. Power and communication schematics for the integration of the three modules.

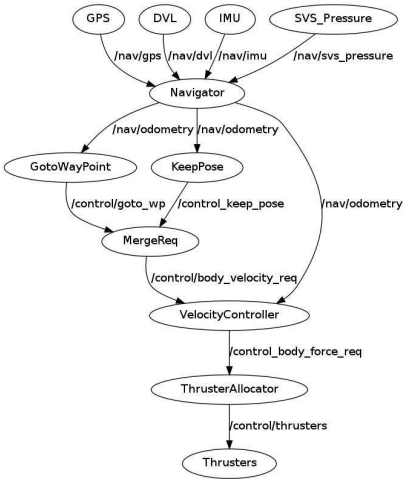

Fig. 3. The Girona 500 architecture.

the vision system pressure hull. This same switch hosted Ethernet connections with the computers in charge of the vision and the manipulation. On the other hand, the battery power was directly provided to each subsystem and regulated to the required voltages inside their own pressure hulls.

III. SOFTWARE

A. Vehicle Navigation

The Girona 500 architecture is divided into three modules (See Fig. 3): the vehicle interface module, the perception module and the guidance and control module. The vehicle interface module contains components, referred to as drivers, which interact with the hardware. It includes the navigator sensor drivers (GPS, DVL and IMU) as well as the thrusters driver. For this experiment, only one perception module was necessary, the navigator. It estimates the vehicle's position and velocity merging the data obtained from the navigation sensors by means of a Kalman filter [3]. The guidance and control module includes a set of behaviors (GotoWaypoint and KeepPose), a component to merge the behavior responses if necessary, a velocity controller and a thruster allocator module. Behaviors receive data from the vehicle's

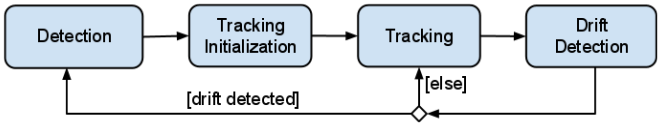

Fig. 4. Detection and tracking.

interface or perception modules, remaining independent of the physical sensors and actuators used. The MergeReq component is used to combine all the responses generated by the behaviors into a single one. In general, a behavior response is a velocity request specified in the vehicle's body frame. This body velocity request may effect only some degrees of freedom (DOFs, e.g. a bottom following behavior controls only the vehicle's heave leaving the other DOFs unactuated). Thus, a component for merging the responses of simultaneously enabled behaviors is implemented. The velocity controller, takes the merged velocity request and turns it into a force request. A simple Proportional Integral Derivative (PID) for each DOF is used for this task. The low level controller takes as input the force request provided by the velocity controller and computes a set-point for each thruster to achieve the desired force. A Thruster Allocation Matrix (TAM) plus a function for each thruster relating its input set-point to the force it generates are used.

B. Vision System

Besides the camera drivers, the vision system software contains modules for distortion correction, rectification and scene interpretation. To achieve autonomous manipulation, a detailed model of the environment is needed - especially, precise information about the 3D pose of the object of interest. The algorithm for object pose estimation is divided into two parts which work hand in hand: detection and tracking (see Fig. 4). The detection process determines the 2D position, rotation and scale of the object in incoming images while the tracking process calculates the full 3D pose. Tracking is initialized by the output of the detection and reinitialized when it tends to drift.

In the following we describe the detection and tracking algorithms in detail.

C. Detection Algorithm

Our previously described scenario limits the complexity that our algorithm has to deal with: As the camera is mounted looking downwards and the object is assumed to lie on a flat seafloor, appearance changes can be mostly limited to scale, in-plane rotation, and 2D translation.

To be able to visually detect an object, the system has to learn its appearance beforehand and save it as the object model. Our model for detection consists of a colour histogram that contains the probabilities for pixels to be part of the object and of a polygon that describes the shape of the connected component of these pixels.

The colour histogram is constructed from a training image containing the object and a polygon marking it. Colour

histograms are created for both the object and the whole image. Dividing the histogram of object colours by the histogram of the whole image bin by bin results in a new histogram. This new histogram contains in each bin the likelihood of a pixel that falls into this bin of being an object pixel. Colours that occur both in the object and in the background or more often in the background than in the object obtain a low probability whereas colours that occur mostly in the object only obtain a probability near one. Back projecting this histogram on an image results in a probability image in which each pixel receives a grey-level according to the probability of being an object pixel. By applying a threshold to this probability image followed by a connected component analysis we gain a set of polygons, each being an object candidate. Each of these polygons is compared to the object shape that was trained using the same procedure on the training image. The trained object shape has to be positioned, rotated and scaled in a way that maximizes a measure of overlap with the detected shape. When the overlap measure exceeds a certain threshold, a detection is reported. The detection result is the set of parameters (2D pose and scale) that resulted in an optimal fit during shape matching. The whole procedure works in realtime at a camera frame rate of 10 fps.

D. Tracking Algorithm

The tracking algorithm is template-based. More concretely, the Efficient Second-Order Minimization method (ESM) is used [4]. The same image that is used to train the detector (selected offline, after the seafloor survey) defines the template to track. The template-based tracker is initialized from the output of the detection, by warping the template to track according to the actual position, orientation and scale reported by the detector. Once the template has been matched in the image, the tracking algorithm consist of a minimization problem that continuously tries to decrease the error between the template gradient and that present at its estimated position in the image. The result is a local warp of the template around its estimated pose that is finally translated into an homography between the template image and the actual one.

Switching between detection and tracking allows to accelerate the vision-based control loop, since tracking is normally much faster than detection because of its local search. However, it is quite common to loose the tracked object, especially under fast relative motion of the camera and object, but also due to occlusions, strong illumination changes, etc. It is therefore desirable to combine detection and tracking so that the tracker can be re-initialized in case of failure. In our approach, tracking failures are detected by a large inter-frame displacement of the tracked template, or if the template normal being tracked in the image is not normal to the camera plane.

When a tracking error is detected, the tracker is re-initialized and waits again for the detection module to report the pose of the template in the image. For manipulation purposes, the 3D pose of the object relative to the vehicle needs to be known. As the camera is calibrated and the size of the object can be known from the geo-referenced mosaic built during the survey phase, its 3D pose can easily be calculated, using, for example, the Dementhon algorithm [5] with the four corners of the tracked template.

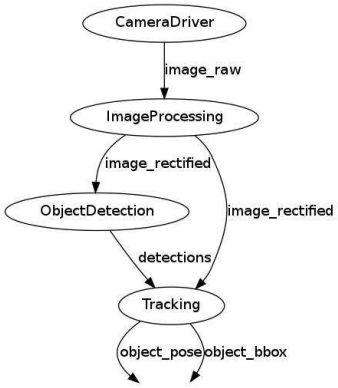

Fig. 5. ROS nodes involved in the vision system.

For the implementation, camera control, image processing, object detection and tracking are distributed into several ROS-nodes [6]. See Fig. 5 for a full overview of the vision related modules.

E. Arm control

The low-level arm control software has been implemented in C/C++ and uses ROS for inter-module communications. At the lowest level, a communications module is in charge of receiving and sending RS232 data packages to the manipulator. Other modules interact with this one in order to send control signals and process sensor data. More specifically, the following modules are involved (see also Fig. 6):

- ARM5Coms. This module constantly sends a motor velocity control signal (in RPM) to each joint of the robot via a RS232 package. Then, it waits until the manipulator controller sends back another data package with sensor information, including the joint ticks sensed by Hall Effect sensors.
- ARM5Init. This package performs the automatic initialization of the arm. It sequentially moves each joint towards the limit by sending a constant motor velocity to the ARM5Coms module, and calling the Set Zero service of the ARM5Control module, which is in charge of setting an absolute zero for each joint.
- ARM5Control. This module is in charge of providing a comprehensive view of the arm state to the rest of the system. First, it implements a virtual joint position sensor that transforms Hall Effect sensors ticks into absolute joint angles. In addition, it receives joint velocity references and transforms then into motor RPM that are sent to the communications module.
- ARM5Kinematics. Finally, the kinematics module computes forward and inverse kinematics of the arm. On the one hand, it reads the virtual position sensors and computes forward kinematics with them, thus providing

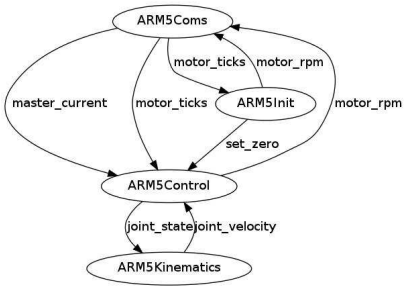

Fig. 6. ROS nodes involved for arm control.

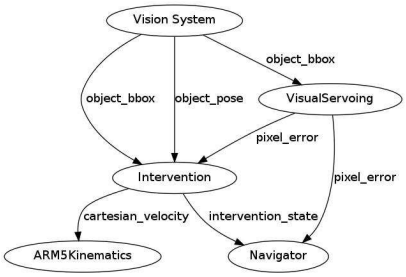

Fig. 7. Communication among the different software modules.

the end-effector Cartesian pose. On the other hand, it receives a Cartesian velocity reference, and transforms it into joint velocities via a Resolved Motion Rate Controller (RMRC).

F. Software integration

The three software subsystems (i.e. navigation, vision and manipulation) need to exchange information in order to fulfill the following tasks:

- Seafloor mapping: The images acquired by the vision system need to be combined with navigation data in order to build a geo-referenced mosaic of the environment
- Station Keeping: Keep the pose of the vehicle on a desired reference with respect to the target object. This needs a continuous communication between the tracker and the vehicle navigation software.
- Visual guided manipulation: The output of the vision system is also used as input for the control of the manipulator.

This inter-module communication is performed using the ROS communication mechanisms, mainly by sending messages over predefined topics. See Fig. 7 for a complete diagram of all the modules involved. More details are given in the following.

G. Seafloor mapping

While performing the seabed survey, camera images together with navigation data are synchronously grabbed and saved to disk using ROS bagging tool. When the survey is finished, the recorded data is transferred to an operator station for the offline process of seafloor mapping.

H. Visual station keeping

The output of the tracking algorithm, executed inside the vision subsystem, is used to visually keep the vehicle on top of the target. To achieve this, the tracker output is sent to the VisualServoing node, that computes the error in pixels between the center of the tracked template and its desired position in the image, together with the yaw angle error in radians. This error is published and used in a velocity control loop on the vehicle.

I. Visual control of the manipulator

The output of the vision system is also used for controlling the arm end-effector towards the target. The target pose, given in camera coordinates, is transformed into end-effector coordinates through the kinematic chain composed of the camera-arm base calibration and the arm forward kinematics. This allows to compute an error in the cartesian space between the current pose of the end-effector, and the desired one, which is transformed into a control velocity reference sent as input to the arm cartesian controller (*ARM5Kinematics* node).

IV. OBJECT RECOVERY EXPERIMENT

The hardware and software of the integrated system have been validated with an object recovery experiment, carried out at the Roses Harbour (Girona, Spain). A black box mock-up was thrown to the sea, and the vehicle was programmed to conduct a visual survey. With the imagery gathered, a seafloor orthophotomosaic was setup and the black box position was identified. The appearance of the black box in the mosaic was used to initialize the detector and tracker appearance models. Next, the vehicle navigated autonomously to an area close to the black box. Because it is assumed that in real conditions the I-AUV will not land on the target object but in its neighbourhood, it was programmed to perform a sweeping search path. When the black box appeared in the image, it was detected by the vision system, and the vehicle switched to station keeping mode. While keeping station with vision feedback, the manipulator perfomed the hooking action. Finally, the vehicle returned to the surface. Some details of each part are provided in the following sections.

A. Survey

For the survey operation, a grid trajectory was programmed to cover a search area of 8 by 10 m at a constant altitude of 1.3 m approximately. This altitude guarantees, for the field of view of the installed camera, a sufficient overlap between images and a full coverage of the area. The trajectory was executed with the dead-reckoning estimate from the navigation system which merges information from the onboard DVL, AHRS and pressure sensor. Both the navigation data and the images were synchronously grabbed and saved to disk for post-processing.

After the survey mission, and taking advantage of the synchronization of the visual and navigation data, a simple preliminary photomosaic was built by projecting the images to the dead-reckoning estimate of the vehicle position and

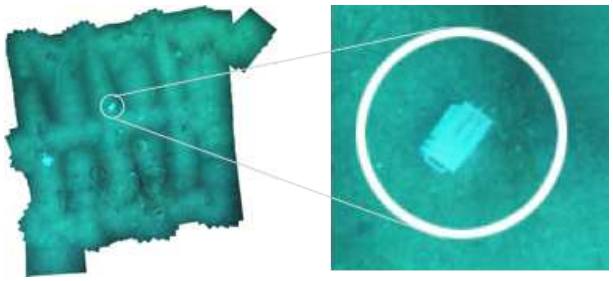

Fig. 8. Mosaic of the surveyed area. The white circle marks the position of the black box.

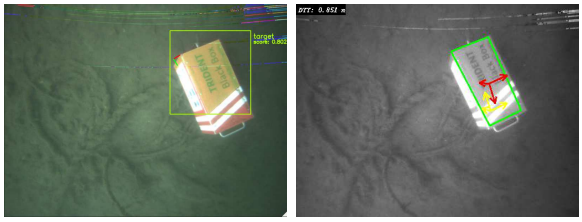

Fig. 9. (Left) Output of the detector. (Right) Tracking already initialized. The red frame indicates the object pose computed from the template corners.

altitude over the floor. This allows to rapidly explore the visual map in search for the object to recover. Alternatively, a higher quality complete mosaic was also generated following the process detailed in [7] (see Fig. 8). This improved mosaic was obtained by first performing image registration over image pairs where the navigation information predicted there would be overlap, followed by a joint optimization of the image locations taking into account the registration and navigation information. As a final step, a seamless composite image is created by suitably blending the registered images [8].

B. Searching for the target

With the vehicle in the neighborhood of the black box (after its identification in the mosaic), a local search was performed until the black box appeared in the image and was detected by the vision system. At this point, the vehicle switched to station keeping mode and was guided by the visual error between the actual object pose and its desired position in the image. Fig. 9 shows two frames of the real experiment, at the moment the black box is detected in the image.

C. Intervention

While keeping station, the arm end-effector was controlled towards the handle, using the visual feedback as well. The manipulator was controlled in the cartesian space, from the error between the actual pose of the end-effector, and the estimated target pose on the handle of the black box. Fig. 10 shows two photos taken by divers of the I-AUV autonomously recovering the black box, with the vehicle doing visual station keeping. Success or failure in the intervention was detected by monitoring the variation in Z of the object pose estimation. If the pose of the target is the same after the intervention, it is considered as a failure, and the action is repeated (a maximum of three times). If the pose changes, it is considered as a success, the intervention task is reported as concluded, and the vehicle automatically returns to the surface.

Fig. 10. Autonomous recovery of the black box.

V. CONCLUSIONS

This paper has described hardware and software integration aspects of an I-AUV designed for autonomous intervention missions. On the part of hardware, three different systems (an AUV, and underwater manipulator and a vision system) have been mechanically and electrically integrated into a highly stable I-AUV. Regarding software, all the modules for vehicle navigation, manipulation and vision processing have been seamlessly combined into a ROS network.

Experiments performed in a harbour validate the system at both levels. From the point of view of hardware, the I-AUV was able to navigate with a high level of position accuracy and stability in the water. All the systems worked fine during three days of experiments. From the software point of view, a combination of detection and tracking led to a robust and fast 3D object pose estimation. The accuracy of the vision system showed to be high enough to guide the autonomous manipulation and for station keeping. The success in hooking an object autonomously demonstrates the accuracy of the vehicle-arm system and its autonomous control.

REFERENCES

[1] G. Marani, S. K. Choi, and J. Yuh, "Underwater autonomous manipulation for intervention missions AUVs", *in Ocean Engineering*, vol. 36, 2009, pp. 15–23.

[2] D. Ribas, P. Ridao, and J. Neira, "Underwater SLAM for Structured Environments Using an Imaging Sonar", *in Number 65 in Springer Tracts in Advanced Robotics*. Springer Verlag, Heidelberg (Germany), 2010.

[3] D. Ribas, N. Palomeras, P. Ridao, M. Carreras, and A. Mallios, "Girona 500 AUV, from survey to intervention", *in IEEE/ASME Transactions on Mechatronics*, vol. 17(1), 2012, pp. 46–53.

[4] S. Benhimane and E. Malis, "Real-time image-based tracking of planes using efficient second-order minimization", *in IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2004, pp. 943–948.

[5] D. Dementhon and L. Davis, "Model-based object pose in 25 lines of code", *in International Journal of Computer Vision*, vol. 15, no. 1/2, 1995, pp. 123–141.

[6] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler and A. Y. Ng, "ROS: an open-source Robot Operating System", *in ICRA Workshop on Open Source Software*, 2009.

[7] J. Ferrer, A. Elibol, O. Delaunoy, N. Gracias, and R. García, "Large-area photo-mosaics using global alignment and navigation data", *in Oceans MTS/IEEE*, Vancouver (Canada), 2007, pp. 1–9.

[8] N. Gracias, A. Gleason, S. Negahdaripour, and M. Mahoor, "Fast image blending using watersheds and graph cuts", *in Image and Vision Computing*, vol. 27(5), 2009, pp. 597–607.