Implementing Robots in Hardware as a Tool for Integration

Gaurav S. Sukhatme; James F. Montgomery and George A. Bekey

gaurav@monty@bekey@robotics.usc.edu

Department of Computer Science
Institute for Robotics and Intelligent Systems
University of Southern California
Los Angeles, CA 90089-0781

Abstract

We suggest that integration in robotics research is facilitated and enhanced by the process of building embodied robots. It is in the translation of conceptual and simulated models into hardware that the integration takes place. Further, multiple techniques that address similar aspects of the same robotic task have to be merged to achieve good overall performance. In this paper we present two examples of successful integration and cross platform validation namely: 1. A walking and a wheeled robot used for planetary exploration (both used the same obstacle avoidance technique but the implementation was necessarily different) and, 2. Two robot helicopters (where behavior-based control used on the first robot is augmented using learning and enriched sensing on the second). The paper reports both case studies and discusses a list of lessons learned in the process of implementation.

1 Introduction

It is well known in the design of complex engineering systems that integration takes place as the various subsystems find their relative positions within the overall constraints of cost, space, weight, availability etc. A similar process takes place in the design and fabrication of a robot. The design and construction of a complex physical system involves integration, either explicitly or implicitly. On the other hand, simulation of robot behavior does not depend on the integration of subsystems, and such issues as control, navigation, obstacle avoidance, or learning can be studied in isolation. It is in the heat of physical implementation that the integration takes place. This is not to say that integration cannot proceed without fabrication, but rather to suggest that it is a natural consequence of fabrication.

A guiding design philosophy in our lab is to create robots with increasing limits of autonomy. Although simulation is useful in examining issues of robot autonomy it is not enough. We believe a complete study of autonomous agents must take place in the environment in which they will ultimately operate. Therefore, they must be embodied and situated in the real world. These beliefs have driven the design and implementation of our robots. It may be noted that embodying robots in the early prototyping stages is a method that has gained considerable support in recent years. Prominent examples include [3, 5, 4] and the various Navlab vehicles [10] used for off-road as well as highway tests. Other examples of successful integration and cross platform validation using hardware platforms include the Dante robot built at CMU for exploring volcanic craters, which went through two iterations and the Rocky series of vehicles at JPL [6] which were the precursors to the Sojourner rover that was deployed on Mars in 1997.

2 Obstacle Avoidance using Legged and Wheeled Robots

In this section we examine two hardware implementations, a quadruped robot and a modified four wheeled car. Both robots were designed and built as part of a project that dealt with benchmarking the performance of Mars rovers. During the course of the project a statistical methodology was developed to compare the performance of any two robot rovers. The two robots described here were used as evaluation testbeds and were analyzed for performance. The evaluation results have been reported elsewhere (see [13, 14, 12]). In this paper we focus on another aspect of the work, namely lessons learned from the hardware
implementation of two robots designed to do the same overall task (navigation) but with very different kinematics and dynamics. The differing kinematics lead to a necessarily different sensor suite as well as different algorithms for achieving obstacle avoidance and posture control. We show how the low level details of the obstacle avoidance implementation are necessarily different across the two robots. We also point out the need for a posture control module on the legged robot which is achieved using additional sensing and processing. This module is absent in the wheeled robot.

2.1 MENO: A Quadruped Robot Rover

MENO is a 12 DOF statically stable quadruped. Each leg is a rotary-rotary-prismatic (RRP) design. The body of the robot and the first two links of each leg are in the horizontal plane and the prismatic joints (the most distal joint of each limb) are in the vertical plane. This orthogonal design was inspired by the design of Ambler [1]. The main advantages of such a design are ease of motion planning and energy efficiency. Figure 1 shows the robot and Table 2 gives some of its mechanical parameters. The robot is equipped with the sensors shown in Table 1

Computing for both MENO and Marscar is done using an onboard custom-built Motorola 68332 microcontroller-based computer board. A tether is used to supply power and to gather telemetry. All navigation tests (for both MENO and Marscar) were done on a nominally flat sand surface.

Both robots use a hierarchical behavior-based control architecture [2]. Low-level behaviors in the hierarchy are responsible for robot functions requiring quick response while slower acting higher-level behaviors meet less time critical needs. For MENO, low level behaviors are responsible for maintaining balance; the mid-level behaviors for avoiding obstacles and reorientation of the body axis, and the highest level navigator behavior for sequencing the lower level behaviors.

2.2 Marscar: A Wheeled Robot Rover

Marscar is a four wheeled rover with Ackerman steering.\(^1\) The front wheels are actuated with a single motor. The same holds for the rear wheels. The kinematic parameters of the robot are summarized in Table 2. As in the case of MENO, all sensing and navigation decisions are made onboard. However the need for an elaborate balancing and guarded motion strategy is absent since the robot is inherently stable.

The Marscar control system has two mid-level behaviors, obstacle avoidance and reorientation towards the goal. It also has a navigator that acts as a sequencer. There is no low level postural control.

2.3 Comparison

As one can see from Figure 2 the control systems for the two robots are different. The high level navigation algorithm (labeled Navigator in Figure 2) is the same in both cases. At each timestep the robot

---

\(^1\) Ackerman steering maintains a particular relationship between the steer angles of the inner and outer wheels in order that the entire robot turn about a single point.
examines if it is roughly pointing in the direction of the goal. If it is not, the reorient behavior is triggered. If it is oriented correctly the robot moves toward the goal while avoiding obstacles. The entire process terminates when the goal is reached. The avoid_obstacles() behavior uses information from the sonar and IR sensors in the case of the wheeled system. If the sonar detects an obstacle in front of the robot, the robot backs up, turns in a direction which aligns it better with its goal and then moves forward. If the IR sensor on the right (left) detects an obstacle the robot backs up, turns left (right) and then moves forward. In the case of MENO the same behavior uses information from the sonar and contact switches on the individual legs. While a leg is being swung, if it comes into contact with an obstacle, the leg is lifted higher and the swing is retried thus mimicking a well known animal stepping reflex. If this does not work then the robot walks backward, turns (in a direction that aligns it better with its goal) and attempts to walk forward again.

The second difference between the two implementations is the extensive need for monitoring posture as the legged robot walks. This is done by executing guarded motions and constant monitoring of the roll and pitch angles of the robot using the inclinometer. This is shown in the balancing module which is present only in the case of the MENO control system. At each step the balance is checked. If the robot is balanced it is allowed to step. Otherwise its joint angles are adjusted according to a shifting algorithm (discussed in detail in [11]) with all four legs on the ground thereby stabilizing it and enabling it to take a step. At each juncture the robot is also leveled to make sure it is not standing too tall off, or too close to, the ground). Ensuring stability for MENO is also complicated by the fact that its joints have a significant degree of play. This makes the control problem harder since the difference between the actual center of mass and calculated center of mass is substantial at times (see [11]).

3 Helicopter Control

In this section we compare two robot model helicopters. Our implementations have evolved due primarily to three factors; changing system requirements, increasing experience in building robot helicopters and increasing financial resources.

What has not changed is our desire to create robots with high levels of autonomy. This desire is often at odds with the robot implementation which necessitates design tradeoffs. For example, to increase autonomy we want minimal robot dependence upon external resources. Therefore, we locate system power, sensors and computers on the robot itself. Choices made for each are driven by the weight-lifting capabilities of the helicopter. These choices consequently affect the robot control system design.

3.1 The AFV

The design of the AFV (Autonomous Flying Vehicle) (Figure 3) was driven by the requirements of an aerial robotics competition [7]. At the time, the competition required flying robots to locate and retrieve objects and transport them from one location to another. These tasks had to be completed without human guidance and within a fixed time limit. To meet
these competition goals, a platform needed hovering and quick maneuvering capabilities.

**Hardware:** Due to limited financial resources we chose the Kyosho Concept 60, a relatively inexpensive radio control (RC) helicopter, as the platform for the AFV. The Concept 60 was powered by an O.S. 61 size nitro-methane fueled two-stroke engine. Weight-lifting limitations of the helicopter drove us to later replace the O.S engine with one more powerful, the Enya 80. The Concept 60 has a five feet diameter main rotor. It has five degrees of control: main rotor aileron and elevator cyclic pitch (cyclic), tail rotor pitch, main rotor collective pitch (collective), and throttle. The first three control the craft’s roll, pitch, and yaw, and the last two control its thrust.

A variety of sensors are mounted on the craft: a flux-gate compass for measuring heading; three downward facing ultrasonic sensors (two mounted on a crossbar on the front of the robot and one mounted on the tail boom at the rear) for determining the craft’s roll, pitch, and height; an RPM sensor mounted on the main rotor mast for measuring engine speed; and a gray-scale CCD camera to provide visual information. In addition, three solid-state gyros provide angular rate signals used to dampen rotations about the roll, pitch, and yaw of the robot. Finally, the craft’s ferromagnetic object retrieval device uses a tactile sensor that indicates when an object has been acquired.

The robot contains two custom-built Motorola 68332 microcontroller-based computer boards; one to collect data from sensors and control actuators based upon the sensor data and the other a dedicated vision board to process CCD camera data. There is a two-way serial communication link between the two boards. A wireless modem provides a communication link between the robot and a human using a laptop. The human sends high level commands to, and receives telemetry from, the AFV. Nickel-metal hydride batteries supply power to the electronics.

**Control architecture:** Figure 4a shows the AFV’s behavior-based control system architecture. At the lowest control level survival is the main priority. To this end, the robot has a set of fast-acting reflex behaviors that attempt to maintain system stability by holding the craft in a hover, pointed in a specified heading. When the robot detects deviations, the appropriate reflex returns the craft to its stable configuration. The *heading control* behavior attempts to hold a desired heading by using the compass data to drive the tail rotor pitch. The *thrust control* behavior uses the sonar and RPM sensor data to control the collective pitch and throttle. This behavior is responsible for maintaining a desired height above the ground. The *attitude control* behavior tries to hold a stable hover (zero roll and pitch orientation and rate). It uses the sonars to determine attitude and then controls the aileron and elevator cyclic pitch to keep all three sonars the same calibrated distance from the ground.

The *thrust control* behavior uses PID control, while the *heading control* and *attitude control* behaviors use PD control. The *object retrieval* behavior uses target information from the *ego-centric target position* behavior and tactile information to control the retrieval device.

Higher level behaviors in the hierarchy alter the desired goals of lower level behaviors. For example, the *transition to height* behavior inputs a desired height to *thrust control* to move the robot to a new height. At the highest level of control is the sequencer which determines which behaviors to activate and what parameters to instantiate to achieve a desired subgoal. This demonstrates a key advantage of the behavior-based approach, namely the ability to create greater capabilities for the robot by layering more complex behaviors on top of previously constructed behaviors. This addition is transparent to the lower level behaviors, modulating but not destroying their underlying functionality. The remainder of the control architecture will not be discussed. The interested reader is referred to [9].

### 3.2 The AVATAR

The AVATAR (Autonomous Vehicle Aerial Tracking And Retrieval) is currently being built and is our third generation autonomous flying helicopter. The robot was redesigned due to more demanding system requirements, experience with the AFV, and increased financial resources.
The AVATAR system requirements are driven by the needs of a DARPA contract we are currently funded under. The AVATAR must autonomously interact in a dynamic environment in coordination with other autonomous aerial and ground robots. Also, the AVATAR must provide situation information to, and be capable of being retasked by, a human. In order to meet these needs the AVATAR requires increased sensing, communication, and computational resources as well as increased payload capability.

**Hardware:** The AVATAR is based upon the Bergen Industrial Helicopter, an RC model helicopter with a six feet diameter main rotor and powered by a twin-cylinder gas engine. It has a payload capability of approximately 25 pounds (the AVF's is approximately 10 pounds). The AVF has the same five degrees of control as the AFV.

All sensors used by the AFV are also used by the AVATAR. An integrated GPS/INS system has been added that provides position (latitude, longitude, and altitude), velocity (horizontal and vertical), attitude (roll and pitch), heading (yaw), delta theta, and delta velocity information. The gray-scale CCD camera has been replaced with a color version and the roll and pitch solid-state gyros have been removed, leaving only the yaw gyro. The object retrieval device has been removed since it is not currently required.

We have replaced the Motorola 68332 computer boards with a number of commercial off the shelf embedded PC/104 boards. Two 486DX4 CPU boards are used, one as a dedicated host for a color frame grabber and the other for all other AVATAR computing needs such as flight control and interfacing with other PC/104 boards. These boards include a timer/counter board used to generate actuator commands, a serial port board for interfacing to the GPS/INS, and a PC/104 to PCMCIA interface board to allow the use of PCMCIA cards onboard the AVATAR. PC/104 boards stack together via a header that doubles as a communication and power bus. There is a two-way serial communication link between the two 486DX4 CPU boards. A wireless ethernet LAN PCMCIA card provides a two-way communication link between the AVATAR, other robots, and a human using a laptop. The human receives situation information and telemetry from the AVATAR and sends commands to the AVATAR via this link. In addition, differential GPS corrections are sent from the laptop to the GPS receiver onboard the AVATAR through the wireless LAN. A one-way wireless video link sends visual CCD camera information from the AVATAR to a video monitor used by a human. Nickel-metal hydride batteries supply power to the electronics.

**Control architecture:** Figure 4b shows the AVATAR control system. It is a modified version of the AFV architecture which reflects the addition of the GPS/INS system, the removal of the object retrieval device, and a simplified control approach due to the GPS/INS. The GSP/INS is the primary source of sensory data used in heading and attitude control. The sonars are no longer used for attitude control.

The low-level reflex behaviors for the AVATAR are implemented as hybrid fuzzy-neural controllers that are learned using a human teacher [8].

### 3.3 Comparison

Lessons learned from the AFV implementation have helped us to improve the AVATAR in numerous ways, with two example comparisons given here. First, the AFV reflex behaviors were implemented as simple PD
and PID loops with gains laboriously tuned through a trial and error approach. This was both time intensive and dangerous. We are currently developing a “teaching by showing” methodology to automate controller generation and tuning. Each behavior is implemented as a hybrid fuzzy-neural controller generated and tuned using training data gathered while a human teacher controls the helicopter. The methodology has been successfully applied in simulation and will be applied on the AVATAR for real world validation.

Second, we depended upon the sonars as our primary source of attitude control and vision as our primary source of navigation on the AFV. These choices were made due to payload and fiscal constraints. Sonar-based attitude control limited the AFV to flight over level terrain at low altitudes. Vision-based navigation limited the AFV to areas mapped in advance and rich in landmarks. The AVATAR’s flight domain will be increased and the control architecture simplified by the addition of the GPS/INS system for attitude and navigation control.

4 Conclusion

We have presented two case studies in this paper. The first is a comparison of a legged walking robot and a wheeled robot, both built for planetary exploration. We reported how the hardware implementation of the robots forced us to use different low-level algorithms and sensors for each robot to achieve the same task of obstacle avoidance. In many simulated instances, a robot is represented by its convex hull and obstacle avoidance is solved by invoking a collision detection function that typically deals with intersecting polyhedra. This level of abstraction is impossible in a hardware implementation.

The second case study is a comparison between two implementations of autonomous robot helicopters. The earlier approach lacked a strategy to generate and tune control parameters and suffered from a paucity of accurate sensing. Without giving up the advantages of a behavior-based control architecture we are able to facilitate robot integration by introducing learning into the system and enriching sensing.

Acknowledgments

We thank all the members of the AFV, AVATAR, MENO and Marscat teams for their help in building these systems. This work is supported in part by JPL, Caltech under contract #939816; by DARPA under contract #F04701-97-C-0021 and by Raytheon Systems Company through a Doctoral Fellowship for the second author.

References