Tele-robotics:

Distributed Training-oriented Navigation Framework

Thomas Geerinck*, Eric Colon**, Sid Ahmed Berrabah*, Kenny Cauwerts*,Hanene Bahri*, Hichem Sahli*

* Department of Electronics and Informatics (ETRO),

Vrije Universiteit Brussel, Brussels, Belgium

** Department of Mechanics

Royal Military Academy, Brussels, Belgium

Abstract

The purpose of this article is to present a tele-robotic application, developed on an advanced demonstration platform, incorporating reflexive tele-operated control concepts elaborated on a mobile robot system. By voice command, the operator controls the behavior of the robot, being 1 out of 4 levels of robot-operator interaction, while environmental exploration is possible by means of the head mounted display in combination with an inertial tracker. Based on the distributed framework, named Controlling Robots with CORBA (CoRoBa), results of the tuned robot navigation strategy are presented. The results achieved might also be used in domains such as surveillance, remote monitoring and risky interventions.

1. Introduction

The purpose of this article is to present ongoing research and further developments on an advanced demonstration platform, incorporating reflexive tele-operated control concepts developed on a mobile robot system. The functional aspects and operability of the complete system have been extensively described in [6] as well as in [5] for a more complete description. Also a quite general overview about the concepts and state-of-the-art of remote control of mobile robots is presented there. By operability, the opportunity to develop, simulate tune and test in real-world environment mobile robot navigation algorithms is meant. The system functionalities, being tele-robotic applications in real-world environment as well as in simulation environment, fit into our distributed framework. According to requirements in present robotic software architectures, being object-oriented component based reusable software patterns, we developed such

a framework, named Controlling Robots with CORBA [7] (CoRoBa), written in C++ and based on CORBA, a standardized and popular middleware. CoRoBa relies on well-known Design Patterns [3] and provides a component based development model. Concerning the tuning and simulation of navigation algorithms, a Java based multi mobile robot simulator (MoRoS3D) exists, and integrates seamlessly in the framework as described in detail in [5].

The contribution of this paper focusses on a tele-robotic application in real-world environment, developed on this framework, allowing an operator to control the robot remotely in different ways of robot-operator interaction, by selecting one of the different levels of autonomy by voice command. The use of an inertial tracker in combination with a head mounted display improves significantly the situational awareness of the operator in semi-structured environments, creating a certain feeling of presence, at the remote site. As such an application of semi-autonomous remote environmental exploration is presented. The results achieved can also be used in domains such as surveillance, remote monitoring and risky interventions.

The remainder is organized as follows. In section 2 CoRoBa is very shortly discussed. In section 3 the whole system architecture is presented, highlighting the recently developed and implemented features. Section 4 presents some results and finally section 5 draws the conclusion.

2. CoRoBa Framework

Among all existing Middlewares, CORBA [7] has been selected because of its language and platform independence. As such a Middleware is quite complex and brings some overhead. We typically have a 20 to 30% overhead in comparison with raw socket communication [3]. With in creasing computing power and communication bandwidth, the overhead becomes less significant every day.
The Controlling Robots with CORBA (CoRoBa) framework has been developed to implement sensor networks and distributed control applications. CoRoBa provides a component based development model, relying on well-known Design Patterns [3]. Components, unit of independent deployment, are regarded as black-boxes only known by their access points or interfaces. These interfaces specify the way of interaction between components and their clients. According to the classical control paradigm, CoRoBa defines three types of components, namely Sensors, Processors and Actuators. Two communication modes, a classical synchronous call method and an event based communication scheme, are proposed in CoRoBa. Communication between the different components occurs through Events and Event Channels and is based on CORBA Notification specifications. The main advantage of event based communication is the decoupling between producers or suppliers and consumers. Consumers can receive events from different producers and producers can send different kinds of events.

3. System Overview

Figure 1 shows the different components involved in this application, whereas figure 2 represents the real existing system. The processors form the skeleton of the application. Actuator and sensor components only serve as translators. As argued in section 2 communication between the three kinds of components is realized in the CoRoBa frame-

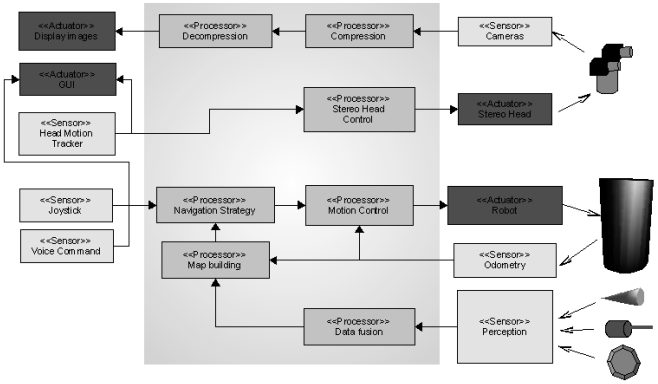

Figure 1. Global system architecture

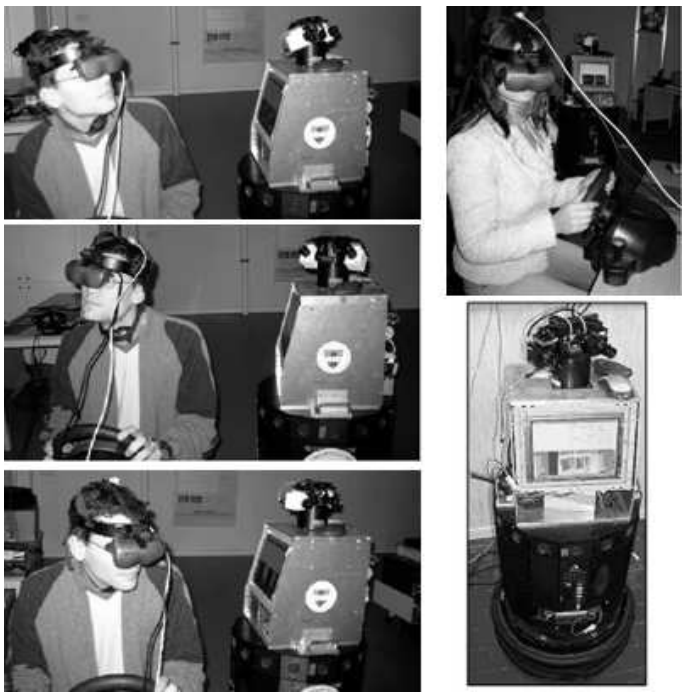

Figure 2. The Nomad200 robot with the upper PC-platform and the biclops head on top. The stereo vision system forms the robots eyes.

work. In the following the focus will be laid on the different ways of robot-operator interaction, as well as how this level of interaction might be changed. This interaction refers to the responsibilities in control of both robot and operator. The basic mode of operation for the system is traditional or direct tele-operation, including the creation of feeling of presence. In order to introduce shared or supervisory autonomy control aspects to the existing architecture of direct tele-operation, a choice must be made in how to define the responsibilities for both robot and tele-operator. We chose to provide fixed static responsibilities for human and robot. Based on the statement that the aim of robotics is to serve and help humans, our implemented system is well-suited for exploring purposes. The fixed responsibilities are defined in 4 levels of autonomy:

Tele-operation Mode The user has full, continuous control of the robot at low level. The robot takes no initiative except perhaps to stop once it recognizes that communications have failed. It does indicate the detection of obstacles in its path to the user, but will not prevent collision. This is the default autonomy level.

Safe Mode The user directs the movements of the robot, but the robot takes initiative and has the authority to protect itself. For example, it will stop before it collides with an obstacle, which it detects via multiple US and IR sensors.

Shared Control Mode The robot takes the initiative to choose its own path in response to general direction and speed input from the operator. Although the robot handles the low level navigation and obstacle avoidance, the user supplies intermittent input to guide the

robot in general directions.

Full Autonomy Mode The robot performs global path planning to select its own routes, acquiring no operator input. The goal of the robot can be specified by the operator or by the robot's vision system by introducing target recognition techniques and tracking.

Note that, the change in autonomy level is made dynamically; whenever the operator desires to change the level of autonomy the robot changes its behavior. Moreover, this change of autonomy level is realized by a speech recognition module. Subsequently this voice command recognition module is explained in more detail and the navigation strategy and motion controller will be emphasized with extra details about the implementation of the robot-operator interaction in Shared Control Mode.

3.1. Voice Command Module

A speech recognition module has been developed and implemented in C to allow the robot to respond to voice commands. This enables the operator to control the robot vocally while performing other manual tasks and it also opens the option to control the robot over a voice link using for example a cell phone or a VoIP link. Since the voice command module required only a small vocabulary and needed to be controllable by only a few operators, the easiest and most efficient way for designing and implementing the speech recognition unit was to use the traditional pattern matching approach. In this approach, the training phase of the recognizer consists in storing several example recordings of the desired commands by the different operators as templates in memory. During the recognition phase, the unknown word is then compared to every template in memory. The template that has the smallest global spectral distance to the unknown command is selected and the command that corresponds to this template is recognized as the intended command. In our current implementation, the global spectral distance used is the so-called bandpass liftered LPC cepstral distance [1] which is calculated using a Dynamic Time Warping (DTW) [4] procedure. This DTW allows accounting for possible differences in speaking rate between the vocal commands that are issued and the corresponding templates in the recognizer's memory. At present the voice control module of the robot has not yet been applied in critical situations. Therefore, recognized commands are executed immediately without requiring confirmation from the operator first. The only measure taken against the negative influence of varying background conditions such as noise, is the use of a close talking, head mounted microphone. If the number of vocabulary words and/or operators where to increase, standard noise suppression techniques such as spectral subtraction [2] could be applied before, eventually, more elaborated recognition techniques, such as Hidden Markov Modelling [9] would have to be used.

3.2. Navigation Strategy and Motion Controller

The basic building block of the present navigation strategy is a behavior, defined as representation of a specific sequence of actions aimed at attaining a given desired objective. Each behavior comprises a set of fuzzy-logic rules. The navigation strategy used in this application is a reactive navigation. While a mission is assigned or a goal lo-

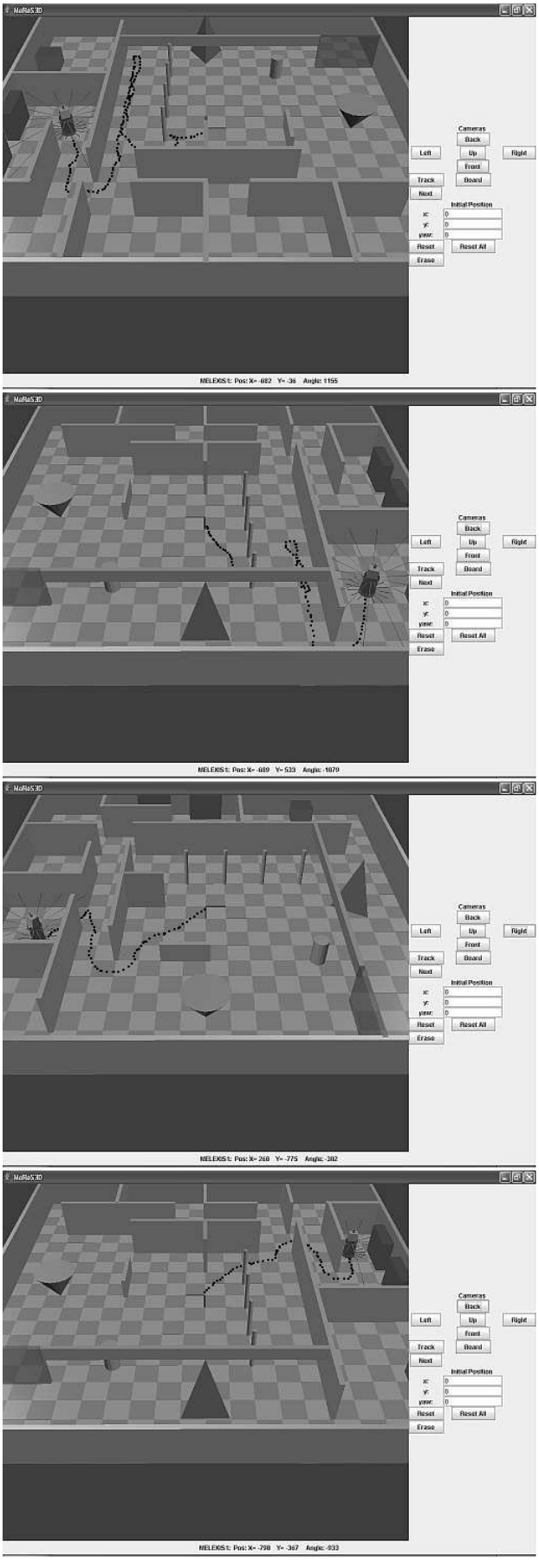

Figure 3. Results of the navigation strategy, presented in the MoRoS3D multi robot simulator

cation is known, the robot does not plan its path but rather navigates itself by reacting to its immediate environment in real-time. The result of applying iteratively a reactive navigation method is a sequence of motion commands that move the robot from the initial location towards the final location, while avoiding collisions. In our approach for robot navigation we describe the possible situations by a set of basic rules:

- If no obstacle is detected then use the ”Goal seeking behavior”.
- If it’s not possible to change direction toward the goal and there is no obstacle in front of the robot, then use the ” Go straight ahead behavior”. This behavior avoids the swaying of the robot due to it hesitation.
- If an obstacle is detected in front of the robot and it’s still possible to change direction (to turn), then use the ” Obstacle avoidance behavior”.
- If there is an obstacle in front of the robot and there is no possibility to change the direction, then use ” Make U-turn behavior”.

Figure 3 shows results acquired by simulating the robot’s navigational behavior in the simulator MoRoS3D. The figure represents 4 screen shots of the simulator in action. In the upper two figures, the robot travels a path towards a certain goal using a combination of the 3 behaviors ”Goal seeking”, ”Going straight ahead” and ”Obstacle avoidance”. In the lower two subfigures, two different viewpoints of the same robot travel path are presented, combining the 2 behaviors, ”Goal seeking” and ”Obstacle avoidance”.

MoRoS3D seamlessly integrates with our framework

CoRoBa. Having such a simulator allows focusing on intelligence and control, while disposing of other, less interesting problems. A simulator as such also increases safety when developing and testing algorithms. In MoRoS3D a robot can be placed in a 3D environment and interact with that environment in a manner similar to that of the robot in the real physical situation. Although MoRoS3D visualizes the entire surroundings of the robot, the robot software only ”sees” the information it collects through its sensors, just like with a physical robot. The MoRoS3D simulator provides simple interaction with the user and offers different virtual cameras including on-board and tracking ones. Simple distance sensors, such as Laser, US and IR, are simulated. Sensor simulation is actually a geometrical problem that comes to calculating intersections between shapes. For visual simplicity the sensors are represented by beams instead of cones, although the actual field of view of some sensors (namely ultrasonic) is cone-shaped.

According to the selected level of autonomy, the navigation strategy controller selects the proper driving behavior. For direct tele-operation, this behavior is straightforward: simply feed the acquired speed and steering commands to the robot's motion controller. In safe mode a check is performed for collision danger and if necessary an emergency stop is executed. In shared control mode as well as in autonomous mode the robot has the responsibility of the local navigation. When autonomous mode is selected, the input from the operator can be: the relative coordinates of a final goal if way and time to reach this goal or not important; a final goal with a set of desired intermediate passing points; or a completely continuous path until the final goal. If an obstacle is detected, the robot uses the obstacle avoidance behavior to bypass it and retrieve the path afterwards. In shared control mode of operation, robot and operator work together executing their exploration or data retrieval tasks in a certain environment. The goal is provided to the robot in terms of polar coordinates, a distance and an angle relative to the robot. The distance is set to a fixed value of 5m. The orientation of the goal with relation to the robot, is set on a circle with radius 5m. This orientation results form the operator's Head Mounted Display (HMD), providing the operator with visual feedback, as well as registering the operator's head movement via a built in inertial tracker. At present a total of 6 voice commands are implemented. Concerning the selection of level of autonomy, these commands include, , ”TELE”, ”SAFE”, ”SHARED” and ”AUTO”, respectively corresponding to the 4 levels mentioned earlier. In Shared Control Mode of operation, local navigation is the responsibility of the robot. However, motion of the robot is controlled by the operator by means of 2 simple voice commands: ”MOVE” and ”STOP”. In the future, this actual list of commands might be extended.

4. System Performance

The performance of control and visual feedback loops are essential when the performance of the tele-operation system is evaluated. The entire system setup exists on a a Nomad200 robotic platform, an electrical driven mobile robot build by the Nomadic Technologies Inc. company, as shown in figure Y.

Concerning the operability of the robotic platform, important delays should be kept in mind by the operator. It is clear that experience in driving the robot system, dealing with all the delays including the reaction time of the operator itself

is a primordial factor. It takes about 2 s for the robot to turn 30 degrees and an acceptable speed of 0.2 m/s is reached after 350 ms [5]. Compared to the mean human reaction time, the speed delay is satisfactory. The turn delay however, makes the operator wait for the robot's accurate reaction. The update of the commands from operator to robot occurs every 100 ms (10 Hz), to keep this system reactive.

Regarding the vision loop two processes are considered. First of all the motion-head-tracking-servo-head loop provides some delay (600ms/50dgr) mainly caused by the servo system. Internally the update rate of the head motion occurs at 180 Hz. Second, the loop of capturing, compressing and sending stereo images contains two delays. The first delay is the time it takes to compress and decompress, the second is the transmission delay. The time lost with compression and decompression is on the other hand recuperated by the much faster transmission of the images compared to the case without compression. Using a state of the art coding scheme, SQuare Partitioning (SQP) [8], a frame rate of approximately 20 Hz is obtained.

Concerning the simulator performances, typical figures for 10 robots with 16 laser distance sensors is 80% processor activity (Centrino 735) and a memory usage of 40MB, image refresh period in this configuration is 80ms.

5. Conclusions

In this paper, an advanced mobile robot platform is shortly presented. Two interesting features of the system, the voice command recognition and the implementation of the 4 levels of autonomy, in particular the shared control mode of operation, have been highlighted. The tele-robotic application is perfectly suited for exploration and surveillance purposes. The operator selects the global direction of navigation. An obstacle avoidance algorithm based on fuzzy logic has been implemented, giving the robot the responsibility of the local navigation. The results of this algorithm have been presented by means of screenshots in the MoRoS3D simulator, perfectly suited for development and testing of new navigation algorithms.

References

[1] Juang Biing-Hwang, Lawrence R. Rabiner, and Jay G. Wilpon, *On the use of bandpass liftering in speech recognition*, IEEE TRANSACTIONS ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING **ASSP-35** (1987), no. 7.

[2] S.F. Boll, *Suppression of acoustic noise in speech using spectral subtraction*, IEEE Trans. ASSP **27** (1979), 113–120.

[3] E. Colon, H. Sahli, and Y. Baudoin, *Coroba, a multi mobile robot control and simulation framework*, Special Issue on "Software Development and Integration in Robotics" of the International Journal on Advanced Robotics **3** (2006), no. 1, 73–78.

[4] J.R. Deller, J.H.L. Hansen, and J.G. Proakis, *Discrete-time processing of speech signals (2d ed.)*, IEEE Press, 2000.

[5] Thomas Geerinck, Eric Colon, Sid Ahmed Berrabah, and Kenny Cauwerts, *Tele-robot with shared autonomy: Distributed navigation development framework*, Integrated Computer-Aided Engineering (ICAE) (2006), tbp.

[6] Thomas Geerinck, Valentin Enescu, Ioan Alexandru Salomie, Sid Ahmed Berrabah, Kenny Cauwerts, and Hichem Sahli, *Tele-robots with shared autonomy: tele-presence for high level operability.*, ICINCO, 2005, pp. 243–250.

[7] Michi Henning and Steve Vinoski, *Advanced corba programming with c++*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[8] A. Munteanu, J. Cornelis, G. Van der Auwera, and P. Cristea, *Wavelet-based lossless compression scheme with progressive transmission capability*, International Journal of Imaging Systems and Technology, Special Issue on Image and Video Coding **10** (1999), no. 1, 76–85.

[9] L. Rabiner and J. Biing-Hwang, *Fundamentals of speech recognition*, Prentice-Hall, 1993.