Motion Planning for Manipulating Deformable Objects without Modelling

Final Project Report

Anand Parwal

RBE-550

Abstract— **The project involves developing, implementing and testing a motion planning algorithm to manipulate deformable objects that does not require modeling and simulating deformation. The method explains a different way of representing deformable objects that allows both physical and qualitative properties to be captured in an efficient representation This representation will allow formulation of a cost function that directly assesses the cost of deformation without expensive physical simulation or computation of deformed geometry. This cost of deformation is then incorporated with a modified optimal discrete planners for 3-dimensional path finding problem. The experiments are designed to perform a motion planning tasks for deformable object though obstacles. Even though experiments are conducted in Bullet physics simulator, note that the method does not have access to the model of the deformable object used by the simulator, although it's assumed that robot is able to sense the geometry of the object. A novel "True collision" checker for soft collision is also introduced.**

I. Introduction

The project involves implementing and improving upon a motion planning algorithm addressing the problem of computing motion plans for scenarios where the object being manipulated by the robot and/or the environment are deformable. This important set of problems arises in everyday environments, such as putting on clothing or cooking food, as well as in surgical and industrial settings. From folding a bed sheet in the home to tying suture in the operating room, the ability to manipulate deformable objects is an important capability for assistive and autonomous robots.

The main difficulty with manipulating deformable objects is that modelling and simulating their behavior is very hard as their motion is affected by large number of variables in contrast to rigid objects where system dynamics calculation is straightforward. Therefore, modeling such systems has received much attention from the research community. It is now possible to model known deformable objects in known environments [1] pretty accurately. However, online modeling of unknown deformable objects in new environments is still an open problem, though some methods can be used to acquire model parameters by probing the object [2]. Even if a perfect model is obtained of the deformable object, simulating that model is also very challenging.

Due to these challenges this project explores the possibility and efficacy of planning with deformation costs without explicitly modelling or simulating the deformation. It will be assumed that the objects under totally elastic deformation and that there is no motion of center of mass of the obstacles. Two types of deformation costs are defined and compared using a modified A* planner.

The report organization is as follows. Section II describes some of the relevant research in this field. The proposed framework is described in section III. Section IV and V explains the experiment conducted to test this framework and ultimately section VI discusses the outcome and concludes the report.

II. Related Work

The primary challenge of motion planning for deformable objects/environments is that deformability is very difficult to simulate accurately. Unlike rigid objects, whose dynamics are well-understood, the motion of a deformable object depends on a large and complex set of parameters that define its stiffness, friction, and volume preservation. Computing the geometry of a deformable object in contact with another object is particularly challenging, especially if both objects are deformable. Many earlier methods for deformable object simulation include mass-spring model simulation [1] but as the model overly simple, the accuracy suffers .The more accurate way to simulate is using finite element method [2][3]. Finite element method simulation, even though considered most accurate, could be very time consuming to compute for fine discretizations as it is highly sensitive to the discretization of the deformable object. Meshless models [4] have also been used in simulation methods.

Given the difficulties of deformable object simulation, it seems beneficial to explore the practicality of performing useful motion planning for deformable objects without explicitly simulating them. This was explored in [5] by moving twine which is fixed at one end and picking up the corner of a piece of cloth and folding it in collaboration with two manually controlled manipulators.

For this project the costs functions used in [6] were used as a starting point and then modified upon to achieve lower deformations. [9] uses a similar approach for planning in that it considers costs based on both, deformation and path length but is based on expensive and time consuming FEM. One major drawback of [6] was that in accommodating path containing deformations, true collisions were ignored, i.e. an object was allowed to pass through another. This project also attempts to overcome this serious flaw by implementing a check for true collisions. This is further explained in section III C.

III. Proposed Method

The following three subsections explain the proposed methodology to deal with planning with deformation. Firstly the environment was set-up to represent the soft body nature of the environment. Then based on this representation two deformation cost expressions were developed. And finally then an existing motion planning algorithm was modified to account for deformations and also to check for these true collisions.

A. *Environment Setup*

The environment and moving object are modelled using voxel grids as it is very suited for this representation. The elastic properties of the system are captured by two extra values per voxel:

- Deformability, *d*: representing how compressible the voxel is.
- Sensitivity, *s*: representing the penalty for deforming that particular voxel.

Deformability is a measure of the ability of the object to deform i.e. the stiffness of the object. So the softer the material is, the lower its d value. Sensitivity is used to allow the motion planner to avoid some objects more than others in the planning process, which is useful if different objects have different sensitivity to deformation. For example, during surgery, heart may be more sensitive to compression than lungs, even though both can be equally deformable. It can be set to infinity to prevent deformations in a particular object.

Following assumptions were made to represent this information in the environment:

- Objects are completely elastic i.e. the restore their original volume after a deformation when the deforming force is removed
- Environment objects (or fixed anchor points in the objects) do not move under application of any forces, although they may deform.

B. *Deformation Costs*

The cost of a given configuration of the object can be evaluated by computing a cost function that combines deformability and sensitivity into a single value. Cost is computed for each voxel of the object being deformed in collision, and the sum of these per-voxel costs represents the deformation cost of that configuration. If a collision is detected between object M and N, then the cost of deformation for a common voxel, let it be voxel i in object M and voxel j in object N, the sensitivity parameters for voxel i in object M is $s_i(M)$ while for voxel j in object N is $s_j(N)$. Similarly, $d_i(I)$ and $d_j(N)$ are the deformability parameters of voxels i and j. The direct deformation cost function can be formulated as follows

$$Direct\_Cost_i(M,N) = d_i(M) * s_i(M) + d_j(N) * s_j(N)$$

Please note that deformation costs is not the same as deformity value of voxels but includes both the deformability and sensitivity.

Another cost function can be formulated by considering a normalized deformation cost based on the total of two deformability values as

$$Norm\_Cost_i(M,N) = \frac{d_i(M) * s_i(M) + d_j(N) * s_j(N)}{d_i(M) * d_j(N)}$$

In Bullet physics simulation engine voxels are made of boxes. Therefore to mimic the custom voxel as described in the proposal, objects are created using btCompoundShape with boxes as the sub-shape (and adjacent boxes are coalesced to build an optimized set). The deformability and sensitivity values are stored independent of an objects geometry. Now after a collision is detected, the number of boxes (voxels) actually colliding, NC, is found using btCollisionShape and therefore the total cost of deformation is calculated as $NC * Cost$ where $Cost$ is either the direct cost or the normalized cost of collision of any one pixel from the two objects calculated using those independently store values.

This formulation puts some serious limitations like objects need to be uniformly deformable but this should not affect our end results as it is generally true.

C. *Path Planning and True Collisions*

The usefulness of these cost functions is demonstrated using a discrete planner by modifying A* to plan an optimal path (more aptly called Pareto-optimal [10]) considering both the deformation costs and the path length.

A* alone is sufficient for use in rigid environments in which states are either feasible and free of collision, or infeasible due to collision; however, A* is insufficient to handle planning in deformable environments with feasible collisions. Thus, fvalue function of A* algorithm is adapted to account for the deformation cost in addition to the path length cost as seen in algorithm 1.

The two costs are not merely combined but a provision is made to generate solution with different definitions of optimality. This control to assign dissimilar importance to either path length or deformation cost is provided by introducing a parameter *k*. Intuitively, for *k* close to 0 planner should neglect deformability and produce shorter paths with large deformation while a k value closer to 1 should produce longer but relatively deformation free paths.

In essence, *k*=0 will be a conventional A* considering the maximum deformed volume of the robot while *k*=1 will be a greedy best first search considering only the costs of deformation.

An interesting phenomenon observed while testing the algorithm in this state is that when all collisions are deemed feasible, it is possible for the planner to plan a path through an obstacle even if the opening isn't large enough to allow the object to pass through it under maximum deformation, if the path length costs to go around the obstacle was too high. Going back to the literature, it was found that this issue was completely ignored in the previous works [6] as the experiments that were conducted didn't account for this possibility.

A novel collision check was implemented in the planner that deemed the paths containing these "true collision" as infeasible. The maximum deformation volume of objects under collision was calculated as the product of its volume in free space and the deformability value, d. Then if a rigid body collision was detected for this volume using the previously computed path, it was termed as a "true collision" and those solutions were discarded.

IV. Experiments

The efficacy of this method is verified in a virtual environment using the Bullet physics simulator [7] and implementing the planner in OpenRAVE [8]. The object of interest is a cube of adjustable deformability. The environment set-up will n number of wall with different openings that will require the object of interest to deform in order to pass through them.

The proposed method is implemented and tested using 2 different experiments for a 3 dimensional motion planning problem. The environment used in the first experiment is rigid while the environment has different sensitivity and deformability values in the second. The above discussed modified A* is used as the planner for both the experiments.

A. Experiment 1

The first experiment is conducted to demonstrate the control provided by the parameter k .It involves a three walled set-up with openings at different locations on the walls as shown in Fig 1. The opening are 80% of the size of one face of the cube, which is shown in Fig 1 at its starting position, goal position being on the right of the third wall as seen in Fig 3e. The cube is modelled as a deformable object while the walls (obstacles) are rigid. Different paths are generated by for different *k* values. As no explicit limits are imposed on the workspace a deformation free solution exists which can be used for comparison. This solution is generated with *k*=1 which, in this case, is same as stimulating a fully rigid environment.

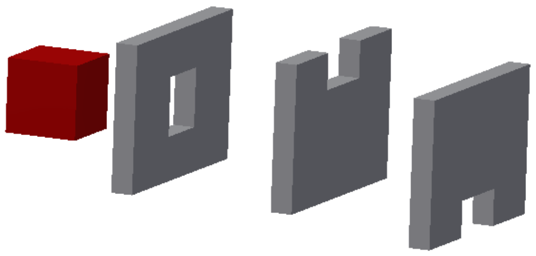

Fig. 1. Three walled setup of the first expeiment. The deformability and sensitivty of the object are taken as 0.5 while the walls are rigid bodies

B. Experiment 2

The second experiment is conducted to demonstrate the effect of deformability and sensitivity values. It involves two walls with three openings each that have the same deformability value (d=0.5) but different sensitivity values to indicate the different level of severity for deformation as shown in Fig 2. Similar to previous experiment, parameter *k* is used to control the nature of paths and start and goal positions are towards the left and right of the wall respectively. Workspace is restricted so as to avoid any deformation free path.

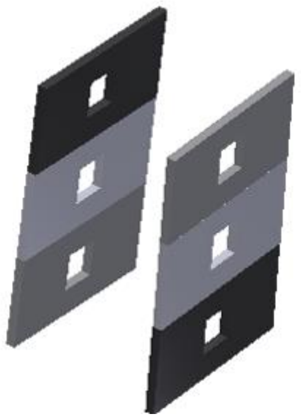

Fig. 2. Two walled setup of the seconf experiment. The wall are now deformaable with d = 0.5 and higher sensitivty represented by darker shade of grey. Lightest wall has s = 0.9, medium being s= 0.5 and darkest being the least sensitive to deformation with s = 0.1

V. Results And Discussions

For the first experiment, the planner was run for many different values of k from 0 to 1 two of which are shown in fig 3. The longest path is found for *k*=1 where only the deformation cost was considered while path length cost is ignored. When *k* is around 0.5 equal importance is given to deformation and path length costs. For *k* closer to zero, higher deformation path is computed. Table I provides the path length and deformation costs including direct and normalized costs.

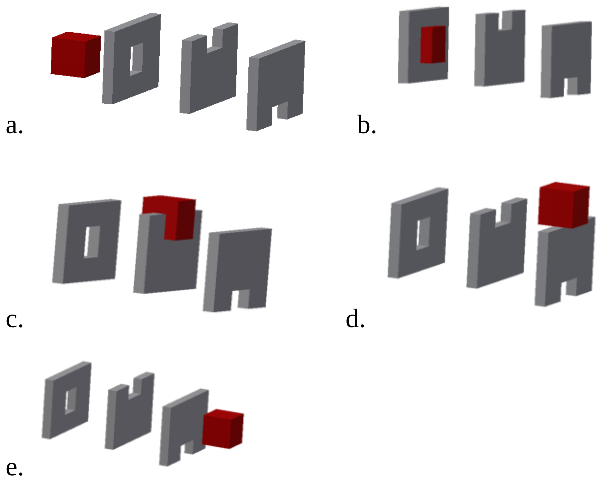

Fig. 3. **The path generated with k =0.1** : For lower k value the paths generated try to minimize path length than deformations costs.

TABLE I. RESULTS OF EXPERIMENT 1

k	*Path length*	*Deformation 1*	*Deformation 2*	*Planning time*
0.1	56	981	1062	21.73
0.5	58	746	837	21.85
0.9	61	33	35	22.13
1	73	0	0	14.30

[a] Deformation 1 is the direct cost while Deformation 2 is the normalized cost

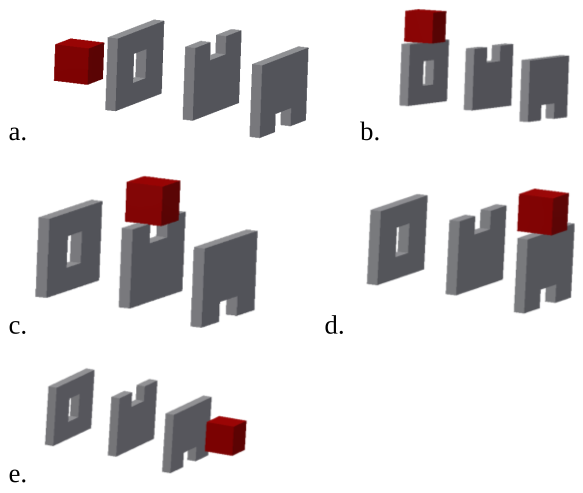

Fig. 4. **The path generated with k =0.9** : As k value is incresed the paths avoid collsions so as to minimize deformations.

For the second experiment, change in value of k resulted in drastically different paths. As the object was not allowed to go around the walls, the paths ended up passing through different sensitivity openings in the walls. For lower *k* values path goes through the lightest opening with max sensitivity giving the shortest paths. For *k* values around 0.5, path goes through medium sensitivity opening while for *k* values closer to one, longer paths were generated through lower sensitivity opening that minimizes the deformation costs

The control parameter *k* for the cost function worked as expected leading the planner to generate high deformation short paths for its lower values or lower deformation longer paths for its higher values.

Interestingly a steep jump in planning time was observed as *k* was changed from 1.0. This can be explained by considering that fact the as *k* value is reduced, previously infeasible paths also needed to be considered as some collisions suddenly became feasible.

VI. CONCLUSIONS

The proposed method of representing deformable objects that allows both physical and qualitative properties to be captured in a voxel-based representation is proved to successfully allow a planner to consider deformations. Using this representation, the designed a cost functions directly assesses the severity of deformation without expensive physical simulation or computation of deformed geometry. This cost function is particularly suitable for motion planning, and its application is demonstrated in a discrete motion planning problem with 3 dimensions. It is shown that the technique can generate paths that minimize deformation with either hard or soft robots in both hard and soft environments.

A novel “true collision” check was successfully implemented preventing solutions that allowed soft objects to pass through each other with high deformation costs. This was not foreseen in the project proposal as the previous research in modeless planning did not consider such cases.

While these experiments proved that deformable objects can be successfully manipulated without an expensive modelling based planner. The future works should include a study comparing the actual deformation using a soft body dynamics solver to the predicted deformation costs, providing more insight into the selection of deformation cost formulation. While deformability parameters can be derived from physical properties of an object, tuning sensitivity parameters is more complicated. Another good future work would be to investigate the automatic generation of sensitivity and deformability parameters.

ACKNOWLEDGMENT

This project was completed under the guidance of Prof Zhi Li for RBE-550 motion planning course. I would also like to thank Aditya Bhatt and Calder Phillips-Grafflin for their inputs and advice during the conceptualization of the project.

REFERENCES

[1] S. F. F. Gibson and B. Mirtich, “A survey of deformable modeling in computer graphics,” Mitsubishi Electric Research Laboratories, Tech. Rep., 1997.

[2] M. M¨uller, J. Dorsey, L. McMillan, R. Jagnow, and B. Cutler, “Stable real-time deformations,” in ACM SIGGRAPH/Eurographics Symposium on Computer Animation, ser. SCA ’02, New York, New York, USA, 2002

[3] G. Irving, J. Teran, and R. Fedkiw, “Invertible finite elements for robust simulation of large deformation,” in ACM SIGGRAPH/Eurographics Symposium on Computer Animation, ser. SCA ’04, New York, New York, USA, 2004.

[4] F. Faure, B. Gilles, G. Bousquet, and D. K. Pai, “Sparse meshless models of complex deformable solids,” ACM Transactions on Graphics, pp. 73–73, 2011.

[5] D. Berenson, "Manipulation of deformable objects without modeling and simulating deformation," 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, 2013, pp. 4525-4532.

[6] C. Phillips-Grafflin and D. Berenson, "A representation of deformable objects for motion planning with no physical simulation," 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, 2014, pp. 98-105.

[7] E. Coumans, “Bullet 2.73 Physics SDK Manual,” p. 47, 2010.

[8] R. Diankov and J. Kuffner, “Openrave: A planning architecture for autonomous robotics,” Robotics Institute, Pittsburgh, PA, Tech. Rep., 2008.

[9] B. Frank, C. Stachniss, N. Abdo, and W. Burgard, “Efficient motion planning for manipulation robots in environments with deformable objects,” in IEEE/RSJ International Conference on Intelligent Robots and Systems, Sep. 2011

[10] M. M¨uller, J. Dorsey, L. McMillan, R. Jagnow, and B. Cutler, “Stable real-time deformations,” in ACM SIGGRAPH/Eurographics Symposium on Computer Animation, ser. SCA ’02, New York, New York, USA, 2002.