Dalek-Zero Mobile Robot Project Overview

Jeremy Tavan and Jason Rohrer

May 14, 2001

1 Introduction

The goal of this project was to develop an extensible mobile robot foundation that would be usable for long-term robotics research. When planning this framework, we had the following requirements in mind:

- real-time performance
- robustness against system error or failure
- ease of programming and framework extension

To accomplish these goals, we developed a framework that has several interchangeable modules. On-board, the robot contains local control hardware capable of making split-second reactive decisions based on input from local sensors. Remotely, the system has software running that makes high-level decisions to direct large-scale robot behavior.

2 Hardware

The hardware framework features the following components:

- 1 B12 mobile robot base from Real World Interfaces
- 2 Axis web cameras
- 2 wireless network adapters
- 2 lead acid batteries capable of supplying current for hours of continuous operation
- 1 sonar board capable of supporting up to 60 sonar sensors
- 1 custom-programmed PIC microcontroller for interfacing with the sonar board
- 1 custom-programmed PIC microcontroller for executing local control
- 1 32-processor Linux cluster

3 Control components

The control framework is segmented into two subsystems. The first is a high-level control system which is easily extensible and customizable, and runs on a remote machine. This can't have fast enough response time to avoid obstacles, though, so we have a second control layer running locally on the robot itself which responds quickly to the changing environment.

3.1 High-level remote control

The 32-processor cluster runs the high-level control software. This software makes motion decisions based on image data retrieved from the Axis web cameras. Currently, we have implemented vision algorithms that perform stereo, motion, and color analysis. We have implemented move generating algorithms that perform basic navigation and following. For computationally-intense operations, such as stereo computation, we have implemented a framework that can share the load among all 32 of the processors in the cluster. However, for less complex tasks, such as motion following, only one processor is needed.

3.2 Low-level local control

While powered on, one of the PIC microcontrollers continuously polls the sonar sensors for distance readings. Should any of the readings come back indicating obstacles within a minimum threshold range (currently 3'), the control board sets a pin high. The PIC microcontroller handling movement checks this pin while the robot is moving forwards, and should it go high, the robot initiates basic obstacle-avoidance behavior. This simple algorithm is sufficient to keep the robot from bumping into walls, people, tables, etc.