

Design of a Teleoperated Mobile Robot Test Platform*

L. Wyard-Scott and Q.-H. M. Meng[†]

Advanced Robotics & Teleoperation (ART) Lab
Department of Electrical Engineering
University of Alberta
Edmonton, AB, T6G 2G7, CANADA
E-mail: wyard@ee.ualberta.ca, max.meng@ualberta.ca

Abstract—

This paper presents the design of a teleoperated mobile robot test bed developed at the ADVANCED ROBOTICS AND TELEOPERATION LAB, University of Alberta. A brief mechanical description is given, followed by presentation of a highly flexible workcell hardware structure utilising four distinct computational elements. The system software, written using TCL/TK and C, is presented along with an outline of the overall control algorithm. The operator's graphical interface is also described.

I. INTRODUCTION

Telerobotic and telepresence systems have been developed for use in submarine [1] [2], outer space [3] [4], and hazardous environments [5]. This paper presents the design of a teleoperated mobile robot system intended for *research*. Unlike the projects mentioned above, the development of a test bed requires a great deal of flexibility to accommodate future research directions.

This paper begins with a short description of the mobile robot (workcell) mechanical configuration and the physical configuration of the sensor array. The modularity and distributed processing nature of the workcell electronics are described and an overview of the software for the system is presented. Future directions are outlined followed by a section of conclusions in which the suitability of the system design is evaluated.

II. MECHANICAL CONFIGURATION

The workcell consists of an inexpensive, modifiable chassis which incorporates flexible positioning of the electronics modules and sensors. This section describes the chassis design and sensor placement for the mobile

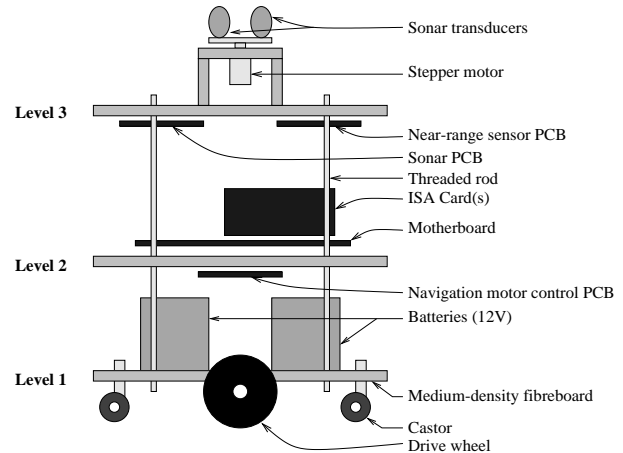


Figure 1: Side view of the mobile robot workcell.

robot system component configured for indoor applications.

A. Structure

Fig. 1 shows a side view of the mobile robot.

Two Polaroid sonar transducers are mounted at the top of the mobile robot upon a stepping motor with a step resolution of 1.8° . Rotation of the sonar array in this manner makes it possible to obtain satisfactory results using only two of the relatively expensive modules.

Two 24 V gear-head motors provide locomotion via two differentially configured drive wheels. Encoders are attached directly to the output shaft and provide the feedback necessary for closed-loop control of the mobile robot's speed.

B. Sensor Placement

Near-range proximity sensors include 26 infrared (IR) pairs and 16 bump switches mounted at multiple levels. Around the drive wheels and castors, sensors are arranged to detect obstacles in the direction of travel.

* This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada under grant OGP170446 to M. Meng.

[†] To whom all correspondence shall be addressed.

CdS photocells (photoresistors) are used to measure ambient light levels, which may be used to adjust analog IR sensor data.

III. ELECTRONIC CONFIGURATION

This section presents the workcell electronics and the distributed processor architecture.

System hardware is designed so that if the workcell motherboard becomes unstable, function will not be completely compromised. Thus, with software designed accordingly, potentially system-stopping events could result in nothing more than a loss of communication with the control station.

This robustness is achieved through the use of a four-element distributed processing architecture consisting of a motherboard and three microcontrollers. Survival behaviours are programmed directly into the microcontroller layer.

A test platform's need for flexibility puts great motivation behind electronic modularity. Each modular component described in this section can at any time be replaced, added to, upgraded, or in some instances left out, depending upon the application being addressed.

Fig. 2 is a block diagram showing the modular system electronics. Only those blocks with bold lettering are described here.

A. 68HC11 MCU PCB

MC68HC11 microcontroller units (MCUs) form the core of the MCU boards. Three of these PCBs are used in the system, each with its own 32K of RAM, and chip-select circuitry for memory-mapped peripherals. The HC11 is commonly used in robotics projects due to the wide range of circuitry it offers in addition to the CPU [6].

Attesting to the advantages of a modular design, the MCU boards have been used as stand-alone devices in other projects at the ART LAB.

B. Parallel Port Expansion PCB

The Parallel Port Expansion PCB is a digital I/O card intended specifically for interfacing to 8-bit peripherals. Several additional digital outputs serve to control the mode of operation of the three microcontroller boards:

- the connection of the MCU to the serial network;
- the reset state of the MCU;
- the mode (download or run) of the MCU; and
- interrupt lines.

Again, as with the MCU boards, this PCB has also been used outside the scope of this project.

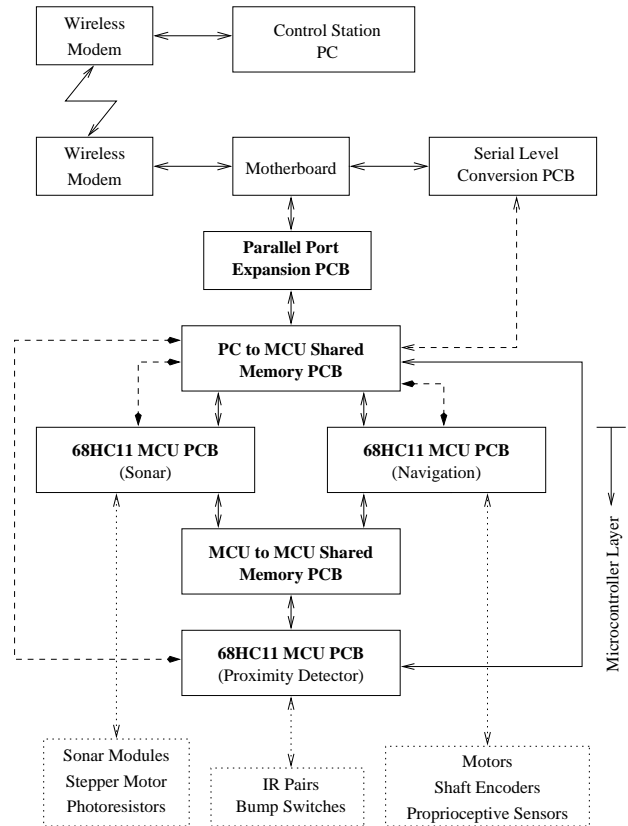


Figure 2: A block diagram of the system electronics.

C. PC to MCU Shared Memory PCB

The purpose of this PCB is three-fold: to provide the primary means of communication between the MCUs and the PC, logic-type conversion (LS to HCMOS TTL) of MCU control lines, and serial network flow control.

Shared memory between the PC and the microcontrollers consists of three banks (one for each MCU) of 4K dual-port static RAM. Shared memory space allows data to be easily transferred to and from the sharing processors and allows great flexibility and efficiency in the implementation of control and intelligence algorithms.

D. Inter-MCU Shared Memory PCB

There is also one 4K dual-port SRAM between each pair of MCUs. Using this configuration, it is possible for the microcontrollers to communicate with one another even if the motherboard has become unstable, yielding robust, safe performance.

IV. SOFTWARE DESCRIPTION

This section provides an outline of the software that has been developed for this project.

All code developed and the various compilers are available to the system user (the researcher) and any desired changes can be made to suit the research area. The software described here is considered a basic ‘building block’, although it was chosen to implement some novel algorithms to achieve a level of performance such that development time of additional software could be minimised.

The teleoperation system makes extensive use of software written in ANSI C, the Tool Command Language (TCL) [7], and the TCL graphical extension, TK. Both the control station and the workcell motherboard run Linux, a PC version of UNIX. Linux comes with a full suite of software tools and is well-suited for connection to networks.

One of the exciting aspects of this teleoperation system is that by using Linux, the workcell is a fully operational network machine. Familiar tools such as **rlogin**, **telnet**, and **ftp** can be used to communicate with the robot. Teleprogramming becomes a trivial task, as all the protocols that are needed are already present. Additionally, the programming tools, such as the GNU C compiler, and TCL, run directly on the workcell.

TCL was chosen for its wonderfully flexible nature. Establishing Internet-Protocol socket connections is very simple. Handling data passed through a socket can be done using either traditional flow-control techniques, or by passing the data to the receiving TCL interpreter itself. With this realisation, the syntax of any teleoperation directive or returning data is completed with little effort. Additional procedures and functions are added to implement the graphical user interface (GUI) and hardware control.

However, TCL is not intended for numerically intensive applications or low-level hardware I/O. In cases where this limitation is encountered, C code is written and compiled directly into a new version of **tclsh**, the TCL shell. The particulars of meshing with the TCL C interface are made transparent with the use of the Software Wrapper Interface Generator (SWIG) [8].

A. Control Station Software: The GUI

The control station’s GUI is written using TCL/TK. There are 5 different GUI modules:

1. The **socket control** module handles connection between the control station and the workcell. It is responsible for routing data to and from the workcell and the other GUI components. Termination of communication with the workcell can be simulated within this module for a convenient method of testing the robustness of system software.

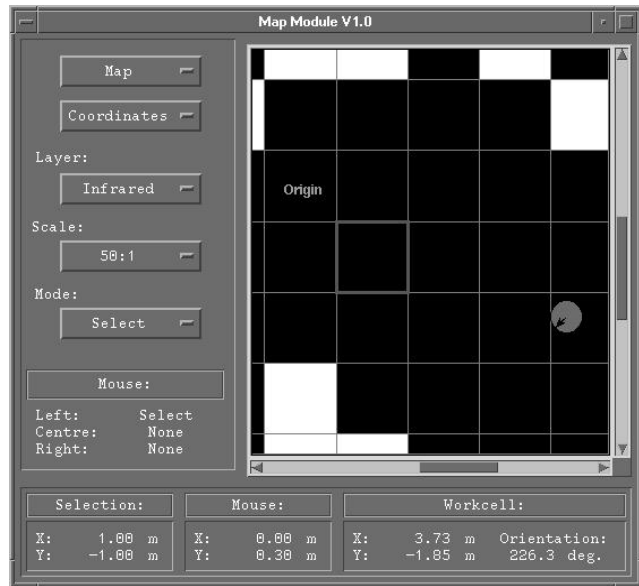


Figure 3: A screen shot of the control station mapper.

2. A **sensor status display** module is used to graphically relate the current status of the workcell bump switches, IR sensors, photoresistors, and proprioceptive sensors.
3. A **navigation control** module is the interface for the user to take control of workcell movement. Establishing speed setpoints, command duration times, and triggering of autonomous navigation behaviours (search and wander) are implemented here. A simplified view of obstacles immediately surrounding the workcell is provided to aid in interactive navigation.
4. The **message display** module, particularly useful during debugging, presents messages intended for the operator from any of the four computational elements located on the workcell.
5. A **mapping** module provides a top-down two-dimensional representation of the workcell’s environment. Obstacles as detected by IR, bump, and sonar sensors are represented on their own layers. Interactive map development, implemented on a dedicated map layer, allows the user to add or delete map obstacles. Already built into this GUI component is an ‘algorithmic’ layer which would allow a pattern-recognition algorithm to place virtual obstacles within the map. Fig. 3 is a screen shot of the mapper.

As mentioned previously **ftp** and **rlogin** or **telnet** are used for teleprogramming.

B. Workcell Software

Most mission-critical functionality is written directly into the microcontroller layer to take advantage of the robustness presented by the hardware design. For instance, survival reflexes, such as stopping the motors when a collision is detected, are written directly on the MCU responsible for controlling the drive motors. Information about the collision-detection sensors is passed through shared memory from the proximity detector MCU to the navigation MCU. The MCU software is written in C and assembly language.

The motherboard, running Linux, establishes socket communication with the control station through use of TCL. Most commands passed to the workcell merely communicate the directive to the MCUs via shared memory. In addition to handling wireless communication with the control station, the motherboard also looks after some computation for which an HC11 is unsuited.

B.1. Autonomous Navigation

Two distinct autonomous navigation behaviours are implemented.

One is a pseudo-random wall-following behaviour (denoted *wander*), which is convenient for creating maps in unknown environments. The entire algorithm is implemented on the microcontroller layer, although the motherboard has the ability to move the workcell into an unexplored region.

The second navigation algorithm is a *search* routine which attempts to achieve a specified map location. The algorithm makes use of discretely assigned virtual potentials and is a modified version of the algorithm found in [9].

V. FUTURE DIRECTIONS

From conception, the intention of the design of this teleoperated mobile robot test-bed system has been expansion and modification. This could not have been achieved if future research areas and system improvements had not been identified. These include:

- changing the workcell motherboard to a notebook computer;
- connection of the system to the World Wide Web, a task which will likely make use of the Netscape TCL plug-in;
- addition of one or two cameras to provide visual feedback to the operator;
- addition of a manipulator to the top of the mobile robot;

- modification of the chassis to handle more varied, rugged terrain.

VI. CONCLUSIONS

This paper has described the key features of a telerobotic system intended for use in a wide range of research areas. The chassis and sensors are suited for indoor operation but the highly modular structure of the system's electronic components would allow smooth re-configuration of the system to operate in other environments. The software has been developed in a variety of programming languages, not only to accommodate the multiple (different) processors present in the hardware, but also to make possible efficient porting of common code. As a beginning point for future development, complete software applications including basic intelligence, autonomous navigation routines, and a control station GUI have been developed. The combination of features provided by this system makes it well-suited for a variety of research topics in teleoperation and mobile robotics.

REFERENCES

- [1] Donald G. Langrock and David R. Broome. Advanced telerobotic controller. In *IEEE Oceans Conference Record*, volume 2, pages 157–162, 1994.
- [2] E. Le Rest, L. Marcé, and V. Rigaud. VORTEX-PILOT: a top-down approach for AUVs mission telerobotics language. In *IEEE Oceans Conference Record*, volume 2, pages 102–107, 1994.
- [3] C.R. Weisbin and D. Lavery. NASA Rover and Telerobotics Technology Program. In *IEEE Robotics & Automation Magazine*, volume 1, number 4, pages 14–21. IEEE, Dec. 1994.
- [4] Paul G. Backes, John Beahan, Mark K. Long, Robert D. Steele, Bruce Bon, and Wayne Zimmerman. A prototype ground-remote telerobot control system. In *Robotica*, volume 12, pages 481–490. Cambridge University Press, 1994.
- [5] J. Benner, J. Hansemann, A. Weber, H. Haffner, and W. Till. Telerobotic control system development in the CEC project TELEMANN-INGRID. In *Proceedings of the Fifth World Conference on Robotics Research*, pages 5–65–5–78, 1994.
- [6] *M68HC11 Reference Manual*. Motorola Inc., 3rd edition, 1991.
- [7] Brent B. Welch. *Practical Programming in TCL and TK*. Prentice Hall PTR, 1995.
- [8] David. M. Beazley. *SWIG Users Manual Version 1.1*. University of Utah and the Regents of the University of California, March 1997.
- [9] L. Wyard-Scott and Q.-H. M. Meng. A potential maze solving algorithm for a micromouse robot. In *Proceedings of the IEEE Conference on Communications, Computers, and Signal Processing*, pages 614–618, 1995.