16

Visually Guided Reaching for an Autonomous Mars Rover

by

Chandana Paul

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degrees of

Masters of Engineering in Electrical Engineering and Computer Science

and

Bachelor of Science in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 1998

Author .....
Department of Electrical Engineering and Computer Science
February 4th, 1998

Certified by .....
Rodney A. Brooks
Professor, Department of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by .....
Frederic R. Morgenthaler
Chairman, Department Committee on Graduate Theses

Visually Guided Reaching for an Autonomous Mars Rover

by

Chandana Paul

Submitted to the Department of Electrical Engineering and Computer Science
on February 4th, 1998, in partial fulfillment of the
requirements for the degrees of
Masters of Engineering in Electrical Engineering and Computer Science
and
Bachelor of Science in Computer Science and Engineering

Abstract

This thesis describes the design and implementation of an engineering solution to the problem of autonomous geological sampling with a manipulator mounted on a robot Mars Rover. Most of the research to date in the field of remote manipulation has focused on teleoperated technologies. The Pebbles Mars Rover project pushes the envelope of autonomous operation instead, in the interests of drastically reducing operation time and costs. The following implementation of rock collection by the Pebbles rover, in the absence of human control, using a five degree-of-freedom arm with a parallel jaw gripper and a single "eye-on-hand" camera is aimed at showing the viability of this approach. The implementation has three main components: a vision algorithm for rock detection, a visual servoing mechanism to reach to the rock, and a dextrous manipulation scheme to grasp the rock. The thesis will present the design of these three components separately and describe how they operate together. It will also present performance data and results for the operation of these components.

Thesis Supervisor: Rodney A. Brooks
Title: Professor, Department of Electrical Engineering and Computer Science

Acknowledgments

I would like to thank my advisor, Rod Brooks for giving me the opportunity to work on this exciting project and for being there to bail me out of all the tight situations along the way.

I would like to thank Matt Williamson, James McLurkin, Liana Lorigo, Milyn Moy, Yoky Matsuoka and Jonah Peskin for their major contributions to the realization of this project. I also want to thank them and others including Robert Irie, Brian Scassellatti, Matt Marjanovich, Cynthia Ferrell, Charlie Kemp, Makoto Yoshida, Juan Velasquez, Junji Yamato and James Speranza for their continual support and friendship.

I would also like to thank Rosario Robert and Eddie Miller at ISR for their excellent technical support.

Thank you to the Zoo members for all the stimulating discussions.

Thanks, as always, to all my friends for helping me get through this one.

And finally, I would like to thank my family Nirmalendu, Santa, Tias and Vivek Paul for always believing in me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Sample Collection on Mars

Remote sample collection on the surface of Mars, in the absence of human control, presents a unique and challenging problem in the field of robotic manipulation. There are several issues that make traditional robot manipulation strategies unfit for this task.

In industrial settings where robot manipulators are used for grasping the size, shape and hardness of the objects are fixed, or vary within a small range. This makes it simple for a robot with a single hard-wired manipulation strategy to be successful at grasping every object it encounters. In the case of the sample collection problem there is no such regularity. Rocks vary in shape, size and hardness so the grasping strategy must be flexible enough to work with a wide range. Further, unlike in a factory situation where the small range allows for all objects to be grasped by the manipulator, in nature there are always rocks that are too large or too small for a manipulator so the grasping strategy must incorporate a way of selecting only those rocks which can be reasonably grasped and ignore others.

In addition to this first problem of having a wide range of object characteristics there is a further problem that the nature of these characteristics are largely unknown to us. We are fortunate enough to have a large database of digital images from the Mars Pathfinder which give us an approximate idea of the visual characteristics of

the surface. However, while the APXS readings provide scientists a good feeling of the chemical composition of the geology, they do not give a good description of the haptic nature of the environment as in how hard or soft things will be, how sticky or hard the soil is. A grasping strategy must prepare for this unknown range of haptic characteristics.

In most traditional grasping tasks the objects lie on rigid horizontal surfaces. Soil, however, may not always be rigid or horizontal. In fact it is easily manipulable itself so the shape of the surface and position of objects on it can change rapidly on contact with the manipulator.

Natural environments unlike factory environments are unstructured so exact locations of objects are unknown and estimates are usually error prone. Manipulation algorithms for sample collection must therefore be much more robust in the face of uncertainty than algorithms that can successfully be used in industrial settings.

The final problem is that a natural outdoors environment is much more rugged so the system must be prepared to deal with physical interactions in a much harsher environment without causing damage to its equipment.

1.2 An Autonomous Solution

The traditional answer to manipulation problems in uncertain environments is to introduce a large amount of human supervision and control into each robot action. While this may be a successful method of robot control it is often slow, tedious and expensive in terms of time and resources.

This thesis describes an alternative solution that does not rely on human supervision to deal with the problems described above. Instead it uses a combination of different engineering techniques that allow it to deal with each of these problems. The wide range of object sizes and shapes is dealt with by using a vision algorithm that only detects objects within a certain size range. The unknown nature of the location of objects to be grasped is solved using a vision based reactive reaching strategy. The uncertainty in the size, shape and hardness of objects, and the non-rigidity of

the underlying surface is dealt with using proprioception for grasping. Finally, the unstructured and rugged nature of the environment is coped with using a compliant manipulation strategy that reacts to the forces of the environment.

The remainder of the thesis will discuss each of these components in detail, and describe how they are combined to form a successful sample collection strategy for the Pebbles Mars Rover.

1.3 Organization of Thesis

The remainder of the thesis is presented as follows:

Chapter 2 reviews some of the previous research relevent to this work.

Chapter 3 presents the physical design of the Pebbles Mars rover, and its computing hardware.

Chapter 4 describes the visual rock detection algorithm which calculates visual error.

Chapter 5 discusses compliant manipulation, and the visual servoing algorithm that makes the arm reach towards a detected rock.

Chapter 6 describes the dextrous manipulation by the parallel jaw gripper hand to grasp the rock, and ways in which the system verifies that it has acquired the rock.

Chapter 7 concludes the thesis with a review of the project and ideas for future work.

Chapter 2

Related Work

This chapter will review the related research which has provided the foundation for the work in this project. The first section will provide a background in traditional manipulator force control. It will then discuss the force control method used in this project which is a large departure from the previous work. The second section will discuss background work in the field of vision based position control of manipulators which relates to the current work in reactive visual servoing. The rock detection algorithm uses techniques of image segmentation and color histogramming. These are discussed in the third section. The fourth section will describe research on grasping using parallel jaw grippers. The fifth section will discuss the ideas of reactive control and subsumption architecture which provide the design principles for the organization of behaviors for the Pebbles reaching task.

2.1 Manipulator Force Control

Traditional robot manipulators in industrial settings have been using stiff joint control for high bandwidth force control and precise position control. There are several variations in this type of force control which are reviewed in detail by Whitney [48]. However, stiff joint control suffers from problems of contact instability and high force errors. To overcome these problems a new Series Elastic Actuated joint was designed by Pratt and Williamson [39]. These joints are controlled as virtual springs enabling

compliant force control that reacted to the forces in the environment. This strategy was first implemented and used on the arm for the Cog humanoid robot [49]. It is also the force control method used for Pebbles' four degree-of-freedom manipulator.

2.2 Vision Based Position Control

There have been several ways in which vision sensors have been used to provide sensory feedback for manipulator control. The traditional approach is to use the visual input to construct a 3D representation of the robots visual field [30]. This method requires precise calibration of the cameras and involves the additional step of mapping the visual information to the 3D coordinate space. A broad review of these strategies is provided by Weiss *et al* [47].

Recently, the biologically inspired active vision paradigm, proposed by Ballard [3] and Aloimonos [2], has had a large impact on vision-based manipulator control. Active vision algorithms only extract task-specific sensory information instead of constructing a 3D model containing large amounts of irrelevant information. This paradigm has inspired the use of visual servoing methods for manipulator control, first used by Espiau *et al.*[15] [14], in which the error function is directly computed in image coordinates.

There are three configurations that are used in visual servoing systems. In fixed camera systems, the vision sensors are static and positioned so that both the end-effector and the object are in the field of view. The error is calculated by detecting the difference in location between the end-effector and the object in the image. This scheme has been used by Koivo and Houshangi [28], Allen *et al* [1], and Metta *et al* [35] among others.

In moving camera systems, the vision sensors are not static, but move independent of the motion of the arm. This kind of system is rare. The single prominent example of this is the visual servoing algorithm implemented for the Cog humanoid robot [32].

In "hand-in-eye" systems the vision sensor (usually a single camera) is mounted on the arm, near or on the end effector. The advantage of this configuration over a static

or moving camera system, is that the visual error measurement only requires detecting the object position, as the arm position can be assumed to be slightly behind the camera along its optical axis. Thus, the visual detection problem is reduced allowing for more efficient control. A number of control strategies for these systems have been studied by Jang *et al* [26], Feddema [16], Hashimoto *et al.* [22], and Papanikolopoulos and Khosla [38]. Pebbles visual motor configuration is most similar to these systems, except that the camera has an extra degree-of-freedom which is independent of arm motion. This leads to a different kind of control algorithm than the ones previously studied.

2.3 Image Segmentation

In order to use vision for position control, the location of the object must be detected in the image. This involves segmenting the image into homogeneous regions and locating the region which corresponds to the object. Image segmentation is a well studied field and has been based on several different image features such as color, brightness or texture [4] [21] [40].

The algorithms developed for segmentation can be classified into three categories. Neighborhood based segmentation algorithms use Markov Random Fields (MRF), which assume that the probability of a pixel to belong to a certain object is dependent only on its neighbors and not on the whole image. There have been several variations on MRF algorithms mainly for color and texture segmentation developed by Dubes *et al* [12], Liu & Yang [29], Geman *et al.* [19] and Muzzolini *et al.* [36] among others.

Physically based segmentation uses underlying models of the image formation process to extract color difference metrics. Klinker, Shafer and Kanade proposed an image formation model called the Dichromatic Reflection Model (DRM) [27] which has been subsequently used by Healey [23] for color image segmentation. However, this model assumes stringent image conditions and is not robust on realistic scenes.

The histogram based segmentation approach is most closely related to the work in this thesis. This approach assumes that regions of the image that belong to the

same object form clusters in the measurement space. Witkin proposed a method for histogram analysis called the Scale Space Filter (SSF) "which enables most prominent and stable peaks to be found" [50]. The main techniques used in this approach are clustering [46], relaxation [13], and region splitting [37].

2.4 Grasping

Research on robotic hands can be clearly divided into two categories: anthropomorphic dextrous hands and parallel jaw grippers. The first successful anthropomorphic hand was called the Utah/MIT hand, built by Jacobsen *et. al.* [25]. Much research in force contact analysis of these hands and grasping strategies for known shapes has been done by Salisbury, Brock and Bicchi among others [5] [6] [44]. Recently a dextrous hand has also been designed for the Cog humanoid robot by Matsuoka [34].

While being much simpler, the parallel jaw gripper is also robust for several grasping tasks. This has been shown by Canny and Goldberg who favored its use due to its *Reduced Intricacy in Sensing and Control* (RISC) [10]. This claim has also been supported by Goldberg and Rao [20] [43] [42]. Recently, Teichmann and Mishra have also developed a robust reactive algorithm for such a gripper using light beam sensors [45]. The grasping algorithm that will be described later is comparable to that developed Teichmann and Mishra, but uses haptic sensors.

2.5 Reactive Control

The implementation of Pebbles controller is based on the philosophy of reactive control. In reactive control each of the robots actions are an immediate reaction to a sensory perception. Due to this tight loop between sensing and action, a robot can act rapidly and robustly in widely varying environments.

The design principle of *subsumption architecture* was put forth by Brooks [41] for structuring a robot task implementation based on reactive control. In this architecture, each task is composed of a number of subcomponents called *behaviors*. The

lowest level behaviors are simple reactive sensory motor loops as described above. More complicated higher level behaviors are emerge by *layering* the lower level behaviors.

In the original formulation each low level behavior was viewed as an Augmented Finite State Machine. It received sensory inputs which were fed through the FSM combinatorial logic to produce outputs to the actuators. A higher level behavior could be connected to a lower using *suppression*, *inhibit* or *default* nodes. If a suppression node was used, the higher level behavior could suppress the inputs to the lower level behavior. If an inhibit node was used, the higher behavior could inhibit the outputs of the lower behavior. If a default node was used the lower level wire could gain control of messages along that wire. In each of these cases it was required that messages continue to be sent to the node in order for suppression or inhibition to continue.

The subsumption approach outlined a clear method for the incremental design of complex robotic tasks. This architecture was successfully used by Mataric [33] to develop an autonomous mobile robot that could build maps based on sensory data. It was used by Ferrell [17] to implement adaptive rough terrain walking for the Hannibal hexapod. It was also shown by Connell [11] to be a successful strategy for implementing a controller for a robot with a manipulator, which was used to collect Coke cans around the lab.

While the controller design for Pebbles has not adhered to the rigid formalism of the original subsumption architecture approach, it has utilized its general principles to structure the implementation. The lowest layer of behavior for Pebbles is a reactive control strategy for navigation developed by Lorigo [31]. This layer receives sensory inputs from the vision sensor and sends output commands to the tread motors to avoid obstacles. The implementations of reaching and grasping are layered on top of this primary avoidance behavior.

Chapter 3

The Pebbles Mars Rover

The following chapter will introduce the Pebbles Mars rover robot. The first section will describe the physical rover chassis and its high level control hardware. The second section will describe the structure of the arm and its degrees of freedom. It will also explain the design and control issues of each joint actuator. The third section will describe the design of the hand and joint control. The last section will describe the vision hardware and its interface with the main control hardware.

3.1 Main Control

The Pebbles III mobile robot platform was designed and built by IS Robotics. Inc. The main body consists of a hollow rectangular chassis (Fig. 3-1). It is 16" long, 10" wide and 5" high. Mounted on tread wheels it stands 7" off the ground. The tread wheels make the rover suited for navigation on rough outdoors terrain.

The chassis carries all the computing hardware on board. A schematic of the main hardware is shown in Fig. 3-2. The main high level control of the rover is performed by the Motorola 68332 processor. This processor communicates with the 6811 status processor which monitors the low level functionality of the rover. The 68332 is also responsible for sending commands to the tread motors.

The processor runs the Venus operating system developed for behavior based

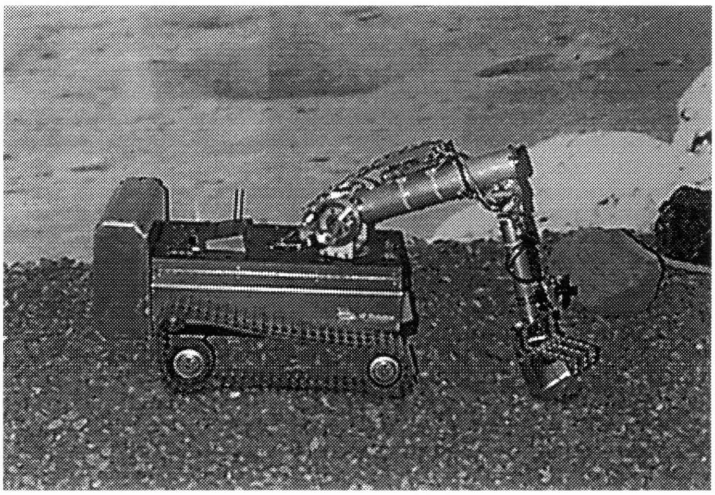

Figure 3-1: The Pebbles Mars Rover

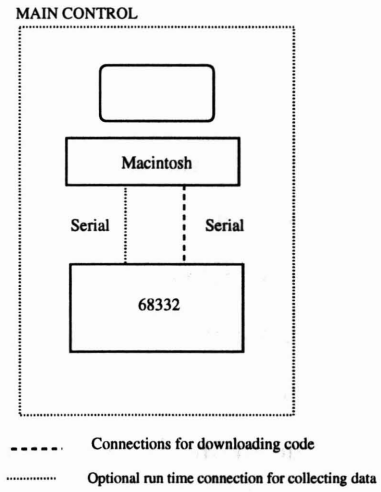

Figure 3-2: Main Control

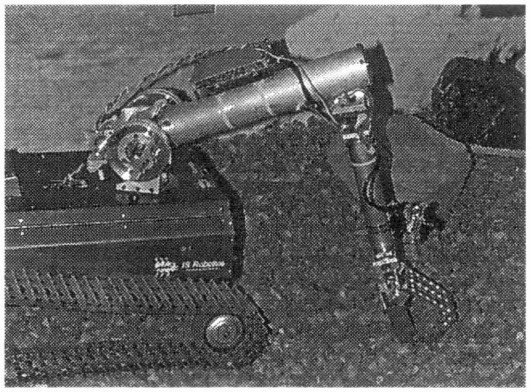

Figure 3-3: Closeup of the arm

control of mobile robots. The programming language for the system is L which is a subset of Common Lisp developed by Brooks [7]. The MARS [8] and Humanoid [9] programming environments are embedded in L. They allow for structured design of multiple asynchronous parallel process programs.

The front end interface to the 68332 is a Macintosh PC. L code can be written and compiled within the MCL environment on the Macintosh and downloaded to the 68332 through the serial line. During execution of code, data can also be returned to the Macintosh interface via this connection. The run time connection shown in Fig. 3-2 between the Macintosh and the robot is entirely optional and can be used for retrieving data back from the robot. The robot can function autonomously without this connection once the code has been downloaded and stored.

3.2 Arm

Pebbles' arm-like manipulator is mounted on the front end of the rover chassis (Fig. 3-3). It has four degrees of freedom as labeled in Fig. 3-4 The *base* joint is a revolute joint which rotates the arm about the vertical axis. The shoulder joint is composed of two actuators connected in a differential drive configuration. This gives the shoulder two degrees of freedom. If both actuators are driven either clockwise or counterclockwise a rotational motion of the joint about the axis of the L_1 link results. This is called the *rotate* degree of motion. If the actuators are driven in opposite directions,

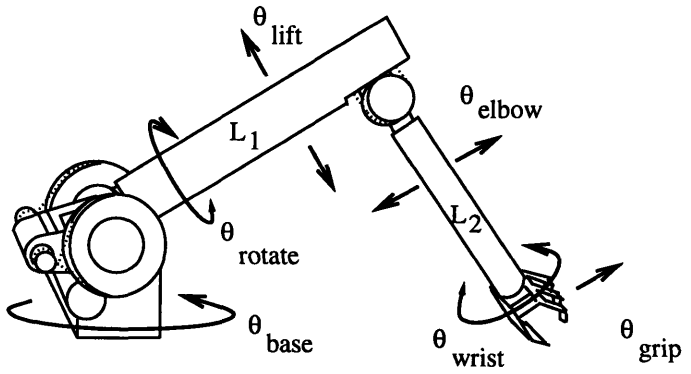

Figure 3-4: Schematic diagram of arm joints

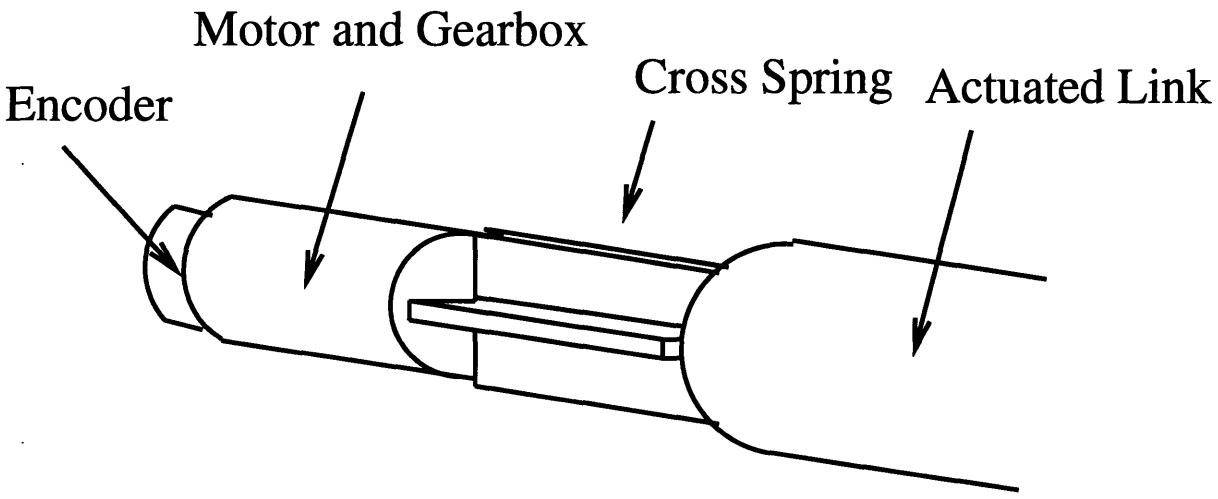

Figure 3-5: Series elastic actuated joint

the L_1 link rotates higher or lower in the vertical plane. This is labeled as the *lift* degree of freedom. The *elbow* is also a revolute joint which allows the L_2 link to rotate about its proximal end. The plane of rotation is defined by the axes of the L_1 and L_2 links.

Each of the joints on the arm is cable driven. Cable drive mechanisms facilitate the change of direction between the motion of the motor end piece and driven link. Since motors do not have to be placed in direct series with the link, cable drives are useful for positioning motors in space-efficient ways with respect to the arm links. For example, the motor for the L_2 link is housed inside the hollow shaft of the L_1 link.

Each arm joint is a series elastic actuator ([39]). In this design, an elastic element

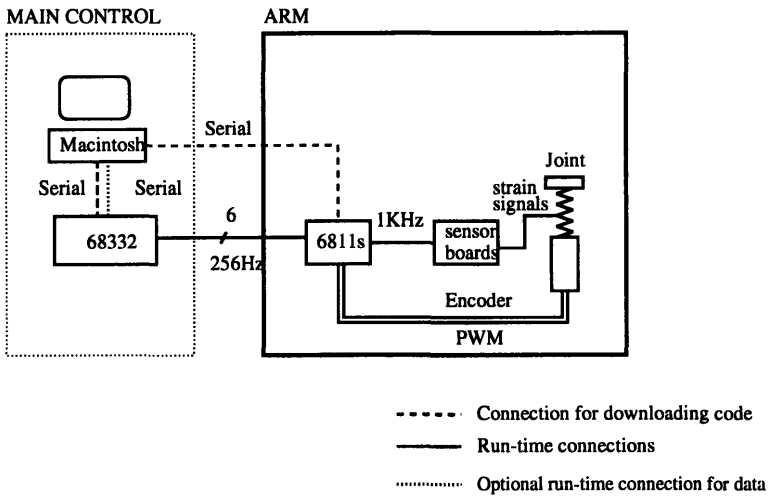

Figure 3-6: Block diagram of hardware setup for arm control

is incorporated between the motor and the driven load. The elastic element is a cross shaped torsional spring made of hardened steel (Fig. 3-5).

Each spring has a pair of strain gauges mounted on opposite sides of one of the plates to measure the twist of the spring. The strain signals are sent to sensor boards with Analog Devices 1B31ANs where they are filtered and amplified (Fig. 3-6). The strain and differentiated strain velocity signal are sampled via analog to digital converters on the motor control board.

The main processor for the motor control boards is a Motorola 6811 microcontroller. 6811 code is programmed in Assembly language on the Macintosh and downloaded via the serial port. The 6811 runs a joint control loop at 1KHz. The control code uses the strain, strain velocity and motor encoder values to calculate the actual joint angle of the link. The code is calibrated to each joint so that the range of joint motion is represented by position values between -128 and 128. Compliance is introduced by controlling each joint as a virtual spring at the low level. To do this, the servo loop calculates the force to be applied to the joint as a PID function of the difference between the actual and desired joint positions. This force is then converted to a PWM signal which is sent out to the motor.

The desired position of a joint is called its *setpoint.* The setpoint represents the equilibrium position of the joint in the absence of environmental forces. High level

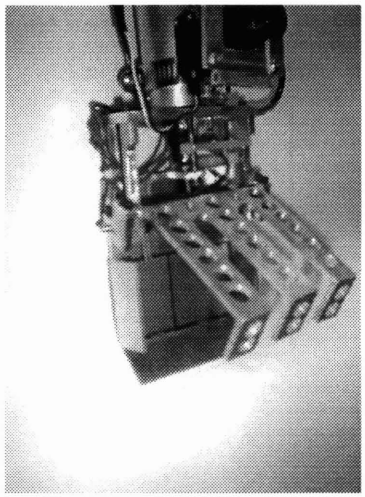

Figure 3-7: The three fingered parallel jaw gripper

position control of the arm is performed by processes on the 68332 which communicate the joint setpoint values to the 6811. This communication link runs at a speed of 256Hz for each joint.

3.3 Hand

At the end of the arm is attached a "hand" (Fig. 3-7). It has one degree of freedom at the wrist Θ_{wrist} which allows it to rotate about the axis of the L_2 link. The second degree of freedom is the movement of the parallel jaw gripper Θ_{grip}. The gripper consists of a rigid flat scoop-shaped bottom plate and a top plate with three finger like extensions. The bottom plate does not move. The top plate rotates up or down to open or close the gripper.

Unlike the arm joints, each of the joints on the hand are driven directly by 12V motors with a 15/5 gear ratio. Each motor has a 10KΩ potentiometer attached at the end which is used to measure the position of the top plate. Both the gripper and wrist joints are controlled by dual 6811 boards like the arm joints. The control loop for each joint is run at 1KHz. The potentiometer values are read and converted into the actual joint position. A constant force control algorithm is implemented. If the actual position of the joint is within 3 units to the left or right of the desired setpoint a force proportional to the difference in position is applied. If the actual position is

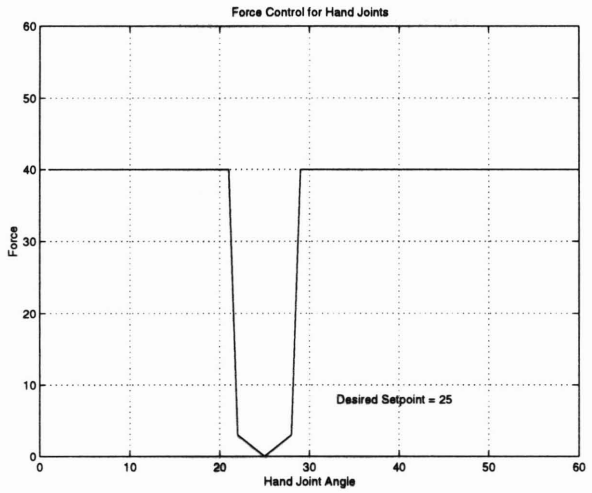

Figure 3-8: Low level force control of hand joints

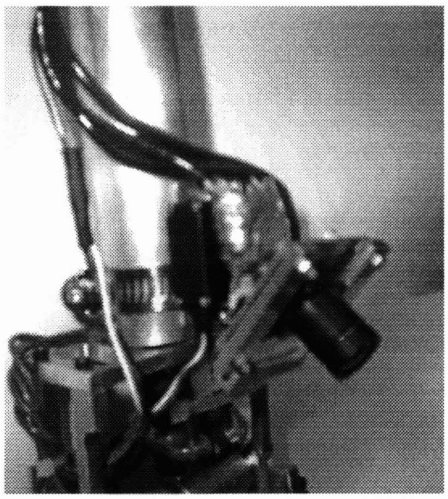

Figure 3-9: The "eye-on-hand" camera

further than 3 units away, a constant large force is applied. This is shown in Fig. 3-8. Proportional control is used instead of constant force control in the local vicinity of the correct position to prevent chatter caused by an unstable equilibrium. As with the arm, the joint setpoints are communicated from the processes on the 68332 at 256Hz.

3.4 Vision

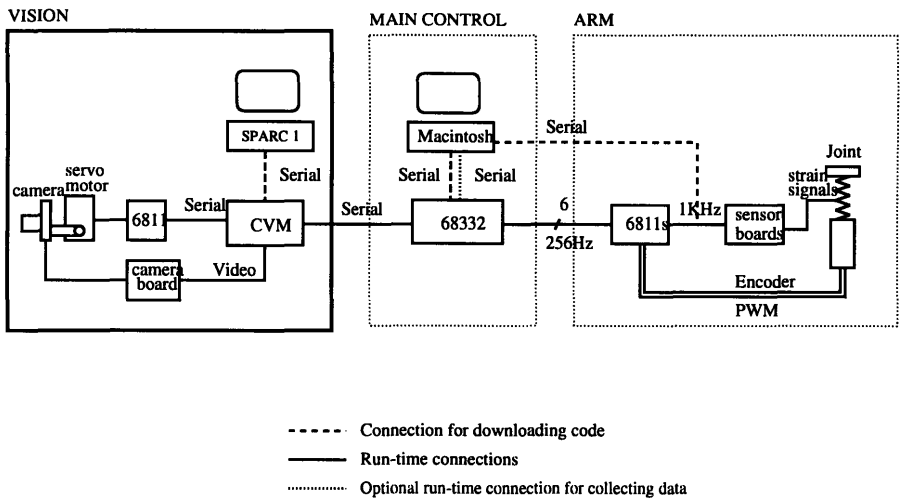

Figure 3-10: Block diagram of hardware setup for vision

The single vision sensor for Pebbles is a Chinon 3mm wide angle camera mounted on the L_2 link directly above the hand (Fig. 3-9). This is the only sensor used for both navigation and rock collection.

The video output from the camera is digitized and processed as shown in Fig. 3-10 by the Cheap Vision Machine (CVM). This vision processor designed by DIDEAS. Inc. is based on Texas Instruments C30 DSP. It holds 1MB of memory. A Sparc Station provides the front end interface for writing and compiling C code. The code is downloaded to the CVM via a serial port connection.

The camera is mounted on a Futaba servo motor as can be seen in Fig. 3-9. This allows it to rotate in the vertical plane. The motor is controlled by a 6811 processor which runs a simple control loop to read values sent over the serial port and produce a proportional PWM signal to the camera motor. The camera position values are sent to the 6811 board over a serial port connection from the CVM.

The CVM communicates with the 68332 via a serial port connection as well. The CVM sends the result of the vision processing algorithms for navigation and rock detection to the 68332. It receives high level position commands for the camera motor from the 68332.

Chapter 4

Rock Detection during Navigation and Reaching

The following chapter will present the method in which the visual input from the camera is processed to detect the existence of a rock. The first section presents a discussion of the characteristics of the visual field at the "rest position" of the arm and the camera during navigation. The second section shows how these characteristics are used to segment the original image and retrieve the location of a rock. Finally, the performance of the rock detection algorithm during navigation is analyzed on a set of images and the results are discussed in the last section.

4.1 The Visual Field

The image of the ground captured by the camera is dependent on the position, the angle and the height of the camera with respect to the ground. Since the camera is mounted near the end effector, the camera position and angle with respect to the ground are dependent on the joint angles of the arm and the angle of the servo motor. Thus the characteristics of the image captured can vary widely with the position of the arm and servo motor.

In order for the robot to be efficient it would be desirable that navigation and rock detection occurred simultaneously. Since there is only one camera for both of these

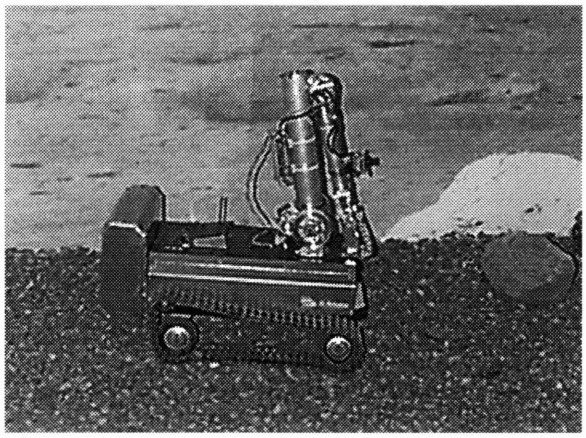

Figure 4-1: Arm and Camera in *rest position*

activities, the camera must be placed in such a position that the camera looks out far enough to detect all obstacles in its way but is also positioned such that it has a clear view of everything on the ground immediately in front of it. In order to achieve this ideal position for the separate activities, we hold the arm and camera at *rest position* during navigation. In this position, the L_1 link of the arm is at 0° with respect to the vertical axis, and the L_2 link is at approximately 10° with respect to the L_1 link. The *base* joint and the *rotate* joint are both at 0° which means that the camera is facing in the forward direction with respect to the robot body. The camera is held at a setpoint of 185, which translates to an angle of 40° with respect to the axis of the L_2 link. This configuration is shown in Fig. 4-1. Holding a constant configuration during navigation ensures that the camera remains at a fixed egocentric position with respect to the chassis providing an appropriate field of view for both the navigation and rock detection algorithms.

Using the same video input, the CVM can grab images at four different resolutions. Although the camera remains at the same position, a low-resolution wide-angle field of view image is grabbed for navigation Fig. 4-2, while the rock detection algorithm focuses on a high resolution image of the patch of ground directly in front of the robot, as shown in Figs. 4-3 and 4-4. Thus rock samples are only detected in the local proximity of the robot. This choice is necessary for a few reasons. Firstly, the size of rock that can be picked up by the robot hand is fairly small and due to the diminishing apparent size of objects with distance, these rocks appear even

Figure 4-2: Wide angle field of view image used for navigation

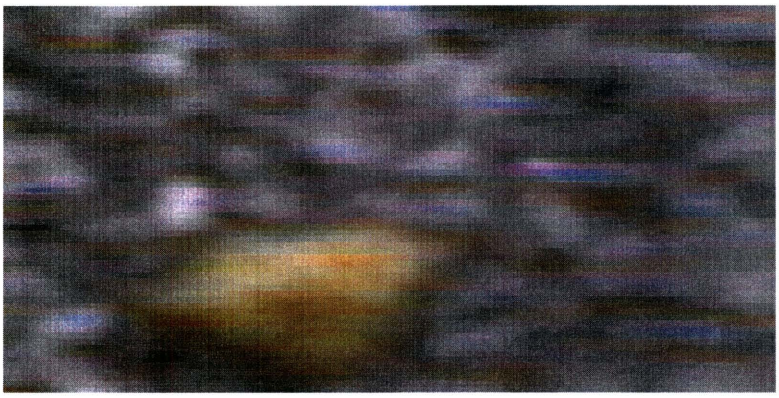

Figure 4-3: Local field of view image with reddish rock on grey rocky ground

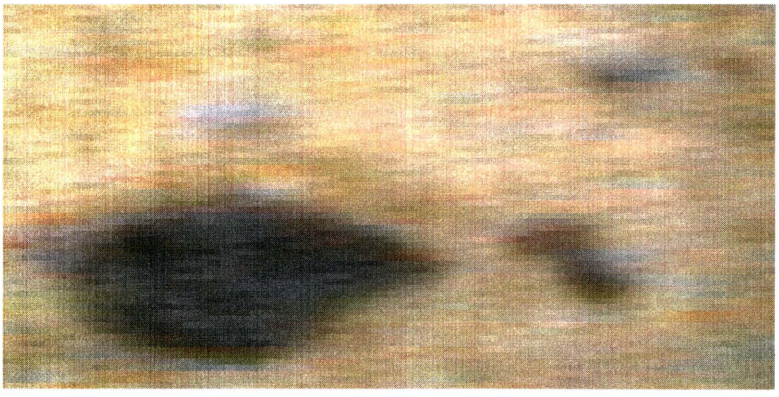

Figure 4-4: Local field of view image with rock with grey rock on sandy soil

smaller as they get further away. It would be quite difficult to detect them using any vision algorithm at a distance. Secondly, even if it were possible to locate a rock at a distance, this would involve spending time, energy and computational resources navigating to the rock which is more inefficient than an opportunistic local sampling strategy.

The local visual field, making a planar ground assumption is a patch of ground, 6" long and 5" wide directly in front of the robot. It usually consists of ground with some rocks and looks as shown in Figs. 4-3 and 4-4.

The problem of selecting the right size rocks in the local visual field is simplified to a large extent due to the layering of the detection on top of the navigation behavior. Because the navigation behavior causes the robot to veer away from large rocks before they appear in the local visual field, the problem of dealing with large rocks in the local field of view is eliminated. Only in extremely rare situations the will rover get trapped in a rock configuration, where it must very close to large rocks in order to find its way out.

4.2 Detection Algorithm

4.2.1 Single Image Detection

The rock detection algorithm detects the rock using a single image. This allows the rover to continue navigating as it scans single snapshots of the ground for possible rocks. A multiple image algorithm, for example using motion cues, would require that the same rock be visible in successive images which would restrict the rovers navigation during this time.

4.2.2 Image Features

In a single image, a rock can be distinguished from the ground based on aspects such as color, brightness and visual texture. In the high resolution images of the local visual field, as shown in Figs.4-3 and 4-4, color and intensity differences are salient

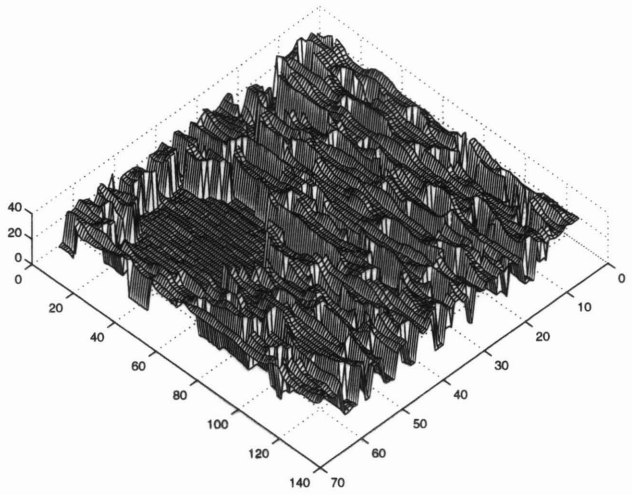

Figure 4-5: Result of RGB to Hue transformation on local field of view image

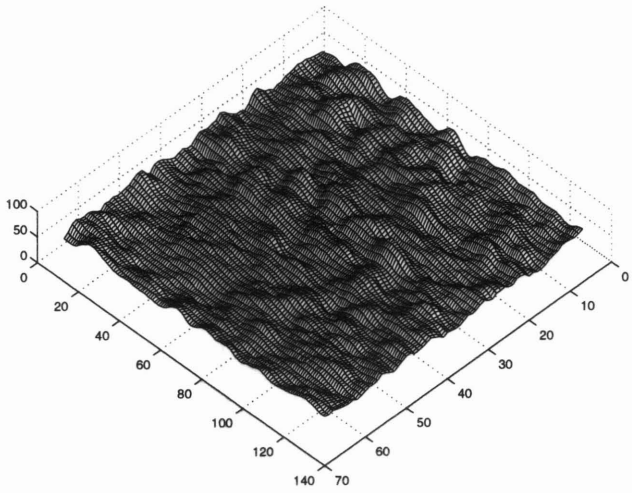

Figure 4-6: Result of RGB to Intensity transformation on local field of view image

and can be used to detect the rock.

Each raw image digitized from the video input is a 128X64 pixel RGB image. Each RGB value is represented by a byte of data. To extract the color at each pixel, independent of the effects of light shadow, an RGB to Hue conversion is performed [18]. This reduces the three dimensional RGB image to a one dimensional image representing hue, in which low values represent reddish tones, and high values represent blues. The Hue is determined by the following conversion involving r, g ad b, which are the red, green and blue components at each pixel.

$$h_{max} = MAX(r, g, b) \tag{4.1}$$

$$h_{min} = MIN(r, g, b) \tag{4.2}$$

$$\delta = h_{max} - h_{min} \tag{4.3}$$

IF r = h_{max}

$$hue = \frac{5(g - b)}{\delta} mod 30 \tag{4.4}$$

IF g = h_{max}

$$hue = \frac{5(b - r) + 10}{\delta} mod 30 \tag{4.5}$$

IF b = h_{max}

$$hue = \frac{5(r - g) + 20}{\delta} mod 30 \tag{4.6}$$

Fig. 4-5 shows the result of this RGB to Hue transformation performed on a local field of view image with a reddish rock on the rocky ground as in Fig. 4-3.

To extract intensity at each point, independent of color, an RGB to grayscale conversion is carried out which gives a one dimensional image representing brightness. The intensity, I, at each pixel is determined by the following equation:

Fig. 4-6 shows the result of the RGB to grayscale transformation performed on a local field of view image as in 4-3.

4.2.3 Image Filtering

If there is a rock in the original image, the hue and/or intensity image has an enclosed blob shaped region in which the values are significantly higher or lower than the values in the surrounding region (Figs. 4-5 and 4-6). Using an intermediate threshold

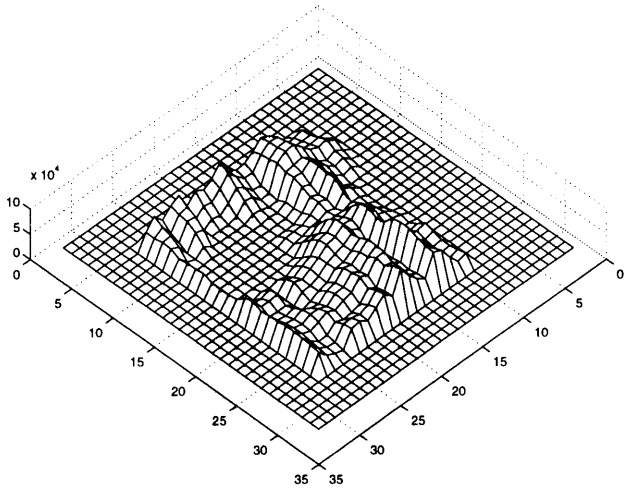

Figure 4-7: Result of reducing the hue image and filtering it with a blob detection filter

value to partition these images results in a noisy segmentation in which the blob corresponding to the rock cannot be distinguished from other anomalous patches. Further processing is needed to single out the blob corresponding to the rock from the others.

The image is first reduced to a size of 32x32 to average out some of the noise. Then it is filtered with a scaled "blob detection" filter. The blob detection filter used here generalizes on the convolution strategy for edge detection [24]. For unidirectional edge detection, the image is convolved with a stencil with 1s down one side and -1s on the other. This has the effect of giving values near zero when there are no edges and extreme values where there is an edge along the axis of the stencil. The mask used to filter the images in this algorithm is a square mask with a positive constant value in the center and a negative constant value in the outermost periphery layer. This kind of filter can be thought of as combining the effects of four unidirectional edge detection filter placed in a particular configuration with respect to each other. If similar edges are found in all four directions an extreme positive or negative value results. If the separate filters produce values of opposite signs or zero, the resulting value is closer to zero.

The size of the mask determines what size blobs produce the highest absolute values. For example, a 3x3 mask with a positive value in the center and a negative

values around it, will respond with highest absolute value to a blob of size 1. Since the goal of the detection algorithm is to detect rocks which will fit in the hand, the size of the mask used is dependent on the size of the rock as it appears in the image. Usually the rocks which are of a size to fit in the hand are around 1" in diameter. They appear in the image as about 10-20 pixels in diameter. If a mask of size smaller than 11 is used, the averaging is not performed over a large enough area to smooth out blobs caused by noise. However, if a mask of size larger than 11 is used some smaller rocks may be smoothed out so much so that they are undetectable. A mask size of 11 gives an ideal balance between these two effects for selecting hand size rocks.

The stencil for filtering the hue image has a value of 1 in the center and -1 in the periphery. The stencil for the intensity image has values of 1 in the center and -1.5 in the periphery. These values were tuned to improve the final observed performance of the algorithm.

The images which are obtained as a result of filtering the raw hue and intensity images are called the saliency images, as they accurately locate the salient hue and intensity features in an image which correspond to a rock. The saliency image obtained by filtering the raw hue image in Fig. 4-5 is shown in Fig. 4-7. There is a 5 pixel wide peripheral ring in the image which does not contain any values. This is the image size reduction caused by filtering the image by a square mask of size 11. The actual saliency image thus only contains values in the inner 22x22 square grid.

4.2.4 Segmentation

In the saliency images, the values in the vicinity of the rock may be either higher or lower than the rest of the image. In order to determine the location of the rock, the correct segmentation of the image requires that we determine whether a blob is found in either the low values of the image or the high values.

In both the hue and intensity image, the maximum and minimum values are determined. The image is first segmented using a threshold at a low value to determine if a blob exists at near the lower extreme values. If a single connected patch exists in the low image which is less than a certain number of pixels in area, then we assume

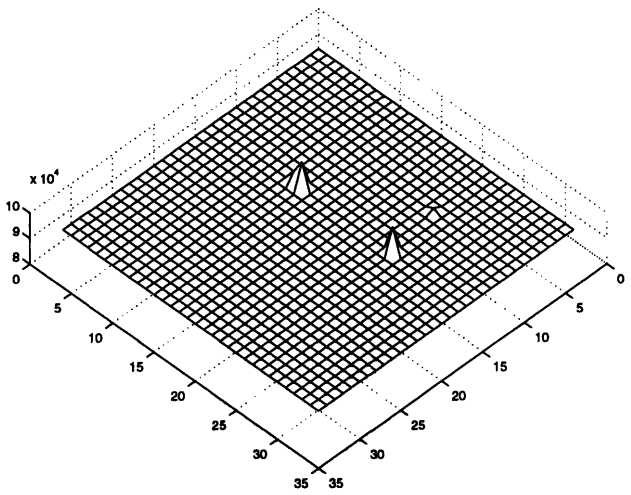

Figure 4-8: Result of segmenting the saliency image, by only considering the high values

that this is caused by a rock at that location. If however, several such patches exist then we conclude that the ground yields these patches and a rock to be detected, would be found in the higher extreme values of the image instead, so the second segmentation occurs using a threshold near the relative maximum of the saliency values in the image.

Fig. 4-9 and Fig. 4-8 show the low and high segmentation of Fig. 4-7. For our purposes, we have defined a desirable blob to be a large connected region of size greater than 50 pixels. In order to locate this blob, we compare the number of connected regions in the low and high segmented images. Notice that Fig. 4-8 has more small patches, whereas Fig. 4-9 has only one large connected region. Thus, the correct segmentation is indicated by the image with has only a single connected region of the correct size, which in this case is the low end segmentation. If on the other hand there are multiple connected regions in each segmentation then we conclude that there is no rock in the current image.

The segmentation thresholds at the high and low end are a function of the maximum S_{max} and minimum S_{min} saliency values in an image. In the low end segmentation all pixels that have a value lower than S_{min} + 100 are considered to be part of the figure, while all the other pixels form the ground. Similarly the high end segmentation is done by considering all pixels with value greater than S_{max} - 100 to be

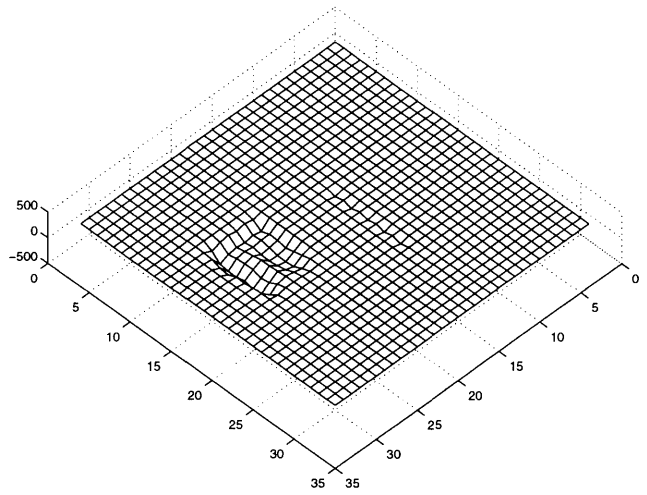

Figure 4-9: Result of segmenting the saliency image, by only considering the low values

part of the figure and the rest is ground.

Once the correct blob has been located in both the intensity and hue images the position of the rock along the vertical axis of the image is calculated by averaging the y-values of each pixel belonging to the correct blob in the hue and the intensity image.

4.3 Performance Analysis

The performance of the detection algorithm has been tested in the simulated rocky and sandy environments in the indoor MIT Mars yard. The tests were run on several rocks of different colors were collected from the New England coast, representing a wide variety of colors, brightnesses and textures. All these rocks were tested on both the rocky surface and sandy surface. The performance results are as follows.

A total of 8 rocks were tested. Each rock was laid on the rocky surface and then on a sandy surface. For each of these conditions the algorithm was run five times. The Table 4.1 shows what percent of the times the algorithm was run the rock was actually detected by the algorithm.

The overall algorithm thus currently has a false negative rate of 23.75%. While detection on rocky surfaces is close to perfect, the false positive rate is brought high

Rock	On Rocks	On Sand
Rock 1	100%	100%
Rock 2	60%	40%
Rock 3	100%	80%
Rock 4	100%	80%
Rock 5	100%	20%
Rock 6	100%	60%
Rock 7	100%	100%
Rock 8	100%	0%

Table 4.1: Percentage of rocks successfully detected on rocky and sandy surfaces

by the algorithms weakness in detecting several rocks on a sandy surface. In future algorithm, perhaps other modules besides hue and intensity would serve better to distinguish between rocks and the sandy soil.

The false positive rate for the algorithm was collected by running the algorithm on patches of ground which were sandy and rocky. The false positive rate for both the grayish rocky surface and the sandy soil was 0

Currently the algorithm has only been tested at the MIT Mars yard which is simulated natural environments. However, as it is indoors there are several things that are different. For example, the intensity of overhead lights is constant and unchanging. To work in the changing intensity of the sunlight, with lights and shadows, newer means of detection using information such as the proximal nature of rocks and their shadows could be added to make the detection more robust.

Chapter 5

Reaching

The last chapter described how a rock was detected in the local field of view. This chapter explains how this visual information is used to control the arm motor reflexes which result in reaching towards the rock. As discussed in the last chapter, the rock detection algorithm runs as the robot is navigating through the world. The first section in this chapter explains how control is transferred from the navigation to reaching behavior when a rock is detected. The second section describes the visually guided reaching strategy. The third section shows how the reach terminates when the ground is sensed. The final section is a performance analysis of this reaching strategy in terms of the proximity of the final position of the end effector with respect to the rock.

5.1 Transfer of Control to Arm

The primary behavior of the Pebbles robot is navigating while avoiding obstacles. This navigation system developed by Lorigo [31] is designed as a visual motor algorithm. The visual component which runs on the CVM grabs a wide field of view image and detects the difference in proximity between obstacles to the left and right of the robot. It then generates 4-bit tread motor velocity commands to cause the robot to turn away from the closer obstacles. These commands are sent to the 68332 via the serial port connection.

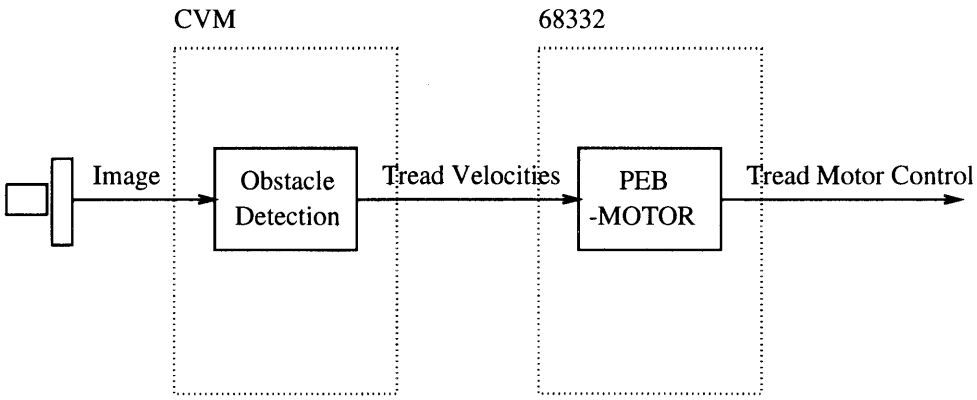

Figure 5-1: Schematic diagram of navigation system

The MARS and Humanoid environments in L, structure the use of multiple processes. A group of processes can be programmed as an *assemblage* where a common lexical environment and port interfaces are shared. On the 68332, the PEB-MOTOR assemblage is responsible for managing the tread motor velocity commands. Here the commands sent from the CVM are read. They are either assigned to the motors or ignored in special situations such as when the robot is stopped or stuck. Fig. 5-1 shows the navigation system.

For rock detection the CVM grabs a local field of view image, as described in Chapter 4. The output of the detection algorithm is a number between 0 and 32. A 32 indicates that a rock does not exist in the image. The numbers from 0 to 31 represent horizontal rows in the image, 0 being at the top and 31 being at the bottom. Therefore a number between 0 and 31 represents that a rock exists at that height in the image. This 6-bit number is also sent to the 68332 via the serial port.

Since the CVM is not a multi-processing system it executes an infinite loop in which both the visual avoidance and rock detection code are executed on every code cycle. Thus at every cycle two motor commands, and a rock position are sent along the serial port. The motor commands and rock position are intended to be received and used by separate processes on the 68332. The nature of the information sent in a packet must be identified so it can be used by the correct process. A packet of information thus consists of a 2-bit prefix which serves as an identification code for the following 6 bit data (Fig. 5-2).

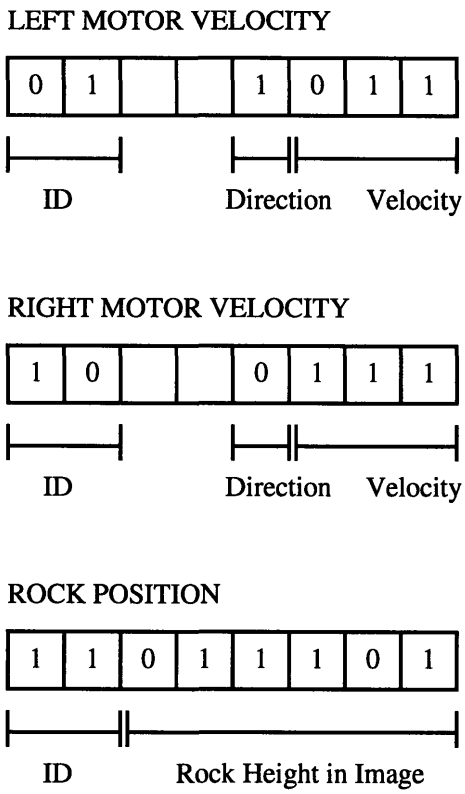

Figure 5-2: Packets sent from the CVM to the 68332

On the 68332 the communication with the CVM is mediated by the READ-SERIAL-PORT process which constantly awaits packets. Each of the three bytes received is stored in a port variable so it is accessible to all other processes in the system. Each of the agents which require the use of visual data have a 2-bit personal identification code. When these agents are run, they read the three current values of the READ-SERIAL-PORT port variables and use the data which has the matching ID.

As described in Chapter 3, the arm joint angle positions are controlled at a high level by the joint handler processes on the 68332. These processes are responsible for communicating the position setpoints and other force control parameters to the 6811 boards. However, the handler processes do not decide the actual values of the position setpoints. They simply receive set-point values from other processes and execute them.

The REACH assemblage is responsible for controlling the joint setpoints of the arm during navigation and reaching. When the system is first started, the navigation

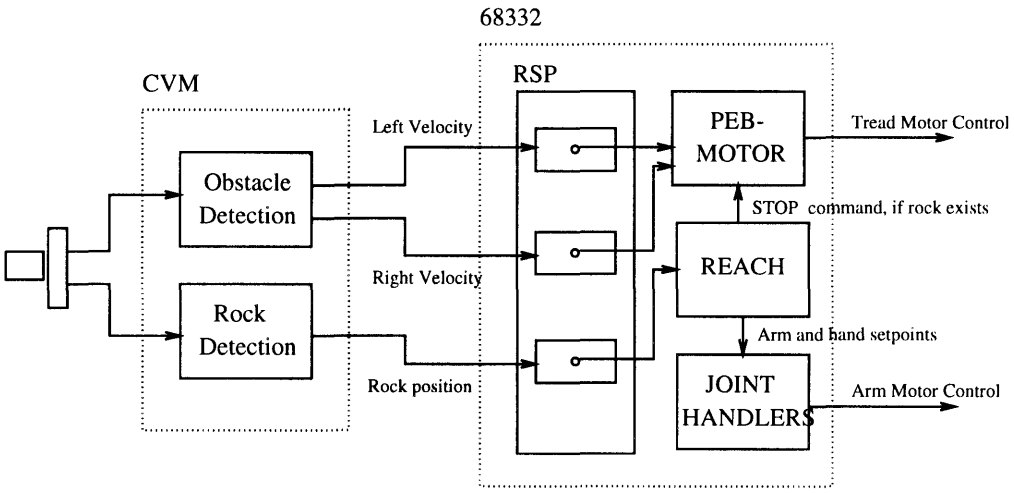

Figure 5-3: Organization of the REACH behavior

process sends non zero speed commands to the motors and the REACH assemblage sends setpoint commands to the handlers to hold the arm at the constant rest position. One REACH process constantly monitors the data from the rock detection algorithm. As long as the data is 63, indicating that no rock is seen in the image, the navigation continues. However, if a value is read between 5 and 58, the process immediately interrupts navigation by sending a *stop* command to PEB-MOTOR, and activates all the agents in the REACH assemblage. This organization is diagrammed in Fig. 5-3.

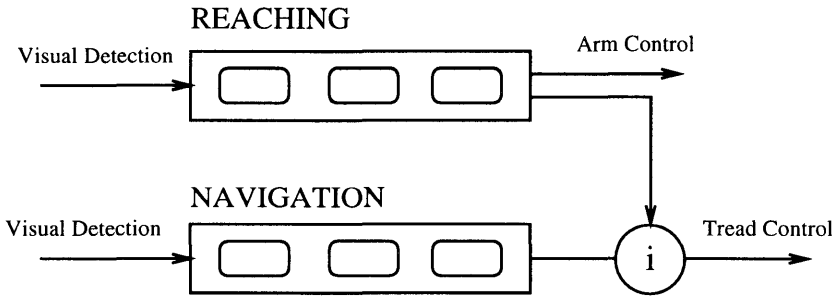

Figure 5-4: Transfer of control between the navigation and reaching behavior layers

This organization is a standard combination of behavioral layers using inhibition, in subsumption architecture, in which a higher behavioral layer inhibits the outputs of a lower layer. This high level view of the transfer of control using an Inhibit node is shown in Fig. 5-4.

This behavioral organization determines that as soon as a rock is seen in the local

field of view image the robot stops moving so the rock remains in the image. Since this is the only initial condition for the reaching algorithm, it can begin immediately. This transfer of control is natural and elegant due to the fact that the same visual field and detection algorithm are used for both rock detection during navigation and visual feedback during reaching. If separate algorithms were used there would be a disparity between the position of a rock computed by the two algorithms. If a rock was found by the detection algorithm there would be an additional step of positioning the robot correctly with respect to the rock for the visual feedback algorithm. Our organization circumvents this problem.

5.2 Visual Servoing

When the rock is detected the arm is at rest position as shown in Fig. 4-1. At this position the width of the local field of view corresponds to a patch of ground of width approximately 4 inches. The width of the hand is 3" so if the rock is of the size detected by the vision algorithm, approximately 1" in diameter, if the end effector is moved to position on the ground which corresponds to the center of the image, a grasp maneuver can contact and grasp the rock. The reach algorithm is thus only concerned with attaining the correct position of the end effector in the vertical direction with respect to the rock.

When the rotate joint and base joint are held at the rest position locations the movement of the lift joint in the lowering direction has an effect of moving the camera lower and therefore raising the rock in the image. The movement of the camera in the horizontal direction also lowers the camera and has the same effect of raising the rock in the image. On the other hand the movement of the elbow outward has an effect of raising the camera, and therefore lowering the rock in the image. This effect is shown in Fig. 5-5

The reaching is a visually reactive scheme in which each of the lift and elbow joints move independently of knowledge of other joints locations and merely react to the visual input. The motion of the joints is dictated by the following two tiered

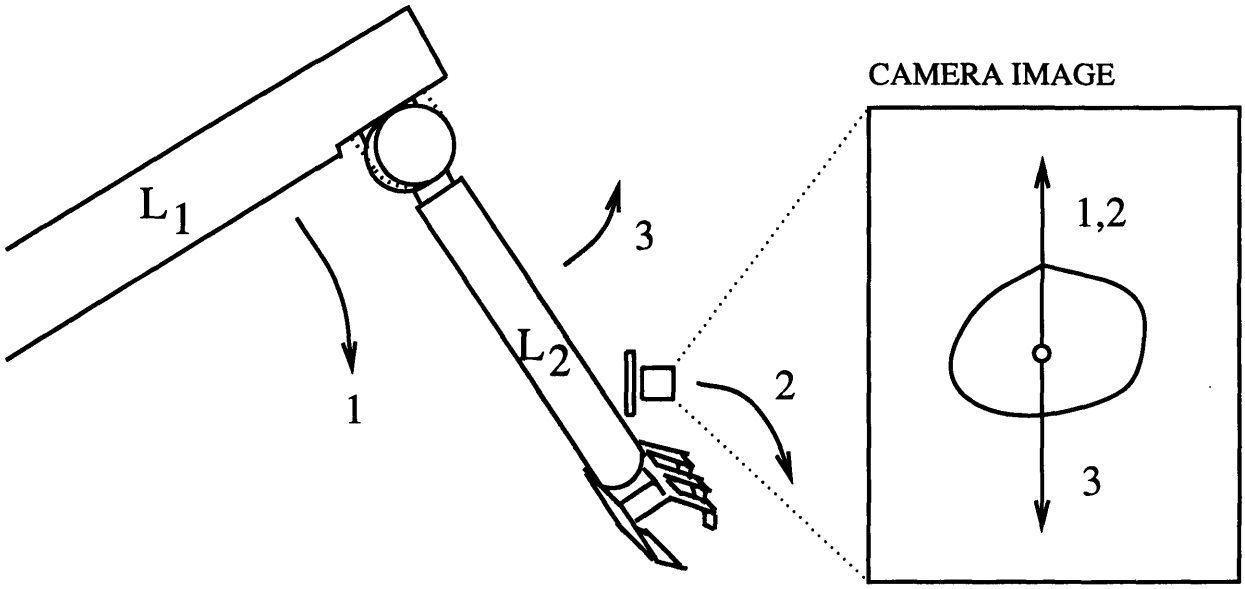

Figure 5-5: Effect of joint motions on the position of the rock in the image

scheme. The goal of the motion is to keep the rock centered in the image. The first layer is the MOVE layer. In this the lift joint moves lower by 3 setpoints and the camera joint moves lower by 2, if the rock is too low in the image. The elbow joint moves higher by 2 setpoints if the rock is too high in the image The move behavior is shown in Fig. 5-6).

However, since this is a multi-processing system and not all the processes are run in an ideal order, the situation may occur when the motion of either the elbow joint occurs before the camera and lift, or vice versa, causing the rock to disappear from the image altogether. When this occurs, control is transferred to the FIND layer.

The FIND agent is responsible for relocating the rock in the image. If the last position of the rock in the image before it disappeared was in the high zone, then the assumption is that the lift and camera joints which raise the rock in the image, overshot and so to compensate we move the elbow joint higher until the rock reappears. Similarly if the last seen position of the rock was low in the image then the assumption is that the lowering joint, overshot and so the compensating joints move to relocate the rock in the image. The motion of each of the joints in reaction to the vision is shown in Fig. 5-7.

The algorithm is purely reactive to vision and does not use current joint angle

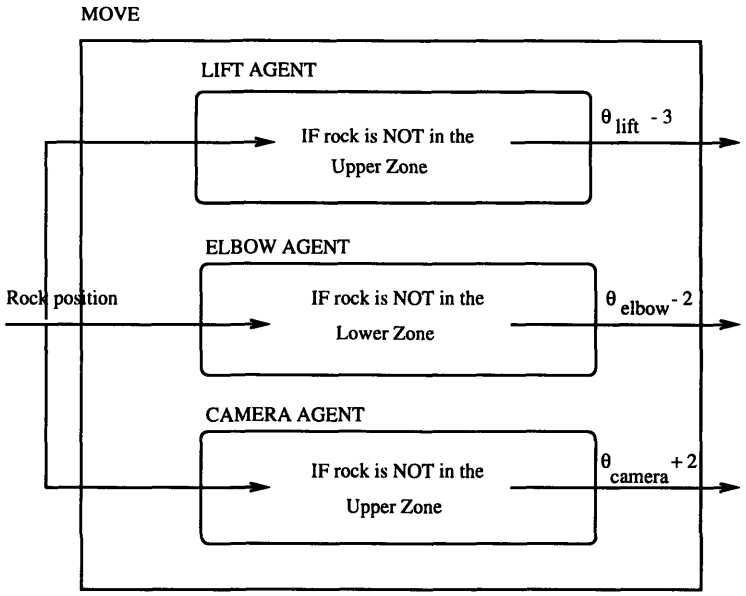

Figure 5-6: The organization of the MOVE behavior

values. The effect of the visual feedback is constant incremental change to the joint setpoint values and not a function of the arm position. Further, due to compliant control, the actual positions of the joints are a function both of the setpoints and the gravitational torques at each joint, so the algorithm does not directly control the arm position.

5.3 Sensing the ground

The FIND and MOVE reflexes continue to move the end effector closer to the position of the rock.

As the arm moves out and towards the rock, the end effector soon touches the ground at some point. From this point onward even though the lift joint keeps moving lower and the elbow joint setpoint is moved out the actual arm position itself is locked in place due to the friction of the end effector and the ground. It is at this point that as the lift pushes lower but the actual joint does not move that the strain on the torsional springs increases in a large way. Thresholding the strain reading for the two springs in the differential joint gives a virtual *ground-contact* sensor.

The ground touch process is an independent process, which constantly monitors

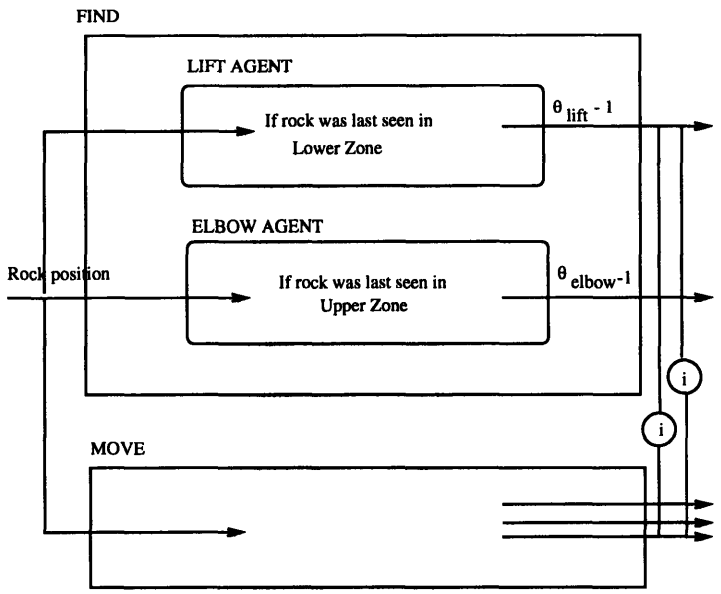

Figure 5-7: The two-layered organization of the visual motor reflexes in the REACH behavior. It uses the MOVE module shown in Fig. 5-6 as its lowest level.

the value of the ground-contact sensor. When contact with the ground is sensed increase the outputs of the reaching algorithm are inhibited and both the shoulder and elbow joints are powered off leaving the arm in the exact same physical position, due to friction.

This method of termination is only possible in natural ground environments where rocky or sandy surfaces provide high lateral friction. On polished artificial surfaces where the friction between the metal of the hand and the ground is severely reduced there may be slippage of the hand from the initial ground-contact position.

The termination strategy requires compliant arm control. The compliance allows contact with the hard surface of the ground without anticipatory changes. It also enables the algorithm to continue changing the desired setpoint of the joints after ground contact exploiting the fact that the force of lateral friction will act on the virtual spring joints to lock them in place.

5.4 Performance Analysis

The ideal performance of the reaching algorithm leads to an end effector trajectory which terminates at the location of the rock. There can be a wide range of variability in the shape of an ideal trajectory as the only critical point is the trajectory endpoint. The performance of the algorithm is best measured by the proximity of the trajectory endpoint to the rock, for trajectories generated under widely varying values of the initial condition parameters.

To determine which initial condition parameters influence the shape of the trajectory we must analyze how the trajectory emerges from the operation of the algorithm. The algorithm dictates that each joint move until the rock has been centered in the image. The actual degree to which each joint moves is dependent on the rate of change of the rock position in the image in response to joint motion. This function is different at each moment in time. The shape of the trajectory that emerges is directly determined by this function at each successive arm and camera position.

The rate of change of image features with the motion of the end effector is defined as the *Image Jacobian.* The relationship is described by the following equation:

$$\dot{f} = J_{\theta-camera}(r)\dot{r} \tag{5.1}$$

where $\dot{f}$ is the rate of change of the image features, $\dot{r}$ is the rate of change of the camera position mounted on the end effector, and $J_v(r)\epsilon\mathcal{R}^{kxm}$ is the image Jacobian matrix where k is the number of image features and m is the dimension of the task space in which the end effector moves. The Jacobian will be different for each different value of the camera angle position.

The rate of change of the end effector position is in turn dependent on the rate of change of the joint angle positions. They are related by the *Jacobian* matrix for the arm as follows:

$$\dot{r} = J(\theta)\dot{\theta} \tag{5.2}$$

In this equation $\dot{\theta}$ describes rate of change of the joint angle positions and $\dot{r}$ is the rate of change of the end effector position.

The final relationship between the rate of change of image features and the joint angle positions is obtained by chaining these two equations.

$$\dot{f} = J_{\theta-camera}(f(\theta))J(\theta)\dot{\theta} \tag{5.3}$$

where the end effector position r is a direct forward kinematic function of the joint angle positions.

$$r = f(\theta) \tag{5.4}$$

From these equations it is clear that the first condition which effects the final shape of the trajectory is the initial arm position. As described previously, the arm is always held in rest position at the beginning of the algorithm. However, this is not an exact position but some approximate position in which the camera can be pointed appropriately towards the ground. The first test of performance is to show that the algorithm continues to operate successfully despite varying initial position of the arm. The algorithm was tested at three different initial positions of the arm, which covered the usual range of variability in the rest position.

The second condition that influences $\dot{f}$, the rate of change of the rock height in the image, is the initial value of f. When the rover stops in front of a rock, the rock may be at any position in the image. To test that the algorithm worked correctly regardless of this initial parameter, the rock was tested at three different positions representing the range of possible initial positions in the image.

The camera position in every run is started at exactly the same angle and position with respect to the L2 joint. So although varying the camera angle would affect the visual Jacobian, this parameter is not varied in our tests since it is not changed in regular usage.

There were other sources of variability in the outcome of the algorithm which were not a direct result of the equations above. When the reach is initiated, the

rock is at some height in the image. This height is sent over the serial port to the READ-SERIAL-PORT process. The next time that the REACH processes which set the position of the arm are run, they respond to the position of the object in the image and according either the elbow joint, or the camera and lift joint, or all three are set to new set-point values. However, the physical position of the arm will not change until the handler processes are run again at which time the value is actually sent to the 6811 motor controllers and executed. Since the CVM code cycles are not synchronized with the 332 processes in any way, the next time the position of the rock in the image is calculated, may be before, after or even during the physical arm position change. Due to the independent and unsynchronized nature of scheduling processes, there is a large amount of variability in the actual trajectory of the arm that unfolds. Even if the arm, camera and object are in exactly the same position when the reach is initiated, the emergent trajectory is slightly different each time.

In order to test the robustness of the algorithm we vary each of the parameters which influence the arm trajectory for a comprehensive test of the algorithm performance. The initial arm position is varied among three positions. In each such condition three different positions of the rock in the image are tested. Finally, each identical condition is tested four times to show that the algorithm is robust with respect to process scheduling variability.

This gives us 3x3x4 data points. At each of these data points the success measure was single real number representing the distance of the end effector from the rock. Since the lateral placement of the end effector is not a concern the success measure is measured as the vertical distance between the parallel horizontal axis through the center of the hand and the proximal surface of the rock. The same rock was used in the same orientation each time.

Table 5.1 shows joint angles of the lift and elbow joints of the arm in Arm Positions 1-3 used in Table 5.2. In Table 5.2, the results of the algorithm are shown. In each box there are four numbers which represent the error in millimeters for each of the four identical trials.

Arm Position 1 represents the rest position of the arm during normal operation.

	θ_{lift}	θ_{elbow}
Arm Position 1	0°	10°
Arm Position 2	10°	20°
Arm Position 3	20°	30°

Table 5.1: The joint angles of the three arm positions used for testing the reaching algorithm

Rock Position	Arm Position 1	Arm Position 2	Arm Position 3
Low	0	0	19
	0	0	21
	0	missed	133
	0	0	15
Center	0	2	135
	0	2	21
	0	0	12
	0	11	151
High	6	5	missed
	1	0	39
	0	13	24
	0	14	22

Table 5.2: This table shows the distance of the end effector from the position of the rock at the end of each reach trajectory. Each measurement is in millimeters.

As seen in Table 5.2 In this location a rock that is detected at any location in the image is almost always reached successfully or within a very close range.

While the error between the end effector and the rock increases in Arm Position 2, the rock is still well within reach of the grasping maneuver so this is reasonably successful as well.

The algorithm does not fair as well in Arm Position 3 in which both the L_1 and L_2 joints are outstretched. When the links are outstretched the lift joint experiences a much larger torque. Since the joints are controlled compliantly, the larger torque determines that at the same setpoint position of the lift joint, the actual position of the joint will be much lower in Arm Position 3 as opposed to Arm Position 1. This causes the trajectory to veer prematurely towards the ground in Position 3. In a future implementation this problem can be dealt with by decreasing the joint position increment of the lift joint based on the torque sensed.

Chapter 6

Grasping

The following chapter will present the grasping algorithm which has been designed as the next layer of behavioral competence for sample collection. The first section will present the design of the grasping algorithm. The second section will discuss the method of gauging a successful or unsuccessful grasp. The third section will present a performance analysis of the grasping algorithm independent of reaching.

6.1 Picking up a rock

At the end of the reaching behavior the the REACH assemblage goes into ground mode which inhibits the algorithms further effect on the position of the arm. The arm lift and elbow joints go are power-off so the arm remains in the same position on the ground due to gravity and friction of the ground.

The assemblage responsible for picking up a rock is called the GRASP assemblage. All through navigation and reaching this assemblage is dormant. There is a single agent in this assemblage which listens to the mode of the reaching algorithm. When the REACH assemblage goes into ground mode this assemblage becomes the active assemblage and takes control of the positions of the joints.

Unlike in the REACH behavior, where multiple processes are responsible for a group of sensory motor reflexes, the GRASP behavior is composed of a single dominant process which influences the position of the arm and hand. The process moves

the arm and hand in a fixed sequence of motions, in the absence of sensory feedback. This algorithm relies on the successful completion of the reach where the hand is positioned at a constant location with respect to the rock. The fixed nature of the initial condition for the grasp simplifies the problem, so a non sensor based algorithm can be used. However, in a future version, the reliability of the grasp behavior could be improved by including more sensory feedback to relax the initial condition constraints. Then the grasping behavior could correct for small errors in reaching.

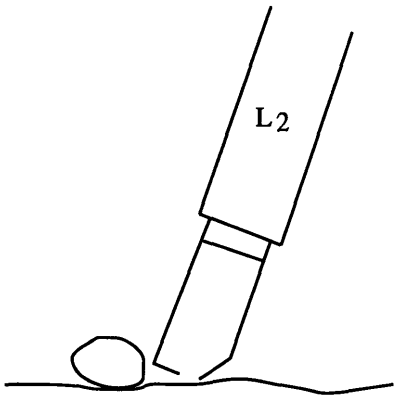

Figure 6-1: The Arm and Hand are at position P_1 at the beginning of the grasp sequence

When the REACH assemblage has gone into *ground* mode, the arm is powered off. The joint angle positions of the arm in this condition, P_1 can be assumed to be those that result in the end effector being in the correct position with respect to the rock (Fig. 6-1). These joint angle positions are recorded as they describe the location of the rock on the ground in terms of joint angles.

At the beginning of the grasp, the desired joint setpoints are set to be the actual positions recorded in the previous step. Then the arm is power-up again and put into *spring* mode. This is done so that when the arm goes into spring mode there is no discrepancy between the desired position and the actual positions of the joints so there is no sudden force. This provides a smooth startup for the arm, without moving it from its current location.

During reaching, the camera can reach a maximum camera angle of up to 90 degrees, with respect to the axis of the L_2 joint. At this position of the camera,

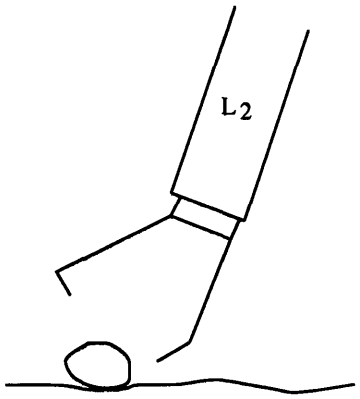

Figure 6-2: At position P_2 the gripper is preshaped and the Arm is positioned above the rock.

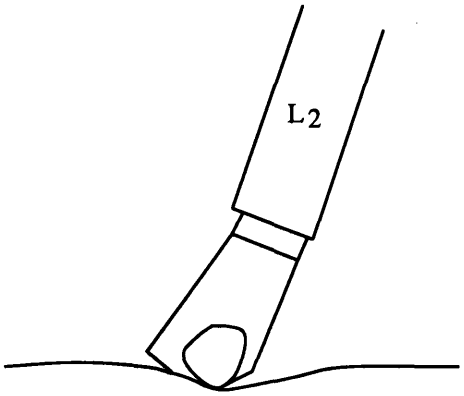

Figure 6-3: The arm and hand at grasp position P_3.

unless the hand is completely closed, it is detected as an object in the bottom of the image and is mistaken to be the rock. Thus, the eye-on-hand configuration does not allow preshaping of the gripper.

The next step is to preshape the gripper. In order to open the gripper, the arm must be lifted so that it opens in the air, instead of scraping along the ground and pushing the rock out of the way. This is done by lifting the shoulder joint by 60 setpoints. Then the gripper is opened to the widest extent possible. This is described as position P_2 (Fig. 6-2).

In the third step, the lift joint is lowered again to the setpoint position in P_1 and the gripper setpoint is set to the closed position. In this position P_3, due to the constant force control of the gripper joint, the gripper will only close until contact with an object provides an equal opposing force to prevent further closing. If a rock

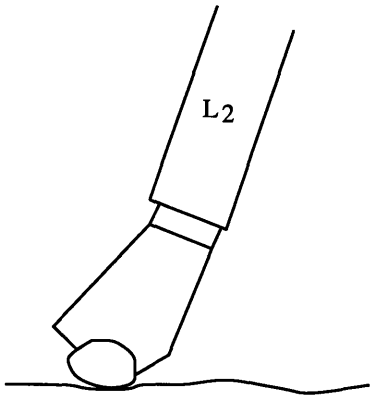

Figure 6-4: The arm and hand at grasp position P_3.

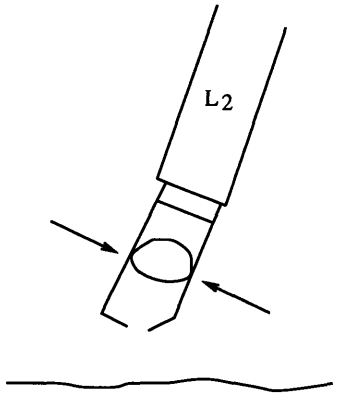

Figure 6-5: The arm and hand at grasp position P_4.

exists, this can lead to two potential outcomes based on the shape, size and orientation of the rock and the angle of the vertical axis of the hand with respect to it. In the first case, shown in Fig. 6-3, the rock may be within the gripper, but the gripper is prevented from closing further due to friction with the ground. In the second case, shown in Fig. 6-4, contact point on the surface of the rock provide the opposition to further closing motion.

In the final step, the lift joint is raised back to the same setpoint position as in position P_2. This is position P_4 as shown in Fig. 6-5. In the case in which the gripper motion is stopped by contact with the ground, as soon as the arm lifts off the ground, the gripper closes in further until it makes contact with the rock.

In the final step, the lift joint is raised back to the same setpoint position as in position P_2. In the P_3 situation described by Fig. 6-4, where contact points with the rock were made on both of the parallel jaws of the gripper, the contact forces

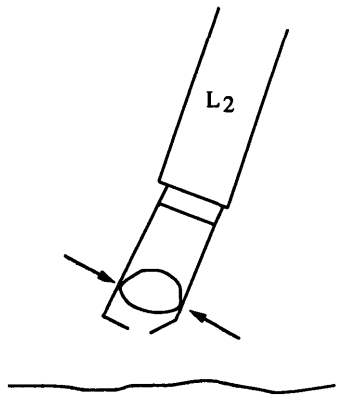

Figure 6-6: The rock has slipped from its original contact position and has been grasped at a lower position.

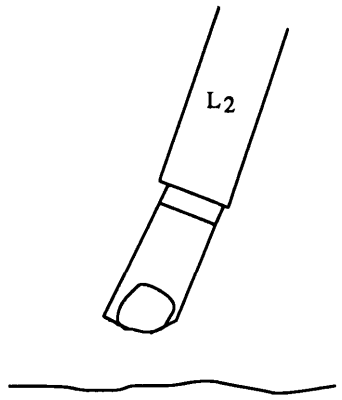

Figure 6-7: The rock has slipped down all the way to the bottom without making contact with the parallel plates and lies in the enclosure between them

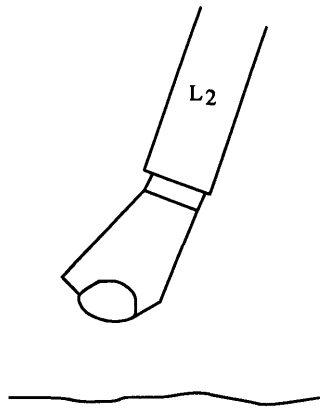

Figure 6-8: The rock has slipped down all the way to the bottom without making contact with the parallel plates and lies in the enclosure between them

normal to the surfaces of contact may or may not be enough to counteract the weight of the rock as the arm lifts. If the forces are large enough the rock stays in the same position within the gripper as the arm is lifted as shown in Fig 6-5. If the forces are not strong enough, the rock starts slipping lower within the gripper. As the rock slips, the contact points no longer oppose the closing motion of the gripper. The gripper closes in further until contact with the rock is made again at a lower position as in Fig. 6-6. A less likely scenario may be that the rock has slipped down all the way without making contact and lies at the bottom of the enclosure between the two parallel plates (Fig. 6-7).

In the P_3 position in which the gripper motion was opposed by contact with the ground, as soon as the arm lifts off the ground, the gripper closes in further until it makes contact with the rock. This usually results in a grasp as shown in Fig. 6-8, although it may although result in the grasps described previously by Figs. 6-5, 6-6 or 6-7.

6.2 Verification of Grasp

In each of the grasp scenarios described above in which the rock prevents the gripper from closing completely the grasp can be verified by checking the final position of the gripper after it has lifted off the ground. If the gripper position is greater than 0, then it has not closed completely. This indicates the existence of a rock in the gripper. However, in the case where the gripper position is 0, it cannot be immediately assumed that a rock does not exist, as it may be a situation as shown in Fig. 6-7.

The second form of verification uses the compliance of the lift joint to sense the additional weight of the rock. In position P_2 the hand is empty. The desired setpoints of the arm in both position P_2 and P_4 are identical. However, if in position P_4 there is a rock in the hand, the additional torque on the lift joint due to the additional weight at the end effector will cause the actual position of the lift joint to be lower than it was in P_2. If such a difference in position between the actual joint angles is sensed, then it is verified that a rock has been picked up.

Both these forms of verification require that the rock be in the gripper. If a rock is not detected in the gripper, then the two possible explanations are that the rock was dropped or that vision algorithm detected a false positive and a rock did not exist in the first place. In the current implementation, since the rock would need to be in the gripper for its existence to be verified, the only way to do this is to retry the grasp maneuver. However, this has small chances of succeeding since the first grasp maneuver most likely displaced the rock from its original location. Even if the entire reach sequence was to be attempted again, there is no guarantee that the rock would appear in the local visual field after the displacement. Thus, this implementation does not retry the grasp if it has failed. In future versions using haptic feedback on the gripper surfaces would make it simple to verify that a rock existed, in which case the grasp could be attempted again.

6.3 Performance Analysis

A successful grasp is indicated by the outcomes shown in Fig. 6-5, 6-7 and Fig. 6-8.

To measure the performance of the algorithm the grasp was tested in several different positions of the arm, and with several different rocks in different orientations. The success rate of the grasp maneuver was approximately 20%. The grasping strategy had problems grasping rocks that were of a flat shape and laying on the ground in a flat orientation. Since the approach of this grasp was from above, the parallel jaws landed on top of the flat surface and closed without being able to make proper contact with the rock.

While the grasp faired better on rocks that were closer to spherical shape, an aspect of the motion caused problems in these situations as well. In the grasp, the transition between the P_2 and P_3 positions was implemented as a direct change in setpoint of the lift joint between the two positions. This direct change caused a high speed motion of the lift joint, causing the hand to have a hard impact with the ground. This had two unwanted effects in that it causes an indentation in the ground where the rock lay, causing the rock to become more deeply lodged in the ground. Also

if the hard impact occurred with the surface of the rock it caused the rock to move away from its current location near the hand, making it nearly impossible to enclose.

This implementation led to key insight about using compliant control on natural surfaces. While, the compliance makes it safe for the arm to impact the ground without knowing its actual physical location, any algorithm which depends on knowing the position of an object on a natural surface should try to make as little contact with the ground as possible. Any contact with the ground causes shifting of the soil and rocks which may propagate position errors into the ensuing task. In the reaching and grasping behaviors implemented here, there were two points in which strong impact with the ground was made. The first was when the virtual ground sensor at the lift joint was employed to sense the ground. This sensor depended on the fact that a large torque was felt on the strain gauges of the shoulder. This situation demanded that the arm press strongly against the ground. The second strong impact was made in the transition between the P_2 and P_3 positions in the grasp sequence as discussed above. Since both these impacts occurred after the visual feedback stopped having an effect on joint motion, there was no way to correct for position errors produced due to them, which proved lethal in this implementation.

In the future, this problem could be solved by making alterations to these parts of the algorithm. The hard impact could be largely avoided by changing the direct set-point modification into a smooth motion with a bell shaped velocity profile, between the two positions. Moreover, instead of relying on the lift strain gauges to provide a virtual ground contact sensor, a much more sensitive haptic sensor mounted on the lower surfaces of the hand could be used to sense the ground as soon as gentle contact was made. This would prevent further movement of the rock and soil.

Chapter 7

Conclusion

7.1 Review of Thesis

This thesis described the implementation and operation of a visually reactive reaching and grasping strategy for the Pebbles robotic manipulator. The implementation consisted of a visual rock detection component, a behavior based reaching strategy and a grasp maneuver, which were layered in a subsumption architechture design to produce the behavior.

The key feature of the rock detection algorithm was that the detection was a multi-module algorithm based on color and intensity differences between the rock and the ground. There was no use of prior knowledge of visual characteristics of the ground.

The arm control was novel in that it employed compliance. This allowed the reaching strategy to be conceived in a new way in which the end point did not have to be determined before it was physically reached. Sensory feedback could be used to terminate the reaching behavior instead of prior knowledge of the location of the ground.

The reaching was based on a single vision sensor mounted near the end effector. Although similar configurations have been used previously, what was novel about Pebbles was that the camera had an extra rotational degree of freedom independent of the arm motion. The reaching strategy was thus unique as it was based on both

the independent effect of the camera motion on the visual image as well as the effect of arm motion on vision.

The reaching behavior was an emergent property of the interaction between the independent motions of the camera and each arm joint based only on visual feedback. No explicit communication was used between the three independent sensory motor loops in order to guide the arm. This made the implementation simple and computationally efficient while being robust under varying conditions.

7.2 Further Work

In this implementation the vision algorithm was composed of two modules using color and intensity. These were sufficient for successfully detecting rocks on the sandy and rocky surfaces at the indoor MIT Mars yard. However, in outdoor environments there is a larger variability in the hue and intensity of the ground characteristics as well as lighting conditions which may call for the addition of further modules to make the algorithm robust.

In the current implementation the local visual field is a small section of the reachable ground in front of the robot. In a future version it would be desirable to make the visual field for rock detection larger so that the robots visual observation of the environment would cover a larger area. Making the visual field larger would mean that control of motion of the arm in the lateral direction would become critical. A behavior using the rotate and base joints layered on top of the current reaching behavior in the vertical direction could be used to move the arm in this larger work space.

This implementation made a first attempt to solve visually reactive servoing for an eye-on-hand system using a purely behavior based scheme. It used two arm joints and a single camera joint. Future work should focus on determining if such a method of arm control scales well to more complex manipulators using more degrees of freedom and to tasks in which the target object lies in an entire 3D workspace.

Bibliography

[1] P. K. Allen, A. Timcenko, B. Yoshimi, and P. Michelman. Automated tracking and grasping of moving objects. *IEEE Trans. on Robotics and Automation*, 9(2):152–165, April 1993.

[2] Y. Aloimonos. *Active Vision as as Methodology.* Lawrence Erlbaum Associates, 1993.

[3] D. H. Ballard. Animate vision. *Artificial Intelligence*, 48:57–86, 1991.

[4] D. H. Ballard and C. M. Brown. *Computer Vision.* Prentice-Hall, Englewood Cliffs, NJ, 1982.

[5] A. Bicchi, J. K. Salisbury, and D. L. Brock. Contact sensing from force measurements. *The Intl. J. of Robotics Research*, 12(3):249–262, June 1993.

[6] D. L. Brock. *A sensor based strategy for automatic robotic grasping.* PhD thesis, Massachusetts Institute of Technology, Department of Mechanical Engineering, 1993.

[7] R. A. Brooks. *The L Manual.* IS Robotics, Cambridge, MA, 1994.

[8] R. A. Brooks. *MARS: Multiple Agency Reactivity System.* IS Robotics, Cambridge, MA, 1996.

[9] R. A. Brooks, J. Bryson, and M. Marjanovic L. A. Stein M. Wessler. *Humanoid Software.* MIT Artificial Intelligence Laboratory, Cambridge, MA, 1996.

[10] J. F. Canny and K. Y. Goldberg. A risc paradigm for industrial robots. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, Atlanta, GA, May 1993.

[11] J. H. Connell. A behavior-based arm controller. AI Memo, No. 1025, 1988.

[12] R. C. Dubes, A. K. Jain, S. G. Nadabar, and C. C. Chen. Mrf model based model algorithms for image segmentation. In *Proc. of 10th Intl. Conf. on Pattern Recognition*, pages 808–814, Atlantic City, NJ, 1990.

[13] J. O. Eklundh, H. Yamamoto, and A. Rosenfeld. A relaxation method for multispectral pixel classification. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-2:72–75, 1980.

[14] B. Espiau. Effect of camera calibration errors on visual servoing in robotics. In *Proc. 3rd Intl. Symposium on Experimental Robotics*, Kyoto, Japan, October 1993.

[15] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8(3):313–326, June 1992.

[16] J. T. Feddema and O. R. Mitchell. Vision-guided servoing with featre-based trajectory generation. *IEEE Trans. on Robotics and Automation*, 5(5):691–700, October 1989.

[17] C. Ferrell. Robust agent control of an autonomous robot with many sensors and actuators. Masters thesis, Massachusetts Institute of Technology, 1993.

[18] J. D. Foley, A. Van Dam, S.K. Feiner, J. F Hughes, and R. L. Phillips. *Introduction to Computer Graphics.* Addison-Wesley, 1986.

[19] D. Geman, S. Geman, C. Graffigne, and P. Dong. Boundary detection by constrained optimization. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12:609–628, December 1990.

[20] K. Y. Goldberg. *Stochastic Plans for Robotic Manipulation.* PhD thesis, Carnegie Mellon University, Pittsburg, PA, 1990.

[21] R. M. Haralick and L. G. Shapiro. Image segmentation techniques. *Computer graphics and image processing*, 29:100–132, 1985.

[22] K. Hashimoto, T. Kimoto, T. Ebine, and H. Kimura. Manipulator control with image-based visual servo. In *Proc. Intl. Conf. on Robotics and Automation*, pages 2267–2272, Sacramento, CA, 1991.

[23] G. Healey. Segmenting images using normalized color. *IEEE Trans. on Systems, Man and Cybernetics*, 22:64–73, 1992.

[24] B. K. P. Horn. *Robot Vision.* The MIT Press, 1986.

[25] S. C. Jacobsen. The utah/mit dextrous hand: Work in progress. In *Proc. of the First Intl. Symp. on Robotics Reseach*, 1984.

[26] W. Jang and Z. B. Feature-based visual servoing of an eye-in-hand robot with improved tracking performance. In *Proc. Intl. Conf. on Robotics and Automation*, 1991.

[27] G. J. Klinker, S. A. Shafer, and T. Kanade. A physical approach to color image understanding. *Intl J. of Computer Vision*, 4:7–38, 1990.

[28] A. J. Koivo and N. Houshangi. Real-time vision feedback for servoing robotic manipulators with self tuning controller. *IEEE Trans. on Systems, Man and Cybernetics*, 12(1):134–142, February 1991.

[29] J. Liu and Y. H. Yang. Multiresolution color image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(7):689, July 1994.

[30] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293, 1981.

[31] L. M. Lorigo, R. A. Brooks, and W. E. L. Grimson. Visually-guided obstacle avoidance in unstructured environments. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, pages 373–379, Grenoble France, September 1997.

[32] Brian Scassellati M. Marjanovich and Matthew Williamson. Self-taught visually-guided pointing for a humanoid robot. In *Proc. of Fourth Intl. Conf. on Simulation of Adaptive Behavior*, number 4, pages 35–44, Cape Cod, MA, 1996. MIT Press.

[33] M. J. Mataric. *Interaction and Intelligent Behavior.* PhD thesis, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, 1994.

[34] Y. Matsuoka. Embodiment and manipulation learning process for a humanoid hand. Masters thesis, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, May 1995.

[35] G. Metta, A. Oddera, M. Salganicoff, and G. Sandini. A direct approach to vision guided manipulation. In *Proc. ICAR-93*, Tokyo, Japan, 1993.

[36] R. Muzzolini, Y. H. Yang, and R. Pierson. Multiresolution texture segmentation with applications to ultrasound images. *IEEE Trans. on Medical Imaging*, 12(1):108–123, 1993.

[37] R. Ohlander, K. Price, and D. R. Reddy. Picture segmentation using a recursive region splitting method. *Computer Graphics and Image Processing*, 8:313–333, 1978.

[38] N. P. Papanikolopoulos and P. K. Khosla. Visual tracking of moving target by a camera mounted on a robotl a combination of control and vision. *IEEE Trans. on Robotics and Automation*, 9(1):14–35, March 1993.

[39] G. A. Pratt and M. M. Williamson. Series elastic actuators. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1:399–406, July 1995.

[40] W. K. Pratt. *Digital Image Processing.* Wiley Interscience, New York, second edition, 1991.

[41] Brooks R. A. A robust layered control system for a mobile robot. *IEEE J. of Robotics and Automation*, RA-2:14–23, April 1987.

[42] A. S. Rao and K. Y. Goldberg. Grasping curved planar parts with a parallel jaw gripper. Technical Report 299, University of Southern California, Institute of Robots and Intelligent Systems (IRIS), Los Angeles, California, August 1992.

[43] A. S. Rao and K. Y. Goldberg. Shape from diameter: Strategies for recognizing polygonal parts. Technical Report 292, University of Southern California, Institute of Robots and Intelligent Systems (IRIS), Los Angeles, California, 1992.

[44] J. K. Salisbury and J. J. Craig. Articulated hands: Force control and kinematic issues. *Intl. J. of Robotics Research*, 1(1), 1982.

[45] M. Teichmann and B. Mishra. Reactive algorithms for grasping using a modified parallel jaw gripper. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, volume 3, pages 1931–1936, 1994.

[46] T. Uchiyama and M. Arbib. Color image segmentation using competitive learning. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(12):1197, December 1994.

[47] L. E. Weiss, A. C. Sanderson, and C. P. Neuman. Dynamic sensor based control of robots with visual feedback. *IEEE J. of Robotics and Automation*, RA-3(5):404–417, October 1987.

[48] D. E. Whitney. Historical perspective and state of the art in robot force control. *J. of Robotics Research*, 6(1), 1987.

[49] M. M. Williamson. Postural primitives: Interactive behavior for a humanoid robot arm. *Proceedings of Fourth Intl Conf. on Simulation of Adaptive Behavior*, 1:124–131, September 1996.

[50] A. P. Witkin. Scale space filtering: A new approach to multi-scale description. In S. Ullman and W. Richards, editors, *Image Understanding*, pages 79–95. Ablex Publishing, NJ, 1984.