

Constraint-Based Motion Planning in Deformable Environments

William B. Moss
wmoss@cs.unc.edu

Abstract—I present a novel algorithm for motion planning for a deformable robot in a deformable environment. Given the initial and goal configurations of the robot, I use a constraint-based planning approach to simulate deformations to the robot and the environment and compute a physically viable path to the goal. The algorithm takes into account standard geometric constraints like non-penetration and also considers physical constraints such as volume preservation.

I. INTRODUCTION

The classic motion planning problem considers a rigid robot planning a path through a static, rigid environment. Nevertheless, as the physical modeling of deformable objects has improved, developments in motion planning have followed to allow for motion planning with deformable robots and in deformable environments. There are many situations one can envision where the robot or the environment presented in a motion planning problem are deformable, however, the human body, especially the interior provides a particularly interesting example. Due to the highly deformable nature of the human interior and the need for exact solutions (due to the extremely high risk involved in damaging the interior of the body), it is poorly approximated using rigid models and deformable robots and environments become a necessity. By modeling the organs within the human body as deformable objects, we are able to plan paths that will cause the least damage for the hand or an organ as it is moved through the body cavity during surgery.

A. Issues

Due to the extremely large number of movable points in the environment (since each vertex of each deformable object must be considered one degree of freedom), standard motion planning algorithms that operate in C-obstacle space become completely intractable. Alternative planning frameworks that operate in the work space must therefore be explored.

B. My Approach

My main goal was to investigate the possibility of using a constraint based approach to plan in a deformable environment with a deformable robot. I also sought to do this using a finite element method to model the deformations. This allows for more efficient and accurate calculations of the stress on objects as well as the changes in volume. My proposed method uses a PRM to plan a guiding path in the environment, although any standard randomized motion planning algorithm could be used. I then use non-penetration and volume constraints to deform the robot when it comes into contact with an obstacle.

C. Outline

The rest of this paper is organized as follows. Section II presents related work, section III presents an overview of our algorithm and my results are presented in section IV. Finally I present some conclusions and ideas for future work in section V.

II. RELATED WORK

The primary problems facing motion planning with a deformable robot in a deformable environment are simulating the deformations in a physically realistic and efficient manner and developing a planning framework capable of handling the large number of degrees of freedom inherent in a deformable environment. In this section I will outline some of the previous work on these two problems.

A. Deformable Modeling

There has been a great deal of work in modeling deformable object in recent years, however, for our purposes, we will only consider a mass-spring system and the finite element method. These two methods are the most useful for our purposes since they both provide reasonable efficient and accurate simulations of the deformations and are commonly used throughout the literature. For a more complete survey of deformable modeling, please refer to [7].

1) *Mass-Spring Systems*: Mass-spring systems are, perhaps, the most common method for simulating the deformations of objects. In this framework, the object is discretized into masses that are connected to each other by springs. Although the coding of this method is simpler than that of the finite element method, a great deal of user tuning is often required to determine the optimal way to connect the masses with springs and to determine the function to map spring displacement to force.

2) *Finite Element Method*: Finite element methods (referred to as FEMs) have gained a great deal of traction lately due to their stability and their more solid grounding in physical and mathematical theory. An FEM, like a mass-spring system, discretizes the continuum of the object, however, it is usually done using tetrahedra. An equation of motion for the nodes of the tetrahedra can then be determined and the system is can be solved using an implicit solver on the entire system. It is important to note that FEM simulators, compared to mass-spring systems, excel at simulating large deformations.

B. Deformable Planning

The previous work in motion planning for deformable robots and environments falls into two categories, constraint-based motion planning and randomized roadmap planning.

1) *Constraint-Based Planning*: Constraint-based planning was first proposed by Garber and Lin [2] and provides an excellent framework for constraint based approaches. In this framework, the motion planning problem is reformulated as a series of constraints that represent the restrictions required by the motion planning problem. These restrictions are further classified into hard-constraints, which must be met at every time step, and soft constraints, which need not always be met. The hard constraints handle geometric constraints like non-penetration and are solved using a relaxation method at every time step. The soft constraints, such as obstacle avoidance and goal attraction, are implemented using penalty forces, and therefore provide only a guide to the robot.

This method was extended by Gayle, et al. [3] to allow for deformable robots in a rigid environment. To provide a global path for the robot and avoid local minima, the robot is simplified to a point and then a PRM is used to generate a roadmap. Using this roadmap as a guide, a path is found using the constraint-based framework. The robot is deformed by contact forces with the obstacles as necessary.

2) *Randomized Roadmap Planning*: Rodriguez, et al. [8] developed a framework for planning in fully deformable environments using an RRT. The method differs from standard RRTs in that the roadmap is generated in force space not c-obstacle space. Given an initial configuration and a goal, an RRT is grown from the initial configuration in the same manner as a standard RRT, however, each node in the tree represents the state of the entire system (i.e. the position and velocity of every movable vertex) and the edges represent a force being applied to the robot to transition it from the previous state to the new state. This allows them to keep track of the deformations of the robot and environment over time, since nodes deeper in the tree depend on the nodes above them. This would be impossible in c-obstacle space or using a PRM in force space. Their framework is able to solve a large number of problems; however, their runtime is somewhat slow, and the paths generated by an RRT tend to look jerky and unrealistic.

III. IMPLEMENTATION

My algorithm consists of two distinct stages: first, a PRM is used to generate a roadmap and second, the constraint based framework is used to generate a dynamic simulation. The second stage can be further broken down into the path following phase and the contact and deformation handling.

A. Roadmap Generation

This phase is important in the constraint-based framework because it provides a global path that helps the robot avoid the local minima inherent in a simple penalty-force based approach. The goal of this stage, just like a standard PRM, is to create a series of milestones in the environment and then to connect them, creating a graph. This graph can then

be traversed to determine a path from the start to the goal configuration. To incorporate the fact that both the robot and the environment can deform, I implemented a simple heuristic to shrink the obstacles that the PRM uses in planning and the robot is treated as a point. The heuristic takes into account the geometry of the obstacle so that a long skinny obstacle will be shrunk more in the long direction than the short and also scales the shrinking according to the Young's modulus of the object (a measure of stiffness).

This path can, of course, result in a milestone that requires the robot to be in collisions with an obstacle. If one of these milestones ends up in the path computed from the start to the goal, the robot and the environment will simply deform to accommodate the state.

One potential downside of this implementation is that it provides no guarantee that the amount of deformation (and therefore energy) is minimized on the path selected by the robot. There are various modifications to the heuristic that will be presented in the future work section that can help generate a path of minimal (or close to minimal) energy.

B. Path Following

The path following is handled by the application of an acceleration to all the vertices in the robot. At the creation of the robot a control point is selected (I used the point closest to the geometric center of the robot for simplicity) and I attempt to keep this point as close to the path as possible while exerting a force along the path. To do this, a maximum distance from the path is specified and if the robot is at or beyond that maximum distance, it feels an acceleration directly back towards the path. If it is already on the path, it feels an accelerations along the path and it feels a linear combination of those two forces for points in between. The maximum velocity of the robot is also bounded, scaling the acceleration that is applied if the velocity is too close to the maximum.

C. Contact Handling

The contact determination was done using a modified implementation of CULLIDE [9], which provides a set of potentially colliding triangles in each object and then an exact implementation of vertex-face collision for triangles in the potentially colliding set. Although I implemented edge-edge collision detection, it proved to be unnecessary given the density of the meshes used in simulation. Once a PCS is determined using CULLIDE and refined to a set of vertex-face intersections, the collisions could be handled either elastically or inelastically. In either case, the vertex and the face were first moved so there was no longer any interpenetration and then the velocities of the vertex and the three vertices of the face were adjusted depending on whether elastic or inelastic collision was being simulated.

D. Deformation Modeling

The deformations are handled using a finite element method using stiffness warping as presented in [6]. I will refer the reader to [6] for a more detailed explanation of using the finite

TABLE I
TIMINGS AND PROPERTIES FOR THE RESULTS

Name	Total Tets	Planning Time (s)	Frames per Second
Parallel Plates	376	0.7	8.1
Offset Plates	376	0.9	8.0
Spheres	1753	3.4	1.1

element method to simulate dynamic deformations, however, I will present a brief overview here. As with most dynamic systems, we are presented with an equation of motion for the system, which, in my case, manifests itself as

$$M\ddot{x} + C\dot{x} + K(x - x_0) = f_{ext}$$

where x_0 , x , \dot{x} and \ddot{x} are vectors of length $3n$ containing the initial position, current position, velocity and acceleration of all the vertices in the object, M is a diagonal matrix of the mass of each vertex, C is a damping matrix which we will assume is diagonal and dependent on a damping constant and the mass of the vertex, K is the stiffness matrix which will be discussed momentarily, and f_{ext} is a vector representing the external forces being exerted on each vertex in the object. The K matrix represents the coupling of the vertices in the object and is what ensures the volume and distance preservation of the vertices in the object. For simplicity and ease of implementation, I used the stiffness matrix available in the OpenTissue Library [10]. Once the matrices have been calculated, the system is solved implicitly by reformulating the equation of motion in terms of v_{i+1} and solving using a conjugate gradient solver [11].

IV. RESULTS

All the tests were run on a 2.0 GHz Intel Core2 Duo notebook with 2 Gb of memory running Windows Vista (see table I for timings). Animations for these tests can be found on the project website.¹

A. Parallel Plates

The obstacles in this scenario are two deformable plates placed an equal height above the robot (see figure 1). No external gravity is applied to the system, so all deformations are the result of contact between the robot and the obstacles. The goal state for the robot is directly above the two plates. The same setup is shown in figure 2, however, the relative stiffness of the robot and the obstacles are modified.

B. Offset Plates

The obstacles in this test are the same as the obstacles used in the previous example, except this time the plates are offset, with the right side plate higher than the left (see figure 3). The relative mass of the robot has been increased to display the effect of the elastic collisions on the simulation.

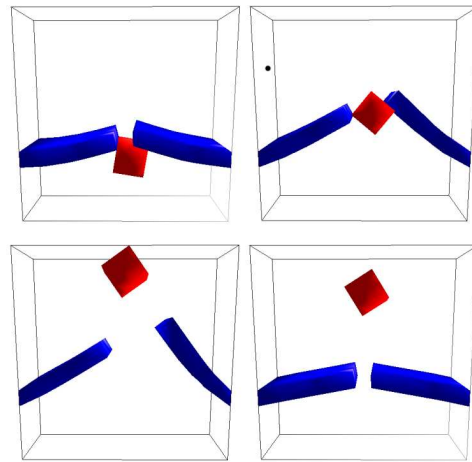


Fig. 1. Simulation of the robot planning a path between two deformable plates. The robot is much stiffer than the plates.

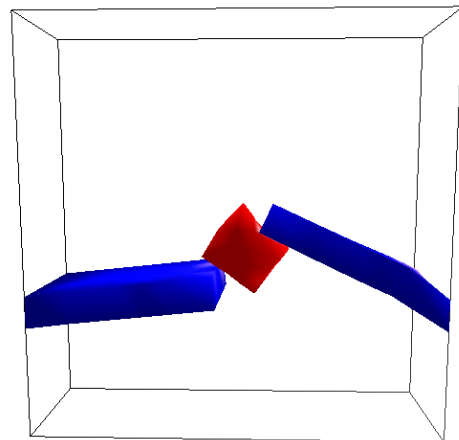


Fig. 2. Simulation of the robot planning a path between two deformable plates. The plates are much stiffer than the robot.

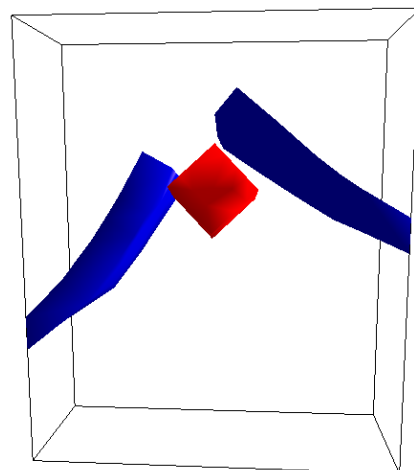


Fig. 3. Simulation of the robot planning a path between two deformable plates that are offset. The robot is much stiffer than the plates.

¹<http://cs.unc.edu/wmoss/courses/comp790-058/project/>

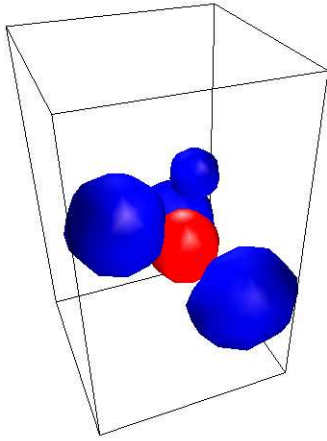


Fig. 4. Simulation of a spherical robot planning a path between four spheres. The robot and the spheres have the same stiffness and density.

C. Spheres

The obstacles in this test are four spheres of various sizes (see figure 4). Again, the robot must plan a path from its initial configuration to the top of the box. The robot and the spheres have the same stiffness and density.

V. CONCLUSIONS, ANALYSIS AND FUTURE WORK

In this paper I presented a new method for motion planning with a deformable robot in a deformable environment. A constraint-based planning approach was used in conjunction with a PRM planner to plan a path for the robot. Deformations were modeled using a finite element method and were handled by contact forces between the robot and the environment. I applied my algorithm to multiple scenarios with promising results.

Our approach does have some limitations, some of which I would like to resolve in future work and some of which are inherent in the constraint-based approach. Although we use a PRM planner to provide a high-level path for the robot, a constraint-based approach is still a local approach and therefore inherently struggles from the problem of local minima. The PRM planner goes a long way towards avoiding this problem, however, it is still not possible to guarantee that the robot will not get stuck in a local minimum. This planning method also does not necessarily provide the energy optimal path for the robot. A few possible heuristics were discussed above, however, without exploring more of the work space (like Rodriguez et al. do in [8]), I cannot guarantee anything about the energy expended by the robot.

There are a few things that I did not have time to complete that I feel would substantially improve the algorithm discussed in this paper. They deal primarily with enhancements to the PRM planner to provide a better high-level path for the robot. Currently the obstacles are shrunk based on their geometry and then scaled based on the Young's modulus, however, this process does not necessarily provide an outline

of where the obstacle can or cannot deform. Given that I have already implemented an FEM, it would be possible to simulate the maximum allowable deformations of an obstacle (given constraints on the change in volume and shear forces) and determine an outline of the immovable section of the obstacle. The PRM planner could then be provided this physically-based shrinking of the obstacle along with the original obstacle. Paths that pass through the immovable section would not be allowed and paths that pass through the original obstacle could be weighted more to encourage an lower energy path.

There are also ample other opportunities to expand this work in other directions. Of particular interest to me are applications in multiple robot planning and crowd simulation. In a multiple robot planning scenario, each robot could have a different set of constraints on the forces it is capable of applying and the deformations it is capable of withstanding. The lowest energy path could, therefore, require that one robot deform the environment so that the other robot can pass through. Crowd simulation could also be performed in a deformable or destroyable environment. Muller et al. present a fracture model for an FEM in [6] that could be incorporated into the system and then situations like crowds pushing against and possibly destroying a fence or other obstacles in the environment could be simulated.

VI. ACKNOWLEDGMENTS

I would like to thank Jason Sewall and especially Nico Galoppo for their help getting my FEM simulator running and stable. I would also like to thank Russ Gayle for his help and code base that provided an invaluable tool and reference for getting this project together.

REFERENCES

- [1] A. Witkin and D. Baraff, "Physically Based Modeling: Principles and Practice", ACM Press, 1997, course notes of ACM SIGGRAPH.
- [2] M. Garber and M. Lin, "Constraint-based motion planning using voronoi diagrams," Proc. Fifth International Workshop on Algorithmic Foundations of Robotics, 2002.
- [3] R. Gayle, M.C. Lin, and D. Manocha, "Constraint-Based Motion Planning of Deformable Robots," Proceedings of International Conference of Robotics and Automation (ICRA), 2005.
- [4] S.M. LaValle, Planning Algorithms, Cambridge University Press (or <http://msl.cs.uiuc.edu/planning/>), 2006.
- [5] M. Miller, J. Dorsey, L. McMillan, R. Jagnow, B. Cutler, "Stable real-time deformations," Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation, 2002.
- [6] M. Miller, M. Gross, "Interactive virtual materials," Proceedings of Graphics Interface, 2004.
- [7] A. Nealen, M. Miller, R. Keiser, E. Boxerman, M. Carlson, "Physically Based Deformable Models in Computer Graphics", Eurographics STAR, 2005.
- [8] S. Rodriguez, J. Lien, and N. Amato, "Planning motion in completely deformable environments," Proc. of IEEE Int. Conf. Robot. Autom. (ICRA), 2006.
- [9] N.K. Govindaraju, S. Redon, M.C. Lin and D. Manocha, "CULLIDE: interactive collision detection between complex models in large environments using graphics hardware," Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware, 2003.
- [10] OpenTissue, "Opensource Project, Generic algorithms and data structures for rapid development of interactive modeling and simulation," <http://www.opentissue.org>
- [11] J.R. Shewchuk, "An Introduction to the Conjugate Gradient Method Without the Agonizing Pain," 1994