Reactive Control of a Mobile Manipulator using Pseudo-joint Damping

Keith R. Ward[1] and Ronald C. Arkin

Mobile Robot Laboratory
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280

Abstract

Reactive control for mobile manipulation involves a tight coupling of sensors to motor response. Pseudo-forces exerted on the end-effector and limbs of the mobile manipulator must be distributed over the entire structure. Pseudo-joint damping provides a mechanism for this process.

Damping functions were used to create motion of the end-effector which emulated the characteristics of biological systems. The target biological characteristics were a general bell-shaped curve of the end-effector speed profile, an initial and final end-effector speed of zero, and a scaling of the speed profile with distance to goal. The damping functions were then evaluated for time, distance, safety, and flexibility to determine if the biological characteristics corresponded to improved performance of the mobile manipulator.

The simulations showed that by reproducing the characteristics of biological systems, improved performance of the mobile manipulator was realized. The set of linear and square damping functions were found to perform best in a series of varying environments. Both damping functions were shown to produce different ending configurations of the arm, and to be more or less aggressive by varying the damping parameters in the functions. This allows the performance of the mobile manipulator to be adjusted based on any *a priori* information about the environment.

1. Introduction

An approach to deal with an unmodeled or changing environment which can react in real time is needed to perform in a truly Flexible Manufacturing System. The robot will need to avoid unmodeled obstacles and be able to approach a target object even if the goal has changed locations.

The approach used in this research employs reactive control for a mobile manipulator that consists of an integrated mobile robot base and robotic arm. The path the robot takes to reach a goal is heavily dependent on the feedback of sensors during the movement. A

[1]Keith Ward's current address is VP/US6, Michelin Tire Co., P.O. Box 19014, Greenville, SC 29602.

complete planned trajectory is never formed. Instead, the reactive controller reacts at each point in time based upon current sensor readings, continuously updating its motion. The result is that the mobile manipulator can react in real-time to incoming sensor data in a changing world.

The mobile manipulator used in this research consists of a Denning MRVII mobile robot base and an integrated CRS A251 robotic arm (Figure 1) The base is a three wheel configuration with a ring of 24 ultrasonic sensors covering 360 degrees and shaft encoders for recovering distance traveled, direction, speed, and acceleration. The CRS arm has five degrees of freedom including waist, shoulder, elbow, wrist pivot, and wrist rotate. Shaft encoders are used for joint position, speed, and acceleration. The end-effector is a two-finger open/close gripper with an added force-torque sensor between the wrist and gripper used during interaction with the part being manipulated.

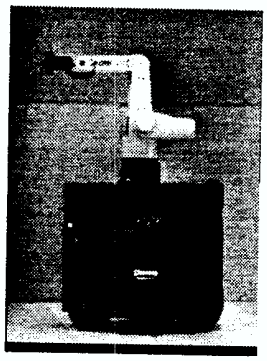


Figure 1: Mobile Manipulator

2. Related Work

There has been considerable research in the general area of mobile manipulation. This section is not intended to be an exhaustive compilation of this research, but rather is a representative selection of different approaches.

Control strategies for a mobile manipulator can be divided into two general categories. The first approach involves complete trajectory planning while the second involves reactive control or dynamic path planning. Complete path planning assumes that the geometry of the robot and the environment are known beforehand and do not change during execution

(e.g., [9,19,22]). The task is to define a complete path based on information available before any movement has occurred, and then to follow it. Emphasis is placed on optimization. The drawback of this approach involves the assumption that the world is static. If the environment changes, the path has to be re-formulated based on the new constraints. As trajectory planning based on well-founded *a priori* knowledge is a different problem than the one addressed in this research we will not further review this area.

The second approach, dynamic path planning or reactive control, assumes that the environment is not completely known or that changes in the environment occur during movement of the mobile manipulator. The controller reacts to the environment for an incremental movement, updates the constraints of the environment using sensing, and reacts again. Although the use of reactive control techniques in mobile robot navigation is now widespread (e.g., [2,6,12,14,20], there are few instances of this technique applied to the mobile manipulator problem.

Reactive control holds paramount the ability for a mobile manipulator to create a path on-the-fly. This approach does not consider the robot omniscient, where everything about the world is well known. The path the robot takes is determined by moving into the environment, sensing what is necessary, and reacting accordingly. The robot then senses again and responds appropriately.

In [11], Connel uses a subsumption-based architecture to control a mobile manipulator. The mobile manipulator, consisting of an arm mounted on a mobile base, is given the task of collecting soda cans in a cluttered, changing environment. A collection of fifteen modules interact through nodes to provide the necessary behaviors. The modules can be grouped into "levels of competence" (e.g., Cradle, Grip, Path, Park, Skim). As required during the task, different modules dominate and force certain behaviors to be active. Characteristic of subsumption architectures is that only one of the competing modules has control of the node at a time (arbitration).

A drawback to this system is apparent in the motion of the mobile manipulator. The motions of the base and the arm compete for control, therefore concurrent movement is not possible. Thus when approaching a can to be retrieved, the base travels to the area where the can is located and then parks. After the base is parked, the arm reaches out for the can. Only after the can has been retrieved and the arm motion stops will the base move.

Biological studies have also contributed to the control of mobile manipulators. Models drawn from this type of research are currently being studied as control systems and can serve as the basis for emergent behavior. Some of the studies which influenced our research follow.

Bizzi et al [5], studied the central nervous system of a frog by surgically disconnecting the spinal cord from the brainstem and stimulating different areas of the spinal cord. The result of the experiment was a mapping of the forces exerted by various positions of the leg for each different area of brainstem stimulation. They showed that for each location on the spinal cord, a unique potential force resulted which converged at one point; one position of

the leg was at equilibrium. Further experiments combined the stimulation of two areas on the spinal cord, and showed that the movement corresponded to a vector addition of the two potential fields.

Georgopoulos [15] has performed studies on the path and trajectory of reaching in humans. He explains that the inverse kinematics for unrestrained three dimensional arm movements are intractable [24]. His studies are concerned with the two dimensional case of a human reaching out to a goal using the arm and shoulder only. He characterizes the movements as being composed of two parts: an initial large amplitude movement that approaches the goal, and a subsequent smaller amplitude movement that brings the hand to the final position. The initial movements primarily use the elbow and shoulder joints, while the second movement uses the wrist and fingers to orient the hand to grasp the goal.

These two characteristic movements can be labeled as ballistic and micro-manipulation respectfully. Note that movement does occur with the wrist and fingers in the ballistic movement when the goal is to be grasped. In this case, a gross preshaping of the hand occurs before the final accurate positioning in the micro-manipulation. There is a trade-off between speed and accuracy which explains the difference between the two movements. More accurate movements require slower speeds, and conversely, faster movements are less accurate. Therefore, in the ballistic movement where the general location of the goal is being approached, less accurate but faster movements are used. In contrast, when the general area of the goal is reached, more accurate and thus slower movements are used for exact positioning of the hand.

In Georgopoulos' work, all targets are stationary and the goal is not to grasp the target, but to point to the target. Goodale [16] took the study of arm movements one step further by moving the target during the motion of the arm. Again the goal of the subject was to point to the target, not to grasp it. He showed that when the target was perturbed during the motion of the arm, the movement distance increased by the amount that the target was perturbed. A smooth transition occurred to the new trajectory (no secondary accelerations) showing that initial movement completion was not necessary before implementing a new trajectory. The study was done in the dark, using LED's as targets so that visual feedback of hand position was prevented.

Not considered by either Georgopoulos or Goodale was the actual grasping or interaction of the hand with the target. All motions occur in free-space. In grasping and mug placing Arbib et al [1] show that there is a final phase where the subject uses tactile and force feedback. Arbib and Hoff [17] further detail arm movements and their characteristics. Ballistic movement is defined as feedforward where the whole movement is based on the initial sample of environmental variables. Continuous feedforward control is differentiated by the adjustment of the control signal based on the continuous sampling of relevant environmental parameters. By these definitions, arm movements must be controlled by a continuous feedforward method to allow a novel trajectory to begin before the initial trajectory is completed.

Arbib and Hoff also cite MacKenzie et al [18] in characterizing the effect of accuracy demands on the speed profile of the hand. MacKenzie shows that the motion of the hand is characterized by a bell-shaped speed profile. The profile consists of an initial acceleration followed by a deceleration. It is shown that as accuracy demands are increased, the deceleration phase is lengthened. They also show that as the distance to the target is increased, the speed profile is scaled up. Arbib and Hoff restrict themselves to the free-space paradigm, so no interaction with the target is required. They suggest a single feedback process which could be responsible for both the quick and the slow accurate phase. This process consists of multiple feedforward and feedback sub-processes, some being conditionally executed based on sensory and motivational inputs. The optimization of the path is based on the minimum mean-squared jerk. Their results imitated human arm movements without actual interaction with the target.

In current research in our laboratory [4,7], the control of a mobile manipulator is broken down into the two phases detailed in the previous paragraphs. The ballistic motion described above is considered macro-manipulation. In the macro-manipulation phase, the mobile manipulator approaches the goal placing the end-effector within reach of it. Micro-manipulation motion is then used. In this phase, the exact placement and orientation of the end-effector is adjusted using additional feedback through vision and force/torque sensors. This motion replicates the biological division of motion.

This research describes the ballistic control of a mobile manipulator in an unmodeled environment with and without obstacles. A reactive control algorithm is developed which is capable of imitating biological arm movements but is not restricted by them. The relevant characteristics of biological arm movements considered are:

- Use of joints with highest potential for translation at longer distances from goal
- Bell-shaped speed profile of the hand
- Scaling speed profile by distance to goal

Additional non-biological parametric characteristics are examined and compared to the biologically motivated algorithms, with the goal being to identify those parameters which perform best in particular environments.

3. Approach

The central concept of mobile manipulation involves not merely treating the robot as an arm added to a mobile base, but rather a system that completely integrates the overall control of both. The approach is further motivated by a body of psychological and neurophysiological evidence which documents the preshaping of the arm and hand prior to picking up an object

[1]. To make efficient use of the robot, it is necessary to preshape the robot during the macro-motion as the mobile manipulator approaches the work area.

The preshaping of the arm is conducted by concurrently executing the motions of the base and the arm rather than a sequential move-vehicle followed by a move-arm. The motion is also decoupled into macro-motion and micro-motion components which permit the use of *a priori* expectations from both environmental and object models. The details of this approach appear below.

3.1 Overview

The basic concept in reactive macro-motion is to guide the arm and the base by creating artificial (pseudo) forces that pull the end-effector towards the goal while pushing the arm and base away from obstacles. An analogy to this paradigm is leading a child by the hand. The leader pulls the hand of the child towards the goal while pulling the child away from obstacles along the way as shown in Figure 2.

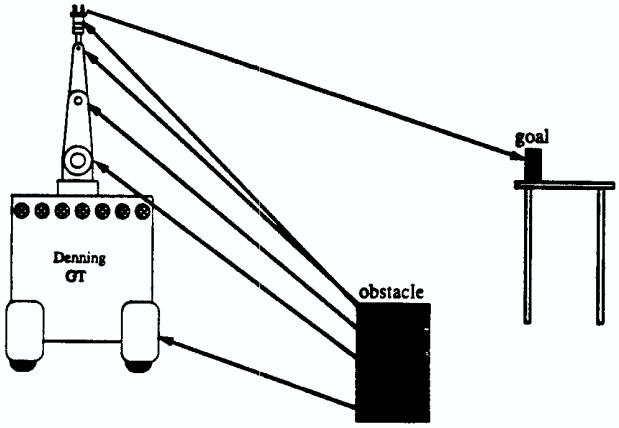


Figure 2: Macro-manipulation Paradigm

The architecture for the macro-motion involves an integrated sensor-based hierarchical/reactive planning and control architecture as discussed in [4,7]. Sensing, planning, and control are coordinated through the use of motor and perceptual schemas [3]. A motor schema is a basic unit of motor behavior such as **move-to-goal** and **avoid-static-obstacle**. A perceptual schema is dedicated to serving the perceptual requirements of a motor schema. Each active motor schema receives data from the appropriate perceptual schema and outputs a velocity vector in a manner analogous to the potential fields method. To apply the paradigm of leading a child by the hand, the **move-to-goal** schema creates a pseudo-force pulling the end-effector towards the goal using the following equation:

$$f = f_c \mathbf{u}_{g/ee} \tag{1}$$

where $\mathbf{u}_{g/ee}$ is a unit vector pointing from the end-effector to the goal and f_c is a constant gain.

The **avoid-static-obstacle** schema creates pseudo-forces and pseudo-torques which repel both the robot's joints and limbs [7]. Joint repulsion is a pseudo-force pushing each joint directly away from an obstacle within a threshold distance. It can be envisioned as a spherical repulsion field centered at each joint which pushes the joint away from obstacles. Limb repulsion is a pseudo-force and pseudo-torque applied at the previous joint to rotate the limb away from the obstacle. These pseudo-forces and torques act on each joint individually, and are not propagated through the robot. Without limb repulsion, it would be possible for an obstacle to strike the center of the limb (away from both adjoining limbs) unless the repulsive sphere-of-influence distance is large.

The **move-robot** schema sums the output of all the active motor schemas to derive the overall pseudo-forces and pseudo-torques acting on each joint. The schema organization is shown in Figure 3.

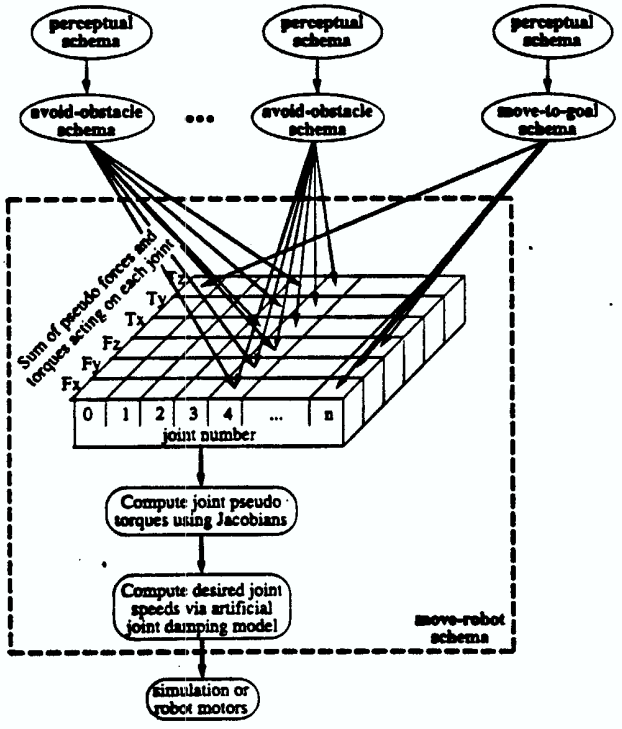


Figure 3: Schema Organization

The pseudo-forces and pseudo-torques determined in **move-robot** are converted to joint pseudo-torques using the mobile manipulator's Jacobian matrices. The Jacobian matrix is

defined as the relationship between static forces and the corresponding joint torques [7]. A Jacobian matrix can be constructed to relate the velocity of any joint in any frame of reference with the corresponding joint speed. The relationship between static forces on any joint and the joint-torque required to generate those forces is also given in terms of the same matrix:

$$\{\tau\} = [\mathbf{J}]^T\{\mathbf{f}\} \tag{2}$$

where τ is the n-vector of joint torques and $\mathbf{f}$ is the 6-vector of forces and torques acting on the joint.

The Jacobian for our manipulator is determined symbolically in an integrated fashion by a Mathematica [25] package constructed for this research [7]. This package automatically generates the code for the Jacobian matrix based on a symbolic description of the joints. The package can handle a manipulator composed of any number of limbs connected by joints of various types in a serial chain [8].

3.2 Damping Functions

Without any pseudo-resistance to movement in the mobile manipulator joints, the pseudo-torques in the joints would create an unstable system. To alleviate this condition, pseudo-damping is added to each of the joints. Since damping is the natural physical relationship between forces and speeds (when no springs are involved), damping is an appropriate analogy. The relationship between the pseudo-damping values in the joints and the speeds of the joints resulting from the pseudo-torques is obvious. If the damping value is high, motion is decreased, and for low damping, the motion is increased according to the following equation:

$$dq_i/dt = \tau_i/c_i \tag{3}$$

where τ_i is the pseudo-torque on joint i, c_i is the artificial damping for joint i, and dq_i/dt is the desired speed for joint i.

If constant and equal damping is added to each of the joints in the mobile manipulator, and a force is exerted to pull the end-effector to the goal, each joint will be used equally. The resulting motion of the robot in the absence of obstacles is marked by the extension of the arm towards the goal as the base translates. If the goal is not reached before the arm is fully extended, the robot will continue to translate with the arm remaining fully extended. This "Frankenstein walk" shown in Figure 4 needs to be avoided in areas cluttered with obstacles and when lifting a heavy object. An extended arm amongst obstacles becomes a hazard, and the joint torques required to lift a heavy object with the arm fully extended are very large.

A more natural movement evidenced in psychological and neurophysiological studies (e.g., [20]), is to use the base to translate towards the goal and to begin preshaping the arm as the manipulator nears the goal. The preshaping ideally finishes just as the end-effector is

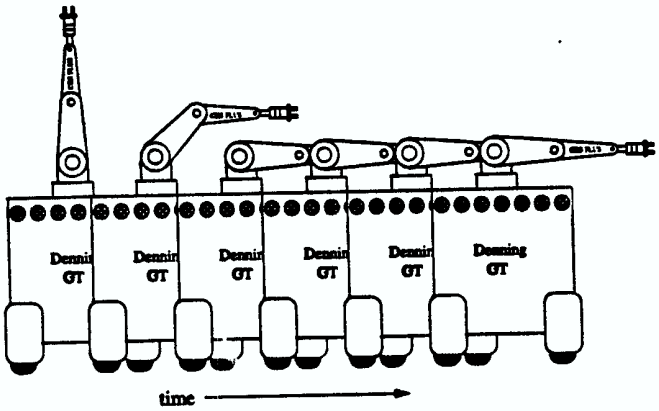


Figure 4: Frankenstein Walk

delivered to the goal. To provide this more natural movement, damping functions are used to vary the joint dampings individually in a distance dependent manner. The equation takes the form:

$$c_i = f_i(distance\_to\_goal) \tag{4}$$

where c_i is the pseudo-damping of joint i, and f_i is a function of distance to goal for joint i.

Typical joint damping functions shown in Figure 5 vary as a function of distance from the end-effector to the goal. To provide the natural movement mentioned above, the damping of drive and steer in the base is increased as the distance to the goal is decreased. This results in heavy utilization of the base to translate when the goal is distant. The damping of the remaining joints in the arm are decreased as the distance to the goal is decreased. This results in little movement of the arm until the goal is nearby. When the robot is a short distance from the goal, the arm damping is low, and the arm preshapes before the goal is encountered as shown in Figure 5.

There are an infinite number of possible damping functions with an infinite number of input parameters which could be used to modify the behavior of the mobile manipulator. It is not possible to exhaustively search this space, thus a limited number of damping functions and input parameters were chosen based on their expected merit. All of the chosen damping functions are intended to loosely follow the actions of biological systems. Biological systems are characterized by the following actions:

1. The initial and final speed of the end-effector are zero [17].
2. The speed profile of the hand is a bell shaped curve [17,18].
3. The speed profile of the hand is scaled by the distance to goal as seen in Figure 6 [18].

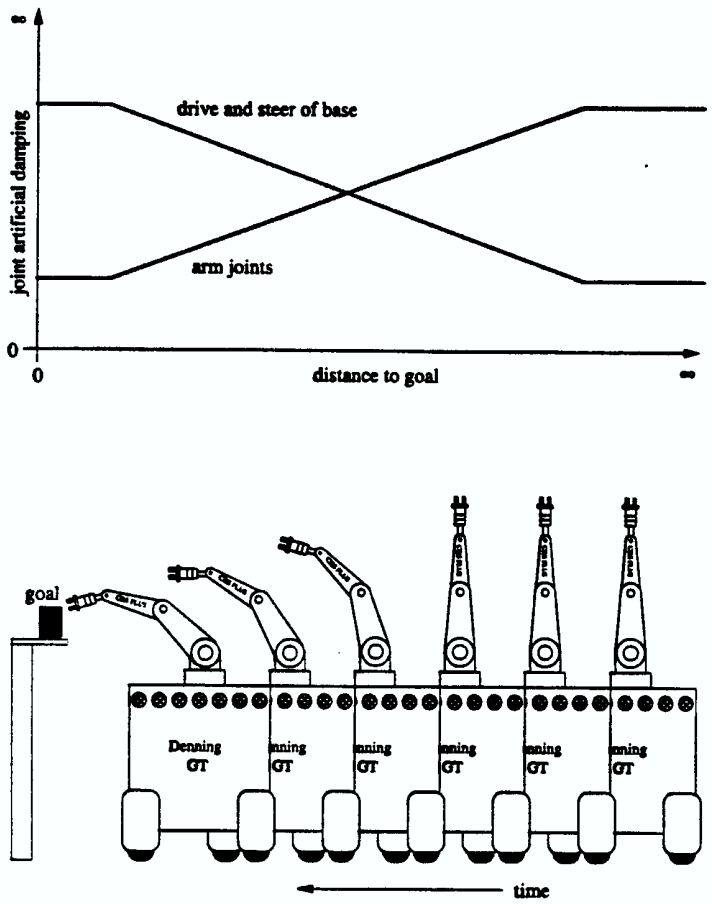


Figure 5: Typical Joint Damping

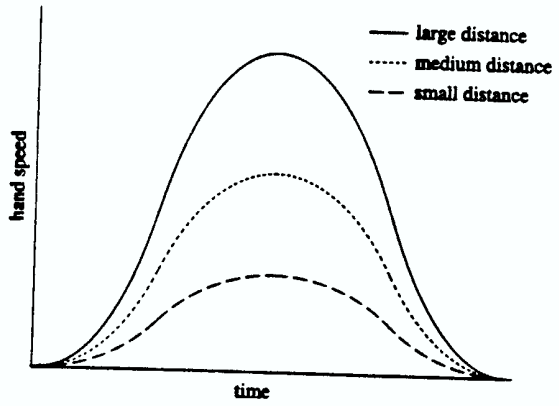


Figure 6: Scaling of Bell Shaped Curve with Distance

As a consequence, the damping of joints 0 and 1, the base drive and steer, (highest potential for translation) are inversely proportional to the distance to the goal:

$$c_{0,1} \propto 1/distance\_to\_goal \tag{5}$$

The damping of joints 2 through 6, the arm joints, are directly proportional to the distance to goal:

$$c_{2,3,4,5,6} \propto distance\_to\_goal \tag{6}$$

Since different damping functions are compared, an attempt was made to normalize the damping in the robot. The intent of the normalization was to keep the damping of different functions on the same scale. The first technique normalized the joint dampings by dividing each individual calculated joint damping by the sum of all the joint dampings. Each joint damping was then a percentage of the overall robot damping. This percentage was then multiplied by a constant, representing the total amount of damping to be used on the robot. The equation took the form:

$$damping_i = damping_i / \sum damping_j * constant \tag{7}$$

where $damping_i$ is the damping of joint i, $\sum$ is from $j = 0$ to $j = n$ where n is the number of joints, and *constant* is the total amount of damping for the robot.

This normalization caused significant problems in the simulation. Since the total damping allotted to the robot was distributed by the relative amount of damping in each joint, it was impossible to heavily damp all the joints in the arm without having the total amount of damping for the robot very high. If the total damping was set high to facilitate locking the arm, it became impossible to have little damping in the joints of the arm without heavily damping the base. The normalization of the joint damping also confuses the damping functions, making the damping a function of distance and of other joint dampings.

To keep the various damping functions on the same scale, the functions had the same maximum and minimum damping values for all joints. The difference between damping functions was the functions that were used to transition from maximum to minimum damping. This method does not guarantee that one damping function uses the same amount of overall damping as another, but it does at least keep the amount of damping used on the robot on the same scale.

Many damping functions have been simulated to survey the resulting movements of the mobile manipulator. Constant, step, linear, square, and cubic functions of distance to goal were investigated in many different environments. The simulation results follow.

4. Simulation

The simulation program calculates the resulting joint speeds from the effects of goal pseudo-forces, obstacle pseudo-forces, and damping values in each joint. The joint speeds are

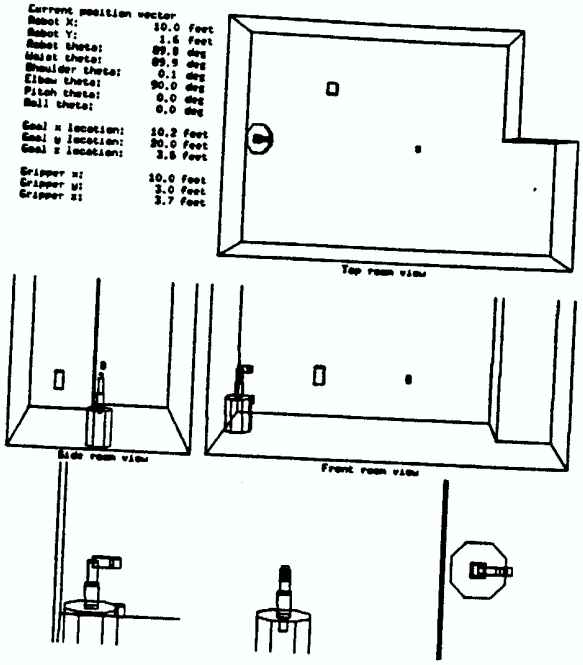


Figure 7: Snapshot of Simulation

sent to the robot for actual movement and/or sent to a display package using sphigs [13] for visualization as seen in Figure 7.

The simulation begins by initializing the start X,Y location of the robot base, the beginning joint angles in the base and arm, the goal X,Y,Z position, the obstacle X,Y,Z positions, obstacle height, and obstacle radius. After initialization, the program reads in the current joint angles. A function call is then made to the transform to determine the current position of the end-effector in world coordinates. The function uses the robot location in world coordinates and the joint angles to return the location of the end-effector relative to the robot and to the world coordinate systems. The distances from the end-effector to the goal in the X, Y, and Z directions, are then compared to the goal tolerance value to determine if the goal has been reached. If the distance to the goal in each direction is less than the tolerance value, the goal has been reached, and the robot is stopped. If any one of the distances is greater than the tolerance, new joints speeds are calculated.

New joint speeds are determined by first calculating the individual joint damping values. The individual damping functions use the distance to the goal and the current joint angles to return individual joint damping values. The function used is described by a set of four parameters: *max_dist*, *min_dist*, *max_damp*, and *min_damp*, which correspond to two points in a function. These parameters correspond to the maximum and minimum distance evaluated in the function, and the damping values at these distances. The simulator automatically determines the equation which fits these endpoints. Beyond the maximum distance, the damping value is held constant at the maximum damping value. Inside the minimum distance, the damping value is held constant at the minimum damping value. For example, to determine the square function ($y = Ax^2 + C$), two equations corresponding to (*max_dist*, *max_damp*) and (*min_dist*, *min_damp*) and two unknowns, A and C, are solved.

The pseudo-forces and pseudo-torques acting on the robot from the goal are then determined. The pseudo-forces from the goal are applied on the end-effector directly towards the goal by first determining the current end-effector and goal positions in robot coordinates. The positions in X, Y, and Z of the end-effector are then subtracted from the X, Y, and Z positions of the goal to determine the distance to the goal in each direction. The magnitude of the goal pseudo-forces in each direction are calculated by multiplying a gain factor by the distance to the goal in that direction divided by the straight line distance to the goal.

$$F_x = GAIN * dx/(dx^2 + dy^2 + dz^2)^{0.5} \tag{8}$$

$$F_y = GAIN * dy/(dx^2 + dy^2 + dz^2)^{0.5} \tag{9}$$

$$F_z = GAIN * dz/(dx^2 + dy^2 + dz^2)^{0.5} \tag{10}$$

where F_x, F_y, and F_z are the pseudo-forces acting on the end-effector in each direction; $GAIN$ is a constant; and dx, dy, and dz are the distances from the end-effector to the goal in each direction.

Additional pseudo-torques required to turn the wheels and the waist of the arm towards the goal are calculated by similar equations. A small torque to pull the wheels towards the arm waist direction is added to keep the wheels and the waist of the arm headed in the same direction:

$$\tau_{wheels} = GAIN * (goal\_angle) + GAIN * (q_{waist} - q_{wheels})/2.0 \tag{11}$$

$$\tau_{waist} = GAIN * (goal\_angle + q_{wheels} - q_{waist})(11) \tag{12}$$

where τ is the pseudo-torques acting on the wheels and the arm waist; $GAIN$ is constant; $goal\_angle$ is the angle between the direction of the wheels and the direction of the goal; and q is the angle of the wheels and the waist of the arm.

The pseudo-forces and pseudo-torques acting on the robot joints and links from obstacles is determined subsequent to the calculation of the positions of the obstacles in robot coordinates. The repulsive pseudo-forces from the obstacles are determined for the robot by calculating the distance from each limb and joint to the obstacle in X, Y, and Z. The distance to the obstacle is calculated by modeling the obstacle as a sphere:

$$base : r = (dx^2 + dy^2)^{0.5} - rad_{obstacle} \tag{13}$$

$$limbs : r = (dx^2 + dy^2 + dz^2)^{0.5} - rad_{obstacle} \tag{14}$$

where r is the distance to the obstacle, dx, dy, and dz are the X , Y, and Z distance to the obstacle center, and $rad_{obstacle}$ is the radius of the obstacle.

If the distance from the joint or limb is less than the sphere of influence of the obstacle, the pseudo-forces and pseudo-torques required to push the joint or limb away from the obstacle are added to the robot. The magnitude of the pseudo-force or pseudo-torque is directly

proportional to the distance to the obstacle to increase the pseudo-forces as the obstacle gets closer. By applying pseudo-forces from the obstacle on the limbs as well as the joints, a small obstacle located between two joints will still push the limb away.

All of the pseudo-forces and pseudo-torques acting on the robot are converted to the resulting joint pseudo-torques using the Jacobians as shown in Figure 3.

After the pseudo-torques acting on each joint are determined, the joint pseudo-torques are converted to joint speeds by applying the joint damping values. This is done by using the following equation for each joint:

$$S = F/D \tag{15}$$

where S is the resulting joint speed in ft/sec for a prismatic joint and radians/sec for a rotational joint, F is the applied force in lb_f or torque in $ft \cdot lb_f$, and D is the joint damping value. Damping has the units $lb_f \cdot sec/ft$ for a prismatic joint and $ft \cdot lb_f \cdot sec$ for a rotational joint.

The simulator then runs the robot at the calculated speed for the time step specified by the user. If the rotation of the joint will exceed the limit after the time step, the speed is clipped to prevent the rotation of the joint beyond its limit. The time step used is one-tenth of a second, which provided smooth motion in the simulated robot. The program then looped back to determine if the end-effector had reached the goal. The loop continues until the goal is reached.

The simulation does not include the dynamics of the robot. As a result, another level of servo-control must be used to convert our velocity commands to motor torques when controlling the actual robot. This is done by the CRS robot arm controller, and the Denning robot base controller (Sec. 5.5).

The entire simulation is run inside a series of loops which iterates the program through a set of pre-determined input worlds. The input worlds are described by the following parameters:

- damping function used
- beginning straight-line distance to goal
- height of goal
- number of obstacles
- X,Y position of obstacles
- obstacle heights
- obstacle radius
- starting position of each joint

By varying these input parameters, a large range of possible input parameters is simulated.

The placement of the obstacles is determined according to a three by three grid. The nine grid positions are determined by the straight line distance to the goal, the radius of the obstacle, and the sphere of influence of the obstacles. The sphere of influence of the obstacles defines the radius of the sphere surrounding the obstacle. Inside this sphere, the obstacle will repulse the robot, outside the sphere, the obstacle is ignored.

In the simulation, there exists the possibility of the robot getting trapped in a singularity. If an obstacle is located directly between the robot and the goal, the repulsive forces from the obstacle will be equal and opposite to the attractive forces of the goal as the obstacle is encountered. This singularity can be avoided by applying noise [2], by using learning methods [10,21,23], or by recognizing that the robot is not making progress, and replanning the behavioral configuration. The singularities in this simulation are avoided by placing the goal slightly off center from the obstacles (the use of noise would avoid this problem as well [2]). A total number of nine different worlds are simulated, placing the single obstacle at each of the nine possible obstacle positions.

4.1 Metrics

For each simulation run, an array of metrics is stored to measure the performance of the robot. The metrics fall under four categories: time, safety, distance, and flexibility. The metrics used are shown in Table 1.

Time is measured by the number of programs cycles required to complete the goal acquisition task. There is a maximum number of steps, set to 50,000 program cycles, that are allowed before the simulation is halted. If the maximum number of steps reaches this point, the program signals that the goal was not attained and saves the values of the other metrics up to that point.

Safety is measured by the minimum distance from each joint to an obstacle. The distance from each joint to the nearest obstacle is determined at each program step, and the minimum value was stored. The distance traveled by each joint is determined by keeping a running sum of the movement between each program cycle. Thus, this distance is total movement of the joint, not the difference between beginning and ending position. The last metric used for flexibility is the number of cycles that each joint was at its limit. A degree of freedom is lost when a joint reaches its limit, hence flexibility is lost.

4.2 Basis for Analysis

The simulation run's parameters for each input variable and the array of metrics are sent to an output file to be processed after all simulations are complete. A separate program loads the loop parameters and the array of metrics for each simulation, and post-processes

Table 1: Metrics

Number	Measure	Dimension
metric[0]	program cycles	time
metric[1]	min obstacle clearance of joint 0 (drive)	safety
metric[2]	min obstacle clearance of joint 1 (steer)	safety
metric[3]	min obstacle clearance of joint 2 (waist)	safety
metric[4]	min obstacle clearance of joint 3 (shoulder)	safety
metric[5]	min obstacle clearance of joint 4 (elbow)	safety
metric[6]	min obstacle clearance of joint 5 (wrist tilt)	safety
metric[7]	min obstacle clearance of joint 6 (wrist pivot)	safety
metric[8]	distance traversed of joint 0	distance
metric[9]	rotation of joint 1	distance
metric[10]	rotation of joint 2	distance
metric[11]	rotation of joint 3	distance
metric[12]	rotation of joint 4	distance
metric[13]	rotation of joint 5	distance
metric[14]	rotation of joint 6	distance
metric[15]	cycles at limit for joint 1	flexibility
metric[16]	cycles at limit for joint 2	flexibility
metric[17]	cycles at limit for joint 3	flexibility
metric[18]	cycles at limit for joint 4	flexibility
metric[19]	cycles at limit for joint 5	flexibility
metric[20]	cycles at limit for joint 6	flexibility

the data. This allows the data to be analyzed by looking at the average performance of a damping function over all of the simulations, or to look more closely at a particular world configuration.

The post-processing program averages the performance data of a specified damping function over a particular set of input worlds. The averaged set of performance metrics is sent to an output file to be plotted. By plotting the performance data of different damping functions over the same set of input worlds, the damping functions are compared.

5. Results

The first step in developing the different damping functions was to compare the speed profile characteristics of the end-effector in robot coordinates to that of the biological system discussed in Section 2. The characteristics of the biological system which were replicated were the initial and final speed of the end-effector approaching zero [17], general bell shape speed profile of the end-effector speed [17,18], and scaling of the speed profile with the distance to the goal [18]. The parameters used to describe the functions were modified until the resulting motion of the end-effector contained the previously mentioned characteristics. Replicating the biological systems provided a method both for tuning the damping functions and a providing a standard for comparison.

After replicating biological systems, each function was tested in different environments, to determine the effectiveness of the function in time, distance, safety, and flexibility.

5.1 Biological Characteristics

5.1.1 Constant and Step Damping

The first functions tested were constant damping and step function damping. The robot was started 25 feet away from the goal, the arm was extended straight up (since the natural position of the arm hanging straight down was not possible), the base and waist of the arm were in the direction of the goal, and the goal was placed at 3.5 feet above the floor. The initial parameters used to describe the functions were modified to determine the best approximation of the biological system. Many different damping values were tested for the constant damping function, a representative set was 4 $lb_f \cdot sec/ft$ for the base drive, 2 $ft \cdot lb_f.sec$ for the base steer, 2 $ft \cdot lb_f \cdot sec$ for the arm waist, and 1 $ft \cdot lb_f \cdot sec$ for the remaining joints of the arm. For the step damping function, the best approximation of a biological system was defined by values of 2 and 10 $lb_f \cdot sec/ft$ with the step at 6 feet for the base drive, 1 and 1.5 $ft \cdot lb_f \cdot sec$ with a step at 10 feet for the base steer, 1.5 and 1 $ft \cdot lb_f \cdot sec$ with a step at 10 feet for the arm waist, and 30 and 1 $ft \cdot lb_f \cdot sec$ with a step at 7 feet for the remaining joints of the arm. A plot of the functions is contained in Figures 8 and 9.

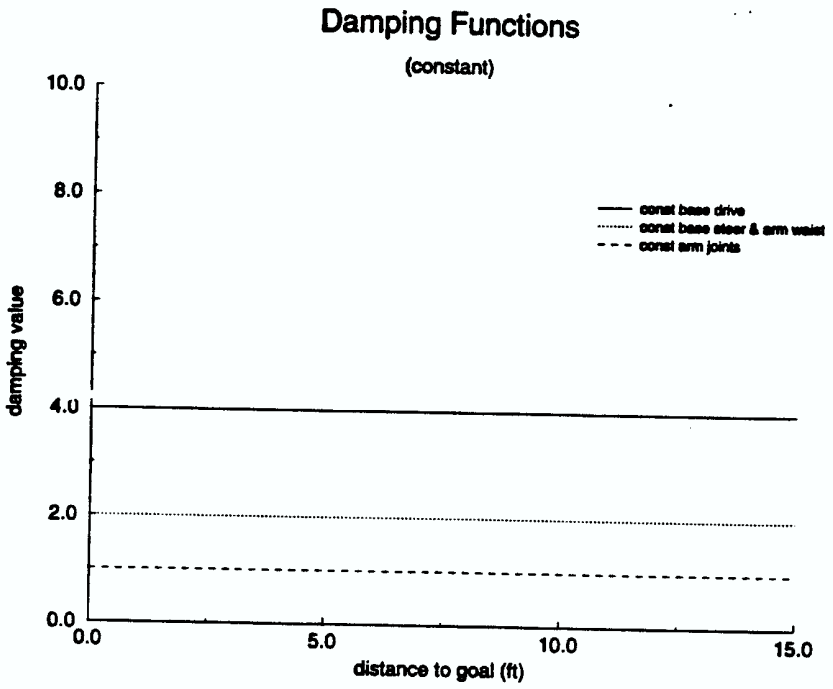


Figure 8: Constant Damping Functions

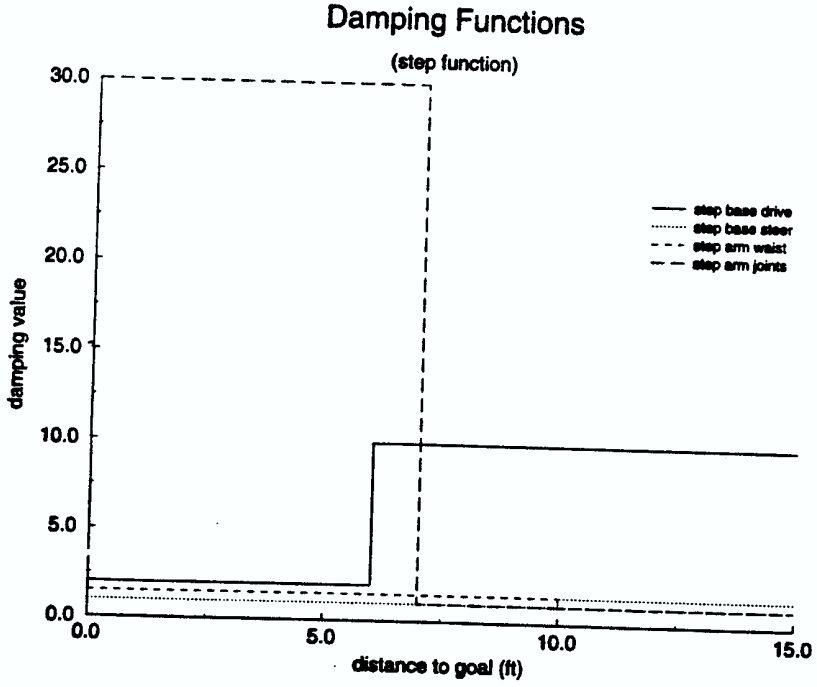


Figure 9: Step Damping Functions

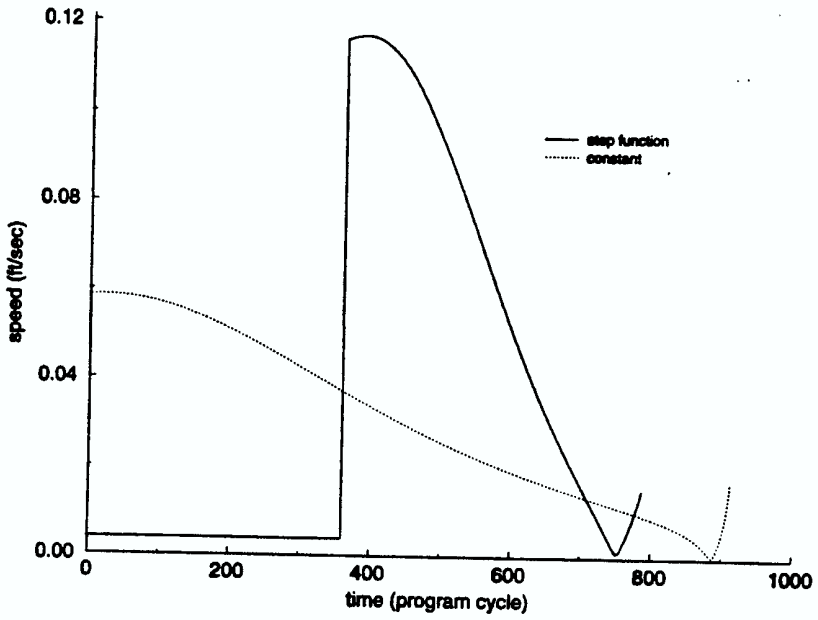


Figure 10: End-Effector Speed Profile for Constant and Step Damping

Plots of the end-effector speed demonstrate that these functions did not resemble a biological system (as seen in Figure 10). Constant joint damping was marked by an initial infinite acceleration followed by a decreasing end-effector speed. The profile neither had the bell shape nor the initial and final speed approaching zero. This scenario resembles that of the "Frankenstein walk" discussed in Section 3.2.

The step function, however, does begin with minimal movement in the arm and has the general bell-shaped curve of a biological system. Of concern, was the infinite acceleration at the point of the step.

In both the constant and step damping function, an overshoot occurred. The end-effector reached below the goal height (Z), and had to make up the difference at the end of the motion. This is what produced the "lip" at the end of the speed profile.

The joint torque is calculated by applying the force on the end-effector to each joint individually. The joint torque equals the end-effector force times the moment arm of that joint. In the only case where the arm was extended upwards and the goal was located above the elbow and shoulder joints, the torque created in the joints rotated the end-effector below the Z position of the goal. It can be seen in Figure 11, even after the end-effector reaches the correct Z location of the goal, the moment arm still creates a torque in the shoulder and elbow joints to rotate the end-effector below the goal Z location.

The moment arm about the joints was largest for the shoulder joint, less for the elbow joint, and smallest for the wrist. This caused a greater rotation of the larger limbs. Not until the robot became closer to the goal, and the moment arm reversed direction in the elbow joint, did the goal force tend to rotate the end-effector back up to the correct goal height.

A simulation which demonstrates the movement is shown in Figure 12. This phenomena of our approach was avoided by an increase in the damping of the arm with a decrease in the distance to the goal in the Z direction. This increased damping was used in subsequent

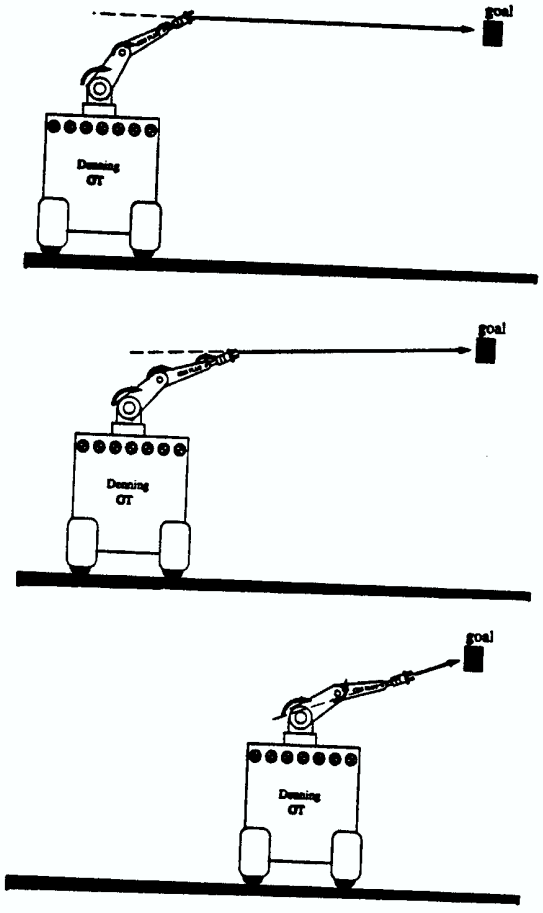


Figure 11: Diagram of Robot Overshoot

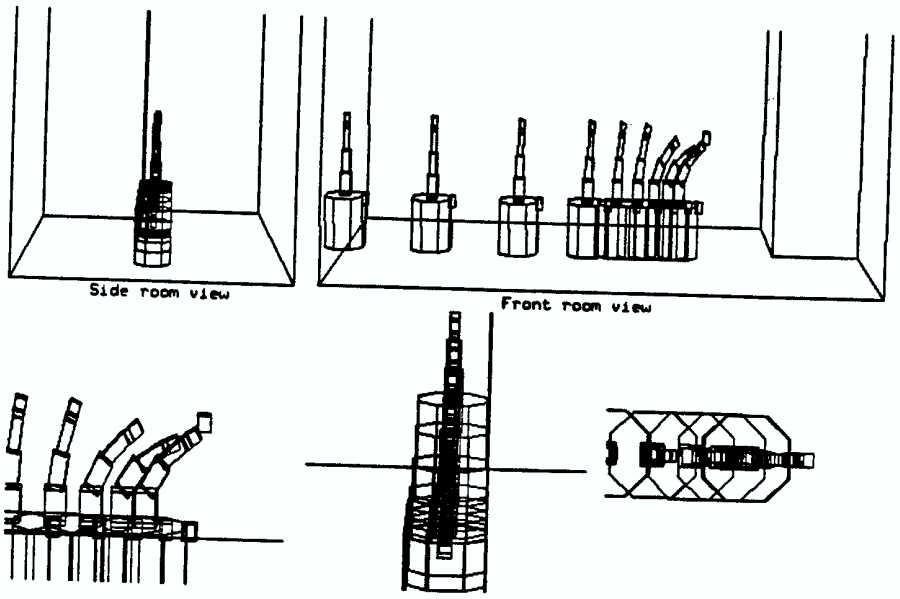


Figure 12: Simulation of Robot Overshoot

damping functions.

5.1.2 Linear Damping

Linear Damping functions were tested next. Again, the parameters describing the linear functions were adjusted to produce the characteristics of a biological system described earlier. In this case, the robot was started with the arm straight up, the goal 25 feet away, and the goal height varied from 2.5 feet to 4.5 feet. The limits the arm can reach are actually 5 feet at full extension straight up, and 2.0 feet if reaching straight down. These extremities were not simulated since they should be avoided by design. The linear functions used can be seen in Figure 13. Plotting the speed of the end-effector shows that the bell-shaped speed profile is attained, but the occurrence of overshooting and undershooting in the Z direction are apparent in Figure 14. In biological systems, there is a scaling of the profile with the distance [18]. To provide for this scaling, the damping of the arm must be increased if the distance to the goal in the Z direction is small, and decreased if the distance to the goal in the Z direction is large. The next set of functions includes this addition.

The same set of goals were simulated using the new linear damping functions. These functions adjusted the damping of the arm at the minimum distance according to the initial distance to the goal in the Z direction. The adjusted minimum damping of the arm was empirically determined by adjusting the minimum damping value for goal heights of 2.5, 3.0, 3.5, 4.0, and 4.5 feet until the overshoot and undershoot of the end-effector was minimized. The curve fit of these data points is defined by:

$$minimum\_damping = 3.13 - 2.0143 * dz + 0.37143 * dz^2 \tag{16}$$

where dz was the initial distance to the goal in the Z direction.

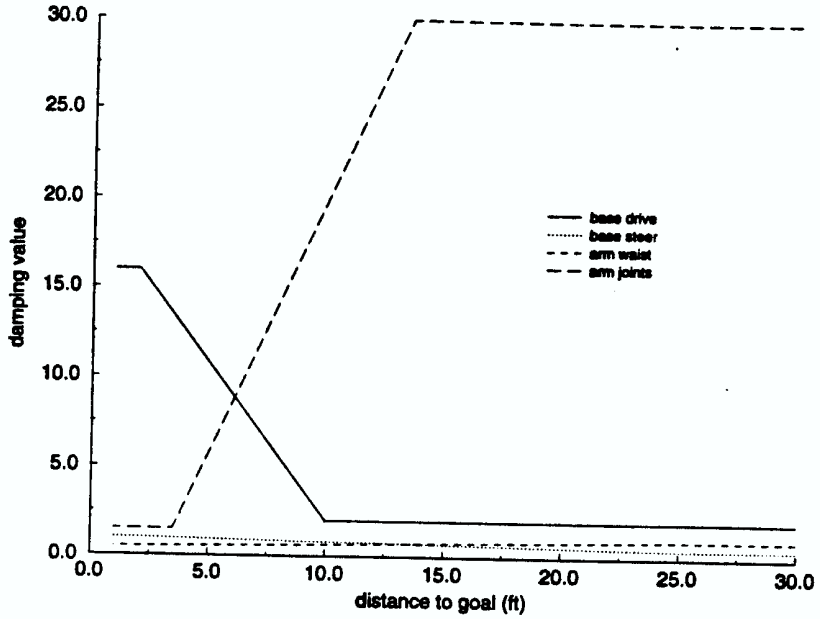


Figure 13: Linear Damping Functions

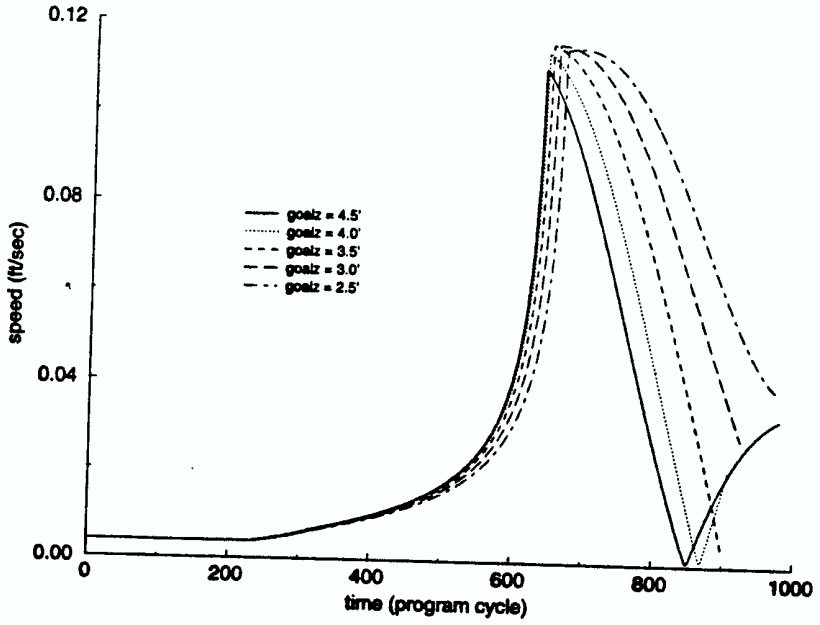


Figure 14: End-Effector Speed Profile for Linear Damping Function

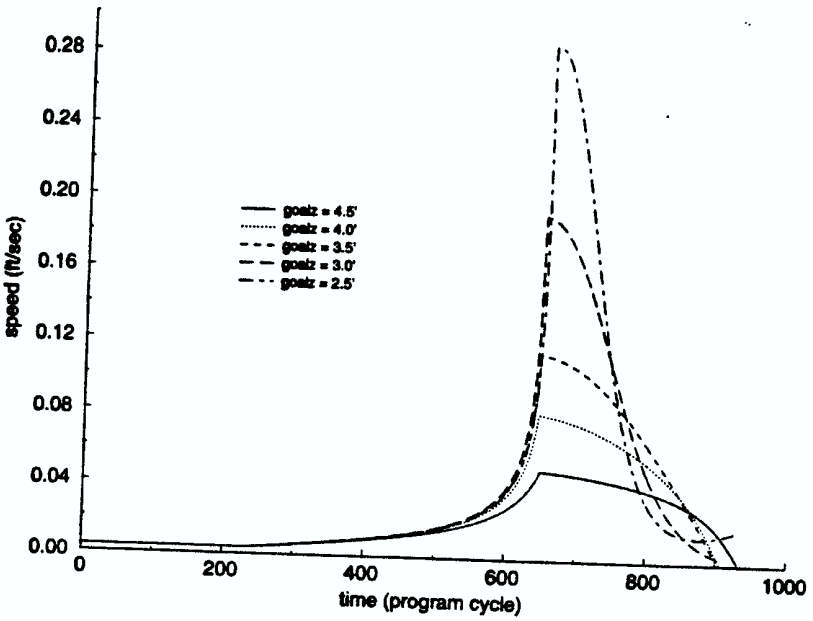


Figure 15: End-Effector Speed Profile Using Modified Linear Damping

As seen in Figure 15, the speed profile of the end-effector retains the bell-shaped curve and now is scaled with the distance that the end-effector must travel. To test the linear damping functions further, obstacles were added, and both the heading of the robot and the starting configuration of the arm was varied.

Figure 16 plots the end-effector speed with obstacles placed at all of the nine positions discussed in Section 4. The spikes seen in the profile are due to the interaction of the arm and the obstacle. The obstacle with the greatest effect on the motion of the arm was, not surprisingly, located directly in front of the goal. Because the obstacle was close to the goal, the damping in the arm was lower, and the arm was pushed away. After the robot cleared the obstacle, roughly the same speed profile was attained with some overshoot as explained before. The robot took longer to reach the goal after the damping in the arm had begun to decrease, so the arm had longer to preshape, and longer to overshoot.

To determine the effects of the initial heading of the robot, the heading was varied from directly towards the goal (0 degrees) to directly away from the goal (180 degrees). The robot took longer to reach the goal as the heading was further away from the goal, but the bell shaped curve and initial end-effector speed of approximately zero were still attained. As the heading of the robot approached 180 degrees, the final speed of the end-effector increased to a maximum of 0.025 ft/sec (Fig. 17).

The last variation tested involved the starting configuration of the arm. Configurations were varied by changing the angle of the shoulder, elbow, and wrist of the arm. The linear damping functions without the addition of modifying the minimum damping according to the Z distance to the goal were compared to linear damping functions with this addition. In each case, the speed profile loosely followed the bell curve and the scaling without modifying

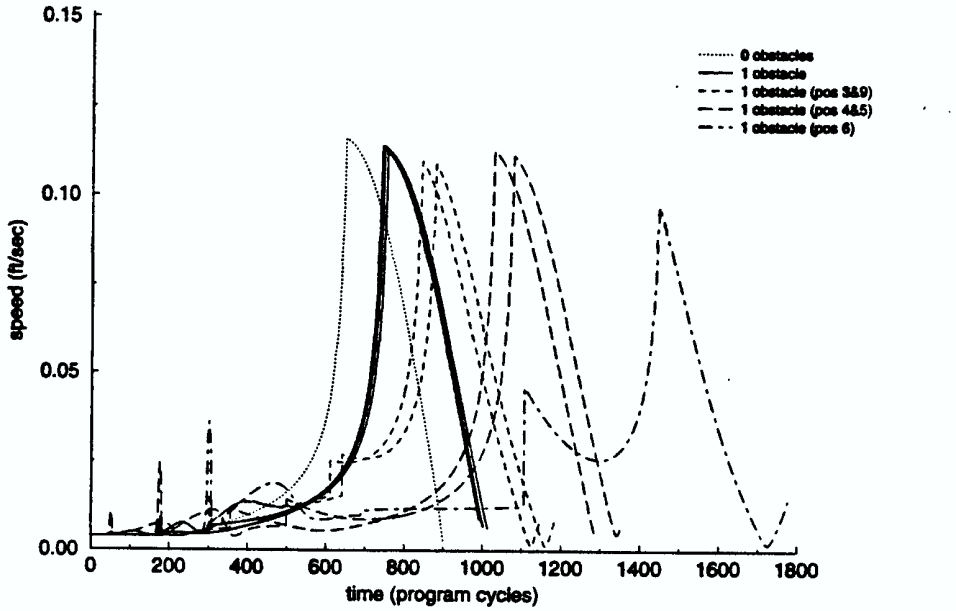


Figure 16: End-Effector Speed Profile for Modified Linear Damping with Obstacles

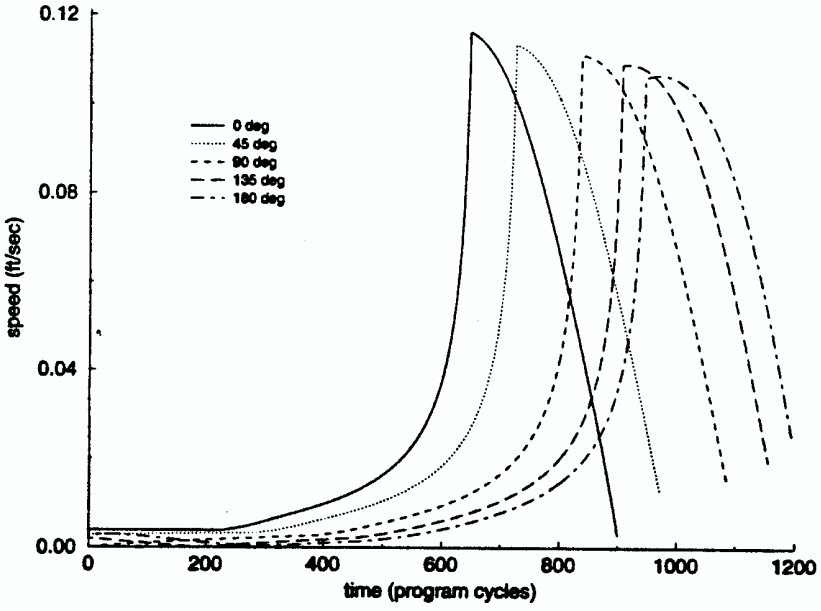


Figure 17: End-Effector Speed Profile for Modified Linear Damping and Varying Robot Heading

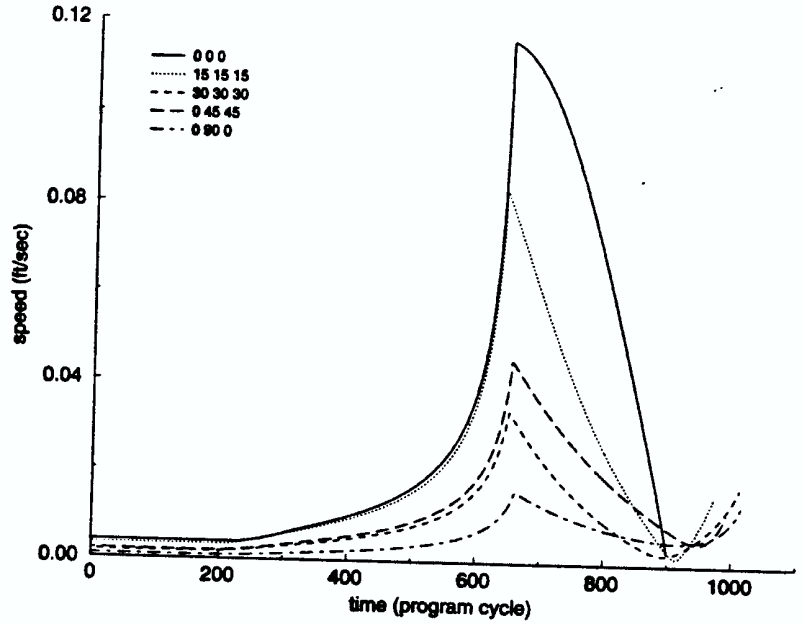


Figure 18: End-Effector Speed Profile for Varying Configurations (Linear)

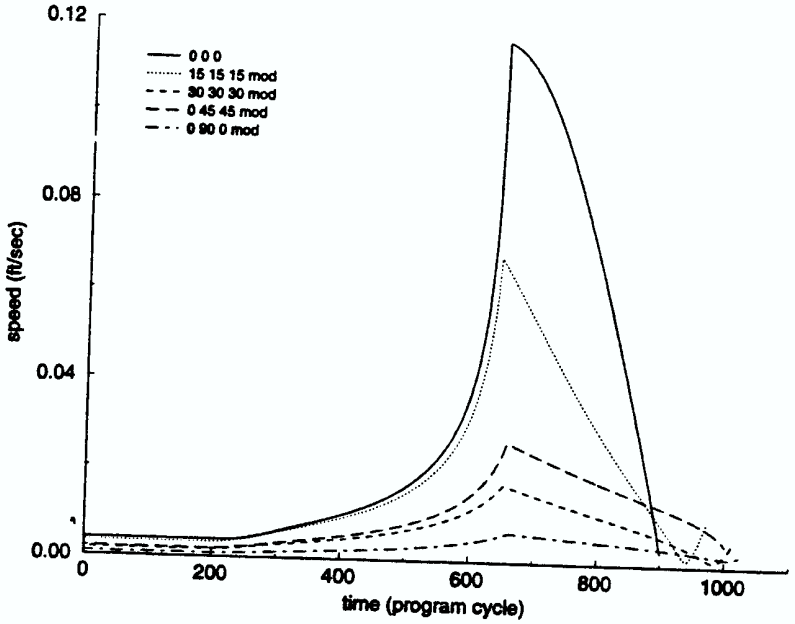


Figure 19: End-Effector Speed Profile for Varying Configurations (Modified Linear)

the minimum damping, but better results were produced when the minimum damping value was determined by equation 16. Results are shown in Figures 18 and 19.

Combinations of different goal height, obstacles, heading of the robot, and starting configuration of the arm all verified that the linear damping function was effective. In all situations, the bell shaped profile of end-effector speed was retained, the initial and final end-effector speed was approximately zero, and the profile was scaled according to the distance to the goal.

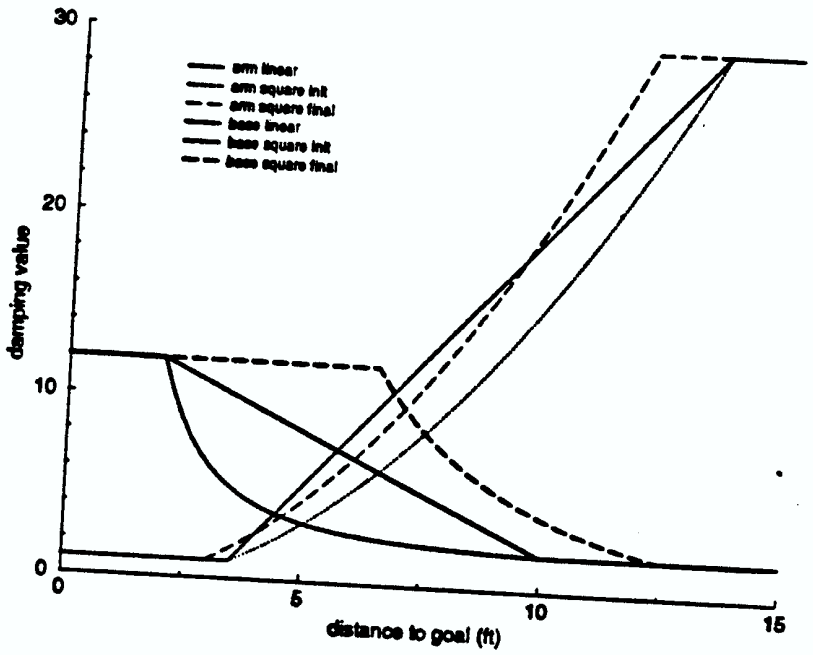


Figure 20: Square Damping Functions

5.1.3 Square and Cubic Damping

The square and cubic damping functions used took the form:

$$damping = C_1 * (distance)^n + C \tag{17}$$

where C and C_1 were constants and n was 2 and 3 for square and cubic functions respectfully.

The functions were initially started with the same endpoints as in the linear case. In both instances, the parameters had to be modified to approach a bell shaped profile for the end-effector speed. Since the slope of the square damping function for the arm was initially steeper, this function had to be moved closer to the goal to attain the biological characteristics. The slope of the damping function for the base, however, was considerably less steep initially and had to be moved further away from the goal. The resulting square damping functions can be seen in Figure 20.

The plot of the end-effector speed for linear, square, and modified square damping functions is shown in Figure 21. The initial square damping functions created a very steep slope in the end-effector speed caused by the damping in the base being insufficient as the arm began to preshape for the goal. When the square functions were modified by applying equation 16, the base damping was increased enough to allow the arm to reach the goal properly.

Varying the goal height from 4.5 feet to 2.5 feet, as for the linear damping functions, produced similar results. The goal below 3.5 feet had significant overshoot, while the goals above 3.5 feet had significant undershoot. To attempt to alleviate the problem, the same modification of the minimum damping value using equation 16 was applied to the square damping case. Again, identical results were found. The end-effector speed profile was scaled with the distance that the arm must travel, and the final speed of the end-effector was closer to zero.

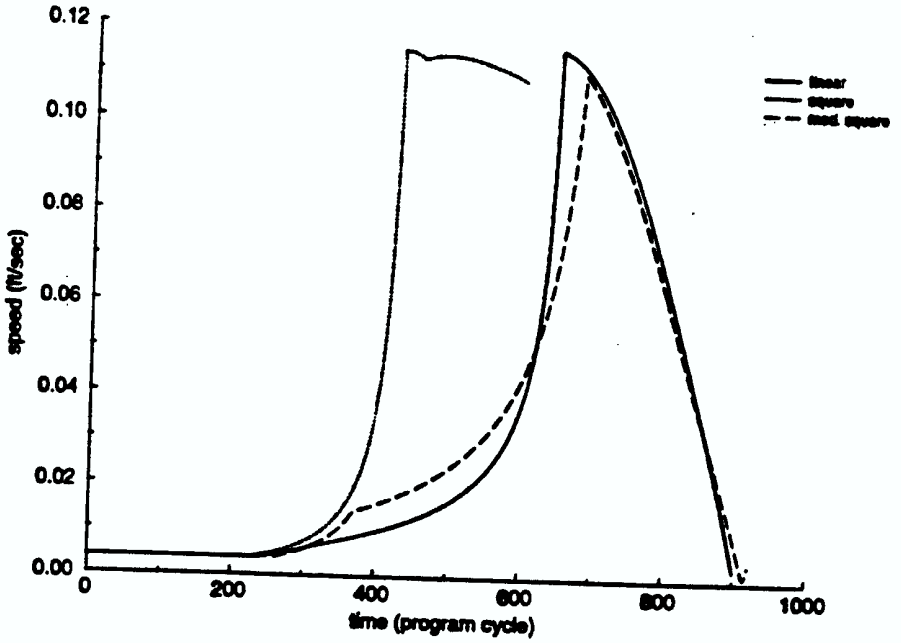


Figure 21: End-Effector Speed Profile for Square Damping Functions

Cubic damping functions were tested next. The cubic functions had a very high slope, and were approaching a step function. To modify the cubic damping functions, the arm function had to be moved closer to the goal again, and the base function further out again. The resulting cubic damping functions are shown in Figure 22. The resulting speed profile of the end-effector did not match the biological model as well. The initial cubic damping function had sharp corners and steep slopes due to the steep slope of the base damping function. After being modified, the slope of the base damping function was decreased, helping to bring the speed profile closer to the biological model. This can be seen in Figure 23.

By varying the height of the goal from 4.5 feet to 2.5 feet, similar results were found. Low goals were marked by significant undershoot of the end-effector, while high goals were marked by significant overshoot. As before, the same modification to the minimum distance damping value, produced a better scaling of the end-effector speed profile, and lower end-effector speeds at the completion of the task.

A summary of the biological characteristic results is shown in Table 2. It can be seen in this table that the modified linear and modified square damping functions fit the characteristics of the biological system the best. Modified cubic damping functions did not perform as well, but were a close runner-up to the modified linear and square damping functions.

5.2 Analysis Using Metrics

5.2.1 Biological Characteristics

The metrics of time, distance, safety, and flexibility were used to compare the constant, step, linear, and modified linear damping functions. The purpose of the comparison was to test the performance of the damping functions which closely represented the target biological characteristics.

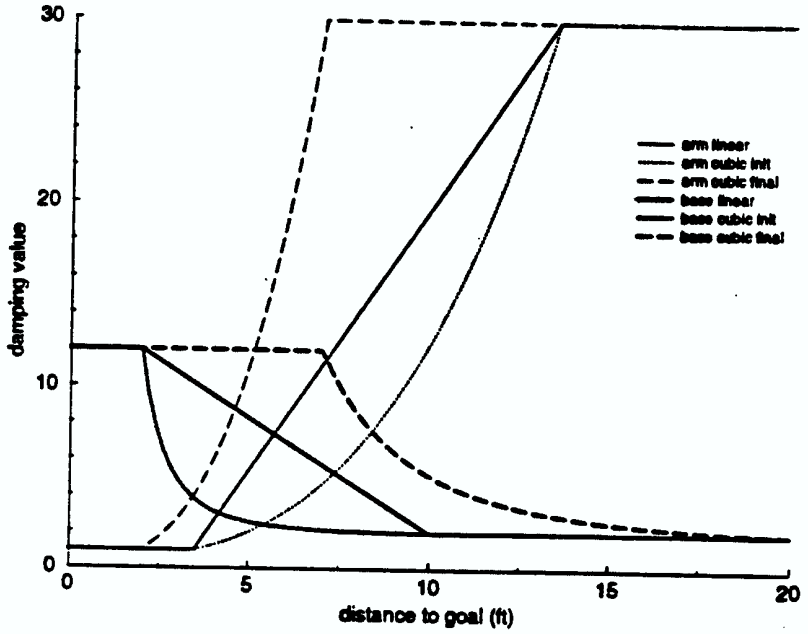


Figure 22: Cubic Damping Functions

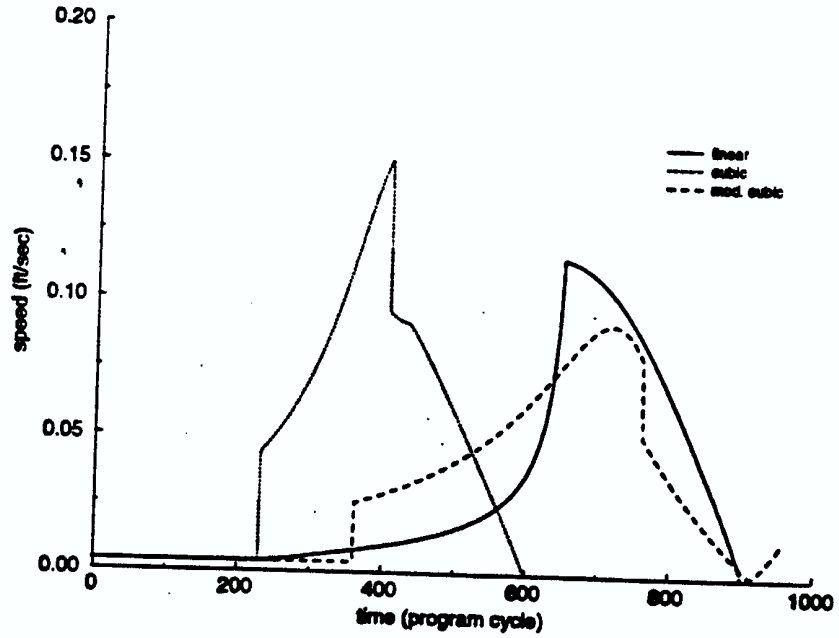


Figure 23: End-Effector Speed Profile for Cubic Damping Functions

Function	Bell Shape	Scale w/Dist	Init. Speed	Final Speed
Constant	not possible	poor	poor	good
Step	good	poor	good	poor
Linear	very good	poor	good	poor
Linear w/mod	very good	very good	good	good
Square	very good	poor	good	poor
Square w/mod	very good	very good	good	good
Cubic	fair	poor	good	good
Cubic w/mod	fair	very good	good	good

Table 2: Summary of Biological Characteristic Results

The constant, step, linear, and modified linear damping functions were tested in a series of 360 different world configurations which varied the distance of the goal from 15 to 25 feet, goal height from 2.5 to 4.5 feet, beginning robot heading from 0 to 90 degrees, arm starting angles from 0 to 90 degrees, and the number of obstacles from 0 to 1 in all 9 positions of the obstacle grid. The objective was to determine if an improvement in the performance of the robot was associated with a closer approximation of a biological system. If this improvement was true, the best performance would be realized by the linear damping function with the scaling modifications, down to the worst performance by the constant damping function.

The results of the test confirmed the improvement of all robot performance metrics with closer approximation of a biological system. The most dramatic improvements from constant to linear damping, were seen in the time to complete the task and the number of cycles a joint was at its limit. The plots are shown in Figures 24 and 25.

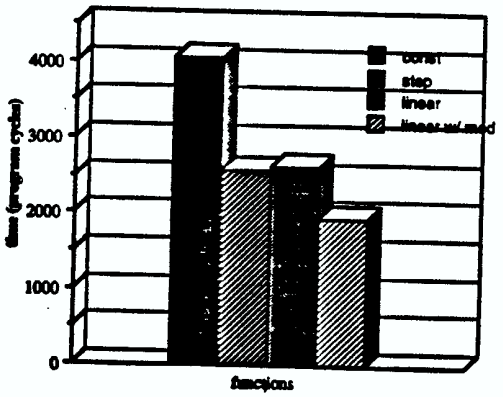


Figure 24: Time to Complete Task

Average time to complete the task was decreased by more than 50% from constant to modified linear damping functions. The time performance of the step and linear functions were nearly identical. This trend held true for all the performance metrics.

A reduction of over 75% in average number of cycles a joint was at its limit was realized from constant to modified linear damping. Again, the step and linear damping functions

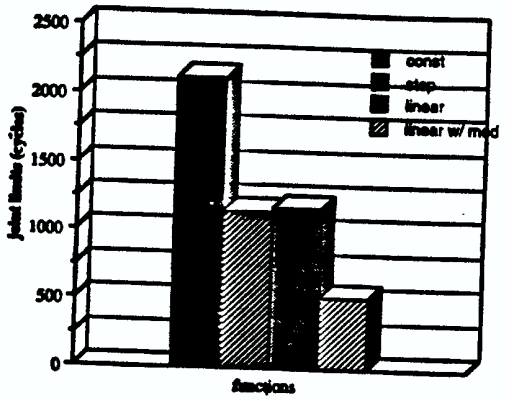


Figure 25: Average Cycles of Joint at Limit

performed nearly identically. This highlighted the fact that the step and linear functions produced similar end-effector speed profiles, although the step function was marked by infinite acceleration at the point of the step.

The performance of the damping functions which produced the best match to the biological system, modified linear, modified square, and modified cubic damping functions, were then compared in the same set of 360 worlds used previously. Time to complete the task was minimized by the square damping function as seen in Figure 26. The difference between the linear and square damping function time to completion was approximately 250 cycles, or 25 seconds at 10 cycles per second. This savings in time amounts to a 10% improvement. More dramatic improvement in robot performance was seen in the obstacle clearance and joint limit metrics.

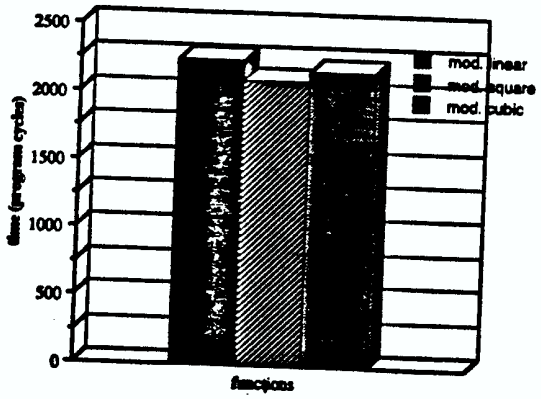


Figure 26: Time to Complete Task

The trend in distance traveled by the robot was a decrease in distance traversed by the base and an increase in the rotation of the joints of the robot base and arm when changing from linear to square to cubic damping functions. This is explained by the increase in initial slope of the damping in the joints, and a decrease in the initial slope of the damping in the base when using higher power damping functions.

The use of higher power damping functions comes at a cost. Base minimum obstacle clearance is decreased and cycles at joint limits are increased as the damping function changed from linear to square to cubic. These trends are shown in Figures 27 and 28. The decrease in the base minimum obstacle clearance from linear to cubic damping functions is over 2.1 feet, a 41% reduction. This reduction in the safety of the robot is accompanied with an increase of 19% in the number of cycles a joint was at its limit. A steady increase in the number of cycles at limits is associated with a decrease in the flexibility of the robot. A summary of the performance metrics is shown in Table 3.

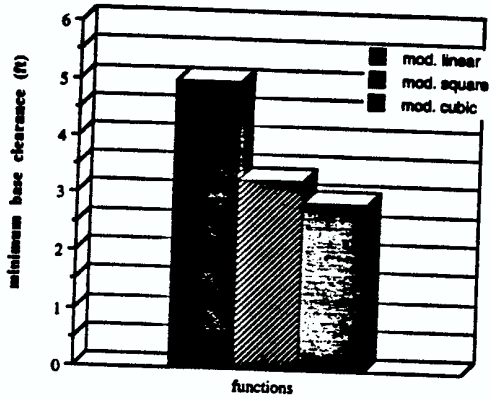


Figure 27: Average Base Minimum Obstacle Clearance

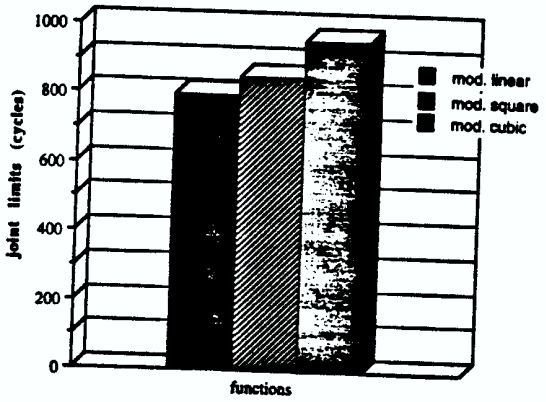


Figure 28: Average Cycles at Joint Limit

Table 3, shows that the 25 second gain in time to complete the goal acquisition task by using square versus linear damping, is equaled by the decrease in safety and flexibility of the robot. It is clear, however, that these two damping functions outperform the cubic damping function.

Function	Time	Base Dist	Joint Rotate	Safety	Flexibility	Total
Linear	3	3	1	1	1	9
Square	1	2	2	2	2	9
Cubic	2	1	3	3	3	12

metric rank: 1 = best 3 = worst

Table 3: Summary of Metric Results

5.2.2 Variation in Robot Speed

The speed of the robot as it approaches the goal can be controlled by changing the damping value at the minimum distance. The robot will make a slower approach, if the damping of the joints is increased. A faster approach is made when these damping values are lowered. The effect of this speed variation is tested by running the robot through 360 different world configurations, under three variations of the damping at the minimum distance. The first variation increases damping by approximately 50%, the second is identical to previous runs, and the third decreases the damping by approximately 50%.

For all of the three damping functions used, (linear, square, and cubic), the same trends were found. As the speed is increased by the decreased damping, the time required to complete the task is shorter. The actual travel of the base and rotation of the joints (the distance metric) is increased with the increase in speed. A result of the higher speed, is also a decrease in the safety and flexibility of the robot. It is clear that if the speed is increased, the efficiency of the robot motion is decreased. This is of special concern when the actual robot is controlled. The delay between commands is approximately 1 second, so the speed of the robot must be decreased to realize the same performance as in the simulation.

5.3 Specific Robot Constraints

Two specific constraints investigated were a target ending configuration of the arm and the location of a goal at the limit of the arm waist joint. The purpose of the macro-motion phase in mobile manipulation is to deliver the end-effector to the goal. A particular arm configuration at the delivery point may be preferred. The next section tests the effects of damping functions on the ending configuration of the arm.

Reaching the limit of the arm waist while trying to rotate towards a goal posed a problem for our approach. The torque pulling the arm towards the goal, was useless after the arm reached its limit because the Jacobian was no longer correct. The base would not make up for this problem. This problem is discussed earlier in the article.

5.4 Final Arm Configuration

The ending arm configuration was easily modified by changing the damping values of the arm. What needs to be known is the starting configuration of the arm, and the desired ending configuration of the arm. An example was a starting configuration with the arm pointing straight up. By increasing the damping of the elbow and wrist joints, the shoulder was left to do most of the motion. The result was an ending configuration with the arm reaching straight out (Fig. 29). In contrast, when the damping of the elbow and wrist were decreased, the ending configuration of the arm wa · bent. This ending configuration is shown in Figure 30.

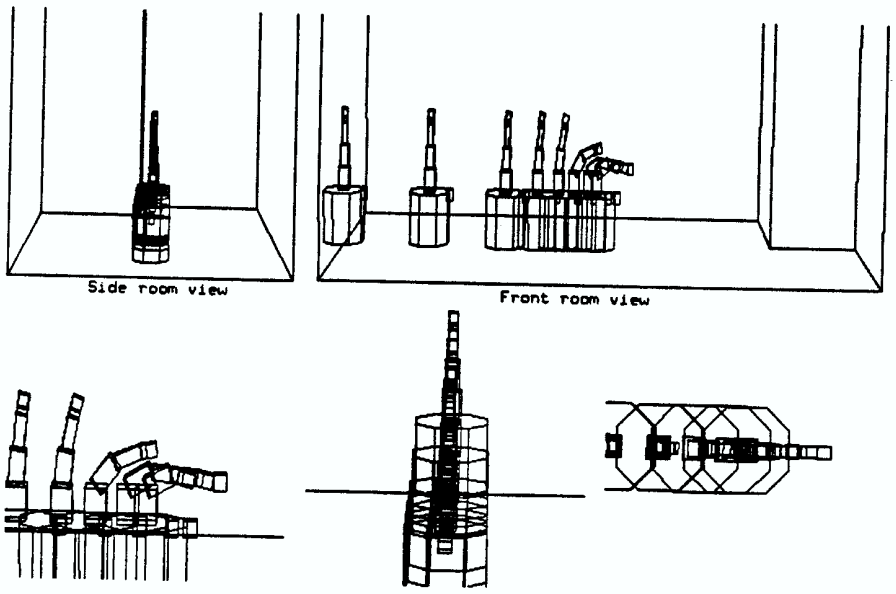


Figure 29: Ending Configuration Arm Extended

Another typical starting configuration is the shoulder joint at 0 degrees, the elbow at 90 degrees, and the wrist at zero degrees. To result in an ending configuration with the arm reaching out, the wrist was heavily damped, and the shoulder and elbow joints had decreased damping. This allowed the force on the end-effector to straighten out the arm. The simulation is shown in Figure 31. When the bent configuration was desired, the shoulder and elbow joint were heavily damped, and the wrist had decreased damping. This ending configuration is shown in Figure 32.

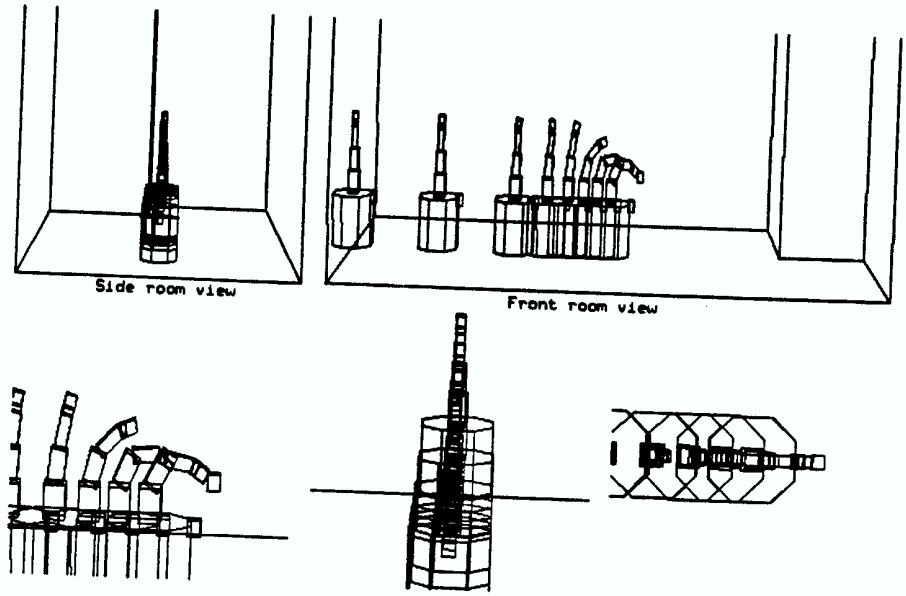


Figure 30: Ending Configuration Arm Bent

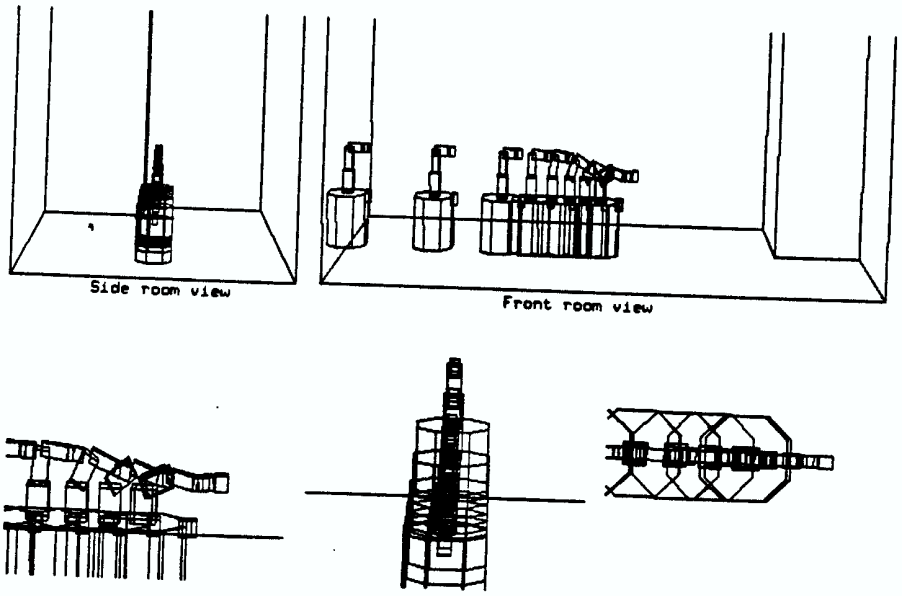


Figure 31: Ending Configuration Arm Extended

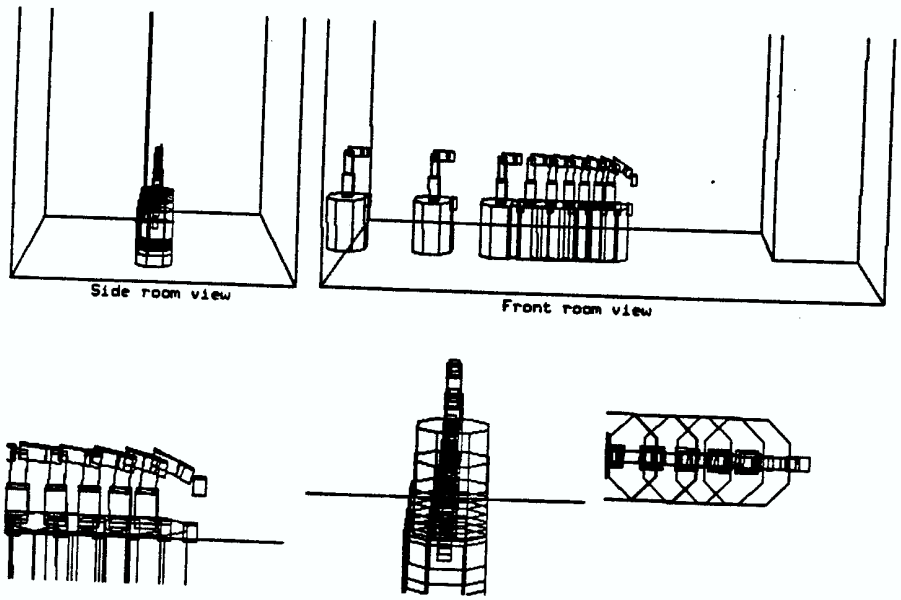


Figure 32: Ending Configuration Arm Bent

5.4.1 Goal at Limit of Arm Travel

One situation that the current robot configuration cannot solve with this method, is when the waist of the arm is at its limit (175 degrees) when directly in front of the goal. In our approach, the torque on the waist of the arm and the base is dependent on the angle between the robot joint and the goal. The arm is pulled towards the goal, but cannot rotate any further. There is no torque on the base because the base is heading directly towards the goal. Because the force on the end-effector was not used to produce the torque on the arm or the base, the robot was not able to correct for the problem. This situation is shown in simulation in Figure 33. This is an artifact of the particular hardware robot configuration used. If the waist had a full 360 degrees of rotation, this problem would not occur.

5.5 Robotic Runs

The reactive control algorithm using damping has been successfully ported to our Denning/CRS mobile manipulator. Figures 34 and 35 show specific results. Shaft encoders were used to provide proprioceptive information regarding the location of obstacles and the target object. This perception will be replaced in future research by ultrasonic algorithms for obstacle detection and vision for target recognition.

In the first run (Fig. 34) the robot starts with the arm completely extended facing the target object (a joystick) which is located approximately 10 feet away. There are no obstacles between the mobile manipulator and the goal. A linear damping function was used. While the robot is far from the target, most of the motion occurs within the base. As it approaches the target, the base progressively becomes more damped while the arm's motion is more free. The robot finally delivers the end-effector to a point near the target, ready for the micro-manipulation phase [4].

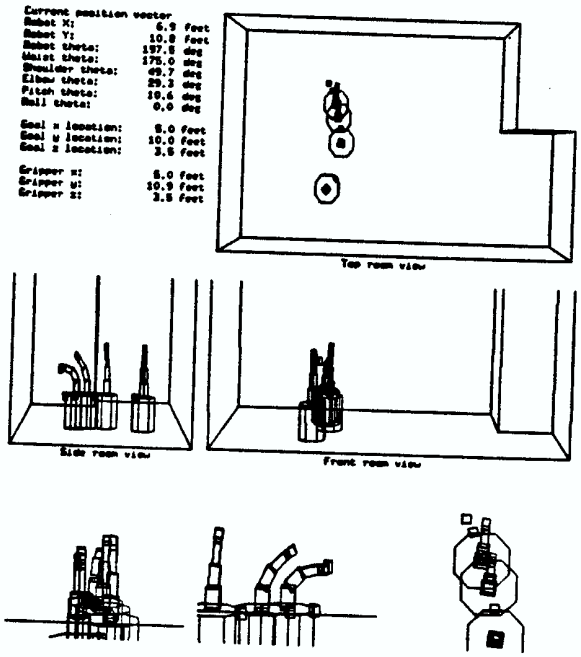


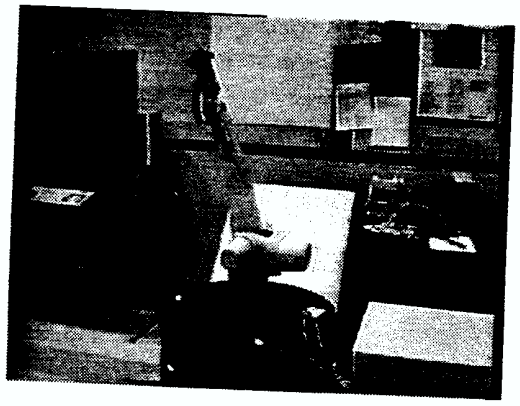
Figure 33: Simulation of Arm at Limit

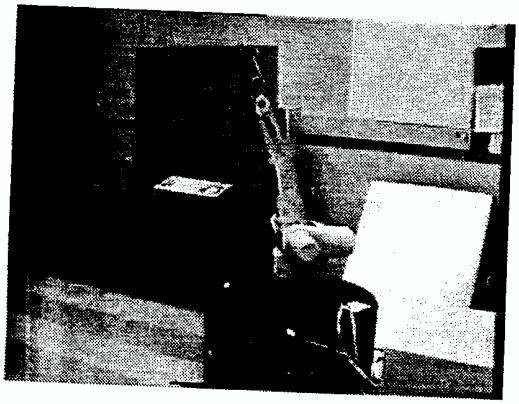
The next run (Fig. 35) is similar, but an obstacle has been added between the robot and the target object. As the robot enters the obstacle's sphere of influence it is repulsed by the obstacle while still being attracted towards the goal. The mobile manipulator successfully circumnavigates the obstacle and reaches the goal without ever computing a global trajectory. This reactive method has the decided advantage of being capable of handling dynamic objects involving the motion of the target object and various obstacles in the way.

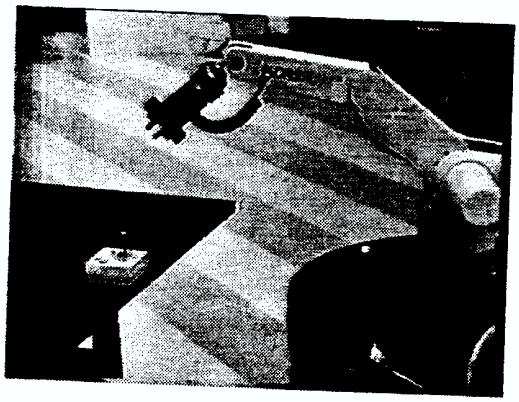
6. Conclusion and Summary

These simulations indicate that the paradigm of an attractive pseudo-force on the end-effector towards the goal and repulsive pseudo-forces on the joints and limbs of the robot from obstacles, along with the use of joint pseudo-damping functions, is a useful approach for the control of a mobile manipulator. Damping functions which produce the biological characteristics of bell-shaped end-effector profile, initial and final end-effector speed approaching zero, and a scaling of this speed profile with the distance to the goal are possible. By producing these damping functions, the mobile manipulator's performance in time, safety, distance, and flexibility is improved.

The damping functions which best approximated the biological characteristics and had the best performance were the modified linear damping and the modified square damping. Modified linear damping excelled in safety and flexibility, while modified square damping decreased, the time to complete the task with decreased safety and flexibility. If knowledge







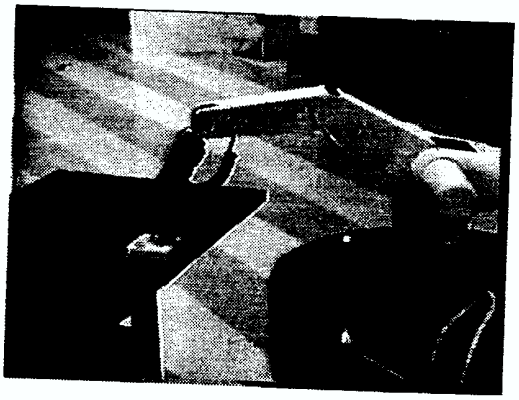
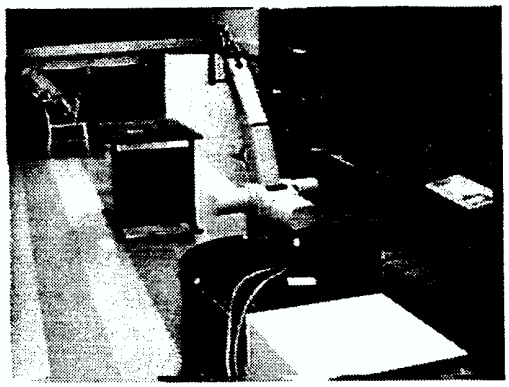
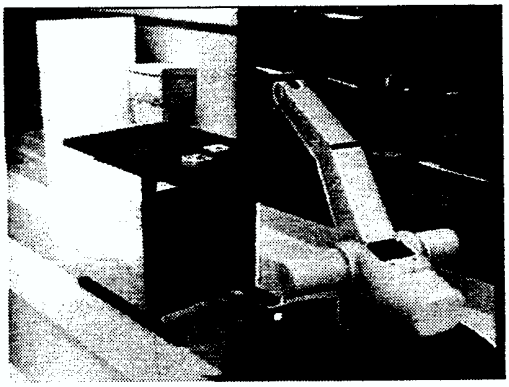
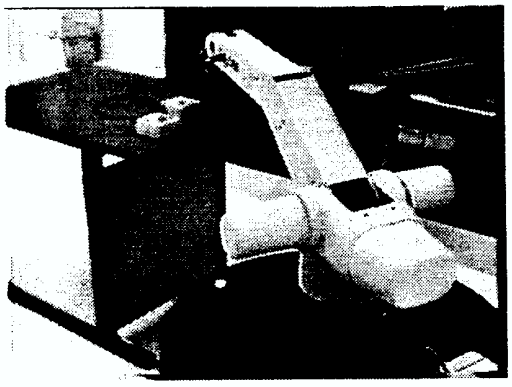


Figure 34: Experimental Run 1







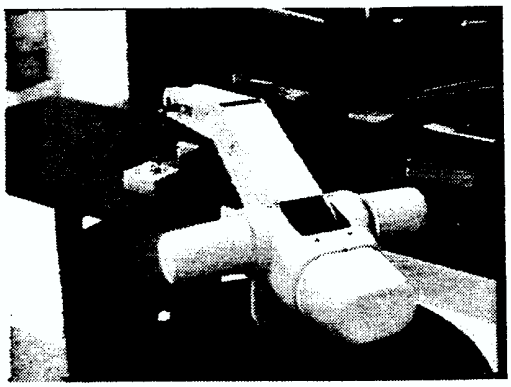


Figure 35: Experimental Run 2

of obstacle density is known *a priori*, the most effective damping function can be chosen. For an environment with obstacles, the modified linear damping functions will provide the safety and flexibility to maneuver through these obstacles. If there are few or no obstacles, the modified square damping functions will decrease the time to complete the task.

The performance of the mobile manipulator with either of these damping functions can be adjusted by an increase or decrease in the overall speed of the motion. An increase in the overall speed by decreasing the damping at the minimum distance will decrease the time to complete the task at the expense of safety and flexibility. By increasing the damping at the minimum distance, and thus decreasing overall speed of motion, safety and flexibility are improved with an increase in the time to complete the task. By changing the speed of the robot using the damping functions, the metrics of time, safety, and flexibility can be adjusted to fit the environment.

Damping functions also have an affect on the final configuration of the arm. If the initial and final configurations of the arm are known, the damping function can be modified to produce an approximate target final configuration. Final adjustments of the arm and interaction with the part will occur in the micro-manipulation phase.

The affect of the arm waist limit on the acquisition of the goal is being examined. The approach to this problem should distribute the torque of the arm to the base when the arm is approaching its limit. By coupling these joints, the robot will be able to turn the base of the robot away from the goal, and bring the arm within reach of the goal. The actual robot has been driven using the identical code as in the simulations using damping functions.

Acknowledgments

The Mobile Robot Laboratory is supported in part by the National Science Foundation under grants #IRI-9113747 and #IRI-9100149, and the CIMS program and Materials Handling Research Center of Georgia Tech. The authors would like to thank the other researchers involved in this project especially Dr. Wayne Book, Jonathan Cameron, and Doug MacKenzie.

REFERENCES

[1] Arbib,M.A., T. Iberall, D. Lyons, "Coordinated Control Programs for Movements of the Hand", Technical Report 83-25, Department of Computer and Information Science, Univ. of Mass., Amherst, 1983.

[2] Arkin, R.C., "Motor Schema-Based Mobile Robot Navigation", *International Journal of Robotics Research*, Vol. 8, No. 4, August 1989, pp. 92-112.

[3] Arkin, R.C., "Integrating Behavioral, Perceptual, and World Knowledge in Reactive Navigation", *Robotics and Autonomous Systems*, 6 (1990), pp. 105-122.

[4] Arkin, R.C., Book, W., Lawton, D., Vachtsevanos, G., MacKenzie, D., Arya, S., Cameron, J., Gardner, W., Ramanathan, V., Son, C., and Ward, K., "Integrated Control for Mobile

Manipulation for Intelligent Materials Handling", in Georgia Tech Material Handling Research Center Report OP-92-19. 1992.

[5] Bizzi, E., F.A. Mussa-Ivalsi, and S. Giszter, "Computation Underlying the Execution of Movement: A Biological Perspective", *Science*, Vol. 253, July 1991, pp. 287-291.

[6] Brooks, R., "A Robust Layered Control System for a Mobile Robot", *IEEE Journal of Robotics and Automation*, Vol.2, No. 1, pp. 14-23.

[7] Cameron, J.M., D.C. MacKenzie, K.R. Ward, R.C. Arkin, W.J. Book, "Reactive Control for a Mobile Manipulation", *IEEE International Conference on Robotics and Automation*, Atlanta, Ga, vol 3, 1993, pp. 228-235.

[8] Cameron, J.M., "Modeling and Motion Planning for Nonholonomic Systems", *Ph.D. Thesis*, School of Mechanical Engineering, Georgia Institute of Technology, November 1993.

[9] Carriker, W.F., P.K. Khosla, and B.H. Krogh, "The Use of Simulated Annealing to Solve the Mobile Manipulator Path Planning Problem", *IEEE International Conference on Intelligent Robotics and Automation*, vol 1, 1990, pp. 204-209.

[10] Clark, R.J., Arkin, R.C. and Ram, A., "Learning Momentum: On-line Performance Enhancement for Reactive Systems", *Proc. 1992 IEEE International Conference on Robotics and Automation*, Nice, France, May 1992, pp. 111-116.

[11] Connel, J. H., "A Behavior-Based Arm Controller", *IEEE Transactions on Robotics and Automation*, Vol. 5, No. 6, December 1989, pp. 784-791.

[12] Firby, R.J., "Adaptive Execution in Complex Dynamic Worlds", *Ph.D. Dissertation*, YALEU/CSD/RR #672, Yale University, Jan. 1989.

[13] Foley, van Dam, Feiner, and Hughes, *Computer Graphics Principles and Practice*, second edition, Addison-Wesley Publishing Company, Reading, Massachusetts, 1990.

[14] Gat, E., "Integrating Planning and Reacting in a Heterogeneous Asynchronous Architecture for Controlling Real-World Mobile Robots", /it Proc. AAAI-92.

[15] Georgopoulos, A. P., "On Reaching", *Annual Review of Neuroscience*, 1986, Vol 9, pp. 147-170.

[16] Goodale, M.A., Pelisson D., Prablanc C., "Large Adjustments in Visually Guided Reaching Do Not Depend on Vision of the Hand or Perception of Target Displacement", *Nature*, vol. 320, no. 6064, pp. 748-750.

[17] Hoff, B., and M. A. Arbib, "A Model of the Effects of Speed, Accuracy, and Perturbation on Visually Guided Reaching", *Control of Arm Movement in Space: Neurophysiological and Computational Approaches*, ed. R. Cammiti, Experimental Brain Series.

[18] MacKenzie, C.L., R.G. Marteniuk, C. Dugas, D. Liske, and B. Fi[illegible]