Synthesis of Tactical Plans for Robotic Excavation

Sanjiv Singh

Submitted in partial fulfillment of the
requirements for the degree of Doctor
of Philosophy in Robotics

The Robotics Institute
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213

January 1995

This research was sponsored in part by the USAF and NSF. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of USAF, NSF or the U.S. Government.

Abstract

This thesis describes an approach to synthesizing plans for robotic excavators. Excavation tasks range from loading a pile of soil to cutting a geometrically described volume of earth— for a trench or foundation footing. The excavation task can be stated in terms familiar to researchers in robotics and artificial intelligence: Transform the world from its current state to another state. Two important characteristics, however, distinguish the excavation domain. First, soil is deformable, and, hence, a complete state-space description of terrain to be excavated requires a high-dimensional representation. But, for the state-based representations that traditional planners use, very large spaces are simply infeasible. Second, the response of soil varies immensely, and in general, it is not possible to analytically describe the mechanics of soil motion and its interaction with tools. A soil's shear strength depends not only on its physical and chemical makeup, but also on factors such as the compaction experienced in the past. Thus a robotic excavator is forced to deal with approximate and incomplete models of its actions and their results.

This research has developed and demonstrated a unified approach to deal with each of these issues. Instead of posing the planning problem in state space, this thesis employs a dual representation called "action space." An action space is spanned by parameters that abstract the actions that a robot excavator might perform. This formulation allows posing an excavation task as a problem of constrained optimization over the space of prototypical, one-step excavating plans. To reason about resistive forces encountered while digging, this work has developed a method that learns to predict resistive forces encountered during excavation. The action space representation allows incorporating goal configuration and geometric and force constraints within a single framework. Planning is thus reduced to finding a subset of plans that meet the constraints and optimize an appropriate performance measure.

Over 400 experiments were conducted on a specially-developed testbed to validate the action-space/force-prediction approach. The testbed robot has a hydraulically manipulated shovel and sensors that measure both contact forces and terrain contour. Given a geometric specification, the current implementation can excavate a trench to predictable tolerances. Testbed results reinforce the notion that a force-based model is essential to successful robotic excavation. These findings also point to the necessity of a lower-level control mode that can modify excavation plans based on the nature of the terrain encountered.

Acknowledgments

Foremost I would like to thank Reid Simmons for six years of advising. Reid allowed me much freedom to work independently, asked tough questions, and read the drafts of my thesis closely. His guidance in writing the thesis was extremely helpful.

Red Whittaker has fired my imagination in robots working in the outdoors for the past decade and remains an inspiration. Thanks to him for nurturing the early work and for setting me on track to investigate soil mechanics. Thanks to Dan Koditschek for the constant exhortation to be rigorous. Dan talked to me about the utility of demonstration and also the need to understand the limits at which methods fail.

Special thanks to Randy Brost, long time friend, roommate and supplier of high quality New Mexican green chiles, for many insightful comments. The seeds of the action space approach were planted directly as a result of an early conversation with him. He also pointed out the use of Reuleaux's analysis to compute rotation centers, which has proven to be very useful. Thanks to Thanasis Kehagias for his friendship and philosophy at moments of despair. Thanasis convinced me that more than a matter of jumping through the hoops, a Ph.D. can also be a matter of mastering a craft.

Thanks to Herman Herman for the collaboration on the testbed. Many experiments would not have been possible without his work. Thanks to Regis Hoffman for help with generation of terrain maps, Chuck Whittaker for help with calibration of the laser scanner, and David Bourne for making the T3 robot available for experiments.

Thanks to Andrew Moore for help with memory based learning and Justin Boyan for help with neural nets. Thanks to Mike Trick, Paul Christiano, Yangshen Xu, Mike Erdmann, Tony Stentz and Matt Mason for comments on the research. Thanks to Jim Martin and Bob Smith for logistical support at the lab. Thanks to Dave Wettergreen for years of answering Framemaker questions and for reviewing the introduction. Thanks to Roy Taylor for advice on writing style and tight editing of sections of the thesis.

Thanks to my officemates Dave Wettergreen, Herman Herman, Dimitrious Apostolopoulos, Mark Ollis, Gerry Roston and Mike Heiler for fascinating conversations about gun control, volcanoes, range imaging, martial arts, night life in Alaska and the Greek military, to name a few. Thanks to the CMU Dinner Coop for seven years of great food and camaraderie.

Thanks to my parents for giving me the room to follow my dreams and for their patience with the perpetual "six more months". Finally, I owe much gratitude to Sonia, my wife, for the love, the company, for putting up with the long hours, and for being the type of person that the Ph.d. was the obvious thing to do.

Table of Contents

Chapter 1 Introduction

> **excavate** (EK'skuh-vayt') v. --tr. 1. To make a cavity or hole in; hollow out. 2. To form by hollowing out. 3. To remove by digging or scooping out. 4. To expose or uncover by or as if by digging. --intr. To engage in digging, hollowing out, or removing. Etymology Latin. excavare, excavat-, to hollow out: ex-, out + cavare.
>
> American Heritage Dictionary

Although excavation is ubiquitous in the construction industry, most day-to-day operations proceed on technology that is decades old— technology that has not kept pace with other industries. A recent trend towards greater automation of excavation machines reflects a larger movement in the construction industry to improve efficiency. Currently, human operators require ten to fifteen years of experience before they can be considered experts. Their work is often dirty, strenuous and repetitive. In effect, manual operation of an excavating machine can be an expensive proposition. Even without the economic motivation, automated earthmoving machines are needed in worksites that are hazardous for humans. The most immediate application of autonomous excavation is in cleanup operations at sites where toxic and nuclear wastes have been stored. Many cleanup efforts are progressing slowly because they expose human workers to substantial risk. Another application is excavation on the Moon and Mars. NASA studies conclude that excavation will be one of the first activities necessary to establish planetary habitats [NASA89, Register90]. Since manned

operation in space is extremely expensive, a reduction of manpower as a result of automation is beneficial. Automation has the potential of reducing the level of skill required, improving safety and decreasing monotony.

Consider the following problem: An excavating machine, such as one that might be found on a construction site (Figure 1), is to be completely automated. Imagine that an operator is able to specify the desired geometry of the terrain and the automated machine (a robot excavator) can transform the terrain without any further assistance or intervention.

Figure 1 An excavating machine commonly used at construction sites.

The task as addressed in this thesis can be stated in terms familiar to researchers in artificial intelligence and robotics. The world is in some state and must be transformed into some other state. To accomplish this, an excavating robot must have two capabilities. First, it must be able to discriminate between states of the world. Second, it must be able to act in response to the perceived state of the world. How a robot discriminates between states, how the states of the world are mapped into action, that is, how the problem is posed, what the inputs and outputs are, and how the mechanism is controlled, that is, how various parts of the implemented system are organized, are all questions that the designer must consider. This thesis develops and demonstrates an approach for an excavating robot to map world state into action aimed at achieving a specified goal. The approach boils down to a method of

constrained optimization over a space of feasible robot actions. We have built a robotic excavator to validate this approach. This thesis develops the rationale for the approach, and discusses the results on our experimental excavator.

1.1 What makes this problem different from others?

This domain is distinguished in two important ways. First, soil, the media that an excavator must interact with, is deformable. This means that it is not simple to describe the state of the world as a snapshot in time. This is because the number of variables required to uniquely describe the state of even a small patch of terrain is immense. In the worst case, the position of every distinct particle of soil must be accounted for. Even with geometric approximations and simplifying assumptions, the number of variables required are two or three orders of magnitude larger than in other domains that have been automated. The implication of a very large state space is that a traditional state-based representation will be impractical. A new way to pose this problem is necessary.

Second, the interaction between an excavating tool and soil is complex and hard to describe analytically. For a particular action, say for example *move a cutting surface along angle,* α, *through terrain that has properties* β_i", it is usually not possible to accurately describe the effect of the action on the terrain. This is chiefly because the response of soil is based not only on its physical and chemical makeup, but also factors such as stresses and strains that have built up over time. Soil has been analyzed only in controlled settings and no unifying theory of soil mechanics relates actions to resulting terrain states. The terremechanics literature provides approximate partial methods such as models to predict the resistive forces experienced by simple tools moving through homogeneous soil. Finite element methods provide more detailed answers at the cost of careful modeling of soil and large amounts of computation. Realistically, there is no hope of modeling soil accurately to answer the types of questions we are interested in. The computational requirements of a hypothetical soil model that accurately accounts for geometry and force, prohibit its usage in any robotic system that had to behave in a timely fashion. The implication is that a robotic excavator must deal with approximate and incomplete models of the results of its actions. It will be even less useful to consider long sequences of actions—the associated uncertainty will quickly render such estimation useless.

1.2 Excavation as a planning problem

Above, I stated that a robotic excavator, like other intelligent agents, must be able to discriminate between states of the world and be able to affect useful change on the world.

Although colloquially, the term “planning” has a narrower definition, for the purposes of this thesis, it is defined as the mapping between state of the world and the actions that a robot invokes. This section discusses some of the forms that this mapping might take.

Traditionally, the job of a planner is to transform the world from an initial state to a goal state using a set of primitive actions that are directly executable. Commonly, planners are designed to seek a point in a search space that is defined as a solution. The definition of the search space varies; some planners define points in the search space as states of the world while others define points in the search space as actions. Depending on the application, a state description may include variables to describe position, velocity, force or other relevant parameters. A point in state-space corresponds to a unique state of a robot (and possibly the world). The world state can be traversed by application of an applicable operator to a particular state, leading to a different state. Thus the solution is a sequence of operators that traverse the search space from an initial state to a goal state. Alternatively, we might plan using an action-space which is defined by the range of parameters used to define a robot action. We might specify a robot action by a set of actuator positions or, more abstractly, parameters (such as a direction to move, or an end-effector stiffness) that imply a prescription for the actuators. There is a well-known duality between states and actions and as with most dualities, the choice of one representation over another doesn’t affect any essential capability for expression. However, sometimes, a particular representation makes certain properties more natural to express and reason about.

Another traditional view of planning is that it should be hierarchical. This view has developed in part because examples of hierarchical systems abound in our world. Likewise, it might be argued, that planning should be hierarchical. At the very highest level a planner ought to abstract the world and worry about goals while lower levels incrementally flesh out the directives. Some researchers have argued that it is sufficient (and perhaps necessary) to abstract the actions of a robot into high level constructs devoid of implementation details such as control of actuators. This allows planners to work with statements like <MOVE A, B>. That is, at a high level, a planner need not worry about how the robot moves from A to B and can construct plans based on such atomic actions. It was argued for many years that lower level modules were analogous to computer programs written in assembly language. The thinking went that development of artificially intelligent systems, like the development of software systems, must be conducted at a higher, more abstract level. While most complete robotic system embody some form of hierarchy, there is much contention in the artificial intelligence literature as to where the abstraction boundaries should be drawn and what this implies about the organization of elements of intelligence.

To make this discussion concrete, consider an example of an entire task that a robotic excavator might have to perform— having a robot automatically dig a footing for a foundation of a building. The robot must excavate the footing to some acceptable tolerance, that is, it must completely remove the specified volume, without disturbing nearby terrain (to ensure structural integrity). Assume that the volume that must be excavated is large enough that the robot must move around to be able to accomplish the task. Here are some (of many) tasks that the robot must accomplish to be successful.

- The robot must be efficient, that is, it must complete the task in a reasonable number of steps.
- The robot must not get stuck by "painting itself into a corner". That is, it must not dig in such a manner that keeps it from completing the task.
- The robot must take into account the shape of the terrain where it digs.
- The robot must control its limbs so as to perform scooping, respecting the constraints of the mechanism and the geometric specification
- The robot must be able to cope with terrain that is stiffer or softer than expected.

There is a qualitative difference in the nature of these sub-tasks. Consider a continuous progression between two extreme types of tasks (Figure 2).

Activity	Deliberation	Action
Domain	Geometry	Dynamics
Constructs	Shapes	Forces
Scope	Global	Local
Cycle time	Slow	Fast
Computing	Centralized	Decentralized

Figure 2 A continuous spectrum along which specific robot tasks might lie. A traditional view is that goals start at the strategic end, and are incrementally refined until they reach the end at which action is controlled.

At one end of the spectrum is strategic planning. At this end, planning tasks deal with problems with global properties. For example, the problem of planning the motion of the excavator such that it does not get stuck, has a global quality to it. It is not possible to solve this problem unless the geometry of the entire worksite is considered. The nature of the

problem is deliberative, that is, a robot must reason about the future to select its next action. The task is not time-critical and, for the most part, the robot must deal with geometry. On the other hand, the task of ensuring that the robot can deal with terrain that is stiffer or softer than expected has a more local nature to it. Given a nominal trajectory, or a contact force to maintain, the task must be executed at sub-second intervals.

Hierarchical planning provides a systematic framework to break a goal into a sequence of actions. However, exactly how the tasks are layered along the strategic—reactive spectrum and how these subproblems are couched, is a matter of much discussion. It is possible to pose parts of (or in some cases the entire) problem in many different paradigms. Researchers have posed planning as problems in physics (newtonian mechanics), biology (behavior based systems, genetic optimization), math (gradient functions, constrained optimization), computer science (search), control (stability and convergence), probability (markov processes and uncertainty propagation), and logic (predicate logic).

This thesis assumes a hierarchical framework. That is, it is assumed that a strategic planner is able to decompose the task of excavating a large volume into reasonable subgoals. The planner described in this thesis produces nominal trajectories for a robot to execute. In an extended scenario, the planner would also specify parameters that allow the regulation of a high-bandwidth interaction between the robot and the world.

1.3 Scope of this thesis

The task of excavating a large volume of soil, as discussed in the previous section, can be broken down into three large problems. First, an autonomous excavator must tessellate the entire worksite into patches, each of which can be excavated from a unique setup location. These patches are constructed based on the excavator's workspace and ordered in an efficient manner. Since excavating machines are planar, these patches must be further broken down into smaller swaths (trenches), that are as wide as the excavating tool. The task of tiling the worksite into these swaths is called the *coverage* problem. A coverage planner must reason about the geometry of the worksite and the mechanism of the excavator. Without question, the coverage planner must incorporate some notion of the type of soil that must be excavated, although, perhaps only in a very approximate manner.

The second task is to plan single digging operations given the state of the terrain. That is, at this level, the planner must decide where to dig and how to dig within the geometric limits of the swatch specified to it. At this level, the planner must reason about the kinematics of the mechanism, the shape of the terrain, the resistive forces experienced during excavation and the torques that can be delivered by the actuators. The output of this level is a

nominal trajectory, and a specified interaction between the mechanism and the environment. This level of planning is referred to as Tactical Excavation planning.

The third task is to execute a nominal trajectory of the end-effector, modifying it as the terrain is found to be stiffer or softer than expected. At this level, a planner (conventionally called a “controller” in this context) must implement a dynamic relationship between excavator end-point position and force rather than just control these variables alone. This capability is referred to as Low Level Control.

This thesis, has devised a working excavating robot, but focuses on tactical excavation planning. Goals are specified as single swaths (as wide as the excavation tool) and the execution is currently performed using position control. The other two tasks are important and constitute a major part of future work.

1.3.1 Philosophy

The philosophy espoused by this thesis can be expressed by two tenets.

- *Encode the task, not the mechanism*. There are many ways in which the problem of excavation could have been posed. Among them, are many methods that are specific to particular types of mechanisms. Plans made in the space of joint motions for one excavator are likely to be very different from those constructed for another mechanism, although they might perform the same task. This thesis proposes a planning methodology that encodes the task directly— the plans dictate motion of the tool, and joint trajectories are generated as a consequence.

- *The more reasonable the plan, the more likely a robot is to be able to perform the action*. This thesis rests on a detailed analysis of mechanisms, soils and their interaction. Considerable attention has been spent on generating force and shape models. One question often asked is “why not just put the shovel in the ground and start digging?”. The answer is that it is a matter of efficiency. This thesis contends that to be efficient, a robotic excavator must explicitly consider excavator mechanism, terrain shapes and interaction forces.

1.4 Summary of Approach

Above, we saw that it will not be tractable to define points in a search space as states of the world. Instead, we explore the space of feasible actions that a robot can execute. That is, we look at the space of all possible digs that the robot can execute and choose one that meets specified criteria. Since there are an infinite number of trajectories that the robot might execute at every dig, we restrict our attention to a subset of digging trajectories described by a compact set of discretized variables. Now, the problem of planning can be posed as constraint optimization in the space of feasible actions.

Assume that an autonomous excavating robot is equipped with a perception system that provides information about the state of the terrain to be excavated and a digging implement (called excavator *bucket*) that can be made to follow a prescribed trajectory. The one-dig problem is formulated as follows: *Sense the terrain. Of all possible digs, consider a subset that the robot can feasibly perform and out of those, choose one that satisfies a bound on a cost criterion. Execute the dig.* Complete excavation is accomplished by concatenating a sequence of such plans.

Let us explore this method in greater detail. To start, let us develop the notion of an action space. We will consider digging actions that are parameterized by one or more variables. The values that these variables can attain define a multi-dimensional space of possible actions. For example, consider the space formed by the variables that describe the dig shown in Figure 3. That is, all digs considered are described by an approach angle (α), a height at which the bucket enters the pile (h), and a dig distance (d). Since this parameterization might not specify a unique dig if the terrain is undulating, for the sake of this example, the dig that is closest to the foot of the pile is chosen.

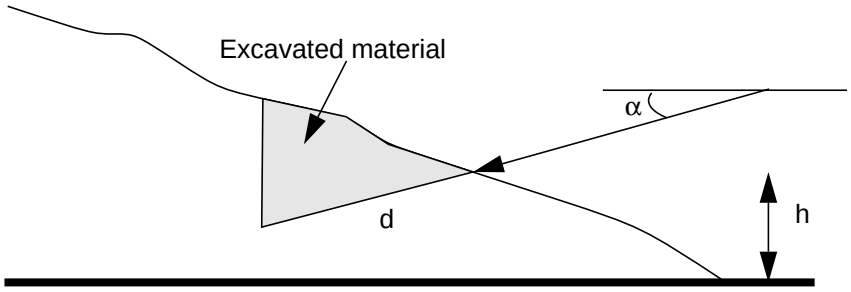


Figure 3 A prototypical dig

In the above example, a three dimensional action space (α, d, h) represents the set of all digs under consideration, independent of the terrain and the excavator. Certain digs are excluded because they are not feasible. A dig might be infeasible because it violates geometric or force constraints. Geometric constraints are relatively simple to specify. A dig may be infeasible if the excavator is required to reach outside its workspace, or, if the dig intrudes past specified geometric boundaries, or, if it sweeps a volume of soil that is clearly much larger than that can be accommodated in an excavator bucket. Force constraints are harder to specify because they depend on the resistive forces that are developed during the motion of an excavating tool passing through soil. These resistive forces are difficult to model a priori and in response this thesis proposes a method to learn to predict resistive forces through experience. This method allows prediction of resistive forces for candidate digging trajecto-

ries. These forces are transformed into torques that are experienced at the joints of the robots. A candidate digging action violates the force constraint if it requires greater torque than can be developed at the joints.

Among the set of actions that satisfies all the constraints, it remains to find one dig that satisfies a threshold on a cost function, for example, the amount of soil excavated. Figure 4 shows the action space spanned by the variables (α, d, h) and a hypothetical surface that constrains the set of feasible digs for a particular excavating robot and a particular terrain. Here, the constrained volume lies beneath the surface shown. Note that the constraint surfaces change as the excavation proceeds. In general, the constrained volume need not be a single, connected space.

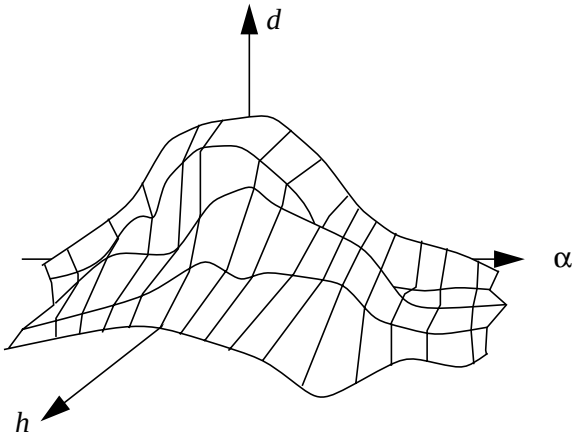


Figure 4 A schematic representation of a three variable action space. Each point in this space represents a complete dig. In this case, the set of feasible digs lies below the surface.

Each point in the action space can be classified as: *infeasible*, due to the constraints posed by the world, *feasible but suboptimal*, or, *optimal*[1] If the action space is small it is possible to enumerate all the options by discretizing the space. Otherwise, as is often the case for describing realistic digs, a numerical method is necessary. Since there is no guarantee of a unique extremum in the cost function, a method like simulated annealing can be used optimize the cost function. Once the optimization procedure has selected a dig, it can be mapped back to the joints of the excavator.

There are two advantages of formulating the problem such that the action space is independent of the configuration of the digging machine. First, it provides generality. Consideration of a variety of machines and terrains results in different constraint surfaces on the

1. An optimal action satisfies a threshold on the cost criterion.

parameter space, but the space itself does not change. Second, digs parameterized as a combination of task variables are easier to analyze than those that are represented as trajectories in the joint space of the robot actuator.

It is not surprising that the one-step plan suffers from problems common to greedy methods— it selects the dig that optimizes the cost function at the next step only, and in general, will generate a sequence that is suboptimal. However, it is not possible to achieve global optimality without looking ahead all the way to the goal. Since forward models of excavation are approximate, and the uncertainty associated with sequences of action is large, it is futile to seek global optimality.

The approach described in this section has been implemented for two common excavating tasks— loading (a pile of soil must be leveled) and trenching (a trench of specified dimensions must be created). The tasks differ slightly in their nature and the mechanisms typically used. This thesis presents simulation results for the loading tasks. Only geometric constraints are used. The task of trenching has been implemented on our excavation testbed (Figure 5). The testbed consists of a hydraulic manipulator equipped with an excavator bucket. Both geometric and force constraints were developed for this task.

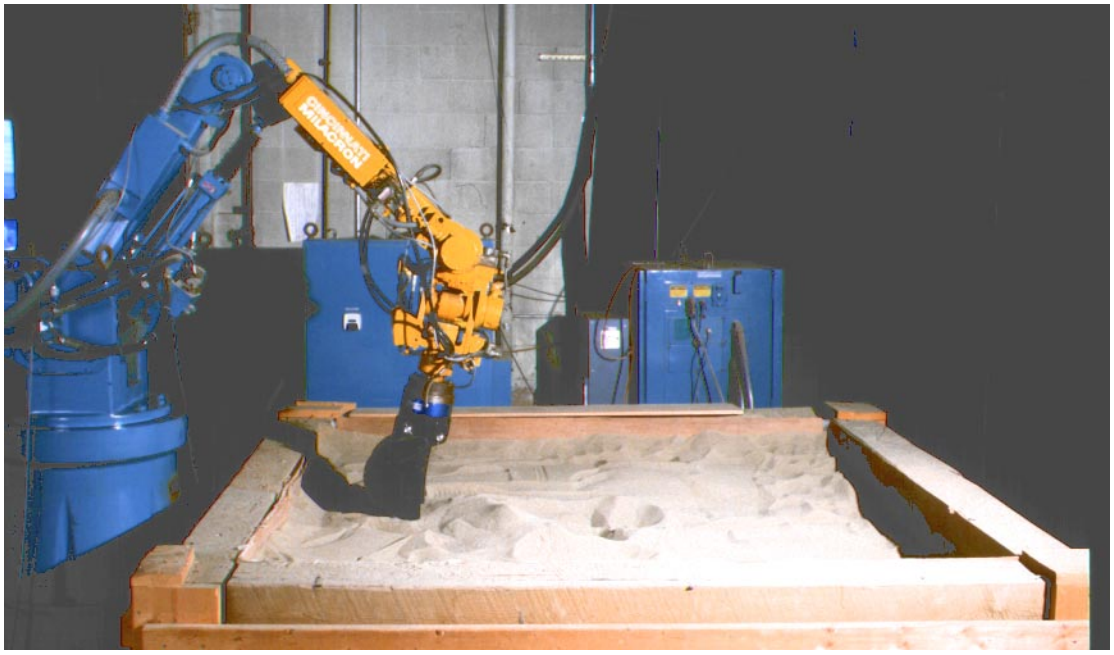


Figure 5 Experimental testbed developed for this thesis.

1.5 Summary of results and contributions

This thesis has developed and demonstrated a unified approach to deal with each of these issues. Instead of posing the planning problem in state space, this thesis employs a dual representation called “action space.” An action is parameterized by a compact set of parameters that are chosen through a thorough examination of the task. This thesis suggests methods to develop the encoding of excavation tasks based on observation of the human operators as well a contact analysis. This formulation allows posing an excavation task as a problem of constrained optimization over the space of prototypical, one-step excavating plans. To reason about resistive forces encountered while digging, this work has developed a method that learns to predict resistive forces encountered during excavation. The action space representation allows incorporating goal configuration and geometric and force constraints within a single framework. Planning is thus reduced to finding a subset of plans that meet the constraints and optimize an appropriate performance measure.

Over 400 experiments on a specially-developed testbed were conducted to validate the action-space/force-prediction approach. Given a geometric specification, the current implementation can excavate a trench to predictable tolerances. Testbed results reinforce the notion that a force-based model is essential to successful robotic excavation. These findings also point to the necessity of a lower-level control mode that can modify excavation plans based on the nature of the terrain encountered.

1.6 A Road-map to the thesis

Chapter 2 discusses related work done by other researchers. Both technologies that are relevant to the proposed approach and alternate systems developed by other researchers are discussed. Chapter 3 introduces the notion of action spaces and shows how the representation for excavation tasks is developed. Chapter 4 deals with geometric constraints. This chapter shows how the set of geometrically feasible plans are developed. Chapter 5 discusses force constraints, starting with a review of soil mechanics. Several learning methods and representations are compared. Chapter 6 discusses the numerical methods used to perform the constrained optimization. Chapter 7 discusses the experimental system devised for this thesis and presents the results achieved. Chapter 8 summarizes the conclusions and contributions of this thesis.

Chapter 2 Background

This chapter focuses on two areas. First, it discusses previous work by other researchers that is directly related to autonomous excavation. This includes work done on other excavation systems, as well as research in soil mechanics. Next, attention turns to the relationship between the ideas proposed in this thesis and other work in robotics and artificial intelligence at large.

2.1 Previous Work Related to Automated Excavation

2.1.1 Studies on automated excavation.

There have been numerous studies that have explored the feasibility of automating excavation. Many of these studies have considered extraterrestrial scenarios [Bernold89, JSC89, Toups90, Boles90]. NASA is interested in setting up lunar and martian habitats for humans, and it is expected that automated excavators will do much of the work before humans arrive. Another set of studies has examined the utility of automated excavation on Earth [Sanvido83]. To date, these studies have been limited simply because there are only a few examples of autonomous excavators.

2.1.2 Soil Mechanics

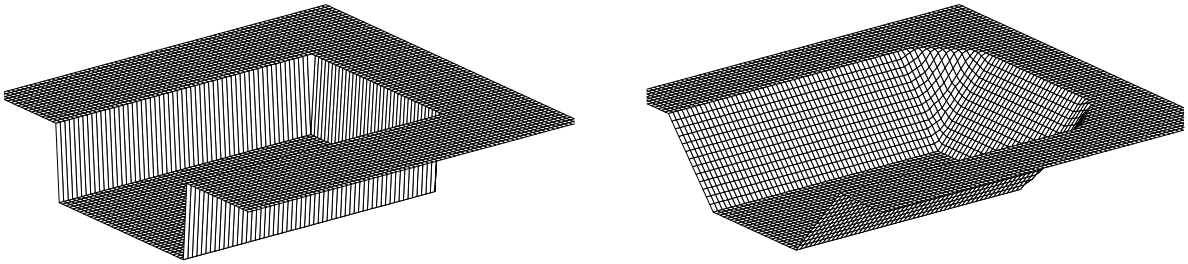
By its nature, excavation involves forceful interaction with terrain. The nature of this interaction is most influenced by soil properties. Intuitively, it is obvious that digging in loose, dry sand is very different from digging in a compacted, clayey medium. Indeed, this difference can be so large that strategies for digging in various media differ radically. Chiefly, we are interested in the mechanics of excavation. Ideally, we would like a process model that would model the effect of a robot's action. At the very least, we would like to know the effect of the world on the tool. That is, we would like to characterize the interaction between a tool (an excavator bucket) and soil. The literature in this area spans several fields—mechanical, civil and agricultural engineering.

The first major question posed is "what is the effect of a robot's action on the world?" In the excavation domain, we ask: what happens when a bucket sweeps along a trajectory in the soil? What forces are involved? How does the soil move and settle? How much of the soil ends up in the bucket? It should be clear that no closed-form models exist to answer such questions since soil is complex enough that the differential equations required to describe evolution of the soil mass over time are local in their effect not global. Additionally, no analytical solutions exist and numerical integration must be used to determine evolution.

The second major question is "what is effect of the world on the tool?" That is, we are interested in knowing the resistive force experienced by a bucket as it excavates. One method to answer the two questions posed above is to use finite element methods (FEMs). FEMs provide a method to model a large system composed of many elements (possibly very irregular) with a large number of partial differential equations. Indeed, some researchers have focused their attention on FEMs to answer the types of questions we have raised [Cundall88a, Cundall88b]. Yong and Hanna have specifically studied the performance of a flat blade moving through short distances (less than one foot) in clay soil [Yong77]. They developed a FEM that provides detailed information on the stress and deformation of the soil as well as the forces developed at the tool. The advantage of using these kinds of methods is that they allow estimation of both forces experienced by the bucket as well as geometrical effects. Unfortunately these methods are so taxing in computational resources that their use is prohibitive for our purposes since it is possible that many hundreds of digs must be evaluated before the robot is moved once. Given that a single FEM prediction might take minutes, the use of such methods does not seem tractable.

If requirements are relaxed, the finite element methodology becomes more attractive. For instance, if we need a model to approximate the motion of soil for purposes of simulation, we could use some methods that are inspired by finite element analysis. These methods discretize the world into a two or three dimensional grid. Iterative algorithms are applied to

the grids until equilibrium is reached. Homma et al propose a volumetric model to model the settlement of soil [Homma90]. Terrain is tessellated into a three dimensional grid and each cell is marked as occupied/empty. At each step of the simulation, every grid cell is checked to see if the contents will stay unchanged, or move to another cell, using a few simple rules. The rules are repeatedly evaluated until all cells achieve equilibrium. This method is slow simply because it is three dimensional and requires treatment of a large number of cells. Puhl has described a similar method that makes the assumption that the elevation of soil is a single-valued function, that is, no voids exist in the soil [Puhl92]. This allows for a two dimensional treatment of the soil surface and is hence much faster (quadratic instead of cubic) to compute. Figure 6 shows how this a modified version of this method[1] could be used to predict settlement of soil after a volume has been removed. While the method assumes instantaneous removal of soil, the results seem to approximate the settlement seen after a bucket has removed soil along a trench[2]. This model has been used in simulation experiments conducted for the dissertation.



(a) (b)

Figure 6 Settlement of the banks of a trench using a modified version of Puhl's method. (a) before settlement (b) after settlement. The angle of repose of the walls is 30 degrees. The surface consists of 10,000 facets. This example required approximately 15 seconds to compute on a Sparc 10 workstation.

Li and Moshell have developed a sophisticated simulation of soil movement [Li92a]. They use theory of slope settlement from the civil-engineering literature to model soil slippage. Relaxation is based on calculation of forces acting on small sections of soil in a method that is closely related to the finite element scheme. They have extended their analysis of soil settlement for simple soil-tool interaction. However, their model ignores some important phenomena in the shearing of soil. Since the criteria for their models is that the "simulations look very realistic" it is hard to tell how accurate these models are.

1. Puhl's method only allows for an angle of repose of 45 degrees. This method has been modified to simulate an arbitrary angle of repose.
2. Presuming the soil is homogeneous, dry and uncompacted.

The above three models can run roughly in real-time with today's technology but even that is too slow for the purposes of this thesis. Since a model that can be used to select between hundreds of actions is needed, it will be necessary for the model to run two to three orders of magnitude faster than real-time.

A larger literature exists if we restrict ourselves to estimating the resistive forces experienced by a tool as it moves through the soil. One body of work has attempted to estimate cutting resistance for various types of earthmoving machines, based on empirical results for those machines [Alekseeva85, Zelenin86, Nedoredzov92]. Various indices are suggested based on the configuration of these machines and computation of resistive forces can be roughly estimated by numerically integrating simple equations. The problem with this body of work is that rather than using first-principle mechanics, it relies heavily on empirical results to capture characteristics such as the ability of a particularly shaped tool to penetrate into the terrain. Hence, it is not clear how extrapolate to arbitrary mechanisms and soil.

In contrast there exists another body of work in the field of agricultural engineering, that attempts to estimate cutting forces based on first principle mechanics. This work is motivated largely by the need to estimate the forces experienced by tools used to perform tillage. The seminal ideas in this work come from the civil engineer's load bearing equations for foundations [Terzaghi47]. These models were extended by several researchers who recognized the similarity to the case of a blade moving through soil [Reece64, Siemens65, Hettiaratchi67, Gill68, Luth65]. These researchers added provisions to account for a variety of tool geometries and orientations. This work is examined in greater detail in Chapter 5; for now it suffices to state that the resultant models use parameters of soil-soil friction, soil-tool friction, soil density, tool depth and tool orientation to obtain order of magnitude predictions of the resistive forces developed in the use of agricultural tools. Although these models will not be enough for our means, there is much to be learned from the analysis. It will provide insights into helping to put together a better model to estimate the resistive force experienced in the excavation scenario.

Any model that predicts resistive forces will suffer in accuracy if nominal values for soil and soil-tool parameters are used. To be accurate it is necessary to experimentally determine these values. Several instruments are available to experimentally determine these properties in a laboratory [McKyes85] although it is obvious that *in situ* testing will provide better results [Wroth84]. Wadhwa recognized that a better prediction of resistive force might be obtained for a tillage tool when the soil parameters are estimated using a scaled version of the tool itself, rather than using other devices [Wadhwa80]. Similarly, other researchers

have suggested active use of a robot to determine the nature of material the robot is to interact with [Krotkov90, Homma92].

This thesis proposes a method to predict the resistive forces that draws on the analysis in soil mechanics developed by agricultural engineers to develop the structure of the resistive force prediction model. The method then uses experimental data to produce better predictions than are possible with the methods discussed above. This work is related to recent work in the area of robot learning, which is discussed below.

2.1.3 Analysis of Mechanisms

In the literature on autonomous excavation, there is a body of work is concerned with modeling of excavating machines. The relevant work can be divided into studies that address kinematics (geometrical relationships) and those that address dynamics (force and acceleration). This section briefly reviews both.

2.1.3.1 Excavator Kinematics

Several researchers have worked out the kinematic relationships that relate joint angles of a backhoe to the pose of the tip of the bucket [Seward88, Vaha91, Koivo92, Bernold93]. Hemami and Daneshmand have developed a similar kinematic model for a front loader [Hemami92]. Since most excavator mechanisms are not directly driven at the joints, but rather are powered by hydraulic cylinders attached to the limbs of the machine, a few researchers have further developed the transformations between the joint angles and cylinder positions [Koivo92, Hemami92]. The kinematic models used for the purposes of this thesis can be found in Appendix 1 and Appendix 2.

The forward kinematic relationship that relates joint angles to the positions of the limbs and the bucket is most useful for simulation of the movement of a machine. Given a path for the excavator bucket to follow, the inverse relationship provides the joint angle reference trajectory for the machine. Further, the inverse relationship provides a method to determine if a hypothetical pose of the bucket is reachable. A pose of the bucket might not be reachable because it requires joint angles beyond the limits of the machine or because it is outside the workspace of the robot. Both of these conditions are detectable through the inverse kinematics solution.

2.1.3.2 Excavator Dynamics

Several researchers have proposed dynamical models for an excavating machine following methods developed for robot manipulators. The purpose of such models is to relate joint torques to the motion of the limbs of the excavator. Given the configuration of the mecha-

nism and desired end-effector motion, the inverse model provides a reference joint torque trajectory. Vaha and Skibiniewski have proposed a model based on the Newton-Euler method of dynamical modeling for robot manipulators [Vaha91b, Vaha93]. This type of a model is meant to capture effects of inertial (centripetal and coriolis) forces that are typically relevant at high accelerations. Since most excavators move with low accelerations, it may be that a simpler static force analysis that captures the torques created by the weight of the limbs of the machine will suffice. The authors do not present implementation results and thus this question remains open for the future. It is also noteworthy that such modeling ignores an important dynamical effect— the resistive force experienced by the bucket due to contact with the soil. The torques due to these forces can be an order of magnitude larger than those due to joint accelerations and weight of the limbs. Hemami and Daneshmand use a static analysis to compute the torques experienced at the joints as a result of the forces at the bucket and then show how these are related to the forces at the cylinders using trigonometric relationships [Hemami92].

2.1.4 Automated Excavation

Continuing with the philosophy of hierarchical decomposition, the task of excavation can be divided into a number of levels and sorted by increasing levels of abstraction. This subsection examines other automated excavation systems and attempts to locate them in a hierarchical framework. At the lowest level are teleoperated machines. The operator is removed from the machine but is still required to control the joints much in the same manner as the original machine is controlled. In some cases, the operator controls a “master”, a scaled down replica of the excavator which itself behaves like a “slave”. The advantage of using a “master-slave” system is that the control of the excavator is more intuitive. Instead of controlling the joints individually, the operator controls the end effector (excavator bucket) directly. Several excavating systems (in fact, the large majority of systems built to date) have focused on the task of servoing the excavator along a path that has been prescribed to it, leaving the generation of the excavator trajectory to a higher level module or a human. It is important to ensure that the robot follows the prescribed trajectory accurately, and more importantly, is able to deal with varying soil conditions. For example, if the soil is stiffer than expected, the robot should alter its trajectory so as to not stall the actuators. The next level of abstraction is “supervisory” control. At this level, a human operator makes some of the decisions and the robot takes over after a certain point, typically carrying out a straightforward set of actions. At the next higher level, are systems that actually generate the trajectory that the excavator bucket must follow to perform a dig. At the highest level, is the strategic or site planning. At this level of abstraction, the task of performing an entire excavation, such as digging a foundation, is broken down into set of subgoals.

2.1.4.1 Tele-operation

In lieu of complete autonomous systems to perform excavation, several tele-operated systems have been built over the years. Wohlford et al describe a backhoe that has been teleoperated for dealing with buried hazardous wastes [Wohlford90]. Similarly, Burks et al describe a modified tracked excavator that has been developed to deal with buried wastes [Burks92]. Nease describes an Air Force program that has automated an excavator for the purposes of tele-operated repair and maintenance of runways [Nease92]. Kojima et al describe a teleoperated backhoe that is used to dig deep foundations [Kojima90]. Recognizing that force information plays an important role in the control of an excavator, some attention has been focused on developing force-reflecting master-slave systems for excavation. Ostaja-Starzewski and Skibiniewski propose a method to provide force feedback for an operator controlling an excavator [Ostaja89]. Kraft TeleRobotics Inc. now sells a teleoperated excavator with a force reflecting master [Kraft90]. With such a device, the operator is able to feel the stiffness of the soil encountered by the bucket in the joints of the master manipulator.

2.1.4.2 Trajectory Servo

Given a trajectory for a robot excavator to follow, we would like the robot to faithfully execute the trajectory. There has been significant attention in the literature to producing robust trajectory tracking for robot manipulators and much of it is relevant to excavators as well. For example, Miller has looked at the optimal tuning of position control for a backhoe [Miller90].

The additional complication in the control of an excavator is that the interaction forces during contact with the terrain can be very significant. Several researchers have made this consideration their focus of attention. Bullock demonstrated a simple scheme to deal with the stiffness of soil [Bullock89, Bullock92]. He instrumented a small, four degree-of-freedom manipulator arm with strain gauges on one of the limbs. These strain gauges were monitored while a scoop attached to the end of the manipulator was pushed through the soil along a prescribed path. When the strain measurement exceeded a preset threshold, the scoop was backed up and a “scooping” operation was performed. Huang and Bernold have extended this type of control to deal with very stiff inclusions such as rocks in the soil being excavated [Huang93]. They use a scaled model of a backhoe that has been outfitted with a load cell between the bucket and the backhoe “stick”. An obstacle is indicated by a high reading on the load cell. When this happens, the bucket lifts up by a preset distance and continues parallel to the planned path. In the same vein is a controller proposed by Kakuzen et al [Kakuzen88]. They have developed a feedback and feed-forward controller for crowd control and level luffing of a crane.

2.1.4.3 Supervisory Control

While tele-operated systems require explicit control of the end effector, and require a large amount of dexterity on the part of the operator, several systems have attempted to implement a “supervisory” level of control. In this paradigm, the operator makes high level decisions, such as selecting the starting location of the excavator bucket. After this the robot excavator takes over and completes the rest of the dig [Yoshinada92, Sameshima92]. The idea here is to let the robot do what it is good at doing (servo to a well defined path) and use the human for those parts of the task that are not straightforward to encode into a controller. Sameshima and Tozawa have implemented a fuzzy logic controller that is motivated by observations of human operators controlling excavating machines [Sameshima92]. The operator places the bucket at a starting location. Starting from this position, three rules are evaluated (Figure 7) at every control cycle to determine the velocities of the boom, stick, and bucket joints.

	Observations		Actions		
	Bucket Vel	**Stick Vel**	**Bucket Vel**	**Stick Vel**	**Boom Vel**
RULE 1	L	L	B	B	M
	L	H	B	M	S
	H	L	S	M	S
	H	H	S	B	Z
	Bucket Angle		**Bucket Vel**	**Stick Vel**	**Boom Vel**
RULE 2	L		B	S	S
	H		S	B	Z
	Depth of bucket				**Boom Vel**
RULE 3	L				Z
	H				S

L:Low H:High B:Big M:Medium S:Small Z:Zero

Figure 7 Fuzzy Rules used by Sameshima et al.

The velocities commanded are dependent on the velocities of the bucket and stick at the previous control cycle, the bucket angle, and the depth of the bucket below the ground surface. The observations are classified as binary values (either low or high) and the outputs velocities are discretized into four steps (zero, small, medium and big). There remains a question as to how these rules are combined to produce the final command. The authors suggest a weighting scheme but it is unclear how the gains are adjusted for excavation in a par-

ticular soil. One of the conditions says that if the bucket and the stick are moving slowly (ostensibly because the soil is stiff), then the boom should be raised quickly (thus allowing digging at a shallower depth). If the bucket and stick move at a high speed, then the soil is deduced to be soft and the boom is kept still, allowing a deeper cut. This sort of system might be well suited to mass excavation since the scheme doesn't offer any way to dig a particular shape.

2.1.4.4 Tactical Planning

At the next level of abstraction, it is necessary to generate the trajectory that is to be executed. Koivo suggests a geometric method to plan trajectories when the ground plane can be assumed to be level [Koivo92]. The depth of a cut is adjusted so that the swept volume equals volume of the bucket. Since this method is geometric, it can not adjust the path of the bucket based on the type of soil that is being excavated. Feng et al have performed simulation studies to plan the trajectory of an excavator [Feng92]. They have proposed a method to select between parameterized digs. Although the details of implementation are quite different to those proposed in this thesis, this work is similar to the planner that proposed in this thesis.

2.1.4.5 Strategic or Site Planning

At the highest level of abstraction, the task can be stated as such: given a robot excavator, location and dimensions of a desired excavation, select a sequence of locations from which the robot can produce the overall desired shape. Romero-Lois and Hendrickson describe a system that they have implemented to address this issue [Romero-Lois89]. They customized an expert system (PLANEX) to plan the sequence of moves of a robotic power shovel equipped with sensors to determine the shape of the terrain and its own position. The planner operates purely on a geometric basis without consideration of the strength of the soil versus the forces that can be developed by the excavator. Similarly, although at a smaller scale, Bullock and Apte describe the generation of subgoals for an excavator [Bullock90]. The task is essentially to decompose a target volume into smaller volumes that can be tackled by an excavator. Both works described above relied only on simulation studies, and hence the resolution of the subgoals generated by a strategic planner for excavation remains an open question. It is also not at all clear that the abstraction of the site planning into a purely geometric basis is warranted.

Further we may ask how resources (excavating machines and hauling vehicles) may be best allocated to perform the overall task. Bernold suggests methods that could be used to schedule a fleet of excavators and haulage vehicles to perform the task of excavation most efficiently [Bernold86].

2.1.4.6 Other Systems

Several other systems that do not fit into the above classification are discussed in this section. A number of researchers have built rule based systems to perform excavation. A research program at the University of Lancaster, England has developed an automated excavator that uses a production system to sequence the actions required to dig trenches [Seward90, Seward92, Bradley93]. Digging is broken down into three phases— penetrate, drag and empty. Production rules are of the form:

```
if (penetration depth > 300 mm) then rotate bucket
if (xerror > 70) then y velocity = -2
```

An interesting aspect of this system is that it uses the servo error as a measure of the soil resistance. The authors report that their system has proven capable of autonomously digging a trench to a controlled depth in a variety of ground conditions.

Sakai and Cho have simulated excavation using a finite state machine [Sakai88]. They show how cylinder pressure and bucket velocity can be used to determine the phase in the digging cycle of the backhoe. A few rules are used to trigger moving from one state to another, and in effect cause changes in the trajectory of the excavator bucket. Lever et al have implemented a similar fuzzy-logic based system [Lever94]. They have used a wrist force-torque sensor and a small shovel mounted on a Puma manipulator to demonstrate digging through sand. Instead of looking at the velocities of the links, they look at the forces and torques observed during digging. Four rules based on force-torque readings are used to determine one of six actions— *lift out*, *up-forward-small*, *up-forward-large*, *down-forward-small*, *down-forward-medium* and *down-forward-large*.

Whittaker et al demonstrated a novel system that uncovered buried pipes using a vacuum tool mounted on a backhoe arm [Whittaker85]. Alternately, thin layers of soil are removed, and elevation maps are constructed using sonar sensors. Edge detection is performed on the elevation map and pipes are indicated when parallel lines separated by four to ten inches are found. Once a pipe is located, the vacuum tool follows the boundaries of the pipe until the pipe is completely exposed. Gocho et al describe the automation of a wheel loader working in an asphalt plan [Gocho92]. The wheel loader navigates by following buried underground lines between loading and dumping points. Loading is accomplished by dropping the bucket to the ground and driving into a pile of granular material until a threshold is reached in the hydraulic back-pressure. At this point, the loader curls the bucket and moves to the dumping point. Brooks et al have proposed that the task of planetary excavation might be performed more effectively by many, simple, small autonomous robots, rather than a single, large, complex robot [Brooks90]. They propose a system which would consist of 20 small bulldozers which work without explicit coordination or communi-

cation, but nevertheless cooperate to achieve tasks useful in a lunar construction site. However, no results exist for this type of system.

2.2 Relation to work in Robotics/AI

This section examines how the ideas proposed in this thesis relate to other work in robotics and artificial intelligence.

2.2.1 Formulation

One approach is to pose the task of excavation problem in a manner similar to other assembly problems— the world is in an initial state and must be transformed to another state. Indeed, it might appear that standard state-space methods could be used to generate a sequence of actions that will take the world from the initial to the desired state. Unfortunately, the nature of the excavation task makes it difficult to couch the problem at hand in the language of state-space methods. In the past decade, researchers in artificial intelligence have recognized the deficiencies in the use of state-space methods [Hendler90]. The main problem is that it is difficult to describe complicated causal and temporal relationships in the corresponding language, and that the search space grows exponentially. There is a similar problem in the treatment of excavation using standard state-space techniques, particularly because the dimensionality of the state space required to effectively describe the state of soil is extremely high, in fact many orders of magnitude higher than in problems involving manipulation of rigid bodies. Hence, another language is needed. It turns out that the approach proposed in this thesis— posing the problem in an action space— is similar in philosophy to recent work in mobile robot navigation [Daily85, Feiten94, Kelly94]. These researchers have found that they are able to pose the task of navigation more simply in a space spanned by variables that the robot can affect directly. Other researchers have combined state and action spaces for the purposes of fine motion planning [Brost88]. The issue of action space vs. state space is discussed in greater detail in Chapter 3.

2.2.2 Search and Optimization

For our purposes, a complete plan would involve the specification of all digging trajectories from start to finish. Although posing the task in an action-space will reduce the dimensionality of our search space, our ability to make a complete plan is dependent on two things. First, the length of the plans that we can make will depend on how coarsely we discretize the action space. If the action space is discretized into m points, an n step plan will require consideration of a maximum of m^n plans. Second, no good *process* models[3] exist for excavation. In most of the planning work done in artificial intelligence, the terms *action* and *event* are

used interchangeably; it is presumed that the effect of an action can be computed accurately and readily. In fact not only are the process models for excavation approximate, but they also require large computational resources. Given the uncertainty, the utility of a prediction based on a sequence of actions is furthermore questionable. In practice our planner can only look ahead by a single step. In the planning literature, a method that seeks to only optimize its benefit over the next step is called *greedy*.

Restricted to one-step plans, the task of our planner will be to choose from the set of admissable actions, one that will optimize an objective function. However, certain characteristics of our problem disqualify the use of analytical optimization techniques such as calculus of variations and projected gradient descent. First, the constraints and the objective function cannot be expressed in closed-form equations. Second, the objective function is not guaranteed to have a single extremum. This means that we will have to formulate the optimization as a search problem, and will either have to search this space exhaustively or add a stochastic element to the gradient descent to keep the search from getting stuck in local extrema. This sort of method is commonly used in a technique called *simulated annealing* [Bohachevsky86].

2.2.3 Control

Excavation is an example of a class of tasks for which a robot cannot proceed purely on a geometric level. Interaction with the world must be taken into account explicitly in planning as well as control. For example, our excavation planner will reason about the forces to keep the robot away from clearly infeasible plans, but will still need a layer of control that can react in real-time to the interaction forces to ensure that the robot doesn't get into trouble. To this end, we note that neither position nor force control will suffice and that it is not possible to prescribe both force and position simultaneously. Instead, there are methods in which the stiffness or the impedance of the robot is prescribed [Hogan85, Hogan87]. This thesis shows how these ideas can be used in the excavation scenario. While no experimental results are presented, the ideas necessary for an automated excavator to alter its path in response to the forces experienced while digging are developed.

3. Also known as *postconditions* in the planning literature and *dynamical*, *plant*, or *forward* models in the control literature. Some researchers have called them *action* models, but this usage is avoided to keep from confusing them with *action* spaces.

2.2.4 Learning

Earlier sections have discussed the need to develop a good model of the resistive force experienced by an excavating robot. Having a robot learn models from experience is an intuitive idea and indeed there is a large body of work (*machine learning*) that addresses many issues in this area. Of particular interest is a type of learning which is often known as function approximation. That is, given sets of inputs and outputs, we would like to predict the output value corresponding to an arbitrary set of inputs. This thesis examines the use of three learning methods that have been developed by other researchers. In particular we compare the predictions obtained by the analytical models developed by agricultural engineers to those that can be obtained through the use of *global regression*, *memory based learning* [Atkeson91, Moore94], and *neural nets* [Mclelland88, Hinton92].

The emphasis in the discussion of robot learning will be to compare the three learning methods. Instead of tuning these methods to perform better, we explore the trade-offs between accuracy, training time, speed of prediction and ability to incorporate new data.

Chapter 3 Action Spaces

Commonly, robot planning has used an abstraction known as *configuration space* (C-space). The main idea is that the configuration of a robot and the objects that it manipulates can be uniquely described by a set of parameters. The initial state and the goal state are represented as points in the space spanned by these parameters. Simplistically speaking, the job of a planner is to find a path in this space from the initial state to the world state. The path in C-space prescribes a sequence of states that must be achieved to accomplish the goal and as a consequence dictates the actions that the robot must execute. There are, however, several cases in which configuration-space methods do not work well. This chapter examines a different approach to posing these problems.

There is a well known duality between events (actions) and states. In a state based representation, the world is viewed as a series of states that are altered by events. Events are modeled solely in terms of their state-changing function. Alternatively, an action based approach represents the world in terms of a set of interrelated events— the state of the world at a particular moment in time is represented in terms of the set of events that have occurred up to that moment. As with most dualities, the choice of one representation over another may not affect any essential capability for expression— one dual representation can be converted into the other. However, the form of a representation may make certain kinds of properties more natural to express and reason about [Lansky86]. State based methods

require a planner to find a mapping relating states, while action based methods require a mapping between actions and states. We will see that in some cases, it is easier to express the mapping between actions and state, rather than a mapping between states.

This thesis proposes a methodology for robot planning using an abstraction known as *action* space.[1] The methodology is described simply as follows. We start by encoding the task (as opposed to the mechanism). That is, a prototypical action is parameterized by a compact set of variables. At every step, the robot selects from the set of feasible actions (a subset of the space spanned by task parameters) available to it, one that optimizes some cost criterion. This chapter starts by taking a closer look at planning using the C-space paradigm and points out some of the cases in which action spaces are more appropriate. The chapter ends with extended discussion about how excavation is posed in an action space.

3.1 Robot Planning Using C-space methods

Consider a simple task— a robot is required to move from one point in Cartesian space to another. For serially chained robots (such as conventional robot manipulators), we are typically interested in placing the end-effector at a specified location. In this case, actuator positions uniquely specify the pose[2] of the end-effector but the converse is not true because more than one set of joint angles might satisfy an end-effector position. It is more natural to plan in a C-space that is spanned by the joint variables of a robot. In this space a “configuration” (a set of joint angles) of the robot is specified by a single point. For a task that requires the end-effector to move between known locations, we determine starting and ending configurations for the robot and the problem is reduced to finding a sequence of movements that will take the joints from between the two configurations. This method is illustrated in Figure 8.

Planning in C-space is a powerful paradigm. If C-space obstacles[3] can be mapped exactly, several algorithms exist to find optimal or near-optimal paths from start to goal. For low dimensional systems it is possible to use dynamic programming by first discretizing each dimension of the configuration-space into a grid [Lengyel90]. Another technique is to con-

1. The term “action space” was suggested to me by Randy Brost who noted that this representation is composed only of action (command) variables as opposed to a representation that uses that uses both action and state variables which has sometimes been referred to as ‘operation space’. Concurrently, other researchers have used the term “action space”, for example [Schneider93].
2. Denotes both position and orientation in cartesian space.
3. In the context of path planning, the Cartesian boundaries of objects that might interfere with the path of a robot are mapped into the C-space. The planner must these avoid these regions (C-space obstacles) to ensure a collision free path. In the general case, C-space obstacles correspond to undesirable configurations of the world and the robot.

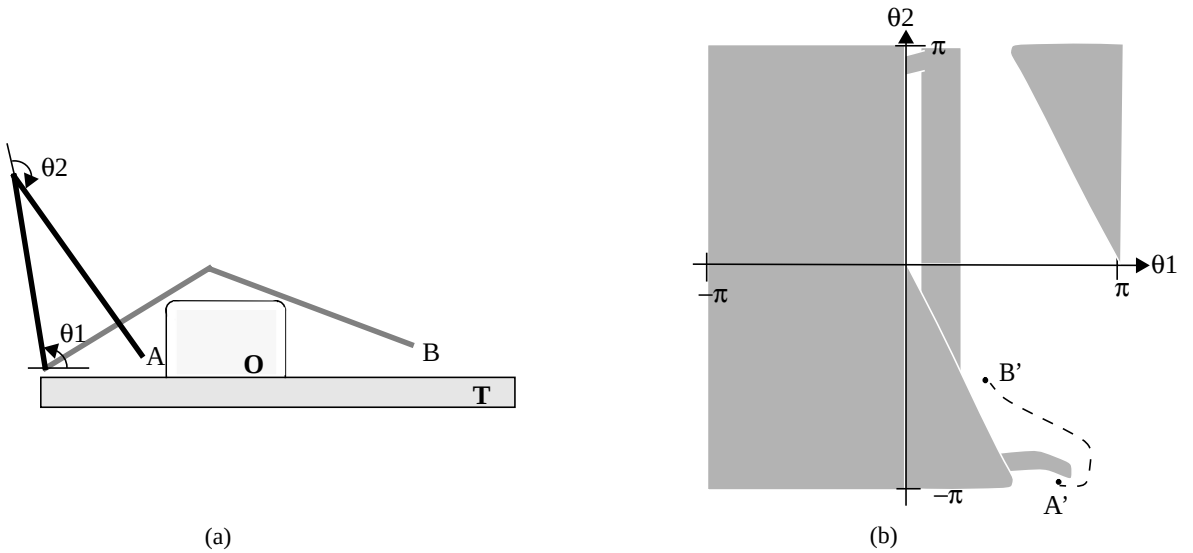


Figure 8 Planning the motion of a two joint manipulator using C-space (from [Brady86]). (a) shows initial and final configurations of a two joint manipulator. (b) shows the C-space spanned by the two joint variables (θ_1, θ_2). Each configuration of manipulator (A & B) maps to a point in C-space (A′ & B′). Objects that the robot must avoid (O & T) are represented by shaded areas (C-space obstacles). The task is posed as finding a path from A′ to B′without entering the inadmissable regions.

struct special potential functions in C-space that have global properties. Following the gradient of the potential function is guaranteed to lead to the goal [Koditschek90, Whitcomb91]. Recently, a number of C-space planners have been developed for practical purposes [Latombe91, Hwang92] and at least for the instances where the C-space is mapped, the mechanism is holonomic, and the number of dimensions is not high (< 30), path planning is considered a solved problem. Another major use of C-space is that it allows analytical identification of sets of configurations that have particular properties. A planner using this methodology will avoid certain regions in the C-space because they represent undesirable configurations. For example, research in grasp planning identifies regions that represents "wedging"— a condition that is important to avoid [Brost88]. Conversely a planner might be interested in a region the robot must enter. Lozano-Perez, Mason and Taylor have proposed a method for automatically generating fine motion strategies in the presence of uncertainty in sensing and control. Given a goal state, the system determines regions from which certain motions are guaranteed to attain the goal successfully [Lozano84].

There are, however, at least three cases in which C-space methods do not work well:

- *If a planning problem has very high dimensionality it is intractable to represent the problem in C-space.* Search in high dimensional spaces is computationally prohibitive. This problem is at its worst when it is necessary to manipulate diffuse media like soil because the number of variables required to uniquely represent the state of the world is very high even with an approximate representation. As an example,

consider uniquely describing the natural state of terrain in small area (3m x 3m). Suppose that we discretize the ground-plane into 10 cm cells and assign an elevation value to each cell. This gives us a configuration space spanned by 900 variables. This is two orders of magnitude greater than the number of variables used by most planners, and is large enough to make a state-based representation computationally intractable.

- *If C-space obstacles are complex, it may not be possible to build them quickly in response to sensor data*. It is often the case that a robot does not have accurate models of the world and may have to discover the world through its sensors. In path planning, C-space obstacles are spatial convolutions of the robot shape with the shapes of the objects in the world. These obstacles are sometimes so difficult to describe analytically that many researchers have adopted grid based methods [Lengyel90] and in some cases approximate objects in the space by geometrically simpler features (C-space obstacles become easier to compute if a robot and/or interfering objects are conservatively estimated by simple geometric shapes). For general planning problems, mapping C-space obstacles can have the complexity of exhaustively treating each cell to determine if it is a part of C-space obstacle. This proves to be a problem in the case where a continuously moving robot must make its decisions based on sensory data in real-time.

- *If it is difficult to compute the actions necessary to get from one state to another, a C-space representation will be ineffective*. Sometimes, it is possible to compute the effect of an action (using a *forward* model) but is difficult to compute the necessary action(s) that must be taken in order to achieve a given state using an *inverse* model (3.2). The difficulty in computing the inverse model is typically due to one of two reasons. An inverse solution may not be unique (in some cases, may not exist) as is the case with underactuated systems (non-holonomic systems can be thought of as underactuated). Alternately, it may not be possible to transform differential equation constraints into C-space from the space in which they are most naturally expressed.

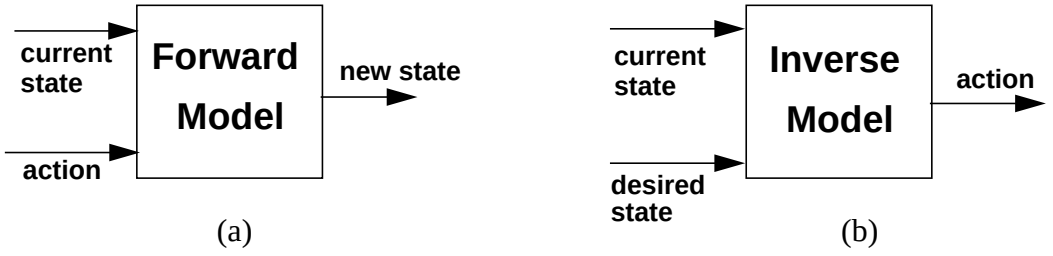


Figure 9 (a) Forward model of a system and (b) Inverse model. C-space methods are intractable when the inverse model is very difficult to compute.

3.2 Action Space methods

Recently, a new set of methods have been proposed. Instead of abstracting robot and world state, these methods abstract the task that the robot performs. These methods iden-

tify the set of feasible actions that the robot can perform and choose one that optimizes a cost criterion. The chosen plan is guaranteed to satisfy constraints (e.g. avoid collision with obstacles) as well as optimize a cost criterion (e.g. minimize joint torques). What makes these methods distinct from C-space methods is that instead of inferring actions from states, they operate directly on the set of robot actions. Simplicity is the main advantage of such an approach. For example, instead of building complicated C-space obstacles and then deducing a path in between the obstacles, now the planner chooses from among a set of actions that best accomplish the robot's goals. Kelly has used this philosophy in a fine resolution navigation scheme for an outdoor mobile robot [Kelly94]. Several times a second, the planner considers approximately ten steering angles to use during the next control cycle. The planner simulates the effect of using these steering angles over a short period of time and evaluates each of the resultant paths. Any steering angles that result in paths that go near or through obstacles (determined from an elevation map) are discarded. The steering angle that results in a path that is optimal based on several criteria is chosen (Figure 10). The choice is necessarily based on predicting the state of the robot a few seconds later. The search space is too large to consider sequences of actions over a longer period.

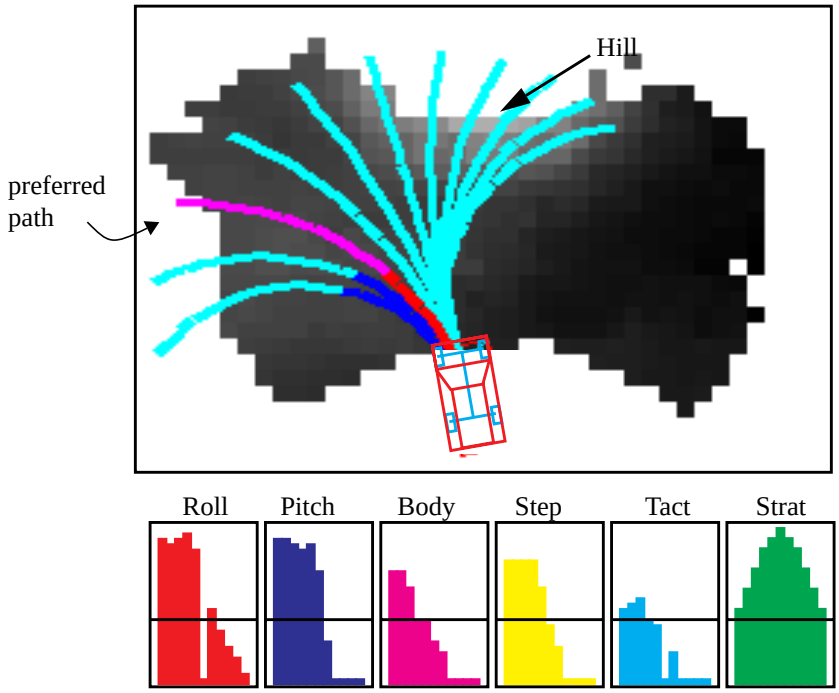


Figure 10 Planning for a mobile robot with non-holonomic constraints and multiple optimization criteria. Kelly's system chooses a steering angle from a handful of candidate trajectories. Here the system issues a left turn command to avoid a hill to its right. The histograms below represent the votes for each candidate trajectory (higher values indicate preferred trajectories). The hazards are excessive roll, excessive pitch, collision with the body, and collision with the wheels. The tactical vote is the overall vote of hazard avoidance. The strategic vote is highest for the action that is most directed towards the goal.

A very similar method is employed by Feiten and Bauer who describe an indoor mobile robot operating in a very cluttered environment [Feiten94]. Instead of looking at a few steering angles, their planner considers a continuous range of steering angles, but the idea is the same— some steering angles are eliminated since their use would cause collisions. Of the remaining, the steering angle which leads the robot most closely to a specified via point is chosen. Both cases accrue the advantages of action space methods— the planning space is reduced to two dimensions, it is not necessary to build complicated C-space obstacles in real-time to incorporate sensor data, and, it is not necessary to invert the non-holonomic model of a mobile robot. It is interesting to note that although the mechanisms are different (Kelly's system is implemented on a conventionally steered vehicle, while Feiten and Bauer use a skid- steered vehicle), both abstract the action as one composed of steering angle and forward velocity. Another point of note is that both methods are local and depend on a higher level planner to specify coarsely distributed via-points.

3.3 Semantics of an Action Space

Let us look more formally at the notion of an action space. An action space is spanned by the range of parameters used to define an action that a robot is capable of executing. Each point in this space represents an atomic action. The space is separated into two sets— one is the set of all feasible actions while the other is the set of actions known to fail; either they are impossible to achieve or they result in an undesirable effect. The task of a planner is to maximize some goodness criteria over the set of feasible actions. This is the form of a familiar problem in optimal control:

$$\textit{Maximize } J(\mathbf{u}) \textit{ subject to } g(\mathbf{u})$$

where u is the space of controls (actions) that might be used by the robot, $J(\mathbf{u})$ is a utility function and $g(\mathbf{u})$ are constraints that delimit the set of feasible actions. Several points need to be made about planning in an action space.

1. A point in action space denotes a unique action that a robot might execute. The semantics of an action (the trajectories executed by the robot's actuators) are dependent on the both the state of the world and the mechanism used to perform the action. In the case of a mobile robot, the trajectory implied by a point in the action space, depends on state of robot just before the action is executed. In comparison, a point is C-space always denotes the same thing.

2. Since the state of the world at a particular moment in time is represented in terms of the set of events that have occurred up to that moment, there is no way of explicitly representing the goal within an action space. An action space planner terminates either when the set of feasible actions is empty, or, if the planner senses (or deduces) that the state of world is sufficiently close to the goal.

3. An action space planner needs two functions. One determines the feasibility of a candidate action and the other estimates its utility. Feasibility is determined by checking to ensure that a point in the action space meets constraints due to the mechanism, interaction between the world and the robot, and those due to the task itself.

Unlike with C-space, there is a lot of latitude in the selection of action space variables. This is because an action can be abstracted at many different levels. The designer of an action-space planner must select the variables much as a software designer selects the level of abstraction of parameters to a function. There are, however, some considerations that can help in the selection of the representation:

1. *Variables should encode the task intuitively.* One of the advantages of using an action space is that it allows direct analysis of the sets of feasible actions; parameterization with variables makes understanding the solutions easier. For example, a skid steered vehicle is controlled by two velocities (one for each track). We could encode the action space with these velocities, but it is more intuitive to represent the action space with path curvature and forward velocity because points in the space can be more intuitively understood.

2. *The representation should be compact.* We would like to use the minimum number of variables necessary to express an action. Generally, there is a trade-off in search complexity and efficiency in the task. That is, we could search through combinations of a large number of variables to find the action that is optimal, but it will generally be better to encode a task with no more than 4 or 5 independent variables. A compact representation will preclude certain actions but it makes the search space tractable.

3. *If some action variables depend on others, they need not be included in the search space.* Some variables necessary to describe an action need not be included in the action space. If a variable can be instantiated based on other variables, then it need not be included in the search space. Additionally, if the evaluation function monotonically increases (or decreases) with a particular variable, there will be no need to include that variable in the action space. In both cases, for a given set of independent parameters, the other parameters can be found readily.

We contend that planning in domains with the features described above is more tractable if the task is represented in an action space. Further, the method allows for generality. An action space allows abstraction from the robot mechanism. The mechanism is represented via the constraints. For two different robots performing the same task, only the function that determines feasibility due to the mechanism need change.

3.4 Action Spaces for Excavation

This section examines the process of posing the task of excavation in an action space. It explores the question as to what makes this domain a good candidate for action space methods, and explains how the representation is chosen.

The main reason that we have chosen to pose the excavation task in an action space, is because of the very high dimensionality of the corresponding C-space and because the mechanics of the task are very complex. In this domain, it is difficult to compute the effect of a robot's action on the world. Given highly uncertain dynamics, it will be of limited utility to consider long sequences of actions. Neither is an analytical method possible that will relate state to a preferred action. Hence not only will we have to search, but the search horizon will necessarily be short.

The action space for excavation has a few interesting properties. First, the goal directly provides some of the constraints, and so even though we cannot encode the goal explicitly, specification of the goal is straightforward. For example, Figure 11 shows a 2D profile of terrain that we would like an excavating robot to produce. For reasons of structural integrity it is important that the robot not overshoot the goal. That is, the robot should not have to put any of the soil back to get closer to the goal state. In this case, the boundaries of the excavated shape provide hard constraints on the trajectories that the excavator must execute.

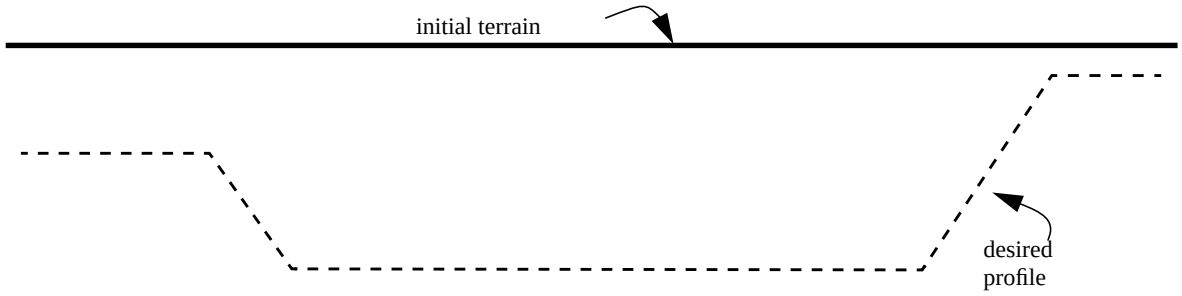


Figure 11 Specification of the goal for excavation. Digging trajectories cannot extrude past the boundaries of the goal.

Second, the mechanics of excavation are complex enough that a planner can't afford to use a full forward model. If we restrict ourselves to evaluating one-step plans, we can use an approximate evaluation function. Although we might not be able to fully predict the effect of an action on the terrain, it will be possible to approximate the amount of soil excavated by a digging action. Further, we might also be able to predict the amount of resistive force experienced by a digging tool. This will be enough to make one-step plans— for a given terrain and

mechanism, we will be able to select from the set of feasible actions, those that optimize the amount of soil excavated, and minimize reactive forces. Third, excavation is essentially a planar problem. Most digging machines operate in a plane, positioning the base and/or a "swing" joint when necessary (Figure 12). We could try to make an excavation planner global by adding variables that would select the position of the base, as well. However, this would unnecessarily increase the search space. Figure 12 shows how the goal of excavating a footing is translated into a sequence of trenches that must be excavated.

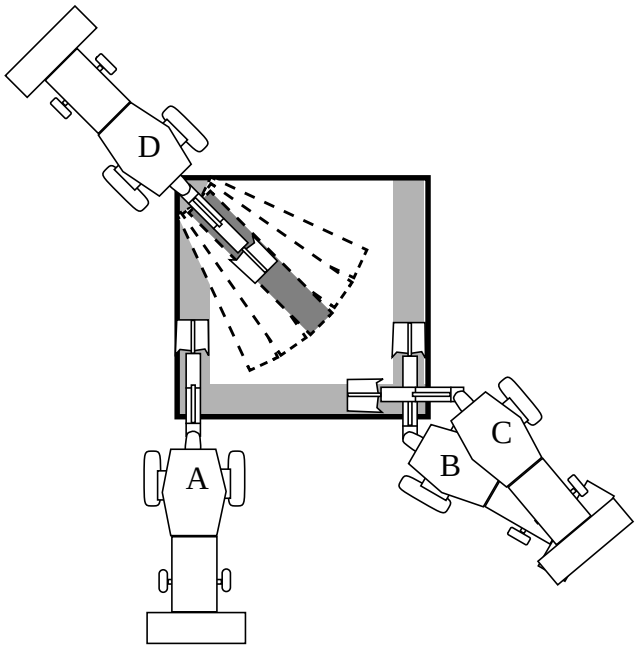


Figure 12 Plan view of an excavator backhoe digging a footing (from [Ober82]). The machine starts at A and digs one side of the footing to a desired depth. Second, the machine moves to B from which a second side can be excavated. A third setup (C) is achieved by rotation of the base. Excavation of the center is achieved by a fan-like pattern of digs from (D).

It is assumed that a higher level geometric planner will plan the "tiling" of the trenches given the task of a large excavation. This thesis will concern itself with the planar problem, in this case, of how to dig a specified trench. Of course, the task of trenching is useful on its own.

We will pose geometric and force constraints in the action space we develop. The geometric constraints will limit the set of actions to those that are within the workspace of the robot, stay within the bounds of the specified shape, do not excavate the more soil than the bucket can hold, and do not require more force than the robot can muster.

3.4.1 Representing the task of trenching

Consider the task of trenching. To develop a range of prototypical digs that are efficient consider the way human operators dig. Figure 13 shows a digging pattern used by a backhoe to dig a precise trench.

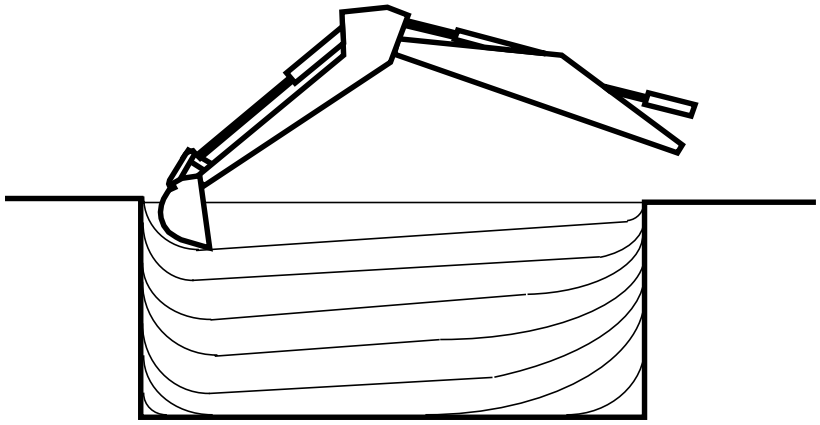


Figure 13 Side view of digging pattern used by backhoe operators to dig a footing.

Since an action space encodes actions, not mechanisms, our task representation will not include any details about the configuration of the machine being used to perform the task. A very differently configured robot that is able to execute the same kind of trajectories with its end-effector would be treated identically. In this light, consider a compact representation of the trajectories that are required to perform trenching. Figure 14 shows only the excavating tool (also known as a *bucket*) moving through soil in a typical digging operation. We refer to this entire operation as a single *action*.

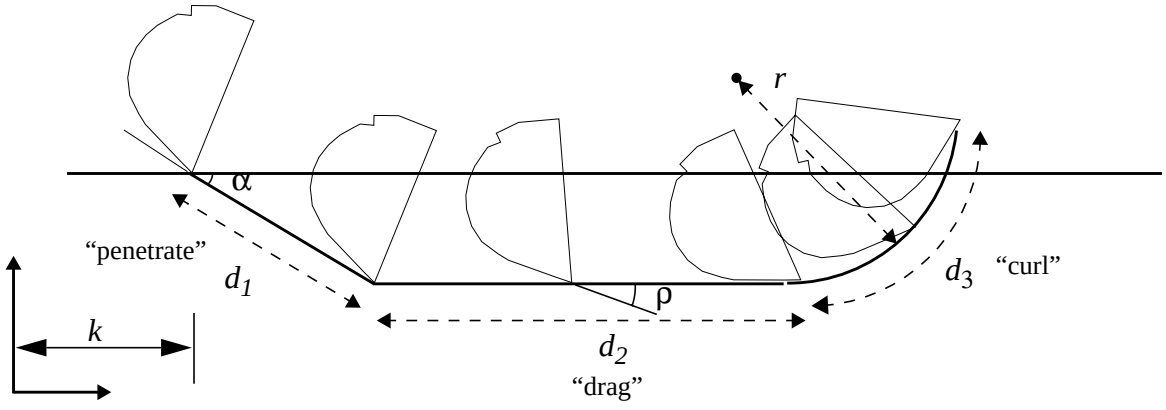


Figure 14 A typical dig during a trenching operation. There are typically three distinct types of motion— *penetrate, drag and curl*. A digging trajectory of this form can be represented by a seven-tuple (*k*, α, d_1, d_2, d_3, *r*, ρ).

This representation requires six variables: k is the distance from a reference frame to the point that bucket enters in the soil, α is the angle at which the bucket enters the soil. It travels along this angle for a distance d_1, and then follows a horizontal path for a distance d_2. Finally the bucket rotates through the soil along an arc of length d_3 and radius r. Also needed is the value of $\rho()$ (the pitch angle of the bucket, relative to a fixed coordinate frame; this is also known as the *rake* angle) throughout the dig.

The following sections show how the seven-tuple can be reduced to a three-tuple of independent variables. First, note that it is reasonable to assume that the objective function varies monotonically with the variable d_2 in the "drag" phase[4]. This means that if the other variables are instantiated, it is always possible to take the maximum acceptable value of d_2. We will find the values of $\rho()$, r, d_3 from an analysis of the task and through experimentation. Our action space, then, will be spanned by the three-tuple (k, α, d_1).

3.4.2 Task Analysis

This section we shows how $\rho()$, r, and d_3 are computed. Since ρ varies along the entire dig and is dependent on the other variables, it is necessary to find $\rho(\alpha, d_1, d_2, d_3)$. It turns out that the analysis can be separated into three cases in which the bucket undergoes pure translation ("penetrate"), simultaneous rotation and translation ("drag"), and, pure rotation ("curl"). First, we review some relevant theory.

3.4.2.1 Contact Analysis

In general we would like to develop a method to rotate and translate the bucket as it moves through the soil, such that it is most effective. This consideration is necessary because some trajectories will cause compression of soil below the bucket rather than shearing of soil in front of the bucket. It is possible to use a geometric analysis developed by Reuleaux [Reuleaux63] to design good trajectories for an excavator bucket.

Figure 15 (a) shows a planar object with two point contacts. We would like to know how to move this body such that it moves away from the contacts. Since all planar displacements can be expressed as a rotation about some (possibly infinite) point, it is possible to determine the set of all rotation centers for which the body will move away from the contacts.

4. For an evaluation function that is based on the volume excavated and the resistive forces experienced, this means that with an increase in d_2, both the volume of soil excavated, and the resistive forces increase. While there are a few conditions in which this might not hold true, we will make this assumption.

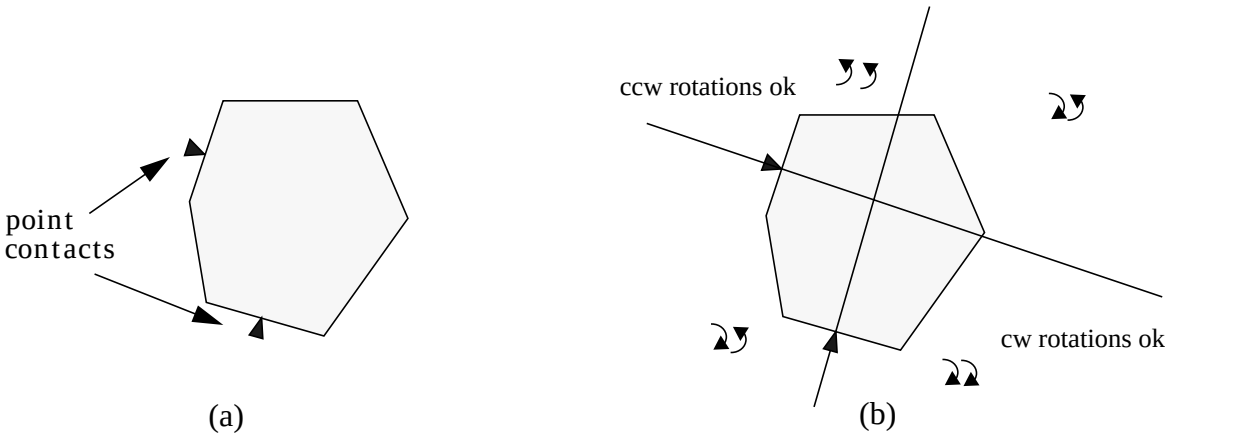


Figure 15 Determining rotation centers for a planar body moving away from point contacts. Each contact separates the plane into half planes— in one half plane a clockwise rotation moves the body away from the contact, in the other, a counter-clockwise rotation moves the body away from contact.

To do this, draw a line normal to the contact surface for each contact. On one side of this line are all the rotation centers for which the body will escape the contact under clockwise rotation. and on the other side of the line are the rotation centers for which the body will escape contact under counter-clockwise rotations. Repeating this procedure for the second contact, we get four quadrants (Figure 15 (b)). In two of these quadrants, those in which the arrows point in the same direction, the appropriate rotation will result in the body moving away from *both* contacts.

A similar analysis can be performed for an excavator bucket. Let's presume that the bucket is deep in the soil. To rotate the bucket out of the soil such that the bottom of the bucket moves away from the soil, the center of rotation must lie in a region which satisfies multiple contact constraints on the back and the bottom. As in the example above, it is possible to draw normals to the contact surface (Figure 16). Since we are only concerned with counter-clockwise direction, the set of feasible centers of rotation is easily found (shaded area in Figure 16).

Next, consider the issue of determining the instantaneous center of rotation for a moving object. Assume that the instantaneous direction of motion of two points on an rigid object are known, for example the points A& B in Figure 17. It is possible to readily compute the center of rotation by drawing perpendiculars to the directions of motion. The center of rotation is found at the intersection of the two perpendiculars.

The above is sufficient to analyze the three modes (penetrate, drag and curl) of movement through the soil.

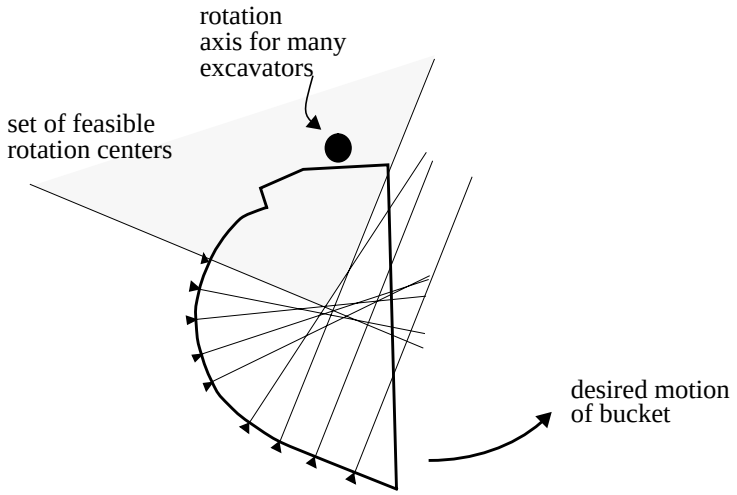


Figure 16 Determining allowable centers of rotation for an excavator bucket.

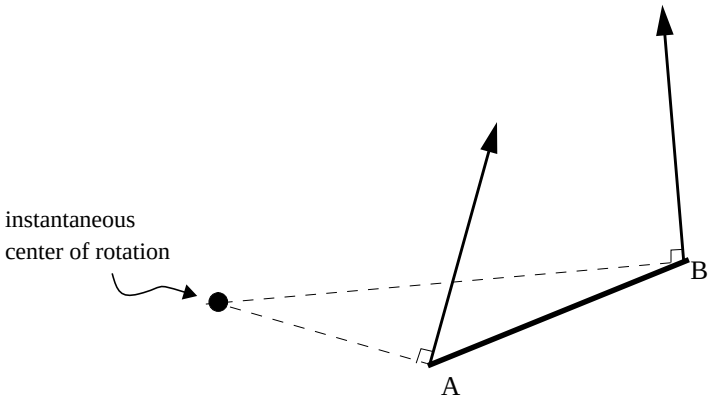


Figure 17 Computing the instantaneous center of rotation for a moving object (AB). The arrows indicate the direction in which these points move instantaneously.

3.4.2.2 Pure translation (“penetrate”)

In the first phase of digging, the bucket moves without rotating. In this case, it is necessary to determine the rake angle ρ for a given the angle of approach α. If a local coordinate frame is defined at the tip of the bucket such that the *x* axis is aligned with the face of the bucket. The rake angle (ρ) is defined as the angle between the *x* axis and a global reference frame. This is the angle that is commanded to a robot when moving the robot from point to point.

One simple heuristic is to ensure $\alpha \leq \rho$. If this condition doesn’t hold, the bottom of the bucket presses into the soil and the reactive force build up below the bucket (Figure 19 (b)). In fact we find that in practice, $\rho > \alpha + \varepsilon$, where ε is a small angle (10 degrees for our excavator). This offset is necessary due to uncertainty in end effector control and, because we

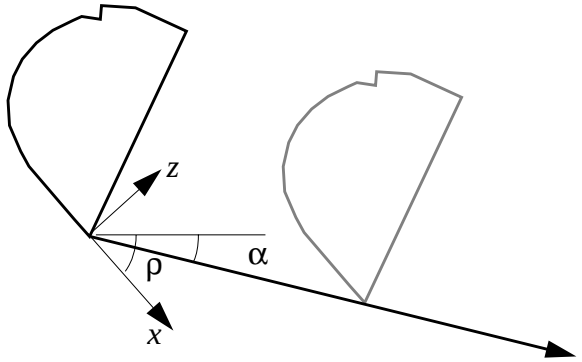


Figure 18 Rake angle vs. angle of movement. The bucket moves along angle α. ρ is the pitch angle of the bucket (also known as the rake angle).

model the bucket as a simple shape. In effect, this offset keeps the back of the bucket from the rubbing against the soil.

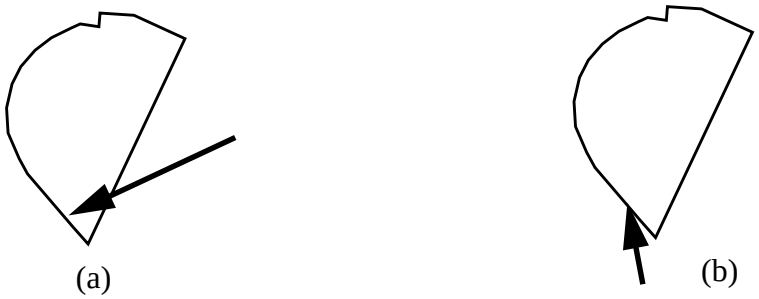


Figure 19 For the bucket to be effective, the reactive force should result from contact with soil inside the bucket (a), not behind it (b). The arrows show the direction of the forces acting on the bucket.

This phenomena has been verified experimentally. Figure 20 shows reactive force in the *z* direction (local coordinate frame at the bucket tip as in Figure 18) from three experiments in which α and ρ were varied. For ε less than 10 degrees, the reactive force in the *z* direction is positive indicating that the contact with the terrain is at the back of the bucket.

Apart from the phenomena of contact between the back of the bucket and the soil, there is an optimal rake angle for the bucket following a trajectory that minimizes the contact force. This minimum is a function of the soil properties and the orientation of the cutting surface and is not simple to compute analytically. This relationship is revisited in Chapter 5. In practice ε is kept small (or large) for reasons of reachability. For our testbed, we would like to keep ε small and ε = 10 degrees.

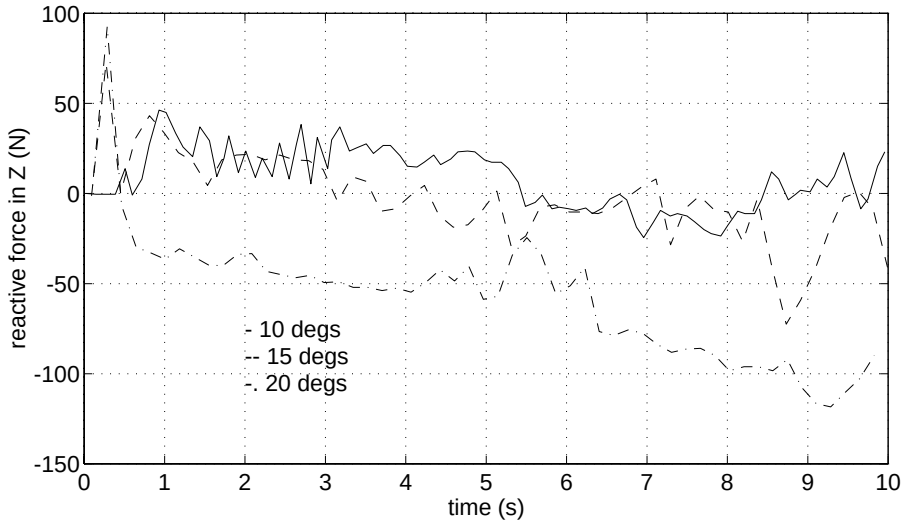


Figure 20 Reactive force in *z* direction for three rake angles. Three rake angles (ρ = 10, 15, 20 degs) were tried for the same trajectory (α = 10 degs). Positive values of the reactive force indicate contact between the back of the bucket and the soil.

3.4.2.3 Rotation and translation (“drag”)

During the “drag” phase, the bucket must follow a straight line while it rotates. The rotation is essential because at the end of the drag phase the bucket should be in the best possible position to be “curled”, i.e. the pitch angle should be minimized. However, it is not obvious how fast the bucket can be rotated per unit distance along d_2 since it is necessary to keep the bottom and back of the bucket from pressing against the soil as it rotates. Once again this is a matter of ensuring the rotation centers stay within an admissable region.

Figure 21 shows a bucket that is both rotating and translating through soil. AB is a segment on the bottom of the bucket. Given the directions that points A and B are moving along, it is possible to compute the instantaneous center of rotation. Since point B is moving horizontally, all rotation centers must lie on a vertical line going through point B. Point A is significant in that the surface normal generated due to contact at this point will be one of the limits to the set of rotation centers. To ensure a good trajectory for the bucket, the instantaneous rotation center must not get closer to the tip of the bucket than the closest admissable rotation center. Recall from the analysis in Figure 16 that the closest admissable rotation center comes from assuming a point contact at A.

It is difficult to analytically determine the maximum rate of rotation for translation. It is simpler to examine the effect of several rates of rotation. Figure 22 shows the distance from the bucket tip to the instantaneous rotation centers for various rates of rotation (degree change in pitch per unit distance along *t*). It also shows the smallest admissable radius of

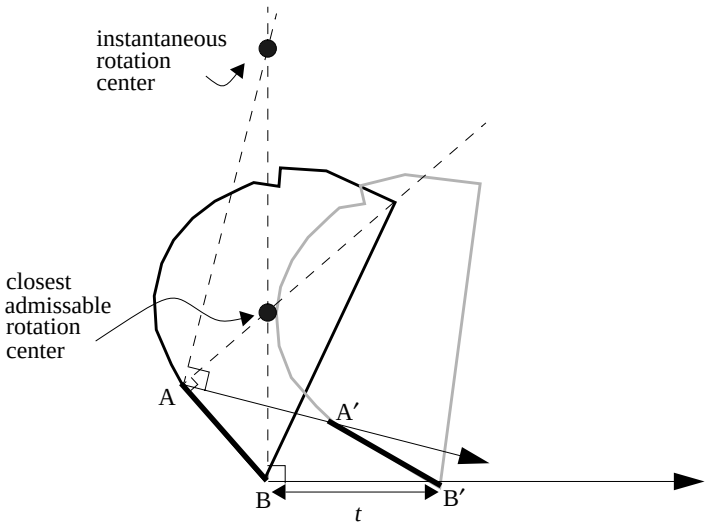


Figure 21 Rotation centers for a bucket that is rotating and translating. The shape of the bucket and its pitch angle dictate the closest admissable rotation center. To ensure a good path, point A should not move in a direction that will cause the instantaneous rotation center to fall below the closest admissable rotation center.

rotation. For small pitch angles, this distance can get to be bigger than the radius of instantaneous rotation especially for high rotation rates. Two solutions are possible. First, the rotation is done up to the angle at which the closest admissable rotation center is equal to the instantaneous rotation center, at which point the bucket continues to translate only. The second solution is to switch to a lower rate of rotation.

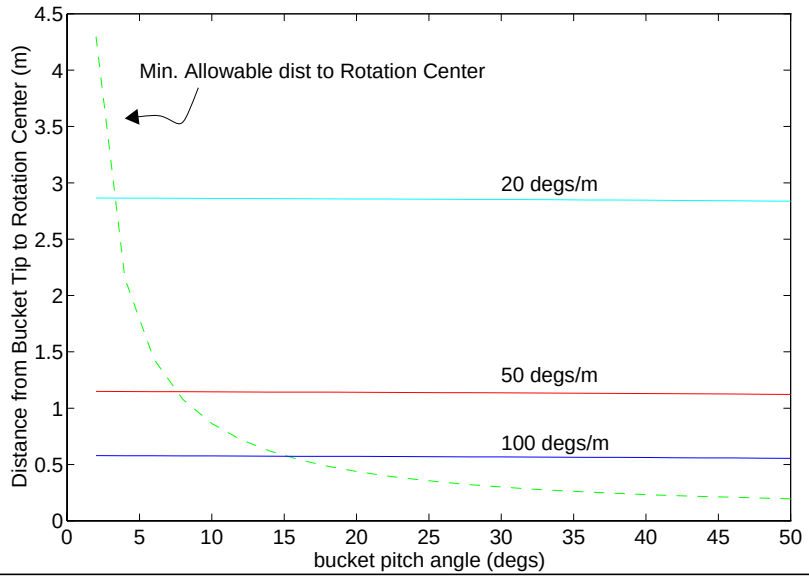


Figure 22 Rotation centers as a function of rates of rotation for the bucket used in our testbed. The solid lines show the distance of the rotation center from the tip of bucket. The minimum allowable distance to rotation center (dashed line) is a function of the bucket shape.

Hence the rake angle at any given distance along d_2 is given by:

$$\rho = max(\zeta(\omega), \rho_i - d_2\omega) \quad (1)$$

where $\zeta(\omega)$ is the minimum rake angle acceptable for a given rate of rate of rotation, ω, and ρ_i is the pitch angle at the start of the "drag" phase. We use ω = 50 degrees/m in our testbed. In this case, $\zeta(50)$ is approximately 8 degrees. For smaller pitch angles, the bucket starts to push down on the soil.

3.4.2.4 Pure rotation ("curling")

At the end of the drag phase, the bucket should move such that the maximum amount of soil is collected in the bucket. In essence, it is desirable to "curl" the bucket by rotating the tip with the shortest radius of curvature feasible. Given this criteria, the following develops a method to produce values for d_3 and r.

The rotation axis for most excavators happens to lie in this region so for those machines, pure rotations are guaranteed to be good. However, if the bucket cannot physically rotated in this manner (as is the case with our excavator in which the wrist-pitch joint is far from the bucket), then the effective center of rotation must lie in this region. That is the planar joints of the robot (boom, stick and bucket) should be coordinated such that the effective center of rotation lies in the admissable region.

Given a pose of the bucket from which the curl must be accomplished, the algorithm is to start with the smallest admissable radius of rotation. Next the pose of the bucket after rotation to a fixed pitch angle (corresponding to the ending configuration of the bucket shown in Figure 23) is computed. If the pose is reachable, the rotation center is chosen. However, the ending pose might not be achievable if the trajectory of the curling motion violates joint limits. In this case the radius of rotation is increased and the process repeated.

Given the shape and pose of the bucket, this analysis provides the *minimum* radius of rotation, r, to be used to curl the bucket. Given the starting and ending pose of the bucket, and the radius of curvature, it is straightforward to compute the distance d_3.

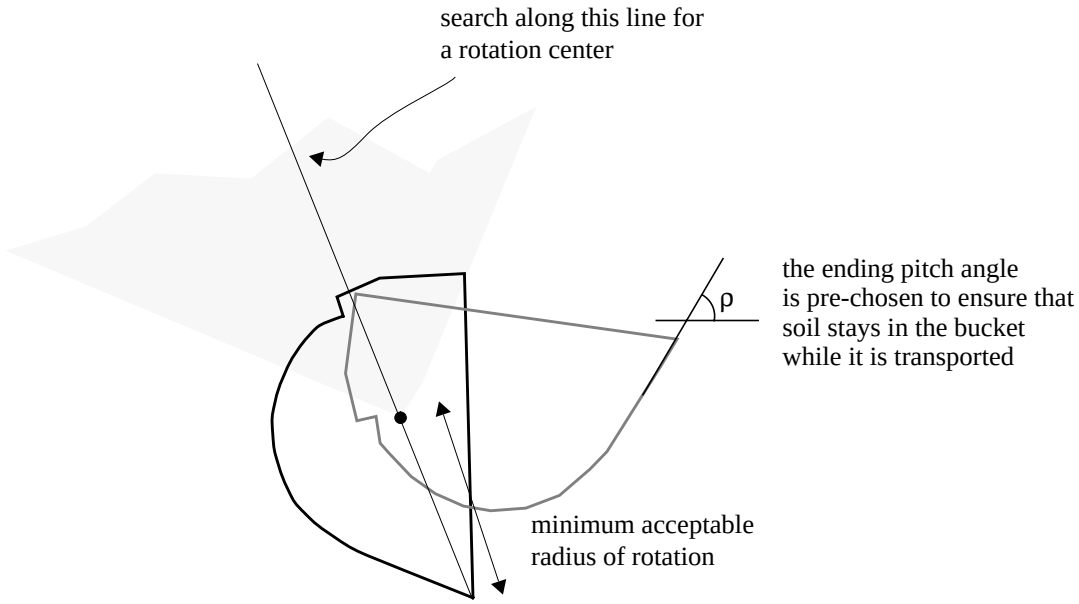


Figure 23 Search for an effective rotation center. When the bucket cannot be rotated directly by actuating a joint at the bucket, the "effective" rotation center is caused by the motion of other limbs. A proposed rotation center is tested. The ending pose produced at the end of the curl is checked using inverse kinematics.

3.5 Encoding another excavation task

Consider another common excavating task, called loading. A variety of machines are used to scoop up material that has been piled. Figure 24 shows a mechanism called a “wheeled front loader” operating on a pile of soil.

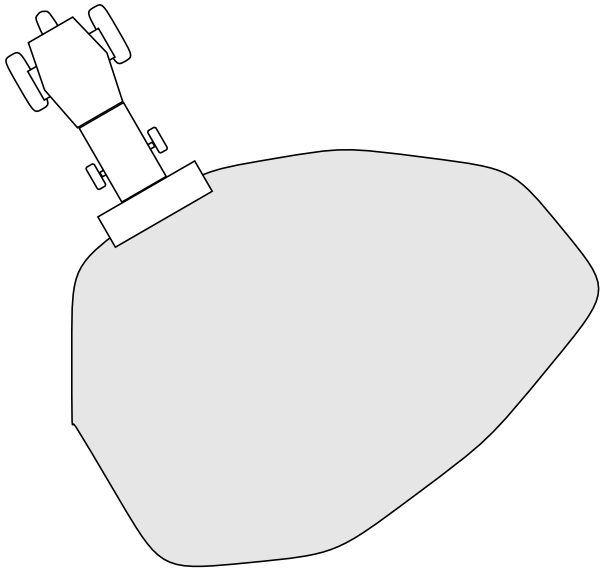


Figure 24 Plan view of a front-loader loading from a pile of soil.

Once again from observation of the way human operators load from piles (Figure 25) it is possible to represent the trajectory of the bucket with the variables (k, α, d_1, d_2, r, ρ) (Figure 26).

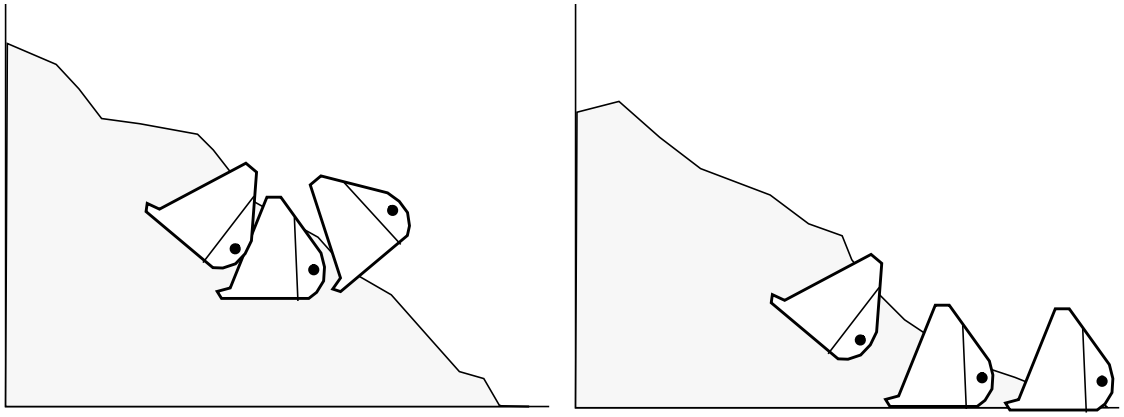


Figure 25 Side view of loading techniques. (a) loading from a high pile (b) from a low pile.

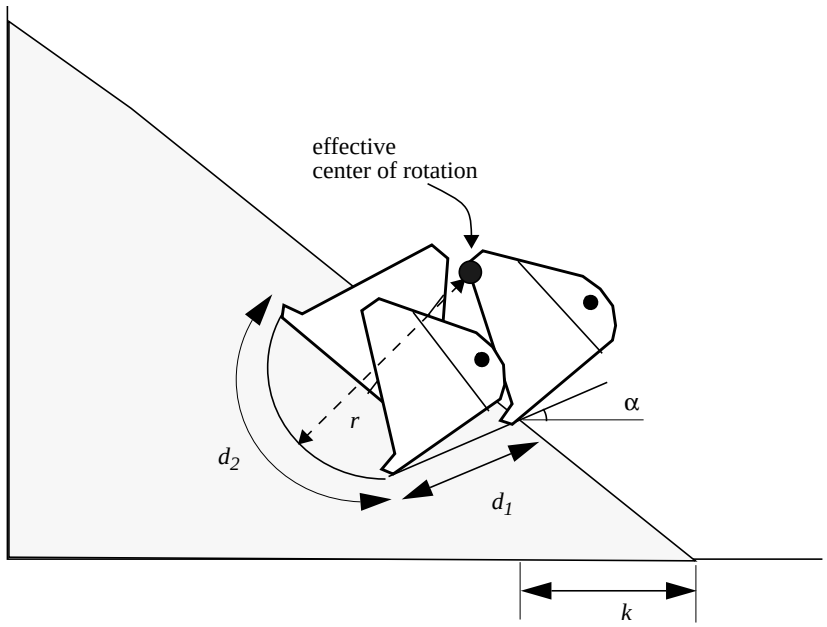


Figure 26 Parameterization of a loading operation. *k* is the horizontal distance from the foot of the pile that the bucket enters the soil. α is the approach angle. After moving along the angle α for a distance d_1, the bucket follows an arc of distance d_2 with a radius of curvature *r*.

As in the case of trenching, the representation can be reduced to a smaller set of independent parameters such as the pair of independent parameters (k, α). In this case d_1 is maximized and the variables r and d_2 are found as in Sec. 3.4.2.4. A three-tuple (k, α, r) could also be used in which case, r is a searched for and d_1 is maximized. The difference in the trajectories produced by these representations is that in the first case, the bucket will curl about the tightest radius with the long straight segments of d_1. In the second case, the curl will vary, with likely shorter segments of d_2. While it is obvious that the latter case produces more realistic trajectories, the choice between the two representations will be based on whether the increase in the search space is justified by increase in the ability of the machine to perform the task.

3.6 Action Spaces— Summary

Several researchers in robotics and artificial intelligence have found that the commonly used method of planning in a state (configuration) space is intractable for certain problems. This may be because the space has very high dimensionality, the “obstacles” in the configuration space are too difficult to compute, or, because a mapping between desired states and actions is not straightforward. Instead of using an inverse model that relates a desired state to an action to be executed by a robot, this thesis has developed a methodology that selects

between the feasible actions that a robot might execute, in effect, circumventing some problems faced by configuration space planners.

This chapter has proposed a methodology and some guidelines to develop a compact and meaningful parameterization of prototypical actions. In specific, a compact representation is proposed for two excavation tasks— trenching and loading. The representation is based partly on observation of patterns used by human operators and partly on an analysis of the physics of the task. There is generally a trade-off between search complexity and efficiency of performing the task. We may further reduce the search space by using nominal values for some of the action variables, at the cost of poorer performance.

One weakness of action space methods is that in practice they are short-sighted— a high branching factor and uncertain dynamics will typically keep us from looking ahead by enough steps to get to the goal. While in principle this is not an intrinsic property of the action space approach, and has to do more with the nature of the task itself, in general, it is harder to develop algorithms with global properties using action space methods than with C-space methods. In practice, action space methods must be used in conjunction with coarse-grained planners that specify intermediate subgoals.

Chapter 4 Geometric Constraints

The previous chapter introduced the concept of an action space and showed how several robotic tasks can be formulated in this framework. This chapter presents geometric constraints that can be posed on action spaces that encode excavation tasks. The set of plans that satisfy these constraints are guaranteed to be feasible if the task is purely geometric. This can be safely assumed only if the interaction between robot and environment is insignificant, that is, if the robot can always generate the forces necessary to perform a task. Specifically, we seek explicit geometric criteria that separate an action space into sets of feasible and infeasible plans. Recall that each point in an action space represents an unique action and that some regions in the action space represent plans that are geometrically infeasible. A plan may be *geometrically* infeasible because it requires a robot to exceed its range of motions or because it violates some geometric criterion associated with successful accomplishment of the task.

Geometric constraints are examined in the context of two tasks— *trenching* and *loading*. Further, the differences in the feasible action space for loading given two different mechanisms are examined.

4.1 Loading

Consider the task of loading from a pile of soil using a front-end loader shown in Figure 27. The loader is to completely excavate the pile, without intruding below the surface of the ground. The previous chapter proposed a compact set of parameters (α, h, d) to represent this task. This section describes how geometric constraints are specified in the space spanned by these parameters. Since the action space approach encodes the task and not the mechanism used to perform the task, it is relatively simple to generalize the method to other mechanisms. To make this point clear, this section compares the geometric constraints for the same task performed by a different mechanism (front shovel) as well.

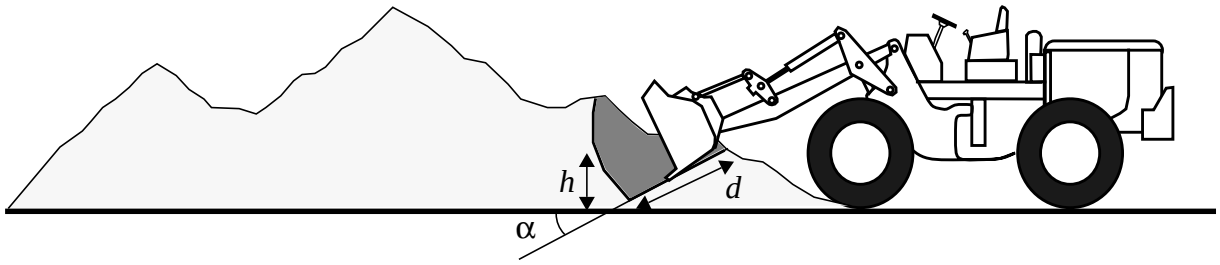


Figure 27 A typical loading task. A prototypical action is parameterized by three variables. α (angle of approach of bucket) and *h* (height at which the bucket enters the pile) are independent variables. *d* is a dependent variable— for every (α, *h*), the largest permissible value of *d* is chosen.

Three constraints are considered— *shaping*, *volume and reachability*. The shaping constraint ensures that a digging action does not violate the shape of the desired terrain. The volume constraint ensures that the excavator does not attempt to scoop a larger volume of soil than can be held by the excavating tool and the reachability constraint reduces search to actions that are within reach of the robot. Note that the shaping constraint is completely independent of the mechanism used to perform the task. The volume constraint depends on the capacity of the excavator bucket, while the reachability constraint is very dependent on the configuration of the robot used.

4.1.1 The shaping constraint

Successful completion of the loading task requires at least two things. First, all the soil from the pile must be removed, and second, the resulting terrain should be flat and level. In this case, the only constraint imposed by the task is that of the final shape of the terrain. To ensure this, a simple constraint is to stipulate that no trajectory of the cutting tool descends below the ground level.

Figure 28 shows the shaping constraint in the (α, h, d) space. Since d is a dependent variable, the constraint surface is given by the maximal permissible value of d for every value of α, h. That is, the set of plans that meet the shaping constraint, lies below this surface.

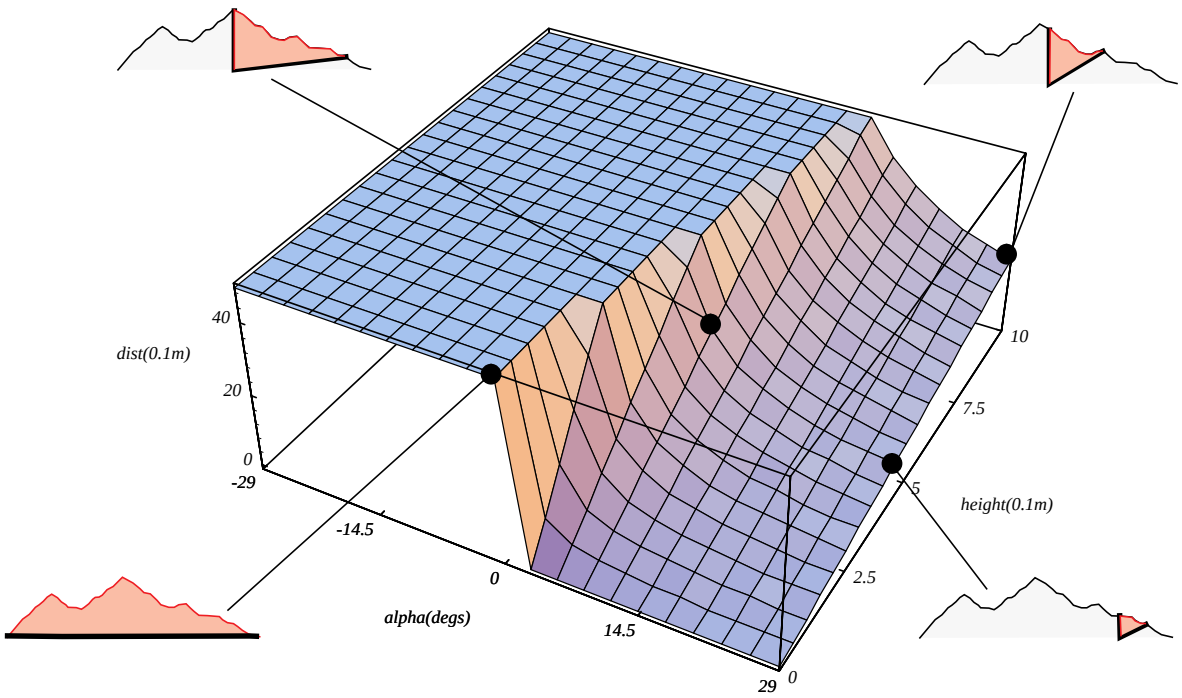


Figure 28 The shaping constraint for the loading task given the terrain and mechanism from Figure 27.

4.1.2 The volume constraint

Of the geometrically feasible actions that the robot loader might perform, some will yield a partially filled bucket, some will result in a full bucket and yet others will sweep through the terrain to excavate more soil that can possibly be held in the bucket. If it were possible to accurately predict the yield (amount of soil excavated) for a given action, it would be straightforward to stipulate a constraint that would exclude all actions that result in a yield of greater volume than the bucket capacity. Unfortunately, accurate calculation of the yield requires a computationally prohibitive calculation with the complexity of a finite element model. At the cost of decreased accuracy, a simpler calculation of swept volume can be used. That is, the yield can be estimated by the volume of soil swept by the bucket.

It is worth noting that the calculation of swept volume provides an upper bound on the amount of soil that will end up in an excavator bucket. For cohesive soils, the swept volume is approximately equal to the amount of soil excavated, but for looser soils, soil settles as it is pushed and it is sometimes necessary for a bucket to sweep a much larger volume than the

bucket capacity to get a full bucket. This constraint is somewhat arbitrary because it is necessary to adjust the limit of the swept volume for a particular soil type. However, without considerations of force, some limit on the volume to be excavated is essential— without it, actions that are clearly inefficient will be considered feasible. Later, in Chapter 5, we will see how additional consideration of the forces involved in excavation makes it possible to get rid of the volume constraint altogether.

Swept volume is computed by integration of a three dimensional shape marked by the trajectory of the bucket. A numerical method based on gaussian quadrature is used to compute the swept volume [Press88].

Figure 29 shows the volume constraint for the loading task given a nominal maximal volume. It shows that for shallow angles α, large values of d_1 are permissible while for steep

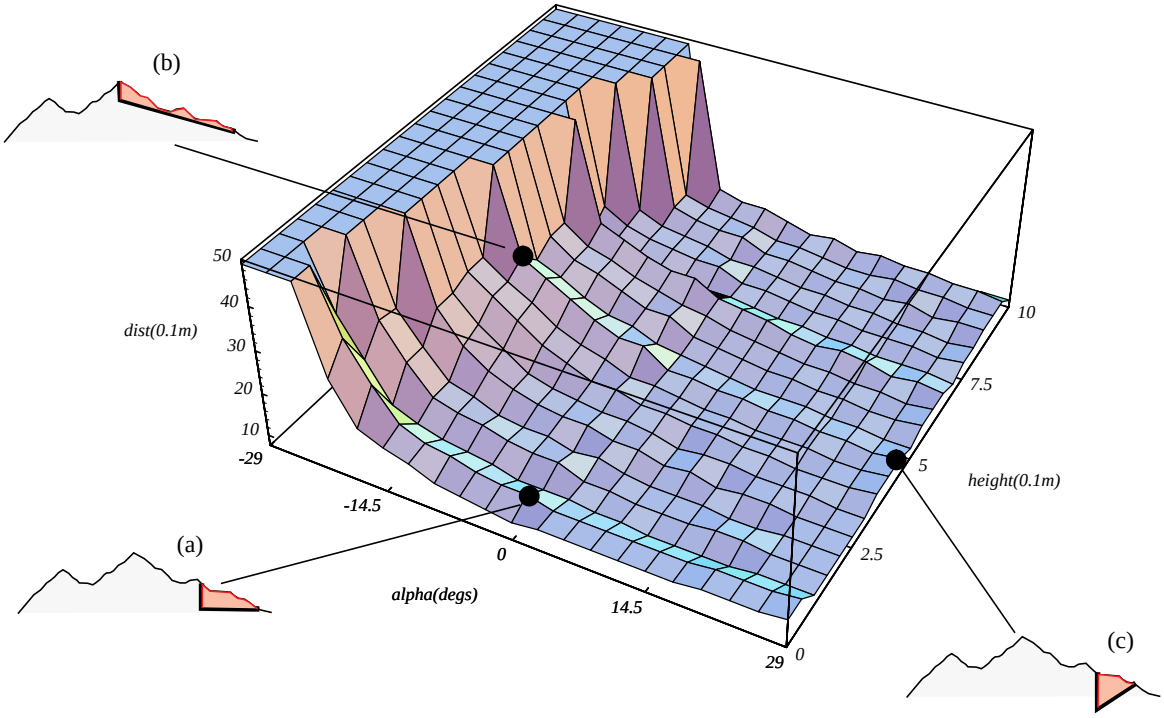


Figure 29 The volume constraint for the loading task given the terrain and mechanism from Figure 27. (a), (b) & (c) shows digs that are limited by volume. Note that (c) cuts below the ground surface and would be restricted by the shaping constraint.

angle of α, shorter distances of d_1 are allowed. In all cases the swept volume is no larger than the capacity of the excavator bucket.

4.1.3 The reachability constraint

In addition to the shaping and volume constraints, the action space for loading can be further limited by considerations of reachability. That is, the range of actions are limited to those that do not require a robot to go outside its workspace. In addition, actions that cause collisions between the terrain and parts of the excavator other than the cutting tool, are excluded. For example, given the scenario of Figure 27, the lateral motion of the loader is limited to the point where the wheels touch the base of the pile.

The reachability constraint requires a kinematic model of the mechanism. Since for this example, the loader of Figure 27 lives in a two dimensional world, it can be modeled as a manipulator with one prismatic and two rotational joints as shown in Figure 30. This model is used to develop forward and inverse kinematic relationships. Forward kinematic equations provide the pose of the tip of the bucket given joint angles, while inverse kinematic equations provide the joint angles for a given pose of the bucket (Appendix 1).

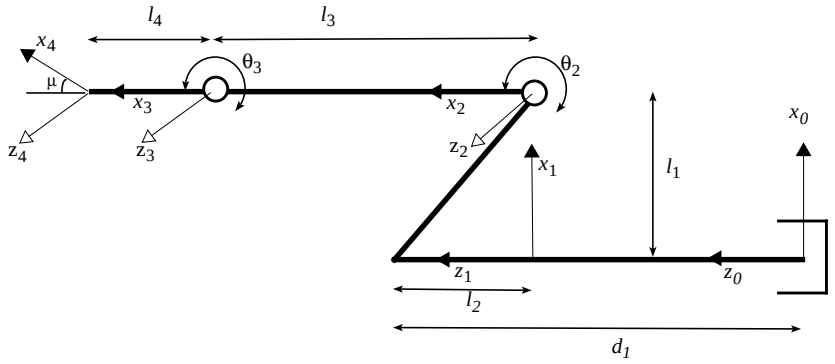


Figure 30 Kinematic model of excavator in Figure 27. d_1 is the lateral distance from the foot of the pile to the local coordinate frame on the vehicle. Travel along this axis is modeled as movement of a prismatic joint. θ_2 and θ_3 are the shoulder and elbow joint angles.

Given a Cartesian space trajectory of the bucket, the inverse kinematics method is used to compute the corresponding joint displacements (d_1, θ_2, θ_3). A candidate dig may fail this constraint if it requires the excavator to reach outside its workspace (exceed joint limits) or if in the course of the dig, one or more of the links are required to interpenetrate the terrain. A complete algorithm to detect the latter condition, must check the position of the various limbs with respect to the terrain, along the entire trajectory. Figure 31 shows the reachability constraint for the loader.

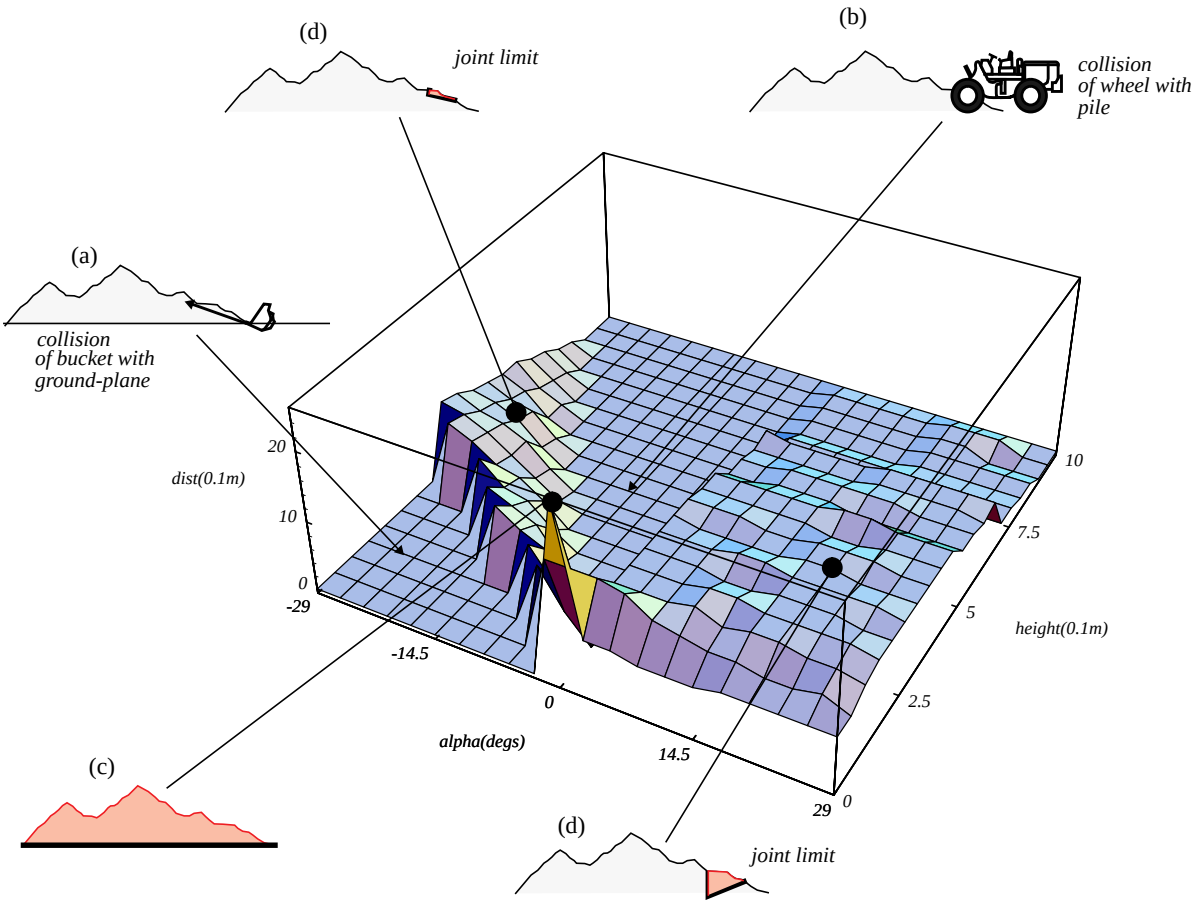


Figure 31 The reachability constraint for the loading task given the loader in Figure 27. This constraint is comprised of three conditions. First, plans that cause collision between the bucket and the terrain are disallowed (a). Second, plans that cause the wheels to go beyond the foot of the pile are disallowed (b). An exception is the case where the foot of the pile is actively excavated (c). Third, a dig might be limited because of joint limits (d).

4.2 Loading with a Front Shovel

Consider the same task of loading from a pile, using a different mechanism (a front shovel) as in Figure 27. Principally, the difference in operation between a loader and a front shovel is that the loader moves its base during excavation, while a front shovel only moves its revolute joints (the base is positioned between digs). The front shovel is modeled as a manipulator with three revolute joints and once again a standard inverse kinematic method (Appendix 1) is used to determine if a given trajectory corresponding to a (α, h, d) triplet is reachable by the mechanism.

The differences in the mechanism will affect the constraint surfaces in two way. First, shovels typically have a smaller capacity than loaders, and thus the volume constraint is

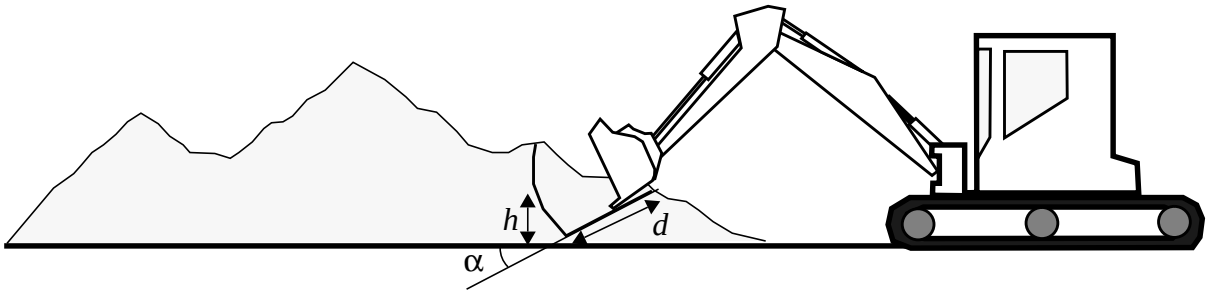


Figure 32 The loading task using a front shovel. The parameterization of the action remains unchanged.

modified.[1] In effect the volume constraint still has the same general shape, but the smaller capacity means that the extension along *d* will be reduced. Second, because of its configuration, the shovel gives rise to a significantly different reachability constraint. Figure 33 shows the reachability constraint for the front shovel.

4.2.1 Composite Geometric constraints

Figure 34 compares the composite geometric constraints for a loader and a front shovel. This figure shows the complete set of feasible actions available to each robot. Figure 35 shows a comparison of the utility of the actions in the (α, *h*) space. It is evident from these figures that a larger set of feasible actions is possible for the front-shovel than for the loader. This is chiefly due to the larger workspace of the mechanism as can be seen by comparing the reachability constraints for the two devices. Note that the comparison is totally on a geometrical basis. In fact, a loader is able to develop much larger forces (by approximately an order of magnitude) than a front shovel because of its configuration. A front shovel is like a robot manipulator— a large amount of torque must be used to move the limbs themselves.

1. It is not possible to realistically simulate the effect of a larger capacity in a two dimensional work because typically, a loader has much wider bucket, that is, it is larger in a dimension that doesn't exist in a 2D world. For sake of comparison, the length of the shovel bucket (from joint to tip) has been modeled as 75% of the length of the loader bucket, and the volume of the shovel bucket as 56% of the loader bucket.

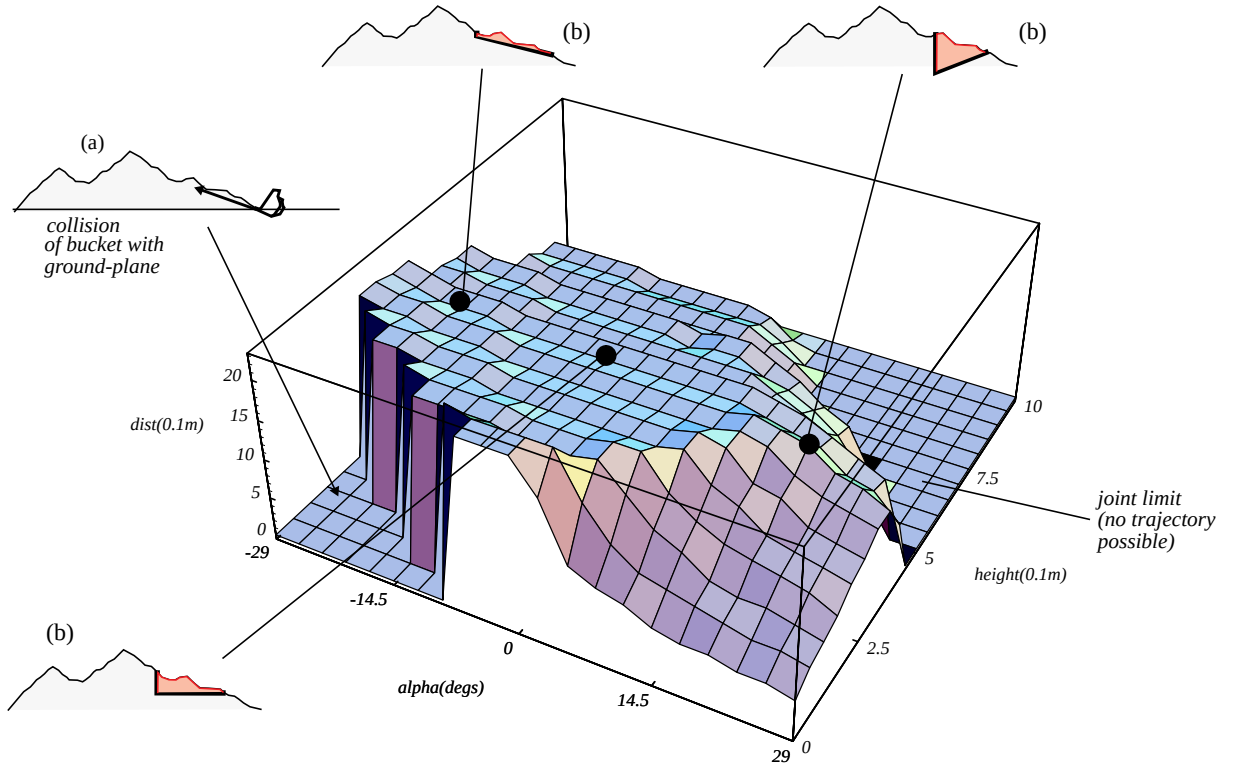


Figure 33 The reachability constraint for a front shovel given the mechanism and terrain in Figure 27. This constraint is composed of two conditions. First, plans that cause collision between the bucket and the terrain are disallowed (a). Second, a dig might be limited because of joint limits (b).

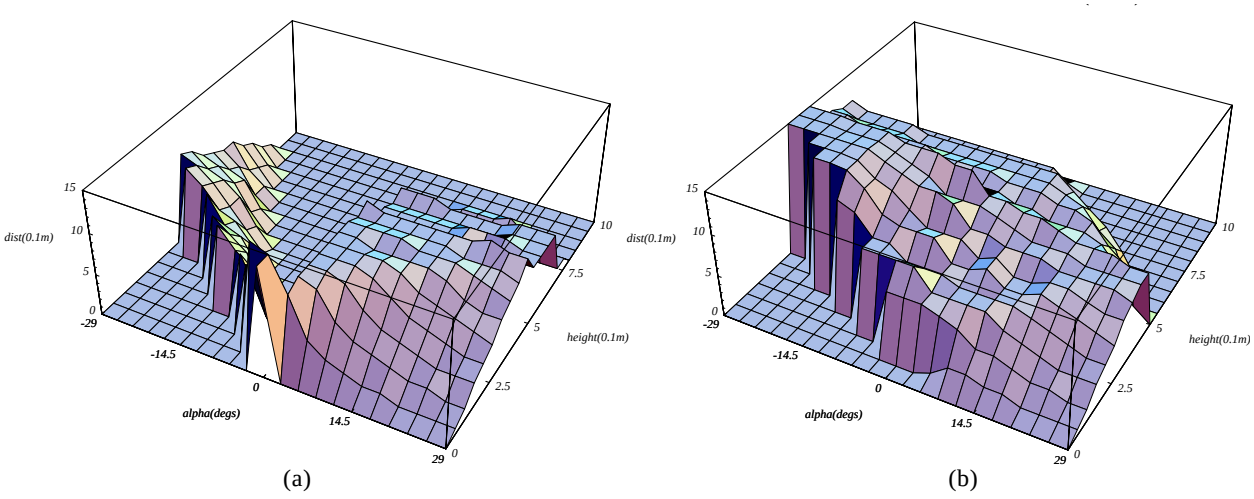


Figure 34 Composite geometric constraints (a) front loader (b) front shovel for the terrain in Figure 27.

This analysis does however confirm the intuition that a shovel is useful for its flexibility, that is its ability to reach a large part of the workspace. A loader on the other hand has a narrow range of capabilities available to it, but is able to excavate larger amounts of soil.

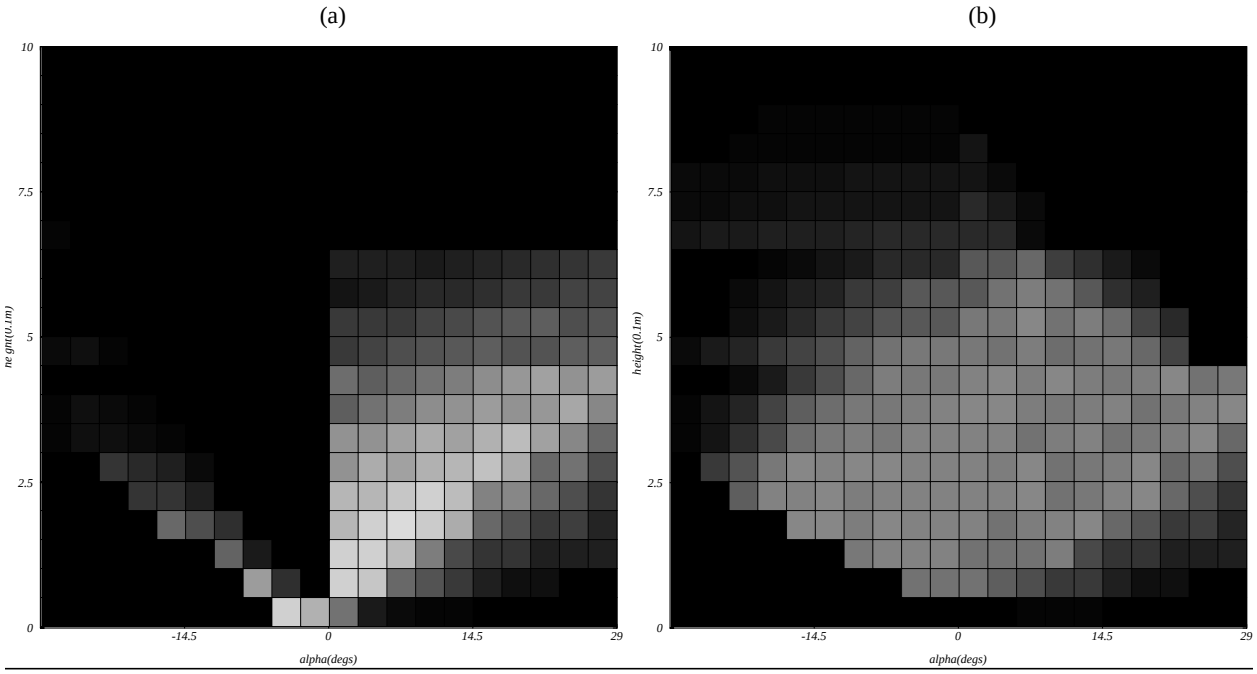


Figure 35 Comparison of the utility (amount of soil excavated) of the actions for two mechanisms (a) loader (b) shovel. Dark areas represent lower utility, and progressively brighter areas represent higher utility.

4.3 Trenching

Consider the trenching operation shown in Figure 36. This task is different from loading in two main ways. First, the prototypical trajectories of the cutting tool in trenching are different from those required for loading. Second, while loaders typically scoop from piles of loose soil, trenching machines are designed to operate in the presence of significantly greater resistive forces encountered in harder and more compact soil. This section will address the former issue. The latter issue calls for consideration of resistive forces in the planning phase and is discussed in Chapter 5.

Apart from the difference in the parameterization of prototypical actions, the approach to posing geometric constraints for trenching will be the same as for loading. Once again, the constraints to be considered are shaping, volume and reachability. The set of actions that meet the various geometric constraints for the manipulator and two different terrains (Figure 37 and Figure 44) are shown below.

A few notes on the graphical representation of the action spaces for trenching are necessary:

- An intuitive set of task parameters for trenching is given by the quadruplet (α, k, d_1, d_2). Since these parameters span a four-dimensional space, it is not possible to

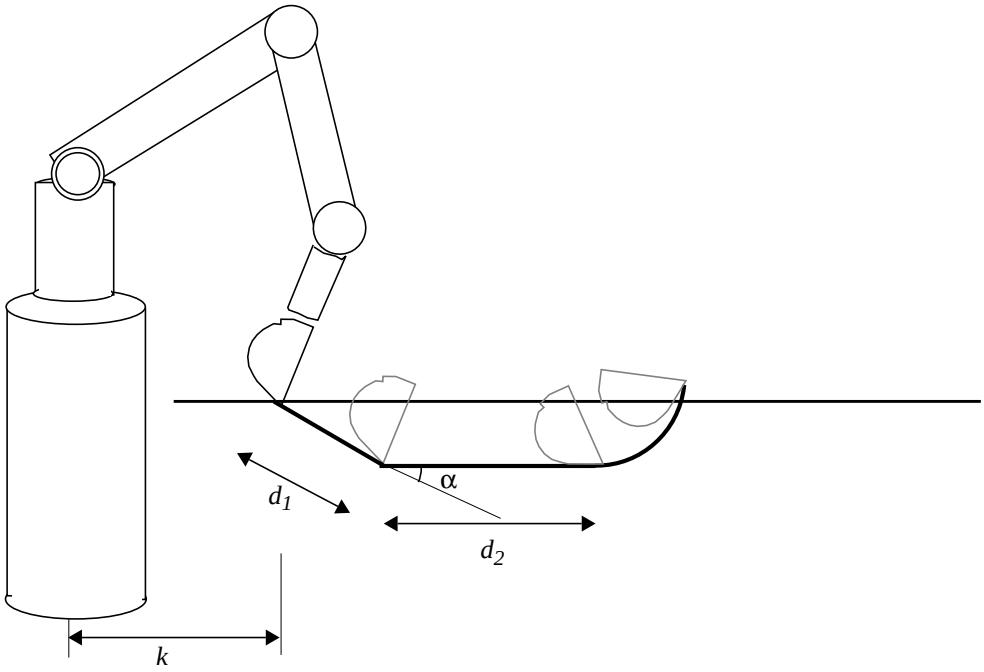


Figure 36 Manipulator arm equipped with a shovel, creating a trench. *k* is an offset from the robot coordinate frame to point where the excavation starts. α is the approach angle of the bucket into the soil, along distance d_1 (penetrate), d_2 is a line parallel to the bottom of the trench. The final stage is a curling operation.

graphically represent sets in this space fully. Recall that d_2 is a dependent parameter, that is, it is to be maximized for every (α, k, d_1). The constraints and the sets of actions they bound are shown in the space of independent parameters.

- The fact that these parameters are independent means that the constraint surface is fully three-dimensional, instead of being a single valued function over a two dimensional set, as was the case with the loading example. In general, the constrained sets in this space need not be solid (interior voids are possible) or connected volumes. The proper interpretation of the constrained set of independent parameters is that they are the set of digs for which there exists a valid (non-zero) value of the dependent parameters.

- Rendering the surface of fully three dimensional objects is a difficult procedure. Since the search space is discretized, a simple solution is to represent each cell that is a feasible action, with a solid cube. The set of feasible plans is displayed as a three-dimensional collection of cubes (see Figure 38 for example).

Consider the example of creating a trench, starting with flat terrain as shown in Figure 37. Below we examine the geometric constraints that can be posed on the space spanned by the independent parameters of prototypical trenching actions.

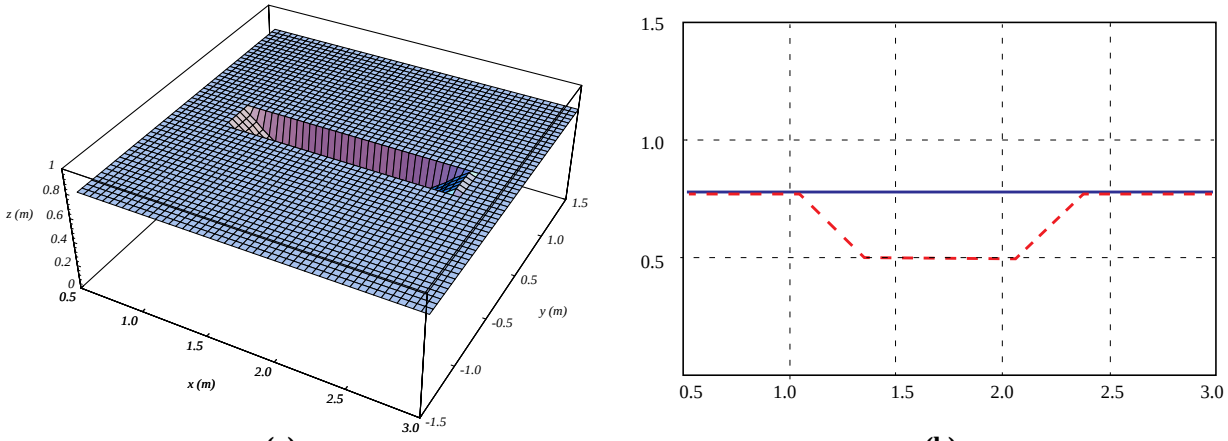


Figure 37 (a) 3D view of the desired shape of the terrain. (b) 2 D section along trench (dashed line) to be excavated. The solid line represents the state of the existing terrain

4.3.1 The shaping constraint

The shaping constraint keeps the trajectory of the bucket from going past the shape of the desired trench (dashed line in Figure 37). The set of actions that meet the shaping constraint is shown in Figure 38. In general, shallower approach angles allow longer penetra-

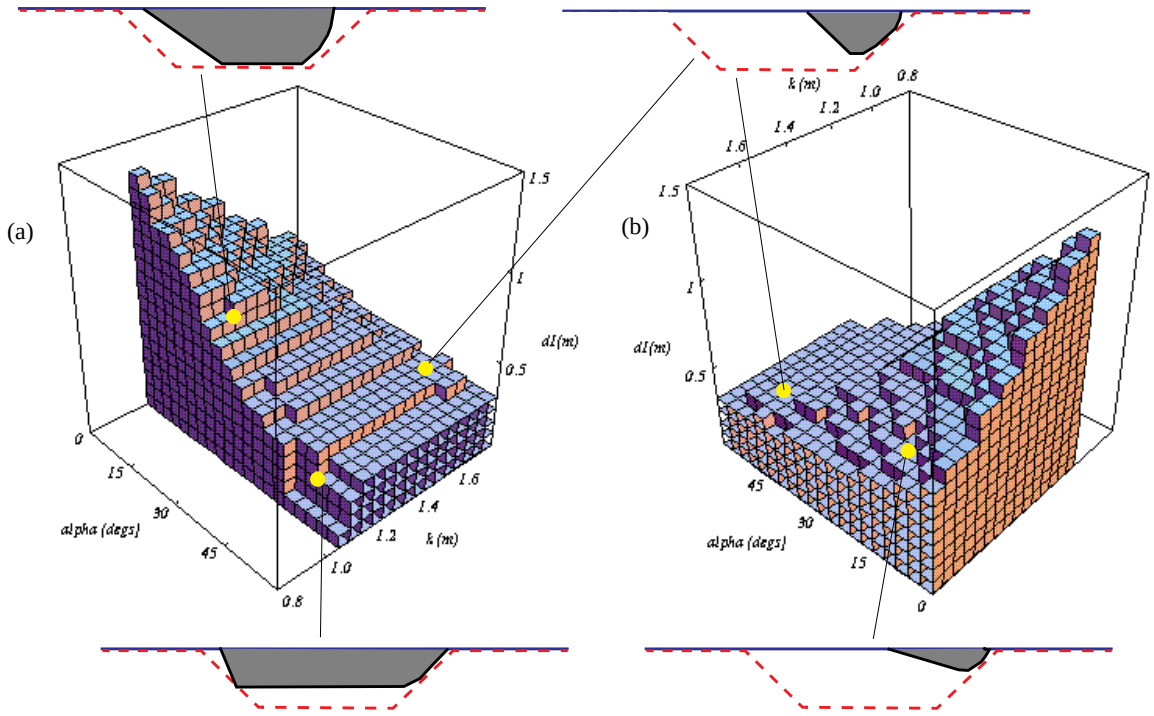


Figure 38 Two views of the set of independent parameters that meet the shaping constraint for the task of trenching, given the scenario in Figure 37. Note that d_1 can be limited by each of the three walls of the trench

tions (higher values of d_1). Note that the shaping constraint is violated for $k < 1.05$ and hence the set is void in this region. Figure 38 also shows the actions associated with four points on the surface of the constrained set.

4.3.2 The volume constraint

The volume constraint for trenching is specified in exactly the same way as it is for loading, except that it is mapped on to the four dimensional space of (α, k, d_1, d_2). Figure 39 shows the set of actions that satisfy the volume constraint for the scenario in Figure 37. The

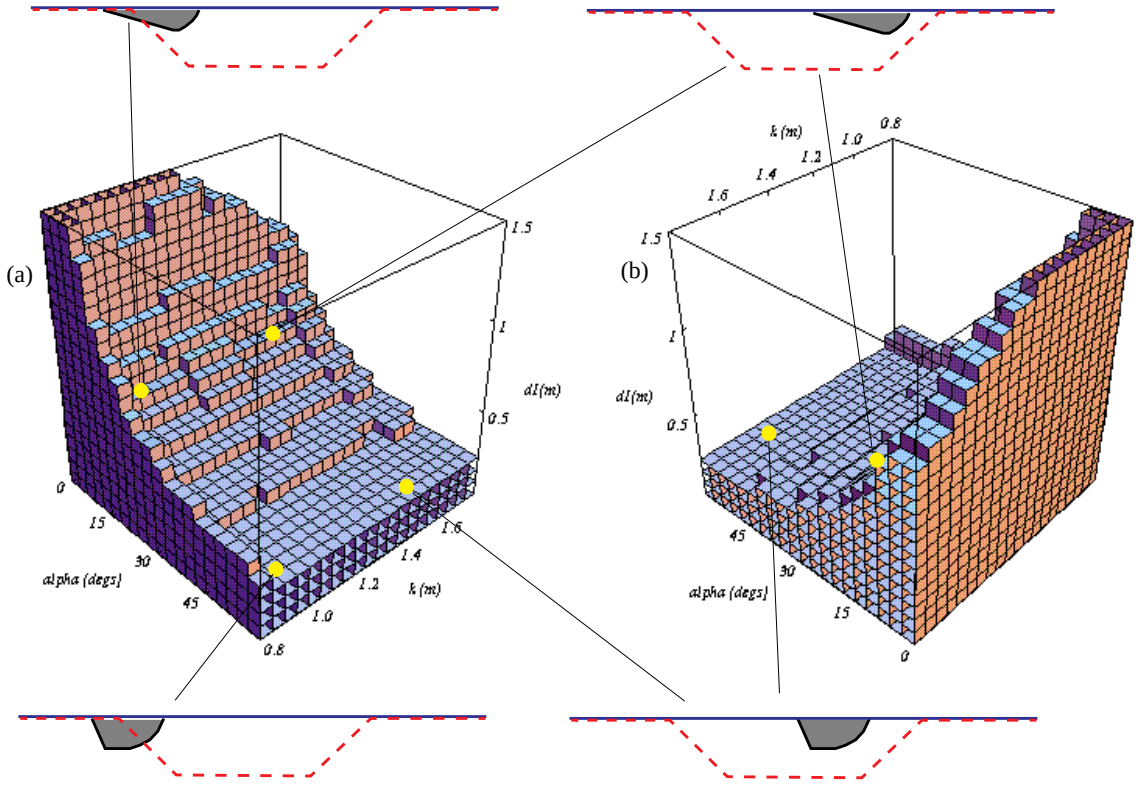


Figure 39 Two views of the set of independent parameters that meet the volume constraint for the task of trenching, given the scenario in Figure 36.

shape of the volume constraint is explained in the following manner. Digging actions with shallow approach angles require a longer penetration phase to reach the volume constraint.

In theory, given flat terrain as shown in Figure 37, the surface due to the volume constraint should be symmetric along k. There are two reasons why this not the case. First, our soil box is finite and so digs with large k and d_1 values represent digging trajectories that surpass the walls of the soil box and hence the feasible set at the top, far corner in Figure 38(a) is curtailed. Second, the curling trajectory is not always identical. This trajectory is

computed based on the shortest radius of rotation possible for the bucket and this radius varies based on the position of the joints before the rotation begins (Chapter 3).

4.3.3 The reachability constraint

Repeating the methodology used in the case of the loader, a kinematic model of the manipulator in Figure 36 has been developed (Figure 30). Although the robot has full six

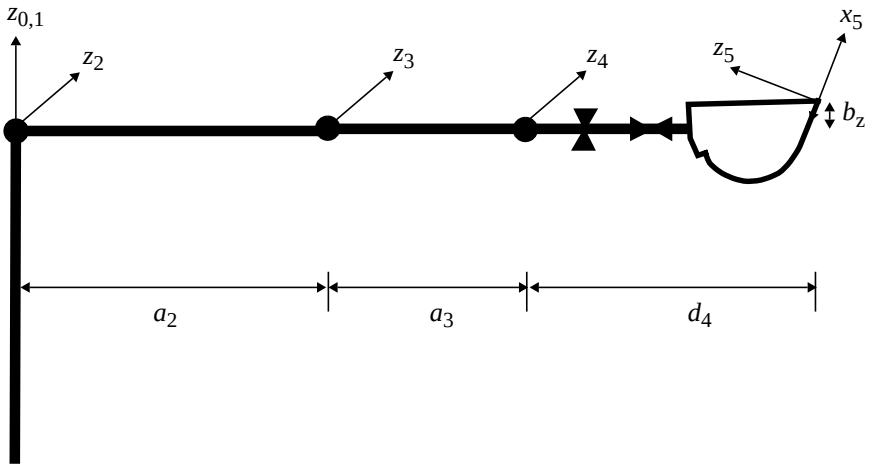


Figure 40 Kinematic model of the manipulator shown in Figure 36. The manipulator has six degree of freedom but only three (shoulder, elbow and wrist pitch) are used during excavation.

degrees of freedom, only three (shoulder, elbow and wrist pitch) are used since trenching is a planar task. This configuration is exactly the same as used by a backhoe (a machine commonly used for trenching), except in the relative length of the limbs. The most notable effect of the particular configuration of this manipulator is that special attention must be paid to "curling" the bucket. For normal backhoe machines, curling is accomplished chiefly by rotating the wrist joint. For our manipulator, since the wrist joint is quite far from the tip of the bucket, it is necessary to find a sequence of motions by three joints so as to accomplish an effective curling operation. Simply rotating the wrist joint produces a very large curling arc. The biggest trouble with these large arcs is that they do not perform the effective scooping that is possible with the tighter arcs and soil often slips out of the bucket. The issue of finding a trajectory of the bucket so as to perform an effective scooping motion was discussed in Section 3.4.2.

Once again, an inverse kinematic method (Appendix 1) is used to determine the whether or not the trajectory implied by a point in action space lies within the workspace of the

manipulator. The set of digging actions that satisfy the reachability constraint given the scenario in Figure 37 is shown in Figure 41.

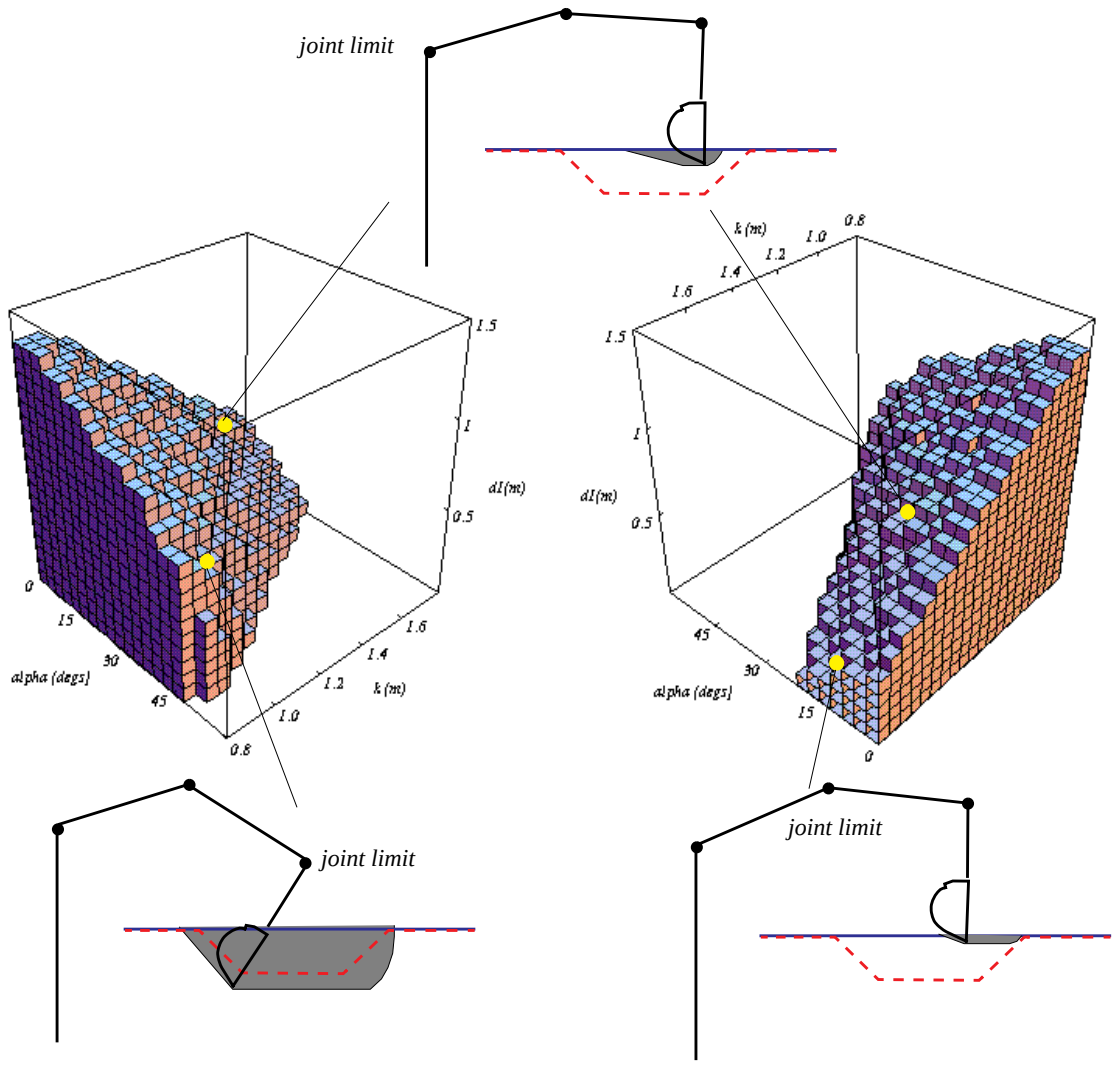


Figure 41 Two views of the set of independent parameters that meet the reachability constraint for the task of trenching, given the scenario in Figure 36.

4.3.4 The composite geometric constraint

Figure 42 shows the composite geometric constraint— the intersection between shaping, volume and reachability constraints. This is the set of digging plans that are geometrically feasible. Also shown are the trajectories corresponding to three points on the surface of the constraint boundary. Each point on the constraint surface corresponds to a trajectory which is on the limit of the robot's workspace.

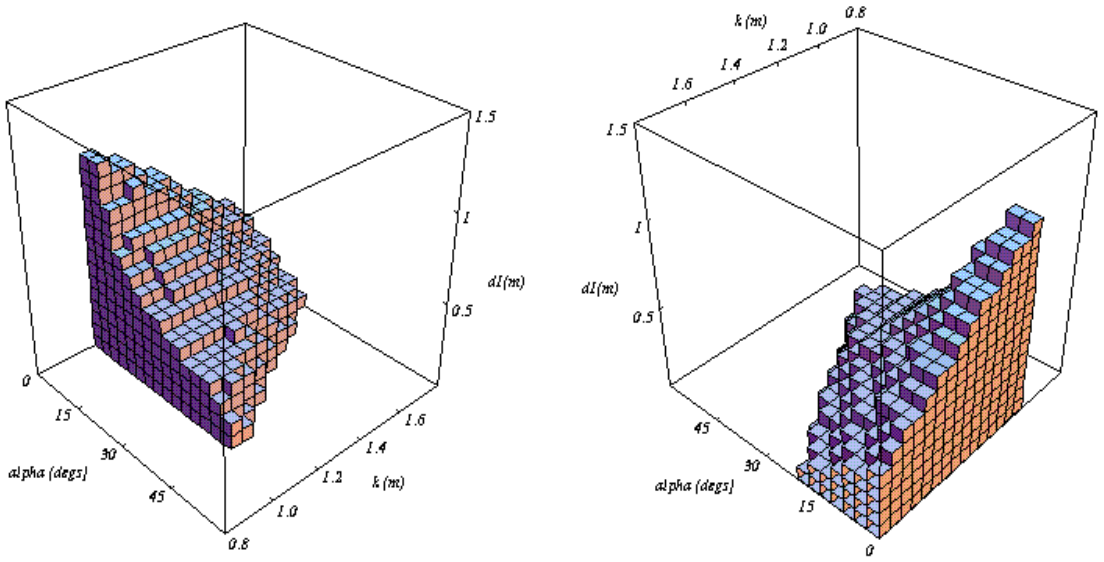


Figure 42 Two views of the set of all feasible plans that meet all geometric constraints given scenario in Figure 37.

As mentioned above, the constrained sets shown above represent the set of actions that are feasible according to various criteria. The utility of these actions varies however. For example, Figure 43 shows the subset of the geometrically feasible actions that sweep a volume of soil that is at least 90% of the capacity of the bucket. Note the difference between

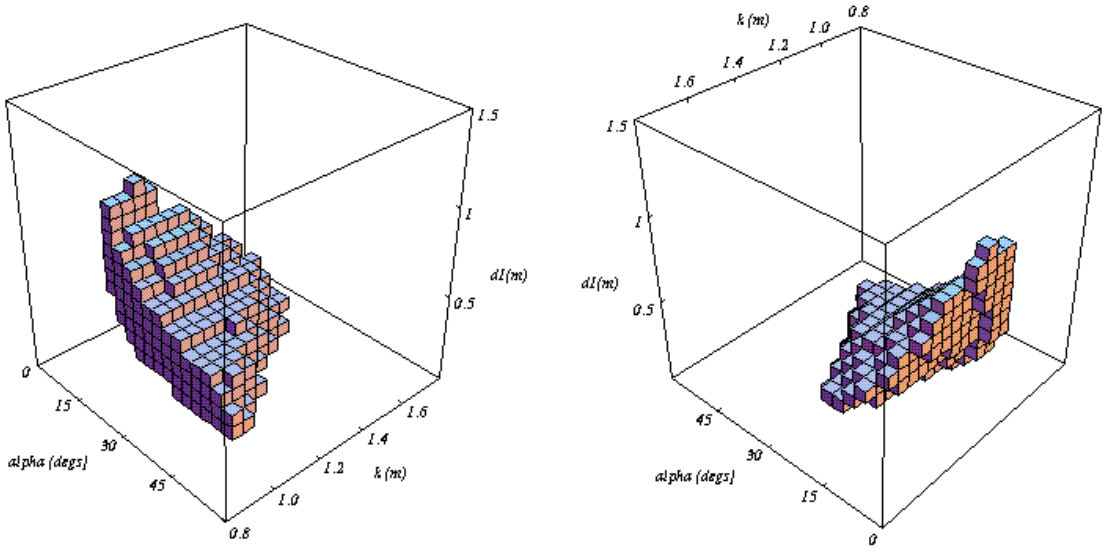


Figure 43 A subset of the geometrically feasible action set— those that excavate a volume of soil that is at minimum 90% of the bucket capacity.

Figure 42 and Figure 43. Digging actions with smaller d_1 values have disappeared in the latter.

4.3.5 Another example.

Recall that the way the action space parameters are defined, the semantics of a point (the trajectory it represents) in the action space change as the terrain changes. This means that the constraint surfaces change as the terrain changes. Consider the terrain shown in Figure 44.

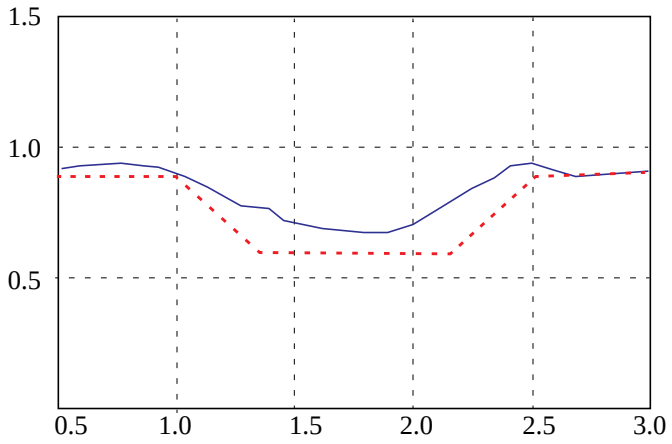


Figure 44 Another terrain.

Figure 45 shows the set of actions that satisfy the composite geometric (shaping, volume and reachability) constraint given the terrain in Figure 44 and the manipulator from Figure 36.

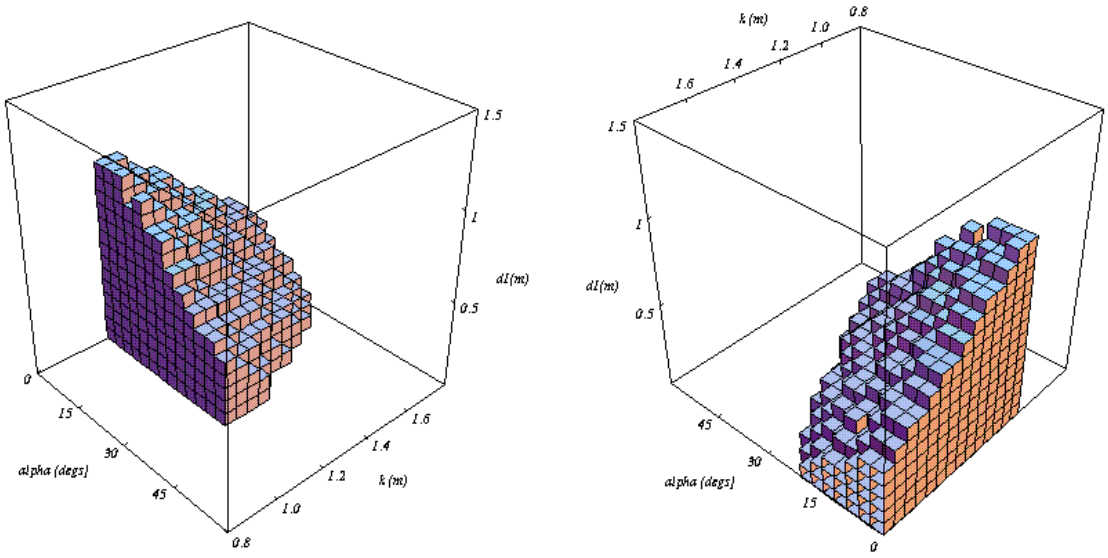


Figure 45 The composite geometric constraint given the terrain in Figure 44 and the manipulator in Figure 36.

4.3.6 Adding a “sensible” constraint.

Apart from the geometric constraints we have posed above, we might constrain the set of feasible actions further using heuristics to eliminate certain undesirable actions. For example, Figure 46 shows an example of a digging action that meets all the geometric constraints stipulated above.

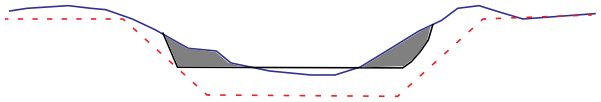


Figure 46 A digging action that meets all geometric constraints but one that should be avoided.

This type of digging action is “undesirable” because soil spills out the bucket as the bucket moves through the air. To effectively scoop soil into the bucket, the force applied by the bucket must be opposed. This is like chasing grains of rice on a plate. One strategy is to push the rice up against the edge of a plate to allow them to be scooped up. Likewise, we would like the excavating tool to be in contact with the terrain while it digs. Adding a simple check to ensure that all digging trajectories travel through the soil, results in the constraint surface shown in Figure 47. Such a heuristic is an easy way of specifying digging strategy

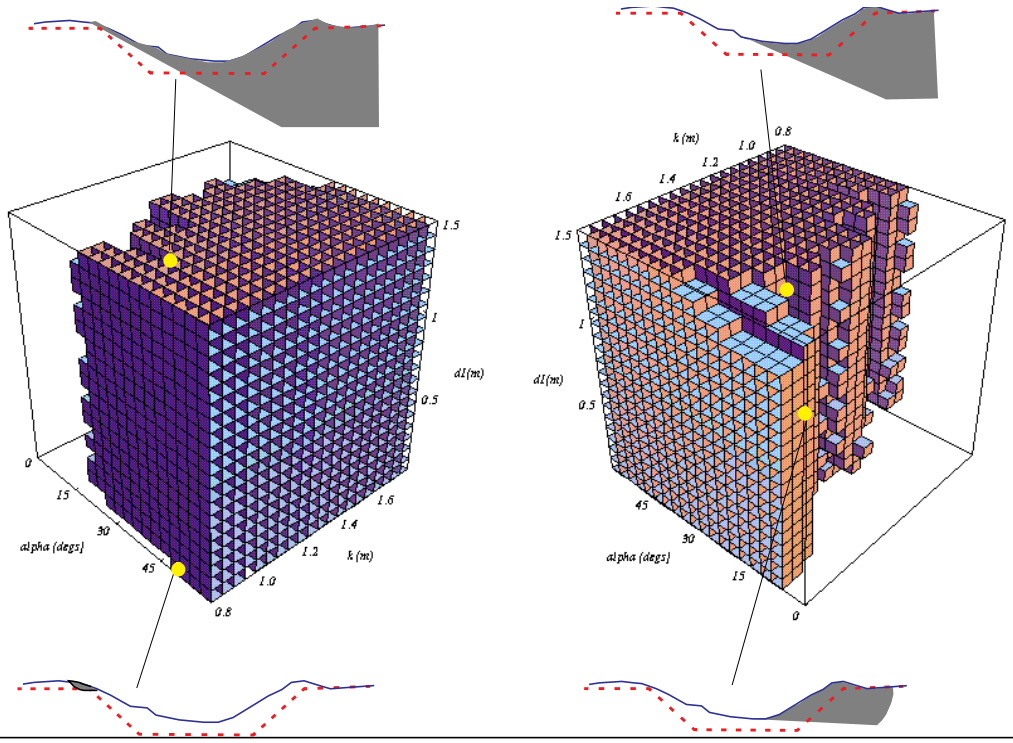


Figure 47 The set of plans that meet the “sensible” constraint. In effect, this constraint rejects those actions who trajectories are not completely below the surface of the soil.

(hence the name “sensible”), effectively by pruning the search space.

The composite geometric constraint plus the "sensible" constraint are shown in Figure 48. Finally, limiting the actions to those that sweep a volume that is at least 90% of the

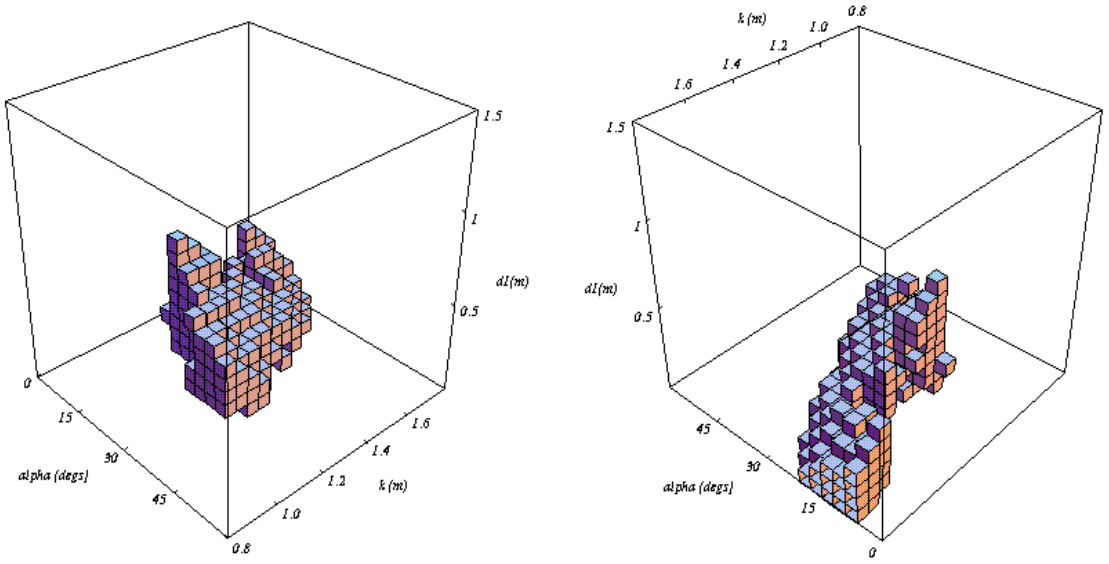


Figure 48 The set of plans that meet the composite geometric constraint as well as the "sensible" constraint.

bucket capacity produces the set of actions shown in Figure 49.

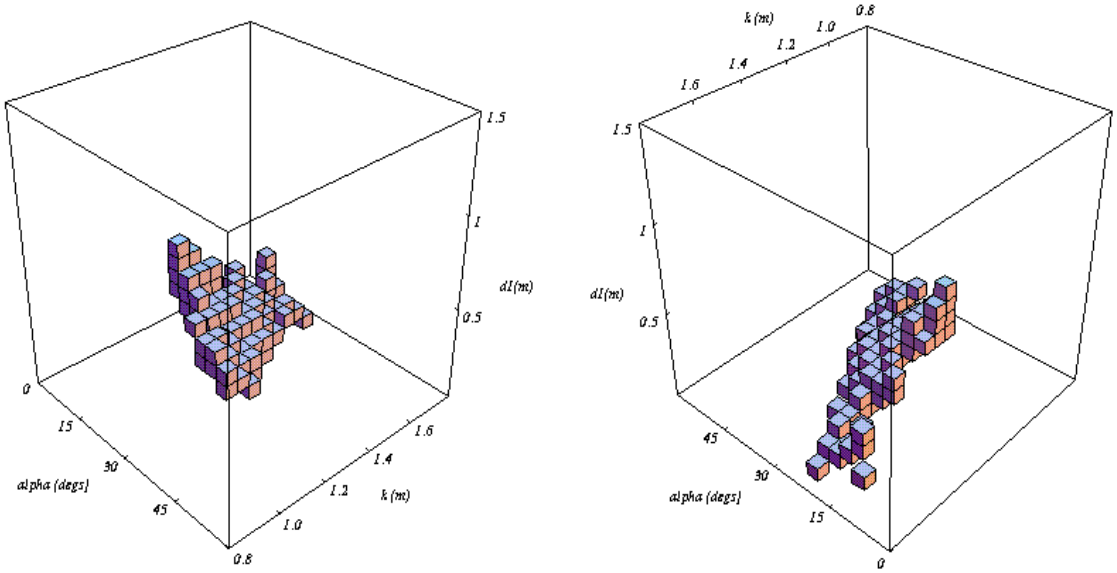


Figure 49 The set of actions that satisfy the composite geometric and "sensible" constraints, further constrained to those that sweep a volume that is at least 90% of the bucket capacity.

4.4 Geometric Constraints— Summary

This chapter has presented a methodology for imposing geometric constraints on action spaces for excavation. Constraints that encode the goal (shaping), excavator capacity (volume) and excavator workspace (reachability) were shown for the tasks of loading and trenching. In conjunction, these constraints define sets in a space spanned by the parameters that represent prototypical actions. A useful property of this methodology is that since it encodes the task and not a mechanism, it is relatively simple to generalize the methodology to various mechanisms.

Given a mechanism, a goal shape and the shape of the terrain, it is the job of the planner to identify a single plan from the feasible set of plans to be executed. Two points need to be made here:

- Picking a point in the feasible set guarantees that the plan will work (modulo correctness of the model), but not all feasible plans are equivalent in their utility. Other criteria (maximize volume, minimize time/joint torques, etc.) can be used to differentiate between the set of feasible plans.
- Identification of an “optimal” point within the feasible set does not require explicit computation of the constraint surfaces. A numerical method can be used that only requires a boolean function that decides whether a candidate point passes or fails a particular constraint.

Both issues are discussed in greater detail in Chapter 6.

Chapter 5 Force Constraints

If a robot excavator is infinitely strong, that is, if it can muster any torque required, then it is sufficient to consider only geometric constraints. More realistically, for robots with torque limits, the forces required to accomplish digging must be considered. Chapter 2 discussed some of the reasons why process models to describe the effect of excavating plans are not computationally tractable. Consider a more limited problem, that of predicting the resistive forces experienced during excavation. If it were possible to predict these forces for a candidate action and a given terrain, we would have a good criterion by which to further restrict the set of digs that are geometrically feasible.

This chapter first examines the mechanics of excavation. The analysis is motivated by a detailed look at the case of a flat blade moving through soil and we see how experimental data from our testbed compares to that predicted by the analytical model. Having found the analytical model deficient, empirical models using two “learning” methods are proposed. We consider how representation and methodology affect performance based on several criteria. Finally, this chapter shows how force predictions are used to compute the force constraint.

5.1 The Mechanics of Excavation

A principled way to approach the topic of resistive forces is to examine the mechanics of tools operating in soil. While it may not be possible to develop analytical models of excavating tools, there is much to be learned from the terremechanics literature. This section starts with some basic principles in soil mechanics that are necessary to develop the mechanics of soil-tool interaction. Next a static force analysis from the literature in agricultural engineering is reviewed, for the case of a flat blade moving through soil, and, finally the relevance of this model to the case of an excavator bucket is discussed.

5.1.1 Basic Soil Mechanics

To understand failure in a soil mass, it is necessary to review some basic theory in the area of soil mechanics. To start, let us develop the notion of *shear strength* and *shear stress*. The basis of soil strength is ascribed to Coulomb who noted that there appeared to be two mechanical processes that determine the shearing strength of a material [Coulomb76]. He noted that one process (friction) was proportional to the pressure acting perpendicularly on the shearing surface. The other process (cohesion) seemed to be independent of normal pressure. Coulomb modeled the shear strength of a soil, τ, as a sum of these two components:

$$\tau = c + \sigma \tan\phi \quad (2)$$

where c is the cohesion, σ is the normal pressure acting on the internal shear surface and $\tan\phi$ is the coefficient of sliding friction. ϕ is also called the angle of internal friction and is directly visible as the angle of repose of a pile of dry, uncompacted granular material like sand or sugar. When soil shears, it does so along a *failure* surface. The shear strength of a soil can be understood to be the resistance per unit area to deformation along the failure surface. Shear stress, on the other hand, is the pressure that pushes soil to move along the failure surface.

While many phenomena such as particle size, chemical composition, compaction, and moisture content may affect soil properties, gross soil behavior can in most cases be studied by the simple parameterization (c, ϕ). In case the soil is compacted or is saturated, these parameters (if they could be measured accurately) are “effective” values. One method to determine soil properties is to use specialized tools to make direct measurements. The simplest test to determine soil properties is performed using a “direct-shear box” in a laboratory setting (Figure 50(a)). A weighed sample of soil is placed in a container of which one part remains fixed at all times, while another part can be displaced. The soil is confined vertically by a force, N, while a horizontal force, F, is applied to the movable part of the container. The

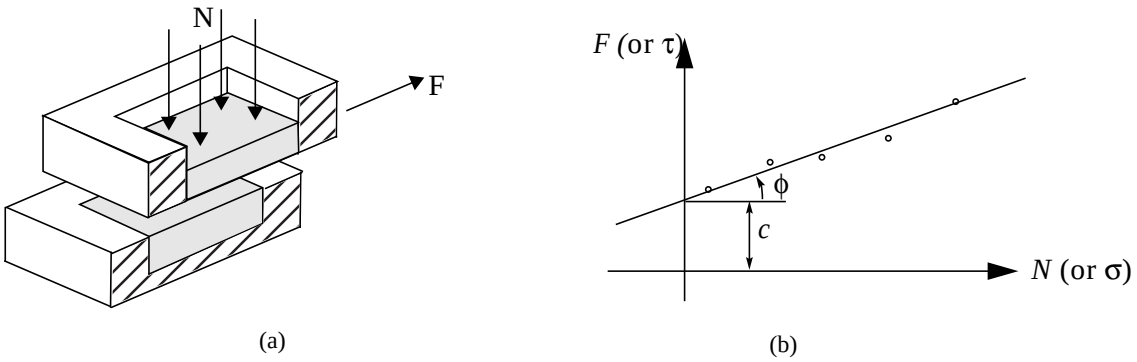


Figure 50 (a) A direct-shear test box. This is commonly used to determine soil properties. (b) Experimental determination of soil properties (cohesion and friction).

amount of force, *F*, required to shear the soil for varying values of *N*, can be plotted to determine values of cohesion and friction. Figure 50 (b) shows a hypothetical graph of experimental data obtained from a direct shear box.

Obtaining soil properties *in situ* (in the field) rather than in laboratory has the advantage that the soil is in its natural state. McKyes describes an instrument called an annular shear graph (Figure 51(a)) that measures soil properties (c, ϕ) directly, in much the same way as a direct-shear box [McKyes85]. Once again, the maximum values of torque, *T*, that can be sustained by the soil for different levels of vertical stress applied, *N*, are graphed to obtain the parameters of cohesion and friction. It is also possible to estimate soil properties through the use of an instrument like a cone penetrometer shown in (b) [Luth71]. Instead of direct measurement of soil parameters, the cone penetrometer measures the penetration force at a certain speed and soil depth. This is normalized by dividing force by the projected cone base area to give a cone index (a pressure unit).

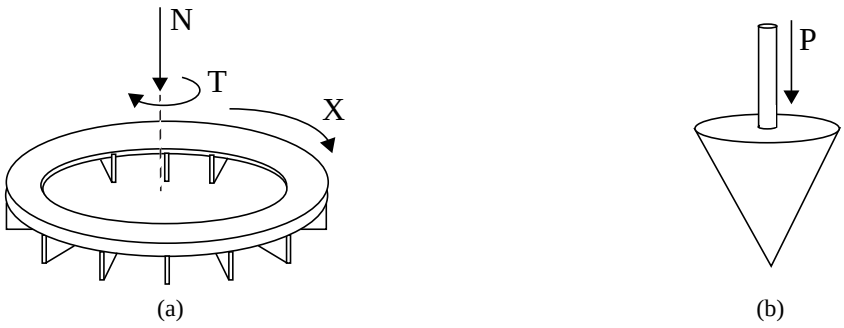


Figure 51 Tools used to determine soil parameters (a) annular shear graph (b) cone penetrometer.

Table 1 shows typical values for moisture content, cohesion (c), angle of internal friction (ϕ) and density (γ) for a few soils.

Table 1 Typical values for moisture content, c, and ϕ for a few soils.

Soil	moisture content (%)	c (kPa)	ϕ (degs)
Sandy Loam	15	1.7	29
Dry Sand	0	1.0	28
Heavy Clay	25	68.9	34
Lean Clay	30	17.2	17.2

To be complete it is also necessary to account for the coefficients of friction and adhesion between soil and the tool(δ, c_a). These are determined in the same way as the internal friction angle of soil, except a metal plate is used at the failure surface in a shear box. A convenient approximation used by many researchers is $\delta = \phi$. Typically, adhesion is small enough that it can be ignored.

5.1.2 The case of a flat blade moving through soil

Let's consider a simple case of soil-tool interaction to start. A flat blade (infinitely wide and high) is pushed along a horizontal trajectory as shown in Figure 52.

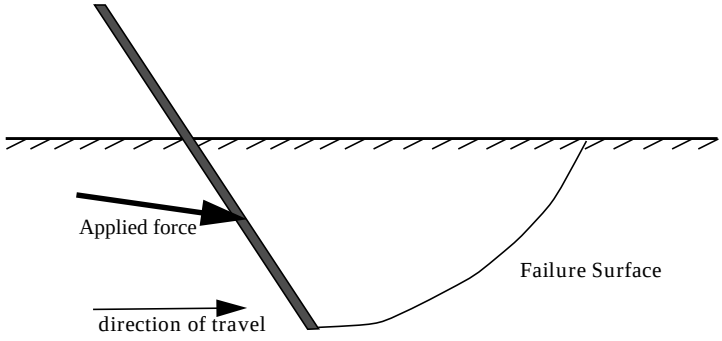


Figure 52 A flat blade moving through soil. Soil shears along an internal surface known as the failure surface and the wedge of soil slides along this surface.

Qualitatively, the total force applied by the blade is a function of the shape of the terrain, soil properties, soil-tool interaction, and tool motion. An examination of the mechanics

reveals that there are distinct phenomena that contribute to the total reaction force experienced by the blade. The total resistive force is comprised of forces to:

- *shear the soil along the failure surface*: Pushing on a section of soil results in failure along an internal failure surface, also known as *sliding* surface. The forces required are dependent on soil properties and the stresses present in the soil;
- *overcome friction between the blade and the soil*: present at the surface of the blade in contact with the soil;
- *overcome adhesion between the blade and soil*: present at the surface of the blade in contact with the soil;
- *accelerate the soil in front of the blade.*

Of these, the force necessary to shear the soil is most significant while the adhesion between the blade and soil is negligible. At low speeds, the forces required to accelerate soil in front of the blade can be ignored; hence our analysis will be *static*.

Figure 53 shows the state of the soil after a blade has been moved for a distance. The soil has sheared along successive failure surfaces and it is about to shear once again. The displaced soil has created an *overburden* (also called *surcharge*).

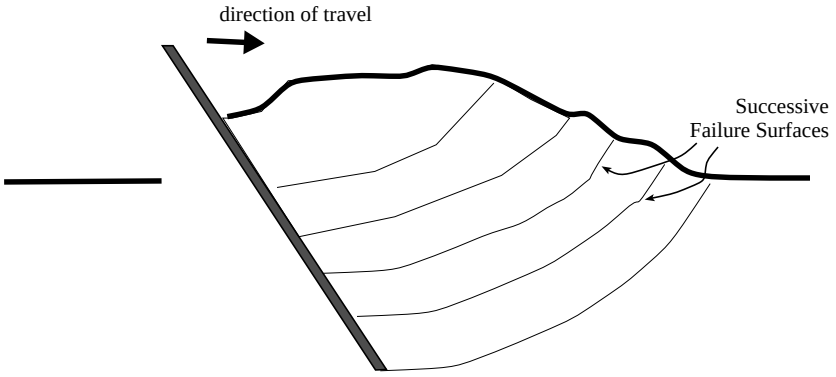


Figure 53 State of soil after some movement of blade.

The repeated shearing effect is clearly visible from a graph of the reactive force experienced by a blade being pushed through soil. In an experiment conducted for this thesis, a blade attached to the end effector of a robot manipulator was pushed through sand at constant speed and constant depth. Figure 54 shows the magnitude of the force recorded by a force sensor during this motion. The reactive force builds up incrementally as the stress on the consecutive shear surfaces rises. Upon shearing, the reactive force drops a little before continuing to build up again.

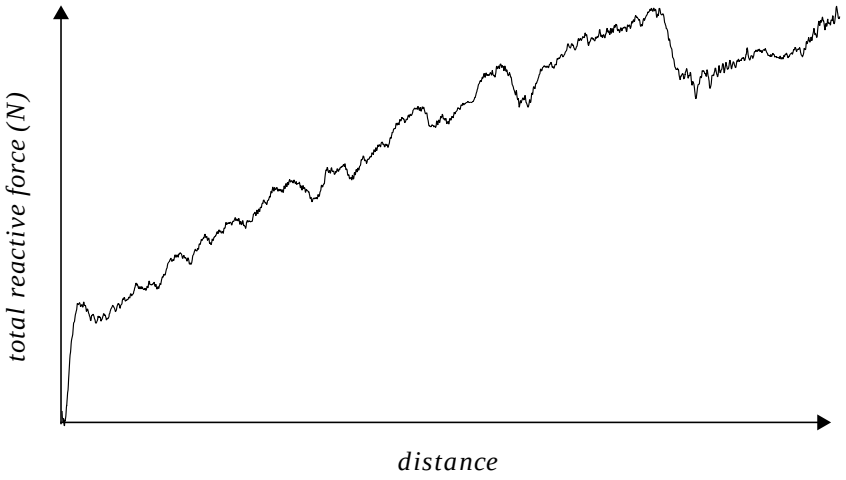


Figure 54 Reactive force experienced by a blade moving through soil. A force sensor (placed in between the blade and the end effector) measured force while the blade was in motion in the soil. Data was taken at 50 Hz and then smoothed with a 9 point moving average.

5.1.2.1 Predicting resistive forces using a static analysis

One reason to start with the case of a wide, flat blade is that this scenario has been examined extensively in civil engineering and agricultural engineering. Civil engineering has developed an analysis of the passive pressure applied by foundations and retaining walls on adjacent soil masses [Terzaghi47]. Additionally, several researchers in Agricultural engineering have developed analyses of resistive forces experienced by tools necessary to perform tillage.

This section reviews an approximate equilibrium model for the mechanics of a flat blade, called the *wedge* model[1]. This model is approximate because due to simplifying assumptions such as the shape of the failure surface and the pressure distribution of the surcharge. Several researchers have shown both theoretically and experimentally that the failure surface looks like a logarithmic spiral [Terzaghi47, Hettiratchi66]. Instead, this analysis approximate the failure surface by a straight line. A model of the case just before the soil fails along a new failure surface using this approximation shown in Figure 55.

With the exception of the force required to accelerate the soil, this analysis takes into account all the phenomena mentioned earlier— the forces required to shear the soil, the forces required to overcome friction and adhesion between the soil and the blade. As the blade moves from left to right, soil rides up the metal and up the failure surface. The blade

1. after the shape of mass of soil that is presumed to move along the failure surface.

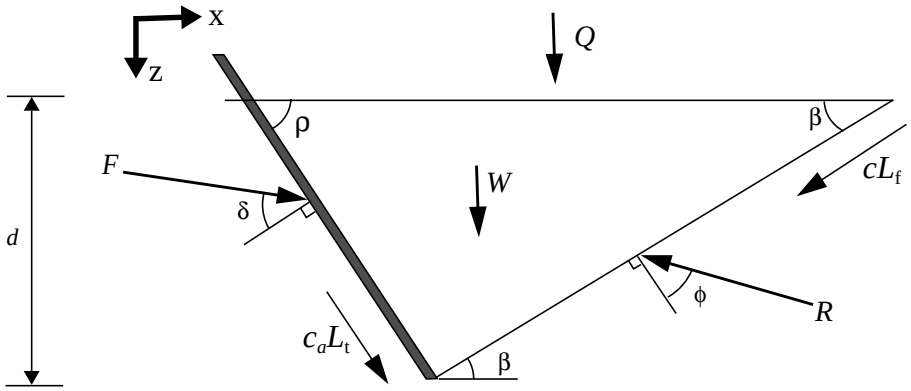


Figure 55 Static equilibrium analysis using an approximation of the failure surface. L_f is the length of the failure surface, L_t is the length of the tool, Q is the surcharge on the wedge and W is the weight of the soil wedge. After figure in [McKeys85].

resists the soil with a force equal to the sum of the perpendicular force that the blade provides, the frictional force (soil-tool) and the adhesion (soil-tool). Similarly the soil resists shearing by a force equal to the frictional force (soil-soil) and the cohesion along the entire failure surface.

Force equilibrium equations for a blade of unit width can be written as:

$$\sum f_x = F\sin(\rho+\delta) + c_a L_t \cos\rho - R\sin(\beta+\phi) - cL_f\cos\beta = 0 \quad (3)$$

$$\sum f_z = -F\cos(\rho+\delta) + c_a L_t \sin\rho - R\cos(\beta+\phi) + cL_f\sin\beta + W + Q = 0 \quad (4)$$

Solving for F:

$$F = \frac{W + Q + cd[1+\cot\beta\cot(\beta+\phi)] + c_a d[1-\cot\rho\cot(\beta+\phi)]}{\cos(\rho+\delta) + \sin(\rho+\delta)\cot(\beta+\phi)} \quad (5)$$

The weight of the soil in the wedge is given by:

$$W = \gamma g\frac{d^2}{2}(\cot\rho+\cot\beta) \quad (6)$$

where γ is the density of soil, and g is the acceleration due to gravity.

Presuming the surcharge is uniformly distributed, that is, the soil displaced by the movement of the blade is uniformly spread over the wedge, the force due to the surcharge is written as:

$$Q = qd(\cot\rho+\cot\beta) \quad (7)$$

where q is the surcharge pressure.

The total force F, (ignoring adhesion) for a tool of width w, can now be written in another form:

$$F = \left(\gamma g d^2 N_\gamma + c d N_c + q d N_q \right) w \qquad \angle F = \pi/2 - \rho - \delta \tag{8}$$

where

$$N_\gamma = \frac{\cot\rho + \cot\beta}{2\,[\cos(\rho + \delta) + \sin(\rho + \delta)\cot(\beta + \phi)\,]} \tag{9}$$

$$N_c = \frac{1 + \cot\beta\cot(\beta + \phi)}{[\cos(\rho + \delta) + \sin(\rho + \delta)\cot(\beta + \phi)\,]} \tag{10}$$

$$N_q = \frac{\cot\rho + \cot\beta}{[\cos(\rho + \delta) + \sin(\rho + \delta)\cot(\beta + \phi)\,]} \tag{11}$$

and $\angle F$ is the angle of the resistive force from the horizontal. (8) is written in this way because it is a familiar form also known as the *Fundamental Equation of Earth Moving* (FEE) proposed [Reece64]. N_γ, N_c, N_q are separated out because they are functions of only the rake angle (ρ) and the friction coefficients (ϕ, δ). These terms have been tabulated for various values of ρ, ϕ, δ and allow for approximate calculation of the reactive force [Reece64, McKyes85].

It remains to be shown how the slip angle β is determined. Coulomb proposed that the shape of the failure surface is such that the force required to produce failure is minimum. As mentioned above, this shape has been accurately modeled using logarithmic spirals, but here we will approximate it by a plane. Since the soil frictional properties alone determine the failure zones in soil cutting, the most likely slip line in the soil does not depend on the magnitude of the cohesion, nor on the surcharge pressure. The most appropriate angle of failure is that which causes N_γ to be minimum. In effect, this identifies path of least resistance for the soil to fail.

When different values of β (between 0 and 90 degrees) are tried, N_γ varies schematically as shown in Figure 56. The most likely value of β is found at the minimum value of N_γ, and is then used to calculate the other terms.

Prediction of the reactive force starts with calculation of β and the N factors. These are only dependent on the soil parameters and the geometry of tool. If the soil properties are constant and the tool moves without changing its rake angle, the N factors need not be recomputed. (22) is solved for the tool at regular increments along the line of travel. At each

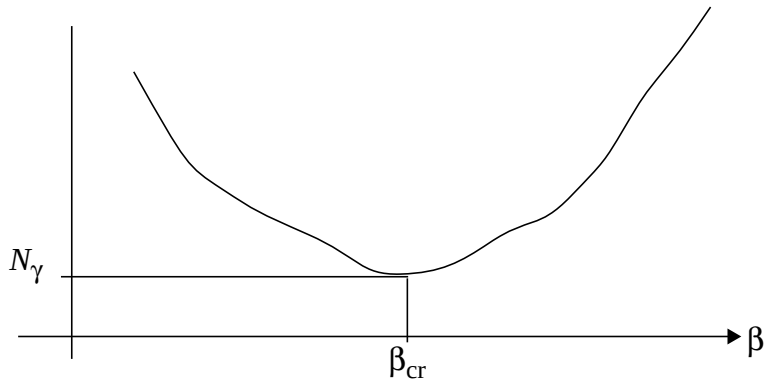


Figure 56 Determination of the angle of the failure surface, and $N\gamma$

increment, the amount of soil displaced (by the movement of the tool) is calculated and is used as a measure of the surcharge (Figure 57).

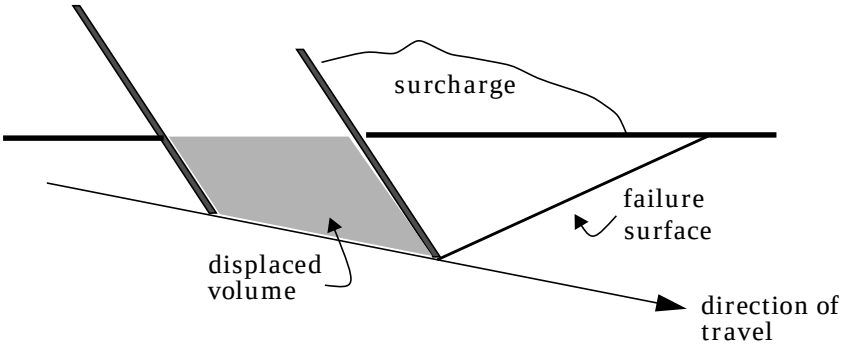


Figure 57 Iterative computation of resistive force. The volume of soil displaced is computed at regular time intervals. This volume is used to estimate the force due to surcharge, pressing down on the wedge of soil that slides along the failure surface.

Although it is possible to obtain order of magnitude predictions using this procedure for a tool moving along an arbitrary path, there are several caveats to using this analysis:

- all dynamic effects such as the forces required to accelerate the soil mass, are ignored;
- the shape of failure surface is approximated;
- the surcharge is presumed to be uniform over the failure wedge. In fact, the surcharge due to the soil excavated is concentrated close to the blade;
- it is assumed that soil is not confined in any way. This allows us to ignore dilation of the soil, which is an otherwise important effect;
- only forces just before failure are modeled. If the blade moves purely horizontally, the soil mass in front of the blade is constantly being brought to failure. However, if

the blade moves in arbitrary directions, not only will the wedge geometry vary, but also failure might not always be imminent.

5.1.3 Extension to the excavation scenario

The above sections have developed an approximate model for an infinitely wide blade moving through soil along a horizontal trajectory at a constant depth. It is worth noting some differences between the case we have modeled and the case of an excavator bucket digging during typical digging motions.

First, excavation tools have finite width and maybe longer than they are wide. The tillage literature has developed special models for long tools because they cause failure in the soil lateral to direction of movement as well as along the direction of movement [Hettiaratchi67, McKyes77]. This means that in the general case, the two dimensional analysis developed above will be insufficient. However, McKyes notes that in the case a tool has side-walls (as with excavator buckets), the walls help push the soil into the bucket and constrain the failure to a volume directly ahead of the bucket. This suggests that in this case a two dimensional analysis will suffice if a typical excavator bucket is used [McKyes85]. Second, the cutting surfaces of many excavation tools (dozer blades, excavator bucket) are curved and in some cases shaped such that after a certain amount of soil is “scooped”, the soil becomes captive and additional travel results in a new cutting surface— the soil at the surface of the bucket acts as the cutting surface. Osman has suggested a more refined model of failure surfaces to deal with the issue of curved blades [Osman64] but there has not been any principled work on the latter topic. Third, excavation often occurs in uneven terrain resulting in varying depth as opposed to the relatively constant depth assumed in the case of tillage. Lastly, excavation tools are required to rotate at various rates as opposed to tillage tools which are typically required to translate at a fixed orientation.

To test the applicability of the model, a series of experiments were conducted in which an excavator bucket mounted on a robot manipulator was used. Below we examine how well the wedge model is able to predict the resistive forces measured in experiments. The inside of the bucket is modeled as a flat blade and nominal soil parameters ($\phi = \delta = 30$degs, $c = 0$, $\gamma = 2000$ kg/m^3) that approximately correspond to soil in our testbed are used. In these experiments depth, approach angle, α, rake angle, ρ, and tool speed were varied. Each is examined in turn below.

5.1.3.1 The effect of varying depth

The excavator bucket was made to follow a horizontal trajectory at a constant depth through the soil. Three experiments were conducted at each of three separate depths (Figure 58).

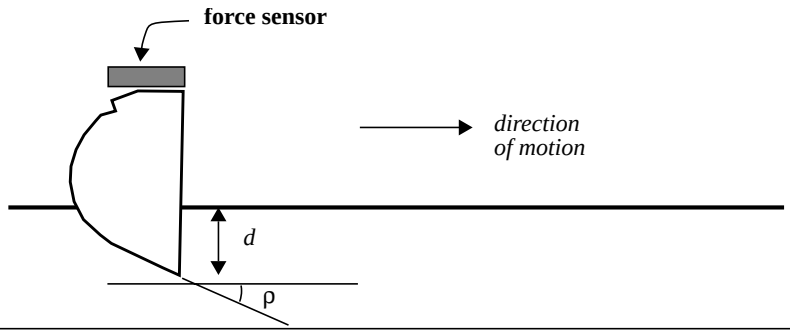


Figure 58 An experiment to verify the effect of varying depth, *d*, using an excavator bucket equipped with a force sensor. The rake angle, ρ, was held constant at 30 degrees. The force data were transformed to measure forces at the tip of the bucket.

Figure 59 compares predicted force with measured force. We note that the experiments conducted moved the blade over a short distance (the bucket doesn't have a chance to fill up) and well approximate the scenario of tilling. For this case, the wedge model provides fairly good results.

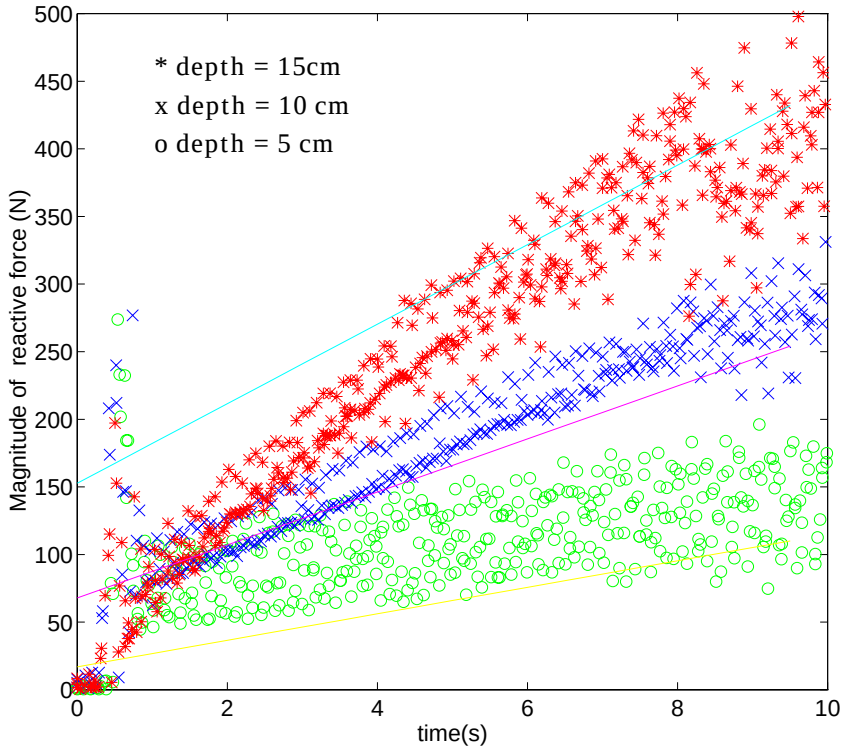


Figure 59 Measured resistive forces (symbols) vs. predictions based on wedge model (solid lines). In this experiment, an excavator bucket was made to follow a horizontal trajectory at three separate depths (5cm, 10 cm, 15 cm). Three experiments were conducted for each depth.

5.1.3.2 The effect of varying rake angle

In another set of experiments the rake angle, ρ, was varied as in Figure 58.

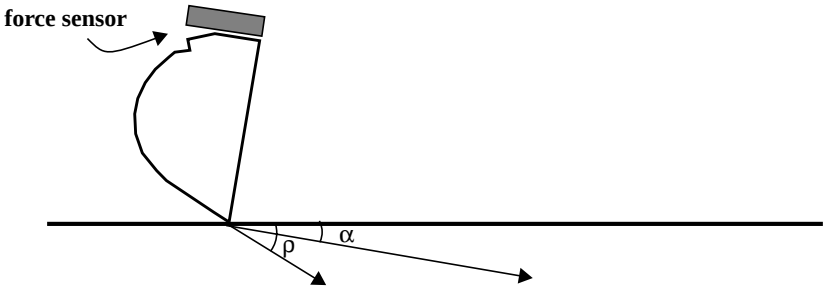


Figure 60 An experiment to verify the effect of varying rake angle, ρ, The approach angle, α, was held constant at 10 degrees.

All three factors (N_γ, N_c, and N_q) as well as the shape of the failure surface are affected by a change in the rake angle. Experimental data show that the resistive force is non-monotonic with the rake angle (Figure 61). That is, as the rake angle increases, the resistive force

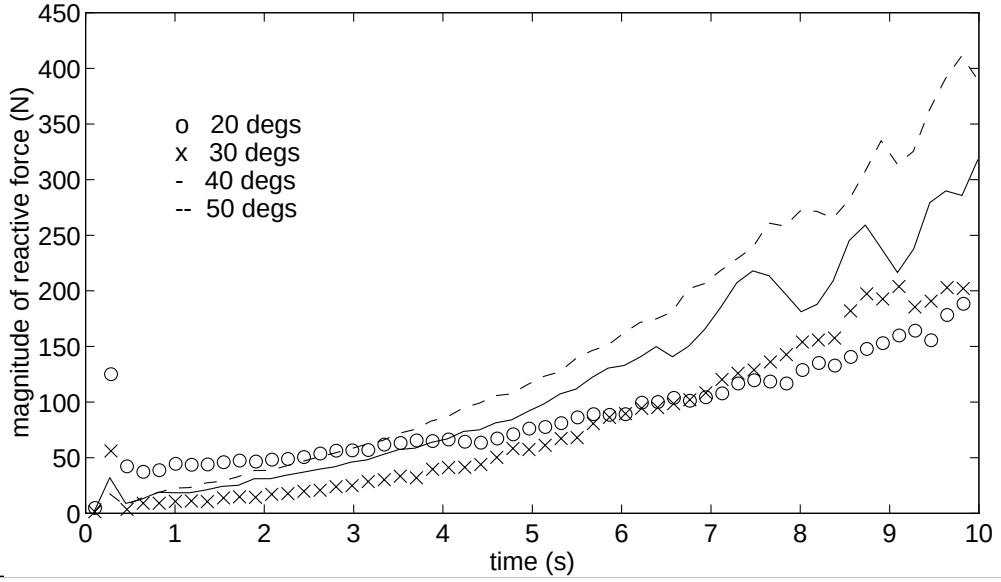


Figure 61 Results from four experiments in which the rake angle was varied (ρ = 20, 30, 40, 50 degs).

drops (the whole profile is lower) first and then increases again. There is a straightforward explanation for this— a graph the angle of the failure surface as a function of the rake angle shows that it varies non-monotonically also (Figure 62).

5.1.3.3 The effects of varying speed

Increasing speed of the bucket results in higher normal stress on the sliding surface and thus increases the shearing strength of the soil. Additionally, the result of pushing the soil

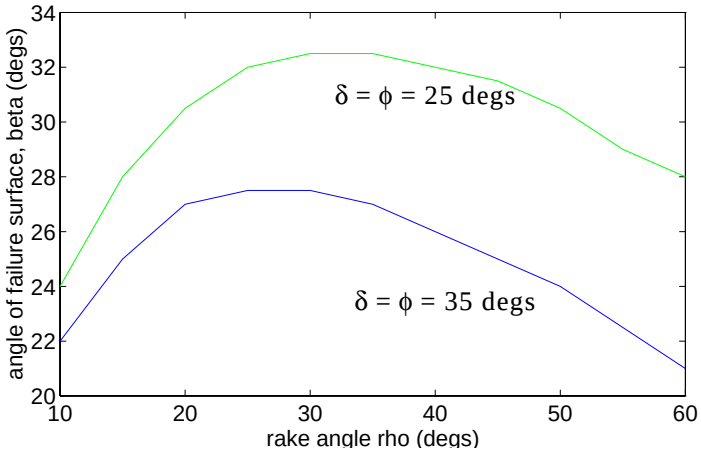


Figure 62 Angle of failure surface as a function of the rake angle computed using the wedge model.

fast is that soil particles are compressed together, closing microscopic gaps and also making the soil harder to shear. Since the wedge model is static, it is not possible to predict these effects (Figure 63).

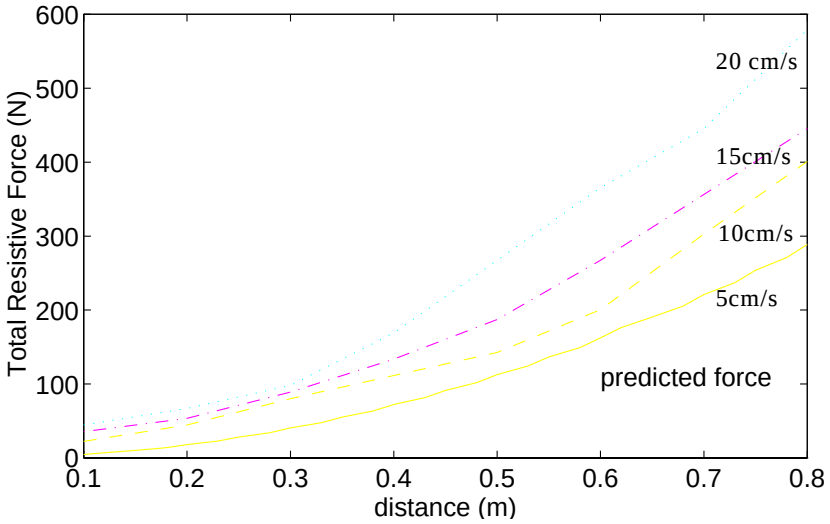


Figure 63 Varying the speed of the tool (α = 10, ρ = 30). Since the model is *static*, it predicts force much better at slow speeds.

5.1.3.4 Predicting the force during a realistic excavation using the wedge model

So far only very simple motions of the excavating tool have been considered. The FEE can be used to predict the forces during realistic excavation. Experimental results show that for small motions of an excavating tool the wedge model provides reasonable prediction, but as the tool proceeds, the quality of the prediction decreases. We have recorded resistive

forces during excavation and have compared these to forces predicted by the wedge model (Figure 64). Later we shall see how prediction can be improved using a variety of methods.

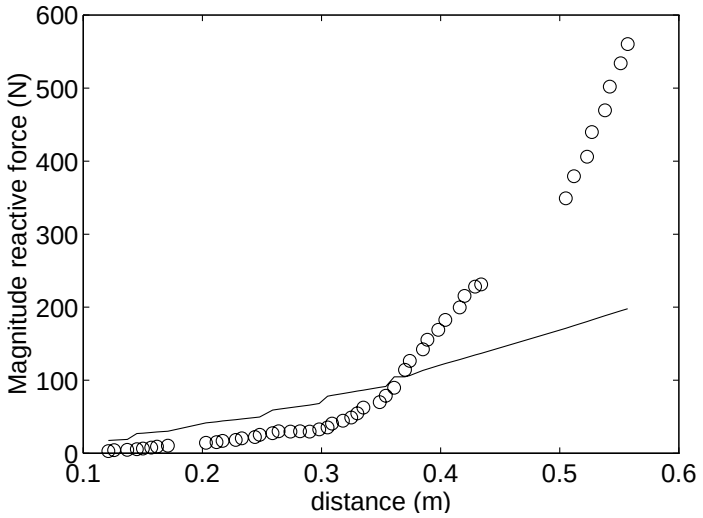


Figure 64 Use of the wedge model to compute resistive force during excavation. Recorded data is indicated by circles while force predictions using the wedge model (FEE) are shown by a solid line.

5.1.3.5 What can be learned from the mechanics of earthmoving

Although the wedge model will not suffice for accurate prediction of resistive forces experienced during excavation, there are several important lessons to be learned from a study of the mechanics:

- Whatever method used, a two-dimensional treatment will suffice. Specifically, the resistive force is affected most by the terrain directly in front of the cutting surface and the volume that shears along the failure surface.
- The resistive force is a function of soil properties, tool geometry and the trajectory of the cutting surface. Soil properties vary tremendously based on chemical composition, particle size, water content and compaction. These properties are not easy to incorporate unless the measurement is done *in situ*.
- The mechanics suggest basis functions which can be used to model the resistive force. For example, (22) tells us that the resistive force is proportional to the depth, the square of the depth and varies linearly with the width of the cutting surface.
- Several non-linear effects impact the resultant resistive force and thus if linear models are used, these models will have to be local rather than global.

In the ideal case, an excavator would have some prior information about the nature of soil it is to excavate. It would then adapt very quickly to the terrain conditions, refining models that change as the nature of the terrain changes. It is unlikely then that a straightforward implementation of the wedge model will be able to provide the necessary fidelity.

The model is deficient in accounting for certain phenomena such as soil compression and acceleration forces and additionally, we would like a method that is able to improve and adapt to varying conditions on its own. To fold in all of these considerations, a method that is able to incorporate its experience into future predictions is needed.

5.2 Learning to Predict Resistive Forces

For purposes of this thesis, the term “learning” is used loosely mean the ability to improve performance with experience. Of specific interest are function approximation schemes that are able to better predict resistive forces based on experimental data. This section examines two fundamental factors in function approximation— *representation* and *methodology*. It is the contention of this thesis that good representations significantly impact learning methods and in particular that representations motivated by physical models provide a substantial benefit. Three learning methods are examined in some detail and their performance is compared using several criteria.

Discussion starts with a physical model— the FEE. Given a set of force data along with shape of the terrain and the trajectory followed by an excavating tool, it is possible to solve for the unknown parameters that best fit the observations. For various reasons this model is awkward to work with and is deficient in capturing the complexity of soil-tool interaction during excavation. In response this thesis has developed an alternate formulation motivated by the wedge model but that is easier to work with.

Ideally, we are looking for a set of basis functions, G_i, that are linearly related to the unknowns. That is:

$$G_1 k_1 + \ldots + G_n k_n = F \tag{12}$$

where the k_i are the unknown parameters. This section shows how G is selected and how the semantics of G affect performance. For every force reading taken during excavation, it is possible to write a separate equation. This is a familiar case of an over-determined system (more equations than unknowns) and can be readily solved.

Three “learning” methods— *global regression*, *memory based learning* and *neural nets* have been examined. These methods are compared in terms of accuracy, training time, prediction time and memory requirements.

5.2.1 Inverting the FEE

One way to proceed is to claim that if the prediction of resistive forces lacks accuracy because it is hard to know soil and soil-tool parameters a priori, then the form of the *Fundamental Equation of Earthmoving* (FEE) can be used to empirically determine these parameters[2]. That is, given the variables that the robot excavator can control during excavation as well as measured forces, it is possible to solve for the unknowns. This line of thinking presumes that the structure of the wedge model is valid for excavation when in fact, analysis and experience indicates significant phenomena that the wedge model does not account for. However, it is instructive to see how far one might get with this model.

Since the FEE is highly non-linear, analytical inversion to determine δ, ϕ, c and γ is not possible and a numerical method is needed. If the determination of unknowns is posed as a minimization of the error between observed outputs and predicted outputs, a large number of methods can be used (for example, the *downhill simplex* method and *conjugate gradient descent* [Press88]). Commonly, these methods start from an initial point (often just a good guess) and follow the gradient of the error until a minimum is found. Since there is no guarantee that the minimum found is the global minimum, some methods add exponentially decreasing noise into the gradient descent with the hope of escaping local minima [Bohachevsky86]. A step is always taken if it improves the error metric but is sometimes taken (based on an exponentially decreasing random variable) even if it results in a worse metric. In general, non-linear optimization is a difficult process requiring iterative refinement.

Since the wedge model is presented only for comparison, a brute force search for the values of the unknown variables (c, ϕ, δ, γ) that best fit the observed force data is conducted. This involves an exhaustive search that is exponential in the number of unknowns. For D unknowns that are discretized in m intervals and n measurements, this method requires $m^D n$ computations.

5.2.2 Selecting a linear basis

To review the problem at hand, we are looking for a method to predict future resistive forces based on past experimental data. The resistive force, F, is a function of the robot's action (A) and the world state (W):

$$F = f(A_i \times W_i) \tag{13}$$

2. In the ideal case, determination of unknown parameters is based on a few experiments and then these parameters are used in later predictions.

If G is chosen to be an appropriate encoding of A,W it is possible to write

$$A_1 k_1 + \ldots + A_n k_n + W_{n+1} k_{n+1} + \ldots + W_{n+m} k_{n+m} = F \quad \textbf{(14)}$$

where n and m are the number of variables needed to encode A and W respectively. Since both the planar components of F are of interest, two separate equations, one for each of F_x and F_z, will be necessary. Now it is possible to use a least-squares method to determine the unknown k_i.

It remains to be shown how to choose the basis functions G. Three ways to build a linear basis are suggested. The first is a "naive" formulation that is based simply on a compact geometric representation of actions and world states. Next, an analysis of the mechanics is used to motivate the basis functions. Finally, a reduced formulation, based on empirical results is suggested.

Consider the trenching action in Figure 65. Before the digging starts, the shape of the

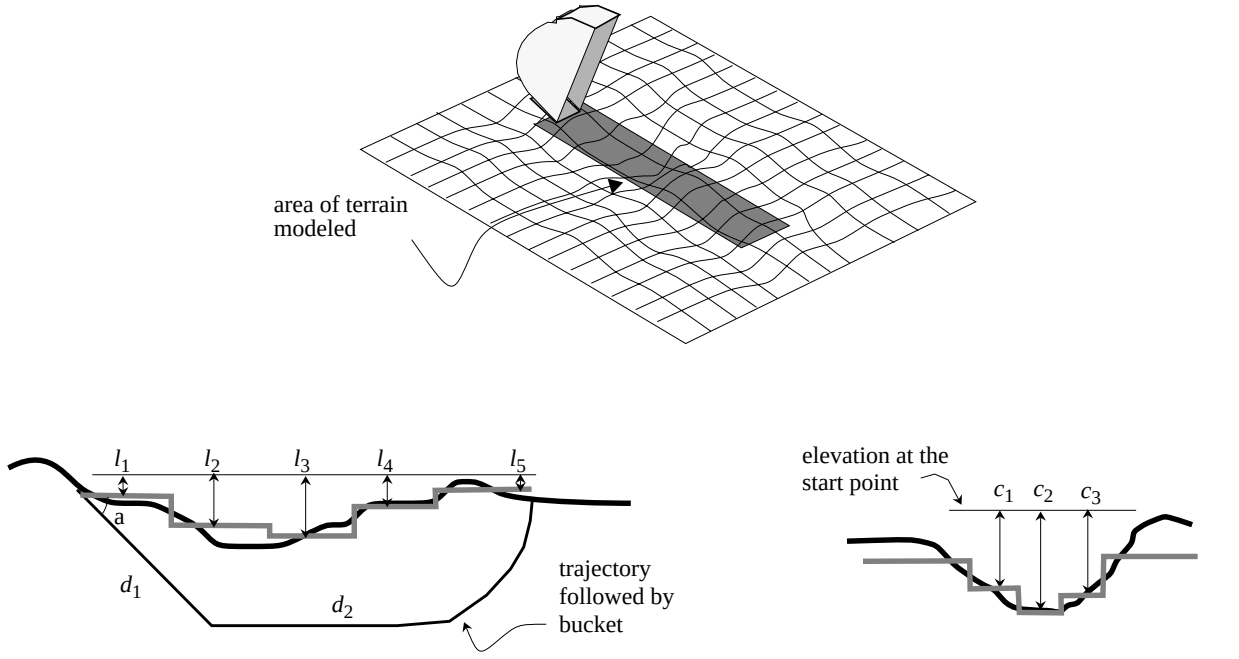


Figure 65 Encoding a trenching action and the state of the terrain. Here we encode an atomic dig with the same variables used to span the action space (α, d_1, d_2). The world state is represented by a sub-sampled elevation profile along the trench to be excavated (l_i) and across the trench (c_i).

terrain is recorded as a set of l_i and c_i from an elevation map[3]. While the excavator moves

along the specified path, the position of the bucket and the forces at the bucket tip measured by a force sensor are also recorded. For each force reading the following basis is created:

$$\alpha, d_1, d_2 \qquad l_1, l_2, l_3, l_4, l_5 \qquad c_1, c_2, c_3 \tag{15}$$

For a particular dig, α and l_i are fixed. d_1, d_2, c_1, c_2 and c_3 vary with the position of the bucket (Figure 66).

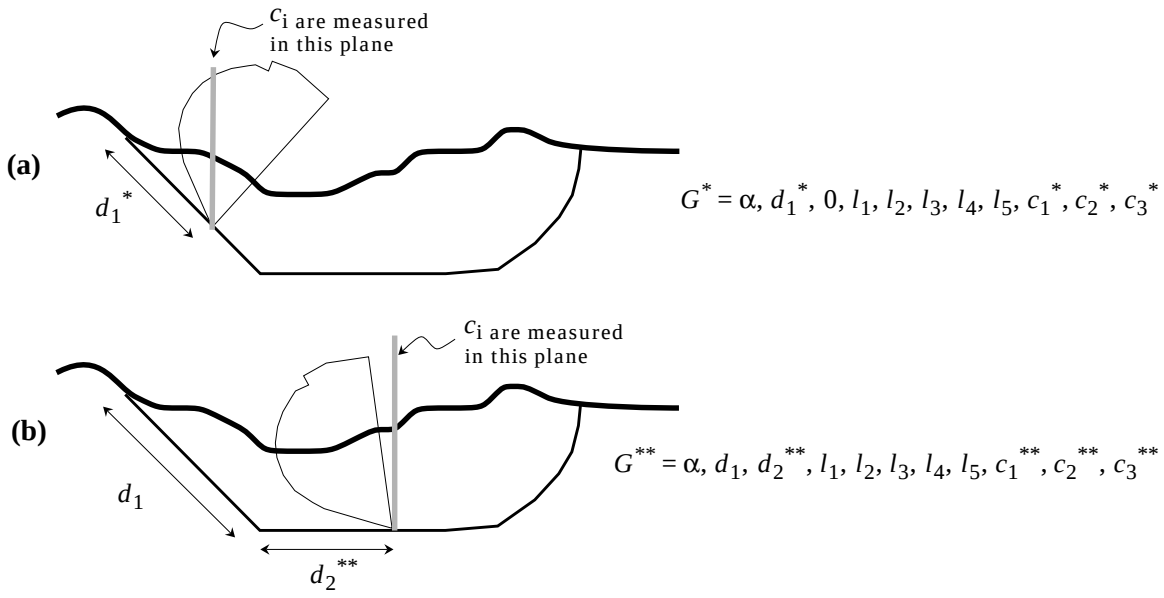


Figure 66 Two examples of computing *G*. The bucket moves along a trajectory described by (α, d_1, d_2). (a) the bucket has moved a distance d_1^* along d_1 (b) the bucket has moved d_2^{**} along d_2.

The above formulation is based purely on geometry. There has been no attempt to use any of the insights gained from the analysis of the mechanics. The next formulation will be *smarter*. For instance, it is known that the resistive force is proportional to the square of the depth of the tip of the cutting surface, and that the resistive force depends on the profile of soil in front of the cutting surface. It is also known that the resistive force depends on the volume of soil displaced (surcharge) and on the angle of the cutting surface. Hence one plausible *mechanics-based* formulation is:

$$\rho, d_1, d_2 \qquad l_1^2, l_2^2, l_3^2, l_4^2, l_5^2 \qquad c_1^2, c_2^2, c_3^2 \qquad v \tag{16}$$

where v is the volume of soil displaced. Figure 67 shows how the semantics of these parameters differs from the previous one.

3. A terrain mapping system is used to determine terrain elevation at the nodes of a regular grid. More details are given in Chapter 7.

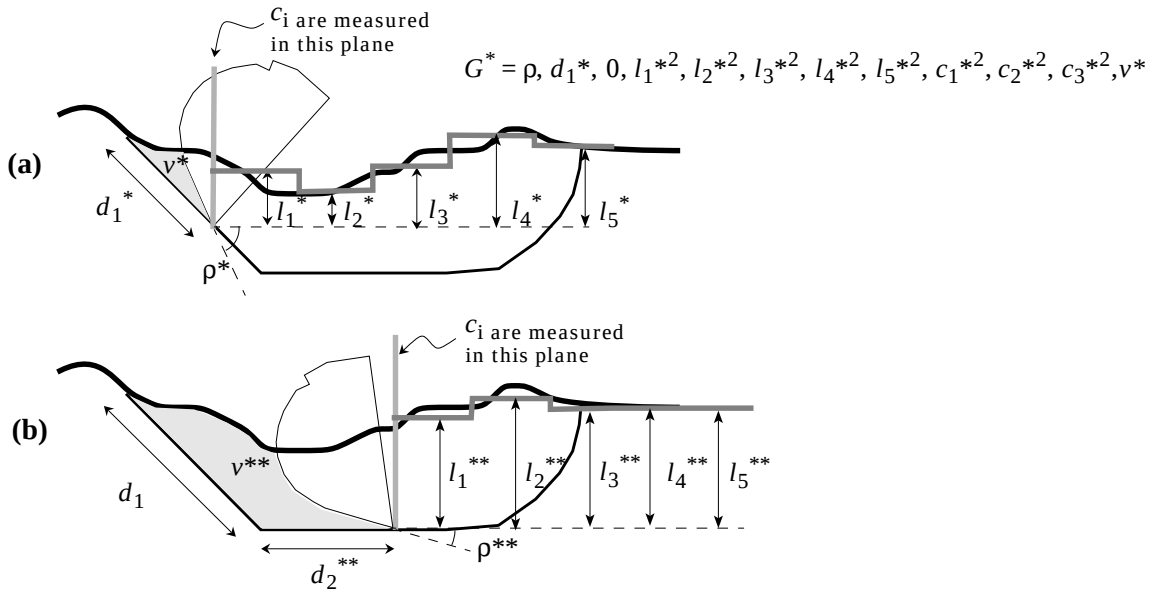


Figure 67 Computing *G* based on an analysis of the mechanics. Terrain ahead of the cutting surface is encoded as a sequences of depths (squared values used). Additionally, we encode the rake angle, and volume of soil displaced.

Experimentation shows that some of the variables in the basis do not have much of an effect. For example, the further one gets from the cutting surface, the less likely the terrain is to affect the resistive force. For comparison consider a reduced *G* (17). Section 5.2.4 com-

$$\rho, d_1, d_2 \qquad l_1^2, l_2^2, l_3^2 \qquad c_2^2 \qquad v \tag{17}$$

pares how differences in formulation affect prediction performance.

5.2.3 Learning Methods

The attention now turns from representation to methodology. Since it is difficult to illustrate learning methods in a high dimensional space, the three methods examined for this thesis are compared using a simple one dimensional example Consider the data set shown in Figure 68. The underlying function is

$$y = 3\sin(x) + 4\cos(0.2x) + (0.02x)^2 \quad \textbf{(18)}$$

To this equation is added gaussian noise with zero mean and variance = 1. Additionally, data in the range [x: 4 → 6] has been removed. Below, the three learning methods are used to predict the underlying function given the noisy and incomplete data.

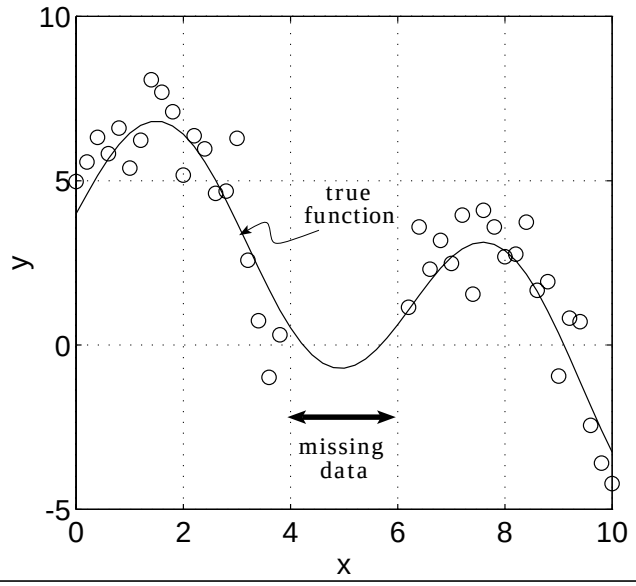


Figure 68 A one dimensional example (one input and one output). A simple function (18) is corrupted with noise and some of the data are removed.

First, a linear global model is used to fit the data using a linear least-squares method. That is the basis functions are linearly related to the unknowns over the whole input space. Even if the structure of the underlying function is not known exactly, it is possible to predict the function reasonably with approximate basis functions. It is often not tractable, however, to represent complex systems with simple global models. This is partly because complex systems are often characterized by very different functions that are local to regions in the input space. Even if there was some global model that would fit the data well, how would we arrive at it? There are two solutions. One is use linear local models and the other is fit nonlinear global models to the data. These correspond to locally weighted regression and neural net back-propagation.

5.2.3.1 Global Regression

Given a set of basis functions *G*, the unknowns, *K*, are found by solving the normal equations:

$$K = \left(G^{\mathrm{T}} G \right)^{-1} G^{\mathrm{T}} y \tag{19}$$

Given a data set as in Figure 68 with no information about the underlying system, the simplest formulation of *G* is

$$x \qquad 1 \tag{20}$$

This models the function as straight line— *K* is composed of the slope (*m*) and the *y* intercept (*b*). Applying this solution produces the line shown in Figure 69(a). However, if it is possible to reasonably guess the basis functions there is a good chance of improving prediction. Note that the basis functions need not be linear themselves but they must be related linearly to the unknowns. For example, *G* might be formulated as:

$$\sin(x) \qquad \cos(x) \qquad x^2 \tag{21}$$

Even though this representation does not capture all the non-linearities in underlying function, prediction is greatly improved (Figure 69 (b)). Of course, if the basis is known exactly, then the function can be approximated very well (Figure 69(c)).

It is noteworthy that even in the presence of significant noise and in the absence of neighboring data, global regression is able to produce good predictions. This is because this method relies heavily on a valid global model. (19) is a *batch* method— all data are processed simultaneously. This method can be modified into an *iterative* scheme where data are incorporated sequentially, as they arrive.

$$\hat{K}_k = \hat{K}_{k-1} + R\left[Y_k - \hat{K}_{k-1} X_k \right] \tag{22}$$

where $\hat{K}_{k-1}$ and $\hat{K}_k$ are the estimates of the unknown parameters at step *k* and *k*+1 respectively, Y_k are the observed dependent variables while X_k are the independent variables at step *k*. *R* is a gain that encodes how much a new data point should be relied upon. In selecting *R*, typically, one is faced with a trade-off between sensitivity to noise and performance.

In the batch case, the complexity of this method is dependent on the method used to solve a linear over-determined system. Assuming LU decomposition is used to invert the matrix $G^{\mathrm{T}} G$, the number of computations required are $Dn^2 + nD^2 + D^2/3$ where *D* is the

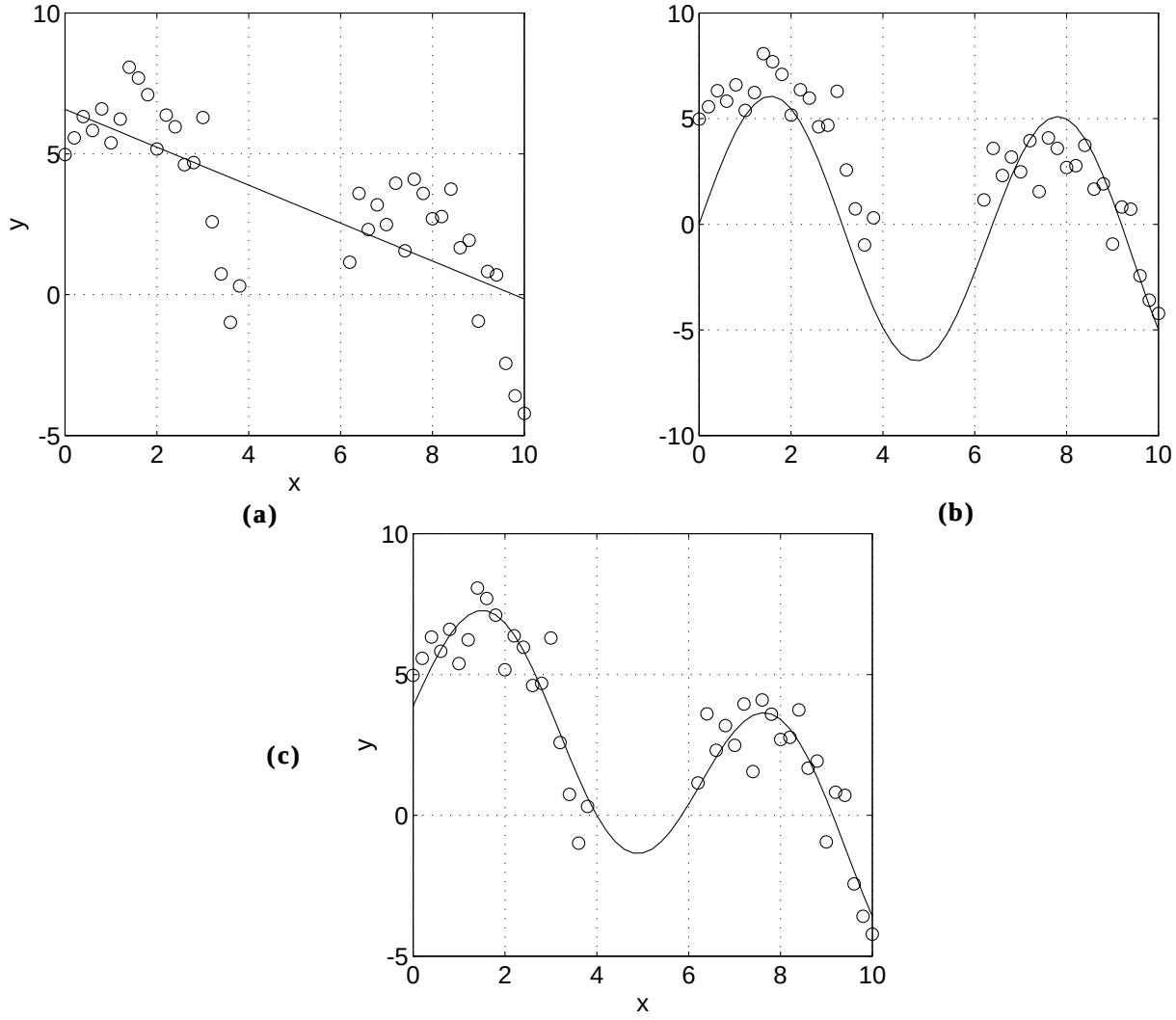


Figure 69 Fitting the data set from Figure 68 using global regression. The Solid line represents the approximated function. (a) The basis encodes a linear relationship between *x* and *y* (b) The basis uses approximate functions (c) The basis uses exact functions.

number of unknowns and *n* is the number of data points. Once the parameters are known, prediction time is constant.

5.2.3.2 Memory Based Learning

Instead of modelling all the data simultaneously, predictions could be based on local models that are fitted to the data in the neighborhood of a query point. Such methods require that all experiences $\{(X_1, Y_1),..., (X_n, Y_n)\}$ are explicitly stored in memory and hence the term “memory based”. A simple memory based method is *nearest-neighbor*. In this method, when a query is made, the memory is searched for the *n* nearest points in the input space. The dependent values of these points are averaged to provide a prediction. The use of

nearest-neighbor for our example is shown in Figure 70 (a). Since there is no model whatsoever, this method is prone to noise. The only recourse is to average over a larger neighborhood. Instead of simply averaging the *n* nearest points, a linear model could be formed over this subset. That is, the surface of the function is approximated by local hyperplane patches. A further extension is to weight the points used in the linear model by their proximity to the query point. Intuitively, the further a point is in the input space from the query point, the less it should be weighted. This method is commonly called Local Weighted Regression (LWR) and has been described in [Cleveland88].

Formally, define W as a square matrix with as many rows as data points and weights w_i on the diagonals (all other entries are zero). Then, weighted regression is written as:

$$K = [G^{\mathrm{T}} W^{-1} G]^{-1} [G^{\mathrm{T}} W^{-1} F] \quad \textbf{(23)}$$

Several weighting schemes have been proposed [Atkeson91] [Moore93]. For example:

$$w_i = \left(d_i^2 \right)^{-p} \qquad w_i = \exp\left(-\frac{d_i^2}{2K_w^2} \right) \quad \textbf{(24)}$$

where d_i^2 is the Euclidean distance of point i to the query point, p and K_w are parameters that determine how local the regression will be. The latter example in (24) provides a gaussian weighting function and K_w is called the *kernel* width. As usual, there is a trade-off in ability to represent the underlying function and noise rejection. For example, Figure 70 (b) shows the results of LWR with $K_w = 2^{-6}$. This provides better results than nearest-neighbor but is still affected by noise. In contrast, Figure 70(c) shows LWR with $K_w = 2^{-3}$. In this case, the kernel is so wide (distant points are given greater weight) that the resulting prediction is too smooth.

As with global regression, knowledge of the underlying function will help. Figure 70(d) shows kernel regression with $K_w = 2^{-3}$, but with the approximate basis function from (21). Comparison with Figure 69 (b) shows that using a local model can do a lot to improve fidelity of fit, even if the basis functions are approximate. Conversely, the weaker the model, the more prone local methods will be to noise and missing data.

Use of LWR is predicated on the search for the *n* nearest neighbors to a query point. A naive implementation requires *n* distance computations, that is, $O(nD)$ operations. A faster implementation is to use K-D trees to partition the input space such that determination of nearest neighbors is improved [Friedman77]. The time to compute a prediction using LWR with K-D trees is given by

$$min(\alpha_1 n, \alpha_2 \log n) \cdot D^2 \quad \textbf{(25)}$$

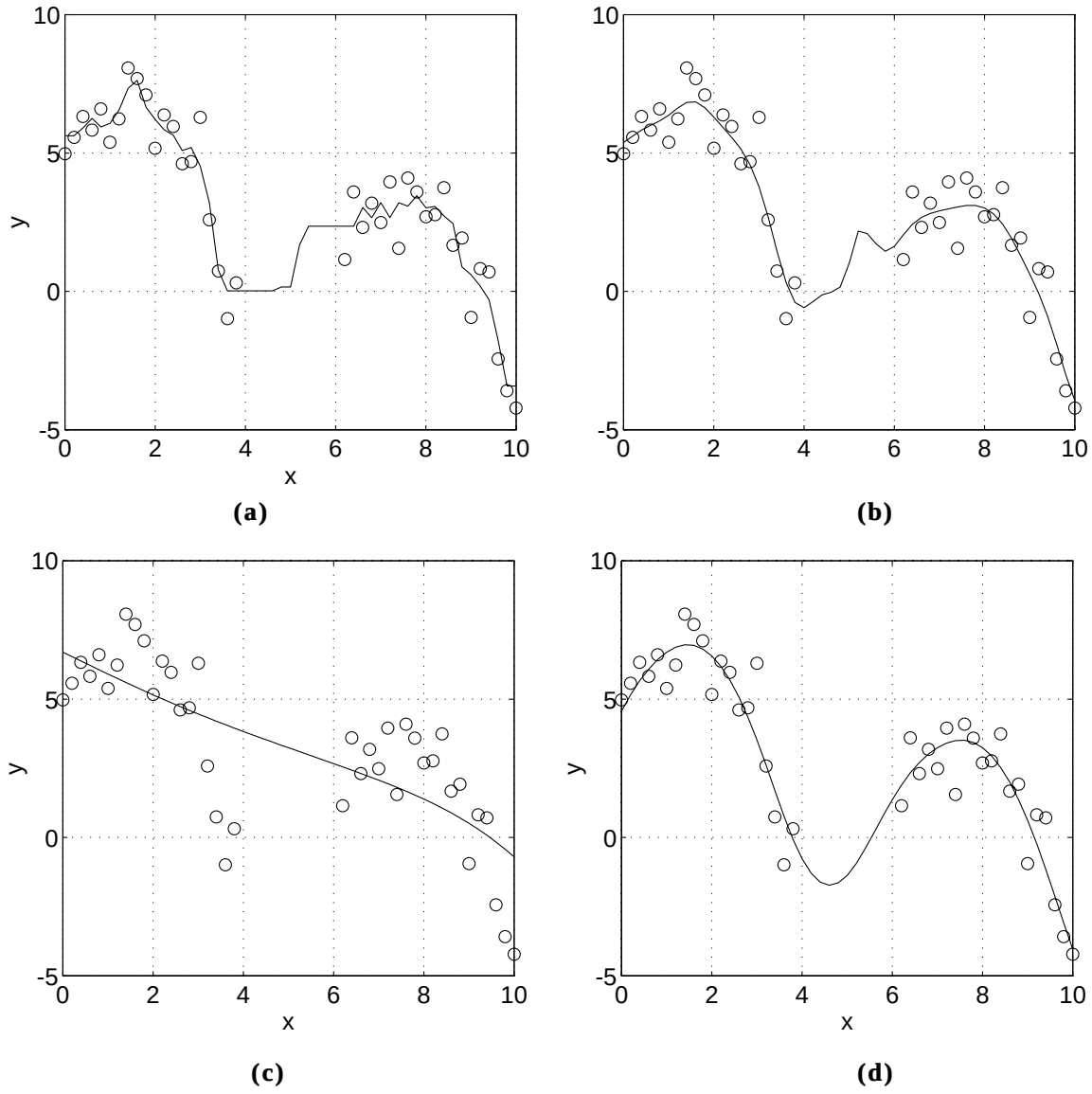


Figure 70 Fitting data set from Figure 68 using memory based methods. (a) three nearest neighbors are averaged. (b) locally weighted regression with a narrow smoothing kernel. (c) locally weighted regression using a wide smoothing kernel. (d) locally weighted regression using a wide smoothing kernel with approximate basis functions (24).

where α_1 and α_2 are constants $(\alpha_2 \gg \alpha_1)$ that encode overhead costs. The quadratic relationship of computation time to the number of variables in the input space is evident in Figure 71(a). Computation time is asymptotically logarithmic in n although for large D (> 8), the overhead costs dictate that the prediction time will vary linearly until the size of the data set, n, is very large as is the case in Figure 71(b). One way to deal with large data sets is to train on a randomly chosen subset. Accuracy is surprisingly unaffected by reducing the number of data points (provided they are chosen randomly from a larger, denser data set).

Figure 71 (b) shows fairly good prediction results for as few as 600 points in the training set (0.2 of total set).

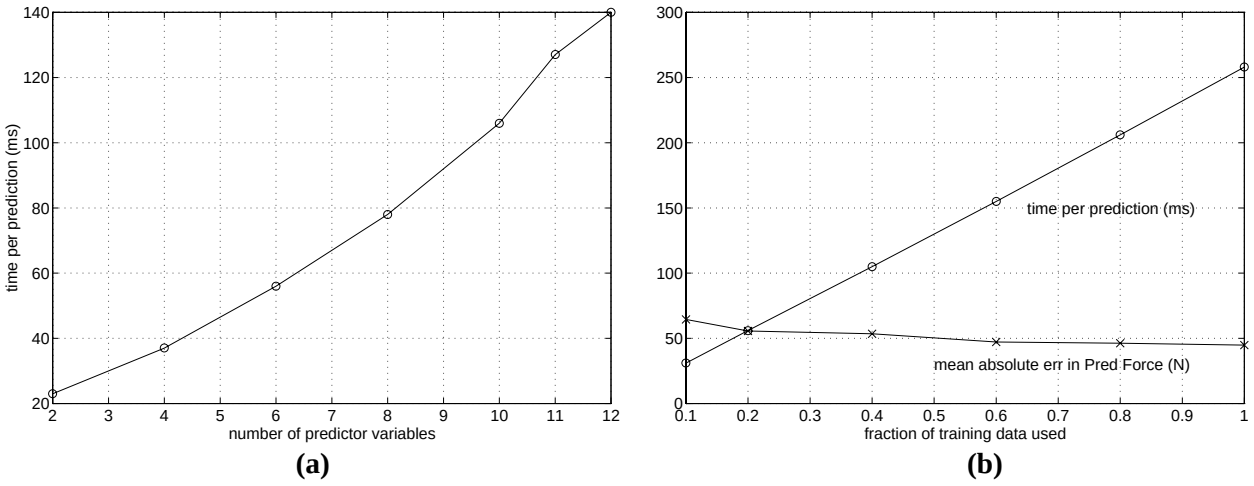


Figure 71 (a) Prediction time vs. number of input variables using locally weighted regression. (b) Prediction time and error as a function of fraction of data set (3000 points) used. Training points were randomly selected from the data set.

Typically, the use of weighted regression as a learning method is accompanied by a "training" phase that searches for the kernel width and the best scaling of the input parameters that best minimize the sum of squares error. Unfortunately, Moore's training scheme is not applicable in a straightforward manner to our case. This is because sometimes, force data from an entire dig should be treated as an outlier[4]. The standard method randomly selects training set from the data set. If points from a group that should be treated as outliers are used in the training set, then the outliers get modeled also and result in a poor predictions when used in actual experimentation.

Even though the basis functions are motivated by the mechanics of excavation, it is not clear if the variables chosen are good predictors. We have used a multivariate regression spline strategy (MARS) developed by Friedman to determine the relative importance of the input variables [Friedman88, Freidman91]. This allows one to scale the parameters in a simple manner, and, more importantly, it allows for some variables that are empirically verified to be poor predictors to be dropped from the basis completely.

4. Outliers are due to disparate initial conditions in our digging experiments. For example, sometimes, the bucket does not start at the surface of the soil but a little above it or worse, below the surface, in which case all the force data are at least shifted by an initial offset.

5.2.3.3 Neural Nets

Another way to deal with complex functions is to use a non-linear mapping between the unknowns and the basis functions. One such method is a back-propagation neural net. Back-propagation works by repeatedly showing a network sample inputs along with the true (perhaps noisy) outputs, while the network learns by adjusting its weights. The knowledge that the network acquires is encoded in its numerical weights.

It is common to connect inputs and outputs with one or more layers of computational nodes called *hidden units*. The simplest computational nodes sum weighted inputs and pass the result through a non-linearity. A node is characterized by an internal threshold θ and by the type of nonlinearity. A commonly used nonlinearity is called the *sigmoid* and is defined:

$$\sigma(\alpha) = \frac{1}{1 + e^{-(\alpha - \theta)}} \quad (26)$$

Formally, a single hidden layer sigmoidal neural network approximates a function Y as:

$$\hat{Y} = \sum_{j=1}^{n} \sigma(w_j X + \theta_j) \quad (27)$$

where w_j are the weights at the hidden units.

Continuing with the example data set from the above sections, consider the case when only the input and output are available. Figure 72 (a) and (b) shows the ability of a network with 10 hidden units and 50 hidden units, respectively, to predict the underlying function.

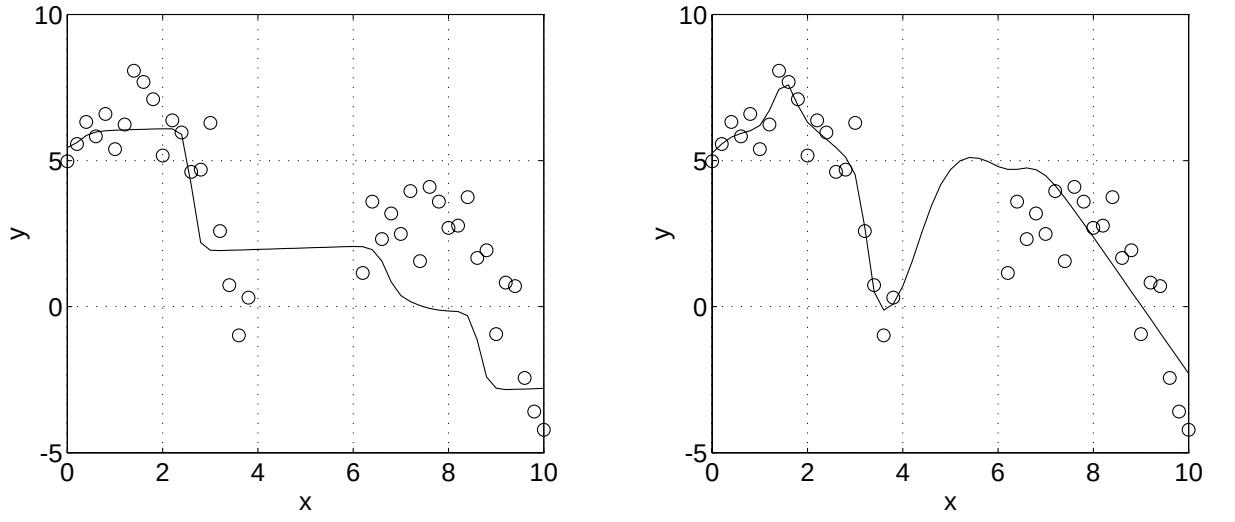


Figure 72 Prediction of the function from Figure 68 using a single layer network (a) 10 hidden units (b) 50 hidden units. Each net was shown the noisy data set 5000 times.

It is evident that an increase in the number of hidden units allows for higher fidelity in representing the underlying function. Since there is no model whatsoever, predictions are at their worst in regions where data are missing. As before, approximate knowledge of the underlying function can help substantially. Figure 73 shows a significant improvement when the basis (18) is used. Here a network with three inputs and one output has been used. Only 5 hidden units were used to produce a prediction that is qualitatively superior to the result produced with larger networks when they are used without any help from a model.

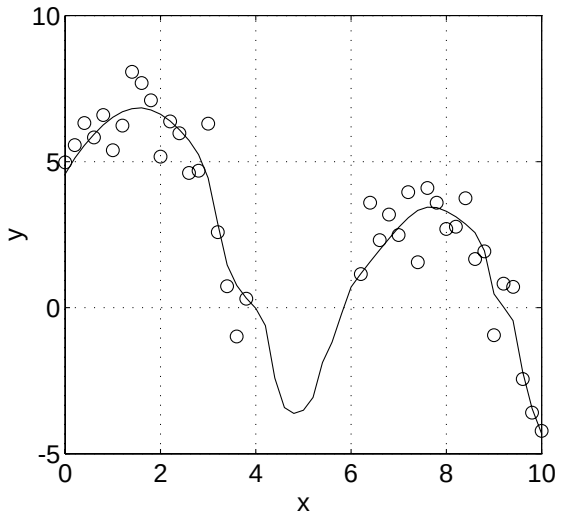


Figure 73 Prediction of the function from Figure 68 using a single layer network using 5 hidden units and the basis *G* from (21).

Training neural networks to approximate a function is somewhat of a black art. Many parameters must be adjusted and in general, the designer has to try a large number of permutations to gain satisfactory results. For example, Figure 74 shows that the mean error in prediction of resistive force is non-monotonic with the number of hidden units. Unfortunately, the number of hidden units for which the error is lowest varies with the number of times the data is presented to the neural net, the learning rate, and the properties of the data themselves. In general this number is found through experimentation. Addition of fur-

ther hidden units can be harmful because they overfit the data. However, once the weights

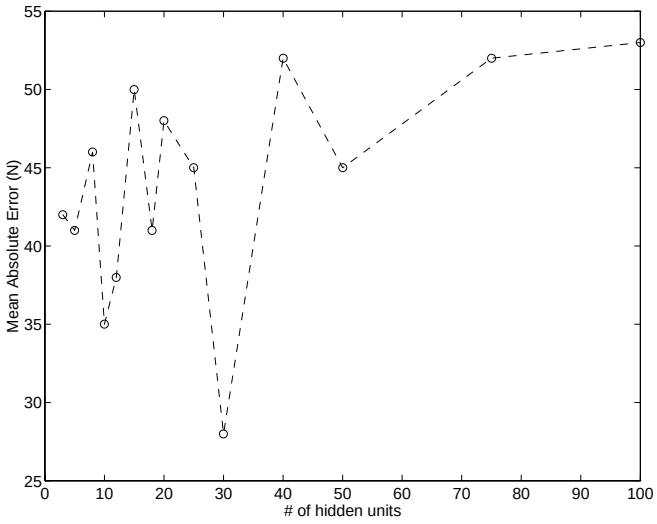


Figure 74 Mean residual error in prediction of the *x* component of the resistive force as a function of number of hidden units. The error is lowest for 30 hidden units for this set of data. After this, the function is overfit by additional hidden units.

have been computed, prediction is very fast and the representation is very compact.

5.2.4 Comparison of learning methods in the excavation domain.

With the above one dimensional example done the discussion now turns to using these methods with the basis functions developed in 5.2.2. Consider two examples of excavation (Figure 75). Force data collected from these experiments is compared to force data predicted based on prior experiments (approximately 4000 measurements from 30 digs are used).

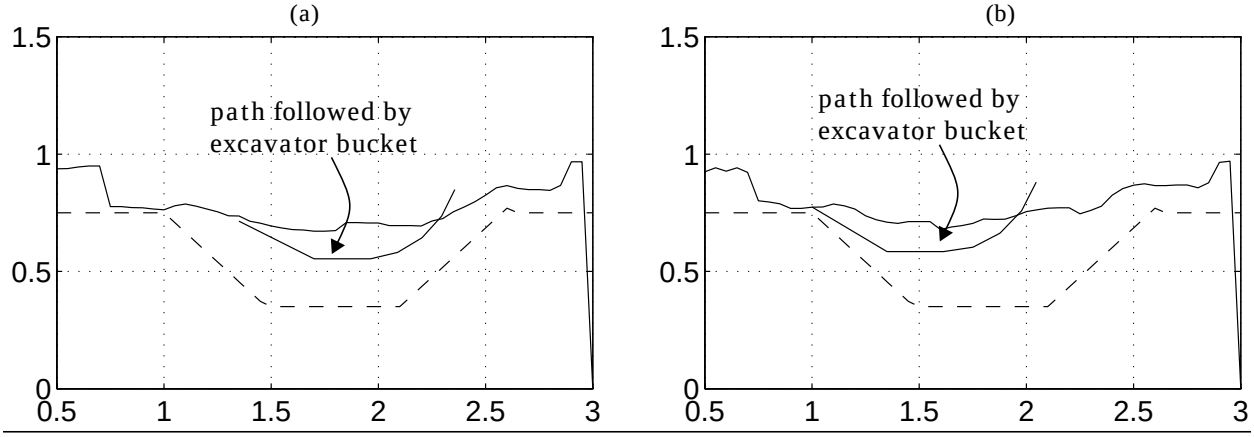


Figure 75 Two digs. (a) and (b) show elevation profiles of the soil in testbed along the length of the trench that is to be created (marked by dashed lines). Forces at the bucket tip were measured at approximately 8 hz. k_a = 1.35, k_b = 1.0, α_a = 25 degs, α_b = 30 degs, $d_{1a} = d_{1b}$ = 0.38 m.

5.2.4.1 Predictions using the wedge model

Brute force search to find the parameters from the wedge model that best fit the observed force data results in a number of solutions that very close in their error metric. 4000 force measurements were used to fit four parameters (ϕ, c, δ, γ). Ranking sets of unknowns by their ability to minimize the error, it is noticed that there are a large number of solutions within 1% of each other, suggesting that there are numerous local minima in the function and/or that the surface of the function has large flat spots.

The advantage of this method is that the representation is very compact and once the unknowns are found, prediction is fast. On the downside, the form of the wedge model and the accompanying FEE is hard to work with— the non-linear nature and multiple local extrema make data fitting a difficult task. In our study, we have used a brute force method that requires a very large number of evaluations of the FEE. Discretizing the four unknown parameters into just 20 intervals, the search required several days of computation.

Although it is possible to find a set of parameters that produce an average absolute error of approximately 100 newtons[5] in the magnitude of the resistive force, qualitatively the predictions produce average estimates, and do not fit the measured data well at any part of an excavation. Accuracy is improved over the prediction done using nominal parameters in that the errors are attenuated but the fundamental deficiency in accounting for second order effects remains (Figure 76). The wedge model does provide a bound on the errors that will be acceptable by other methods.

5.2.4.2 Global regression

The net result of global regression is two sets of 12 parameters (one each for f_x and f_z) (Table 2). It is instructive to examine the magnitude of these weights. To a first approximation they tell us that distance along d_1 and d_2, depth of the bucket tip, l_1, and the volume displaced, v, are significant factors. The weights also show a rapidly decreasing effect from elevation of the soil as one gets further away from the bucket tip (l_1. l_5). This agrees with the intuition developed in the preceding analysis.

That global regression produces better results than the wedge model is testimony to the fact that the wedge model is not able to capture the complexity of soil-tool interaction during excavation. However, the improvement is not major and it seems that global regression is also only capable of capturing first order effects (Figure 77).

5. This is the best possible case since this number is simply the residual error from fitting the entire data set.

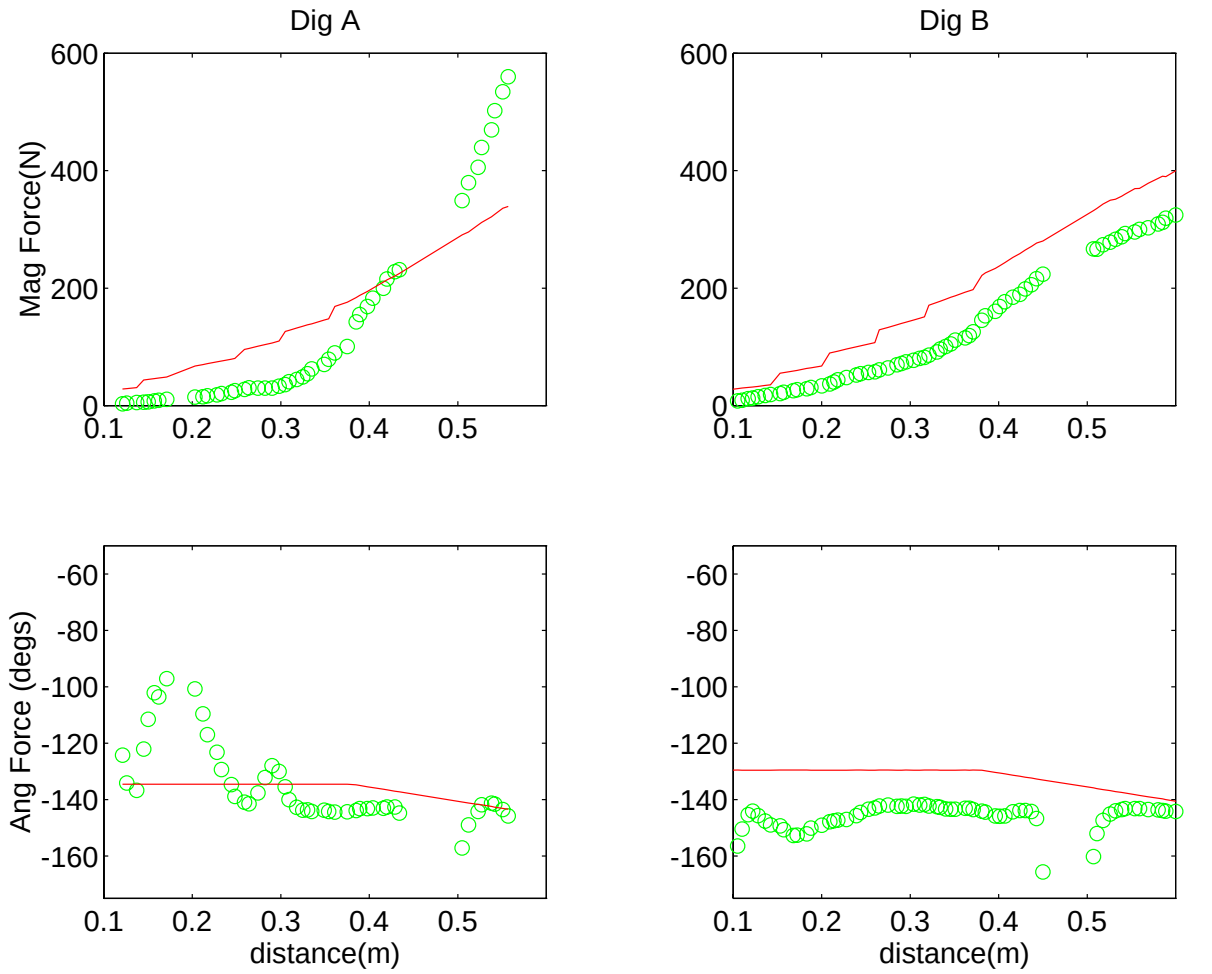


Figure 76 Prediction of resistive forces using the wedge model with parameters determined by brute force search. Measured force data is marked with 'o'. Predicted force data is marked by solid lines.

Table 2 Weights found by global regression

	ρ	d_1	d_2	l_1^2	l_2^2	l_3^2	l_4^2	l_5^2	c_1^2	c_2^2	c_3^2	v
x	-21	236	252	-463	-12	-35	-7	19	99	102	39	-264
z	-35	277	-49	-368	-26	-41	26	-14	120	186	64	-249

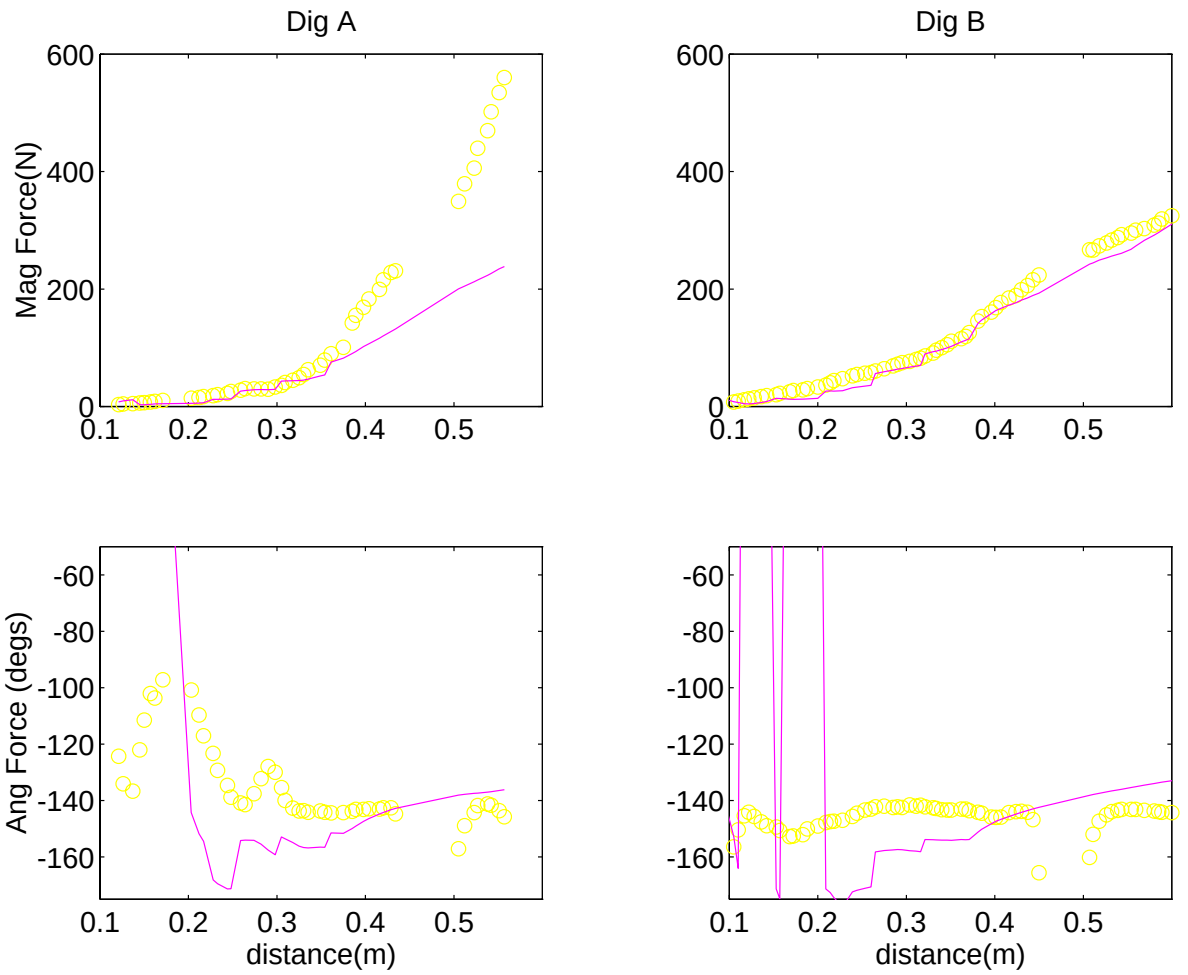


Figure 77 Prediction using global regression with mechanics based basis functions.

5.2.4.3 Local Weighted Regression

Moving from global to local models further improves prediction. Figure 78 shows force data that are predicted using the default local weighted regression scheme in which a kernel width of $K_w = 2^{-6}$ is used and all input parameters are weighted equally.

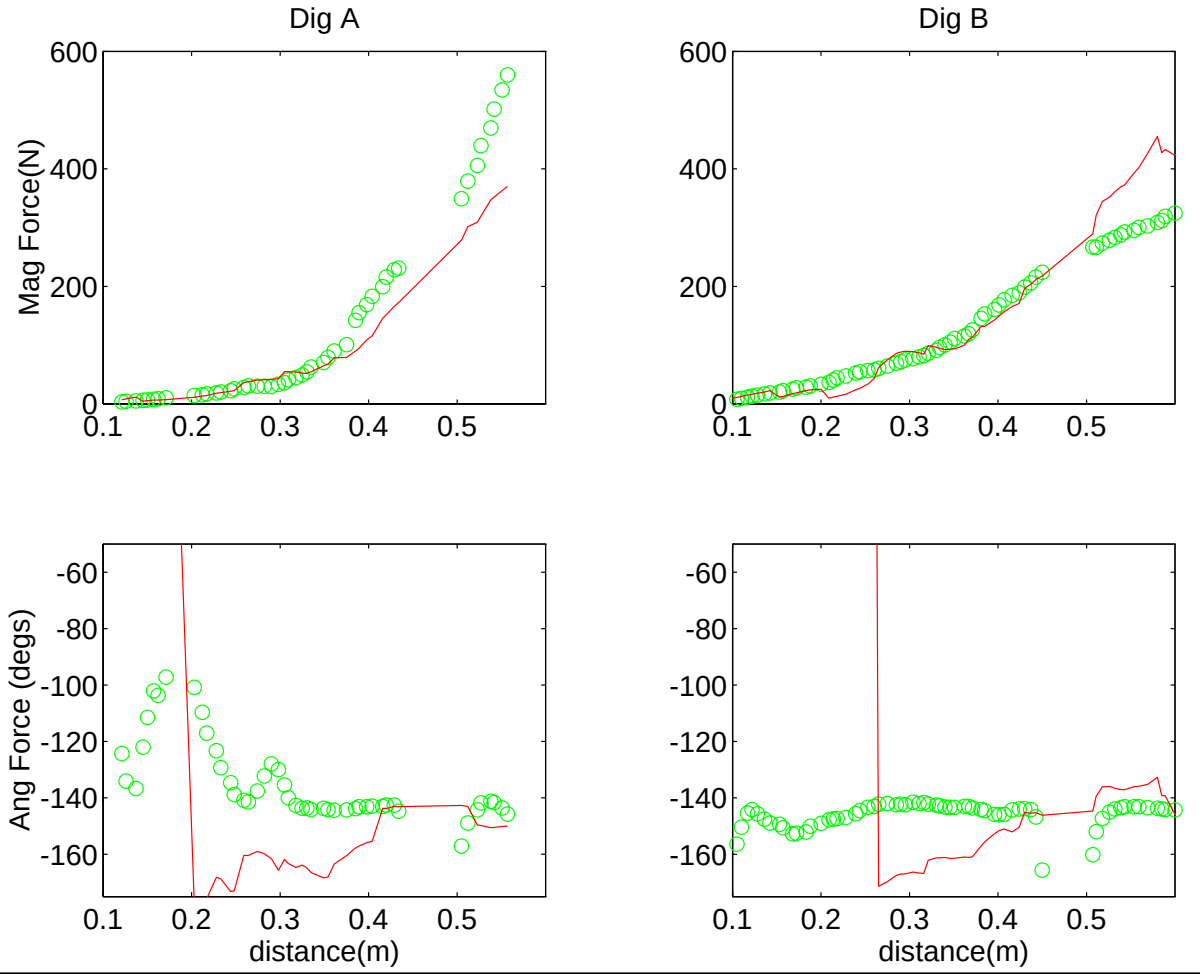


Figure 78 Prediction using default local weighted regression with mechanics based basis functions.

The performance of LWR can be further improved if we have some intuition about the relative importance of the input parameters. MARS finds this information through a statistical analysis of our training set (Table 3).

Table 3 Relative importance of variables found by MARS

	ρ	d_1	d_2	l_1^2	l_2^2	l_3^2	l_4^2	l_5^2	c_1^2	c_2^2	c_3^2	v
x	67	31	100	66	19	18	22	35	14	15	11	71
z	100	81	77	32	80	49	59	15	0	0	0	83

Figure 79 shows prediction of the resistive forces using the same kernel width as in Figure 78 but with input parameters scaled by the relative importance of the input variables. Although the results using "scaled" locally weighted regression for our examples are promising, our results to date indicate that on average, this extension provides small benefit over equal weighting. It is possible that this is due to the fact that this scaling tries to capture all the data at once and ends up not representing any part of the function very accurately.

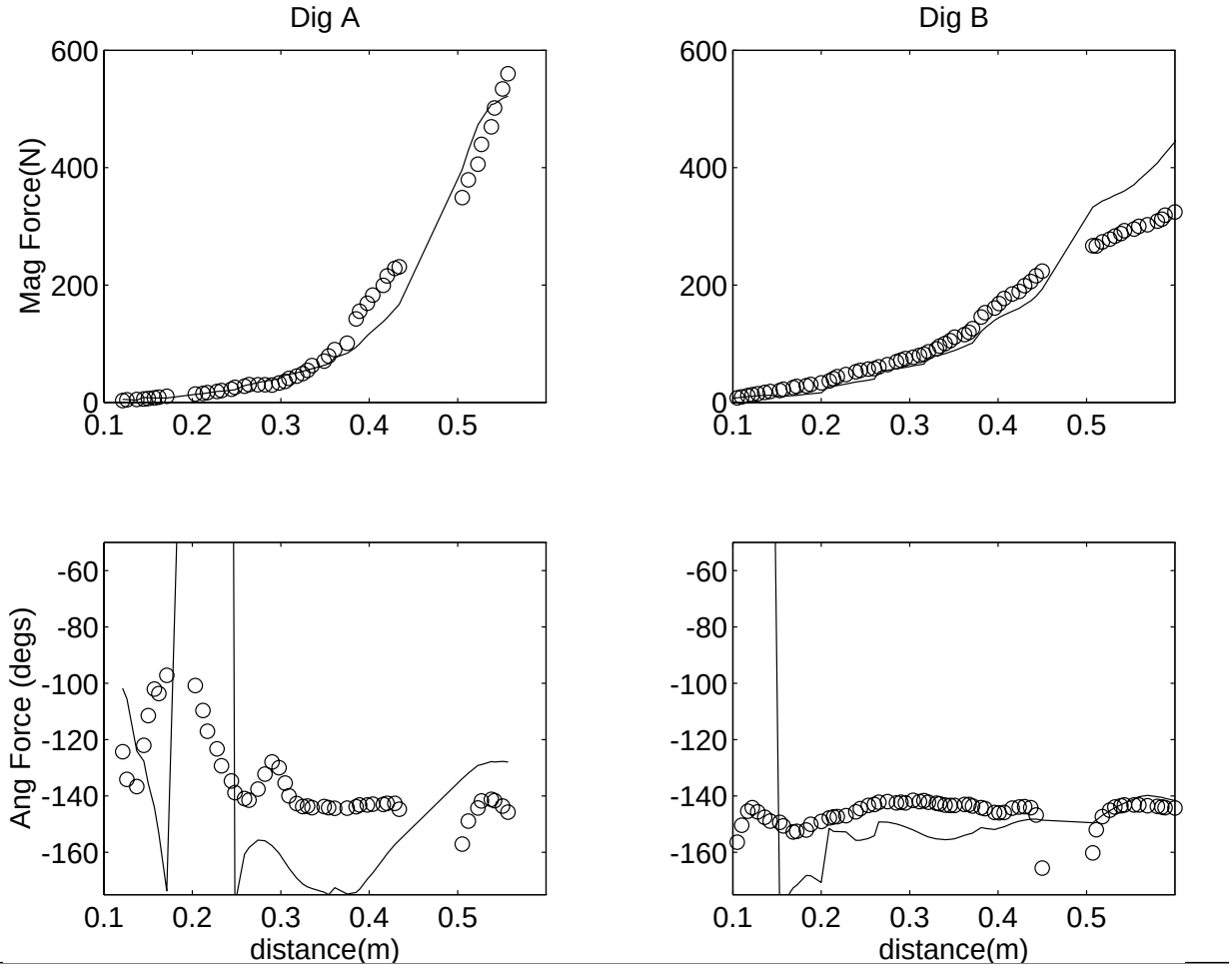


Figure 79 Prediction using LWR with mechanics based basis functions. Input parameters are scaled based on MARS analysis.

5.2.4.4 Neural Networks

For our training set of force data, trial and error experimentation shows that 30 hidden units provide the best prediction over many trials. Training the neural net involved showing the inputs and outputs to the network 10,000 times. Figure 80 shows prediction for the two exemplar cases we have been following.

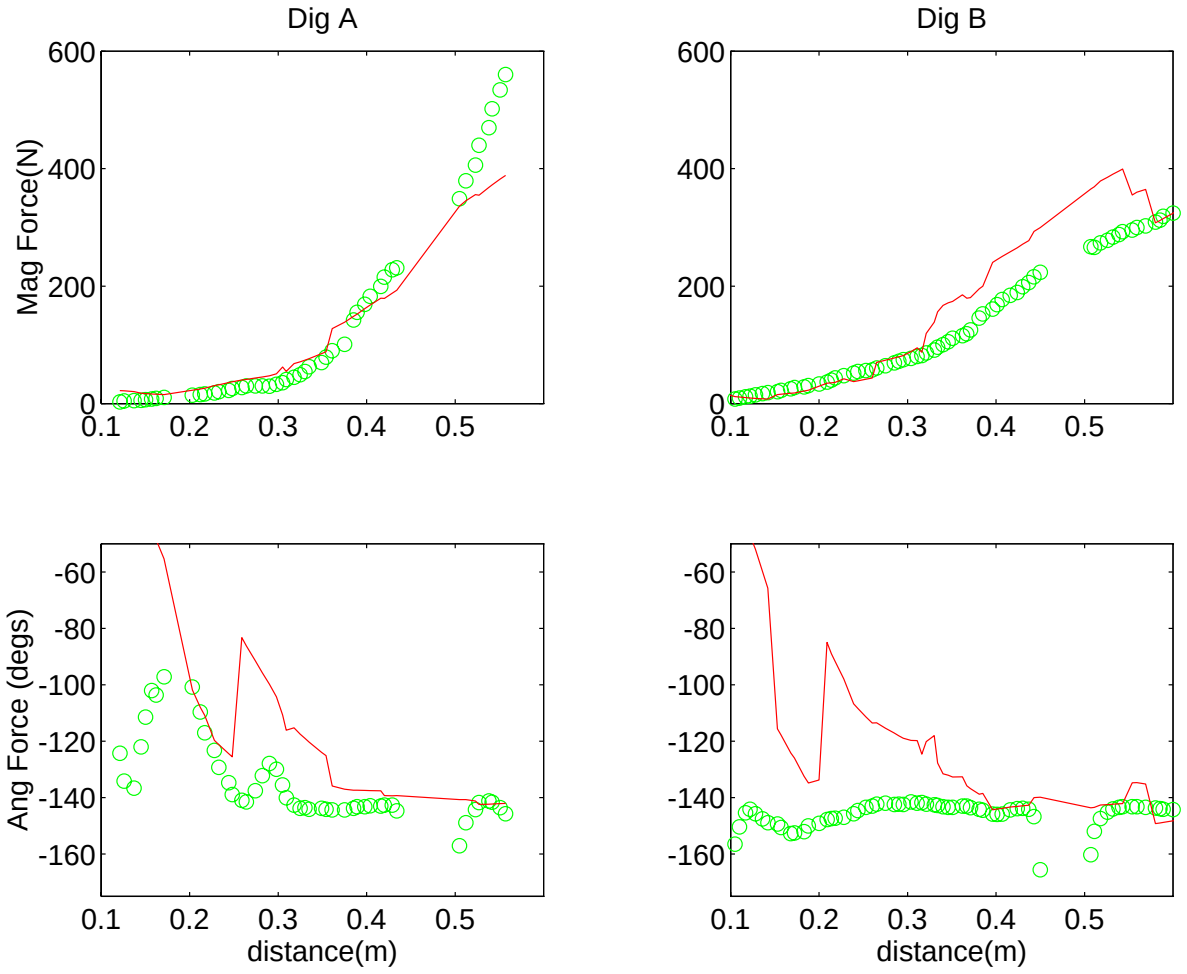


Figure 80 Prediction using a back propagation neural network (30 hidden units) with mechanics based basis functions.

So far we have qualitatively compared the various methods with the mechanics based formulation of the basis functions. Below, we present quantitative results to aid in comparison of the method and the various representations.

5.2.4.5 Criteria

The first criteria to compare methods and representation is *accuracy*. We have conducted a systematic comparison of the average accuracy that results from the various representations and methods discussed above. Each data set was analyzed using 10-fold cross-validation. The data set was divided into ten blocks of near-equal size. For every block in turn, each method was trained on the remaining blocks and tested on the hold-out block. Results, averaged over all test blocks, thus reflect the predictive performance on unseen cases. The mean absolute error is used as the measure of performance (Table 5).

Table 4 Mean Absolute Error (N) between predicted and observed magnitude of resistive force. The results are based on a 10 fold cross-validation with the exception of the results for back propagation which was trained on a block of 3000 points and tested with the remaining 1000 points.

	Naive Basis Functions Eqn (15)	**Mechanics Based Basis Functions Eqn (16)**	**Reduced Mechanics Based Basis Functions Eqn (17)**
Global Regression	81.64	69.52	70.22
3 Nearest Neighbor	79.27	72.37	72.44
LWR (default)	65.83	65.37	64.15
LWR (MARS scaling)	63.65	62.92	68.62
Back Propagation (30 HU)	83.18	64.67	66.91

This comparison confirms a couple of points. First, the mechanics based basis functions provide significantly better accuracy for 3 out of the 5 cases. Improvement is smallest in the case of LWR and in this case it appears that the local nature of the method is able to represent the underlying function well. Second, removing those variables which were detected to have a small influence on the output values does not degrade performance significantly and in one case even improves the error metric. Across methods, apart from discounting global regression and nearest neighbor, it is hard to make a case for particular method based on accuracy.

Precise prediction of resistive force is much more important for forces of large magnitude since these are more likely to be require torques close to the robot's limits, than small forces. Another way to compare the learning methods is to compare the errors between predicted and measured forces above a threshold (Figure 81).

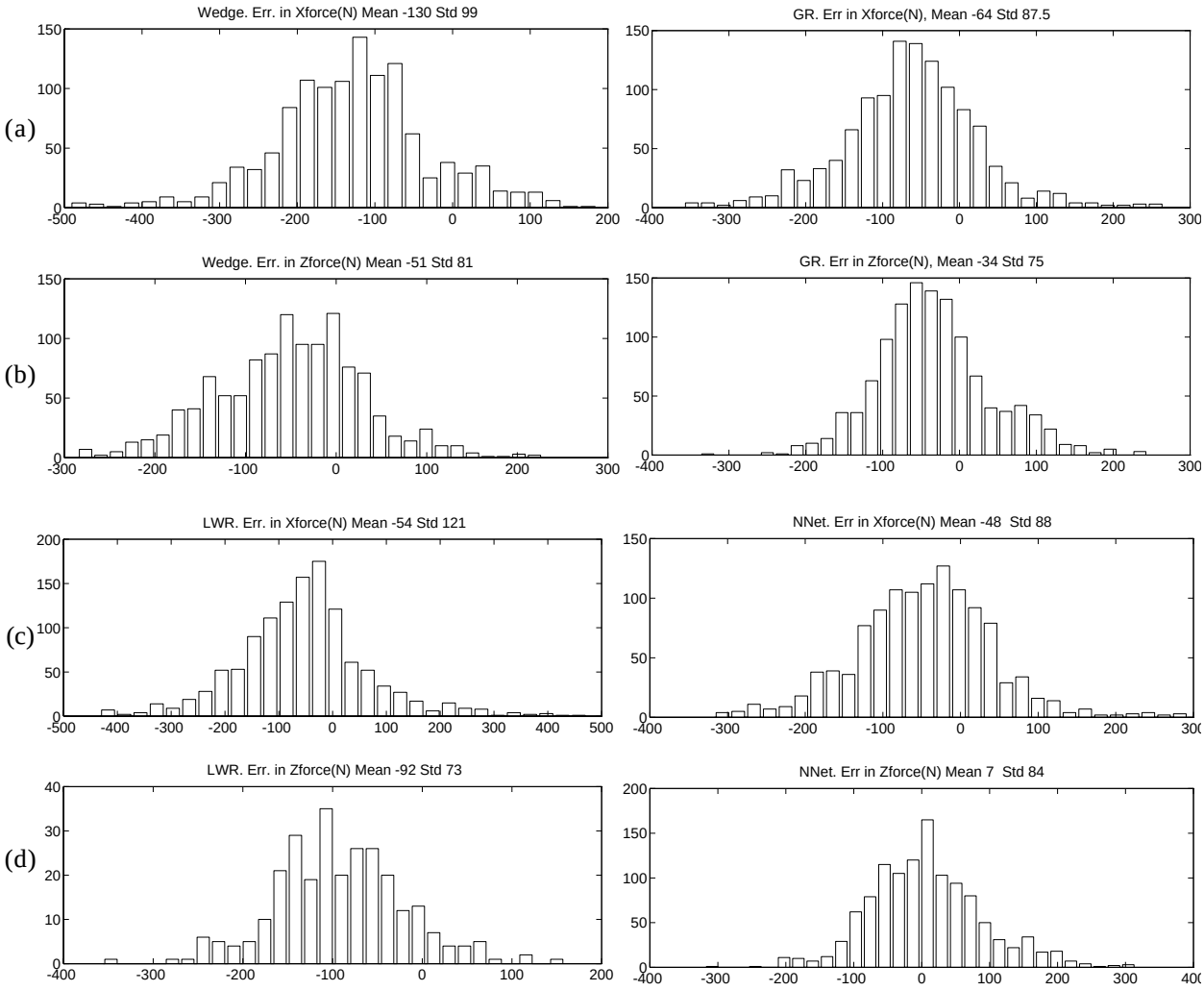


Figure 81 Histogram of errors in prediction of resistive force using only forces with observed magnitude greater than 300 N in either the x or z components of the resistive force. using (a) wedge model (b) global regression, (c) local weighted regression, (d) neural nets. Negative errors mean that the resistive force was underestimated.

Next we compare methods based on training time, prediction time, ease of incorporating new data and memory requirements during prediction.

Atkeson has pointed out that if there is a continuously growing training set intermixed with queries then memory-based learning is appropriate [Atkeson91]. Local methods are able to reduce interference between data both spatially (in the input space) and temporally, and generalization depends on the density of the samples. For the excavation domain this means that if very little is known about the soil-tool interaction in a given environment, and we would like to rapidly improve prediction, then memory based methods will be useful. However, if a fixed training set is available before queries are made, then an approach like neural nets will be preferable.

Table 5 Other criteria to compare learning methods. *D* is the number of basis functions, *n* is the size of the training set, and *h* is the number of hidden units.

	training time	**prediction time**	**ease of incorporating new data**	**memory requirements**
Wedge Model	days (using brute force search)	~ 10 ms	complete retraining/ search required	4 constants
Global Regression	~10 secs	~1ms	iterative scheme possible	2 sets of *D* constants
Memory Based Learning	~1 hr	~100ms	easy	all data must be stored: *nD*
Neural networks	~10 hrs	~1ms	some retraining required	2 sets of *h* constants

5.3 The effect of resistive forces

A robot's strength limitations come from the fact that its joints can only produce finite torques.This section discusses how it is possible to use the robot's configuration and end-effector forces to determine the necessary joint torques. It should be noted that the problem at hand is planar and for most excavating machines will involve only three joints— shoulder (*boom*), elbow (*stick*), and wrist (*bucket*) joints.

5.3.1 Calculation of the resulting joint torques

Given a hypothetical trajectory that the robot is to follow, the resistive forces that the end-effector is likely to experience are predicted as in 5.2.3. Next, these forces are converted into joint torques. A trajectory is feasible only if all predicted torques expected fall below the torque limits. A static analysis is used— that is, we only consider the torques due to gravitational forces affecting the robot's limbs and those due to the resistive forces, leaving out torques due second order effects such as coriolis forces.

The total torques required at the joints to accomplish a digging trajectory can be written as:

$$\underline{\tau} = \underline{\tau}_{tip} + \underline{\tau}_{grav} \quad (28)$$

$\underline{\tau}_{tip}$ is a function of both the forces experienced at the bucket as well as the configuration of the robot. On the other hand, $\underline{\tau}_{grav}$ is only a function of the geometric configuration of the robot. The torques due to tip forces are:

$$\underline{\tau}_{tip} = \underline{J}_m^{\mathrm{T}} \cdot {}^{0}\underline{F}_0 \quad (29)$$

where J_m^{T} is the transpose of the *manipulator jacobian* and ${}^{0}F_0$ is a vector of the end-effector forces expressed in the base coordinate frame. Since forces are measured at the end-effector, they will need to be transformed into the base coordinate frame. This is done through the use another jacobian known as the *frame jacobian* (J_f). While the manipulator jacobian transforms end-effector forces into torques (and vice versa), the frame jacobian transforms forces from one coordinate frame into another:

$${}^{0}\underline{F}_0 = \underline{J}_f^{\mathrm{T}} \cdot {}^{E}\underline{F}_E \quad (30)$$

where ${}^{E}\underline{F}_E$ are end-effector forces.

Appendix 2 shows a complete derivation of the torques due to gravitational forces acting on the limbs of the robot. The results are summarized below. The torque felt at joint *i* in the planar assembly of an excavating robot (*i* = 1 is the shoulder joint) is:

$$\tau_{grav_i} = g \sum_{n=i}^{m} l_{i,n} m_n \cos\psi_{i,n} \qquad (31)$$

where $l_{i,n}$ is the distance from joint *i* to the center of mass of link *n*, m_n is the mass of link *n*, and $\psi_{i,n}$ is the angle between the horizontal and the line connecting joint *i* to the center of mass of link *n*.

5.3.2 An illustrated example

To illustrate the calculation of the joint torques, consider two similar paths taken by an excavator bucket while it moves through soil. Figure 82 shows two trajectories (A & B) followed by the tip of a bucket mounted on the T3. Force data was collected during the trajectory A and was used to calculate the torques on the joints The same force data along with the trajectory B was again used to calculate the torques on the joints.

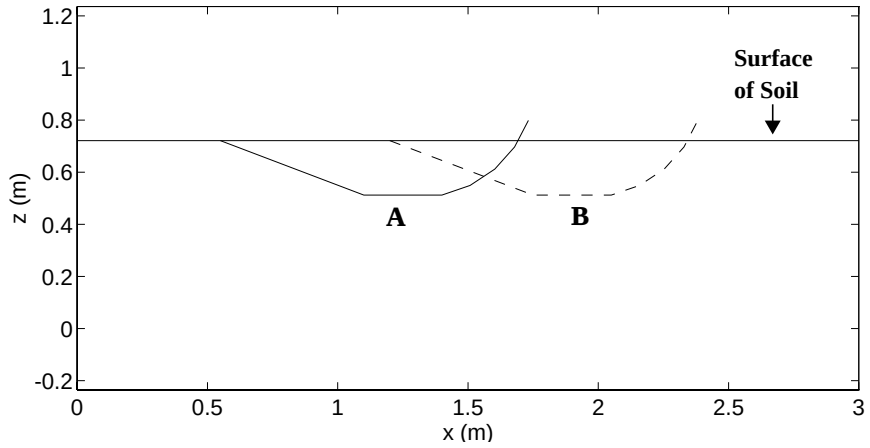


Figure 82 Two similar trajectories followed by an excavator bucket.

Figure 83 shows the torques due to tip forces during travel along trajectories A & B. On the other hand, if we look at the torques due to gravitational forces on the limbs of the robot, there is a larger difference (Figure 84). The torques on the elbow joint are almost the same because they are simply a result of the moment arm from the wrist joint to the center of mass (m_4).

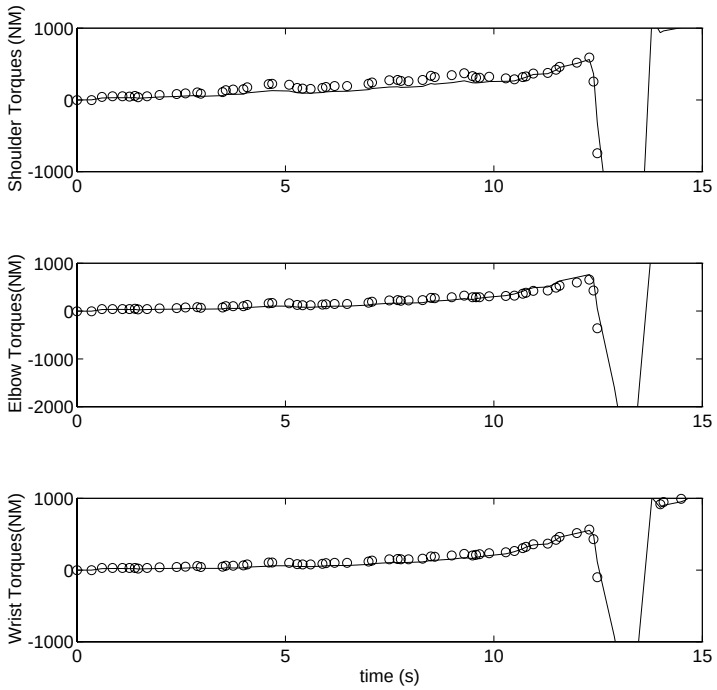


Figure 83 Torques due to tip forces while the robot follows A (solid lines) & B (circles). There is very little difference in the torques because the two Cartesian trajectories are not very different and the resistive force is assumed to be the same in each case.

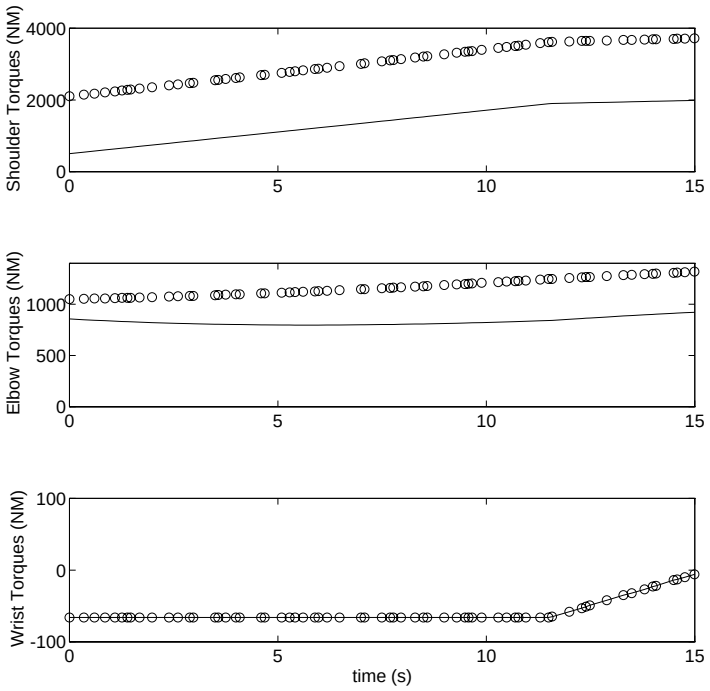


Figure 84 Torques due to gravitational forces while the robot follows A (solid lines) & B (circles).

5.3.3 The Force Constraint in an Action Space

Figure 85 shows the set of plans that satisfy the force constraint for the hydraulic manipulator given the soil in our testbed. Note that this set looks much like the set that satisfies the reachability constraint. This is because the set of force-feasible plans is necessarily a subset of the set of plans that meet the reachability constraint— it does not make sense to consider torques for trajectories that can not be reached by a manipulator.

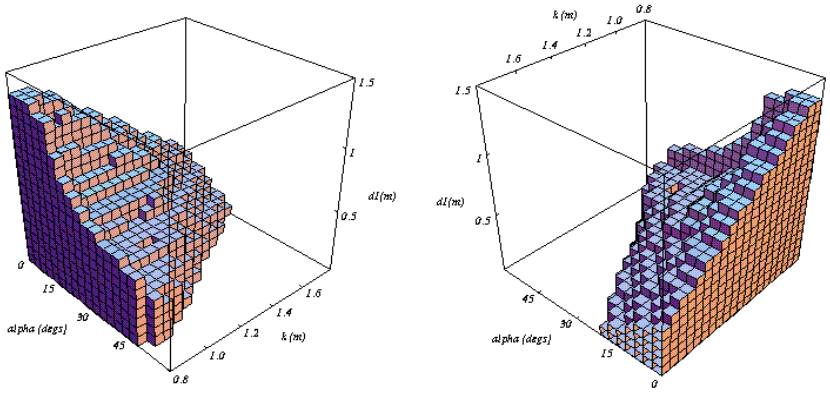


Figure 85 Force Constraint in the action space

The force constraint is much more clear in the (α, k, d_1) space. It is often the case that the force constraint limits the d_2 variable that is not represented in Figure 85. To compare the effect of adding the force constraint, Figure 86 shows the set of plans that meet the geometric constraints and those that meet the force constraint in the (α, d_1, d_2) space for a fixed value of k.

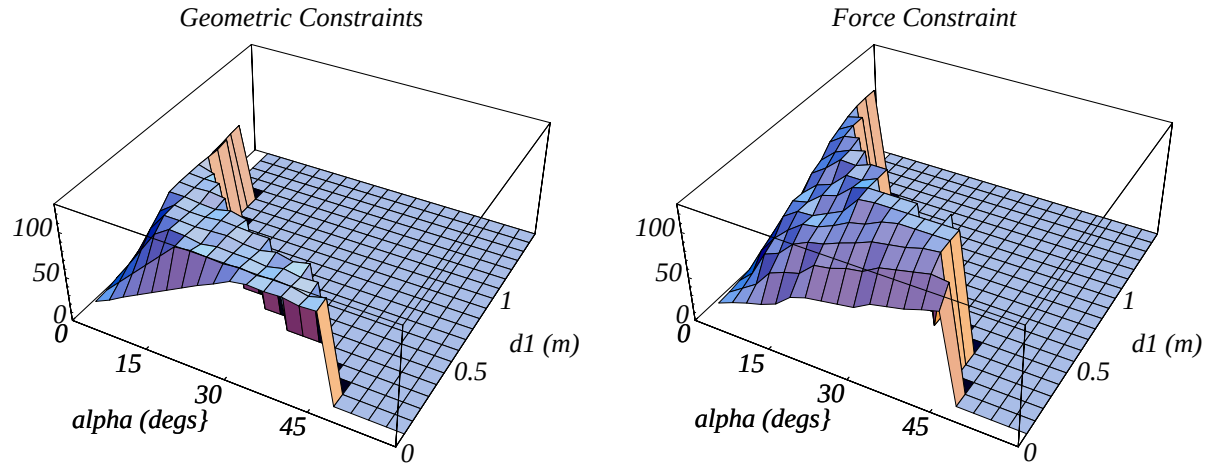


Figure 86 Comparison between the set of plans that satisfy geometric constraints and those that satisfy force constraints for one *k* value. Note that in some cases the force constraint is more limiting and in other cases the geometric constraints are more limiting.

The most convincing argument for including the force constraint is shown Figure 87. This figure shows the areas in the action space where the geometric constraints dominate and those in which the force constraint dominate. In this case, a significant area of the feasible action space is limited by force. The shape of the latter area is intuitively understandable— actions with steep approach angles or long digging trajectories (along d_1) are force limited.:

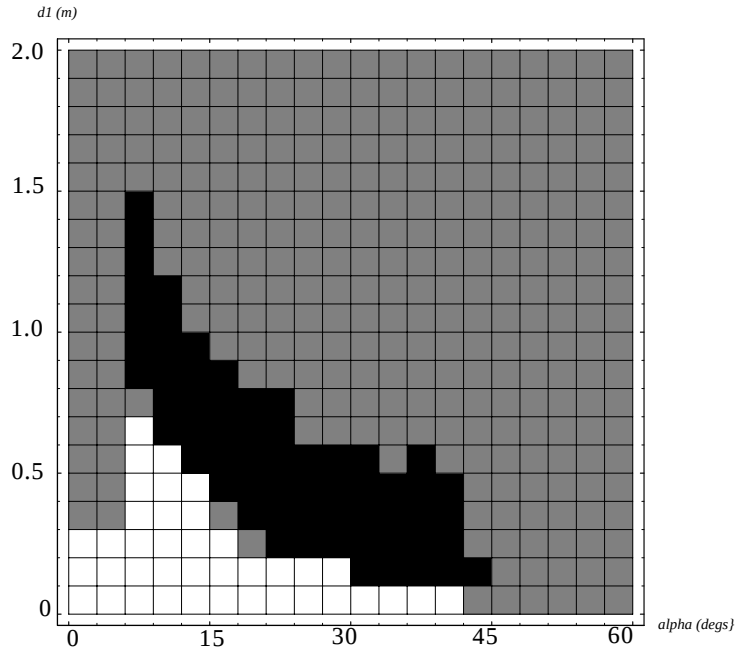


Figure 87 An illustration of the areas in the action space from Figure 86 where the geometric constraints and force constraints are the limiting factors. The white cells represent areas where the geometric constraints dominate while the black cells are areas where the force constraint dominates.

The predictive force model also makes it possible to use various force related optimization criteria. For example Figure 88 shows different digging plans that are selected under two criteria. In one case, the criteria is to minimize the torques experienced at the joints and the best dig selected is a long dig with a shallow approach angle. In the other case, the criteria is to minimize the integral of the torques experiences at the joint and again the result agrees with the intuition— either a very dig or a very long dig is chosen. Note that both actions sweep the same volume of soil and are equivalent on a geometric basis.

5.4 Making the execution robust

So far the emphasis has been on developing a method that estimates the force constraint as well as possible. This is motivated by the philosophy that the more reasonable a plan, the more likely a robot is to be able to accomplish it. In addition to all the work that is done

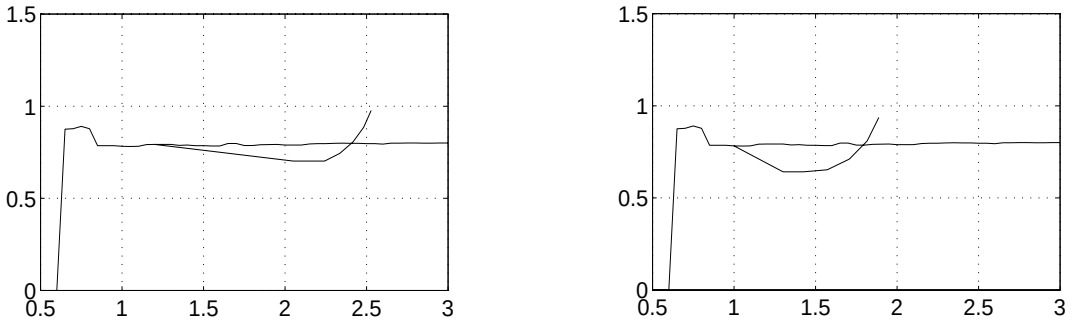


Figure 88 Digs selected based on different criteria (a) lowest torque experienced (b) lowest cumulative torque experienced.

before the robot starts moving, it is desirable to make the execution more robust by having the robot react to the world as it moves. This area lies outside the scope of the experimental work done for this thesis.

Ideally, we would like the robot to exhibit a specified “stiffness” to account for the fact that the resistance forces encountered by the robot might be higher than the robot is capable of surmounting. Resistance forces may be higher than expected due to errors in modeling or even due to a rigid intrusion like a buried rock. Several researchers have proposed methods to deal with such issues although the attention has been directed towards contact with extremely stiff materials [Hogan85, Lawrence87].

The main ideas of stiffness control can be stated as follows. The force experienced at the end effector, *F*, of a robot is reflected in the joints by the relationship.

$$\tau = J^{T} F \tag{32}$$

where τ is the generalized torque and J is the manipulator jacobian. A simple stiffness relationship can be stated as:

$$F = Ke \tag{33}$$

where e is difference between the desired and measured positions of the end effector and K is a diagonal matrix containing stiffness gains for each degree of freedom at the end effector. Now, the effect of the cutting resistance in terms of torques on the joints can be stated as:

$$\tau = J^{T} Ke \tag{34}$$

If a robot is position controlled (i.e. takes only position and velocity commands), it is necessary to develop an alternate formulation. Let δF represent the difference between the expected force and the measured force during a dig. This can be related to a differential in torque at the joints by the following:

$$\delta\tau = J^{\mathrm{T}}\delta F \tag{35}$$

The torque required at the next control instant can be stated as:

$$\tau_{i+1} = \tau_i + \delta\tau \tag{36}$$

Using another form of (34), $\tau = J^{\mathrm{T}}K\delta x$, we can write the full control law as

$$\delta x_{i+1} = (J^{\mathrm{T}}K)^{-1}\ \tau_{i+1} \tag{37}$$

This specifies a differential modification to the specified trajectory of the end-effector in cartesian space. (A similar control law can be formulated for joint space.) That is, at every control cycle, a modified end-effector position can be specified based on the resistance encountered.

While the method is straightforward, the challenge is anticipated to be the specification of the stiffness matrix *K*. Stiffness of the end-effector must be matched to the expected stiffness of the soil, given the maximal torque that the robot can exert. The main difficulty is that stiffness of the soil is not isotropic, that is it varies based on the direction that the force is applied and the configuration of the soil.

Chapter 6 Constrained Optimization

Chapters 4 and 5 have discussed constraints that can be imposed on action spaces for excavation, in some detail. These chapters have shown examples of the sets of plans that satisfy various constraints, that is, those actions that are “feasible”. There has been no mention so far as to how these sets are constructed and how the single action to be executed is chosen. Simply stated, the problem at hand is one of constrained optimization— we are to find a subset in the action space that satisfies all constraints as well as optimizes some performance measure.

Constrained optimization is a well studied problem; indeed a whole discipline is devoted to this topic. This thesis does not propose new methods of constrained optimization, but rather proposes a formulation of the problem using well known methods. To start, this chapter attempts to pose the excavation problem in the language of standard optimal control methods. Action space planning is described as a particular optimal control problem. Next, discussion turns to the method of optimization that was used for the purposes of the thesis. The chapter concludes with a look at other methods of constrained optimization that are more efficient.

6.1 A Traditional Formulation of Optimal Control

Consider a formulation of the excavation problem in traditional terms. Typically, we require:

- *A mathematical model of the process to be controlled*. This model typically consists of a description of states, control variables, and a plant. The model describes the evolution (in terms of resulting states) of a system for admissible inputs.
- *A statement of constraints*. A system is subject to two kinds of constraints. One type limits the states of the plant (for example the initial and goal states). The other type limits the control inputs either because they are physically impossible or because they clearly violate some necessary condition.
- *A specification of a performance criterion*. A performance criterion is a scalar metric which differentiates between the feasible actions.

6.1.1 The Process Model

Typically, the state of a system at time t, is described by $\underline{x}(t) = (x_1, \ldots, x_n)^{\mathrm{T}}$, the control inputs by $\underline{u}(t) = (u_1, \ldots, u_n)^{\mathrm{T}}$, and, the plant is described by a set of differential equations such that:

$$\dot{\underline{x}}(t) = \underline{a}(\underline{x}(t), \underline{u}(t), t) \tag{38}$$

The objective of modeling is to obtain a compact description that adequately predicts the response of a physical system to anticipated inputs. For the moment, let us presume that such modeling is possible and tractable. In the excavation domain, the "system" includes both the robot, R, and the terrain, W, that it must manipulate. A minimal representation is to use only position variables— the terrain can be represented by a list of elevations[1] E_i, and the robot by joint variables Θ_i:

$$\underline{x} = \left[E_1, E_2, \ldots, E_a, \Theta_1, \Theta_1, \ldots, \Theta_b\right]^{\mathrm{T}} \tag{39}$$

At a fundamental level, the robot's ability to affect the world is due to the torques it can generate at its joints. Hence one formulation of the control is given by

$$\underline{u}(t) = \tau_i(t) \tag{40}$$

Under this formulation $a_1(\ ), \ldots, a_n(\ )$ are the differential equations that relate $\underline{u}(t)$ to $\underline{x}(t)$, that is, they describe how robot torques affect the state of the terrain. Alternately, it

1. the ground-plane is tessellated into a grid and each grid point is assigned an elevation.

might be possible to abstract the inputs into trajectories and contact forces of an excavating tool.

6.1.2 Constraints

Presume that to start, the world is in some initial state $\underline{x}_o$, and must be transformed to some final state $\underline{x}_f$. This provides the boundary constraints:

$$\begin{aligned} \underline{x}(t_o) &= \underline{x}_o \\ \|\underline{x}(t_f) - \underline{x}_f\| &< \delta \end{aligned} \tag{41}$$

where δ is a small scalar value indicating the permissible deviation from the goal state.

The other type of constraints limit the control inputs. For example, a robot has a finite workspace and can generate only finite torques. These *m* constraints are written as:

$$g_i(\underline{u}(t))) < 0 \qquad i = 1 \ldots m \tag{42}$$

6.1.3 The Performance Criterion

An optimal control is defined as one that minimizes (or maximizes) the performance criterion. The form of the performance criterion is chosen by the designer to reflect optimality. In some cases, the performance criterion suggests itself. For example, if the control effort over an entire trajectory is to be minimized, a criterion that can be used is:

$$J = \int \|\underline{u}(t)\| dt \tag{43}$$

In the case of excavation, the cost criterion is often not simple. It is not only the case that optimality is defined by more than one criterion, but it is insufficient to stipulate a criteria that is a simple addition of separate functions. For example, we might require that a robot excavator choose from the set of feasible actions, one that excavates a minimal amount of soil and that minimizes the torques on the joints. These objectives can be mutually contradictory. For example, a digging action that barely enters the soil requires very low joint torques but doesn't excavate much soil. Instead of a compromise between the two criteria, we would like to distinguish between necessary and auxiliary criteria. For example, a combined criterion for a single action might be specified as follows:

$$J = \begin{pmatrix} (J_1(\underline{u}) \le z) : 0 \\ (J_1(\underline{u}) > z) : J_2(\underline{u}) \end{pmatrix} \tag{44}$$

where J_1 is a necessary criterion that must provide utility z at minimum, and J_2 is an auxiliary criterion. Here the performance measure is to be maximized.

6.1.4 Formulating the optimization

The ideal method would find an optimal trajectory in the state-time space that transforms the terrain into the desired state. Following the standard method, it is necessary to find the global extremum of the performance measure subject to the state and input constraints. Typically, one of two methods are used to find the extremum— the minimum principle of Pontryagin [Pontryagin62] or the method of dynamic programming [Bellman62]. Pontryagin's method leads to a nonlinear two-point boundary value problem that must be solved to obtain the optimal control. Dynamic programming can be solved by an efficient search method if the system is approximated by a discrete model (difference equations and discrete states and inputs) or by solving the Hamilton-Jacobi-Bellman equation, a non-linear partial differential equation. Unfortunately, it will not be possible to pose the optimization problem using any of these methods. There are two main reasons:

- *The state space is too large*. As pointed out in Chapter 3, the state vector, $\underline{x}(t)$ is very large. It is intractable to seek global optimality because it will require a prohibitive amount of computation. In the case of dynamic programming consider the number of steps that are required. Assume that each of the a variables that describe terrain elevation is quantized into r levels, each of the b joint extensions of the robot is discretized into s levels, each of the c control inputs is discretized into t levels, and L actions are required to complete the task. The number of calculations required by dynamic programming are:

$$\left[r^a s^b t^c\right] L \tag{45}$$

  The term r^j itself can be very large. Assuming a 10 cm quantization the number of possible states of the terrain in our testbed is 10^{900}.

- *The differential equations that model soil-tool interaction are not in closed form*. No simple differential equations exist that describe the state of soil in response to forces. The computational models that are available require iterative solution to large sets of partial differential equations, a form that is not convenient to use with the methods mentioned above.

These two reasons are in fact a reiteration of the arguments for action space planning. There is another complication in formulation excavation planning as an optimal control problem. The solution to the problem at hand requires sequences of separate trajectories to perform the entire excavation. Most optimal control problems are phrased such that the solution is a continuous trajectory or a sequence of discrete operations.

Next we examine how the optimization is performed in the action space. If instead of finding a complete trajectory up to the goal, attention is restricted to a smaller problem, namely optimization over the vector space of feasible one-step control inputs (plans), the problem becomes much more tractable.

Formally, at step k, the performance measure $J(\underline{u}(k))$ is optimized over the space of $\underline{u}(k)$ subject to the constraints $g_i(\underline{u}(k)) < 0$. If the space of control parameters is chosen to be an abstraction of a prototypical action in which $\underline{u}(k)$ describes an entire one-step plan, then the space of control parameters is exactly an action space. Now, instead of a sequence of $\underline{u}$, a single point in the space of control parameters is being sought. The fundamental difference in using this formulation is that it is not necessary to use inverted analytical models of the dynamical system and the constraints to determine the optimal point in the action space. As long as it is possible to reasonably approximate a performance measure, it is possible to directly operate on the points in the action space, using functions $J(\ )$ and $g_i(\ )$.

6.2 Using Exhaustive Search

A simple method to perform this optimization is exhaustive search—all the feasible plans are enumerated and sorted by their utility. Since the optimum is a point in $\Re^c$, it is necessary to first discretize the space. This allows for a grid based approach to the constructing the sets of feasible plans and to performing the search. That is, each cell in the $\Re^c$ can be marked as feasible (if it passes all constraints) or infeasible (if it fails any constraints). Similarly, a performance value can be associated with each cell.

Exhaustive search can be computationally intensive (see complexity analysis below) in the worst case, but it is possible to improve the efficiency of search in most cases by pruning the search space and reducing computations when a cell is evaluated.

6.2.1 Pruning the search space

If it can be safely assumed that the shape of the feasible set is such that the range of feasible plans along a dimension of the search space is connected (no gaps in the feasible set along a dimension) then, it is possible to prune the search space with a simple heuristic. Figure 89 shows illustrates this idea with an example of search in a two dimensional space. Assume that search progresses is a raster fashion, parallel to the u_2 axis. For any value of u_1, search can terminated as soon there is transition from feasible to the infeasible set (Figure 89 (a)). Note that for the same example, it is not possible to prune the space if search progresses parallel to the u_1 axis. Like wise, no pruning is possible in the case shown in Figure 89 (b).

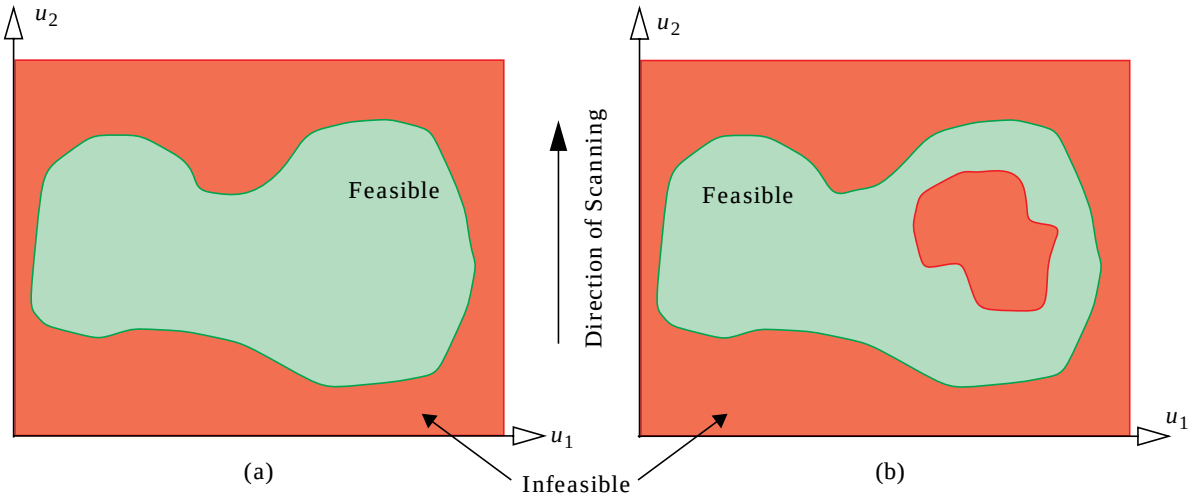


Figure 89 Ability to prune the search space based on the shapes of the feasible and infeasible sets. (a) If the feasible set is solid along the direction of scanning can be terminated when a transition is found from the feasible to infeasible sets (b) Pruning is not possible.

As an example from the excavation domain, consider the action space for trenching (α, k, d_1) and the shaping constraint shown in Figure 90. The shaping constraint provides a con-

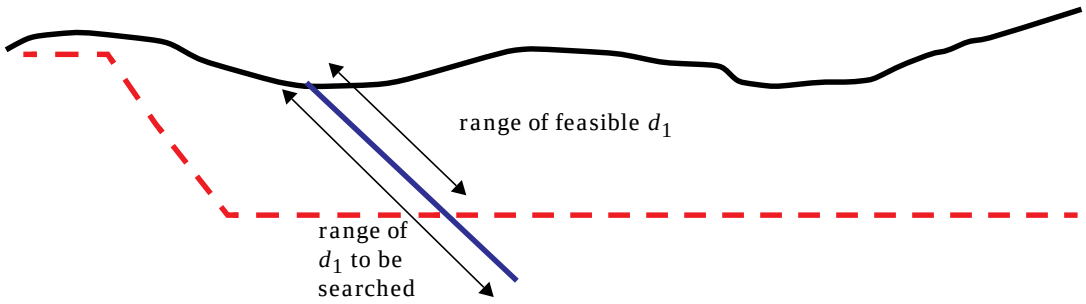


Figure 90 An example in the trenching task pruning is possible. Presuming that search is conducted along d_1, it is possible to terminate search after the shaping constraint is violated for the particular (k, α) pair.

straint along the dimension d_1, such that as soon as an infeasible plan is found, it is possible to disregard all greater values of d_1.

6.2.2 Limiting computation at each cell

The above sub-section showed how it is possible to prune the search space such that possibly large sections can be left unexamined. It is also possible to reduce the computation necessary when a cell in the search space is examined by ordering evaluation of the constraints intelligently. If it were possible to determine an ordering from the least to the most restricting constraint, then it is always best to evaluate the constraints in order of decreasing

restriction. Unfortunately, it is not possible to make this kind of evaluation a priori since this kind of ordering can change during execution. For example, an excavation might start with the force constraint being the most restrictive, but as the excavation progresses, the bucket capacity might become the most limiting. In practice, the evaluation of constraints is ordered by increasing complexity of computation. For example, the reachability constraint has been reduced to a fast table lookup and it is the first constraint to be evaluated. At the other extreme, the force constraint might be expensive to compute (based on the prediction method used) because it involves prediction of forces along a trajectory and then translation into joint torques and should be evaluated last.

6.2.3 The performance measure

To distinguish feasibility from utility, recall that feasible plans are those that are physically possible and cause trajectories that are admissable. Utility varies in this set, based on the performance criteria. As noted above, it is typically not sufficient to optimize a single performance measure such as amount of soil excavated or control effort. An easy method is to do a “dictionary” search. First, the set of feasible plans is reduced to those that satisfy a bound on a primary performance criteria. This set is then iteratively reduced to those plans that also satisfy bounds on auxiliary criteria. It is also possible to perform this optimization in one pass by stipulating the performance criteria as in (44).

For the experiments conducted for this thesis, the swept volume has been treated as a primary performance measure. Minimum torque and minimal cumulative torque were used as auxiliary measures.

6.2.4 Complexity of Search

In the worst case it is not possible to prune the search space u_i, $(i = 1 \ldots c)$ and the constraints $g_i(\underline{u}) < 0$, $(i = 1 \ldots m)$ must be evaluated at each step. Assume that the number of quantizations of each dimension is given by the function $N()$, and that the number of computations required by each constraint is given by the function $P()$. Then, in the worst case the number of calculations required is:

$$\prod_{j}^{c} N(u_j) \sum_{i}^{m} P(g_i) \qquad (46)$$

In other words, exhaustive search is exponential in the number of action space variables and linear in the number of cells to be searched along any one dimension.

Worst case analysis notwithstanding, consider the timing results from excavation experiments. Figure 91 compares computation time taken for three sequences of excavations. The initial conditions for each of the experiments are identical but the optimization criteria are varied. These timing results make two points. First, as the excavation proceeds, the set of feasible plans shrink and the pruning heuristics can make a large difference in computation required. Second, the more limiting the constraint, the less computation that is required. For example, when the torque limit on joint 4 is reduced from 1000Nm to 500Nm, the computation time required is reduced also because the feasible set is smaller.

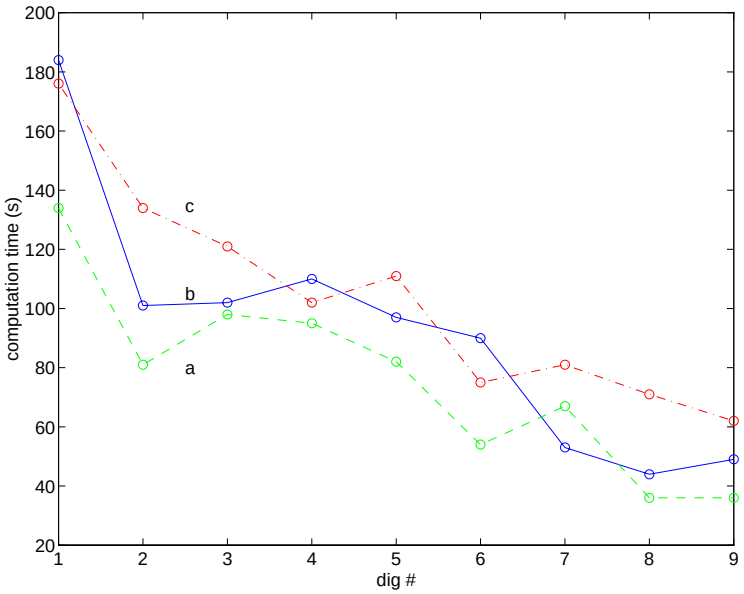


Figure 91 Examples of how efficient search can reduce computation time (execution times on Sparc 10 workstation). Shaping, Reachability and Force constraints were used. (a) Torque limit at joint 4 = 500 Nm, maximized volume and minimized greatest torque (c) Torque limit at joint 4 = 1000 Nm, maximized volume and minimized greatest torque (b) Torque limit at joint 4 = 1000 Nm, maximized volume only.

6.3 Other methods of constrained optimization

Several other numerical methods exist in the literature on constrained optimization. Unfortunately, many of these methods (*gradient projection* [Luenberger80], *penalty* and *barrier* methods [Luenberger80], and *tunneling* [Chua80]) make requirements on the constraints that can be imposed on the vector space. For example, all three require that the constraint surfaces are continuously differentiable. In general, very few methods are able to perform global optimization. Further, most methods are local— success in finding the global optimum is strongly dependent on making a good initial guess.

A method that holds some promise for optimization given the nature of the optimization task, is called *simulated annealing* [Bohachevsky86]. Simulated annealing can be thought of

as a stochastic gradient descent method. A random element whose standard deviation decreases exponentially with time is added to a standard gradient descent method. The main advantages of simulated annealing are that it is able to escape local extrema in the objective function and can deal with an arbitrary number of constraints that need not be smoothly differentiable, nor in closed-form. It is sufficient to represent each constraint with a boolean function that indicates whether a point in the vector space passes or fails the constraint. Since such a method is only guaranteed to asymptotically approach the global optimum, in practice, the method must be modified to be practical. See [Singh91] for implementation details of a practical simulated annealing method for constrained optimization.

6.4 Constrained Optimization— Summary

This chapter has shown that analytical closed-form solutions are not possible for the task of excavation. Instead of seeking a full trajectory in the state-time space, this thesis proposes a method to optimize in the control space of one-step plans. This concept is illustrated in Figure 92. S_1 is the set of all possible plans, S_2 is the set of feasible plans (those plans that are

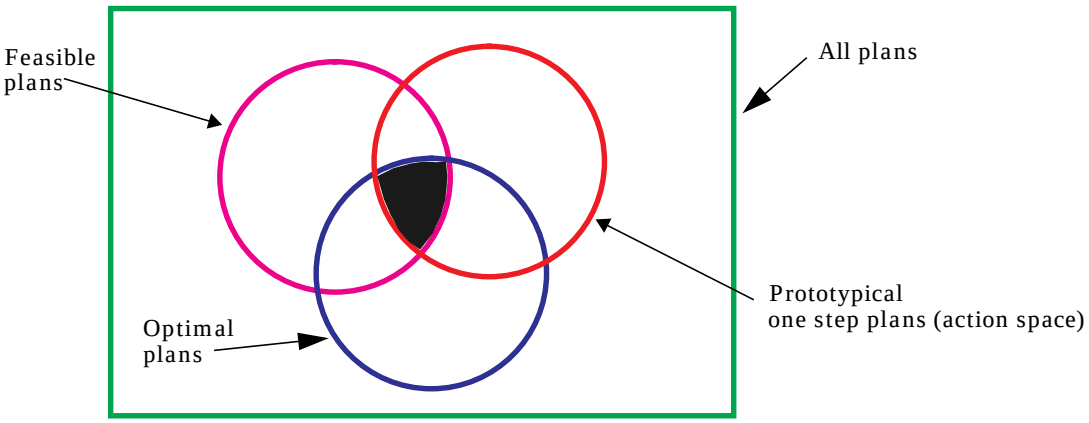


Figure 92 Subsets of the control space. After figure in [Kirk70].

physically possible and that cause admissable trajectories), S_3 is the set of plans that are represented by the action space parameterization, and S_4 is the set of plans that satisfy some optimality criterion. The method proposed seeks to find at least one plan in the set $S_1 \cap S_2 \cap S_3 \cap S_4$. The task of transforming the world state $\underline{x}_o$ to $\underline{x}_o$

It is worth reiterating two points about the solutions that are found by such a scheme. First, the sequence of solutions that emerge from one-step planning will be suboptimal since the emphasis is to always maximize short term benefit. In the general case, it is possible

that the robot will choose and execute a one-step plan that will keep it from completing the entire task. That is, it might find itself in a local extrema in the underlying state space and the only way to progress will be to first undo some work done. This thesis doesn't suggest any answers to this problems and assumes that a strategic planner that provides subgoals will ensure that such a case doesn't occur.

Two methods of optimization, exhaustive search and simulated annealing, have been explored. Exhaustive search is a simple method that in the general case requires enumeration of each cell of a discretized control space. It is possible however, to improve efficiency by pruning the search space. A second method is to use a stochastic version of gradient descent, also known as simulated annealing. Note that simulated annealing is a probabilistic method that is intended to improve search efficiency. In practice, this increase in efficiency requires adjustment of several blackbox parameters such as the rate at which the randomization decreases (also known as the *cooling* rate).

Chapter 7 Results and Analysis

The methodology proposed in this thesis has been implemented in simulation as well as on a real digging robot. An industrial, hydraulic manipulator was modified to perform digging experiments in a soil box. This chapter presents both the simulated and physical environments and discusses the results that were obtained.

Our experiments validate the notion that the real world is much less forgiving than simulated scenarios. This is due to two reasons. First, it is very difficult to model the world accurately, and hence practical simulations must be approximate by necessity. For example, in simulating a digging action, this thesis has assumed that the volume of soil swept by an excavating tool is completely removed. In fact, the actual phenomena is very complex. Soil starts to heap and settle as soon as the robot starts moving, and the amount of soil that ends up in an excavator bucket is a function of soil properties and the efficacy of the trajectory[1]. Second, simulated worlds do not model some interactions between a robot and the world simply because they are not known before physical implementation. For example, in the experiments conducted a force sensor and a laser scanner were used to accurately model the surface of the terrain on which a robot operates. In theory, elevation maps built from laser

1. Irrespective of the swept volume, if there is no scooping motion at the end of a digging trajectory, no soil will be left in the excavator bucket when it lifts out.

range data can be biased due to noise or calibration errors, and it should be possible to correct for these errors through the use of a force sensor to detect contact with the terrain. Surprisingly, some of the failure modes of the force sensor can exacerbate errors due to range data.

An important conclusion of the experiments conducted for this thesis is that consideration of force is absolutely essential for autonomous excavation. Force must be considered at two levels. Force must be considered during planning so that the robot doesn't attempt to execute a plan that is grossly infeasible. It must also be taken into consideration during execution. Without a low level control method that can modify approximate plans, a planner requires accurate sensory data and needs to maintain precise models. In fact, not only are our models approximate due to sensing errors, it is often not possible to perform repeatable experiments. For example, we found that in our testbed, it was impossible to ensure identical initial conditions for digging experiments. Even with homogeneous soil, it is not possible to "reset" to some starting state since once soil is disturbed, its properties change. Our experience points to the need for a control mode that will modify a nominal trajectory based on the soil stiffness encountered.

A stronger dependence on a robust control scheme will reduce the need for some expensive computations that are currently performed by the planner. For example, one of the most expensive calculations performed by the planner is the estimation of the volume swept by a digging trajectory. This calculation is necessary to compute the utility of a candidate dig and also to ensure that a dig does not excavate more soil than a bucket can hold. However, this estimation can be significantly inaccurate, especially for granular soils. Under certain conditions this computation can be done away with altogether. In fact some of the best results obtained to date have been when the planner has used only shaping, reachability and force constraints. This scenario, however, requires that either an accurate predictive force model be available a priori (as was developed for this thesis), or that a reactive controller is available.

7.1 Implementation on Testbed

A special testbed was constructed to conduct excavation experiments for this thesis. The testbed consists of a hydraulic manipulator, a force sensor, a soil box (3m x 3m x 1m), and a laser scanner (Figure 93). The manipulator is equipped with a scooping end-effector (bucket) and is positioned such that it can excavate in the soil box. The laser scanner enables estimation of the shape of the terrain being excavated. The force sensor measures the interaction forces between the robot and the terrain.

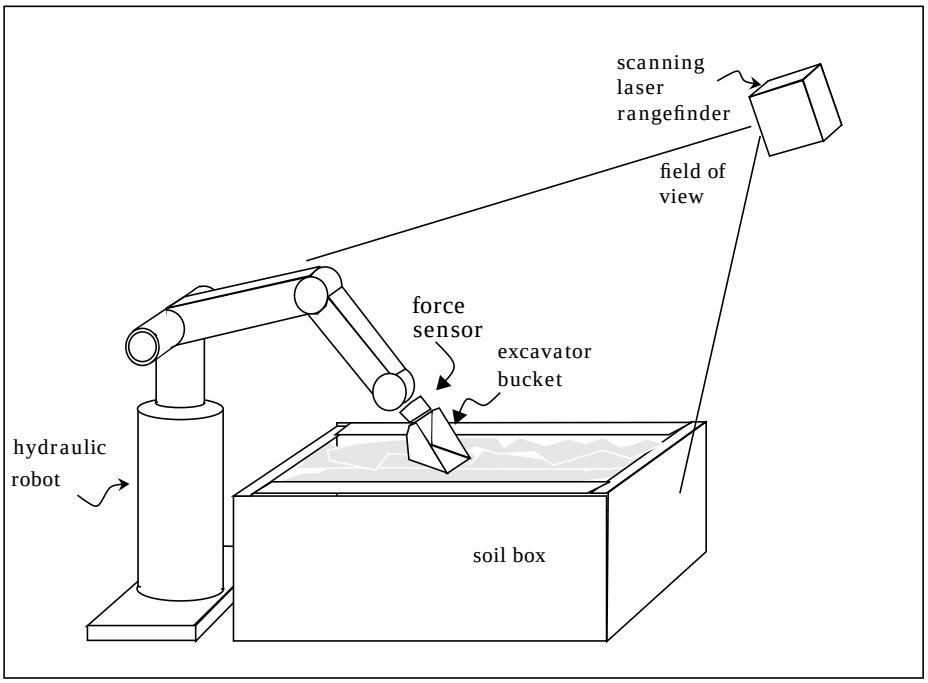


Figure 93 The excavation testbed consists of a hydraulic manipulator, soil box, force sensor and scanning laser scanner.

In the following subsections, each component of the testbed is examined in greater detail. Special attention is paid to the failure modes of the sensors.

7.1.1 Robot

A Cincinnati Milacron (T3-566) hydraulic robot has been used for the experiments. The robot is rated to be able to handle objects weighing 150 lbs at the end-effector. An excavator bucket from a small commercial trenching machine has been mounted as an end-effector on the robot. The robot and its workspace are shown in Figure 94.

Although the robot has a large workspace in terms of volume of points that it can reach, the effective workspace for excavation is much smaller. This is because the digging trajectories not only specify position but also the orientation of the excavator bucket. Thus, although the robot can attain a large volume of positions in its workspace, the volume in which a suitable range of orientations is available is very small (Figure 95). In fact, the workspace in which a range of pitch orientations (-30 degrees to 30 degrees) are all available is so small that the planner must reason about digging trajectories and verify that candidate digs do not require the robot to exceed its workspace.

The T3 has a rudimentary interface which only allows point to point control. That is, the robot is commanded to explicitly move through a sequence of end-effector poses (*x, y, z, roll,*

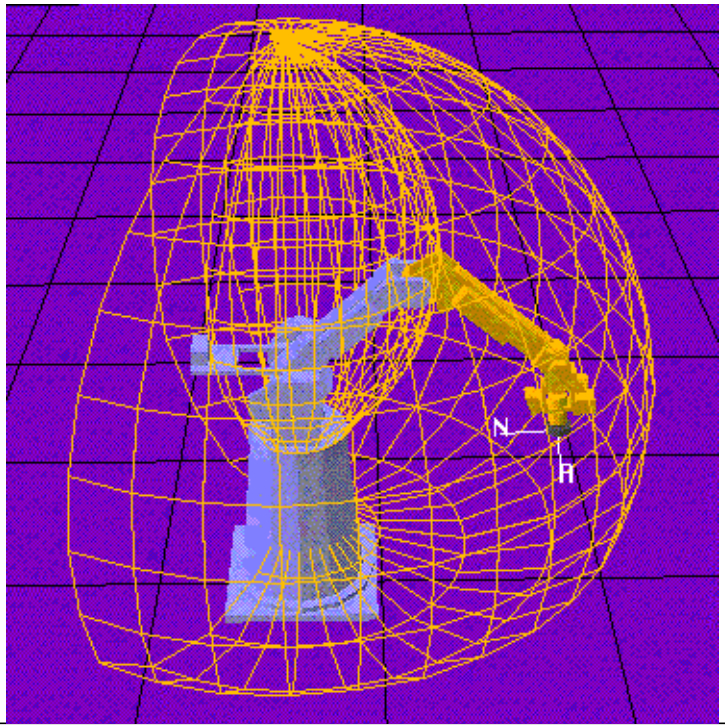


Figure 94 The T3 robot and its workspace.

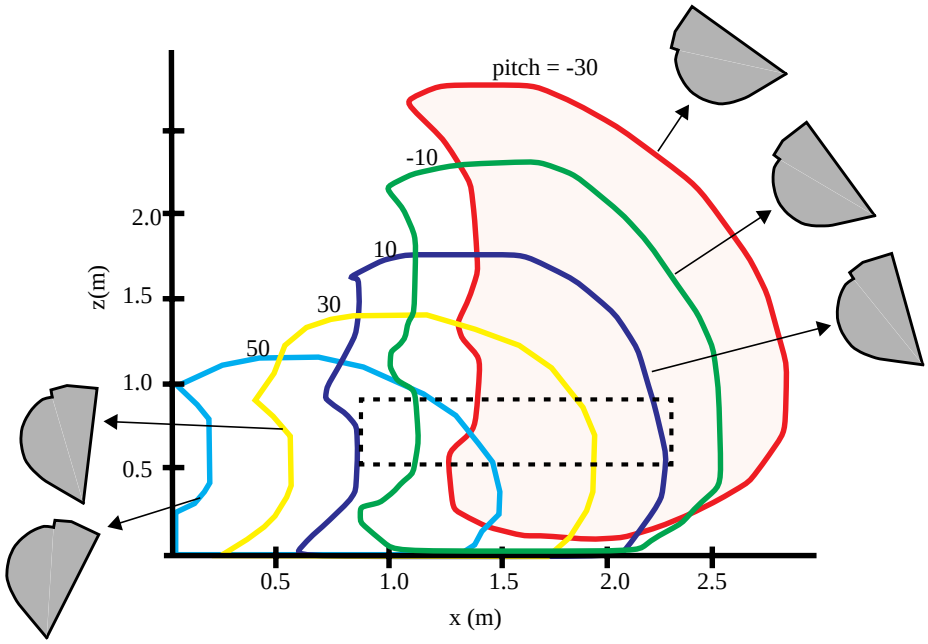


Figure 95 Two dimensional view of the manipulators workspace for various values of the bucket pitch angle. The dashed box shows the area in which the excavation experiments are conducted. Even in this box, the robot is not able to reach all orientations that we might specify. Given this situation, the planner must ensure that all digging trajectories required are kinematically feasible.

pitch, yaw) in cartesian space. When the robot's controller receives a motion command, it executes joint trajectories to implement straight line motion of the end-effector. Once a move command has been issued, the destination of the end-effector cannot be modified until the robot has stopped moving. In addition, while the robot is in motion, it is not possible to query the robot for its position. This type of interface limits the control modes that might be considered for excavation. For instance, it is not possible to implement a scheme like stiffness control because it requires control and sensing at a high frequency. With the given interface, the planner is limited to specifying simple sequences of motion commands.

7.1.2 Terrain Perception

We use a laser scanner to provide feedback about the shape of the terrain after each digging action. We have used a two-dimensional laser scanner built by Perceptron Inc. This device produces both reflectance and range images of its field of view. In the reflectance image, the value of a pixel corresponds to the brightness of spot being imaged while in the range image, the value of a pixel corresponds to the distance from the scanner to the spot being imaged. Figure 96 shows sample range and reflectance images of the soil box produced by the laser scanner. These images are processed through a series of steps to reconstruct the shape of the terrain. This thesis has adapted software developed by researchers who have built terrain maps for use by a walking robot for exploration on Mars [Hebert89, Hoffman92]. This software filters the image, performs the polar to cartesian transform necessary to convert the range images into surface points, and accounts for occluded regions of the elevation map. The output of is an elevation map of the soil box. That is, a 2D patch corresponding to the soil box is tessellated into cells (5cm square in our case), and each cell is assigned a single elevation. While this presumes that the terrain can be described as a single value function (no overhangs permitted), the assumption immensely simplifies the map representation. A sample elevation map constructed from a range image of the soil box is shown in Figure 97.

As with any imaging system, calibration is a major issue. There are two kinds of calibration that must be done. First, it is important to know the pose of the scanner with respect to a global coordinate frame and second, assuming that the range measurements from the laser scanner are linear to the actual range, a scale and offset for the range measurements must be found. The perception system described by Hoffman and Krotkov [Hoffman92] includes a module which solves the calibration problem by finding the pose of the scanner and the scale and offset parameters that minimizes the error given range readings to known points in the field of view of the scanner. Kweon et al found that apart from gaussian noise in the range measurement, there was a systematic effect in range measurement based on the ambient temperature [Kweon91]. Some of these experiments were repeated for this thesis.

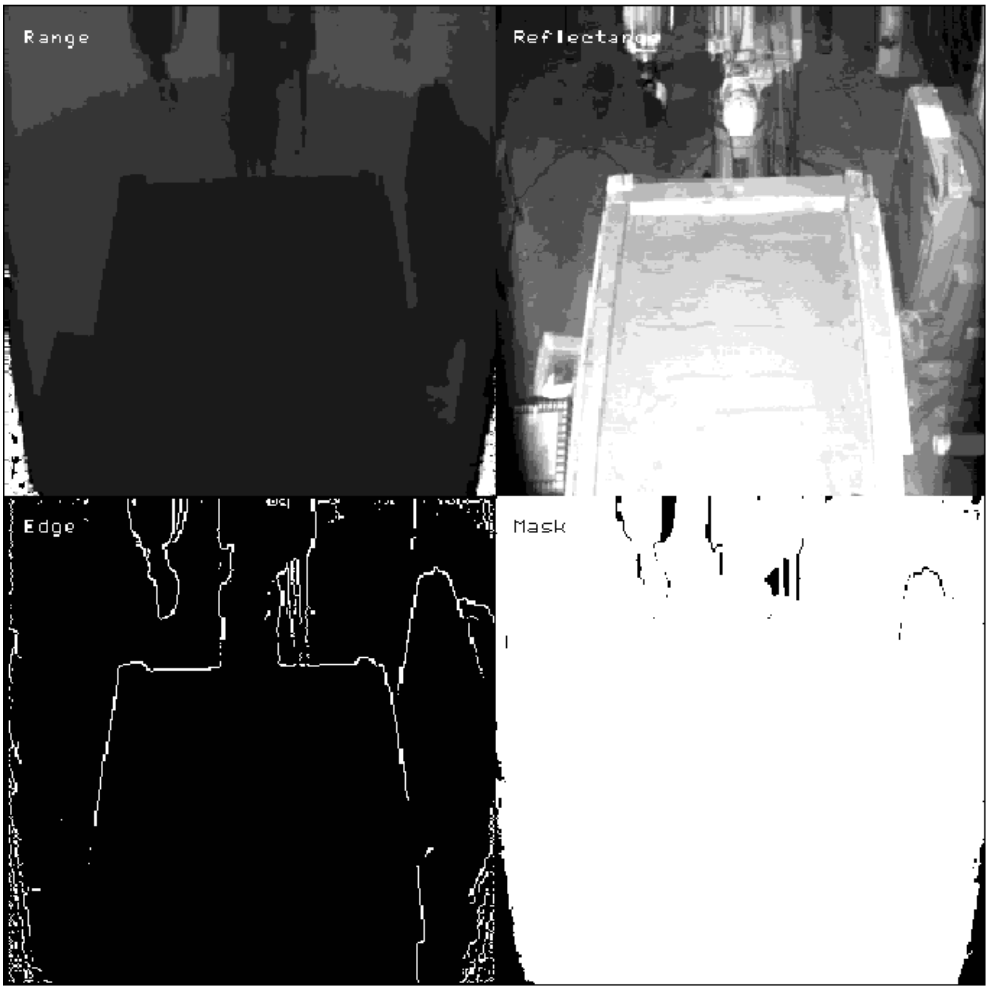


Figure 96 Processing of images from laser scanner. The top left panel shows a range image produced by the laser scanner. The value of each pixel in this image is equal to the range from the scanner to a point in the scene. The top right panel shows a reflectance image. The value of pixels corresponds to the brightness of points in the scene. The bottom left panel shows edges that have been found in the range image. These edges are used to compute areas in the scene that might be occluded. In this case, it is possible to set a bound on the elevation in the occluded region. The bottom right panel shows pixels (dark) that have been filtered out, that is, these pixels are not used in the computation of the elevation map.

For example, Figure 98 shows range to four targets over several hours. Two kinds of error are observed. One is a random error that has a gaussian distribution around the true value. The other error seems to be systematic and varies over a much longer time scale. Experiments have shown that this error is dependence on ambient temperature.

Since the entire calibration procedure involves an expensive computation, Hoffman and Krotkov solved the problem of the drift in range due to temperature by artificially lowering or raising the entire terrain map based on the difference between the estimated elevation and the elevation measured by the amount of leg extension required by the walking robot to

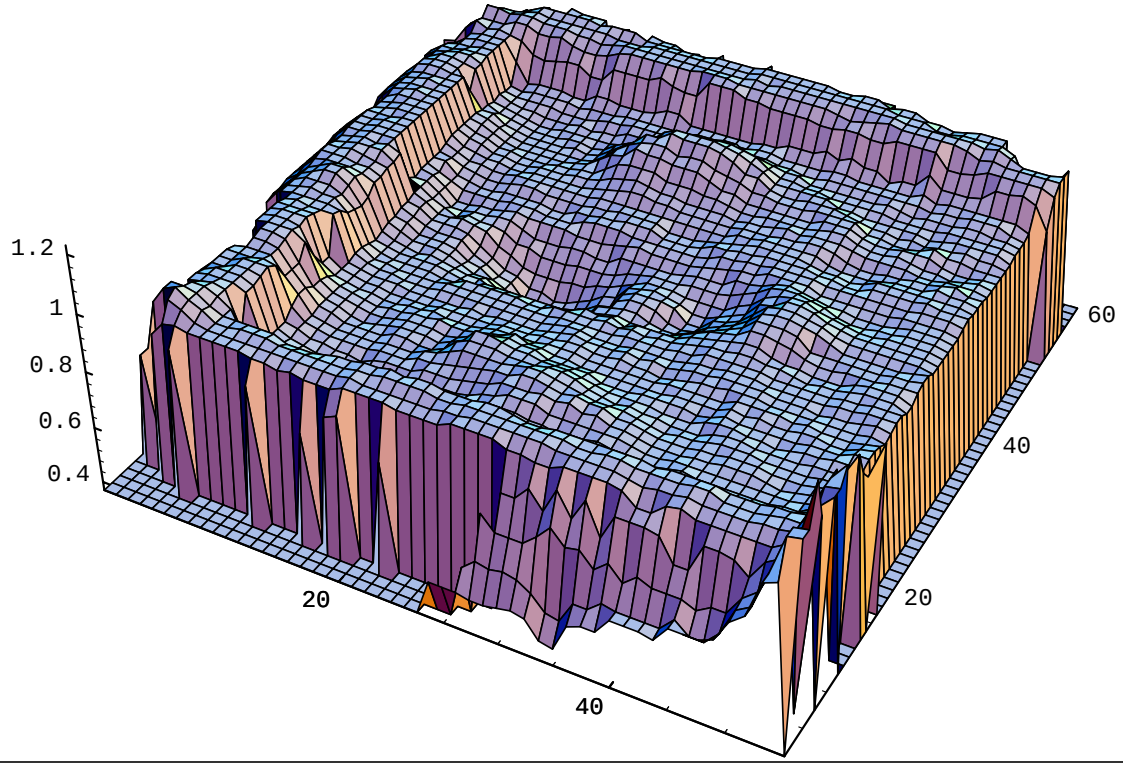


Figure 97 Example terrain map produced from range image

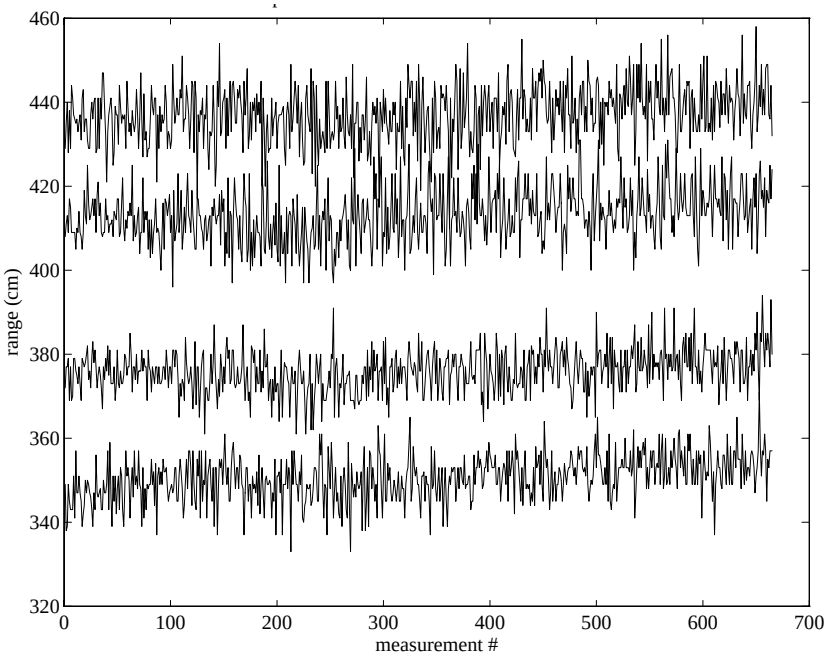


Figure 98 Range values to four points over time. Two kinds of errors are noticed. The first is a random (gaussian) noise around the true value. the second is a systematic variation that happens at a longer time scale. Experiments have shown that this variation is due to ambient temperature.

achieve contact with the terrain. For the task of robot walking, the errors observed in the elevation map (sometimes as high as 30 cm) due to a combination of the drift and noise, were acceptable. In the case of excavation we have found that the errors due to drift and noise can significantly affect performance. One reason is that the digging trajectories are executed open loop. That is, digging trajectories are always executed as planned; there is no chance of correcting the trajectories based on the forces developed.

Figure 99 shows the effect of errors in the scaling of range measurements on the estimation of the terrain map. Small scaling errors can cause problems because plans are made based on the estimated terrain map. The fact that terrain is different when the plan is executed can mean that either the robot does not engage the terrain fully or engages it too aggressively (Figure 100).

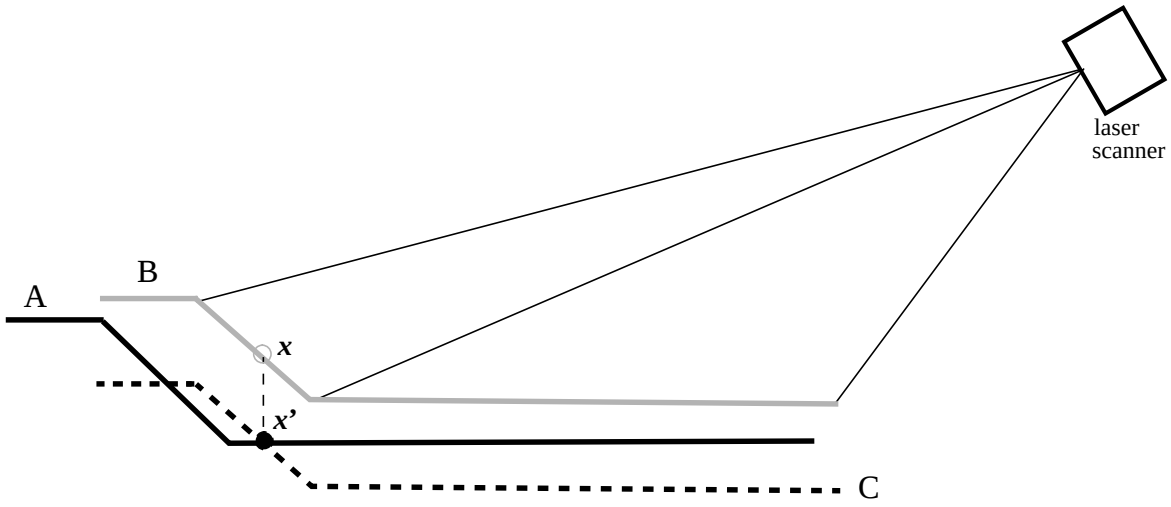


Figure 99 The effect of a scaling error in range measurement on the terrain map. If A is the actual profile of the terrain, a scale error of 10% (each range measurement is 90% of true range) will result in the terrain profile shown in B. One method to compensate for this error is to adjust the elevation map based on the error between estimated elevation and actual elevation found by detecting contact with the terrain. This method fails when the terrain is not flat. If the robot starts at point x and descends until contact is found (x'), then the adjusted profile (C) will not accurately depict the true profile.

Since temperature at our testbed can vary with season and time of day, an additional calibration scheme has been devised to reduce the problem due to scaling errors. Assuming that the transform from the scanner frame to the world coordinate frame can be found and is available, the additional procedure calculates the scale and offset parameters every time a range image is taken from the Perceptron. The method uses targets on the walls of the sandbox that have been surveyed accurately with respect to the scanner (Figure 101). Figure 102 shows a least-squares fit between measured range values and surveyed values. This fit pro-

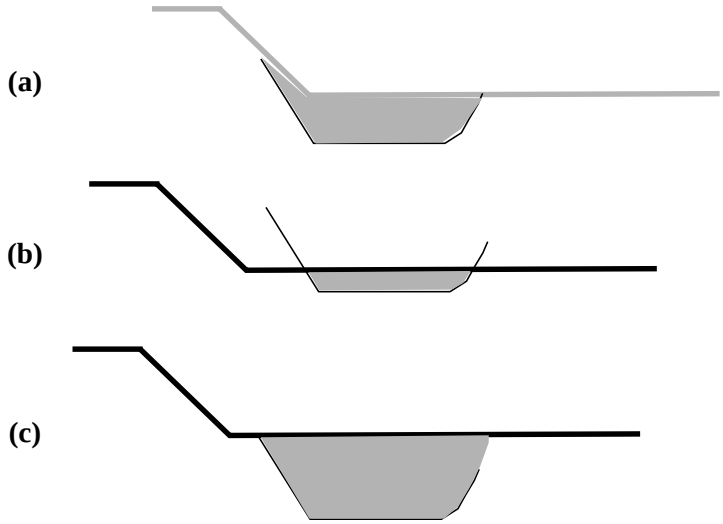


Figure 100 The effect of scaling error on digging trajectories and swept volume (dark area). A dig is planned based on the estimated shape of the terrain (a). If the dig is executed without adjusting for the error in range, the bucket does not engage the terrain fully (b). On the other hand, the if the elevation map is adjusted using the technique shown in Figure 99, the bucket engages the terrain too aggressively (c).

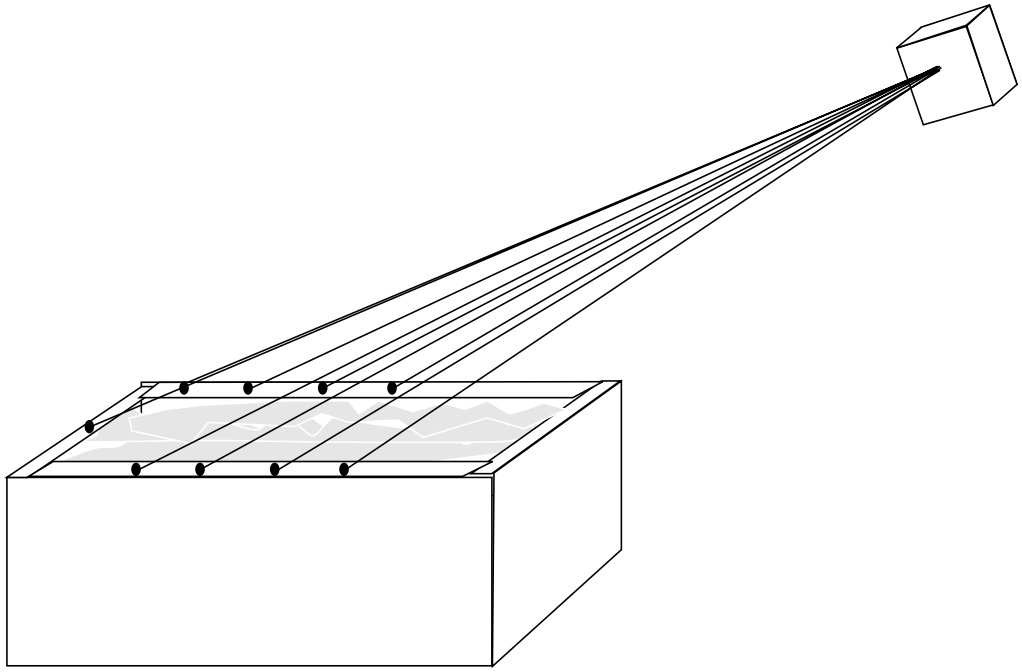


Figure 101 Calibration of range measurements. Range from the laser scanner to a set of targets (dark spots on the walls of the soil-box) in the field of the view has been surveyed accurately. Every time a range image is taken, range measurements to the targets are compared to the surveyed range and scaling parameters are computed.

vides scale and offset parameters (scale = 1, offset = 0 means that the scanner provides per-

fect range) that are used to correct the range measurements in the entire image.

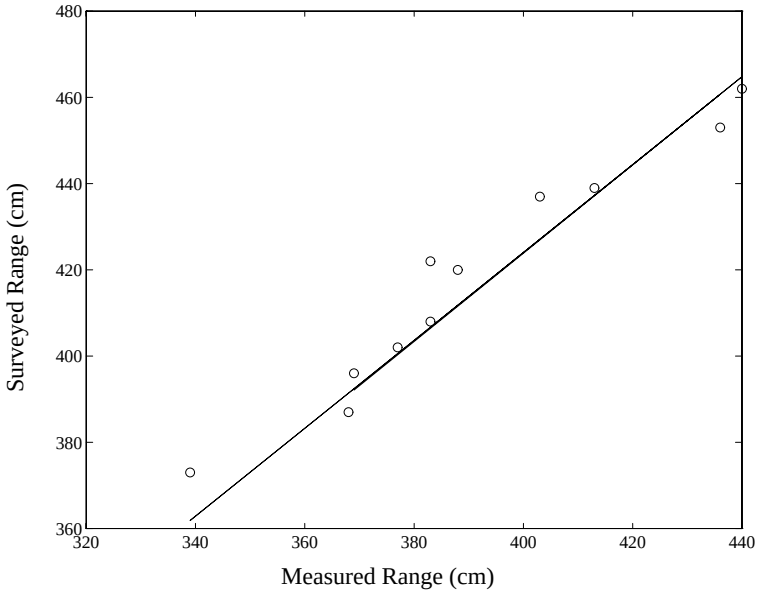


Figure 102 A linear least square method is used to determine scaling and offset between measured and surveyed range values. These parameters are used to adjust range measurements in the entire image.

In addition, to improve noise rejection, for each elevation map we use ten range images. Each pixel value is median filtered across the ten images to remove outliers. Experiments indicate that with the addition of the median filtering and rescaling, the standard deviation of the residual elevation error at the surveyed targets is approximately 2 cm. This error can be greater by a factor of two when the terrain is not flat, but the error is typically small enough that the digging trajectories are not severely affected by the errors in the elevation map.

7.1.3 Force Perception

A six-degree force-torque sensor manufactured by JR^3 Inc. was mounted between the robot and the bucket. (Figure 103). This device provides a convenient method of measuring

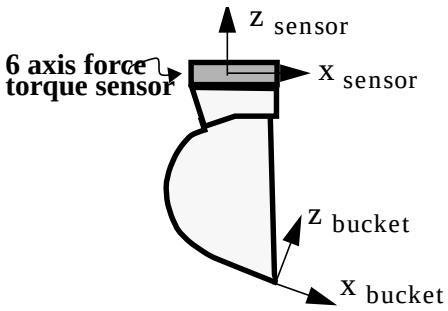


Figure 103 6 axis force torque sensor mounted on the excavator bucket.

interaction forces experienced at the bucket The sensor is equipped with a processor that can be programmed to produce force and torque measurements in any coordinate reference frame. The sensor has been programmed to register forces with respect to the tip of the bucket as shown in Figure 103. A dedicated process communicates with the force sensor and records forces and torques at approximately 10hz.

The force sensor is used in two ways. First, it is used to ensure that the bucket starts digging in contact with the surface of the terrain. Before a digging trajectory is executed, the bucket is brought to a nominal point above the terrain. The bucket moves down until the force sensor indicates that contact has been made with the terrain. Ideally, the bucket contacts the terrain exactly as specified by the elevation map. However, if contact occurs either lower or higher than expected, the difference is used as an indication of an error in the elevation map and a signal that the digging trajectory should be adjusted correspondingly. This simple operation has two failure modes:

- Since the robot's motion cannot be interrupted, search for the terrain is accomplished by incrementally moving small distances towards the terrain. Under certain configurations of the joints, the end-effector overshoots the destination and oscillates briefly before settling. Force readings must be taken after this oscillation has died down. A side effect of this behavior is that the terrain is sometimes not detected until the bucket is below the original surface of the soil. This process is illustrated in Figure 104. If this error is repeated several times consecutively, it can

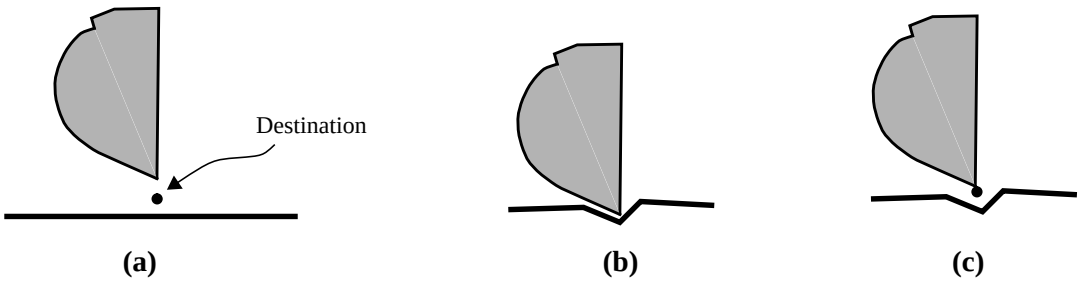


Figure 104 Failure of the force sensor to detect the surface of terrain due to oscillation of the end-effector. Bucket is commanded to a destination (a) but it overshoots and contacts the surface of the terrain (b) deforms it, and settles at the destination. Reliable force readings can only be taken when the bucket has stopped moving, so even though the terrain has been contacted, there is no way of knowing this.

cause a significant adjustment in the digging trajectory since the planner is unable to distinguish this condition from an error in the elevation map.

- When the terrain is uneven, the bucket might not be able to reach a specified pose. Interfering terrain can incorrectly signal an error in the elevation as shown in Figure 105. This condition can result for either of two reasons. Either an error in the elevation map might not reflect the shape of the terrain that would cause collision, or, the planner is not able to detect the collision due to numerical approximation.

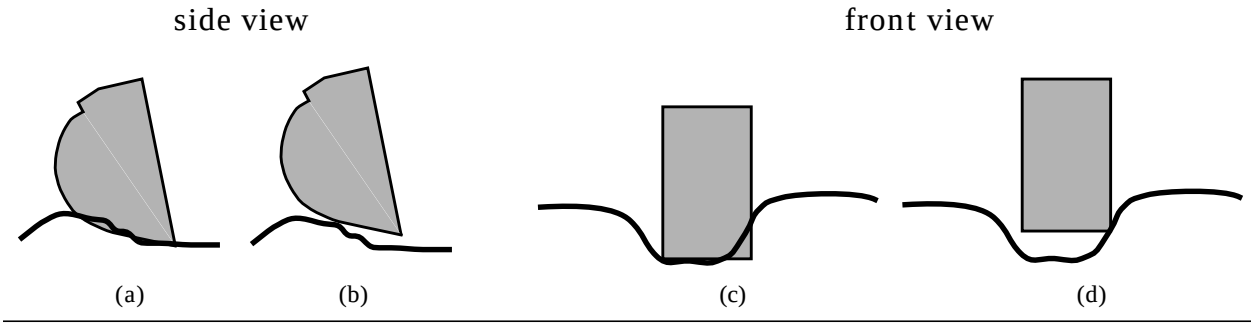


Figure 105 Two conditions in which the bucket cannot get to a specified pose because of interference with the terrain (a) & (c). The force sensor signals contact (b) & (d) before the bucket can get to the specified pose which in turn can lead to erroneously adjusting the elevation map.

The other way the force sensor is used is to develop a model of the interactive forces developed while performing excavation. In this mode, the force sensor starts recording time-stamped force data as soon as the bucket starts excavation. Later, the force data are correlated with the nominal trajectory of the end-effector to develop a force model. Since it is not possible to query the robot for its position while it is moving, two kinds of errors are possible. A digging trajectory is specified as a list of via points that must be achieved. There is a small but varying delay as the robot pauses at these via points. Added to this are small communication delays from the time that the planner specifies that the robot should start moving and the time when it actually starts moving. Cumulatively, these delays amount to an uncertainty in the robot's position along its trajectory. A second error is due to the fact that the interaction between the robot and the terrain can involve significant interaction forces. It is possible that under great resistive forces some of the joints comply. In effect, the end-effector is not where it is expected to be, but there is no way of detecting this condition.

7.1.4 Computing Architecture

Figure 106 shows the architecture of all the modules in the system. Currently, the system runs on four processors that communicate with each other using a message passing scheme [Simmons94]. One processor is dedicated to each of the following:

- I/O to the laser scanner and construction of elevation maps.
- I/O to the robot and the force sensor.
- diagnostics display and a graphical user interface.
- number crunching required by the planner.

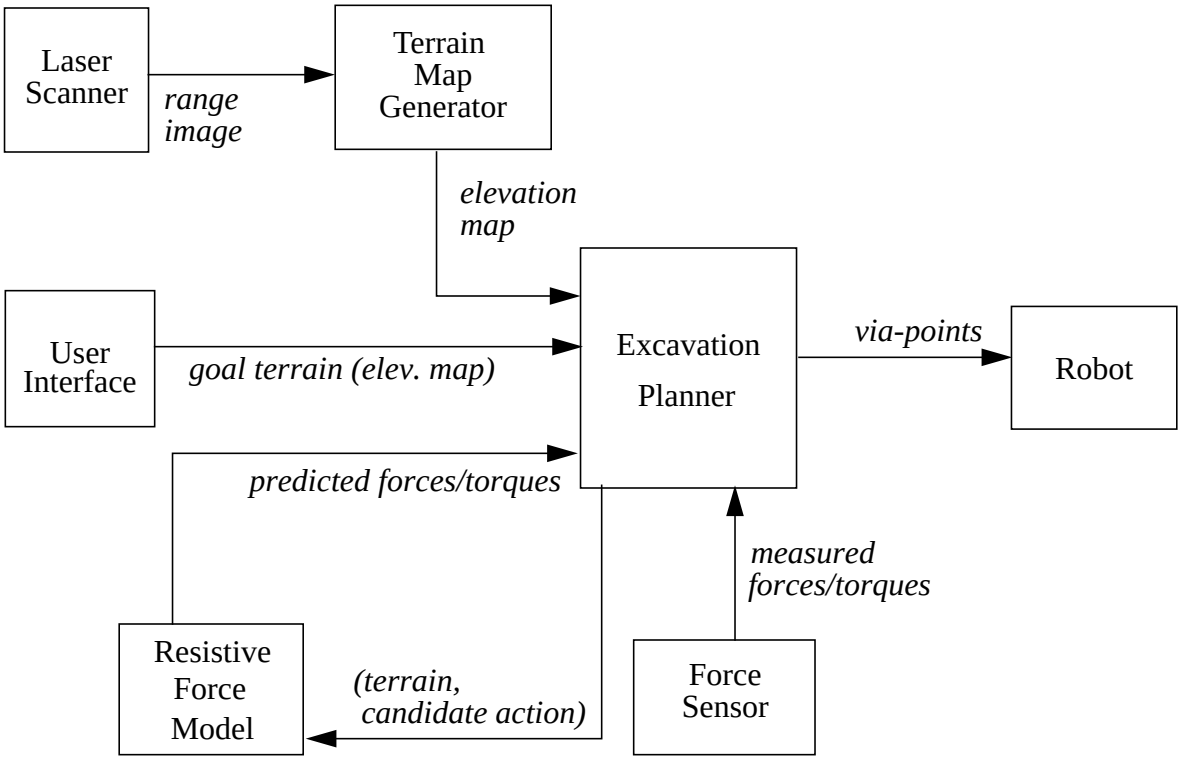


Figure 106 Computational architecture

7.2 Experimental results

This section discusses experimental results, both in simulation and as implemented on the hydraulic robot. Planning in the simulated world is based only on geometry and not force. The planner assumes a completely two dimensional world, perfect sensing and perfect control. That is, the planner is able to obtain an accurate description of the shape of the terrain, and the effect of an action (amount of soil removed) is identical to the forward model available to the planner. To add some realism to the simulation, the terrain is relaxed into a stable state after soil is removed from the terrain using the algorithm described in Chapter 2. Only the loading task has been simulated. In comparison, the planner implemented for use with the robot manipulator considers both geometry and force constraints to dig trenches to geometrical specifications.

Figure 107 shows the cycle used by the excavation system.

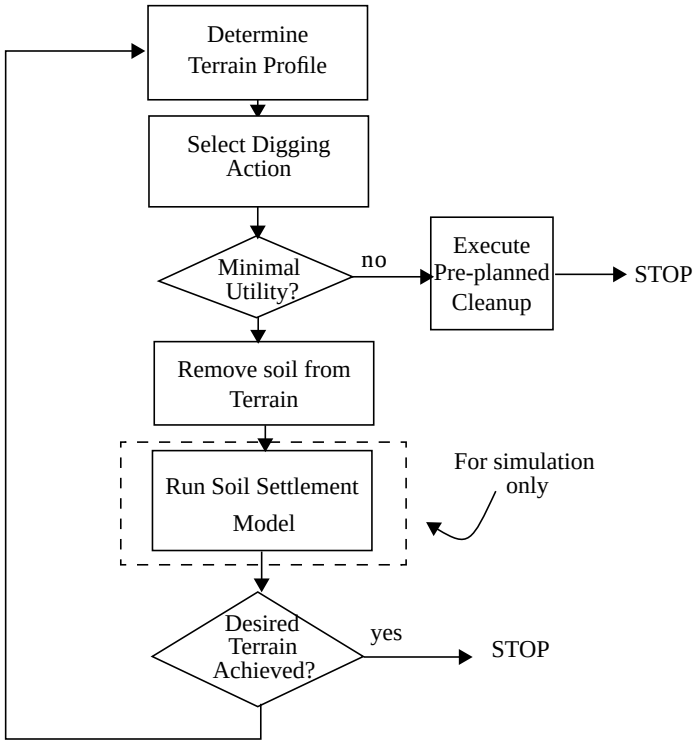


Figure 107 The excavation cycle. The planner selects one digging action at a time. In case of simulation, a settlement model is used to relax the terrain into a stable state after soil has been removed.

The first step is to determine the shape of the terrain. In simulation, this is a trivial task, since the planner is able to obtain the exact shape of the terrain. In the real system, an elevation map is constructed using range data as discussed in Section 7.1.2. Next, the excavation planner chooses a digging action from the set of actions that are feasible. If this action

has at least a minimal utility (a preset threshold), the action is executed. Otherwise, a pre-planned set of actions is executed. For the task of trenching this is done by the having the robot follow the boundaries of the trench. In simulation, the volume swept by the excavator bucket is removed and a relaxation scheme is used to bring the terrain to a stable state.

7.2.1 Loading

Figure 108 shows a two dimensional world being operated upon by a front-end loader. Only geometric constraints have been used. In this example, the loader has a capacity of 48 units of soil. 13 digs are required to completely remove the terrain. On average each action removes 46.4 units of soil.

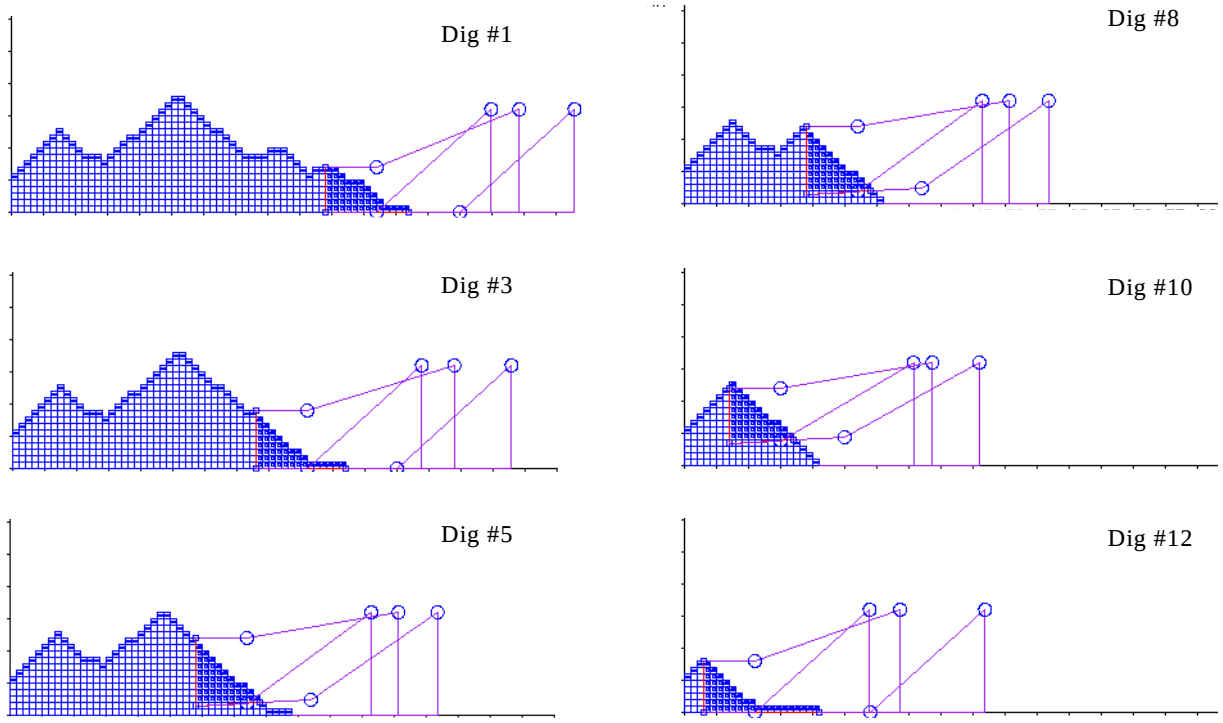


Figure 108 Simulation of loading using a front loader in a two dimensional world. A skeleton of the mechanism is shown at three points during the digging trajectory— at the start, at the end of the penetration phase, and at the end of the lift-out phase. The dark regions represent the volume excavated.

The simulation experiment was repeated, this time replacing the front-end loader with a front shovel. Recall that this machine does not move its base while it excavates and the capacity of the bucket is smaller (27 units of soil; 56% of the capacity of the front-end loader) and thus it takes longer for the pile to be removed (Figure 109). On average each action removes 26.7 units of soil.

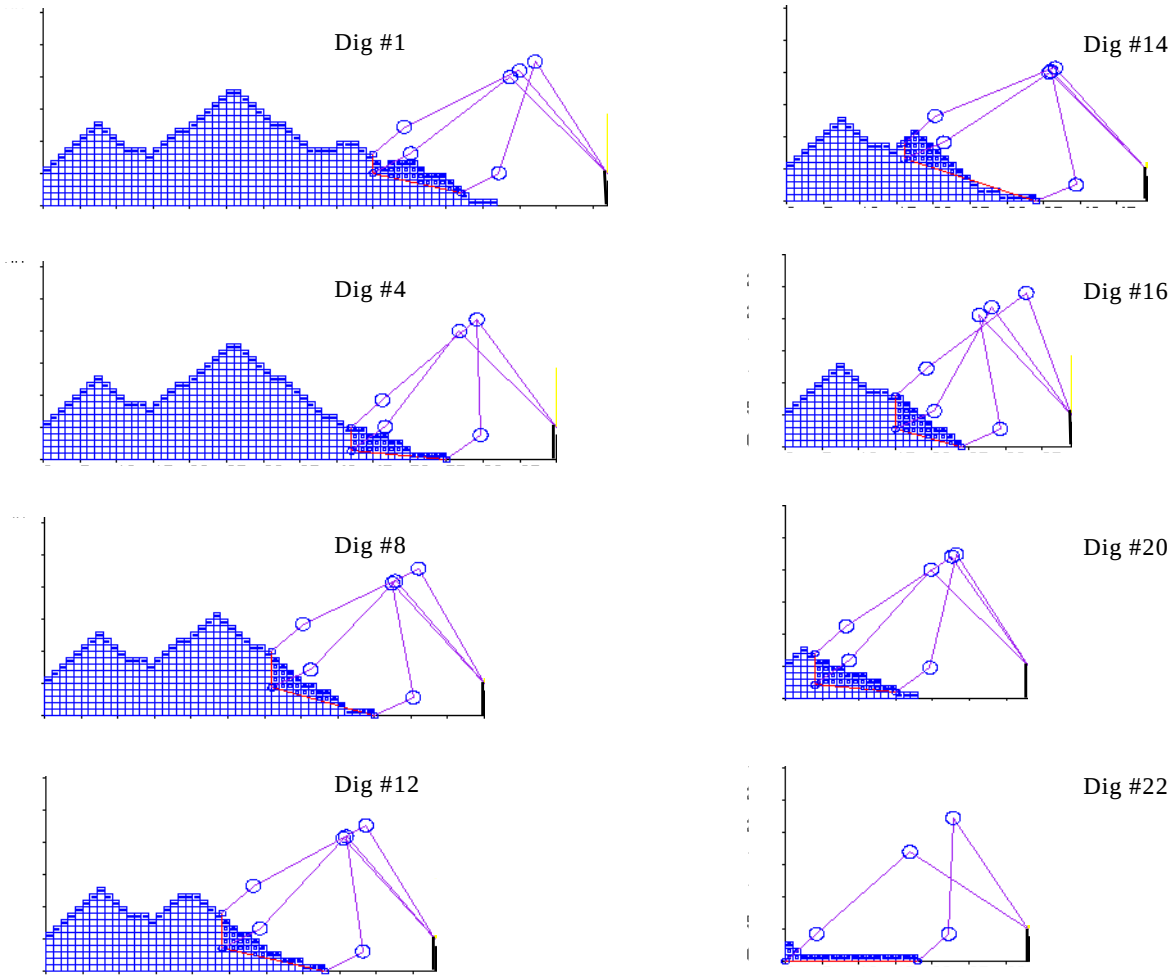


Figure 109 Simulation of loading using a front shovel in a two dimensional world. Skeletonized links of the robot are shown. The dark regions represent the volume excavated.

A simple Mohr-Coulomb model of resistive force was added to compute a force constraint in addition to the geometric constraints used for the above simulations. Figure 110 shows how the shear surface is modeled. The change in digging strategy is not only a function of the resistive force, but also of the joint torques that the robot can generate— obviously if unlimited torques can be generated, considerations of force will not cause any limitations. Figure 111 shows a sequence of digs that a front-end loader performs given a nominal torque limit. In this case, the average throughput is 32.3 units of soil. While in the previous cases, the efficiency (average throughput divided by bucket capacity) was approximately 95%, in this case, efficiency is only 67%.

Another comparison, efficiency as a function of force limitation, can be made between the two machines. For purposes of comparison, identical volume and force constraints were used

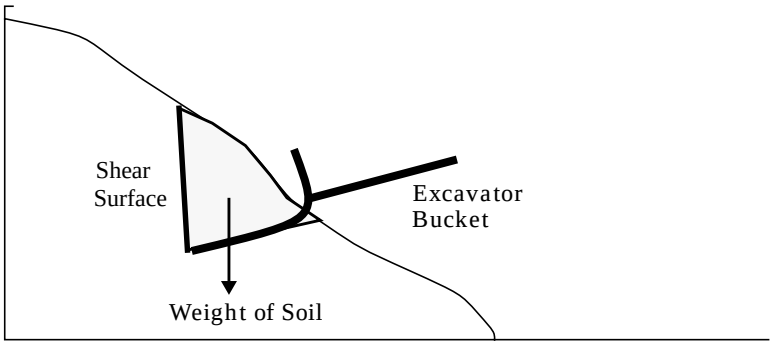


Figure 110 A simple model of resistive force experienced by the excavator bucket during excavation. Resistive force is only computed at the point just before lift out from the pile.

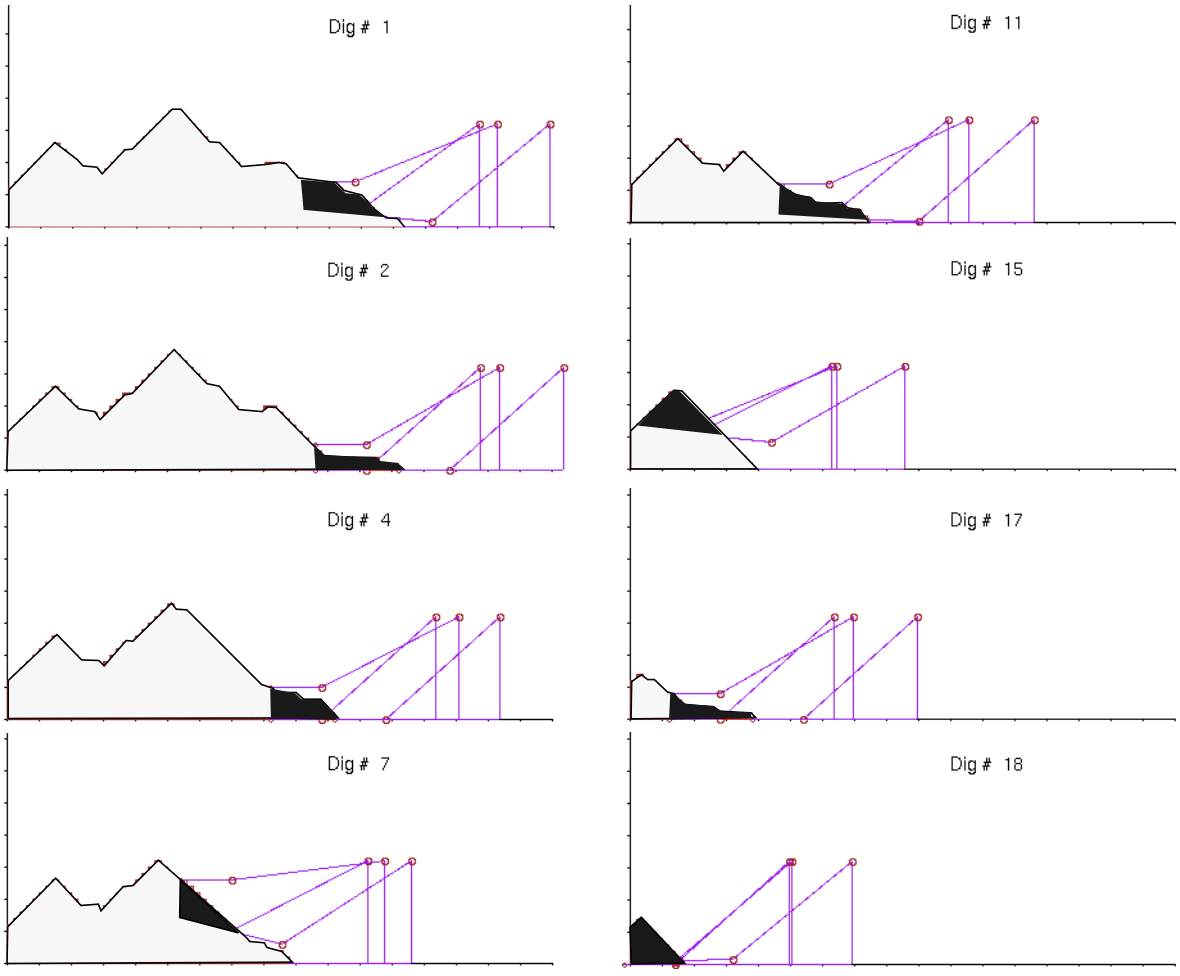


Figure 111 Simulation of a loading task executed by a front loader. Both geometric and force constraints are used.

for the two machines. The same loading task was simulated in 20 runs, each of which required the excavator to remove 1500 units of soil starting from a randomly generated con-

figuration of the terrain. This simulation was repeated for six different force limitations and average efficiency statistics were compiled (Figure 112).

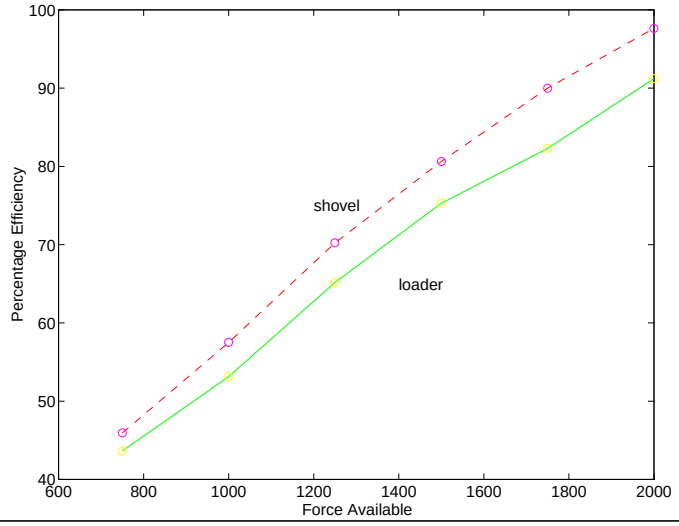


Figure 112 Comparison of a front loader and a front shovel as a function of forces that are available to the mechanism.

These results highlight two points. First, the efficiency of both machines rises approximately linearly with increase in force available. Second, given the artificial assumption of equivalent capacity and force, the front shovel is more efficient than the front-end loader. This difference is attributed to the larger workspace of the front shovel.

7.2.2 Trenching

Approximately 450 experiments were conducted with the T3 robot with two types of soil— sand and top soil. The chief difficulty in performing these experiments is the assurance of identical initial conditions. It is almost impossible to control compaction and water content. In essence, it is not possible to perform controlled experiments making minor changes to control parameters. The sensed data have enough noise that the learning method is not able to distinguish between the resistive forces developed in the two soils. Further, it is not easy to make average efficiency comparisons as with the simulations of loading. This is because the amount of soil that must be removed to clear a specified volume varies based on how loose (or compacted) and cohesive the soil is. In our experiments, as the soil got looser and more dry, the side walls started collapsing into the trench more and more. That is the angle of repose progressively decreased.

It has been possible, however, to test the effect of gross changes. For example, the effect of greatly changing the torque limitation (strength of the robot), presumed by the planner, is clearly visible. Also clear is the difference in adding or leaving out certain constraints.

Figure 113 shows a two dimensional section of the elevation map as one of the excavations progresses.

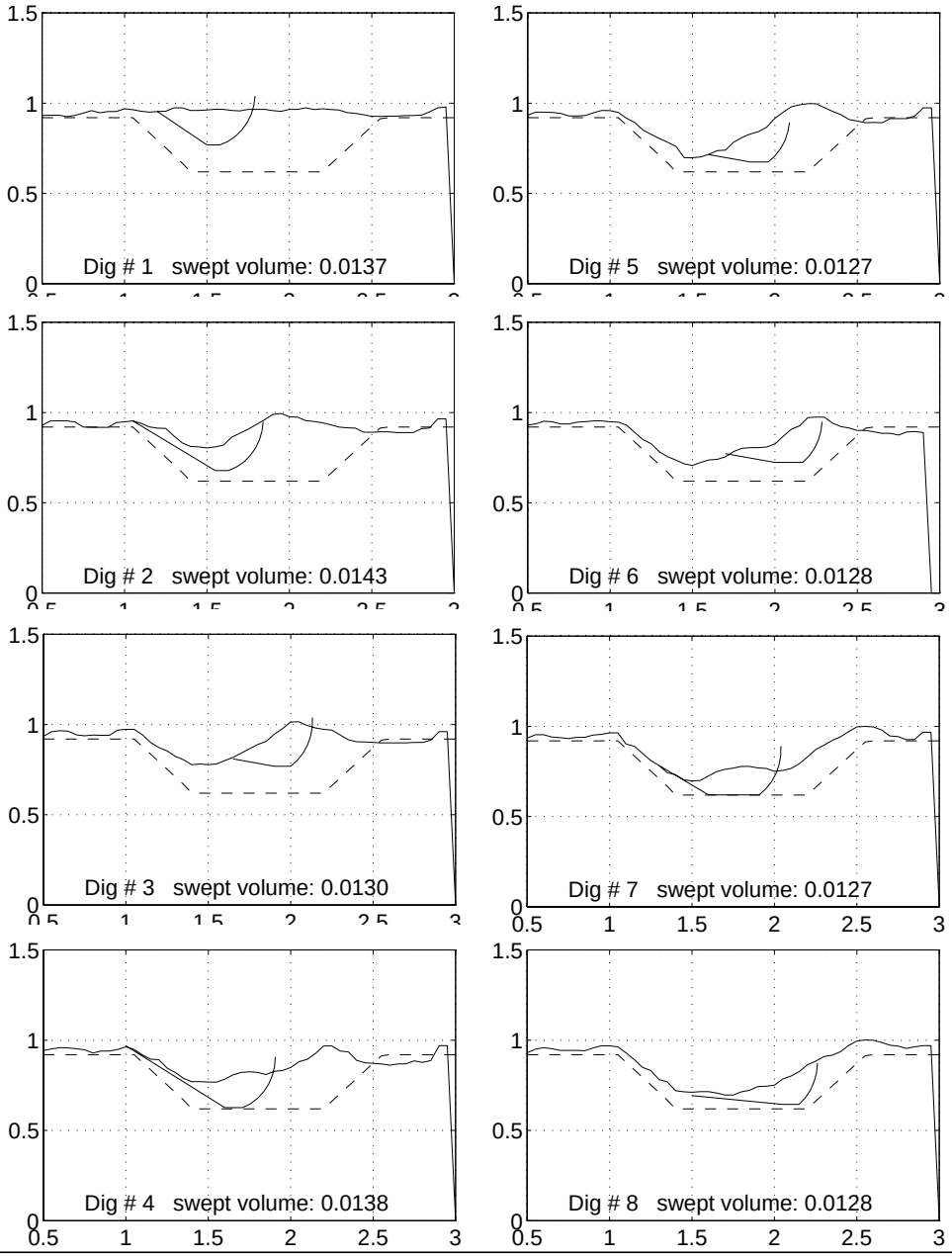


Figure 113 Excavation with shaping, reachability, volume and force (torque limit = 1000 Nm) constraints. The optimization criteria is minimum cumulative torque over the entire trajectory.

In this example, both geometric constraints (volume, shaping and reachability), and force constraints have been used. It is necessary to point out how the force constraint is implemented. Given a candidate dig and a geometric description of terrain that must be excavated, the learned force model (Chapter 5) is used to predict resistive forces. These are translated into joint torques. Since the wrist joint is the weakest joint and is the joint that

stalls when it can not deliver the necessary torques, the force constraint for the T3 robot can be reduced to checking the torques due to the contact and the weight of the robot (Chapter 5) at the wrist joint only. In the experiment shown in Figure 113, the planner assumes that the torque limit for the wrist joint is 1000 Nm. In this example, the digging actions chosen at sweep a volume of at least 95% of the bucket capacity and minimizes the cumulative torques on all the joints.

One way to evoke a different strategy that would be used with a stiffer soil is to artificially lower the joint torque limit. Figure 114 shows a sequence of terrain elevation sections under the assumption that the limiting joint torque is only 500 Nm. For this experiment, two runs under the same starting conditions are compared. It is noteworthy that the two runs start out very similarly, but as the excavation progresses, variations in the terrain and digging actions are amplified.

The experiments conducted for this thesis have borne out the hypothesis that the volume constraint can be ineffective. Approximating the volume of soil that remains in the bucket by calculation of the swept volume has varying utility. For loose soil swept volume consistently overestimates the yield of a digging action, in effect artificially limiting the yield. Conversely, in cohesive soil, the bucket is overfilled because the soil clumps as it is scooped. Since computation of the volume constraint is an expensive operation, there is a question as to whether it is useful at all. Certainly, without a good force model it is necessary to use this constraint to keep the excavator from attempting to dig large volumes of soil. In cohesive soils, the swept volume computation can be useful because if done correctly, digging will often produce bucket loads heaped beyond the volume of the bucket.

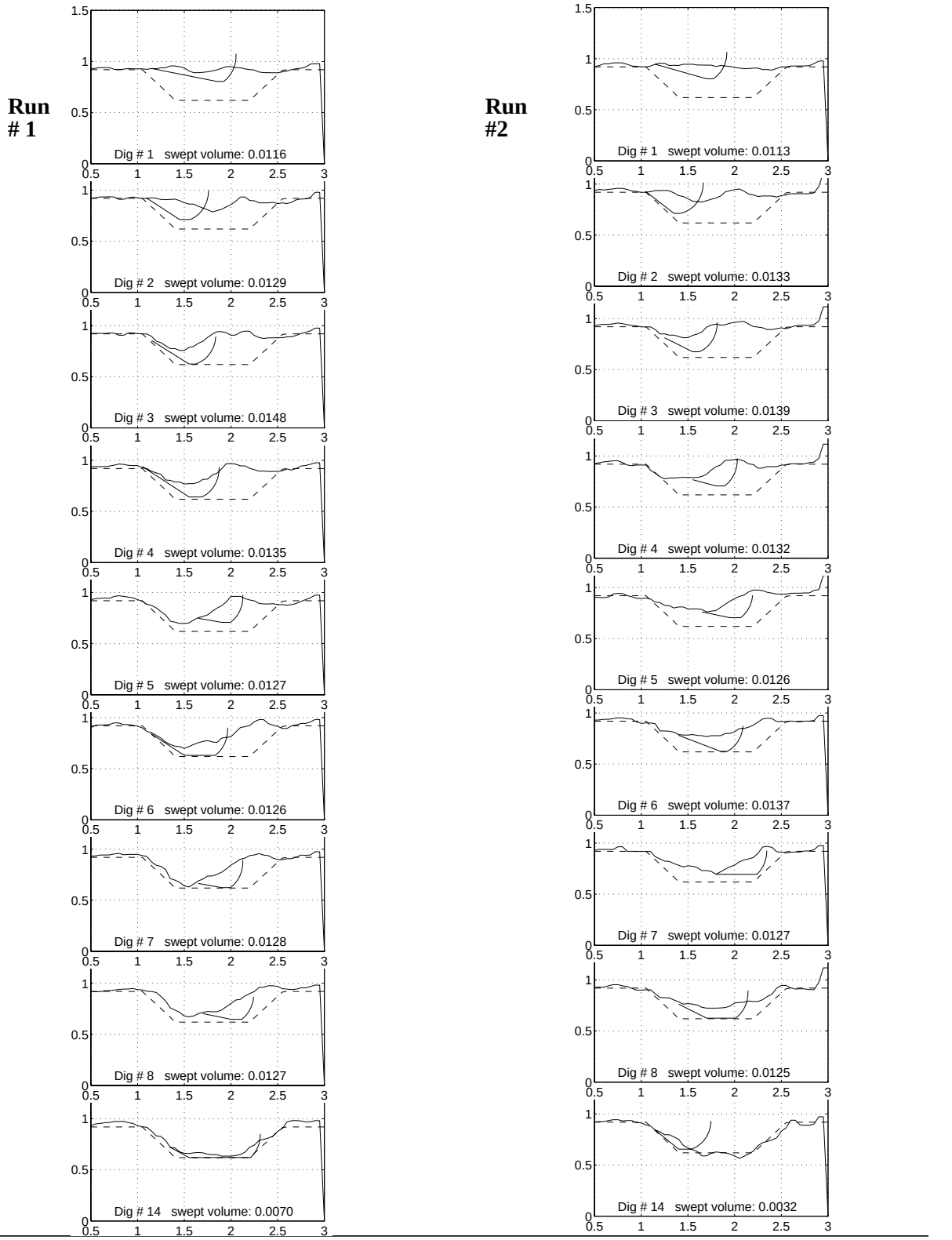


Figure 114 Two runs with similar starting conditions. Excavation with shaping, reachability, volume and force constraints. The torque limit has been reduced to 500 Nm. The optimization criteria is minimum cumulative torque.

A number of experiments have been conducted without the volume constraint. For example, Figure 115 shows a sequence of excavations for which the volume constraint has been turned off. The torque limit for the wrist joint is set at 1000 Nm and the optimization function is based on minimum cumulative torque. The most notable difference between the

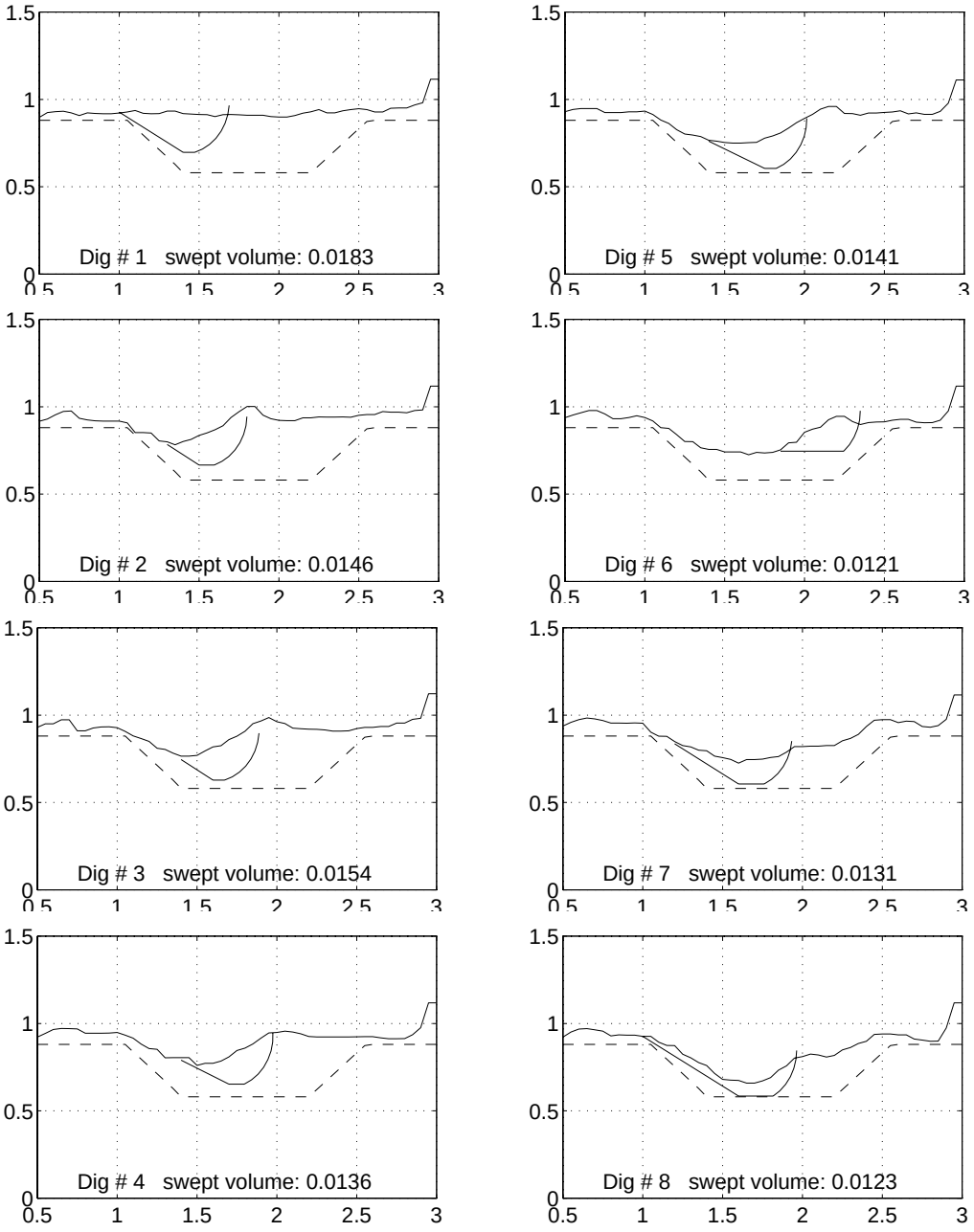


Figure 115 Excavation with shaping, reachability and force (torque limit = 1000 Nm) constraints only. The optimization criteria is minimum cumulative torque.

actions selected under this assumption is that a larger volume of soil is excavated than

before. Another variation is to use a different optimization scheme to select from the set of feasible actions. Rather than minimizing the cumulative joint torques, we might minimize the maximum torque experienced by the joints. (Note that in most cases, the maximum torque is experienced at the end of the drag phase.) Figure 116 shows a progression of excavations using this variation. A small change is observed in the types of actions chosen under

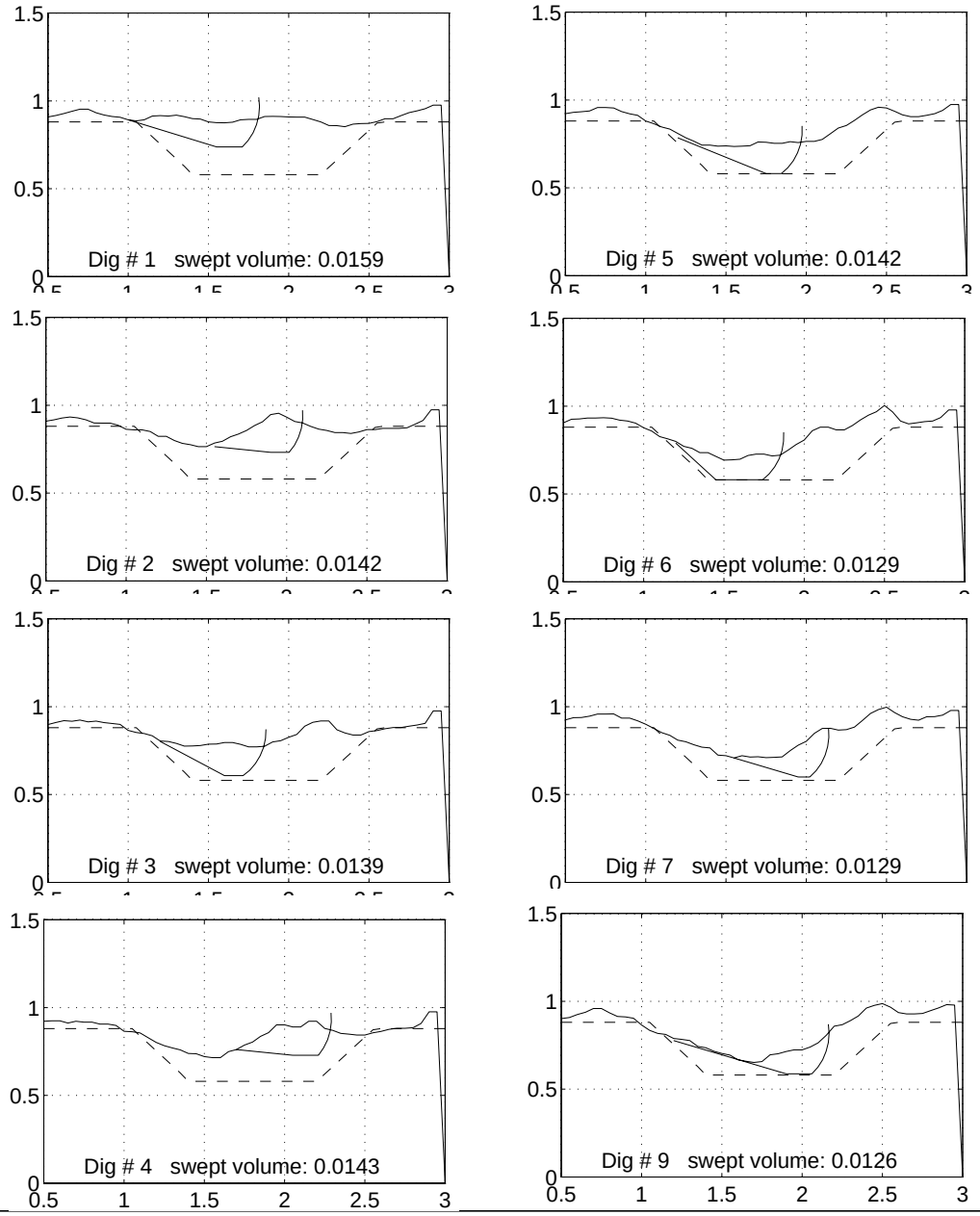


Figure 116 Excavation with shaping, reachability and force (torque limit = 1000 Nm) constraints only. Optimization criteria is to minimize the highest torque (as opposed to minimizing the cumulative torques).

this scheme. Some of the digs are shallower and hence experience a smaller maximum torque.

A final experiment changed the optimization criteria to concentrate only on the volume excavated. That is, the planner was instructed to pick among the plans that satisfied shaping, reachability and force constraints, a plan that optimized the volume excavated. Figure 117 shows the results from this experiment.

Evaluated purely on the basis of average yield, this scenario produces the best results. It should be noted that this result is possible because of a good force constraint. Without the force constraint it is necessary to use the volume constraint.

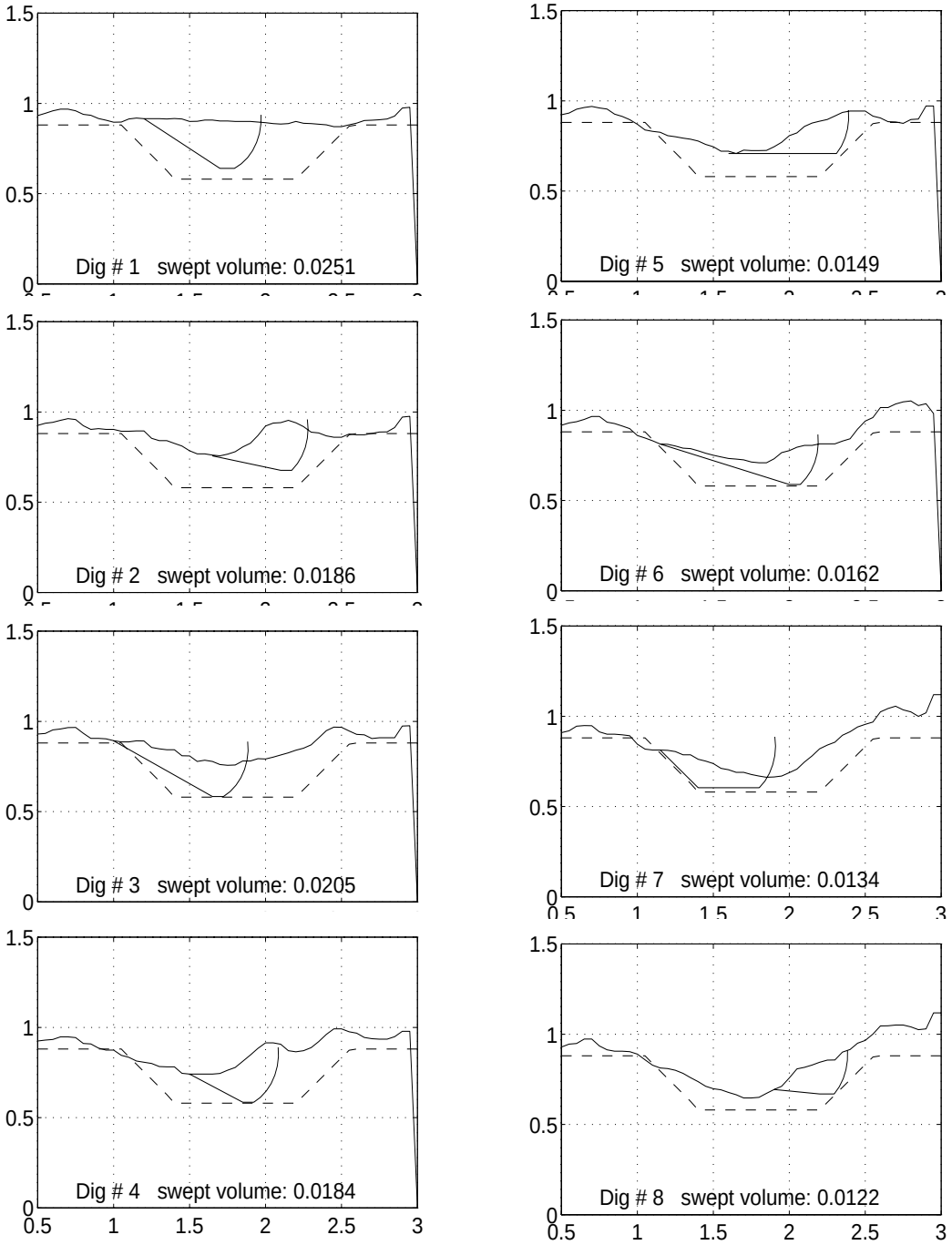


Figure 117 Excavation with shaping, reachability and force (torque limit = 1000 Nm) constraints only. The optimization criteria is maximum volume.

7.3 Performance

Since there has been a major effort to develop a complete working system, it is useful to examine the complexity of the software modules both in terms of code development and in terms of computational resources required. A simple comparison is to look at the relative size of the programs that are used for the purposes of this system (Table 6).

Table 6 Relative size (lines of C code) of the modules in the working excavation system. Some of the modules (neural networks, locally weighted regression and TCA) are included as libraries and are not fully used in the systems.

module	tasks	lines
planning	search, sort, constraint checking, evaluation	2500
robot kinematics	forward/inverse kinematics, jacobians, fast workspace lookup	2000
terrain mapping	calculation of swept volume, map interpolation	2000
learning	manipulating of action/force/terrain data, prediction of force data.	1000
	neural networks	3000
	locally weighted regression	25000
display	screen graphics for display of terrain maps, goal terrain	500
general utilities	user I/O, coordinate transforms, matrix manipulation, general numerical methods	3500
perception	I/O to laser scanner, calibration, terrain maps	5000
execution sequencing	generating robot trajectories, sequencing motions of the robot	1000
robot interface	low level communication to the robot	1000
force interface	I/0 to force sensor	1000
TCA	message passing and task sequencing	35000

Likewise, it is useful to examine the sequence of operations and the computational resources required. The computational requirements are stated in two terms. The first is an estimate of the amount of computing time required by the current system and the second is an estimate of the computing required by an optimized software. For example, the perception system used for this thesis is a modified version of the perception system used by the Mars rover project for a walking robot [Krotkov93]. This robot built terrain maps, sometimes

from long sequences of images to select foot locations. The perception needs of this robot are significantly different from those of an excavating machine. Simpler algorithms can be used to build terrain maps in a fraction of the time taken by the current algorithm. The Mars perception modules were used because they provided a robust set of building blocks, at the cost of lower performance. However, a future excavating machine would benefit from tailor-made software. It is important to note that the optimizations suggested do not require a fundamentally different methodology, or any major algorithm advancements. Rather they suggest that in some cases performance can be improved significantly with the better equipment and more efficient use of existing algorithms. Table 7 presents the computing requirements for perception, planning and execution. Each is divided into two parts.

Table 7 Comparison of average execution times for perception and planning for one cycle of a typical trenchimg task.

	function	computing time required (s)	estimated computing time for optimized version (s)	type of improvement
PERCEPTION	range data acquisition	20	1	faster and more accurate range sensor
	elevation map generation	45	1	faster and more accurate range sensor, use of simpler algorithm
PLANNING	construction of feasible action set[a]	60	30	more efficient use of intermediate results such as computation of swept volume
	selection of the plan based on optimizing criteria[b]	10	10	no simple optimization is evident
EXECUTION	set up of digging tool	30	5	improved robot control allowing interruption of motion
	execution of digging trajectory	30	30	no simple optimization is evident

a. presuming approximately 10^4 plans are evaluated.

b. presuming approximately 10^4 plans are evaluated.

Beyond these simple improvements, two other higher level modifications to the existing system would be beneficial. The planner proposed in this thesis does not address the question as to how the limits on the ranges of action space parameters are chosen. The complexity of search is linear with the dimensions of the area to which candidate digs are restricted.

If these areas can be tessellated into smaller sections, a very straightforward reduction in search is possible. From the perspective of task efficiency, it is important to confront the fact that terrain and force models will always have errors. A second improvement would be to add a reactive level to the execution of digging trajectories as suggested in Chapter 5. This would allow the robot to compensate for sensing and modeling errors.

Chapter 8 Conclusions

8.1 Main Conclusions

This research has developed an action space formulation of the task of excavation. This formulation allows for a computationally tractable approach to optimization over the space of one-step digging plans. The price to be paid for only optimizing near term benefit is that in general, the sequence of plans produced to perform the entire task is not optimal. Since the best process models we have are very uncertain, there is no hope of global optimality because it necessarily requires projection into the future.

This thesis pays a great deal of attention to the development of the parameterization of the action space for excavation. Starting with patterns used by human operators, it shows through contact analysis and experimental results, how a compact set of variables can be used to encode the task. A parameterization was developed for two tasks— loading and trenching.

This thesis has also developed a method of predicting resistive forces for an excavator. This method relies on force and topological data that are collected during prior digging experiments to produce force prediction. Three learning methods (global regression, neural nets and memory based learning) were compared. The conclusion of our experiments is that none of the methods stands out very clearly on the basis of accuracy, but that these methods

should be compared on the basis of prediction time, ease of incorporating new data, and memory requirements.

Most of the experiments conducted for this thesis have used exhaustive search to perform the optimization. While the search space can be very large, this thesis shows how it is possible to reduce computation by heuristically pruning the search space and by intelligently ordering the constraints that must be evaluated.

Over 400 experiments were conducted in our testbed. Given a geometric specification, the current implementation can excavate a trench to predictable tolerances. An important conclusion of the experiments conducted is that consideration of force is absolutely essential for autonomous excavation. Force must be considered at two levels. Force must be considered during planning so that the robot doesn't attempt to execute a plan that is grossly infeasible. It must also be taken into consideration during execution. Without a low level control method that can modify approximate plans, the planner requires very accurate sensory data and needs to maintain precise models.

8.2 Main Contributions

Th three main contributions of this thesis are:

- The action space method developed for this thesis is a new mathematical representation of the excavation task. The representation allows incorporation of goal configuration and geometric and force constraints within a single framework.
- A new method of predicting resistive forces during excavation was developed. It uses analytical models developed by researchers in terremechanics to motivate the structure of the force model and uses empirical data to instantiate the model.
- This thesis has demonstrated a fully working autonomous excavator. It is the first known system to incorporate terrain topology into the planning of digging actions.

Appendix 1 Kinematics

This appendix develops the kinematic models for a typical backhoe and the T3 robot used for experimentation. We start with forward kinematics that allows the calculation of the pose of the end-effector (tip of the bucket) given the joint angles. Next, the inverse kinematic equations to solve for the joint angles given a particular pose of the end-effector are developed. Finally it is shown how the inverse kinematics can be cached such that it is possible to quickly determine if a pose can be reached by the end-effector.

A1-1 Kinematic Model of a backhoe

Figure A-1 shows a backhoe excavator. This type of a mechanism is modeled as a kinematic "serial chain" shown in Figure A-2.

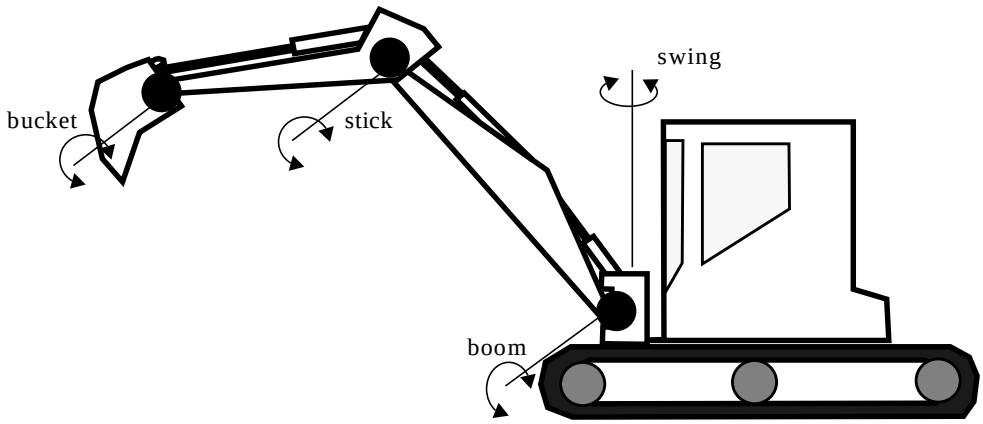


Figure A-1 A backhoe excavator.

Note that the linkage consists of three co-planar links (boom, stick, and bucket) that can rotate around a "swing" joint. During excavation, the swing joint is held fixed and the other three links move in a plane.

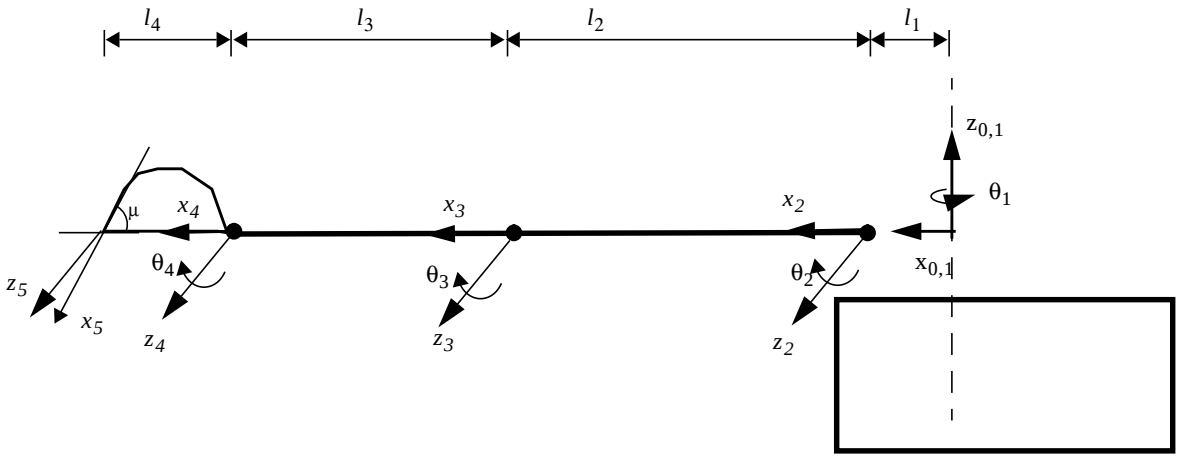


Figure A-2 Kinematic Model of a backhoe.

The backhoe can be modeled kinematically using methods that are commonly used for robotic manipulators. One method derives the composite transform that relates the joint displacements to positions and orientation of the end-effector [Denavit55].

The Denavit-Hartenburg parameters for the backhoe are:

Table A1-1 Denavit Hartenburg parameters for an excavator backhoe.

i	α_{i-1}	a_{i-1}	θ_i	d_i
1	0	0	θ_1	0
2	$\pi/2$	l_1	θ_2	0
3	0	l_2	θ_3	0
4	0	l_3	θ_4	0
5	0	l_4	μ	0

Next, we write the homogeneous transforms related to the D-H parameters as

(A1 1)

$$ {}^{0}_{1}T = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^{1}_{2}T = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & l_1 \\ 0 & 0 & -1 & 0 \\ \sin\theta_2 & \cos\theta_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{2}_{3}T = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & l_2 \\ \sin\theta_3 & \cos\theta_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^{3}_{4}T = \begin{bmatrix} \cos\theta_4 & -\sin\theta_4 & 0 & l_3 \\ \sin\theta_4 & \cos\theta_4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{4}_{5}T = \begin{bmatrix} \cos\mu & -\sin\mu & 0 & l_4 \\ \sin\mu & \cos\mu & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

A1-1.1 Kinematic Model of the T3 Robot

Figure A-3 shows a kinematic model of the robot manipulator that was used for the experiments described in this thesis.

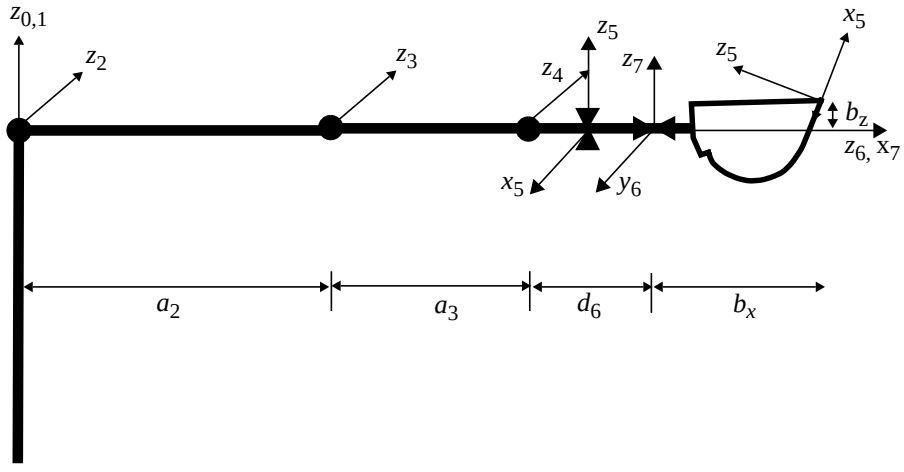


Figure A-3 A kinematic model of the T3 robot outfitted with an excavator bucket.

The Denavit-Hartenburg parameters for the T3 Robot are:

Table A1-2 Denavit Hartenburg parameters for the T3 robot.

i	α_{i-1}	a_{i-1}	θ_i	d_i
1	0	0	θ_1	0
2	$-\pi/2$	0	θ_2	0
3	0	a_2	θ_3	0
4	0	a_3	θ_4	0
5	$\pi/2$	a_4	θ_5	0
6	$-\pi/2$	0	θ_6	d_6
7	$\pi/2$	0	$\pi/2$	0
8	0	b_x	$-\mu$	b_z

The homogeneous transforms corresponding to the D-H parameters for the T3 robot are:

(A1 2)

$$
{}^{0}_{1}T = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^{1}_{2}T = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -\sin\theta_2 & -\cos\theta_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{2}_{3}T = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & a_2 \\ \sin\theta_3 & \cos\theta_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^{3}_{4}T = \begin{bmatrix} \cos\theta_4 & -\sin\theta_4 & 0 & a_3 \\ \sin\theta_4 & \cos\theta_4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{4}_{5}T = \begin{bmatrix} \cos\theta_5 & -\sin\theta_5 & 0 & a_4 \\ 0 & 0 & -1 & 0 \\ \sin\theta_5 & \cos\theta_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^{5}_{6}T = \begin{bmatrix} \cos\theta_6 & -\sin\theta_6 & 0 & 0 \\ -\sin\theta_6 & -\cos\theta_6 & 0 & d_6 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^{6}_{7}T = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^{7}_{8}T = \begin{bmatrix} \cos(\mu) & 0 & -\sin(\mu) & b_x \\ 0 & 1 & 0 & 0 \\ \sin(\mu) & 0 & \cos(\mu) & b_z \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

Note that during excavation, θ_5 and θ_6 are typically held constant (no rolling or yawing of the bucket) and the composite kinematic expressions can be simplified.

A1-1.2 Forward Kinematics for a Backhoe Excavator

Given joint angles, we would like to find the pose of the end-effector (tip of excavator bucket). The composite homogenous transform that relates the base frame to the end effector is:

(A1 3)

$$ {}^{0}_{5}T = \begin{bmatrix} C_1 \cdot C_{234\mu} & -C_1 \cdot S_{234\mu} & S_1 & C_1 (l_4 C_{234} + l_3 C_{23} + l_2 C_2 + l_1) \\ S_1 \cdot C_{234\mu} & -S_1 \cdot S_{234\mu} & -C_1 & S_1 (l_4 C_{234} + l_3 C_{23} + l_2 C_2 + l_1) \\ S_{234\mu} & C_{234\mu} & 0 & l_4 S_{234} + l_3 S_{23} + l_2 S_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

where $C_1 = \cos(\theta_1)$, $S_1 = \sin(\theta_1)$, $C_{23} = \cos(\theta_2 + \theta_3)$, $S_{23} = \sin(\theta_2 + \theta_3)$, $C_{234} = \cos(\theta_2 + \theta_3 + \theta_4)$ $S_{234} = \sin(\theta_2 + \theta_3 + \theta_4)$, $C_{234\mu} = \cos(\theta_2 + \theta_3 + \theta_4 + \mu)$, $S_{234\mu} = \sin(\theta_2 + \theta_3 + \theta_4 + \mu)$.

A1-1.3 Inverse Kinematics for a Backhoe Excavator

This section provides the inverse kinematics for the backhoe excavator. That is, given a pose of the end-effector, we would like to find the corresponding joint angles. Assuming roll, pitch, yaw convention, position and orientation of end effector will be specified by a matrix: (note: roll=0 for this mechanism, α: yaw, β: pitch, position: (p_x, p_y, p_z)

(A1 4)

$$ {}^{W}_{5}T = \begin{bmatrix} \cos\alpha \cdot \cos\beta & -\sin\alpha & \cos\alpha \cdot \sin\beta & P_x \\ \sin\alpha \cdot \cos\beta & \cos\alpha & \sin\alpha \cdot \sin\beta & P_y \\ -\sin\beta & 0 & \cos\beta & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Backing out the transforms that relate ${}^{0}T$ to ${}^{w}T$:

(A1 5)

$$ {}^{0}_{5}T = {}^{V}_{0}T^{-1} \cdot {}^{W}_{V}T^{-1} \cdot {}^{W}_{5}T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & r_{14} \\ r_{21} & r_{22} & r_{23} & r_{24} \\ r_{31} & r_{32} & r_{33} & r_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Equating terms between (A1 13) and (A1 15):

$$r_{13} = S_1 \qquad r_{23} = -C_1$$
$$r_{31} = S_{234\mu} \qquad r_{31} = C_{234\mu} \tag{A1 6}$$

then

$$\theta_1 = \text{atan2}\,(r_{13}, -r_{23}) \tag{A1 7}$$

$$\theta_{234} = \text{atan2}\,(r_{31}, r_{32}) - \mu \tag{A1 8}$$

Once θ_1 is known, it can be backed out of the composite forward kinematics.

$$ {}^1_5T = {}^0_1T^{-1} \cdot {}^0_5T = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{21} & q_{22} & q_{23} & q_{24} \\ q_{31} & q_{32} & q_{33} & q_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} C_{234\mu} & -S_{234\mu} & 0 & (l_4C_{234} + l_3C_{23} + l_2C_2 + l_1) \\ 0 & 0 & -1 & 0 \\ S_{234\mu} & C_{234\mu} & 0 & -l_4S_{234} - l_3S_{23} - l_2S_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A1 9}$$

Equating terms:

$$l_4C_{234} + l_3C_{23} + l_2C_2 + l_1 = q_{14}$$
$$l_4S_{234} + l_3S_{23} + l_2S_2 = q_{34} \tag{A1 10}$$

As a short hand, define the known parts of (A1 10)

$$x^* = q_{14} - l_4C_{234} - l_1 \qquad z^* = q_{34} - l_4S_{234} \tag{A1 11}$$

Then

$$l_3C_{23} + l_2C_2 = x^* \qquad l_3S_{23} + l_2S_2 = z^* \tag{A1 12}$$

$$C_3 = \frac{x^{*2} + z^{*2} - l_3^2 - l_2^2}{2l_2l_3} \tag{A1 13}$$

$$S_3 = \sqrt{1 - C_3^2} \tag{A1 14}$$

Two solutions are obtained for

$$\theta_3 = \text{atan2}\,(S_3, C_3)\,,\ \text{atan2}\,(-S_3, C_3) \tag{A1 15}$$

Since there are two values for θ_3, we have to use both in computing θ_2. With θ_2 known, (A1 11) can be rewritten in the form:

$$mC_2 - nS_2 = x^* \qquad mS_2 + nC_2 = z^* \tag{16}$$

where :

$$m = l_3 \cdot C_3 + l_2 \qquad n = l_3 \cdot \sin(\theta_3) \tag{A1 17}$$

After some manipulation,

$$\theta_2 = \text{atan2}(z^*, x^*) - \text{atan2}(n, m) \tag{A1 18}$$

$$\theta_4 = \theta_{234} - \theta_3 - \theta_2 \tag{A1 19}$$

For most excavating machines, it will be easy to pick θ_2 and θ_3 because one pair corresponds to an “elbow-up” configuration while the other corresponds to an “elbow-down” configuration and the latter is typically outside the range of motion.

A1-2 Kinematic model of a front-loader

The operation of a front-loader shown in Figure A-4 is significantly different from the that of the backhoe mechanism discussed in the previous section. While the base of the backhoe is stationary during excavation, the base of the front-loader is moved while loading. The modeling of such a device is more complicated since this mechanism is a “mobile manipulator”. Since this thesis has considered only two dimensional motion of the loader, it has been possible to model this mechanism as a P-R-R manipulator shown in Figure A-5. Note that while propulsion is modeled, steering is not

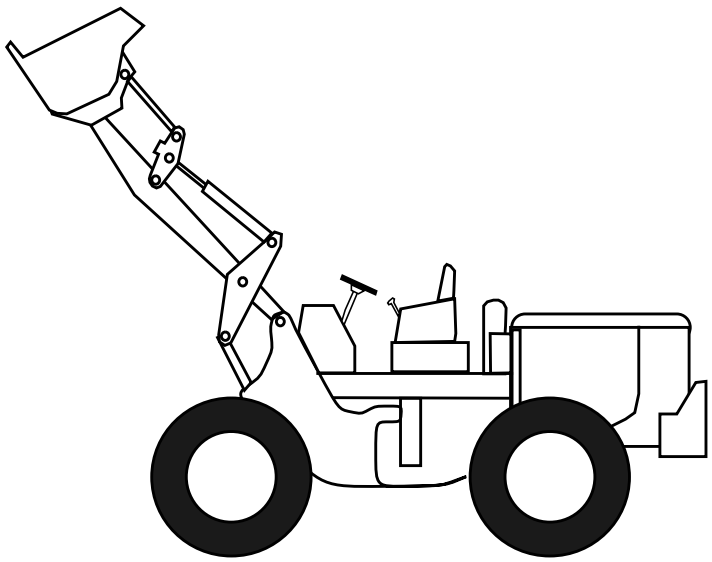


Figure A-4 A front-loader.

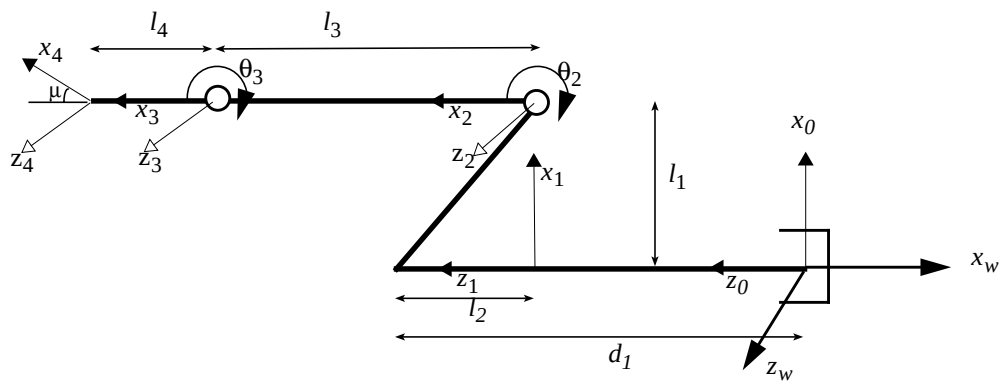


Figure A-5 Kinematic model of excavator in Figure A-4.

The Denavit-Hartenburg parameters for the front-loader are:

Table A1-3 Denavit Hartenburg parameters for a front loader.

i	α_{i-1}	a_{i-1}	θ_i	d_i
1	0	0	θ_1	$d_1 - l_2$
2	$\pi/2$	$-l_1$	$\pi+\theta_2$	0
3	0	l_3	θ_3	0
4	0	l_4	$-\mu$	0

The corresponding homogenous transforms are given by:

(A1 20)

$$ {}^0_1T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_1 - l_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^1_2T = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & l_1 \\ 0 & 0 & -1 & 0 \\ \sin\theta_2 & \cos\theta_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^2_3T = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & l_3 \\ \sin\theta_3 & \cos\theta_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^3_4T = \begin{bmatrix} \cos\mu & \sin\mu & 0 & l_4 \\ -\sin\mu & \cos\mu & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

In addition, there is an additional transform that relates ${}^{0}T$ to ${}^{w}T$:

(21)

$$ {}^{w}_{0}T = \begin{bmatrix} 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

A1-2.1 Forward Kinematics of a loader

The kinematic transformation that describes the position of the tip of the loader bucket with respect to the coordinate frame (x_0, z_0) given joint displacements (d_1, θ_2, θ_3) and excavator dimensions (l_1, l_2, l_3, l_4, μ) is:

(A1 22)

$$ {}^{w}_{4}T = \begin{bmatrix} -S_{23-\mu} & -C_{23-\mu} & 0 & -l_4 S_{23} - l_3 S_2 + l_2 - d_1 \\ C_{23-\mu} & -S_{23-\mu} & 0 & l_4 C_{23} + l_3 S_2 + l_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

A1-2.2 Inverse Kinematics for a loader

The joint displacements (d_1, θ_2, θ_3) are computed in a manner similar to the method used to compute inverse kinematics for the excavator. First, given the position (p_x, p_y) and orientation (θ) of the tip of the excavator, it is possible to write:

(A1 23)

$$ {}^{w}_{4}T = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & p_x \\ \sin\theta & \cos\theta & 0 & p_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Equating terms between (A1 22) and (A1 23):

(24)

$$\begin{aligned} -S_{23-\mu} &= \cos\theta \\ -C_{23-\mu} &= \sin\theta \\ px &= -l_4 S_{23} - l_3 S_2 + l_2 - d_1 \\ py &= l_4 C_{23} - l_3 C_2 + l_1 \end{aligned}$$

Then:

$$\theta_{23} = \text{atan2}\,(-\cos\theta, \sin\theta) + \mu \quad \textbf{(A1 25)}$$

Define

$$x^* = px - l_4 S_{23} - l_2 \quad \textbf{(A1 26)}$$

$$z^* = py + l_4 C_{23} - l_1 \quad \textbf{(A1 27)}$$

Then

$$C_2 = \frac{z^*}{l_3} \quad \textbf{(A1 28)}$$

$$S_3 = \sqrt{1 - C_2^{\,2}} \quad \textbf{(A1 29)}$$

Two solutions are obtained for

$$\theta_3 = \text{atan2}\,(S_2, C_2)\,,\ \text{atan2}\,(-S_2, C_2) \quad \textbf{(A1 30)}$$

Finally:

$$d_1 = -l_3 \sin\theta_2 - x^* \quad \textbf{(A1 31)}$$

A1-3 A fast lookup for inverse-kinematics

Often it is necessary to determine whether a pose of the end-effect is reachable. This computation is used very frequently for the reachability constraint. A fast lookup method has been implemented to determine reachability in the *x*-*z* plane. The method works as follows. The *z* dimension and the pitch angle axis, ρ, are divided into discrete intervals (1cm and 1 degree, respectively). For each (*z*, ρ) pair, a hashtable entry is created that holds the

minimum and maximum feasible values along the *x* dimension. To determine, if a candidate (x_c, z_c, ρ_c) is feasible, first the appropriate hashtable entry is found. The pose is reachable if

$$x_{min} < x_c < x_{max}$$

This method assumes that the workspace is always connected along the *x* dimension. Figure A-6 shows the workspace in the *x*-*z* plane for four pitch angles.

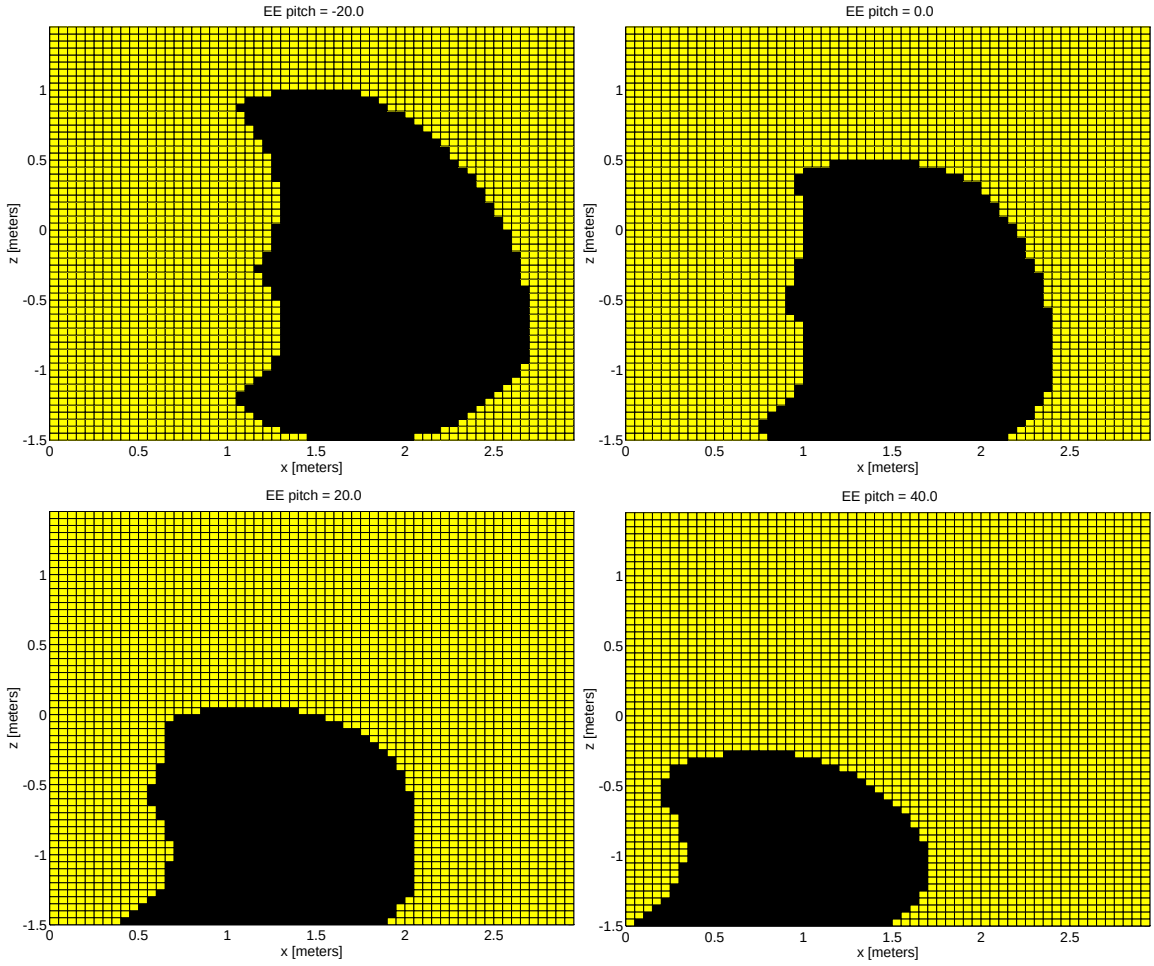


Figure A-6 Reachability in the *xz* plane for the T3 robot for four bucket pitch angles (-20, 0, 20, 40 degs). Dark regions are reachable.

Appendix 2 Dynamics

This appendix shows how the forces experienced at the end-effector are translated into joint torques. This is a static analysis. That is, no accelerations of soil or of the joints are considered. There are two components of the torques required to perform excavation— those due to gravitational forces on the robot's limbs and those due to resistive forces developed during excavation.

A2-1 Torques due to gravitational forces

The robot excavator has to generate torques simply to hold its limbs in place. These torques can vary immensely based on the configuration of the robot because of the varying moment arms involved. Consider for example the torques experienced at the boom joint due to gravitational forces as in Figure A-7:

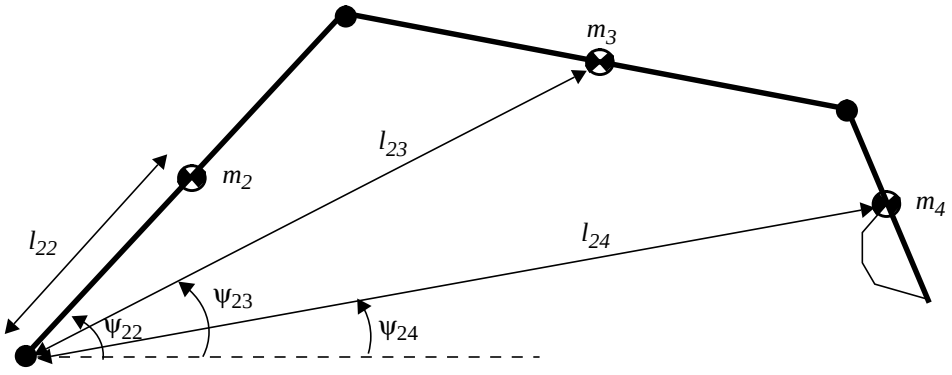


Figure A-7 The torques experienced at the boom joint due to gravitational forces.

This torque is written simply as:

$$\tau_{g_2} = g\,[l_{22}m_2\cos\psi_{22} + l_{23}m_3\cos\psi_{23} + l_{24}m_4\cos\psi_{24}] \qquad \text{(A2 1)}$$

Figure A-8 shows the moment arms that cause the torques at the stick joint. The torque is written as:

$$\tau_{g_3} = g\,[l_{33}m_3\cos\psi_{33} + l_{34}m_4\cos\psi_{34}] \qquad \text{(A2 2)}$$

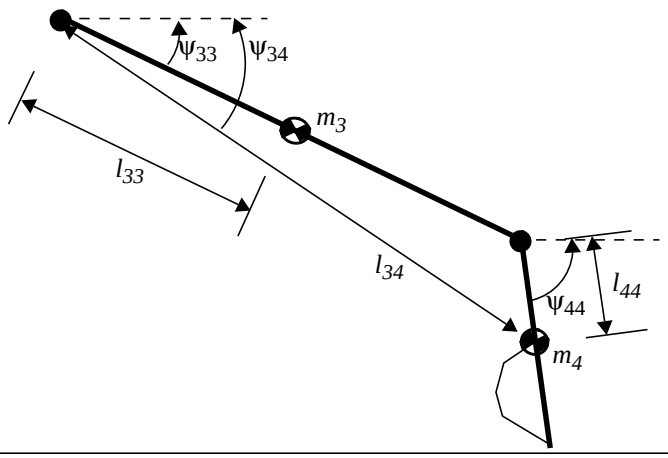


Figure A-8 The torques experienced at the shoulder joint due to gravitational forces.

Finally, the torque at the bucket joint is written as:

$$\tau_{g_4} = g l_{44} m_4 \cos\psi_{44} \quad \text{(A2 3)}$$

In general the torque felt at joint *i (i=2, 3, 4)* due to gravitational forces can be written as:

$$\tau_{g_i} = g \sum_{n=i}^{4} l_{in} m_n \cos\psi_{in} \quad \text{(A2 4)}$$

where l_{in} is the moment arm from joint *i* to center of mass of link *n*, m_n is the mass of link *n*, y_{in} is the angle that the moment arm makes with the horizontal and *g* is the acceleration due to gravity.

A2-2 Torques due to resistive forces

The joint torques due to resistive forces at the bucket tip are:

$$\underline{\tau}_{tip} = \underline{J}_m^T \cdot {}^0\underline{F}_0 \quad \text{(A2 5)}$$

where J_m^T is the transpose of the *manipulator jacobian* and ${}^0F_0$ is a vector $\left[f_x, f_y, f_z, m_x, m_y, m_z\right]^T$ of the end-effector forces (and torques) expressed in the base coordinate frame. Since forces are measured at the end-effector, these forces, ${}^E\underline{F}_E$, will need to be transformed into the base coordinate frame. This is done through the use of another jacobian known as the *frame jacobian* (J_f). While the manipulator jacobian transforms end-effector forces into torques (and vice versa), the frame jacobian transforms forces from one coordinate frame into another:

$${}^0\underline{F}_0 = \underline{J}_f^T \cdot {}^E\underline{F}_E \quad \text{(A2 6)}$$

The frame jacobian is given by:

(A2 7)

$$\underline{J}_f^{\mathrm{T}} = \begin{bmatrix} {}^0_E R & 0 \\ P \cdot {}^0_E R & {}^0_E R \end{bmatrix}$$

where ${}^0_E R$ is the rotation matrix in the homogenous transform ${}^0_E T$ and P is a matrix given by

(A2 8)

$$\begin{bmatrix} 0 & -p_z & p_y \\ p_z & 0 & -p_x \\ -p_y & p_z & 0 \end{bmatrix}$$

where p_x, p_y, and p_z are the position vector in ${}^0_E T$. Expanding (A2 7):

(A2 9)

$$\underline{J}_f^{\mathrm{T}} = \begin{bmatrix} \begin{bmatrix} C_{\mu-(234)} & 0 & -S_{\mu-(234)} \\ 0 & 1 & 0 \\ S_{\mu-(234)} & 0 & C_{\mu-(234)} \end{bmatrix} & 0 \\ \begin{bmatrix} 0 & J_{12} & 0 \\ J_{21} & 0 & J_{23} \\ 0 & J_{32} & 0 \end{bmatrix} & \begin{bmatrix} C_{\mu-(234)} & 0 & -S_{\mu-(234)} \\ 0 & 1 & 0 \\ S_{\mu-(234)} & 0 & C_{\mu-(234)} \end{bmatrix} \end{bmatrix}$$

where

$$J_{12} = l_2 S_2 + l_3 S_{23} + l_4 S_{234}$$
$$J_{21} = -l_4 S_\mu - l_3 S_{\mu-4} - l_2 S_{\mu-3-4}$$
$$J_{23} = -l_4 C_\mu - l_3 C_{\mu-4} - l_2 C_{\mu-3-4}$$
$$J_{32} = -l_2 C_2 - l_3 C_{23} - l_4 C_{234}$$

Under the assumption that ${}^E\underline{F}_E = \begin{bmatrix} f_x & 0 & f_z & 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}}$, that is the tip force is planar and measured at the bucket tip, it is possible to simplify (A2 6) to:

(A2 10)

$${}^0 f_x = C_{\mu-(234)}\, {}^E\underline{F}_{E_x} - S_{\mu-(234)}\, {}^E\underline{F}_{E_z}$$
$${}^0 f_z = S_{\mu-(234)}\, {}^E\underline{F}_{E_x} + C_{\mu-(234)}\, {}^E\underline{F}_{E_z}$$
$${}^0 m_y = (-l_4 S_\mu - l_3 S_{\mu-4} - l_2 S_{\mu-3-4})\, {}^E\underline{F}_{E_x} - (l_4 C_\mu + l_3 C_{\mu-4} + l_2 C_{\mu-3-4})\, {}^E\underline{F}_{E_z}$$

It remains to show how the manipulator jacobian (J_m) is computed. The general expression is:

$$J_m = \begin{bmatrix} \underline{J}_{L_1} & \underline{J}_{L_2} & \cdots & \underline{J}_{L_n} \\ \underline{J}_{A_1} & \underline{J}_{A_2} & \cdots & \underline{J}_{A_n} \end{bmatrix} \quad \textbf{(A2 11)}$$

where the J_{L_i} are vectors associated with linear velocities and the J_{A_i} are associated with angular velocities. Given that all the joints are rotational as opposed to prismatic, (A2 11) is written as:

$$\begin{bmatrix} J_{L_i} \\ J_{A_i} \end{bmatrix} = \begin{bmatrix} \underline{b}_{i-1} \times \underline{r}_{(i-1),e} \\ \underline{b}_{i-1} \end{bmatrix} \quad \textbf{(A2 12)}$$

where vectors $\underline{b}_{i-1}$ and $\underline{r}_{(i-1),e}$ are functions of joint displacements:

$$\underline{b}_{i-1} = {}^{0}_{i-1}R \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad \textbf{(A2 13)}$$

$$\underline{r}_{(i-1),e} = {}^{0}_{E}T \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} - {}^{0}_{i-1}T \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad \textbf{(A2 14)}$$

Expanding (A2 11):

$$\begin{bmatrix} J_{L_i} \\ J_{A_i} \end{bmatrix} = \begin{bmatrix} 0 & J_{12} & J_{13} & J_{14} \\ J_{21} & 0 & 0 & 0 \\ 0 & J_{32} & J_{33} & J_{34} \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix} \quad \textbf{(A2 15)}$$

where

$$J_{12} = -l_2 S_2 - l_3 S_{23} - l_4 S_{234}$$
$$J_{13} = -l_3 S_{23} - l_4 S_{234}$$
$$J_{14} = -l_4 S_{234}$$
$$J_{32} = -l_2 C_2 - l_3 C_{23} - l_4 C_{234}$$
$$J_{33} = -l_3 C_{23} - l_4 C_{234}$$
$$J_{34} = -l_4 C_{234}$$

Now it is possible to write an expression for the joint torques (boom, stick and bucket) due to the resistive forces encountered at the tip of the bucket by expanding (A2 5):

(A2 16)

$$\tau_2 = J_{12}{}^0 f_x + J_{32}{}^0 f_z + {}^0 m_y$$
$$\tau_3 = J_{13}{}^0 f_x + J_{33}{}^0 f_z + {}^0 m_y$$
$$\tau_4 = J_{14}{}^0 f_x + J_{34}{}^0 f_z + {}^0 m_y$$

References

[Alekseeva 85] Alekseeva, T. V. and Artem'ev, K. A. and Bromberg, A. A. and Voitsekhovskii, R. I. and Ul'yanov, N.A. *Machines for Earthmoving Work, Theory and Calculations,*. A. A. Balkema, Rotterdam, 1985.

[Atkeson 91] Atkeson, C. G. Using Locally Weighted Regression for Robot Learning. *Proceedings International Conference on Robotics and Automation*. April, 1991. early article about LWR. Compares CMAC and neural network to LWR.

[Bellman 62] Bellman, R. E., and Dreyfus, S. E. *Applied Dynamic Programming*. Princeton University Press, 1962.

[Bernold 89] Proceedings of One Day Seminar on Planetary Excavation. 1989. Editor L. E. Bernold.

[Bernold 91] Bernold, L. Experimental Studies on Mechanics of Lunar Excavation. *ACSE Journal of Aerospace Engineering*. 4(1), January, 1991.

[Bernold 93] Bernold, L. Motion and Path Control for Robotic Excavation. *ASCE Journal of Aerospace Engineering*. 6(1), January, 1993.

[Bernold 86] Bernold, L. E. Low Level Artificial Intelligence and Computer Simulation to Plan and Control Earthmoving Operations. *Proceedings ASCE Speciality conference on Earthmoving and Heavy Equipment*. Tempe, AZ, 1986.

[Bohachevsky 86] Bohachevsky and Johnson and Stein. Generalized Simulated Annealing for Function Optimization. *Technometrics*. August, 1986.

[Boles 90] Boles, W. W. *Lunar Base Construction*. Ph.D. thesis, University of Texas at Austin, December, 1990.

[Bradley 89] Bradley D. and Seward, D. and Bracewell, R. Control and Operational Strategies for Automatic Excavation. *Proceedings 5th International symposium on Automation and Robotics in Construction*. 1989.

[Bradley 93] Bradley, D.A. and Seward, D.W. and Mann, J.E. and Goodwin, M.R. Artificial intelligence in the control and operation of construction plant-the autonomous robot excavator. *Automation in Construction*. 2(3), 1993.

[Bradley 92] Bradley, D.A. and Seward, D.W. Artificial intelligence and ROBOTIC construction plant-the 'smart artisan. *IEE Colloquium on 'Artificial Intelligence in Civil Eng.* January, 1992.

[Brady 86] Brady, M. and Hollerbach, J. M. and Johnson, T. L. and Lozano-Perez, T. and Mason, M. (editor). *Robot Motion: Planning and Control*. MIT Press, 1986.

[Brost 88] Brost, R. C. Automatic Grasp Planning in the Presence of Uncertainty. *The International Journal of Robotics Research*. 7 1988.

[Bullock 88] Bullock, D. *Supervisory Control For Cognitive Excavation*. Master's thesis, Carnegie Mellon University, 1988.

[Bullock 90] Bullock, D. M. and Apte, S. and Oppenheim, I. J. Force and Geometry Constraints in Robot Excavation. *Proceedings Space 90: Engineering, Construction and Operations in Space*. ASCE, Albuquerque, 1990.

[Bullock 92] Bullock, Darcy M. and Oppenheim, Irving J. Object Oriented Programming in Robotics Research for Excavation. *Journal of Computing in Civil Engineering.* 6(3), July, 1992.

[Bullock 89] Bullock, Darcy M. and Oppenheim, Irving J. A Laboratory Study of Force-Cognitive Excavation. *Proceedings Sixth International Symposium on Automation and Robotics in Construction*. June, 1989.

[Burks 92] Burks, B.L. and Killough, S.M. and Thompson, D.H. Remote excavation using the telerobotic small emplacement excavator. *1992 Winter Meeting. International Conference on Fifty Years of Controlled Nuclear Chain Reaction: Past, Present and Future*, pages 559-60. American Nuclear Society, November, 1992.

[Chua 90] Chua, L. Global Optimization- A Naive Approach. *IEEE Transactions on Circuits and Systems.* July, 1990.

[Coulomb76] Coulomb, C. A, Essai sur une application des regles des maximis et minimis a quelques problems de statique relatifs a l'architecture, *Memoires de Mathematique et de Physique*, presentes a l'Acadamie royale des Sciences, par divers Savant, et lus dans les Assemblees, Paris, Vol. 7.

[Cleveland 88] Cleveland, W. S. and S. J. Devlin. Locally Weighted Regression: An Approach to Regression Analysis by Local Fitting. *Journal of American Statistical Association.* 83(403), 1988.

[Cundall 88] Cundall, P. Numerical experiments on localization in frictional materials. *Proceedings of the workshop on limit analysis and bifurcation theory.* University of Karlsruhe, February, 1988.

[Cundall 88b] Cundall, P. and Board, M. A microcomputer program for modeling large strain plasticity problems. *Numerical methods in geomechanics*. Balkema, Rotterdam, 1988.

[Daily 88] Daily, M., et al., "Autonomous Cross Country Navigation with the ALV," Proc. of the IEEE International Conference on Robotics and Automation, 1988.

[Denavit 55] Denavit, J. and Hartenberg, R. S., "A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices," Journal of Applied Mechanics, pp. 215-221, 1985

[Feiten 94] Feiten, W. and Bauer, R. Robust Obstacle Avoidance in Unknown & Cramped Environments. *Proceedings IEEE International Conference on Robotics and Automation*. San Diego, May, 1994.

[Feng 92] Feng, P. and Yang, Y. and Qi, Z. and Sun, S. Research on Control Method of Planning Level for Excavation Robot. *Proceedings 9th International Symposium on Automation and Robotics in Construction*. Tokyo, 1992.

[Franz 63] Franz Reuleaux. *The Kinematics of Machinery.* Dover, 1963.

[Friedman 84] Friedman, J. H. *SMART User's Guide.* Technical Report LCS 1, Stanford University, Laboratory for Computational Statistics, Stanford, CA, 1984.

[Friedman 91] Friedman, J. H. MultiVariate Adaptive Regression Splines (with Discussion). *The Annals of Statistics*. 191-141, 1991.

[Friedman 85] Friedman, J. H. *Classification and Multiple Regression Through Projection Pursuit*. Technical Report LCS 12, Stanford University, Laboratory for Computational Statistics, Stanford, CA, 1985.

[Friedman 88] Friedman, J. H. Fitting functions to noisy data in high dimensions. Wegman, Gantz and Miller(editors), *Proc., Twentieth Symposium on the Interface*. American Statistical Association, Alexandria, VA, 1988.

[Friedman 77] Friedman, J. H. and Bentley, J. L. and Finkel, R. A. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*. (3), 1977.

[Friedman 89] Friedman, J. H. and Silverman, B. W. Flexible parsimonious smoothing and additive modeling (with discussion). *TECHNOMETRICS*. 1989.

[Friedman 81] Friedman, J. H. and Stuetzle, W. Projection Pursuit Regression. *Journal of the American Statistical Association*. 76817-823, 1981.

[Gill 68] Gill, W. R. and Vanden Berg, G. E., *Agriculture Handbook*. Number 316: *Soil Dynamics in Tillage and Traction*. US Department of Agriculture, 1968.

[Gocho 92] Gocho, T. and Yamabe, N. and Hamaguchi, T. Automatic Wheel Loader in Asphalt Plant. *Proceedings the 9th International Symposium on Automation and Construction*. Tokyo, June, 1992.

[Hebert 89] Hebert, M. and Caillas, C. and Krotkov, E. and Kweon, A. S. and Kanade, T. Terrain mapping for a roving planetary explorer. *Proceedings of the IEEE Conference on Robotics and Automation*. Scottsdale, AZ, May, 1989.

[Hemami 92] Hemami, A. and Daneshmend, L. Force Analysis for Automation of the Loading Operation in an L-H-D Loader. *Proceedings of the IEEE Conference on Robotics and Automation*. Nice, France, May, 1992.

[Hendler 90] Hendler, J. and Tate, A. and Drummond, M. AI Planning: Systems and Techniques. *AI Magazine*. 11(2), 1990.

[Hettiaratchi 67] Hettiaratchi, D. R. P. and Reece, A. R. Symmetrical Three Dimensional Soil Failure. *Geotechnique*. 4(3), 1967.

[Hinton 92] Hinton, G. E. How neural networks learn from experience. *Scientific American*. 267(3), September, 1992.

[Hoffman 92] Hoffman, R. and Krotkov, E. Terrain Mapping for Long-Duration Autonomous Walking. *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*. Raleigh, NC, July, 1992.

[Hogan 85] Hogan, N. Impedance Control: An Approach to Manipulation. *Journal of Dynamic Systems, Measurement and Control*. 107 1985.

[Hogan 87] Hogan, N. Stable Execution of Contact Tasks Using Impedance Control. *Proceedings IEEE International Conference on Robotics and Automation*. 1987.

[Homma 92] Homma, K. and Arai, T. and Adachi, H. Soil Model for Automated Excavation Using Touch Sensed Data. *Proceedings 2nd International Symposium on Measurement and Control in Robotics*. November, 1992.

[Homma 90] Homma, K. and Nakamura, T. and Arai, T. and Adachi, H. Spatial image model for manipulation of shape variable objects and application to excavation. *Proceedings IEEE International Workshop on Intelligent Robots and Systems*. 1990.

[Huang 93] Huang, X. D. and Bernold, L. E. Robotic Rock Handling during Backhoe Excavation. *Automation and Robotics in Construction*. 1993.

[Huang 94] Huang, X. D. and Bernold, L. E. Control Model for Robotic Backhoe Excavation and Obstacle Handling. *Proceedings Robotics for Challenging Environments*. Albuquerque, 1994.

[Hwang 92] Hwang, Y.K. and Ahuja, N. Gross motion planning-a survey. *Computing Surveys*. 24(3):219-91, 1992.

[Jordan 90] Jordan, M. I, and Jacobs, R. A. Learning to Control an Unstable System with Forward Modeling. *Advances in Neural Information Sciences 2*Morgan Kaufmann, 1990.

[Kakuzen 88] Kakuzen, M. and Hirokazu, A. and Kimura, N. Automatic Control Systems for Construction Machinery. *Proceedings 5th International Symposium on Robotics in Construction*. Tokyo, 1988.

[Kelly 94] Kelly, A. *A Partial Analysis of the High Speed Autonomous Navigation Problem*. Technical Report CMU-RI-TR-94-16, Carnegie Mellon University, 1994.

[Kirk 70] Kirk, D. E. *Optimal Control Theory: An Introduction*. Prentice Hall, 1970.

[Koditschek 90] Koditschek, D. E. Task Encoding for Autonomous Machines: The Assembly Problem. *Sixth Yale Workshop on Adaptive and Learning Systems*. Yale University, New Haven, 1990.

[Koivo 92] Koivo, A. J. Controlling an Intelligent Excavator for Autonomous Digging in Difficult Ground. *Proceedings the 9th International Symposium on Automation and Construction*. Tokyo, June, 1992.

[Kojima 90] Kojima, Y. and Fukuda, J. and Kuramoto, S. and Sano, Y. and Akiba, N. Remote Control Type Excavating Robot for Deep Foundation. *Fujikura Technical Review*. 1990.

[Kraft 90] Force Feedback Excavator and Material Handling System. Kraft TeleRobotics Inc. Overland Park, Kansas, 1990.

[Krotkov 93] Krotkov, E. and Simmons, R. and Whittaker, W. Autonomous walking results with the Ambler hexapod planetary rover. *Proceedings of the International Conference on Intelligent Autonomous Systems IAS-3*. Feb, 1993.

[Krotkov 90] Krotkov, E. Active perception for legged locomotion: every step is an experiment. *Proceedings of the Fifth IEEE International Symposium on Intelligent Control*. September, 1990.

[Kweon 91] Kweon, I. S, and Hoffman, R., and Krotkov, E. *Experimental Characterization of the Perceptron Laser Rangefinder*. Technical Report CMU-RI-TR-91-1, Robotics Institute, Carnegie Mellon University, 1991.

[Lansky 86] Georgeff, Michael and Lansky, Amy (editor). *Reasoning about actions & plans: Proceedings of the 1986 workshop*. AAAI, 1986.

[Latombe 90] Latombe, J. C. *Robot Motion Planning*. Kluwer, 1990.

[Lengyel 90] Lengyel, J. and Reichert, M. and Donald, B. R. and Greenberg, D. P. Real Time Robot Motion Planning Using Rasterizing Computer Graphics Hardware. *Proceedings SIGGRAPH*. 1990.

[Lever 94] Lever, P. and Wang, F. and Chen, D. Intelligent Excavator Control for a Lunar Mining System. *Proceedings ASCE Conference on Robotics for Challenging Environments*. Albuquerque, February, 1994.

[Li 92] Li, X. and Moshell, J. M. *Physically based models of dynamic terrain in virtual environments (II).* Technical Report CS-TR-92-27, University of Central Florida, 1992.

[Li 92] Li, X. and Moshell, J. M. *Physically based models of dynamic terrain in virtual environments (I).* Technical Report CS-TR-92-26, University of Central Florida, 1992.

[Lozano 84] Lozano-Perez, T. and Mason, M. T. and Taylor, R. H. Automatic synthesis of fine motion strategies. *International Journal of Robotics Research*. 3(1), 1984.

[Luenberger 80] Luenberger, D. *Linear and NonLinear Programming*. Addison Wesley, 1980.

[Luth 65] Luth, H. J. and Wismer, R. D. Performance of Plain Soil Cutting Blades in Soil. *Transactions of the American Society of Agricultural Engineers*. 1965.

[Mason 86] Mason, M.T. and Brost, R.C. Automatic grasp planning: an operation space approach. *ACM 1986 Proceedings of the Fall Joint Computer Conference*. Dallas, TX, 1986.

[McKyes 77] McKyes, E. and Ali, O. S., The Cutting of Soil by Narrow Blades, *Journal of Terramechanics*, 14(2), 1977.

[Mckyes 85] McKyes, E. *Soil Cutting and Tillage.* Elsevier, 1985.

[Mclelland 88] McLelland, J.L. and Rumelhart, D. E. *Explorations in Parallel Distributed Processing.* MIT press, 1988.

[Miller 90] Miller, W. V. and Varnava, W. V. *Optimal Tuning of Heavy Equipment Motion Controllers*. Technical Report TR-940, Naval Civil Engineering Laboratory, NCEL, Port Huenme, California, 93043, 1990.

[Moore 94] Moore, A. *General Memory Based Learning, A Brief Explanation of the concept, the software and the C interface* Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA 15217, 1994.

[Moore 92a] Moore, A. W and Atkeson, C. G. An Investigation of Memory-based Function Approximators for Learning Control. 1992. Technical Report, MIT Artificial Intelligence Laboratory.

[Moore 92b] Moore, A. W. and D. J. Hill and M. P. Johnson. An Empirical Investigation of Brute Force to choose Features, Smoothers and Function Approximators. S. Hanson and S. Judd and T. Petsche (editor), *Computational Learning Theory and Natural Learning Systems, Volume 3*MIT Press, 1992.

[Moore 94] Moore, A. W. and M. S. Lee. Efficient Algorithms for Minimizing Cross Validation Error. W. W. Cohen and H. Hirsh (editor), *Proceedings of the 11th International Conference on Machine Learning*. Morgan Kaufmann, 1994.

[NASA 89] Report on the 90-day Study on Human Exploration of the Moon and Mars. 1989. Lyndon B. Johnson Space Center.

[Nease 92] Nease, A. D. Air Force construction automation/robotics. *Proceedings National Telesystems Conference*. May, 1992.

[Nedoredzov 92] Nedoredzov, I. Forces prediction of underwater soil cutting by excavating robots. *9th International Symposium on Automation and Construction*. Tokyo, June, 1992.

[Ober 83] Ober, G. *Operating Techniques for the Tractor Loader Backhoe*. Ober Publishing, 1983.

[Ostaja 89] Ostaja-Starzewski, M. and Skibiniewski, M. A Master Slave Manipulator for Excavation and Construction Tasks. *Robotics and Autonomous Systems*. 4 1988/89.

[Pontryagin 62] Pontryagin, L. S. et al. *The Mathematical Theory of Optimal Processes*. Interscience Publishers, Inc., New York, 1962.

[Press 88] Press, W. H and Flannery, B. P. and Teukolsky, S. A. and Vetterling, W. T. *Numerical Recipes in C*. Cambridge University Press, 1988.

[Puhl 92] Puhl, H. On the modeling of real sand piles. *Physica A*. 182(3), March, 1992.

[Reece 64] Reece, A. R. The fundamental equation of earth moving mechanics. *Proceedings of Institution of Mechanical Engineers*. 1964.

[Register 90] Register, B. *Proceedings of FY89 Workshop on Extraterrestrial Mining and Construction*. Technical Report LESC-27585, Lyndon B. Johnson Space Center, 1990.

[Romero 89] Romero-Lois, H. and Hendrickson, C. and Oppenheim, I. A Strategic Planner for Robot Excavation. *Proceedings Sixth International Symposium on Automation and Robotics in Construction*. June, 1989.

[Sakai 88] Sakai, T. and Cho, K. Operation System for Hydraulic Excavator for Deep Trench Works. *Proceedings 5th International Symposium on Robotics in Construction*. Tokyo, 1988.

[Sameshima 92] Sameshima, M. and Tozawa, S. Development of Auto Digging Controller for Construction Machine by Fuzzy Logic Control. *Proceedings of Conference Japanese Society of Mechanical Engineers*. 1992.

[Sanvido 83] Sanvido, V. *Productivity Improvement Programs in Construction*. Technical Report 273, Stanford University, 1983.

[Schall 94] Schall, S and Atkeson, C. G. Robot Juggling: Implementation of Memory Based Learning. *IEEE Control Systems*. 14(1), 1994.

[Schneider 93] Schneider, J. G., *High Dimension Action Spaces in Robot Skill Learning, Technical Report*, Department of Computer Science, University of Rochester, June 1993.

[Seward 88] Seward, D. and Bradley D. and Brasserie, R. The Development of Research Models for Automatic Excavation. *Proceedings 5th International symposium on Automation and Robotics in Construction*. 1988.

[Siemens 65] Siemens J. C. and Weber, J. A. and Thornburgh, T. H. Mechanics of soil as influenced by model tillage tools. *Transactions of the American Society of Agricultural Engineers*. 1965.

[Simmons 90] Simmons, R. and Lin, L. and Fedor, C. Autonomous Task Control for Mobile Robots. *IEEE International Symposium on Intelligent Control*. September, 1990.

[Simmons 94] Simmons, R., Structured Control For Autonomous Robots, IEEE Transactions on Robotics and Automation, 10(1), 1994.

[Singh 91] Singh, S. An Operation Space Approach to Robotic Excavation. *Proceedings IEEE Symposium on Intelligent Control*. Alexandria, August, 1991.

[Stentz 95] Stentz, A. and Hebert, M., "A Complete Navigation System for Goal Acquisition in Unknown Environments," Proc. of IROS '95, August 1995.

[Terzaghi 47] Terzaghi, K. *Theoretical soil mechanics*. Wiley, New York, 1947.

[Toups 90] Toups, L. D. and Herrera, A. *Planet Surface Systems Reference Architecture Description for the Lunar/Mars 90 Day Study Period*. Technical Report, Lockheed Engineering and Sciences Company, Inc., 1990.

[Vaha 91] Vaha, P.K. and Skibiniewski, M. J. and Koivo, A. J. Kinematics and Trajectory Planning for Robotic Excavation. *Proceedings ASCE Construction Congress II*. Cambridge, MA, 1991.

[Vaha 93a] Vaha, P.K. and Skibniewski, M. J. Dynamic Model of An Excavator. *ASCE Journal of Aerospace Engineering*. 6(2), April, 1993.

[Vaha 93b] Vaha, P.K. and Skibniewski, M. J. Cognitive Force Control of Excavators. *ASCE Journal of Aerospace Engineering*. 6(2), April, 1993.

[Whitcomb 91] Whitcomb, L. L. and Koditschek, D. E. Automatic Assembly Planning and Control via Potential Functions. *Proceedings IEEE/RSJ International Workshop on Intelligent Robots and Systems*. 1991.

[Whittaker 85] Whittaker, W. and Turkiyyah, G. and Bitz, F. and Balash, J. and Guzikowski, R. and Montgomery, B. and Akdogan, R. First Results in Automated Pipe Excavation. *Proceedings of the Second International Conference on Robotics in Construction*. Pittsburgh, PA, May, 1985.

[Wohlford 90] Wohlford, W.P.and Bode, B.D. and Griswold, F.D. New capability for remotely controlled excavation. *Proceedings 1990 Winter Meeting of the American Nuclear Society*, pages 628-9. American Nuclear Society, November, 1990.

[Wroth 84] Wroth, C. P. The interpretation of insitu soil tests. *Geotechnique*. 34(4), 1984.

[Yoshinada 92] Yoshinada, H. and Otsubo, K. Personal Communication. August, 1992. Komatsu Corporation.

[Young 77] Young, R. and Hanna, A. Finite element analysis of plane soil cutting. *Journal of Terramechanics*. 1977.

[Zelenin 86] Zelenin, A. N. and Balovnev, V. I. and Kerov, L. P. *Machines for moving the earth*. A. A. Balkema, Rotterdam, 1986.