Reactive Deformation Roadmaps: Motion Planning of Multiple Robots in Dynamic Environments

Russell Gayle Avneesh Sud Ming C. Lin Dinesh Manocha
Department of Computer Science
University of North Carolina at Chapel Hill
{rgayle,sud,lin,dm}@cs.unc.edu
http://gamma.cs.unc.edu/RDR (Video on website)

Abstract— **We present a novel algorithm for motion planning of multiple robots amongst dynamic obstacles. Our approach is based on a new roadmap representation that uses deformable links and dynamically retracts to capture the connectivity of the free space. We use Newtonian Physics and Hooke's Law to update the position of the milestones and deform the links in response to the motion of other robots and the obstacles. Based on this roadmap representation, we describe our planning algorithms that can compute collision-free paths for tens of robots in complex dynamic environments.**

I. Introduction

Systems with multiple robots are becoming increasingly important. In many real-world applications, these systems must also adapt and respond to changes in the environment. For example, in manufacturing or assembly lines, several robotic arms may need to avoid each other as well as other obstacles in the workspace to complete the product assembly. Self-reconfigurable robots can re-connect to provide a variety of functions and motion planning among individual modules is integral to re-connect independent sections to form a new shape. In animation or virtual environments, characters and objects can be treated as a multi-agent system where each agent avoids other agents as well as other moving obstacles in the environment. Environmental changes, such as damaged bridges or collapsed buildings, may force a team of robots to find a new path in a search and rescue mission. It is not uncommon to find applications with tens of robots, each with independent goals moving through dynamic environments.

The problem of motion planning among multiple robots has been investigated for more than two decades. It is well known that the complexity of this problem increases exponentially as a function of the number of robots and their degrees of freedom (DoFs). Thus, the cost of exact motion planning for multiple robots can be prohibitively high even for a relatively small number of robots.

Many planning algorithms for multiple robots relax the completeness requirement to compute a solution in a reasonable amount of time. The problem becomes harder when dealing with dynamic obstacles with no apriori motion information. Two broad-level techniques to handle such cases are *path modification* and *replanning* algorithms. The path modification techniques allow the robot to react to dynamic obstacles, but cannot cope with changes in the connectivity of the free space. On the other hand, replanning methods update the connectivity and compute new paths during execution. However, the cost of performing these updates can be relatively high and current algorithms do not scale well to a large number of robots.

Main Results: We present a novel motion planning data structure, the *Reactive Deforming Roadmap* (RDR), for multiple robots. Each link on the RDR can adapt, or deform, based on the motion of robots and obstacles the environment, similar to path modification algorithms. We compute an adaptive roadmap of deforming links. Our overall representation provides the capability to perform path maintenance at local and global levels, i.e. localized computations via link modifications and a global computation through roadmap maintenance. The modified links react to moving obstacles and remain in the free space. As a result, less connectivity information is lost by these modifications and the overall algorithm performs fewer global roadmap updates.

In order to plan among multiple robots, the basic motion algorithm computes a RDR for each robot. However, the cost of updating the RDRs can be expensive as the number of robots increase. An extension to the basic motion planning algorithm computes the motion of several robots using a single, global RDR. Overall, our approach automatically incorporates collision avoidance and reduces problems with respect to coordination. We have implemented our algorithm and highlight its performance on dynamic environments with tens of robots.

Organization: The remainder of this paper is organized as follows. Sec. II briefly reviews related work. Sec. III gives an overview of the our approach. We present the RDR framework in Sec. IV and show how it can be used for motion planning for multiple robots in Sec. V. Sec. VI describes the results and compare its performance with other multi-robot planners.

II. Related Work

In this section, we briefly describe techniques related to our work, with emphasis on motion planning for multiple robots. Since this problem shares similarities with motion planning among moving obstacles, we also present the relevant literature on planning in dynamic environments. For a more general survey of theoretical and practical results in motion planning, we refer readers to [1–3].

A. Roadmaps for Motion Planning

The roadmaps represent the connectivity of the free space for a given robot. The exact algorithms for roadmap computation have exponential complexity in the number of DoF of robots. Over the last decade, probabilistic algorithms that use random sampling techniques are widely used to build roadmaps for high-DOF robots [4, 5]. They work well in practice, though most of the literature in this area has mainly addressed the problem of motion planning for a single robot among static obstacles.

B. Motion Planning for Multiple Robots

Motion planning among multiple robots is a well-studied problem in the literature [6–9] The problem of motion planning for multiple robots can be roughly classified into two categories: *centralized* and *decoupled*.

The *centralized* methods consider the multiple robots as a single system, rather than independent entities. This approach makes it possible to perform global optimization and develop complete solutions. Some algorithms based on randomized sampling have been used to improve their performance [10]. Identifying and accommodating the groups in close proximity can improve the performance of these planners [11]. Other algorithms provide centralized solutions for specific cases, such as for pairs or triples of robots in a low density workspace [12].

The *decoupled* planners proceed in a distributed manner and coordination is often handled by exploring a *coordination space*, which represents the parameters along each specific robot path. Decoupled approaches are much faster than centralized methods, but may not be able to guarantee completeness. The simplest planners compute the trajectory of each robot independently [13] or use geometric based approaches to coordinate previously built trajectories [14]. Other decentralized planners include reactive style planners based on potential fields [15, 16], which can operate in real-time. However, they are susceptible to local minima or deadlock situations.

The centralized and decoupled planners have been applied to a number of situations such as flocking and shepherding [17, 18]. Most planners for multiple robots have been limited to workspaces consisting of only a few robots with independent goals.

C. Motion planning among Dynamic Obstacles

Many algorithms have been proposed for motion planning among dynamic obstacles and they are also applicable to motion planning with multiple robots. Some earlier algorithms make use of known obstacle trajectories [19], while other algorithms utilize graphics hardware to plan the motion of rigid robots moving in $\mathcal{R}^3$ [20].

1) Path and Roadmap Modification: Path modification methods allow a specified path to move or deform based upon obstacle motion to ensure a collision free path. Quinlan and Khatib proposed "Elastic Bands" to provide fast path modification for moving or otherwise dynamic obstacles [21]. Elastic bands represent the path as an elastic deforming structure. Brock and Khatib extended them to Elastic Strips [22]. In order to improve the efficiency, most of the computation is performed in the workspace rather than in the configuration space. However, by fixing the roadmap connectivity, both Elastic Bands and Elastic strips cannot respond to changes in free-space connectivity. Recently, Yang and Brock present Elastic Roadmaps, which is a modifiable roadmap for satisfying motion constraints for autonomous mobile manipulation [23]. Their work allows the milestones to move while efficiently recomputing the connectivity of the milestones during execution. Our work uses some of the ideas from Elastic Roadmaps in terms of preserving the connectivity of the roadmap.

2) Replanning: Rather than altering the path itself, many approaches replan, or rebuild the connectivity of the free configuration space at each step. The D* deterministic planning algorithm makes use of previous solutions rather than starting from scratch [24, 25].

Additionally, there have been several algorithms that perform a similar task for higher dimensional configuration spaces [26–30]. In general, these methods wait until the roadmap or robot's path becomes invalidated by obstacle motion.

III. Overview

In this section, we introduce the notation used in the paper and give an overview of our approach.

A. Notation

We address the problem of collision-free motion planning for multiple robots in a dynamic environment. Consider a set of n robots, $\mathcal{A}$, where each robot $A_i \in \mathcal{A}$ is a rigid or articulated body. We make no assumptions about the motion of A_i and the goal of each robot is totally independent of the other robots. We represent the workspace as $\mathcal{W}$ with possibly moving obstacles $\mathcal{O} = \{O_1, O_2, \ldots, O_m\}$, whose motion may not be known in advance. A configuration q^i in the configuration space $\mathcal{C}_i$ of robot A_i describes the position and orientation the robot. We use the symbol $\mathcal{F}_i(t)$ to denote the free space of A_i at time t. Similarly, $\mathcal{C}_i$-obstacle is the subset $\mathcal{C} \setminus \mathcal{F}_i(t)$. Given the initial and final configurations of the robots, our goal is to compute a collision-free path for each robot.

B. Reactive Deformation Roadmaps

Given a robot, A_i, we treat all the other robots as dynamic obstacles. As a result, the problem reduces to computing a collision-free path for each robot in a dynamic environment. As the obstacles and other robots undergo motion in the environment, we update the roadmaps so that they capture the connectivity of the free space. We present *Reactive Deformation Roadmaps* (RDRs), which is a dynamic data structure for motion planning of multiple robots in dynamic environments. A RDR dynamically captures the connectivity of the free space. The RDR itself is composed of *dynamic* *milestones* and *reactive links*. The reactive links are collision-free paths between two dynamic milestones. As the milestones associated with a link move, that link deforms based on energy minimization and thereby reacts to the motion of the dynamic obstacles and other robots. (See Fig. 1).

We use motion equations based on Newtonian physics and Hooke's Law to determine how the milestones move and the links deform. Internally, the entire RDR (including each link) is represented as a particle system. We use internal spring forces to prevent the particles from drifting apart while external repulsive forces due to the obstacles and other robots are applied to the system. In this sense, RDRs could also be viewed as a physically-inspired roadmap whose motion is determined by elastic forces due to path-length minimization and an artificial potential function. We also add new milestones and links, as well as deleting some of the milestones and links to dynamically capture the connectivity of the free space. More details are given in Section 4.

C. Motion Planning with RDRs

The RDRs are used for motion planning in the following manner. In the case of a single robot, our planner proceeds in a manner similar to other planners. First, the robot's starting and final configurations are connected to the roadmap. Then, a path is computed via a graph search algorithm such as A*. As the milestones and links along the path move and "deform" the roadmap, we recompute a new path. Our search algorithm also takes into account additions and deletions of links for path computation in a dynamic environment (see Section 5). In case of multiple robots, we may need to compute and update a separate RDR for each robot. In case, a group of robots have the same shape and configuration space, we only maintain a single RDR for each such group.

IV. Reactive Deforming Roadmaps

In this section, we give details on building RDRs and modifying them based on the motion of obstacles or other robots.

A. Reactive Deforming Roadmaps

An RDR, $\mathcal{R}$, as represented as a set of dynamic milestones, $\mathcal{V}$, and reactive links, $\mathcal{E}$ between the milestones, $\mathcal{R} = (\mathcal{V}, \mathcal{E})$. The milestones and links move as the free space of the robot changes over time. Each dynamic milestone is associated with a particle: a point in $\mathcal{C}$-space with a mass to which forces can be applied. The state of a particle i consisting of its position, $\mathbf{x_i}(t)$ and its velocity, $\mathbf{v_i}(t)$, at time t can be defined as

$$\mathbf{p_i}(t) = \begin{pmatrix} \mathbf{x_i}(t) \\ \mathbf{v_i}(t) \end{pmatrix}.$$

Similarly, reactive links are represented as a sequence of these particles, placed at equal distances along an initial straight-line link. The spacing, or density, of the particles along a link is specified in advance. This approach allows the link between connected milestones to behave as a short reactive path. These elements are shown in Fig. 1. In order for the RDR to be valid, the straight-line path between the connected particles must be valid, i.e. lie in the free space.

The RDR is initialized using well-known algorithms for constructing a roadmap based on random sampling in the configuration space. In order to deform the roadmap, we apply two types of forces; *internal* restoring forces and *external* reacting forces. The internal forces are spring-like or elastic such that the links attempt to stay in a state of minimum energy. This constraint helps us minimizing the jittery motion and provides smoother and shorter paths. The external forces are generated in a manner similar to that of potential field methods [15] and are used to retract the roadmaps as they react to dynamic obstacles and other robots in the scene. The total force acting on mass i is given as;

$$\mathbf{F}_i = \mathbf{F}_i^{int} + \mathbf{F}_i^{ext}$$

Based on these forces, the algorithm repositions the affected particles (or milestones) and thereby update the state of the roadmap.

1) Internal Forces: The internal restoring forces are necessary to prevent the particles on the same link from moving unnecessarily far apart during the deformation and also to prevent the roadmap from drifting too far from its inital configuration. The force on particle p_i from a connected particle p_j is given as:

$$\mathbf{F}_{ij}^{int} = -k_s(\|\mathbf{x}_{ij}\| - L_{ij})\frac{\mathbf{x}_{ij}}{\|\mathbf{x}_{ij}\|},$$

where k_s is a spring constant, $\mathbf{x}_{ij}$ is the vector between particle p_i and particle p_j, and L_{ij} is the initial separation distance between the two particles. An equal and opposite force is applied to particle p_j.

2) External Forces: As robots or dynamic obstacles move through the environment, the RDR responds to the changes in the connectivity of the free space by moving the dynamic milestones and deforming the reactive links. External forces from other moving robots and dynamic obstacles are applied to the particles. Given the particle p_i, the force from obstacle O_j is

$$\mathbf{F}_{ij}^{ext} = \frac{b}{d(p_i, O_j)^2 + \epsilon}, \tag{1}$$

where b is a constant. We set $\epsilon > 0$ so that the force due to the obstacle O_j does not cause numerical issues and it helps to improve stability of the system.

3) Roadmap Deformation: Once the forces on each particles have been determined, the state of the roadmap is updated.

A complete dynamics computation is inappropriate since it can capture all the transient states, including unwanted oscillation, in the reactive links. Instead, we preform a quasi-static simulation by using a variant of forward Euler integration that considers each particles to be at rest throughout the time step. For each particle i, we update its position and velocity as follows.

$$\mathbf{x_i}(t + dt) = \mathbf{x_i}(t) + \frac{1}{2}\mathbf{F}_i^{net}(dt)^2.$$

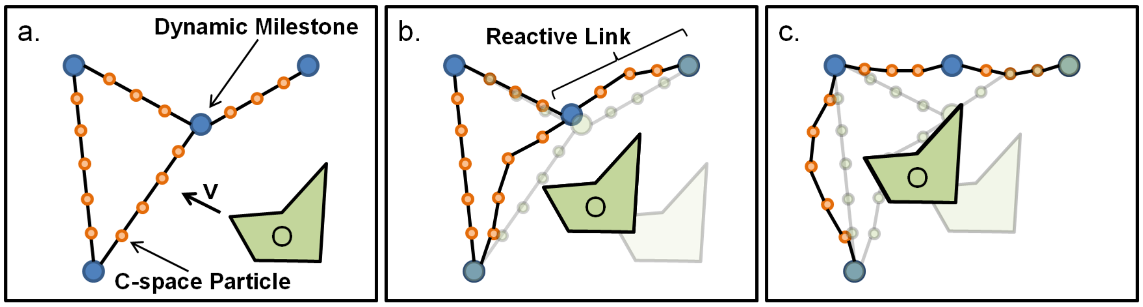

Fig. 1. Reactive Deforming Roadmaps (RDR): The RDR contains a set of dynamic milestones and reactive links. Each of these are represented with $\mathcal{C}$-space particles; a point mass in the configuration space. As the obstacle O moves, the dynamic milestones move as well and the reactive links of the roadmap deform to avoid the obstacle boundary. (c) If a path link deforms too much or is too close to the obstacle O, the link is removed.

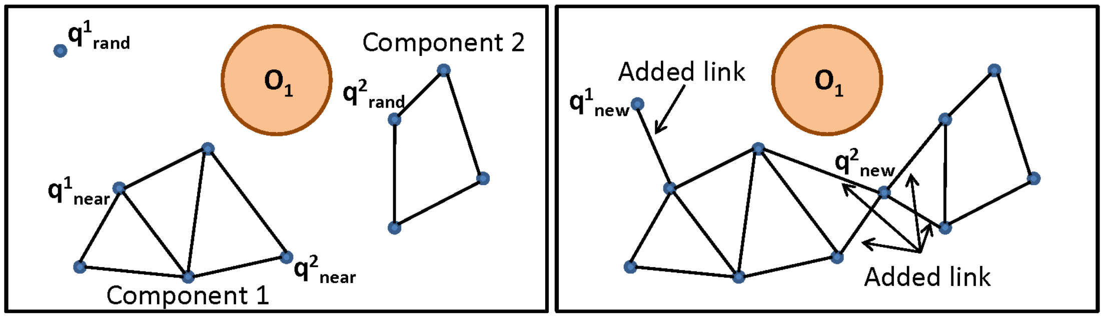

Fig. 2. **Adding links to the RDR**: (Left) To explore, a random sample q^1_{rand} is generated, and its nearest neighbor to the roadmap is found, q^1_{near}. To merge different connected components, a random milestone, q^2_{rand} on Component 2 is selected and its nearest neighbor on Component 1 is found, q^2_{near}. (Right) New milestones, q^i_{new} are added by extending q^i_{near} toward q^i_{rand} and valid straight-line links are added.

Since this method is based on an explicit Euler integration scheme, it is susceptible to numerical issues and instabilities. Many of these issues are reduced by bounding the magnitude of the potential functions. Furthermore, our potential energy based link removal method is tuned to remove links before they become unstable.

B. Global Updates

The local changes may not be sufficient. For instance, it is possible that the motion of an obstacle completely invalidates some current link(s) of the roadmap. In such a case, the link becomes invalid and needs to be removed. At the same time, we need to add new links or milestones to capture the connectivity of the freespace.

1) Link Removal: Link removal is essential for maintaining a valid roadmap and plays a key role in its resulting behavior. If links are removed too frequently, then we may have rather small deformation in the links. On the other hand, if the links are allowed to deform considerably, this may cause problems with the numerical stability of the simulation or may unnecessarily increase the path lengths of some links. We use two properties to evaluate the links for removal in our framework: amount of deformation and proximity to the obstacles.

For the link i, the extent of deformation of a link can be measured through its potential energy,

$$E_i(t) = \sum_{s_j \in l_i} \frac{k_s^j}{2}(d_j - L_j)^2,$$

where k_s^j is the spring constant for spring j, d_j is its current length, and L_j is its rest length when unstretched or compressed. When $E_i(t) > \tau_s$ for some user specified potential threshold τ_s, the link is removed and the milestones incident to the link may no longer be connected. For example, the link in Fig. 1(c) is removed as it needs to be considerably deformed due to obstacle O.

In order to account for proximity to any obstacle or other robot, the minimum separation distance (in workspace) is computed from a path link to the nearby obstacles. If this distance value is less than a pre-defined proximity threshold τ_d, then the link is removed. Specifically, τ_d is chosen such that the link remains at a safe distance from the obstacle, for instance, the radius of the bounding sphere for rigid robots.

2) Adding Links and Milestones: The process of removing links can result in losing connectivity information about $\mathcal{F}$. Moreover, as the obstacles move, we may need to add new links or milestones to capture the connectivity. Our framework uses three methods for this computation.

In the first method, a list of removed links is maintained. Periodically, or when no path can be found for a robot, the straight-line path in the configuration space for some or all of the links in this list is tested for validity. If they are valid, a reactive link is added in its place.

The second approach tries to connect various disjoint components of the roadmap by using a construction algorithm similar to that used in the RRT algorithm [3]. Given a component C_i, for each other component C_j, a random milestone $q_j \in C_j$ is chosen. Then, the nearest neighbor $q_{near} \in C_i$ to q_j is computed. Next, a new milestone, q_{new}, whose configuration extends q_{near} toward q_j is computed such that it has a valid straight-line path to q_{near} in the configuration space. Finally, straight-line paths from q_{new} and the nearest neighbors in C_i and C_j are connected if they lie in the free space.

Our third approach extends the roadmap toward unexplored regions of the free space, also in a manner similar to that of RRT algorithm [3]. First, a random configuration q_{rand} is generated and its nearest neighbor on the roadmap q_{near}. Moreover, we bias these samples toward the obstacles, since least explored areas are likely to be near them. A new milestone q_{new} is generated that extends q_{near} toward q_{rand}. Then, straight-line paths from q_{new} to its nearest neighbors are tested and added to the roadmap if they are valid.

V. Motion Planning with RDRs

In this section, we present our algorithms to compute collision-free path using RDRs. The RDRs can be be used with a single robot to avoid obstacles or with multiple robots.

A. Simple Approach

Performing motion planning using RDRs is similar to other roadmap-based methods. For multiple robots, each robot can have its own RDR. In this way, it treats all other robots as dynamic obstacles.

We first connect the each robot's starting and final configuration to the RDR by finding the nearest dynamic milestone to each configuration and attach them via a reactive link. Next, an A* search algorithm to compute a path. As the path deforms in response to the dynamic obstacles, the robot's motion follows the deformed path naturally.

It is possible that a path can become invalidated during execution. This occurs when a link has been removed due to obstacle motion. When these events occur, first we determine if the robot is still connected to the roadmap, and then replanning is performed via an A* search. If no path is found, links are added as previously mentioned.

B. Global RDR Approach

In many circumstances with multiple robots, each robot is of the same type and has the same configuration space and free space (when only non-agent obstacles are considered). Rather than maintaining a separate RDR for each of these robots, we can use a single RDR (see Fig. 3). In this case, fewer particles are needed, resulting in a better overall performance.

In order to use this single RDR for multiple robots, we make a few modifications to our basic algorithm; robots need to be able to apply forces to the RDR and additional coordination is required.

1) Robot Force: Paths for separate robots can be disjoint but nearby each other. In this case, a robot would need to apply forces to the single RDR to ensure avoidance with other robots. The force a robot applies on the roadmap is the same as that for an obstacle (Eqn. 1). One caveat is that

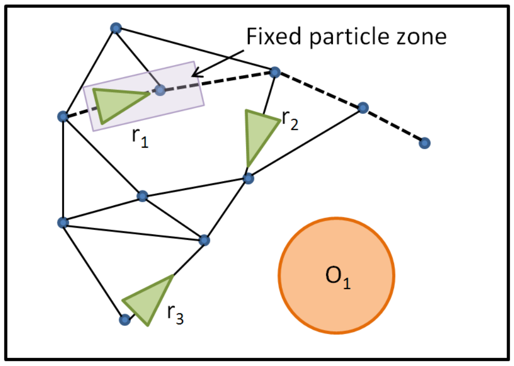

Fig. 3. **Multiple robots on the RDR:** For proper avoidance, each robot applies forces on the RDR except on the particles that are in the fixed particle zone.

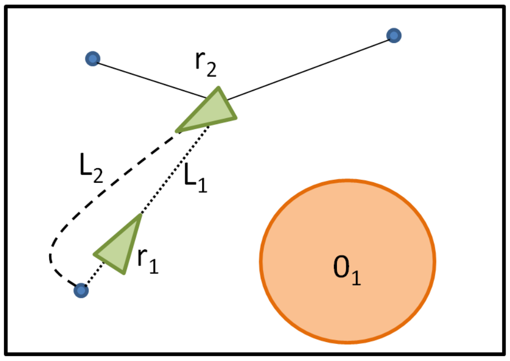

Fig. 4. **Adding additional links between milestones:** As the robot r_2 approaches a link occupied by r_1, an additional link L_2 is added for r_2 to traverse.

a robot would also by applying forces to its own path along the roadmap, possibly causing unnecessary motion in this direction. To prevent this, we define a "fixed particle zone" for a robot i with path P as:

$$\mathcal{Z} = \{\mathbf{p} | (\mathbf{p} \in \mathcal{P}, d(\mathbf{x}, \mathcal{P}) < \epsilon_p) \wedge d(\mathbf{x}, \mathbf{q}(t)) < \epsilon_r\},$$

where $\mathbf{x}$ is the position of particle $\mathbf{p}$, $d(\mathbf{x}, \mathcal{P})$ is the distance from $\mathbf{x}$ to the path link, ϵ_p is a path distance threshold, and ϵ_r is a robot distance threshold. This region may enclose the particles either on or near the path link which the robot is currently traversing, as well as those near the robot (See Fig. 3).

2) Coordination: With multiple robots on a single RDR, it may be necessary to coordinate their motion to prevent deadlock or collisions with other robots.

Coordination is handled through altering agent velocities and creating additional routes. The main issues arise when the robots arrive at a milestone at the same time and a robot is trying to use a link that is already occupied by another robot. When robots are near each other, one robot further away from the milestone is instructed to slow down until the robot ahead is sufficiently clear of the milestone and other robots. To handle link contention, it is possible to make use of the reactive properties of the link itself. Rather than coordinating which robot uses the link at a given time, an additional link is added between the milestones as a reactive link (See Fig. 4). This additional link will react to the motion of the other robot that is already on the link and vice versa for the existing link. This approach also provides an effective mechanism for the robots to avoid each other.

It should be noted that these solutions have their limitations. It is necessary to be careful about how the agent velocities are altered in order to not create a deadlock situation or a situation in which a single robot is stalled and cannot proceed. The additional link is not practical when there is insufficient space for two separate robots to pass through each other safely.

VI. Results

In this section, we analyze the performance of our planner on some benchmarks and also highlight its performance.

A. Results

We highlight the performance of our planner on planar robots undergoing translational and rotational motion, as shown in Fig. 5 and in Fig 6. In this first benchmark, we demonstrate our algorithm on motion planning of 15 star-shaped robots (indicated in different colors) using reactive-deforming roadmaps. Each robot has 3-DOF (2T+R) and acts as a dynamic obstacle for the other 14 robots. The goal for each robot is represented as a thick point of same color in the figure. Thin gray segments represent the initial roadmap, thick colored segments denote the path for each robot. Fig. 5 (b) and (c) show the snapshots of two timesteps of our planning algorithm. Only the deforming paths are shown. Fig. 5 (d) shows the final configuration of the robots, the reactive-deforming roadmaps, and the corresponding paths. Our planner takes a total of $1,525$ simulation steps, with an average time per step of 10.6 msec on a 2.1GHz Pentium Core2 CPU.

The second benchmark incorporates an application of multi-robot planning with dynamic onstacles to crowd simulation. Fig.6 shows an urban environment with 4 dynamic obstacles (cars) and 3-DoF human agents. Each agent has a unique goal near one of the buildings, which is updated as an agent reaches the goal. The dynamic obstacles invalidate links which forces agents to replan. As links become clear, the roadmap is repaired. For an environment with 100 agents, the average time per step is 11.5msec.

Our current system is a proof-of-concept implementation. The code is unoptimized and its runtime performance can be considerably improved.

VII. Analysis and Conclusions

We have presented a physically-based, adaptive roadmap representation that retracts and changes its topology as a function of the dynamic environment. The RDR can be used to plan the motion of a single robot or multiple robots among dynamic obstacles. Our formulation offers several advantages over existing approaches. Most of replanning algorithms remove the links as soon as they are invalidated. Since our links can retract based on the motion of the obstacles or other robots, they are not invalidated as frequently. Thus, the planner removes relatively fewer links and performs fewer milestone and link additions into the initial roadmap. In addition, by maintaining relatively longer links, in terms of path length, roadmap connectivity is not updated as frequently. Moreover, our formulation can easily accommodate moving goals of the robots or changes in the path topology due to dynamic obstacles. It is relatively simple to implement our algorithm using a mass-spring simulation framework, on top of a roadmap-based motion planner.

Our approach also has some limitations. Since we do not make any assumptions on the motion of the obstacles, the generated motion is susceptible to leaving the robot in a state where a collision is inevitable. It is also difficult to give any guarantees on the optimality of the computed paths. If the environment consists of heterogeneous or different robots, we need to maintain and update a separate RDR for each robot. Currently, we have tested the performance of our planner on complex scenarios with tens of planar 3-DOF robots. As part of future work, we plan to evaluate its performance on more complex scenarios with high-DOF robots. Improved coordination could lessen the number of extra lanes, and also improve performance. Multiresolution methods would also help improve the performance. Finally, we want to add a dynamics to the agent motion for more realistic behavior.

VIII. Acknowledgements

The authors would like to thank our reviewers for the useful feedback which helped to improve the work. This work was supported by a Department of Energy High-Performance Computer Science Fellowship administered by the Krell Institute and in part by ARO Contracts DAAD19-02-1-0390 and W911NF-04-1-0088, NSF awards 0400134, 0429583 and 0404088, DARPA/RDECOM Contract N61339-04-C-0043 and Disruptive Technology Office.

References

[1] J.-C. Latombe, *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[2] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, 2005.

[3] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[4] L. Kavraki, P. Svestka, J. C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Automat.*, pp. 12(4):566–580, 1996.

[5] J. Kuffner and S. LaValle, "RRT-connect: An efficient approach to single-query path planning," *Proceedings of the IEEE Intnational Conference on Robotics and Automation (ICRA), San Francisco, CA*, April 2000.

[6] M. Bennewitz, W. Burgard, and S. Thrun, "Optimizing schedules for prioritized path planning for multi-robot systems," *Proc. of IEEE Conf. on Robot. & Autom.*, 2001.

[7] C. Clark, S. Rock, and J. Latombe, "Motion planning for multiple robot systems using dynamic networks," *Proc. of IEEE Conf. on Robot. & Autom.*, 2003.

[8] M. Erdmann and T. Lozano-Perez, "On multiple moving objects," *Proc. of IEEE Conf. on Robot. & Autom.*, pp. 1419–1424, 1986.

[9] S. LaValle and S. Hutchinson, "Optimal motion planning for multiple robots having independent goals," *IEEE Trans. on Robotics and Automation*, vol. 14, no. 6, pp. 912–925, 1998.

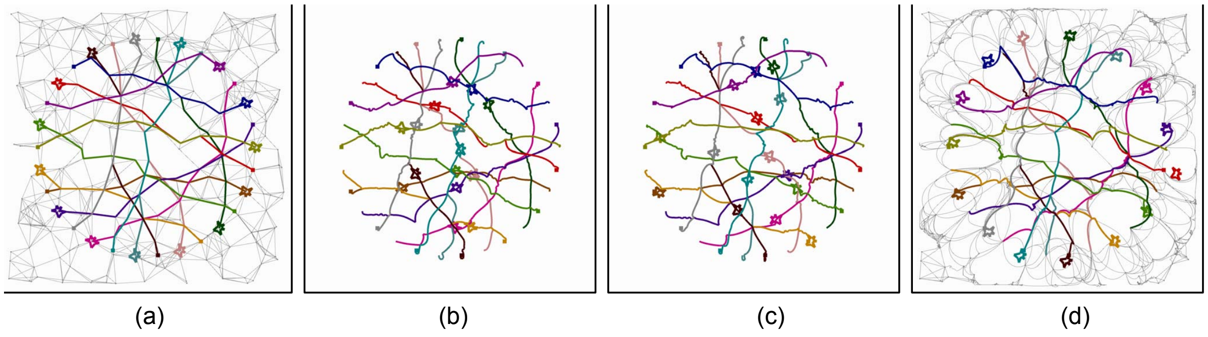

Fig. 5. Multi-robot motion planning of 15 star-shaped robots in different colors using reactive-deforming roadmaps: *(a) Initial configuration. Each robot has* 3*-DOF (*2*T+R) and acts as a dynamic obstacle for the other 14 robots. The goal for each robot is represented as a thick point of same color. Thin gray segments represent the initial roadmap, thick colored segments denote the path for each robot. (b)-(c) Two timesteps of our planning algorithm. Only the deforming paths are shown. (d) The final configuration showing the reactive-deforming roadmaps and the corresponding paths. Total number of simulation steps* $= 1,525$*, average time per step* $= 10.6$ *msec on a* $2.1GHz$ *Pentium Core2 CPU.*

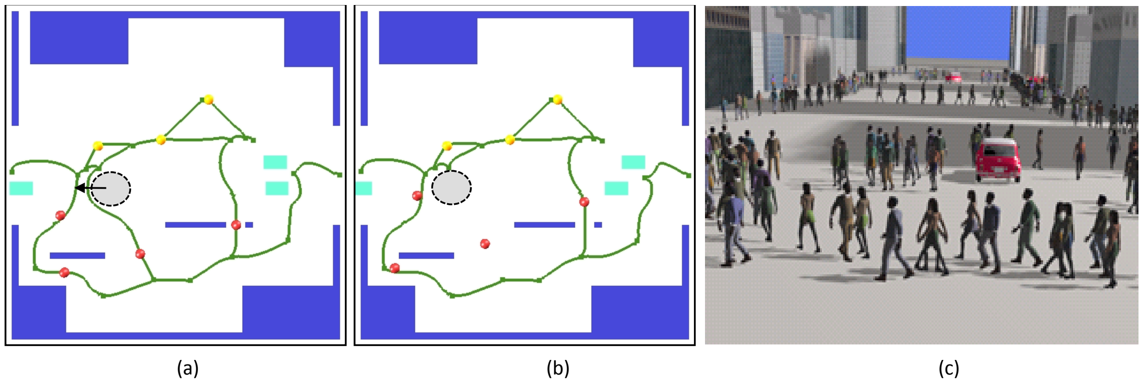

Fig. 6. Application of N-body motion planning using reactive deforming roadmaps to complex crowd simulation with human agents and polygonal dynamic obstacles: *(a)-(b) An instructive example with* 4 *agents (in red) and goals (in yellow). The static obstacles are in dark blue and dynamic obstacles (cyan). The reactive deforming roadmap is shown with green links. The dynamic obstacles represent cars. As the highlighted car (circled) moves, the affected link in the roadmap is removed. (c) A real-time simulation of motion planning for* 100 *human agents and* 4 *cars in same environment. The average time for roadmap update and motion planning per frame* $= 11.5ms$ *on a* $2.1Ghz$ *Pentium Core2 CPU.*

[10] S. Carpin and E. Pagello, "A distributed algorithm for multi-robot motion planning," *Proc. of Fourth European Workshop on Advanced Mobile Robots*, pp. 207–214, 2001.

[11] T.-Y. Li and H.-C. Chou, "Motion planning for a crowd of robots," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003.

[12] B. Aronov, M. de Berg, A. F. van der stappen, P. Svestka, and J. Vleugels, "Motion planning for multiple roobts," *Discrete and Computational Geometry*, 1999.

[13] K. Kant and S. Zucker, "Towards efficient trajectory planning: The path-velocity decomposition," *Int. Journal of Robotics Research*, vol. 5, no. 3, pp. 72–89, 1986.

[14] T. Simeon, S. Leroy, and J. Laumond, "Path coordination for multiple mobile robots: a geometric algorithm," *Proc. of IJCAI*, 1999.

[15] O. Khatib, "Real-time obstable avoidance for manipulators and mobile robots," *IJRR*, vol. 5, no. 1, pp. 90–98, 1986.

[16] C. W. Warren, "Multiple path coordination using artificial potential fields," *Proc. of IEEE Conf. on Robotics and Automation*, pp. 500–505, 1990.

[17] O. B. Bayazit, J.-M. Lien, and N. Amato, "Roadmap-based flocking for complex environments," *Proceedings of Pacific Conference on Computer Graphics and Applications (PG)*, 2002.

[18] J.-M. Lien, S. Rodriguez, J.-P. Malric, and N. Amato, "Shepherding behaviors with multiple shepherds," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2005.

[19] T. Fraichard, "Dynamic trajectory planning with dynamic constraints: A "state-timespace" approach," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, July 1993.

[20] K. Hoff, T. Culver, J. Keyser, M. Lin, and D. Manocha, "Interactive motion planning using hardware accelerated computation of generalized voronoi diagrams," *Proceedings of IEEE Conference of Robotics and Automation*, 2000.

[21] S. Quinlan and O. Khatib, "Elastic bands: Connecting path planning and control," *Proc. of IEEE Conf. on Robotics and Automation*, 1993.

[22] O. Brock and O. Khatib, "Elastic strips: A framework for motion generation in human environments," *Int. Journal of Robotics Research*, vol. 18, no. 6, pp. 1031–1052, 2002.

[23] Y. Yang and O. Brock, "Elastic roadmaps: Globally task-consistent motion for autonomous mobile manipulation," *Proceedings of Robotics: Science and Systems*, August 2006.

[24] A. Stentz, "The focussed D* algorithm for real-time replanning," *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 1995.

[25] S. Koenig and M. Likhachev, "Improved fast replanning for robot navigation in unknown terrain," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2002.

[26] P. Leven and S. Hutchinson, "Toward real-time path planning in changing environments," *Proceedings of the fourth International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2000.

[27] T.-Y. Li and Y.-C. Shie, "An incremental approach to motion planning with roadmap management," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2002.

[28] M. Kallmann and M. Mataric, "Motion planning using dynamic roadmaps," *Proceedings of the IEEE Conference on Robotics and Automation (ICRA)*, April 2004.

[29] L. Jaillet and T. Simeon, "A PRM-based motion planning for dynamically changing environments," *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2004.

[30] D. Ferguson, N. Kalra, and A. Stentz, "Replanning with RRTs," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2006.

[31] A. Sud, R. Gayle, E. Andersen, S. Guy, M. Lin, and D. Manocha, "Real-time navigation of independent agents using adaptive roadmaps," *Proc. of ACM VRST*, 2007, to appear.