An Operation Space Approach to Robotic Excavation

Sanjiv Singh

Robotics Institute
Carnegie Mellon University
Pittsburgh PA 15213

Abstract

We propose a methodology to automatically generate robot actions for tasks that cannot be abstracted into a geometrical basis because of complex interaction between the robot and the world. Our method considers such a task, robotic excavation, by seeking to satisfy constraints posed by the robot and the world to generate "plans" that optimize an arbitrary cost function. We discuss the constraints, the optimizing scheme and preliminary simulation results.

1. Introduction

We are seeking a method to automatically construct strategies for robots performing tasks in which interaction with the world is complex. Two traditional ways to automatically generate robot actions are to use symbolic planning, or alternatively, to use the well developed techniques of automatic control. The former approach most often assumes that relationships in the world are purely geometric- the planning proceeds at a level where the world is abstracted to symbols devoid of physical properties such as mass and force. On the other hand, most traditional control techniques require explicit modeling of the dynamics of the system that is to be controlled. Whether the model is known a priori, or built up through experience, the central idea is to be able to obtain "good" behavior by compensating for the dynamics of the plant. A major deficiency of control regimes is that they require explicit reference signals as input, that is, some other agent must specify "what" must be done. It is not clear how general robotic tasks can be couched in such a framework.

We are interested in generating these reference signals (plans) for robots working in domains where dynamics of the robot-world interaction cannot be ignored, but in fact are complex enough that it is not possible to find a closed form solution. One such domain that is both interesting and useful is that of robot excavation, that is, the automatic operation of a robotic mechanism to excavate soil.

We propose a method that is similar in concept to work by several researchers in domains ranging from planning paths to planning grasping operations. These works have made use of *configuration* and *operation* spaces as tools to help in constructing robot plans. A configuration space (c-space) is a space spanned by the variables that uniquely describe the geometric configuration of an object that is being manipulated. In path planning applications the robot itself is being manipulated, rather than manipulating something else, and hence the dimensions of the c-space are the position and orientation variables that uniquely describe the configuration of the robot. The c-space is divided into admissable and inadmissable regions based on an analysis of all possible configurations that a robot may attain. Inadmissable regions correspond to impossible configurations such as interpenetration of the robot and objects in the environment. The task of path planning is stated as finding a path between two points in c-space (denoting starting and final configurations) while avoiding the set of inadmissable configurations [Lozano-Perez] [Laumond] [Stentz] [Rimon].

Some researchers have used an operation space (o-space), an expanded space that includes essential control variables that uniquely specify task execution [Brost]. For example, in the squeeze grasping domain where the robot is trying to grasp an object, the position and orientation of the object forms the c-space, while the o-space additionally includes separate dimensions for control variables like *pushing direction*. While such a space is harder to construct, it allows the identification of regions that represent guaranteed execution of the task. That is, such a decomposition specifies bounds on the control variables that can be used to successfully accomplish the task.

We propose that the excavation task be posed in an o-space framework. Our o-space is comprised only of the control variables necessary to specify the task- each point in this space represents a complete plan for a single dig, independent of the mechanism that is to execute the plan[1]. It is not feasible to include dimensions that uniquely specify the state of the environment because the excavated material is deformable, and, would require a very large number of variables to represent. The robot and the environment impose constraints on this space, that is, some digging plans are not permitted either because the robot is not able to reach certain parts of the work-space, or, is not able to exert the necessary forces. The lay of the terrain also forces the exclusion of certain digging plans. The o-space volume satisfying all these constraints represents the space of all successful plans. We can further choose among the successful plans for those that will optimize a cost function. Once an optimal plan is chosen, the task parameters can be mapped back to the robot to produce a plan for the robot to play out.

In this paper, we first discuss the formulation of the excavation task in section 2. In Section 3 we discuss posting constraints in the operation space. In section 4 we discuss methods of finding optimal solutions. In Section 5 we discuss extension of the method to multi-step plans. In section 6, we discuss a soil model that was used to verify our digging plans and in section 7, we discuss preliminary simulation results.

2. Formulating the Task

Consider the task of loading loose soil from a banked pile with a shovel attached to the end effector of a robot manipulator. The robot A, perceives the environment E as a profile E'. The robot's task is to formulate a plan P, to alter the environment to achieve the goal state E* (figure 1).

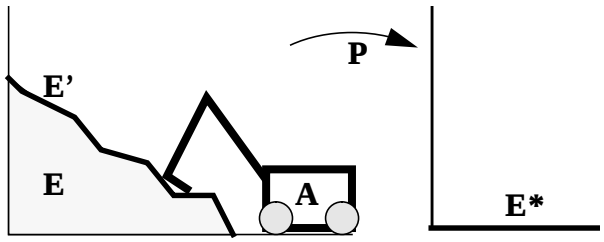

Figure 1: The Excavation Task

First, we parameterize the task such that the task variables span a large number of digging plans. In general we look for a minimal parameterization to span a set of plans sufficient to accomplish the task, and, selection of a "good" parameterization is achieved by insight into the task.

For now we will only concern ourselves with digging in a two dimensional world. We choose to parameterize the digging cycle by three parameters- approach angle, α, digging distance, *d*, and the height, *h*, at which the shovel penetrates the soil, as illustrated in figure 2. Once the shovel has entered the soil in a manner described by the three parameters, the soil is lifted straight up, excavating the triangular wedge also shown in figure 2.

As mentioned, the task variables (α, *d*, *h*) form a operation space. Every point in this space denotes a complete digging cycle. The proposed approach separates the o-space into admissable and inadmissable regions based on constraints posed by the robot and the environment.

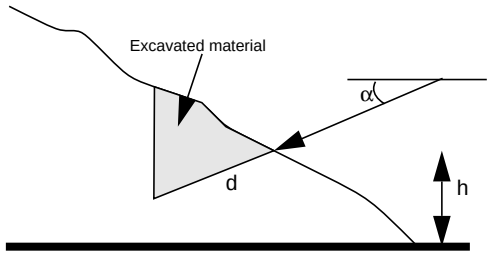

Figure 2: A 2D Excavation Scenario

The volume in the o-space that satisfies all the constraints is searched for a subset that will optimize some cost function like minimum time, minimum energy, or maximum payload.

2.1. Research Issues

While geometric constraints are not difficult to represent in the operation space, the constraints corresponding to robot dynamics (torques) and environment (shear strength and compaction) are difficult to represent meaningfully. This is chiefly because it is very difficult to a priori find a computationally feasible abstraction of the dynamics that also has validity in the physical world. However, it is possible to use approximate models to represent limits or maximal constraints that are imposed by the dynamical properties of the robot and the environment. It is conceivable that the model (the constraint surfaces) will have to be obtained by empirical data, that is, by experiments in which the robot excavator uses its experience in digging to build up

1. Some assumptions are made about the ability of the mechanism to execute certain types of motions.

parts of the model.

Once the required o-space volume has been completely specified, it is necessary to find a subset of the volume that optimizes an objective function- amount of soil excavated for a given dig, for example. However the constraints are often non-linear and not continuously differentiable and therefore standard non-linear programming techniques cannot be used directly. Also, in general there may be local optima which must be avoided.

Finally, there is a problem of finding a reasonable sequence of digs, that is, forming a multi-step plan that will not be grossly suboptimal.

2.2. Scope

In this paper we will concentrate on the kinematic and geometric constraints that can be posed in the excavation task described. We briefly discuss briefly how constraints due to robot dynamics and soil characteristics can be incorporated into the same framework.

3. Posting Constraints in O-space

We start by specifying constraints in the operation space.We consider three types of constraints that are due to the robot mechanism.

Volume Constraint: Given that the bucket can only hold a volume V_{max}, then an (α, *d*, *h*) triplet should not excavate more than this amount of soil.

Driving Distance Constraint: Given that the link of the robot that is to perform the digging is of fixed length, *l*, and that at best the robot can be positioned at the point where the soil tapers off, a constraint can be specified that places a limit on the parameter, *d*.

Approach Angle Constraint: Since the robot has a finite reach, the approach angles, α, will be limited for different values of, *h*, the height at which the shovel enters the pile.

The only geometric constraint posed by the environment that we consider is a limitation on the length of the digging distance, *d*, based on the height at which the soil is penetrated.

A typical o-space volume constructed for the constraints discussed above is shown in figure 3.

3.1. Constraints due to Robot Environment Interaction

If we assume that the robot is infinitely strong- that is it can muster any torque required, then the type of constraints discussed above are sufficient. More realistically, for robots with torque limits, we must consider the forces and torques required to accomplish digging. Studies in soil mechanics show that the relevant dominant characteristic of soil is shear strength. That is, for a particular soil, the amount of force required to “fail” a surface is directly proportional to its shear strength. Provided that the soil is homogenous, simple tests can be performed to determine its shear strength [Winterkorn]. A calculation can be made of the force necessary to fail a surface (or line in our 2D world) corresponding to every (α, *d*, *h*) triplet. The operation space can now be further constrained based on whether or not it is possible for the robot to generate the required forces.

Figure 3: operation space (α,*d, h*) volume with geometric environment and kinematic robot constraints

4. Search for an “Optimal” Dig

The subset of the o-space that meets all the constraints must be examined to find a point that is close to optimal with respect to some cost function. The optimal solution will lie on one of the constraint surfaces but, since the constraints are in general non-linear and not continuously differentiable, it will not be possible to solve for the optimum value in a closed form solution. Simplistically, we only need to find a starting point that is within the constrained volume and follow the gradient of the cost function to arrive at a region in the o-space that is “optimal”. For the case of constrained

optimization we will need to ensure that the search remains within the bounds of the constraints. This is typically accomplished by projecting the gradient on the constraint surface when the gradient points outside the constrained volume[Luenberger]. Alternatively, the objective function maybe modified to be artificially very low at the boundaries. Now the modified function is used with gradient descent. The problem with such approaches is that they are prone to local extrema in the objective function.

Recently, several results have shown that randomizing the search with narrowing the standard deviation of the randomization (exponentially with time) guarantees convergence to the global optimum value in the limit. This method, *Simulated Annealing*, is about the only one that guarantees convergence to global optima. A major advantage is that it does not require smoothly differentiable constraints and is able to deal with an arbitrary number of constraints. For our application we have found that simulated annealing is several orders of magnitude faster than exhaustive search of the constrained volume. While it is simple to implement, by itself, simulated annealing suffers from the problem that it might take an enormously large amount of time to converge to the optimum value [Bohachevsky]. We discuss the method in detail below.

4.1. Simulated Annealing

Simulated annealing is best explained by a 2D example shown in figure 6. Let the unshaded area be the admissible area (that meets all constraints). M denotes the optimal point with respect to an objective function *J*. Each trial starts at an initial point within the admissable area.

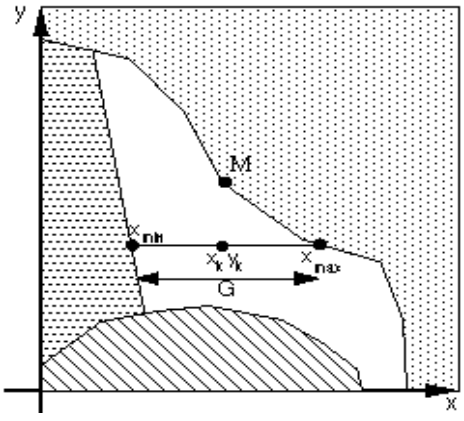

Figure 4: Moving towards the optimum point using simulated annealing

To do this, points could be picked randomly in the entire space till one that meets all the constraints is found. Below, we describe a procedure to iteratively converge to the global optimum in the admissable region.Let x_k, y_k be the solution at step *k*. The task is to iteratively proceed to **M**. The procedure is as given below.

1. x_{min} and x_{max} are found using an iterative line search. Let $G = x_{max} - x_{min}$.
2. Choose a random number r_1 in the interval [0, 1]. Let $x_{new} = x_{min} + r_1 * G$.
3. If the objective function at (x_{new}, y_k) is better than at (x_k, y_k), then the point (x_{new}, y_k) is accepted and the process is started over, this time looking in the *y* direction for a new point just as is it was for the *x* direction.
4. In case x_{new}, y_k does not improve the objective function, let $\delta J = J(x_{new}, y_k) - J(x_k, y_k)$. Define $\delta = e^{\delta J}/\ln(C*k)$. Pick another random number r_2 also in the interval [0, 1]. The rule is to accept x_{new}, y_k even though it does not increase the objective function if $r_2 < \delta$.
5. The procedure is repeated from step 1 for the y direction in order to complete the determination of x_{k+1}, y_{k+1}.
6. The whole procedure is repeated for a fixed number of iterations. Since the annealing might move towards a point that decreases the objective function, the point that has the maximum value of the objective function is continuously maintained.

Note that the *C* term in step 4 determines how fast the randomization dies out. For small values of *C* (close to zero) the randomization decreases very slowly while it dies out relatively quickly for large values of *C* (close to ∞). Since it is the randomization that allows this method to escape local optima, in practice this parameter is chosen by trial and error. In our work, we have chosen $C = 1$.

Experience with simulated annealing has revealed several interesting points. Firstly, in practical implementation, annealing can only be performed for a finite period of time, and, there is no guarantee of finding the global extrema. However, we can increase our chances by running several trials starting from random locations within the constrained volume. Secondly, increasing the number of iterations and trials helps but the marginal utility of increasing the number of trials

and iterations grows exponentially smaller. Lastly, gradient descent is useful in improving the objective function from the point that provides the largest value of the objective function at the time the annealing stops. This hybrid method accrues the advantage of being able to avoid local optima- it is very likely that the best point found by the annealing process will be close to the global optima. Since annealing takes a long time to converge on its own, gradient descent provides an efficient way of converging to the global optima.

Our current planner implements a 3D version of the algorithm described above to search in the (α, *d*, *h*) space.

5. Extension to Multi Step Plans

The method described above seeks to determine the parameters of a dig that would optimize the payload at every dig. However, attention to local optimality can cause problems in the long run. Figure 5 shows an undesirable configuration of the environment E that has been created by the robot. In this case, the robot will continue to maximize its payload at each dig by digging in the larger pile, until the volume it can reach is smaller than the volume that it can excavate from the small pile. Such a scheme is obviously short sighted and is bound to have low overall efficiency. We would like to obtain a more globally reasonable behavior.

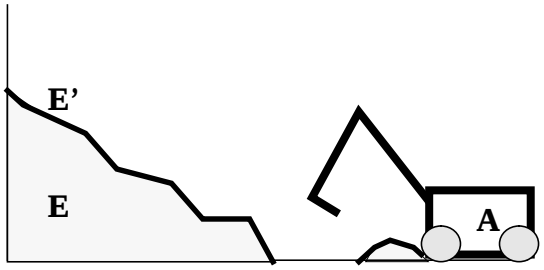

Figure 5: Undesirable Intermediate Configuration

There are two ways to deal with the problem outlined above. Firstly, it is possible to add an additional dimension in the simulated annealing process to represent temporal evolution of the excavation. This is equivalent to looking ahead by a number of digs before the choice of single dig is made. The extent to which the look ahead is done is typically a function of computational resource. However, if the task is well understood, it may be possible to use heuristics to add further constraints in the operation space. In this case, the newly introduced inadmissable regions correspond to those plans that will put the environment in an undesirable state. For example, we might stipulate that all plans that separate the pile, and those that leave the profile (E') with a greater curvature than some preset amount, should be excluded. Now the simulated annealing method proceeds as before with the addition of these new constraints. The major difference, however, is that consideration of these new constraints requires the forward simulation of soil model to verify whether or not a particular (α, *d*, *h*) triplet is admissable.

6. Soil Model

A soil model is needed to verify digging strategies. To this end we have implemented a soil model in which the terrain is represented as 2D grid cells [Homma]. In Homma's model, the 2-D world is tessellated into cells which are "occupied" to denote the existence of soil at that particular location. When some of the soil is removed, as after a dig has been performed, simple heuristics are used to calculate the new stable state of the soil. These heuristics are illustrated in figure 5.

.

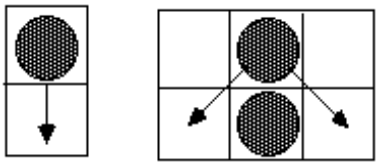

Figure 6: Heuristic Soil Model

In words, the heuristics can be stated thus:

- if the cell below the cell in consideration is empty, move a quanta of soil, downward.
- if the cell to the left of (or right of), one level below is empty, move to either of those locations with equal probability.

During simulation, a move is considered for every cell that is occupied. This process is iterated till no movement is possible and a stable state has been reached. It is worth noting that the above model approximates the behavior of a loose soil with no cohesive properties, like dry sand.

7. Preliminary Results

We are currently in the process of testing the method described above by simulating it with the soil model discussed in section 6. The simulation starts by determining the profile of the terrain, E'. Simulated annealing is performed to identify the dig parameters (α, *d*, *h*). Next, the volume of soil that would be excavated if

the dig is perfectly executed, is removed from the terrain model and a forward simulation of the soil model is used to predict the new state of the of soil. This cycle is repeated till all the soil has been removed.

So far, we have successfully tested a scheme that optimizes the payload at each dig. In a typical experiment, the simulation started off with 800 units of soil. The volume of the soil that can be held in the bucket is 25 units at maximum. 36 digs were necessary to remove all the soil, producing an average efficiency of roughly 88%. We have also experimented with a second scheme that uses heuristics to keep from choosing plans that would result in an undesirable configuration of the environment.While the results are not conclusive, preliminary results have shown that the efficiency is increased further by with the addition of heuristics.

Before each dig (in both types of experiments),5 trials of 100 iterations were performed to find the optimal dig parameters. Addition of the heuristics significantly affected execution time because for every step of the simulated annealing, it is sometimes necessary to run the forward soil model as many as 30 or 40 times. Execution times were approximately 5 secs (on a Sun 4) to find each set of optimal dig parameters, without the heuristics and roughly an order of magnitude higher with the heuristics. The current implementation is naive in that the simulated annealing starts from scratch after each dig. Execution time could be decreased by using the solution of one dig as the starting point for the annealing process for the next dig presuming that the o-space does not change drastically from one dig to the next.

8. Conclusions

The long term objective of our research is to be able to provide automatic excavation capability for various combinations of excavators and soil types. The technique discussed above is able to automatically determine task parameters for soil excavation given robot and environment constraints. An advantage of the proposed method is that an arbitrary number of constraints can be considered since we are not seeking a solution in terms of the constraints. It is possible to go straight from the analysis to optimization because the simulated annealing method uses constraint checking-a candidate point is legal simply if it meets all the constraints.

The ubiquitous dilemma of trading-off local versus global optimality shows up in the selection of dig parameters. We have used heuristics to keep the planner from choosing globally less efficient plans, at the cost of computational resource.

References

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

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

[Homma] K. Homma, T. Nakamura, T. Arai, H. Adachi, Spatial Image Model for Manipulation of Shape Variable Objects and Application to Excavation, In *Proceedings IEEE International Workshop of Intelligent Robots and Systems*, 1990.

[Laumond] J-P. Laumond, Finding Collision-Free Smooth Trajectories for a Non-Holonomic Mobile Robot. In *Proceedings IJCAI-87*, August 1987.

[Lozano-Perez] T. Lozano-Perez, Spatial Planning: A Configuration Space Approach. *IEEE Transactions on Computers*, February, 1983

[Luenberger] D. G. Luenberger, *Linear and Non-Linear Programming*, Addision-Wesley, 1984.

[Rimon] E. Rimon, D. E Koditschek, Exact Robot Navigation in Topologically Simple but Geometrically Complicated Environments, In *Proceedings IEEE International Conference on Robotics & Automation*, May 1990, Cincinnati.

[Stentz] A. Stentz, Multi-Resolution Constraint Modeling for Mobile Robot Planning, In *Vision and Navigation: The Carnegie Mellon Navlab*, Editor C. E. Thorpe, Kluwer Press, 1990.

[Winterkorn] H. Winterkorn & H. Fang, Editors, *Foundation Engineering Handbook*, Van Nostrand Reinhold Company, 1975.