A Vision-based Control and Interaction Framework for a Legged Underwater Robot

Junaed Sattar and Gregory Dudek
Center for Intelligent Machines, McGill University,
3480 University Street, Montréal, H3A 2A7
{junaed,dudek}@cim.mcgill.ca

Abstract

We present a vision-based control and interaction framework for mobile robots, and describe its implementation in a legged amphibious robot. The control scheme enables the robot to navigate, follow targets of interest, and interact with human operators. The visual framework presented in this paper enables deployment of the vehicle in underwater environments along with a human scuba diver as the operator, without requiring any external tethered control. We present the current implementation of this framework in our particular family of underwater robots, with a focus on the underlying software and hardware infrastructure. We look at the practical issues pertaining to system implementation as it applies to our framework, from choice of operating systems to communication bus design. While our system has been effectively used in both open-ocean and closed-water environments, we perform some quantitative measurements with an effort to analyze the responsiveness and robustness of the complete architecture.

1 Introduction

A human-robot interaction mechanism is an essential component for large-scale adaptation of robots in everyday life. For robots to act as assistants to humans, from household applications to industrial deployments, an uncomplicated, easy to use and intuitive interaction scheme is of utmost importance. In this particular work, we look at a vision-based interaction and control scheme, and describe the implementation of this framework in the Aqua family of underwater robots [5]. We choose vision as an interaction modality since it creates opportunities for having many of the sought-after qualities we mention above, as well as being unobtrusive, energy-efficient and widely deployable in many different domains. In our particular robot, the underwater domain renders the traditional radio and wireless communication methods useless, since the transmission energy is absorbed rapidly by the water medium, in particular by sea water. Tactile sensing is possible, but that requires extremely close proximity to the human operator and disturbs the dynamics of the vehicle. Vision-based interaction avoids all such issues, and with the aid of fiducials makes it possible to robustly and accurately detect symbols (and accordingly instructions) coming from the human operator. Our communication scheme has minimal deployment overhead, which, in case of scuba divers (Fig. 1), is particularly important since the diver would be under substantive cognitive load already; for example maintaining the life-supporting dive gear. The vision-based control and navigation mechanisms enable the robot to follow targets using a variety of approaches – for example a scuba diver using frequency-domain tracking.

Figure 1. The Aqua robot operating tetherlessly on visual stimuli: a scuba diver starting the robot by showing a tag.

For robust communication between a robot and its human operator, interactions are necessary at different temporal scales. That is, communication must exist in a broad spectrum of invocation rates, ranging from frequentlyinvoked to rarely-invoked interactions. We have classified a suite of visual sensing algorithms based on invocation rates, and the system implementation is designed to adhere to this taxonomy when executing these algorithms on demand. All variants of the Aqua robot have an internal architecture suitable for handling the computational demands of vision computing, and also for supporting the high data bandwidth almost always associated with such algorithms. In this paper, we describe the implementation details of these architectures, including algorithm implementations, operating platform, bus design and vision hardware. We also briefly look at the control architecture of the Aqua robots, and the RoboDevel software stack [17] whose primarily responsibility is the stable (from a control theoretical point of view) locomotion of the platform.

The rest of the paper is organized as follows: Section 2 looks at existing research in the field of vision-based human-robot interaction (hereafter referred to as HRI), and underwater vision-guided autonomous robots. The core of our vision-based HRI framework along with some algorithms of that framework is described in Sec. 3. In Sec. 4, we introduce the Aqua family of robots, and describe in the following two subsections (4.2 & 4.3) details about the hardware and software architecture,respectively. Section 5 demonstrates some quantitative performance analysis from the vision sensing perspective, and we conclude with some discussions and future goals for our robot architecture in Sec. 6.

2 Related Work

In this section, we discuss past work on vision-based HRI, and also look at other vision-operated robots created with autonomous operations in mind.

For visually interacting with a human operator, a robot must have abilities to accept instructions, and also needs to have capabilities to identify and follow the operator when required to do so. These abilities are incorporated in our framework using algorithms on visual servoing, spatio-temporal and “steerable” [8] filters, fiducial markers and gesture recognition. The different components of the proposed visual framework have seen extensive research in Computer Vision, Mobile Robotics and Augmented Reality literature.

Visual servoing is a well-studied area of robotics [9], one which applies the theories of active vision into practices in real-world applications. A proportional-integral-derivative (PID) controller [12] is commonly found in visual servo systems to control motion of the robot or the manipulator arms.

In computer vision, a large amount of work has been done on tracking algorithms that track features ranging from shape and motion to color and grayscale intensities [10, 22, 3, 25]. Our previous work looked at using visual communications, and specifically visual servo-control with respect to a human operator, to control an underwater robot [20]. In that work, while the robot follows a diver to navigate, the diver can only modulate the activities of the robot by making gestures that are interpreted by a human operator on the surface. Visual communication has also been used by several authors to allow communication between robots on land, or between robots and intelligent modules of the sea floor, for example in the work of Vasilescu and Rus [23].

Niyogi and Adelson have utilized spatio-temporal patterns in tracking human beings on land [15]. Recent advancements in the field of Biometrics also have shown promise in identifying humans from gait characteristics [14]. It appears that different people have characteristic gaits and it may be possible to identify a person using the coordinated relationship between their head, hands, shoulder, knees, and feet.

Gesture-based robot control is an extensively explored topic in HRI. This includes explicit as well as implicit communication frameworks between human operators and robotics systems. Several authors have considered specialized gestural behaviors [11] or strokes on a touch screen to control basic robot navigation. Skubic *et al.* have examined the combination of several types of human interface components, with special emphasis on speech, to express spatial relationships and spatial navigation tasks [21].

Given the many application areas of vision-based robot control and navigation, visual sensing have been successfully used in one form or the other in many robots, although the number of robots operating underwater with vision is not many. This can be attributed to the inherent complexities of operating an autonomous vehicle underwater, coupled with the distortion of optics in the water medium. In spite of these difficulties, several robots, both research and industrial, have been shown to successfully use visual stimuli in underwater environments. As an example, a vision-based interaction mechanism has been used for human-robot and robot-robot interaction for the Twin Burger underwater robot [1]. Localization underwater has also been shown to work well, for both structured [2] and unstructured [6] environments. Visual servo for underwater robots have been also implemented for tasks such as station keeping [13].

3 Vision-based Interaction Behaviors

From a behavioral standpoint, we envision a suite of vision-based algorithms working in concert, to establish a robust mechanism for a robot to interact with its human operator. In our particular case, this framework consists of a number of algorithms for visual tracking, biological motion tracking in the frequency domain, and symbolic gesture-based visual languages, along with lower-level algorithms for robot manipulator control. We look at a hierarchical breakdown of vision algorithms in a three-layer interaction framework. The hierarchy arises from two aspects – the frequency at which interaction is required, and the computational cost incurred to perform the visual task. Thus, our interaction framework classifies vision algorithms from high-to-low interaction frequencies, or conversely, from low-to-high computational cost. The three different categories arise naturally from the need to acquire information for coherent interaction. Capabilities such as visual tracking are semantically low-level, but requires frequent updates for successful operation. Visually commanding the robot is not necessary at such a high frequency, and thus calls for a lower frequency interaction. Object recognition, contextual scene analysis are approaches that are computationally expensive, but are performed even at a coarser time scale. Our current work has addressed techniques belonging to the first two levels of this hierarchy. In future work, we aim to investigate approaches that fit naturally in the lowest frequency level of interaction. A detailed view of of the interaction scheme can be found in Fig. 2.

Our focus in this paper is on the integration of the visual algorithms that make up the control-interaction framework for semi-autonomous behavior, although we discuss the control scheme briefly. The different components that currently make up the visual interaction mechanism for the Aqua robots are briefly explained below.

Visual Servoing: The visual servoing subsystem enables the robot to visually track and follow a target underwater. We primarily use color properties of the object as the feature to track, using color segmentation and mean-shift tracking to track in real-time. We perform *image-based visual servoing*; hence no robot pose estimation is performed. The target coordinates are calculated in image-space, and the set-point is chosen as the image center. That is, the goal of the servoing system is to modify the pose of the robot in a way to keep the target object at the center of the camera image. The error function, ϵ is defined for both the X and the Y axes, as:

$$\begin{aligned} X_\epsilon &= |X_{tracked} - X_{setpoint}|, && (1) \\ Y_\epsilon &= |Y_{tracked} - Y_{setpoint}| && (2) \end{aligned}$$

The tracker outputs (X, Y) coordinate pairs, which are fed into a Proportional-Integral-Derivative (PID) controller. The PID controller gains (for the pitch and yaw axes) are calculated empirically. The tracker outputs and the servo set points are given as inputs to the PID controller, and it outputs leg positions to the robot controller to enforce the desired pose change to minimize the $epsilon$ error functions.

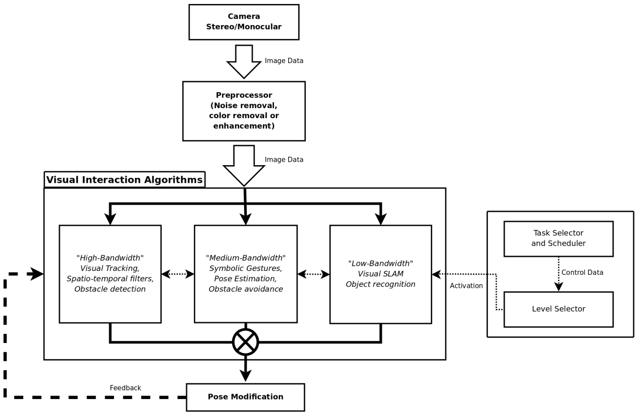


Figure 2. A schematic diagram of our visual interaction framework for mobile robots.

Visual Human-Robot Communication: While the servoing system enables the robot to track and follow targets, even human scuba divers, a human-operator still needs to remain in the control loop to modify the behavior of the robot, for example a diver accompanying the robot. To address this issue, we created a symbolic visual control scheme for a human operator to control the behavior of the robot, without the need of any human operators in the loop. Our visual communication scheme, called *RoboChat*, decodes fiducial tags (bar-code like markers) as input signals to the robot. These signals can be atomic commands (*i.e.* “turn left”), complex sequence of commands, or even semantically higher-level constructs (*i.e.* “go to the reef 50 feet from the shore and perform a transect”). RoboChat does not depend on the actual fiducial scheme being used to signal the robot, although currently we are using the ARTag [7] marker scheme for our operations. In future work, we aim to adopt the Fourier Tag [18] family of fiducials for RoboChat. The RoboChat vocabulary is quite expressive, and thus requires only a small number of tags to embed all the necessary commands required by a diver to operate the robot. This ability directly enables the robot to operate tetherlessly, controlled solely by the tags and the RoboChat scheme. We discuss the tetherless system architecture in Sec. 4.3.4.

Biological Motion Tracking: The presence of multiple targets with similar features often confuses visual trackers. For a robot to keep track of a human operator, it is imperative that the visual tracking algorithms are able to remove ambiguity by selecting features that are unique to biological entities. We choose to use motion as a unique signature in this framework. Biological motion is inherently periodic in the low frequencies, as has been found in previous research discussed in Sec. 2. We focus on extracting periodic gait

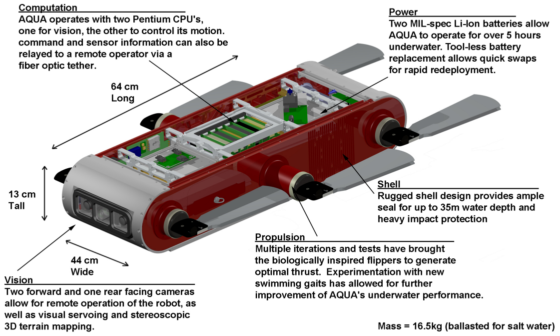


Figure 3. A cutaway annotated diagram of the Aqua 1.5 generation robot, showing important hardware components.

information from human operators, (scuba divers, for example) by analyzing a spatial signal in the XYT domain. That is, a region in the image space is analyzed across successive frames for periodicity, which according to our previous assumption is not only a potential candidate location for a human being in the image, but also can be used to uniquely identify which person we are looking at. The person in question has to travel directly away from the camera for this scheme to work, however. The intensity signal thus extracted is subjected to a Fourier transform [16], and low frequency responses are extracted. Image regions that exhibit high responses in the low frequencies are taken as potential locations. We have validated this approach on video footage of scuba divers swimming underwater [19], by exploiting the undulating motion of the diver's flippers.

4 The Aqua Robots

The Aqua family of robots are hexapodal amphibious robots capable of both underwater and ground locomotion, as well as surface swimming. The robots use six flippers for propulsion, which also act as hydroplanes for depth and direction control under water. They are power autonomous, with two on-board Lithium-ion batteries providing power for up to six hours. The robots have two computers on-board, one for control and the other for visual processing. For sensing, the robots are equipped with three cameras (two in front, one in the back), an inertial measurement unit (IMU), depth sensors, and motor current and thermal sensors. The robots have a fiber optic tether connection which acts as a conduit for video and data for remotely controlled operations. Figure 3 shows a cutaway diagram of the Aqua 1.5 generation robot.

4.1 Computing

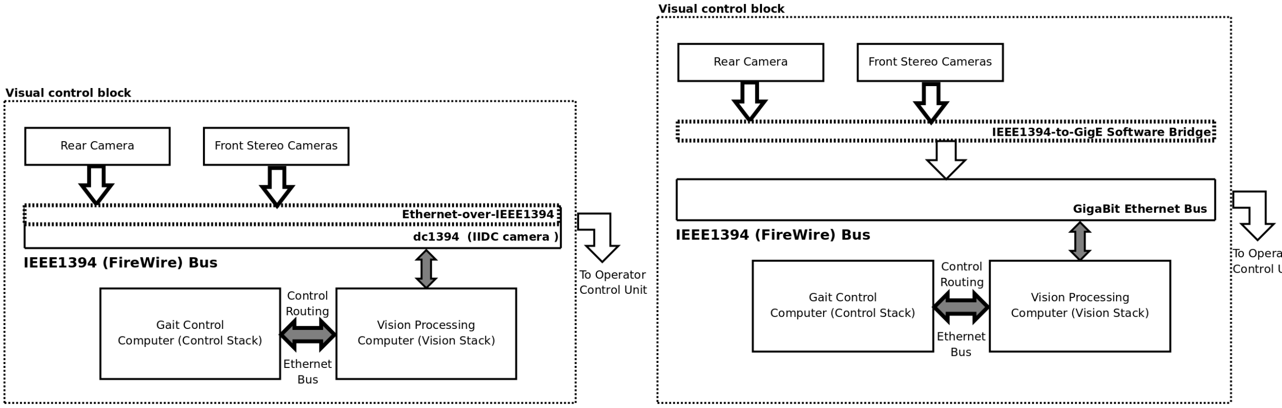
All Aqua robots are equipped with two computers – one for performing computations related to vision sensing (hereafter referred to as the *vision stack*) and the other to generate motor commands to drive the robot within hard real-time constraints (hereafter referred to as the *control stack*). The vision and control stacks conform to the PC104/Plus form factor, and are thus "stackable" to accommodate connections to other peripheral cards, by sharing the ISA and PCI buses with other computers and peripherals on the same stack. The control stack uses a Pentium-class CPU having a clock speed in the order of 300 MHz, and runs the QNX real-time operating system to ensure hard real-time ($\approx$1000Hz) operations for robot control (we will discuss more details about the software system in the following sections). Since neither the control OS and the RoboDevel library has a high memory demand, we use 256MB of RAM on the control stack. On the other hand, the vision stack does all of the vision processing on board, and is thus equipped with a scalable-frequency Pentium-M class processor, with a gigabyte of RAM. The vision stack can operate at a maximum clock rate of 1.6GHz, but during idle times is scaled down by the OS to 600MHz. This features keeps the operating temperature low on the robot, and more importantly, prolongs battery life. A stripped-down, highly-optimized version of the Linux operating system runs on the vision stack. This OS, which we call Vizix, runs the code that implements the vision algorithms described in Sec. 3. In all generations of Aqua robots, the control stack and the vision stack communicate via an Ethernet bus. For storage, both stacks use CompactFlash cards of various capacities, with the control stack using more robust versions of the CF medium (to protect against magnetic fields and high-temperatures arising from the proximity to motors and motor driver circuitry.)

4.2 Vision Hardware

There are three generations of the Aqua robot currently in operation, with very similar mechanical and locomotive capabilities, but with different internal architectures for vision sensing. We discuss these three generations briefly in the following three subsections.

4.2.1 Aqua 1.0

The first generation of the Aqua robot has three cameras for visual sensing, two in the front and one in the back. One front and one rear camera (both analog) are used for teleoperation by a driver using a remote operator control unit (OCU), and one FireWire (IIDC-compliant) camera is used by the vision stack for visual servoing and other autonomous vision-assisted tasks. The data from this camera



(a) Aqua Version 1.5

(b) Aqua Version 2.0

Figure 4. The schematic of the hardware for two generations of the Aqua robots.

is not sent to the remote operator, as the FireWire bus is not multiplexed onto the fiber optic cable connecting the remote OCU to the robot. A pair of fiber optic media converter multiplexer-demultiplexer cards are used on the robot and on the remote OCU, respectively, for channeling the front and rear analog camera data. For remote control, a point-to-point Ethernet connection over the serial port is used through the fiber for communicating with the control stack. While this architecture is sufficient for vision based autonomous operations, a real-time feedback to the OCU is not possible, and tools such as a heads-up display (HUD) cannot be designed with this particular architecture.

4.2.2 Aqua 1.5

In the second generation of the Aqua robots, all three cameras have been upgraded to IIDC cameras, with front cameras having higher resolution than the rear (1024×768 as opposed to 640×480). Internally, the FireWire bus is used to transport image frames from the cameras to the vision stack, and also to the external Operator Control Unit (OCU). For video and control data transport over the fiber optic cable, we use a pair of FireWire-to-fiber mux-demux cards. For communications to the outside world, the Ethernet-over-FireWire protocol is used to route command and control packets over the FireWire bus. The interconnection of various components of the vision subsystem is depicted in Fig. 4(a). The robot is capable of tetherless (*i.e.* autonomous) behavior with this architecture as well, with the added benefit of having live FireWire streams available for a remote HUD application in the tethered mode. Because of the design of the FireWire bus, camera data is readily available at any end point. This also creates an opportunity for running vision processing off-board, for example to perform quantitative analysis and simultaneous data collection at a remote computer.

4.2.3 Aqua 2.0

The latest generation of the Aqua robots maintains the same FireWire cameras and internal camera bus from its immediate predecessor, but the connection to the outside is now over a Gigabit Ethernet (GbE) channel, instead of FireWire. Live video streaming from all three FireWire cameras caused a data congestion due to the 400Mbps bandwidth limit of the FireWire standard (we do not use a FireWire2 data bus, which would double the theoretical bandwidth to 800Mbps). To make the FireWire camera data available on the GbE bus, we use a software bridge (as shown in Fig. 4(b)) to put camera frames into UDP packets. Along with the increase in bandwidth for all cameras, the FireWire cameras are abstracted from user-level applications by the software bridge, and this makes it possible for multiple applications to simultaneously access the camera directly, which is otherwise not possible.

4.3 Software Architecture

To harness the utility provided by the visual sensors and the computing power on board the Aqua robots, a significant amount of infrastructure has been built to establish the proper software system for vision-based operations. Our investigation into the visual interaction techniques includes this system-building task. The software infrastructure spans areas from the choice of OS, implementation of the core vision-based interaction algorithms, support code for device drivers, robot control and robot teleoperation.

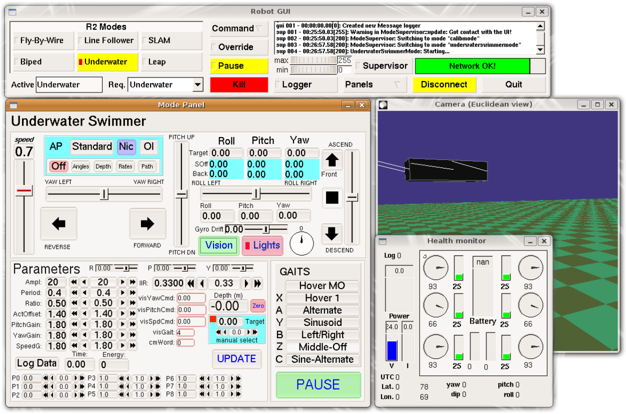
4.3.1 Operating Systems

As mentioned before, the robot control loop runs with a hard real-time constraint, which means a particular operation is considered to have executed successfully if and only if the operation succeeds within the maximum time allocated for the given task. For the Aqua robots, this time limit is 1 millisecond. This constraint enforces the use of an operating system with a real-time kernel, and we have chosen the QNX operating system for our purposes. The QNX OS is a commercial UNIX-like real-time OS, although for non-commercial, research and education purposes, the OS is available for free. QNX is a microkernel-based OS, and thus is based on the principle of running most of the OS in the form of a number of small tasks. It is this microkernel architecture that enables the OS to enforce hard-real time constraints on running programs.

For the vision stack, we did not have a similar hard-real time constraint, but the vision processing needed to be fast for the robot to maintain a responsive behavior. The availability of a rich development platform was also a driving factor for choosing an operating system for the vision stack. From these requirements, we considered using a small footprint Linux distribution, which would be fast and also be well suited for installation on CompactFlash medium without having an adverse effect on the health and lifespan of the card. We performed a trial with Damn Small Linux, but it did not meet the needs of the system (in particular size and speed requirements). In the end, we created the operating environment for the vision stack based on the Ubuntu Linux distribution. This OS, christened *Vizix*, has been designed to be lightweight, fast and robust for deployment in the flash memory storage commonly found in embedded systems. The core of Vizix is built from source, and the kernel and other essential components of the OS are highly optimized for the vision stack hardware. Also, by writing temporary data (such as log files, temporary storage etc)to RAM disks, frequent writes to the CF card is avoided, and this extends the life if the card by preserving write cycles. While Vizix primarily is used as a vision computing platform, it also provides essential connectivity services (over wired and wireless Ethernet (802.11a/b/g/n), and the Bluetooth 2.0 Enhanced Data Rate protocol) and provides support for video and image data logging both on– and off–board the robot. The primary vision application runs on Vizix, and is designed to be an ensemble of the different techniques we have integrated into the robots.

4.3.2 Robot control

Real-time control of the robots motions is performed by a software suite called RoboDevel. While a detailed discussion of RoboDevel is beyond the scope of this paper, RoboDevel is suite of software systems designed to control the RHex family of legged robots, by providing a platform to write robot-specific behaviors, and provide the GUI tools to create a remote OCU. The RoboDevel suite also includes a simulation suite to facilitate development offline. Figure 5 shows an instance of the remote OCU front end, operating in simulation mode.



```
gui 001 - 00:00:00.00[0]: Created new Message logger
sup 001 - 00:25:50.03[255]: Warning in ModeSupervisor::update: Got contact with the UI!
sup 002 - 00:25:50.03[200]: ModeSupervisor: Switching to mode "calibmode"
sup 003 - 00:26:57.58[200]: ModeSupervisor: Switching to mode "underwaterswimmermode"
sup 004 - 00:26:57.58[200]: UnderwaterSwimmerMode: Starting...
```

Active: Underwater
Req.: Underwater

Figure 5. Aqua remote operator console, working in simulation mode.

4.3.3 Implementation of vision-interaction algorithms

Due to the real-time requirements for operating the robot, the algorithm implementations need to adhere to hard real-time constraints, and that calls for efficient implementations in a real-time capable programming language. Till date, visual tracking and a large portion of the gesture-based control scheme has been implemented on the robot in C++, with visual processing code based on the VXL [1] vision libraries. The implementation can run embedded on-board the vision stack, with no external operator assistance or monitoring. In such a "tetherless" operating mode, the robot can operate autonomously, and can interact with a diver with the aid of the RoboChat gesture system. While tethered, the vision code can be operated on-board, but also can be run remotely, off-board at a remote computer. In this mode, for the Aqua 1.5 and Aqua 2.0 generations, it is possible to monitor real-time behavioral responses in the visual interaction mode, and also modify robot behavior through a graphical user interface. This application, working offline, also serves as the development platform for prototyping new algorithms for the vision-interaction framework.

4.3.4 Vision-guided autonomous control

With the aid of the RoboChat scheme, together with the visual tracking and servoing mechanism, the robot has the ability to operate without a tethered remote controller. The vision stack runs a client program that implements a subset of the RoboChat language, and also the visual tracking and servoing algorithms. The robot controller code is a different

---

[1] Vision "Something" Libraries, http://vxl.sourceforge.net

executable that runs on the control stack, and at power-up, both these programs come on-line. The vision client immediately goes into tag detection mode and waits for the human operator to show it tags. Once it detects a valid tag (or a set of tags, correctness of which is enforced by the RoboChat language grammar), the vision client communicates with the control client over the network using the UDP protocol and sends robot behavior commands, and reads back robot responses. Once put into swimming mode, the vision client has the ability to control five degrees of motion of the robot, and also engage the visual servoing system to track and follow an object underwater in full autonomous mode.

5 Performance Measures

While the performance metric for a system as complicated as the Aqua robots is not easy to quantify, we have performed simple latency tests and bandwidth loads on both the GigaBit Ethernet and FireWire buses. At 1024×768 resolution, the FireWire cameras transmit Bayer encoded frames at a maximum rate of 15 frames per second. With two cameras streaming at that rate, each camera pumps 90 Mbps of image data onto the FireWire bus. The rear camera at 640×480 resolution operates at 30FPS, and thus consumes 70Mbps of bandwidth for image data. Three camera, when transmitting simultaneously, thus consume 250Mbps bandwidth of the theoretical 400Mbps. While this rate of consumption for image data is below the maximum theoretical bandwidth limit for the FireWire bus, we have experienced significant amount of flickering and corrupt frames when operating all three cameras together. Also, the Coriander IIDC camera control tool shows 100 *per cent* FireWire bus bandwidth saturation while using all cameras simultaneously. While we have not been able to determine the exact cause of this effect, some possible causes are the presence of control information required for IIDC cameras, simultaneous existence of Ethernet-over-FireWire, muxing-demuxing overhead between the fiber and the FireWire bus, and overheads associated with remote OCU display.

We also measure the latencies associated with the Gigabit Ethernet architecture designed for Aqua 2.0, specifically to address the bandwidth issues above. In doing this, we implement the software bridge between the FireWire and the GbE buses using the GNU Real-time Transport Protocol (RTP) libraries. With two cameras transmitting simultaneously, the frames are received (in the form of UDP packets) at a remote computer on the same GbE local network. We observed minimum (in the order of magnitude of a fraction of milliseconds) display latencies. At the remote computer, reception frame rate measured was approximately 14.48 FPS, averaged over continuous transmission from both cameras for one hour.

We have also evaluated our system qualitatively in open-water trials off the coast of Barbados, where the robot has been controlled tetherlessly without remote operator assistance and only with visual cues carried by divers tending the robot. We also test our visual language scheme on human participants, in a laboratory environments [4, 24], since performing such quantitative tests in underwater environments are not only difficult, but also hazardous for the scuba divers as well.

6 Conclusion

In this paper, we present a vision-based framework for human-robot interaction for mobile robots in arbitrary environments, and present a working implementation of the said framework on-board the Aqua amphibious legged robots operating in the underwater domain. Applied mobile robotics requires a solid theoretical foundation in algorithms and techniques, and a robust, concrete platform for implementing such an algorithmic architecture to exhibit true intelligent behavior. With the Aqua robots, we believe we have made significant progress towards achieving this goal. As we pursue further research to push the current state-of-the-art in vision-based HRI, the practical need to implement these algorithms requires a constant evolution of the robot platform, both on hardware and software. The three generations of the Aqua robots clearly demonstrates this evolution. As of this writing, a newer generation of the Aqua robot is already on the drawing board, with new sensors like a global position system receiver and deeper operating capabilities, to name a few enhancements. We have attempted to keep the focus on the vision subsystem in this paper, but it is also true that an effective combination of vision along with other sensors will enable the robots to exhibit true autonomous behavior.

References

[1] B. Balasuriya, T. Fujii, and T. Ura. A vision-based interactive system for underwater robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS1995)*, volume 2, pages 561–566, Los Alamitos, CA, USA, August 1995. IEEE Compututer Society Press.

[2] M. Carreras, P. Ridao, R. Garcia, and T. Nicosevici. Vision-based localization of an underwater robot in a structured environment. In *IEEE International Conference on Robotics and Automation(ICRA2003)*, volume 1, pages 971–976, Piscataway, NJ, USA, September 2003. IEEE. Copyright 2004, IEE.

[3] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 25(5):564–575, 2003.

[4] G. Dudek, J. Sattar, and A. Xu. A visual language for robot control and programming: A human-interface study. In *Proceedings of the International Conference on Robotics and Automation ICRA*, Rome, Italy, April 2007.

[5] G. D. et al. A visually guided swimming robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Edmonton, Alberta, Canada, August 2005.

[6] R. Eustice, H. Singh, J. Leonard, M. Walter, and R. Ballard. Visually navigating the RMS Titanic with SLAM information filters. In *Proceedings of Robotics: Science and Systems (RSS)*, Cambridge, USA, June 2005.

[7] M. Fiala. Artag, a fiducial marker system using digital techniques. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2*, pages 590–596, Washington, DC, USA, 2005. IEEE Computer Society.

[8] W. T. Freeman and E. H. Adelson. The design and use of steerable filters. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 13(9):891–906, 1991.

[9] S. A. Hutchinson, G. D. Hager, and P. I. Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670, October 1996.

[10] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 1(29):5–28, 1998.

[11] D. Kortenkamp, E. Huber, and P. Bonasso. Recognizing and interpreting gestures on a mobile robot. In *13th National Conference on Artifical Intelligence*, 1996.

[12] J. R. Leigh. *Control Theory*. Institution of Electrical Engineers, July 2004.

[13] J.-F. Lots, D. M. Lane, E. Trucco, and F. Chaumette. A 2-D visual servoing for underwater vehicle station keeping. In *IEEE International Conference on Robotics and Automation*, pages 2767–2772, Seoul, South Korea, May 2001.

[14] M. S. Nixon, T. N. Tan, and R. Chellappa. *Human Identification Based on Gait*. The Kluwer International Series on Biometrics. Springer-Verlag New York, Inc. Secaucus, NJ, USA, 2005.

[15] S. A. Niyogi and E. H. Adelson. Analyzing and recognizing walking figures in xyt. In *In Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 469–474, 1994.

[16] A. V. Oppenheim, A. S. Willsky, and S. H. Nawab. *Signals & systems (2nd ed.)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.

[17] U. Saranli and E. Kavins. Object oriented state machines. *Embedded Systems Programming*, May 2002.

[18] J. Sattar, E. Bourque, P. Giguere, and G. Dudek. Fourier tags: Smoothly degradable fiducial markers for use in human-robot interaction. *Computer and Robot Vision*, 0:165–174, 2007.

[19] J. Sattar and G. Dudek. Where is your dive buddy: tracking humans underwater using spatio-temporal features. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3654–3659, San Diego, California, USA, October 2007.

[20] J. Sattar, P. Giguere, G. Dudek, and C. Prahacs. A visual servoing system for an aquatic swimming robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1483–1488, Edmonton, Alberta, Canada, August 2005.

[21] M. Skubic, D. Perzanowski, S. Blisard, A. Schultz, W. Adams, M. Bugajska, and D. Brock. Spatial language for human-robot dialogs. *IEEE Transactions on Systems, Man and Cybernetics, Part C*, 34(2):154–167, May 2004.

[22] A. Techmer. Contour-based motion estimation and object tracking for real-time applications. In *International Conference on Image Processing*, volume 3, pages 648–651, Thessaloniki, Greece, October 2001.

[23] I. Vasilescu, P. Varshavskaya, K. Kotay, and D. Rus. Autonomous Modular Optical Underwater Robot (AMOUR): Design, prototype and feasibility study. In *International Conference on Robotics and Automation*, Barcelona, Spain, 2005.

[24] A. Xu, G. Dudek, and J. Sattar. A natural gesture interface for operating robotic systems. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA*, pages 3557–3563, Pasadena, California, May 2008.

[25] Z. Zivkovic and B. Krőse. An EM-like algorithm for color-histogram-based object tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, Washington D.C., June 2004. IEEE, IEEE Press.