International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:4, No:11, 2010

An AR/VR based Approach towards the Intuitive Control of Mobile Rescue Robots

Jürgen Roßmann, André Kupetz, and Roland Wischnewski

***Abstract*—**An intuitive user interface for the teleoperation of mobile rescue robots is one key feature for a successful exploration of inaccessible and no-go areas. Therefore, we have developed a novel framework to embed a flexible and modular user interface into a complete 3-D virtual reality simulation system. Our approach is based on a client-server architecture to allow for a collaborative control of the rescue robot together with multiple clients on demand. Further, it is important that the user interface is not restricted to any specific type of mobile robot. Therefore, our flexible approach allows for the operation of different robot types with a consistent concept and user interface. In laboratory tests, we have evaluated the validity and effectiveness of our approach with the help of two different robot platforms and several input devices. As a result, an untrained person can intuitively teleoperate both robots without needing a familiarization time when changing the operating robot.

***Keywords*—** teleoperation of mobile robots, augmented reality, user interface, virtual reality.

I. Introduction

THE enormous improvements in miniaturization and processing power now allow robots to cover new and wider areas of activity. In particular, applications are emerging for mobile robots in rescue operations. For example, these robots can collect information about victims or about building structures in no-go areas that are inaccessible for rescue workers. In this field of research, different approaches exist for the design and control of rescue robots.

Currently, most approaches try to develop autonomous victim-finding and map-generating robots. In contrast to this, our interests are not autonomous robots. Instead, we will focus on the teleoperating approach because we want to incorporate the experience and intuition of a human teleoperator. We are convinced that the advantages of continuous teleoperation of mobile robots in many scenarios cannot be fully exploited yet because of inadequate user interfaces. In this paper, we will present a novel framework for a flexible and intuitive user interface that is not restricted to a specific mobile robot.

Juergen Rossmann heads the Institute of Man-Machine Interaction, RWTH Aachen University, 52074 Aachen, Germany (e-mail: rossmann@mmi.rwth-aachen.de).

Andre Kupetz is with the Dortmunder Initiative zur rechnerintegrierten Fertigung, Department Robot Technology, Dortmund, 44227 Germany (phone: +49-231-9700-778; fax: +49-231-9700-771; e-mail: kupetz@rt.rif-ev.de).

Roland Wischnewski is with the Dortmunder Initiative zur rechnerintegrierten Fertigung, Department Robot Technology, Dortmund, 44227 Germany (e-mail: wischnewski@rt.rif-ev.de).

A. Motivation

A large variety of events like earthquakes, gas-explosions, or acts of terrorism may lead to enormous damage to buildings and result in inaccessible and no-go areas for the rescue workers. The chance of survival in existing cavities is pretty high, but finding casualties is an extremely dangerous and intricate task because of the structural damage to the buildings and the lack of knowledge where to search. Therefore, the active exploration of the danger zone is a critical and necessary task for a precise localization and rescue operation. In doing so, the rescue worker exposes himself to life-threatening danger, because further collapses of the building are most likely. A robust mobile robot can support the rescue worker by exploring the damaged building, searching for victims, and giving information about the statics of the building.

The operator should be able to easily navigate the robot and interpret the sensor data given by the robot. This offers specialists the possibility to evaluate the building's structure. So, the rescue operation can be planned more efficiently and with less danger for the rescue worker.

B. Scenarios

A very common scenario is the occurrence of local damages. Typical examples of such scenarios are gas explosions in residential areas or acts of terrorism like bombing. Even large-area catastrophes like earthquakes can often be broken down into several spots of small operation areas. As a result, the rescue worker will concentrate on one special building. For a fast and efficient exploration of this area, the use of mobile robots is well suited. Figures 1 and 2 show possible scenarios for a rescue robot. Figure 1 illustrates a scenario that is commonly known as "horizontal layer", in which the walls of a floor are collapsed but the ceiling is still intact. Furniture and other objects carry the ceiling so that a survival space exists. Figure 2 illustrates a scenario in which parts of floors are completely intact but the whole building is very close to collapse. Other examples are buildings that are leaning extremely to one side or buildings with severely damaged load bearing walls. All these scenarios have in common that the rescue worker has a relatively small area of exploration.

Fig. 1. Example of a horizontal layer with survival space in the collapsed first floor. Photo: Michael Markus.

Fig. 2. Example of a close to collapse building with intact floor parts. Photo: Michael Markus.

C. Perspectives

An intuitive human-robot interface to control a mobile rescue robot is fundamental for a successful rescue operation using mobile robots, as the human operator is liberated from low-level tasks. Instead of an autonomous operation mode, we want to incorporate the intuition of a human teleoperator, too. In our approach, the robot makes the low-level decisions that would overstrain the operator. For example, the robot is in charge to position its drive assemblies for an optimal traction. Furthermore, the robot calculates joint values for the different joints to ensure an optimal center of gravity to move along on an incline. However, the operator evaluates and interprets the sensor values. He can decide about points of interest to look at closer. For example, the operator may get evidence for the building's stability, or may have an idea which direction is most promising to search for victims.

Therefore, we want to offer the operator a high-level user interface designed for different interaction media. Our novel approach combines experiences in the fields of augmented reality (AR) and virtual reality (VR) we gained in [1]–[4] into a framework for a flexible, reusable, and modular user interface.

II. Related Work

The tragedy at the World Trade Center in 2001 propelled search-and-rescue robotics into its next stage and offered an unfortunate opportunity to evaluate human-robot interaction under real conditions. At this time, most rescue robots were operated with the help of remote control box like devices [5].

The RoboCup-Rescue project [6] promotes research and development in this socially significant domain at various levels. This involves multi-agent team work coordination, physical robotic agents for search and rescue operations, information infrastructures, personal digital assistants, evaluation benchmarks for rescue strategies, and robotic systems which all will be integrated into a comprehensive system in the future. But currently, this approach is not designed for realistic rough terrain, and the focus lies on an autonomous and not a teleoperated exploration [7]. Researchers, e.g. [8], put lots of efforts in solving the "Simultaneous Localization and Mapping Problem" (SLAM). In consequence, the robots need high intelligence and computation power for their algorithms. Despite the progress in the last years, however, these rescue robots are far away from a successful autonomous exploration in a real scenario. For example, [9] shows the design and development of a robot for rescue missions and outlines the weakness in its mechanical design and path-planning in the conclusion. The necessary communication between operator and robot is done via a remote desktop connection between a notebook integrated into the mobile robot and a notebook at the operation station.

To reduce the necessary computational power on the robot, we suggest submitting the sensor values to the teleoperation station, where this information is evaluated and prepared for an intuitive visualization. This concept is in contrast to [9] where all computations are carried out on the robot itself. We propose an IP based communication between robot and operating station, which splits the software into two modular parts: one part is running on the robot and the other one on the teleoperation station. Reference [10] describes a multimodal interactive control method called *telecommanding* for teleoperating wheeled mobile robots over the internet in well structured environments. Though, this concept is not designed for rough terrain, and the robot is equipped with a minimum of sensors only, the paper shows interesting aspects for teleoperating at small bandwidths.

The aim of [11]–[13] is to improve the user interface of rescue robots. With the help of a user study [11], the authors have developed guidelines for an effective design of user

interfaces [12]. Important key aspects of these guidelines are: try to lower the cognitive load with the help of fused sensor information; minimize the use of multiple windows; enhance the video stream and place information close to it because teleoperators are often highly focused on the video stream. Based on these results, [13] is improving an existing interface. Although the graphical user interface (GUI) has been simplified and the sensor information has been exemplarily laid on top of the video stream, we see chances for improvements with a more flexible and modular user interface.

Reference [14] pursues the approach to develop a robust and relatively inexpensive fire evacuation guide robot system. The robot combined with its control unit shall act as a link between firefighter and victim and provide sensor information to the teleoperator. The robot can be operated with two types of GUIs. The first one is a control box type, based on a joystick, wheeled buttons, a display, and predefined communication buttons. A second operation approach has been implemented with the help of a touch screen GUI. Due to the fact that the GUI is designed for a hand held device of small size, the possible display resolution for the GUI is limited. The GUI contains the video stream, some buttons, and some bars for sensor values. The evacuation robot uses an embedded operating system which gets its teleoperating commands via the wireless communication protocol Bluetooth [15]. Additionally, dedicated radio frequency interfaces operating at a frequency of 2.4 GHz are used for transmitting video and audio data. Unfortunately, this analog way of video transmission is not encrypted and also interferes with wireless LAN. The teleoperator has the possibility to switch between four evacuation robots he is currently controlling.

After all, the teleoperator needs to interact with the graphical user interface through a concrete input device. Reference [16] shows the approach of an automobile-like user interface with steering wheel and foot pedals to control a snake-like robot in a mechanical way. This work has been extended in [17] so that arrows occur on the video stream which guide the operator to a carbon dioxide source. One major disadvantage of this intuitively controllable robot is that there is a necessary mechanical connection between robot and operating station.

III. Framework Concept

In the following sections, we will present details of our framework concepts and the resulting software architecture.

A. System Concept

The basic concept of our approach is shown in Figure 3. On the one hand, we have the *operating station*; this is the teleoperation station with its input devices for the teleoperator and its output devices for visualization. The operating station is one part of our framework and may vary from a laptop with touchpad up to several monitors plugged in at a high end workstation, steering wheel, and foot pedals or other input devices. An integral part is our VR simulation system with its powerful render engine that has been enhanced with our teleoperation plugins.

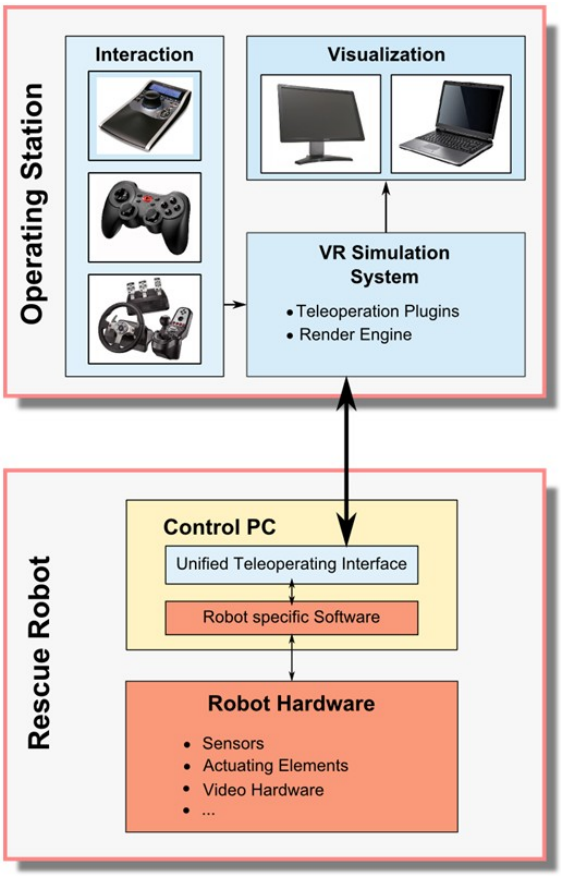

Fig. 3. System concept and communication structure of the presented framework. Our framework is divided into two parts: one part is running on the operating station (top) the other part the unified teleoperating interface on the rescue robot (bottom).

On the other hand, we have the *rescue robot* which contains the other part of our framework: the *unified teleoperating interface*. This software is not bound to any specific robot type and offers interfaces that the *robot specific software* can use. Finally, the robot specific software is interacting with the robot hardware. This includes sensors, actuating elements, video hardware, or the important and difficult task to do the low-level control of the drive assemblies for a proper processing of the movement commands as described in paragraph I.C. This needs to be separated from our framework because it is highly robot specific. Typically, the unified teleoperating interface runs on the control computer of the mobile robot where the robot specific software is running, too. According to this, our framework includes the communication between rescue

robot and operating station and offers a defined interface to the robot software. The communication between robot and operating station must be fail-safe and robust on the one side. On the other side, the unified teleoperating interface must be easy to handle in the robot specific software. So, we decided to use a string based IP communication structure. This has the advantage that we offer a transparent and well-structured interface for the robot specific software. For a fail-safe operation, we propose that each request from the operating station needs to be acknowledged with a solicited message that indicates the status of the request. So, the teleoperator always gets a feedback to his actions.

B. Client-Server Architecture

One major reason for the flexibility of our design is based on our client-server approach and the plugin structure of our simulation system. In this scenario, the operating station is the client and the rescue robot acts as a server. Figure 4 shows this in detail.

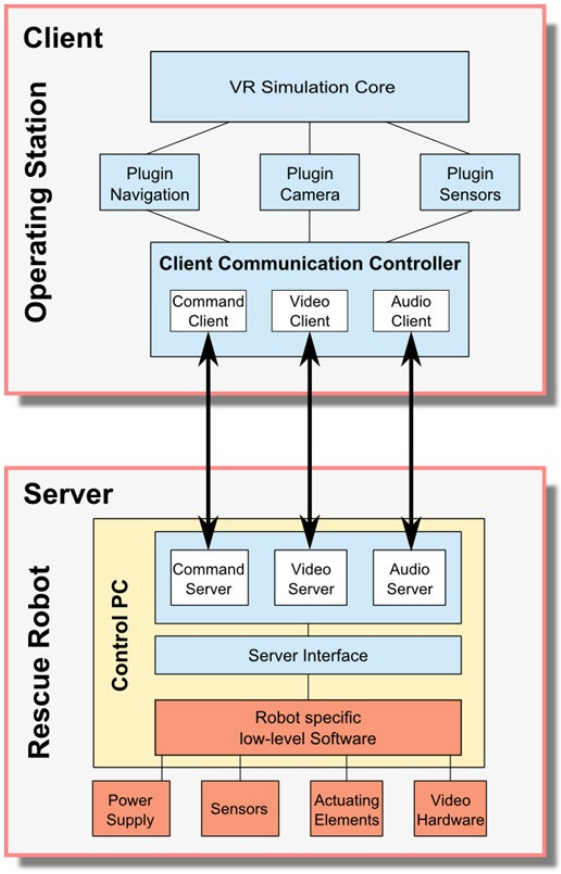

Fig. 4. Client-Server Architecture of the presented framework. The communication between operation station and rescue robot is encapsulated in our framework.

Different plugins enhance the simulation core and add the functionality for teleoperating the robot intuitively. There are three different plugins which use the client communication controller: *Plugin Navigation*, *Plugin Camera*, and *Plugin Sensors*. The *Plugin Navigation* offers the possibility to use different input devices like gamepad, steering wheel, etc. The *Plugin Camera* is responsible for processing video data and visualizes these in the simulation system. The *Plugin Sensors* processes the received sensor data and prepares them for different visualization methods. It is easy to add further plugins if required.

To communicate with the rescue robot, we have developed the client communication controller. Integral parts of the client communication controller are three task-specialized clients: the command client for processing the communication commands, the video client for receiving continuous video pictures, and the audio client for a bidirectional audio communication. These clients will establish a connection to the corresponding servers that are running on the control PC of the rescue robot. With the help of the server interface, our framework offers an application programming interface (API) which the robot specific software can use.

In detail, the usage of the framework looks like this: the client communication controller initializes the connection to the mobile robot. Then a set of defined requests is sent to the robot specific software. More precisely, the client communication controller triggers the command client to add this special request to its command queue. This request is processed and sent to the command server over TCP/IP. The command server puts this request into its incoming queue. The server interface then processes this request and emits a special signal that the specific robot software can listen to. For example, if information about a specific body joint is requested of the mobile robot, the robot software will return the number of joints, the names of the axes, the units of each axis, etc. This information is transmitted backwards up to the GUI. This procedure is repeated for a set of initialization commands defined in our API. User commands like pressing a button on a gamepad are propagated to the mobile robot in the same way.

The structure chosen allows for collaborative work of operators because several clients can connect to the rescue robot simultaneously. This offers the possibility to have one person teleoperating the robot while another specialist is watching the thermal images on a separate screen at a completely different place with its own operating station. Furthermore, it is possible that an operator switches between different rescue robots he is controlling. He just needs to select a different server. Also, this feature gives the director of operations the chance to supervise the rescue operation. Figure 5 illustrates this.

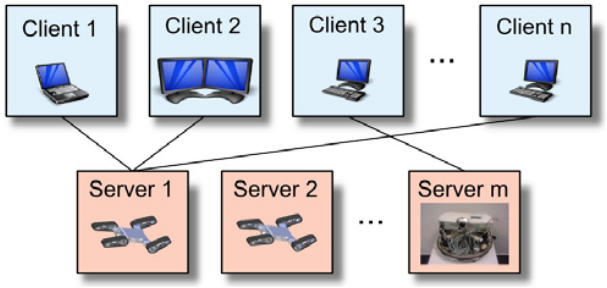

Fig. 5. Client-server scenario; several clients can connect to one server simultaneously. In this example, no client is currently connected to Server 2.

C. Extended Model-View-Controller Paradigm

One important technique for developing flexible, upgradeable, and reusable software is the model-view-controller (MVC) paradigm [18]. By dividing the application into these three parts, a high modularity is achieved and each part has its own responsibility. In general, the model consists of persistent data which can be represented in different ways. This is done with the help of different view objects that all work on the same database i.e. the model. The task of the controller is to react to interaction from the user interface, to commit these to the model, and to provide information from the model to the view.

In our case (see Figure 6), we have extended this paradigm by a central communication part because the model runs on a different PC than the view and the controller. The model is determined by the current robot hardware and the robot specific software. Depending on the implemented sensors, high-level movement commands like 'climb the step', mechanical construction, etc., an individual view object will be created. The controller reacts on user interactions and affects the view and the model with the help of the communication part.

This approach allows us to generate a specialized view object for each robot automatically. During the initialization procedure at the beginning of the teleoperation session, the mobile robot sends specific information about its hardware and commands it can interpret. This information is used to dynamically adapt the user interface for the current robot. Then, the operator still has the possibility to arrange and adapt the user interface to his needs like positioning and resizing GUI elements.

If the teleoperator misses a special feature, it is easy to fulfill the new requirements by creating a new view component for this particular need, for example by enhancing the sensor plugin with a new custom-built GUI overlay or enhancing the navigation plugin with a new input device like a data glove. As a result, this feature is available to all view objects and no changes need to be done to the robot system.

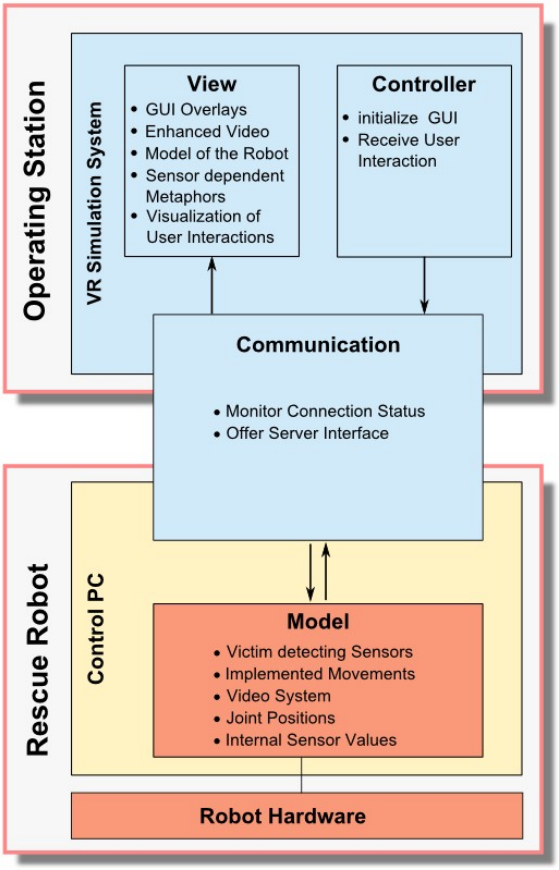

Fig. 6. Extended model-view-controller paradigm of the presented framework. The model is determined by the used mobile robot and will affect the presentation of the GUI.

IV. Implementation

In this chapter, we will present the integration of the described concepts into a simulation system together with a method for augmented information visualization.

A. Integration into the VR Simulation System

The described concepts have been implemented as plugins for the simulation system VEROSIM. This simulation system has been successfully used in several research projects e.g. [3]. The fact that we are using a complete simulation system is based on our future plans to offer the operator in a simulation of the movement commands first before sending these commands to the robot. Figure 7 shows a screenshot of the user interface we created for the simulation system.

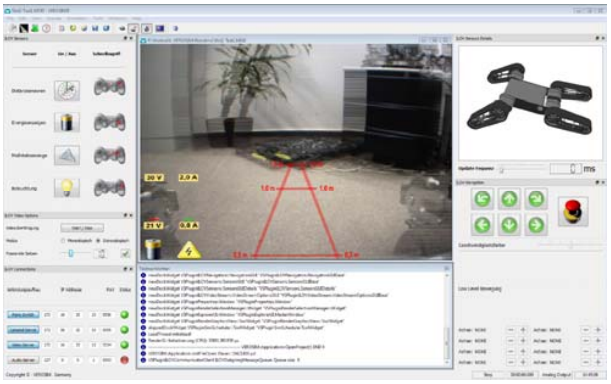

Fig. 7. Screenshot of a possible arrangement of the user interface.

This screenshot shows only one possibility how the operator can arrange the GUI. Each plugin has its own control elements that can be placed and arranged according to the operator's needs. The upper two third contains the video stream in the center and GUI elements on the left to enhance the video stream with overlays. In the top right area, a 3-D model of the current robot is rendered. The orientation of the flippers is corresponding to the received sensor values for each joint. So, the teleoperator can follow the orientation of the flippers in real time. In the lower third, we see from left to right: server connection window, message output window, and a low-level movement window for operating each joint separately.

B. Visualizing and Augmenting the Stereo Video Stream

To take full advantage of the benefits of the continuous teleoperating approach, we implemented a visualization of a real time stereo stream that is rendered in our 3-D render engine. The video client receives two synchronized images – one for each eye. These two images will be interlaced horizontally (even rows from the left image and odd rows from the right image) and rendered on a 3-D screen to give the operator a 3-D impression of the scenery. The usage of the simulation system offers the possibility to easily use overlays on top of the stream for additional information. The spectrum ranges from simple elements like a battery power meter to the CAD model of the robot. Intuitive metaphors can extend these overlays which will pop up on demand on the video stream. For example, if the battery status is getting low or the temperature of a motor reaches a critical value, an intuitive metaphor will pop up on the screen, e.g. a warning sign or a colorization. Figure 8 shows two examples of a metaphor for visualizing distance values in the style of a parking distance control of a car. This metaphor is restricted to a robot platform that supports ultra sonic sensors like the Robotino (see below). If the current robot platform does not support these sensors, the operator will not be confused by inactive or useless buttons. Figure 8 illustrates a top view of the mobile robot with distance bars. As a result, instead of reading numerical values, the operator receives meaningful information at a glance.

Fig. 8. Two examples of a metaphor for an intuitive representation of distance values. The visualization is in style of a parking distance control of a car. The center of each example shows a top view of the robot.

Currently, we are implementing a fusion of an additional video stream into the stereo stream. This additional stream, like the one of a thermal camera, may have a different resolution and a different field of view than the stereo stream. The operating station will calculate a matching for each single picture (frame) of the additional stream into the stereo stream. Therefore, this frame is matched first on the image for the left eye and then for the right eye. After this, these two scaled frames are interlaced and overlaid with transparency on top of the stereo stream.

V. Experiments on different Mobile Robot Platforms

In the previous chapter, we have presented the implementation of our concepts. Now, we are going to describe laboratory tests of our implementation. For this purpose, we had a group of five test persons who have experimented with the arrangement of the GUI elements and who have further tested the teleoperation of the robot. First, we will specify our different setups in more detail.

A. Experimental Setups

We have tested our framework with the help of three different test scenarios. Figure 9 shows the different setups for our experiments.

First, we have developed a complete testbed without any kind of robot. For this purpose, we have used a Kontron PC/104+ board with an integrated AMD 500 MHz CPU. Onto this board we have stacked a Sensoray PC/104+ frame grabber board for attaching analog camera devices and a PC/104+ FireWire board for digital cameras. First tests have been done with the analog thermal camera Thermal-Eye 3600AS and the digital stereo camera BumbleBee2. The testbed has been connected with cable LAN to a wireless LAN access point. The operating station has been connected to the access point over wireless LAN. In the bottom left area of Figure 9, the hardware setup of the described testbed is shown. Due to a missing robot, we have implemented a software emulation for sensors and robot movements. Nevertheless, since we use our unified teleoperating interface, we can apply this testbed for performance and stress tests. These indicate that the bottle neck is currently located in processing the stereo image for sending it to the

video client. So, we offer the possibility to adjust the frame rate according to the computational power of the control PC.

For a second test case, we have used a mobile robot called Robotino from FESTO [19], shown in the bottom middle area in Figure 9. This mobile robot system is based on a three drive omniwheel and is equipped with ultrasonic distance sensors, a USB camera, and an I/O card. Our unified teleoperation interface runs directly on the Robotino as a stand-alone application.

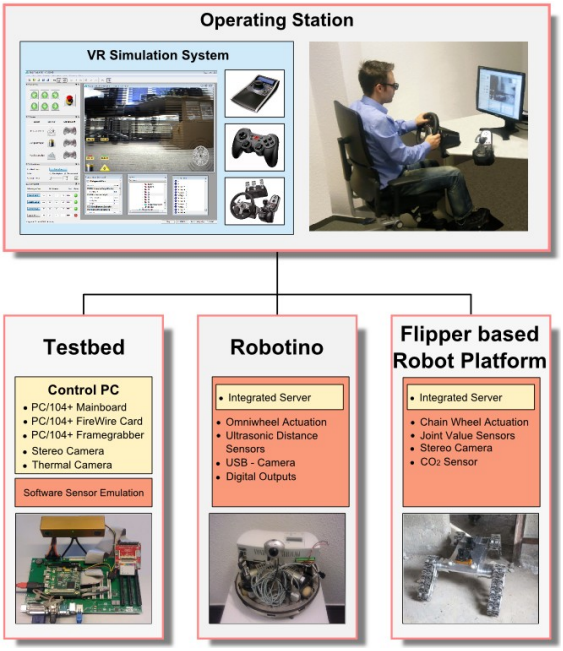

Fig. 9. Experimental setups: screenshot of the operating station with the input devices (top) and different test platforms (bottom). Bottom right photo: Lehrstuhl für Maschinenelemente und Konstruktionslehre (LMK), Ruhr-Universität Bochum, Germany.

The third test case is a flipper based robot platform in the bottom right area of Figure 9. Each flipper is chain driven and can rotate. The motor-control monitors current position, velocity, and amperage. Looking at the degrees of freedom of a flipper, the currently monitored position for this flipper is corresponding to its joint value. These values can be used for a real time visualization of a 3-D model of the robot in the simulation system. Further, this platform is equipped with a BumbleBee2 stereo camera and a BlueSens BCP-CO_2 sensor. Our unified teleoperation interface is integrated into the control PC (Kontron PC/104+ board with Windows XP). The network connection is established over wireless LAN.

B. Teleoperation Tests

The operating station has been equipped with a Zalman ZM-220W stereo monitor, a Logitech G25 steering wheel with pedals, a 3Dconnexion space mouse, and a Logitech gamepad. First, we have tested the latency between sending commands from the operating station and receiving the affirmation from the mobile robot using wireless LAN. This period takes about 50 ms in average. Our test persons have verified that the latency for changing velocities or stopping movements is unproblematic for the use of typical operation speeds of rescue robots. Regarding the limited computational power of the control PC, we set the frame rate of the stereo stream to five frames per second for each eye. This may seem not sufficient, but considering our tests with typical teleoperating speeds where no fast movements or actions have occurred, our test persons deem the frame rate adequate. After a short familiarization time with the polarization filter glasses and finding the right angle of view to the 3-D monitor, the test persons have benefitted from the depth impression very much. For example, the teleoperator does not need to keep his eyes on the distance sensors to avoid collisions.

When using the different input devices, a combination of the gamepad with foot pedals seems most promising for the flipper based robot. Teleoperating the Robotino has been very easy with the space mouse. The reason is that the shape of the Robotino resembles the billet of the space mouse. So, the degrees of freedoms correlate with those of the Robotino. This means, rotating the billet will result in a rotation of the Robotino, etc. This indicates that different mobile robots require different particular input devices. For this reason, our novel framework offers the possibility to use any number of different input devices.

Switching between the two robot platforms has been very easy for the test persons. They just need to click on the corresponding button to change the IP address for connecting to the different platforms. Doing so, no familiarization with another GUI was needed after switching between the Robotino and the flipper based robot.

VI. Conclusion

An intuitive interface that is not restricted to a specific robot offers the rescue worker the possibility to easily explore no-go areas that are inaccessible. If the rescue robot is replaced by another one for the next mission no long familiarization time will be needed anymore. Therefore, we have developed our novel framework that offers a flexible and modular user interface. This user interface is automatically adapted for each current mobile robot individually. Further, we support the rescue worker with an augmented video stream that reduces informational load via intuitive metaphors. In this constellation, we see our approach as most promising for a successful rescue operation with mobile robots.

Our next step will be to extend our framework with more metaphors and sensor fusion. Additionally, we are planning to do field tests in which rescue workers are going to teleoperate a mobile robot in rough terrain with our operating station.

REFERENCES

[1] E. Freund and J. Rossmann, “Projective Virtual Reality: Bridging the gap between Virtual Reality and Robotics,” in *IEEE Transactions on Robotics and Automation*, Vol. 15, No. 3, pages 411-422, June 1999

[2] J. Roßmann, P. Krahwinler, and C. Schlette, “Navigation of mobile robots in natural environments: Using sensor fusion in forestry," in *The 2nd International Multi-Conference on Engineering and Technological Innovation (IMETI)*, Orlando, USA, Vol. 3, pages 5-9, July 2009.

[3] J. Roßmann, M. Schluse, and C. Schlette, "The virtual forest: Robotics and simulation technology as the basis for new approaches to the biological and the technical production in the forest," in *The 13th World Multi-Conference on Systemics, Cybernetics and Informatics: WMSCI 2009*, Orlando, USA, Vol. 2, pp. 33-38, July 2009.

[4] S. Jeschke, T. Aach, L. Eckstein, J. Rossmann, and B. Rumpe, “RescueCars – Autonomous Service Vehicles in Rescue Scenarios,“ RWTH Aachen University, Germany, Tech. Rep., Feb. 2010.

[5] J. Casper and R.R. Murphy, “Human-Robot Interactions During the Robot-Assisted Urban Search and Rescue Response at the World Trade Center,” in *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 33, No. 3, pages 367-385, Jun. 2003.

[6] A. Davids, “Urban Search and Rescue Robots: from Tragedy to Technology,” in *IEEE Intelligent Systems*, Vol. 17, Issue 2, pages 81-83, Mar. 2002.

[7] http://www.robocuprescue.org/, visited Jun. 2010.

[8] M. Montemerlo, S. Thrun, D. Kooler, and B. Wegbreit, “FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem,” in *Proceedings of the AAAI National Conference on Artificial Intelligence*, Edmonton, Canada, 2002.

[9] J. Suthakorn, S.S.H. Shah, S. Jantarajit, W. Onprasert, et al., “On the Design and Development of A Rough Terrain Robot for Rescue Missions,” in *Proceedings of the IEEE International Conference on Robotics and Biomimetics*, Bangkok, Thailand, pages 1830-1835, Feb. 2009.

[10] M. Wang and J.N.K. Lui, “A Novel Teleoperation Paradigm for Human-Robot Interaction,” in *Proceedings of the 2004 IEEE Conference on Robotics, Automation and Mechatronics*, Singapore, pages 13-18, Dec. 2004.

[11] H.A. Yanco and J. Drury, ”’Where am I?’ Acquiring situation awareness using a remote robot platform,” in *IEEE International Conference on Systems, Man and Cybernetics*, Vol. 3, pages 2835-2840, Oct. 2004

[12] M. Backer, R. Casey, B. Keyes, and H.A. Yanco, “Improved Interfaces for Human-Robot Interaction in Urban Search and Rescue,” in *IEEE International Conference on Systems, Man and Cybernetics*, Vol. 3, pages 2960-2965, Oct. 2004.

[13] J.L. Durry, J. Scholz, and H.A. Yanco, “Applying CSCW and HCI techniques to human-robot interaction,” in *Proceedings CHI 2004 Workshop on Shaping Human-Robot interaction*, Vieanna, USA, pages 13-16, Apr. 2004.

[14] Y.D. Kim, Y.G. Kim, S.H Lee, J.H. Kang, et al., ”Portable Fire Evacuation Guide Robot System,” in *The 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, St. Louis, USA, pages 2789-2794, Oct. 2009.

[15] http://www.bluetooth.com/Bluetooth/Technology/, visited Jun. 2010.

[16] R. Muraui, K. Ito, and F. Matsuno, “An intuitive Human-Robot Interface for Rescue Operation of a 3D Snake Robot the Intuitive Human-Robot Interface Aims for Non-Professional Operators” in *Proceedings of the 12th IASTED International Conference Robotics and Applications*, Honolulu, USA, pages 138-143, Aug. 2006.

[17] D. Tamura, A. Fujino, and K. Ito, “A Rescue Robot designed for ease of Use: Development of Exploration System using Behavior of Bombycid,” in *Artificial life and Robotics*, Vol. 13, No. 1, pages 251-254, Dec. 2008.

[18] G. E. Krasner and S. T. Pope, “A cookbook for using the model-view controller user interface paradigm in Smalltalk-80,” in *Journal of Object-Oriented Programming*, Vol. 1, Issue 3, pages 26-49, Sept. 1988.

[19] http://www.festo-didactic.com/int-en/learning-systems/education-and-research-robots-robotino/ visited Jun. 2010.