Auton Robot (2010) 28: 113–130
DOI 10.1007/s10514-009-9151-x

Elastic roadmaps—motion generation for autonomous mobile manipulation

Yuandong Yang · Oliver Brock

Received: 22 January 2009 / Accepted: 23 August 2009 / Published online: 12 September 2009

Abstract The autonomous execution of mobile manipulation tasks in unstructured, dynamic environments requires the consideration of various motion constraints. The task itself imposes constraints, of course, but so do the kinematic and dynamic limitations of the manipulator, unpredictably moving obstacles, and the global connectivity of the workspace. All of these constraints need to be updated continuously in response to sensor feedback. We present the elastic roadmap framework, a novel feedback motion planning approach capable of satisfying all of these motion constraints and their respective feedback requirements. This framework is validated in simulation and real-world experiments using a mobile manipulation platform and a stationary manipulator.

Keywords Motion generation · Mobile manipulation · Planning and control · Feedback planning · Hybrid systems for motion

Electronic supplementary material The online version of this article (http://dx.doi.org/10.1007/s10514-009-9151-x) contains supplementary material, which is available to authorized users.

Y. Yang
Robotics and Biology Laboratory, Department of Computers Science, University of Massachusetts Amherst, 140 Governors Drive, Amherst, MA 01003, USA

O. Brock (✉)
Robotics and Biology Laboratory, School of Electrical Engineering and Computer Science, Technische Universität Berlin, Einsteinufer 17—EN 10, 10587 Berlin, Germany
e-mail: oliver.brock@tu-berlin.de

1 Introduction

Autonomous mobile manipulation is the branch of robotic research concerned with the development of integrated systems capable of autonomously performing general manipulation tasks in unstructured and dynamic environments. Current robots have already revolutionized manufacturing, explored the surface of Mars, automated biology laboratories, and enabled safer and more effective surgeries. But in all of these robotic success stories robots either perform specialized tasks in controlled environments or are teleoperated by humans. True autonomy for general tasks remains beyond our reach. At the same time, the potential of replicating these success stories in unstructured environments and for general and variable tasks is enormous (Brock and Grupen 2005). Robots could help address many societal problems, such as the need to care for an aging population, the urgency to monitor environmental pollution and remove contamination, the pressure to reduce health care costs while improving the quality of service, and the necessity to respond to economic pressures for manufacturing in small batch sizes.

What are the obstacles that have to be overcome for autonomous robotic systems to perform general mobile manipulation tasks in unstructured environments? Prominently among them is certainly the need to continuously perceive the environment and to incorporate feedback about those perceptions into the ongoing actions of the robot. Such feedback is necessary to robustly, reliably, and successfully perform tasks in unstructured environments. In this paper, we are concerned with this issue in the context of motion generation and execution.

In autonomous mobile manipulation, the motion performed by a robot to achieve a task is subject to numerous constraints. We will discuss these constraints using an example task. Consider a mobile manipulator that has to inspect a gas pipe for leaks.

1. **Task constraints:** The task imposes constraints on the robot's motion, most often expressed as position or force constraints at the end-effector. For the pipe inspection task, assuming that a camera is located at the end-effector, the robot has to maintain spatial proximity to the pipe as well as a proper orientation of the camera to keep the pipe in the field of view. Task constraints generally have the highest feedback requirements; force control, for example, can require feedback rates of about 1000 Hz.
2. **Kinematic and dynamic constraints:** The limitations of physical mechanism, such as joint and actuation limitations, have to be considered when generating the robot's motion. Reaching a joint limit may mean that the end-effector task cannot be maintained and the pipe will exit the camera's field of view.
3. **Posture constraints:** Subordinate task constraints may be imposed on the robot's motion to improve some performance metric. For example, in the pipe inspection task it is desirable for the manipulator to remain in the center of its workspace so as to maximize the mobile manipulator's ability to perform other motions without losing sight of the pipe.
4. **Reactive obstacle avoidance:** The necessity to avoid moving obstacles also imposes constraints on the robot's motion. The mobile manipulator may encounter unpredictably moving obstacles during its inspection task and has to avoid them.
5. **Global motion:** To support the execution of the end-effector task, the mobile manipulator has to perform a gross motion that requires the consideration of global connectivity constraints imposed by the environment. While the end-effector task may be fully defined by a trajectory in the proximity to the pipe, the entire mobile manipulator has to move through the environment in such a way that the constraints imposed on the end-effector can always be maintained.

A successful motion generation approach for autonomous mobile manipulation has to address all five types of motion constraints and also satisfy their respective feedback requirements. In addition—to accommodate modeling error, uncertainty in sensing and execution, and the dynamic aspects of the environment—sensing feedback has to be incorporated into the motion generation continuously. The required frequency of feedback varies with the specific constraint and the application. In autonomous mobile manipulation, feedback requirements can range from several hundred times per second for task constraints to about once per second for global motion constraints.

Existing approaches to control or motion planning are not able to satisfy these requirements. Control methods are susceptible to local minima in the controller's potential function and therefore cannot guarantee that the resulting motion meets global task constraints. Planners, on the other hand, are too computationally intensive to incorporate feedback at the rates required for task constraints. This is particularly apparent when one considers tasks for which the robot's end-effector has to move on a specified trajectory. Planners suitable for this problem require complex computation to meet the resulting end-effector constraints (Oriolo 2005; Stilman 2007; Tang et al. 2005; Yao and Gupta 2007).

We present a novel motion generation technique, called *elastic roadmap* (Yang and Brock 2006). Elastic roadmaps generate robust and globally task-consistent motion in dynamic environments. They represent a novel framework for feedback motion planning (LaValle 2006) that combines the advantages of planning and control to address the motion requirements in autonomous mobile manipulation. From motion planning this framework takes the concept of a roadmap (Kavraki et al. 1996) to represent global connectivity information. Based on this information, the elastic roadmap approach composes a hybrid system of robust, task-specific controllers. This hybrid system represents a global, task-specific, multi-objective navigation function. The resulting motion framework addresses all of the motion constraints discussed above while satisfying their respective feedback requirements. The framework is able to generate motion in free space as well as in contact with the environment. We demonstrate the effectiveness of the proposed framework in simulation and in real-world experiments on a mobile manipulation platform.

2 Related work

The literature concerning robot motion is extensive. We restrict the discussion to work that incorporates multiple motion constraints relevant to autonomous mobile manipulation.

2.1 Control

Control methods (Franklin et al. 1994) specify motion behavior using potential functions defined in the robot's state space. Each potential function represents a specific motion constraint or motion objective. Objectives can be combined into multi-objective controllers by combining the respective potential functions. Control-based methods descend the gradient of such a combined potential function to achieve the desired motion behaviors. The use of feedback during gradient descent renders motion generation robust to external disturbances at high rates of feedback but also makes it susceptible to local minima, especially in the combined potential landscape.

The operational space framework (Khatib 1987) achieves complex, multi-objective behavior (Sentis and Khatib 2005) for manipulation tasks by combining potential functions with nullspace projections. These projections ensure that subordinate motion objectives do not interfere with superior ones. The resulting motion satisfies the motion constraints and their feedback requirements for autonomous mobile manipulation, with the exception of global connectivity constraints. Control-based methods are subject to local minima and cannot guarantee the successful attainment of a particular motion objective.

Operational space control will play an important role in the elastic roadmap approach. Elastic roadmaps apply this versatile and effective control method to a representation of the connectivity of the environment. This connectivity information enables the elastic roadmap framework to avoid local minima, thereby overcoming the limitations of purely control-based approaches.

2.2 Sampling-based motion planning

Motion planning methods take a different approach to motion generation than control-based methods. Motion planners construct a global representation of the free configuration space to determine a valid motion. Among a large number of global motion planning techniques (Choset et al. 2005; LaValle 2006), sampling-based motion planners (Kavraki et al. 1996; Kuffner and LaValle 2000) currently represent the dominant planning paradigm. Global motion planners have been extended to specifically address constraints arising in manipulation (Siméon et al. 2004; Stilman 2007) or in the context of dynamic environments (Garber and Lin 2002; Hsu et al. 2000; Gayle et al. 2007; Jaillet and Siméon 2004; Kallmann and Matarić 2004; Leven and Hutchinson 2002; van den Berg and Overmars 2005; Vannoy and Xiao 2004; Zucker et al. 2007). However, these methods are not computationally efficient enough to incorporate feedback at rates of up to 1000 Hz; such rates are required to satisfy task constraints in the context of autonomous mobile manipulation, in particular for motion in contact with the environment.

To improve the computational efficiency of sampling-based motion planners, researchers have developed a number of informed, adaptive sampling strategies. They leverage the information acquired during planning to guide the selection of future configuration space samples. Guidance can be derived from information about local features of the configuration space (Hsu et al. 2005b; Morales et al. 2004; Zucker et al. 2008) or from global information about the entire configuration space and its known free space connectivity (Burns and Brock 2005; Jaillet and Siméon 2008). Other strategies use predefined motion primitives to bias sampling (Hauser et al. 2006) or limit the degrees of freedom that the sampler has to consider (Juan Cortés and Siméon 2008). Planners that use such adaptive sampling strategies generally outperform those that employ simpler, fixed sampling strategies. Nevertheless, the resulting gain in computational efficiency does not suffice to enable planning and replanning at the rates required to satisfy the motion requirements laid out above. This holds true in particular when the tasks that are considered require the end-effector to move along a specified path rather than just to reach a goal configuration (Oriolo 2005; Stilman 2007; Tang et al. 2005; Yao and Gupta 2007).

We believe that motion planners based on configuration space sampling carry an unnecessary computational burden that prevents them from satisfying the motion requirements of autonomous mobile manipulation. These planners attempt to determine connectivity information, which derives from the workspace description of the environment, in configuration space. However, the projection of workspace geometry into the robot's configuration space obscures information that would be easily accessible directly in the workspace. As a result, configuration space can be highly complex, even if the workspace and the robotic mechanism are rather simple. To alleviate sampling-based planners of this unnecessary complexity, some planners employ adaptive sampling strategies that rely on workspace information.

Workspace information provides a key ingredient for computationally efficient planning in high-dimensional configuration spaces. It can be obtained with little computational effort and has been shown to greatly accelerate the planning process. For example, workspace information has been used to adapt the sampling density of motion planners in order to place samples more densely in regions with narrow passages (Kurniawati and Hsu 2004; van den Berg and Overmars 2004; Yang and Brock 2004). This has resulted in substantial performance improvements. Even greater improvements can be achieved when planners decompose the overall planning problem into a workspace problem and a configuration space problem. The workspace problem can be solved efficiently and the resulting information aids in solving the more complex planning problem in configuration space. The resulting planners can address some of the most complex motion planning problems with high computational efficiency (Brock and Kavraki 2001; Yang and Brock 2005; Plaku et al. 2007; Diankov et al. 2008). But even these planners are not generally capable of satisfying the motion constraints required for autonomous mobile manipulation.

The elastic roadmap approach presented in this paper makes extensive use of workspace information. It uses this information to improve the feedback rates for the computation of global connectivity information. We will show this leads to a motion generation approach suitable for autonomous mobile manipulation.

It should be noted that the elastic roadmap approach does not represent an alternative to sampling-based motion planners but instead complements them. Moreover, we will later discuss how sampling-based motion planners can be employed within the elastic roadmap framework. Also, the elastic roadmap framework is inherently incomplete and may fail even when a valid path exists; in those cases sampling-based motion planners can be employed to overcome the incompleteness of elastic roadmaps. Details of this are discussed in Sect. 4.8.

2.3 Feedback motion planning

Feedback motion planners explicitly account for the fact that the information available during planning may be imprecise, that the environment may change during motion execution, and that motion execution results in uncertainty about the state of the robot. Rather than determining a specific solution path, feedback motion planners construct a local minima-free potential function based on global information (LaValle 2006). This can be achieved with navigation functions (Rimon and Koditschek 1992), numerical navigation functions (Barraquand and Latombe 1991), harmonic potential functions (Connolly and Grupen 1993; Sato 1987), or by composing a series of local potential functions (Burridge et al. 1999; Chen and Hwang 1998). The potential function defines appropriate motion commands for the entire permissible state space of the robot. The robot's motion is then generated by descending the resulting potential function. Once a global potential function has been computed, the planner can determine motion commands while sensors update the state of the world. A variety of feedback motion planners exist for low-dimensional or static configuration spaces (Choi and Latombe 1991; Conner et al. 2006; Lindemann et al. 2006; Yang and LaValle 2003).

Changes in the environment or in the robot's task invalidate a computed global potential function. As a result, a computationally complex re-computation of this function is required. Therefore, feedback motion planners that recompute global potential functions from scratch cannot satisfy the feedback requirements of autonomous mobile manipulation.

An alternative approach to feedback motion planning performs incremental modification of potential functions. This enables the continuous incorporation of sensor information into the motion generation process while at the same time providing update rates of the potential function that satisfy the requirements of autonomous mobile manipulation. The earliest of these approaches is the elastic band framework (Quinlan and Khatib 1993), which assumes that a global motion planner has determined a motion that satisfies the task-requirements. Local methods subsequently modify this motion incrementally in response to feedback from the environment.

The elastic strip framework (Brock and Khatib 2002) extends the elastic band framework. It is able to generate global, task-consistent obstacle avoidance and posture behavior for a mobile manipulator (Brock and Khatib 2002). Both elastic bands and elastic strips combine the advantages of control-based approaches with a predetermined global motion. However, neither of these methods can recover from an invalidation of the global motion.

The elastic roadmap frameworks represents important progress relative to elastic strips. In contrast with elastic strips, the elastic roadmap framework is a global approach to motion generation. Elastic strips incrementally modify a path from a single, given homotopy class. Changes in the environment can therefore lead to invalid or arbitrarily bad motions. In contrast, the elastic roadmap framework continuously re-plans globally and therefore generates paths from all homotopy classes while optimizing an specified optimality criterion. The elastic roadmap framework thus combines the benefits of elastic strips with those of global, albeit incomplete, motion planning. This means that elastic roadmaps address a motion constraint that could not be handled by elastic strips.

3 The elastic roadmap framework

The elastic roadmap framework relies on the following two main ideas:

1. The elastic roadmap framework *shifts the boundary* between planning and control. Traditionally, motion generation combines planning and control in some way. In most cases, a planner first determines a complete path or trajectory, which is then executed using a controller. In this traditional division, the planner determines a precisely specified motion, expressed as a curve in configuration space. Control is then relegated to minimizing the error introduced by uncertainties in sensing and actuation relative to the desired motion. This division of labor between planning and control has the disadvantage that each time the environment changes, the planner has to recompute the solution, possibly throwing away most of the motion determined previously. In dynamic environments, this will lead to unnecessary computation.

   In the elastic roadmap framework, the role of planning is to determine an approximate, global understanding of the connectivity of free space. This connectivity information is then used to generate a hybrid system of task-specific controllers. It is this hybrid system then that at execution time generates the actual motion performed by the robot.

As it relies on hybrid systems, the elastic roadmap framework shares the advantages of other feedback planners. It does not determine a single path but instead an infinite number of possible solutions, captured by the hybrid system of controllers. From these solutions, the framework can select the most appropriate one based on feedback about the environment, the state of the robot, or the state of the robot's task. As a result, task-constrained motion execution becomes robust in dynamically changing environments and in the presence of uncertainty.

But the elastic roadmap framework also differs from other feedback planners (LaValle 2006) in an important respect. Most feedback planners are designed to provide guarantees about characteristics of the motion they produce. Such guarantees are highly desirable from an algorithmic point of view. However, the ability to produce such guarantees hinges on the assumption that the environment is perfectly known. In autonomous mobile manipulation this assumption is not realistic. Consequently, any guarantees would be meaningless in practical scenarios. The elastic roadmap framework explicitly recognizes the futility of making assurances based on incomplete and uncertain information. Instead, it consciously trades algorithmic completeness for computational efficiency to satisfy the motion requirements of autonomous mobile manipulation. The result is an approach to motion generation that—in contrast to other feedback planners—generates feedback plans for robots with many degrees of freedom at interactive rates.

2. To generate global feedback plans at interactive rates, the elastic roadmap framework *relies on workspace information* to determine and maintain configuration space connectivity. The connectivity of the workspace is obvious from its geometric description. The connectivity of configuration space, however, arises from a convolution of workspace geometry with the kinematic structure of the robot. This convolution renders the resulting space high-dimensional and complex. Some information about the configuration space connectivity, however, can directly be inferred from the workspace representation of the environment. The elastic roadmap framework leverages this fact to obtain a partial and approximate representation of the configuration space connectivity by examining the robot's workspace. This connectivity information is then used to construct a hybrid system of controllers, as will be described in the next section.

   The elastic roadmap framework constantly updates its connectivity information based on changes observed in dynamic environments. To maintain interactive rates, these updates are also performed based on workspace information. Details of this procedure are provided in the next section.

These two ideas—shifting the boundary between planning and control and leveraging workspace information to understand configuration space connectivity—are realized in the elastic roadmap framework through the following two algorithmic concepts.

Elastic roadmap: In general, roadmaps are data structures that capture global connectivity information in graphs. The graph consists of collision-free configurations, called vertices or milestones, and of collision-free paths between those vertices, the edges of the graph. Conventional roadmaps represent a static view of free configuration space connectivity: once a milestone or an edge is added to the roadmap, neither will be changed or removed. In contrast, an *elastic* roadmap moves its milestones and updates their connectivity to adapt to changes in the environment. The visual effect resulting from the continuous modification of the roadmap gives rise to the name *elastic* roadmap.

Elastic roadmaps also differ from conventional roadmaps in that they are represented in workspace. A milestone is a virtual placement of the robot in the model of the environment. An edge of the roadmap connects two milestones if workspace information indicates that a controller is able to generate a motion that moves the robot from one to the other.

More formally, an elastic roadmap is a multigraph, consisting of a set of milestones $M = \{m_1, \dots, m_n\}$ and an ordered relation $C : \Phi \times M \times M \rightarrow \{\text{true}, \text{false}\}$. The relation $C(\phi, m_i, m_j)$ replaces the notion of edges; it holds true for two milestones m_i, m_j if a feedback controller $\phi \in \Phi$ is able to move the robot from milestone m_i to m_j; otherwise it is false. Two milestones can be connected by multiple edges, differing in the controller associated with them. The relation C captures the known connectivity of the roadmap at any point in time, based on the set of available controllers Φ.

Navigation function: The relation $C(\phi, m_i, m_j)$ holds for two milestones if a feedback controller ϕ is able to move the robot from m_i to m_j. In this view of a roadmap, every milestone m_i is associated with a local potential function for which it is the attractor. Milestone m_j is connected to m_i if m_j is within the region of attraction of the potential function associated with m_i. An elastic roadmap thus defines a hybrid system of potential functions. Given a particular goal state, the connectivity of the roadmap determines how a hybrid system can compose a set of local potential functions into an approximate, global navigation function.

The elastic roadmap changes in response to perceived changes in the environment. Existing milestones are updated in a task-consistent manner, i.e. in such a way that all constraints remain satisfied. New milestones are added to the roadmap when changes in the environment or the discovery of new obstacles make this necessary. The connectivity among milestones is also updated based on whether or not the controller ϕ associated with the connecting edge. As we will show later in this paper, the associated computations can be performed at frequencies suitable for the feedback requirements of autonomous mobile manipulation.

4 Instantiating the elastic roadmap framework

The elastic roadmap framework was described in abstract terms in the previous section. It contains several so-far unspecified algorithmic components. We will now describe one particular choice for these components. In the discussion of these components, we will also briefly mention alternative methods of implementation and discuss some of the trade-offs. In Sect. 5 we then present experimental results for this particular implementation of the elastic roadmap framework.

4.1 Task-level control

Task-level controllers play a central role in our implementation of the elastic roadmap framework. They are used to generate the robot's motion but also to generate and maintain milestones. To facilitate the presentation of the elastic roadmap implementation, we will first discuss these controllers.

Task-level control (Khatib 1987) is a convenient and powerful method for generating multi-objective behavior for robotic systems. Rather than specifying explicit joint trajectories, this framework permits control of the manipulator's end-effectors, greatly facilitating programming for kinematically redundant robots. Task-level control also permits the task-consistent execution of subordinate behaviors through the use of nullspace projections. Given an end-effector task, such as the task of examining a pipe, the end-effector is controlled by specifying an artificial potential field that keeps the end-effector within the task constraints. Any deviation from the task is translated into a force F_{task} acting on the end-effector. A subordinate behavior can be expressed as a vector of joint torques Γ_0 required to achieve the behavior. We can now compose these two tasks in such a way that the subordinate task is guaranteed not to interfere with the superior task. The torque Γ to achieve task and subordinate behavior can be computed as follows:

$$\Gamma = J_{\text{task}}(q)\, F_{\text{task}}^T + N_{\text{task}}^T(q)\, \Gamma_0, \qquad (1)$$

where N_{task}^T represents a projection into the nullspace of the end-effector Jacobian J_{task}. This projection ensures that the subordinate behavior will not alter task behavior, i.e., it will result in task-consistent motion.

This principle of nullspace projections can be extended to cascade an arbitrary number of hierarchical behaviors (Sentis and Khatib 2005). If behavior i results in torque Γ_i, the torque

$$\Gamma = \Gamma_1 + N_1^T(q)\left(\Gamma_1 + N_2^T(q)\left(\Gamma_3 + N_3^T(q)(\ldots)\right)\right) \qquad (2)$$

combines these behaviors in such a way that behavior i does not affect behavior j if $i > j$. In (2), N_i^T is the nullspace projection associated with the task Jacobian of behavior i. Here, we adopt the more compact notation of the control basis (Huber and Grupen 1997) to describe such cascaded nullspaces. We associate a control primitive ϕ_i with each torque Γ_i. If a control primitive ϕ_i is executed in the nullspace of the control primitive ϕ_j, we say that ϕ_i is performed subject to ϕ_j, written as $\phi_i \lhd \phi_j$. We can now rewrite (2) as

$$\cdots \lhd \phi_3 \lhd \phi_2 \lhd \phi_1. \qquad (3)$$

Considering the example of our pipe-inspection robot, controller ϕ_1, the highest-order controller, could be used to avoid joint limits; ϕ_2 then performs reactive obstacle avoidance. The actual task is represented by controller ϕ_3 and is only executed if obstacles and joint limits can be avoided. Posture behavior, such as keeping the end-effector in the center of the manipulator's workspace, can then be implemented using a controller ϕ_4, which is subordinate to all others. Using these cascaded nullspace projections, it is possible to realize complex, multi-objective robot motion.

Task-level control with nullspace projections serves as the underlying control scheme for the elastic roadmap framework. It will enable us to quickly incorporate feedback while at the same time maintain necessary motion constraints. Such controllers will be used to generate and maintain milestones as well as to command the motion of the physical robot.

4.2 Creating milestones

An elastic roadmap has to capture a sufficient portion of the connectivity of free configuration space to allow the solution of motion queries. Milestones play a central role in determining the quality of a roadmap, as they represent via points for possible motions. We will first discuss the requirements milestones have to satisfy to be useful and then explain how these milestones can be created.

Milestones in an elastic roadmap not only have to be collision-free, they also have to satisfy task constraints. The task constraints of milestones are defined by the task. To describe milestone generation, we distinguish between two kinds of tasks: end-effector placement (the end-effector position is only constrained at the final configuration, but end-effector orientation may be constrained throughout the motion) and position-constrained end-effector motion (at least

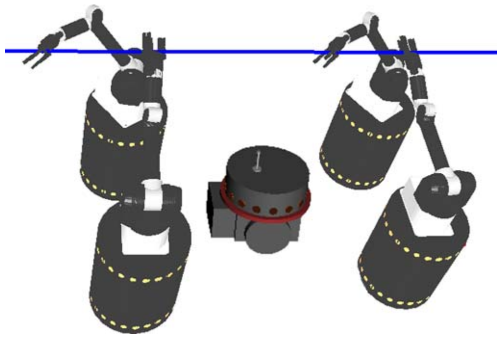


Fig. 1 Four task-consistent milestones associated with a simple obstacle. The task requires the end-effector to remain on the line

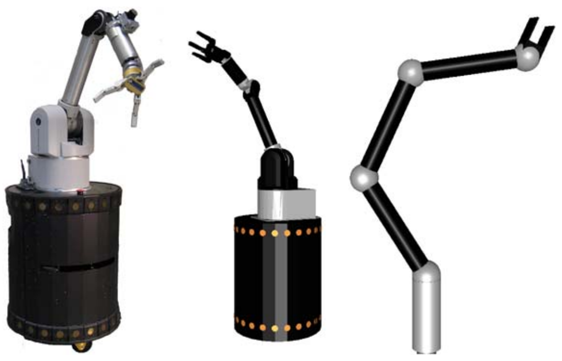


Fig. 2 *Left*: The UMass Mobile Manipulator (UMan) with ten degrees of freedom; *middle*: Model of the real platform; *right*: Stationary robot with four spherical joints for a total of twelve degrees of freedom

one of the translational degrees of freedom of the end-effector is constrained by the task). For end-effector placement tasks, we reduce the notion of configuration space coverage to workspace reach-ability for the end-effector: if an elastic roadmap allows us to reach every workspace location with the robot's end-effector, we have achieved workspace coverage. The motion for the remaining links of the robot will be generated by task-level controllers (Sentis and Khatib 2005), described in Sect. 4.1.

Example milestones for the pipe-inspection task are shown in Fig. 1. The figure shows four task-consistent milestones for UMan (UMass Mobile Manipulator, Fig. 2). The pipe-inspection task falls into the category of position-constrained tasks, as the end-effector is constrained on or close to the horizontal line. All valid milestones in the roadmap have to satisfy this constraint; otherwise they cannot be used to generate task-consistent motion. In this example, all milestones cluster around the pipe to satisfy task constraints. For end-effector placement tasks, such as the task of holding a glass of water upright, milestones would be distributed throughout the workspace.

In addition to being collision-free and task-consistent, milestones also have to be placed in such a way that they can capture as much of the free space connectivity as possible. For sampling-based multi-query motion planning, prior work shows that the adequacy of milestones is largely determined by their visibility properties, i.e., the amount of free configuration space "visible" to them (Hsu et al. 2005a). Milestones with large visibility provide better coverage of configuration space. This has motivated heuristics for placing samples in configuration space (Siméon et al. 2000). Another sampling heuristic attempts to place milestones close to the boundary of configuration space obstacles (Amato et al. 1998), following the intuition that solution paths circumnavigate these obstacles. When sampling in configuration space, however, visibility properties and obstacle boundaries are unknown, making it difficult to *a priori* select milestones with favorable properties.

We need to determine a roadmap that permits the motion of the end-effector from any feature on an obstacle to any other feature on the same obstacle. Effectively, this is an obstacle-specific roadmap. Since the motion between obstacles is by definition obstacle free, a collection of such roadmaps around obstacles for all obstacles would allow us to achieve workspace coverage for end-effector placement tasks.

In this implementation of the elastic roadmap framework, we chose to generate milestones based on workspace information about obstacles. More specifically, milestones are generated for configurations in which the robot is in proximity to workspace obstacles. For such configurations, the corresponding point in configuration space must be close to the boundary of the corresponding configuration space obstacle. Our technique for generating milestones is therefore a special case of other obstacle boundary sampling heuristics (Amato et al. 1998).

Workspace obstacles are decomposed into convex regions; these regions are then approximated by bounding boxes. We select each of the corners of the bounding box as well as the centers of the edges as obstacle features. These features are chosen so that the end-effector can move freely between adjacent features. To create milestones associated with these obstacle features, we pick a nearby configuration and drag the end-effector towards the feature. The controller used to achieve this is given by:

$$\phi = \phi_{\text{posture}} \lhd \phi_{\text{avoidance}} \lhd \phi_{\text{task}} \lhd \phi_{\text{feature}} \lhd \phi_{\text{collision}} \lhd \phi_{\text{kinematic}}, \tag{4}$$

where ϕ_{posture} describes a posture potential for kinematic conditioning of the robot, $\phi_{\text{avoidance}}$ performs reactive obstacle avoidance, ϕ_{task} describes a task potential, ϕ_{feature} is the potential that maintains proximity between the robot and the obstacle feature, $\phi_{\text{collision}}$ is able to prevent imminent collisions, and $\phi_{\text{kinematic}}$ prevents the manipulator from reaching its joint limits. The resulting milestones are added to the roadmap. Throughout the lifetime of the roadmap, each milestone is continuously maintained by its controller ϕ.

This enables the milestone to react to changes in the environment. If the controller ϕ is able to move the end-effector to the obstacle feature without collisions, the milestone is considered *valid*. If in addition all task-constraints are met, the milestone is considered *task-consistent*.

Each of the resulting milestones places the end-effector on an obstacle feature. Milestones associated with adjacent features are likely to be very similar. Furthermore, the motion of the end-effector between two different features is unobstructed. For most of the adjacent features it will therefore be possible to employ a task-level controller to generate the motion from one milestone to an adjacent milestone.

Milestones for position-constrained end-effector motion are created in a similar fashion. To be able to continuously enforce task and pose constraints at the end-effector, we simply ensure that some point on the robot is close to the obstacle feature. In (4), the order of ϕ_{feature} and ϕ_{task} is changed. If the resulting milestone satisfies the task constraints, it is considered *task-consistent*. Four task-consistent milestones for task-constrained end-effector motion that were generated in this fashion are shown in Fig. 1.

This method of selecting milestones is one aspect where the elastic roadmap consciously trades completeness for computational efficiency. It is obvious that the selection of milestones is critical for the quality of the elastic roadmap. Using our workspace-based method, we can make no claims of completeness for our approach (more on completeness in Sect. 4.8). However, this method can handle realistic scenarios for autonomous mobile manipulation. It enables the computationally efficient generation of new task-consistent milestones when a new obstacle is discovered. As a result, we are able to quickly update the elastic roadmap in response to changes in the environment. We will demonstrate this in our experimental validation in Sect. 5.

For scenarios in which the elastic roadmap implementation fails to find a valid motion, a more sophisticated method of milestone generation is desirable. Such a method can rely on the advances of sampling-based motion planners to generate localized, obstacle-based roadmaps. In other words, the generation of milestones represents a natural interface between the extensive research in sampling-based motion planning and the elastic roadmap framework. The investigation of this interface will be the subject of future work.

4.3 Maintaining milestones

The milestones in an elastic roadmap are continuously controlled by their respective controllers (4). This permits them to react to changes in the environment while possibly maintaining task constraints. In particular, when the obstacle associated with the milestone moves, the milestone will move with it. If these changes cause the milestone to violate task constraints (for task-constrained end-effector motion), the milestone changes its status from *task-consistent* to *valid*. If it violates collision avoidance constraints or kinematic constraints, it is labeled as *invalid*. The status of a milestone will become relevant when we explain how a particular sequence of controllers is extracted from an elastic roadmap in Sect. 4.5.

The computational complexity of milestone creation and milestone maintenance is $O(nm)$ for each interaction of the controller, where n is the number of degrees of freedom of the robot and m is the number of milestones generated. Efficient $O(n)$ algorithms for dynamic control (Chang and Khatib 2000) have to be applied to every one of the $O(m)$ milestones. This implies that the computational complexity of milestone maintenance is proportional to the geometric complexity of the workspace.

4.4 Determining connectivity among milestones

To complete the computation of an elastic roadmap, we need to determine the connectivity relation $C(\phi, m_i, m_j)$, where m_i, m_j are milestones and ϕ is a controller. This relation represents an edge of the roadmap.

In our current implementation, we employ workspace visibility as a criterion for the connectivity of two milestones. If two milestones m_i and m_j are mutually "visible," we add $C(m_i, m_j)$ to the connectivity relation of the elastic roadmap. Visibility between milestones is determined by evaluating if designated handle points (Yang and Brock 2004) on the respective milestones can be connected by straight, collision-free line segments. This criterion is computationally efficient, relatively accurate, and conservative, i.e., if the criterion determines that two milestones are visible, in most cases a valid trajectory can be determined using task-level, multi-objective control. In Sect. 4.7, we discuss how the planning method recovers from failure, should the visibility criterion erroneously label two milestones as connected.

While this approximation of connectivity compromises completeness (see Sect. 4.8), it allows us to use task-level, reactive control as a local planner among milestones. Again, the elastic roadmap consciously trades completeness for computational efficiency. As we will see in our experimental validation of the framework (see Sect. 5), this simple heuristic for the connectivity relation can solve challenging motion planning problems in dynamic environments in real time. In future work, we will investigate more sophisticated and complete workspace criteria to determine the connectivity of milestones in the elastic roadmap.

4.5 Extracting a navigation function from the elastic roadmap

An elastic roadmap does not represent explicit configuration space trajectories. Instead, it maintains a graph of task-consistent milestones (vertices) and hypotheses about the connectivity of these milestones (edges). Similarly to other roadmap-based motion planning approaches, we use graph search algorithms to extract a path in this graph that connects the initial and the final configuration of a motion planning problem (see Sect. 4.6). In the elastic roadmap approach, this path represents a sequence $m_1, m_2, \dots, m_n$ of milestones. The motion between two milestones m_i and m_{i+1} can be generated using the multi-objective controller ϕ stored in the edge $C(\phi, m_i, m_{i+1})$. An example of such a controller is given here:

$$\phi = \phi_{\text{posture}} \lhd \phi_{\text{avoidance}} \lhd \phi_{\text{global}} \lhd \phi_{\text{task}} \lhd \phi_{\text{collision}} \lhd \phi_{\text{kinematic}}. \quad (5)$$

Compared to (4), we have removed the control primitive ϕ_{feature} and added the control primitive ϕ_{global}, which is responsible for the global motion towards the next milestone.

Graph search in the elastic roadmap can be guided by a number of criteria. A sequence of controllers that goes through an invalid milestone is rejected. If the task imposes specific constraints on the end-effector, only milestones can be visited that satisfy those task constraints. This ensures that the resulting motion maintains task constraints at all times. In addition, graph search can be guided by optimality criteria, such as path length, path duration, or posture constraints. Due to the fact that elastic roadmaps are represented in low-dimensional spaces, their size is much smaller than configuration space roadmaps. Consequently graph search is computationally efficient and can be performed many times per second.

We view the extracted sequence of milestones as a hybrid system. Controllers are used to generate the motion from m_i to m_{i+1} until the robot has approached milestone m_{i+1}. The hybrid system then discretely switches to the controller that moves the robot from m_{i+1} to m_{i+2}, until the goal milestone m_n is reached. Note that throughout the entire motion all milestones as well as the motion between them remain consistent with all motion constraints. The hybrid system represents an approximate, local-minima free navigation function for the given motion problem. The navigation function is composed of simple, local potential functions by considering the global connectivity information captured in the roadmap (Burridge et al. 1999; Chen and Hwang 1998; Choi and Latombe 1991; Conner et al. 2003; Rimon and Koditschek 1992; Yang and LaValle 2003).

This notion of a hybrid system enables the elastic roadmap to never have to compute an explicit configuration space trajectory. This has two advantages. First, in a dynamic environment much of the computation of an explicit trajectory is wasted, as it will never be executed. Second, the use of controllers to represent motion enables the continuous and computationally inexpensive, continuous consideration of feedback about all motion constraints.

4.6 Updating the elastic roadmap

An update of the elastic roadmap consists of three parts: milestone maintenance, connectivity update, and navigation function extraction. Milestone maintenance (Sect. 4.3) is performed continuously at high frequencies (several hundred times per second) to ensure the maintenance of task constraints. The resulting motion of the milestones may invalidate the connectivity information represented in the roadmap. It is therefore necessary to continuously recompute the connectivity of the roadmap.

Updates are performed as follows: Given a roadmap with n milestones, there are potentially $O(n^2)$ visibility tests to perform during a connectivity update. Due to the small computational cost of a connectivity check and the low update frequency associated with global motion constraints, this is feasible in practice. In the experiments presented in Sect. 5, the computational cost of connectivity checks was dwarfed by the cost of milestone maintenance. In much larger environments, the cost of $O(n^2)$ visibility tests can be reduced to $O(n)$ (assuming a uniform spatial distribution of milestones) by restricting the adjacency of milestones based on spatial proximity and by confining updates to regions of the workspace in which changes were perceived.

Path extraction, the final step of a roadmap update, is described in Sect. 4.5 and is also performed continuously, but at a lower frequency of approximately 1–3 Hz. This means that several times per second a new global plan is computed; this is more than sufficient to satisfy the feedback requirements of global motion constraints.

4.7 Recovering from failure

We distinguish three failure modes. First, a motion can fail because the task-level controller is unable to find a motion between two milestones that are connected in the roadmap. Second, changes in the environment may force the robot to give up the task constraints, leading to task failure. For both of these failures we describe recovery strategies below. The inability to find a valid path, even after these recovery strategies have been applied, constitutes the third failure mode. This last failure has to be attributed to the incompleteness of our method (see Sect. 4.8). As mentioned before, this failure type presents a natural interface with more complete approaches to motion generation. When no valid path can be found using the elastic roadmap, the robot could interrupt its motion to invoke a more complete planner for the invalid edges in the roadmap.

The first failure mode occurs when the robot is moving between two milestones. If a robot fails to make progress without having reached the next milestone, the connection is labeled as invalid. The continuous path extraction process will automatically obtain a new path and the robot will start moving along this path. Currently, the two milestones remain unconnectable, but one could reconsider the connection after the environment has changed.

The second failure mode occurs during task-constrained motion. If the robot or any of the milestones in the current path change their label from task-consistent to valid or invalid, a new path has to be computed. This occurs automatically during path extraction. If a new path with task-consistent milestones can be found, it is executed. If no such path can be found, path extraction considers the shortest possible recovery path through valid milestones to a task-consistent milestone. To follow the recovery path, we use the following controller:

$$\phi_{\text{posture}} \lhd \phi_{\text{avoidance}} \lhd \phi_{\text{task}} \lhd \phi_{\text{global}} \lhd \phi_{\text{collision}} \lhd \phi_{\text{kinematic}}. \tag{6}$$

Once the robot reaches a task-consistent milestone, the original controller (5) is used to resume task behavior.

4.8 Completeness

The elastic roadmap framework is an approach to feedback motion planning that is incomplete by design. It does not possess any of the completeness properties of sampling-based planners. Nevertheless, it is tempting to compare the elastic roadmap framework to sampling-based planning methods. Such a comparison, however, is difficult to make, since the planning problems addressed by these two methods are fundamentally different. The elastic roadmap framework explicitly addresses task constraints and feedback requirements of a specific application and permits the execution of motion in dynamic environments under these constraints. It is able to do so precisely because it sacrifices completeness. To our knowledge, no sampling-based method is able to either consider task constraints in the generality proposed here or to address explicit timing constraints.

A characterization of completeness of the elastic roadmap would be most useful in the context of workspace properties. This has been proposed in Brock and Kavraki (2001), where a minimum clearance about the solution path is required for the method to be complete. Such a notion of completeness could be established for the elastic roadmap framework by showing that the selected milestones can be reached from the entire configuration space, excluding areas that do not provide sufficient clearance for the robot. Furthermore, such a notion would require workspace connection strategies with provable performance to ensure that when the regions of attraction of milestones overlap, task-level planning will find a connection trajectory. We will investigate such notions of workspace completeness in our future work.

5 Experimental evaluation

We demonstrate the performance of the elastic roadmap framework by performing both simulation and real-world experiments. An adequate experiment for the evaluation of the elastic roadmap approach should exhibit the following characteristics:
(1) The robot has to maintain an end-effector task throughout the entire motion. (2) The motion should be subject to kinematic or dynamic constraints. (3) A solution has to depend on global connectivity information. (4) The environment should be dynamic and the motion of obstacles should interfere with the motion. (5) The robot should be kinematically redundant to permit the execution of multi-objective behavior. Following these criteria, we have devised three experiments for the two simulated robots shown in Fig. 2 and one real-world experiment on our UMass Mobile Manipulator, our 10 degrees-of-freedom experimental platform.

In the first experiment, the mobile manipulation platform has to move its end-effector along a horizontal, linear path, indicated by the horizontal blue line in Fig. 3. During task execution, several obstacles move into the robot's path. The direction of motion for the obstacles is indicated by the arrows. The sequence of images in Fig. 4 illustrates how the elastic roadmap maintains task-consistent workspace connectivity and repeatedly generates new sequences of milestones in response to changes in the environment. By following the approximate navigation function constructed from the roadmap, the robot can move to the goal configuration in a task-consistent manner, i.e., by restricting the end-effector position to the line. Control primitives prevent collisions and

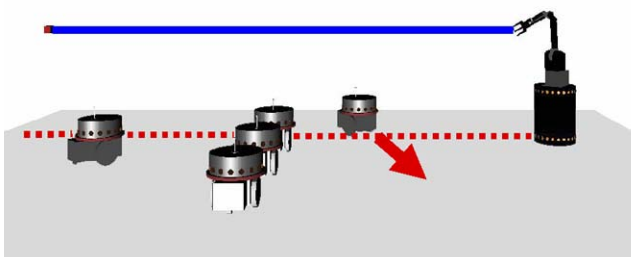


Fig. 3 First experiment: The end-effector of UMan is constrained to follow a line while obstacles move through the work space

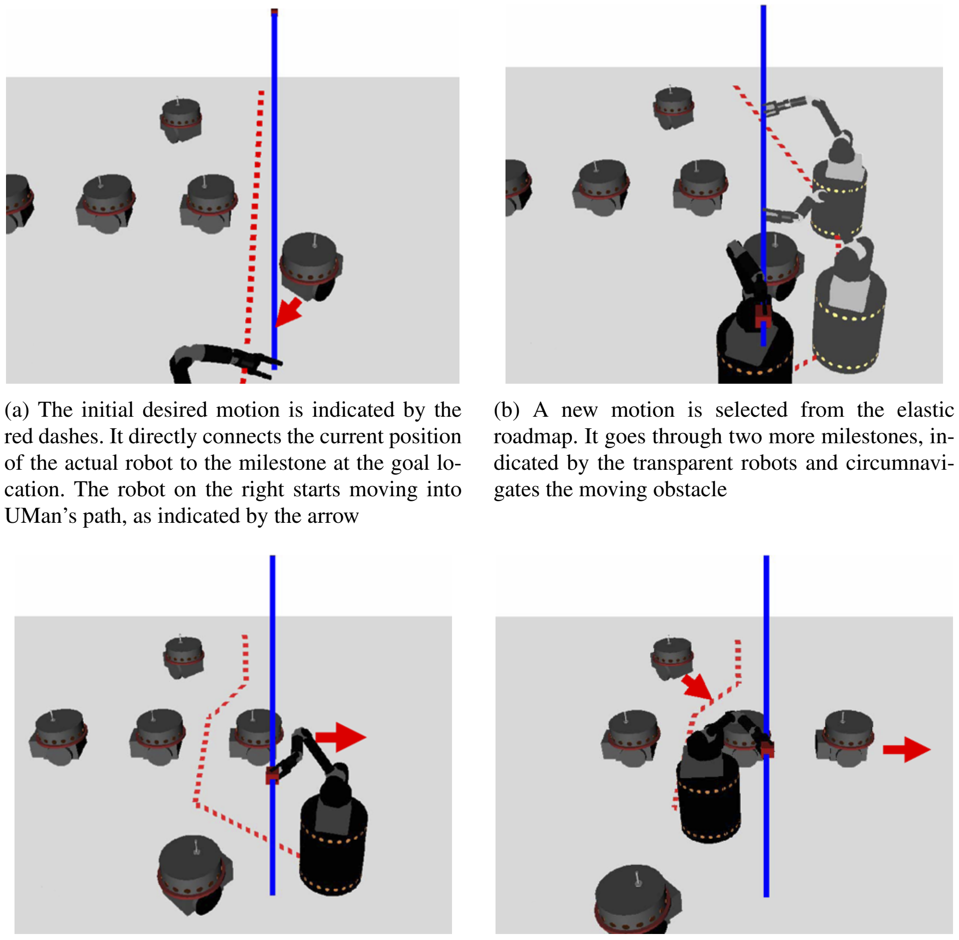


Fig. 4 (Color online) UMan performs a task that requires the end-effector to traverse a line in space. Multiple moving obstacles obstruct UMan's path. Collision-free and task-consistent motion is generated based on the elastic roadmap framework

(a) The initial desired motion is indicated by the red dashes. It directly connects the current position of the actual robot to the milestone at the goal location. The robot on the right starts moving into UMan's path, as indicated by the arrow

(b) A new motion is selected from the elastic roadmap. It goes through two more milestones, indicated by the transparent robots and circumnavigates the moving obstacle

(c) Due to the motion of the three robots (indicated by the arrow), a new motion is shown. It goes through three milestones of the roadmap (not shown) before reaching the goal position

(d) As the three robots continue their motion, another robot starts to move (again indicated by the arrows). Yet another motion is selected from the elastic roadmap

keep the robot away from its joint limits. All computations are performed during the simulation and the motion is generated in real time. The pictures represent snapshots of the ongoing execution.

Note that the narrow passages between the moving obstacles render this motion problem challenging for motion planners operating in configuration space. Furthermore, these planners also have difficulties to generate task-consistent motion, as the computation of the manifold of task-consistent configurations is computationally complex. In the elastic roadmap framework, neither of these difficulties poses a problem.

The transition from Fig. 4(b) to (c) exemplifies the important difference between elastic roadmaps and elastic strips. In the elastic strip framework the path would continue to deform to the right for as long as the small mobile robot continues to move. The elastic roadmap framework, in contrast, combines the incremental path modification of elastic strips with global motion replanning. As a result, a new path is computed, passing the mobile robot on the left, as shown in Fig. 4(c).

It is also important to note that replanning is performed in the full configuration space of the robot. From this specific experiment it might appear as if replanning is performed in a two-dimensional subspace, only considering the base of the robot. This is not the case. All degrees of freedom are used to respond to obstacles, there is no decomposition of the configuration space to render planning more efficient. (The example was chosen so that we would be able to easily recreate it on a real robot, see below.)

In a second experiment with the mobile manipulator, we demonstrate two additional capabilities of the elastic roadmap framework. First, we show that the framework is capable of generating task-consistent motion even for force-controlled tasks, i.e., motion in contact with the environment. This stands in contrast with other approaches to motion generation (Brock and Khatib 2002; Quinlan and Khatib 1993), which require the entire manipulator to move in free space. In the elastic roadmap framework, the only requirement is that motion continuously satisfies all motion constraints. These constraints may include force constraints. Second, we demonstrate that the elastic roadmap framework is able to automatically recover from the violation of task-constraints.

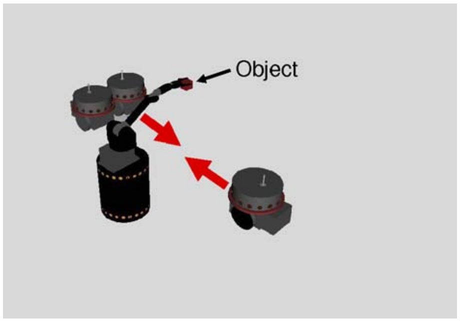
In this experiment, the end-effector of the manipulator tracks the unknown motion of an object based on force control. This can initially be achieved using task-level control alone, since the goal configuration is connected to the robot. As this direct connection is invalidated by moving obstacles forming a boxed canyon around the manipulator, the manipulator is unable to follow the moving object and has to violate task constraints by losing contact with the object. Image 2b in Fig. 5 illustrates how the elastic roadmap framework then determines a path through valid but not task-consistent milestones to re-attain the task constraints in image 2c.

In a third experiment, we demonstrate the effectiveness of the elastic roadmap framework for a stationary twelve degrees-of-freedom manipulator. The task consists of moving the end-effector to a goal location, while maintaining its orientation (task constraint). This motion is performed in an environment that contains a truss moving from right to left, as indicated by the arrows in Fig. 6. The sequence of images illustrates how the robot reaches its goal location, while avoiding the moving truss and maintaining its end-effector orientation. As the truss keeps moving, it forces the manipulator to deviate from its goal location, repeatedly triggering a replanning operation that results in a repeated motion, similar to the one shown in the figure.

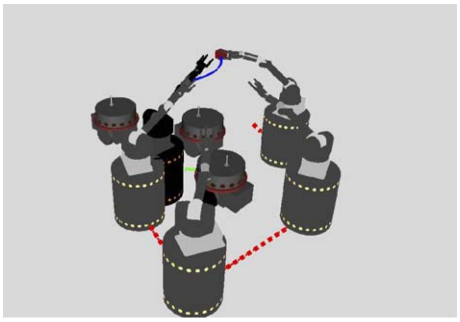
These simulation experiments were performed on a Pentium IV 3.2 GHz PC with 1 GB RAM and a 64 MB DDR Radeon 300 graphics card. The computations associated with the maintenance of the elastic roadmap and the extraction of a path can be performed at a frequency of approximately 5–10 Hz thereby satisfying the feedback requirements for global motion in the context of autonomous mobile manipulation.

To demonstrate the effectiveness of the elastic roadmap framework in real-world scenarios, we implemented the algorithms described above on UMan, our experimental platform for autonomous mobile manipulation. We applied the elastic roadmap framework to a line-following task, motivated by the pipe inspection example. This task requires UMan to move its end-effector along a horizontal line in its workspace to reach a given goal position. At the same time the robot has to avoid a stationary and a moving obstacle. Throughout the motion, joint limits and singularities are avoided. Posture constraints are imposed on the motion to keep the wrist in the middle of the joint range.

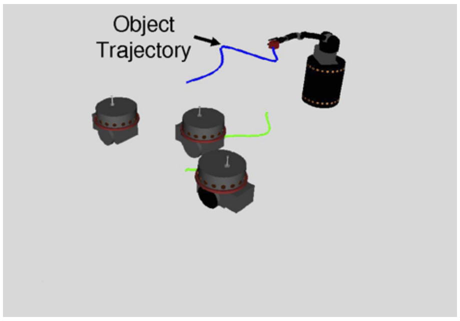
Snapshots of the experiment are shown in Fig. 7; a video is provided as supplementary material. Figure 7(b) best illustrates the experimental setup. Each of the sub-figures is divided into three parts. The left of each images shows the



(a) UMan maintains constant force contact with a cube moving through the environment. Three mobile robots move into UMan's path and prevent it from maintaining contact with the cube



(b) The elastic roadmap generates a motion through three valid but not task-consistent milestones until a task-consistent milestone can be reached and the task can be resumed

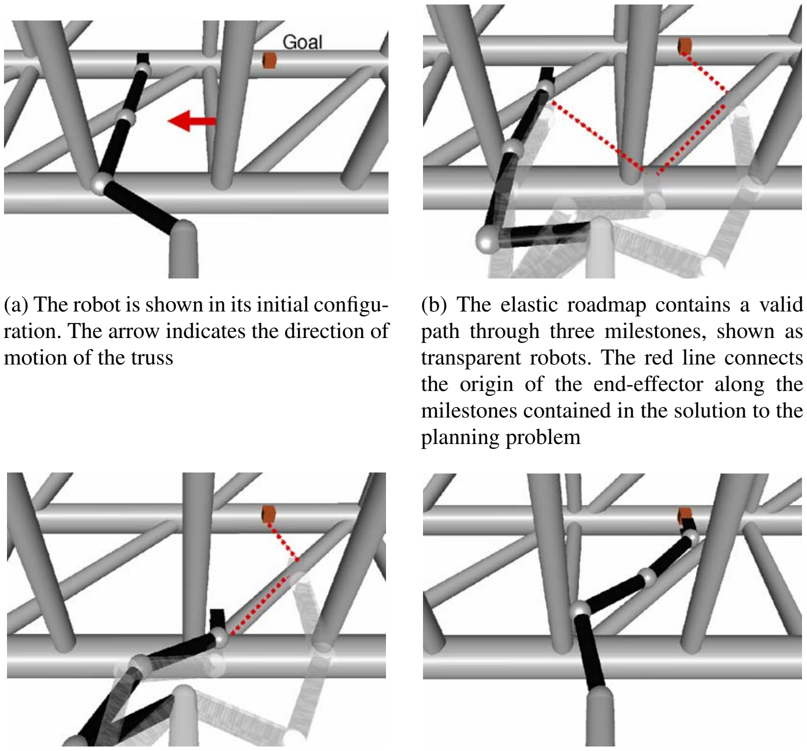


(c) UMan has resumed the task and maintains constant-force contact with the object. The figure also shows the objects trajectory in space (blue) and projected into the ground plane (green)

Fig. 5 (Color online) UMan has to maintain constant force contact with an object floating through space. The task is specified by the object's motion. During motion execution, moving obstacles prevent UMan from maintaining contact. The elastic roadmap determines the fastest way to resume the task and executes the corresponding motion

experiment with a wide field of view. The black line of the floor represent a projection onto the ground plane of the virtual line the end-effector has to follow. The top right of each image shows a close-up on UMan's end-effector. In Fig. 7(b) a red laser point can be seen on a white sphere held in UMan's end-effector. This laser point stems from a

Fig. 6 (Color online) A stationary robot with four spherical joints (twelve degrees of freedom) is performing an end-effector placement task in the presence of a moving truss. The goal is indicated by the red cube. The images show a sequence in which the robot reaches the goal. As the truss keeps moving to the left, the robot will be forced to deviate from its task-constraint, triggering another round of planning with the elastic roadmap. The sequence of images would repeat as the truss keeps moving



(a) The robot is shown in its initial configuration. The arrow indicates the direction of motion of the truss

(b) The elastic roadmap contains a valid path through three milestones, shown as transparent robots. The red line connects the origin of the end-effector along the milestones contained in the solution to the planning problem

(c) This image shows an intermediate step during the robot's motion. The truss' motion continues

(d) The robot has reached its goal location

laser pointer mounted on a tripod. The light ray emanating from the laser pointer represents the virtual line the robot has to follow. As long as the laser pointer is visible on the white sphere in UMan's hand we can be sure that the task constraint is satisfied. Finally, the bottom right shows the robot's view of the world and the part of the elastic roadmap that represents the current solution to the motion generation problem. The blue line represents the task constraint and corresponds to the ray of laser light. Transparent robots, connected by a red line on the ground plane, indicate the path extracted from the elastic roadmap. The orange cylinder represents the stationary obstacle. As the experiment proceeds, a second, mobile orange cylinder appears and forces UMan to extract a different path from the elastic roadmap.

UMan perceives the stationary and the moving obstacle using its laser range finder. It also uses the laser range finder to localize itself. This enables UMan to compensate for slippage of the wheels, which is necessary for it to stay on the virtual line throughout the experiment.

The experiment proceeds as follows. Figure 7(a) shows the initial configuration of the robot. In Fig. 7(b) the robot is still stationary but has attained the task constraint: the laser point is visible on the sphere. The elastic roadmap generated a hybrid system consisting of three milestones, the final one being the goal configuration. This feedback plan has UMan pass the orange cylinder on the right side, seen from the observer's perspective. Figure 7(c) shows how a second orange cone has moved into the robot's path. The current hybrid system is about to become invalid. In Fig. 7(d) the moving obstacle has cut off the path to the right of the stationary obstacle. Passing both obstacles on the right would force UMan to give up the task. The elastic roadmap framework extracts a new hybrid system that is able to maintain the task constraint. This new hybrid system has UMan pass the stationary obstacle on the left. Figures 7(e) and (f) show how UMan then proceeds to execute the motion described by the hybrid system until the end-effector reaches the goal-configuration on the line.

This experiment demonstrates that the elastic roadmap framework is able to maintain the motion constraints for autonomous mobile manipulation. All computations are performed in real-time. Throughout the entire experiment, the laser point remains on UMan's end-effector. This is further illustrated in Fig. 8, which shows as a function of time the x-, y-, and z-coordinate of the operational point in the center of the white sphere. The graph shows that the y-coordinate of the end-effector smoothly changes from its original position until the end of the five-meter virtual line has been reached. The changes in slope in the curve for the y-coordinate correspond to the switches among attractors. These switches either occur because a new hybrid system was generated in response to environmental changes or be-

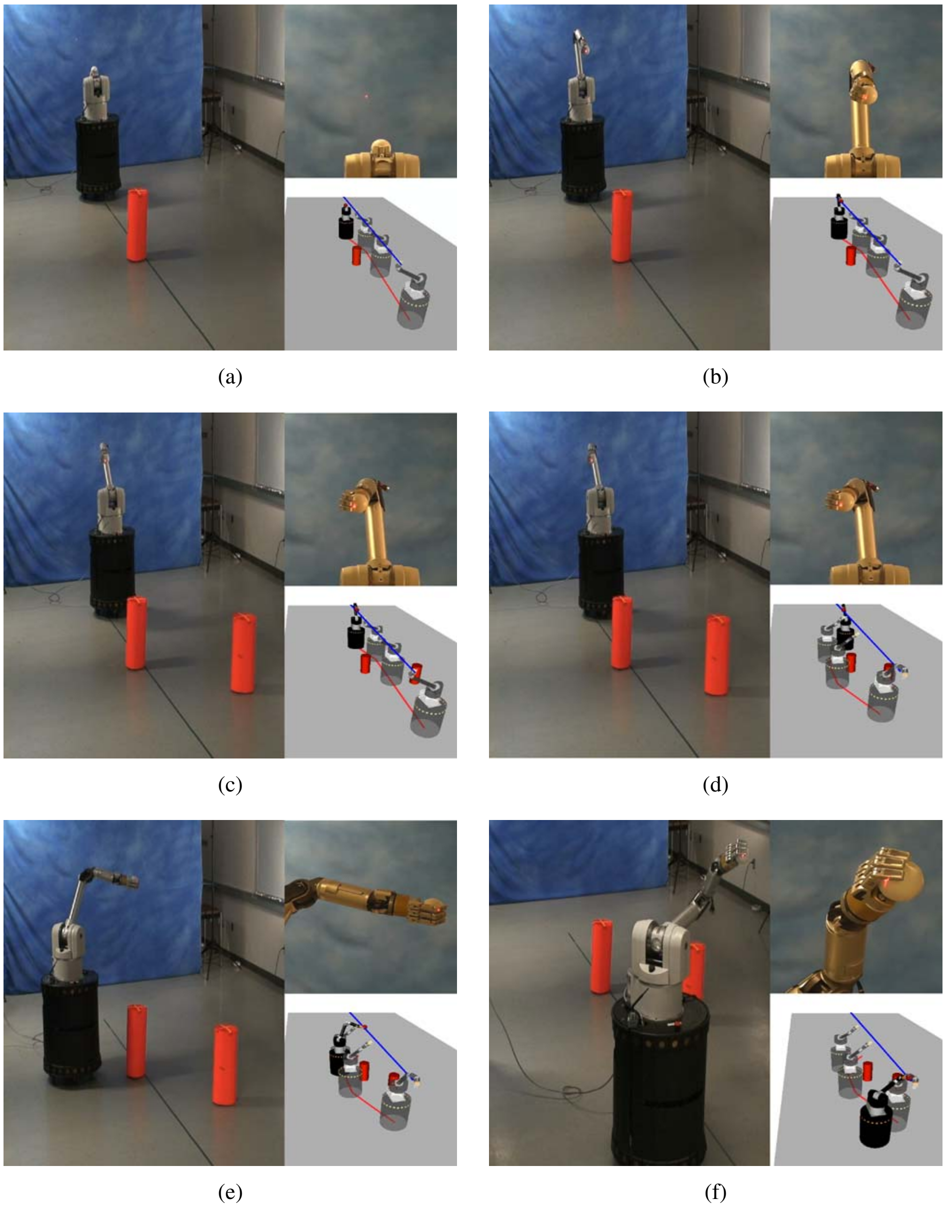


Fig. 7 Line-following experiment on the real robot

cause one of the milestones was reached during the motion. The x- and z-coordinate remain basically unchanged throughout the entire experiment. At $t = 28$ seconds there is a small dip in the curve for the x-coordinate, caused by the robot getting close to a joint limit. To avoid the joint limit, the joint limit controller takes over, briefly suspending the task and causing the dip. The joint limit is avoided and the task controller takes over once again, resuming the motion along the line. Note that the x-coordinate in the graph is recorded based on the localization of the base. Other small discontinuities in the graph for the x-coordinate are caused by a combination of controller error (negligible) and discrete changes in the x-coordinate due to re-localization.

Figure 9 shows the errors in the end-effector coordinates, again as a function of time. The errors are determined based on the encoder measurements relative to the localized base of the experimental platform. The errors remain for the most part are less than 2 cm. At $t = 28$ seconds, the switch in controller priorities and the resulting suspension of task execution results in an error magnitude of 3 cm.

Figure 10 shows the path of the center of UMan's base in the x/y-plane. UMan first attempts to pass the stationary obstacle on one side and is then forced by the moving obstacle to pass it on the other side. After the evasive maneuver, there is no need for the base to move back to the original x-coordinate, as all task and posture constraints are satisfied.

These experiments, and in particular the experiment on the real robot, represent realistic autonomous manipulation scenarios. They demonstrate that the elastic roadmap

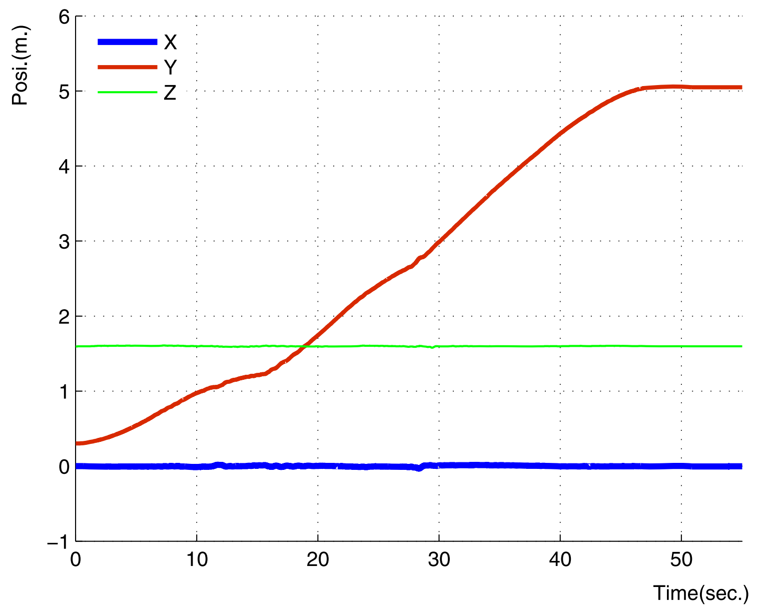


Fig. 8 (Color online) End-effector trajectory

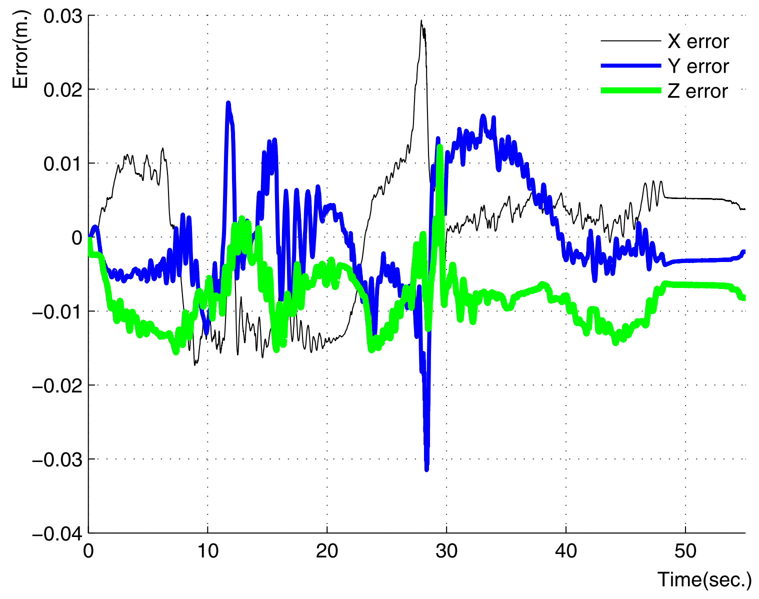


Fig. 9 (Color online) End-effector errors

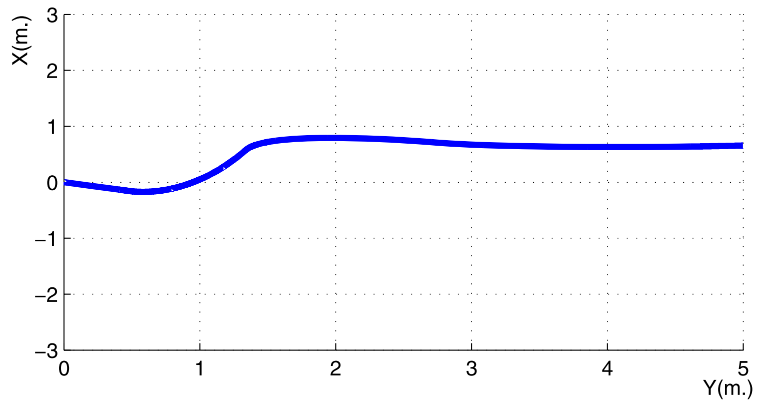


Fig. 10 UMan base trajectory

framework is able to maintain all motion constraints of autonomous mobile manipulation while satisfying their respective feedback requirements. These experiments, therefore, demonstrate the effectiveness of the elastic roadmap framework for the generation of constraint-consistent motion for autonomous mobile manipulation in dynamic environments.

6 Conclusion

Motion in the context of autonomous mobile manipulation is subject to numerous constraints. These constraints are imposed by the task, by kinematic and dynamic limitations of the robot, by moving obstacles in the environment, by the global connectivity of the workspace, and by subordinate behaviors, such as posture control. Existing approaches to motion generation for autonomous mobile manipulation either fail to address all motion constraints simultaneously or do not meet the respective feedback requirements.

We presented the elastic roadmap framework as a new approach to feedback motion planning. This framework satisfies all of the aforementioned constraints and their feedback requirements. Furthermore, the framework is capable of generating constraint-consistent motion in dynamic environments in free space and in contact with the environment in real time. To achieve the required computational efficiency, the elastic roadmap framework relies on two key ideas. First, it shifts the boundary between planning and control, relying on global planning to understand the connectivity of the space and on multi-objective operational space controllers to determine a motion for the robot that satisfies task and posture constraints and performs reactive obstacle avoidance. Second, the elastic roadmap framework relies on workspace information to efficiently determine and maintain connectivity information about the environment. This connectivity is captured in a roadmap that can be interpreted as a hybrid system of multi-objective controllers. The controllers determine the motion and represent the continuous part of the hybrid system. The discrete component switches between the controllers based on global connectivity information.

We presented experiments on a simulated mobile manipulator and a simulated stationary robot as well as on a real-world mobile manipulator. These experiments demonstrate that the elastic roadmap framework is a powerful motion generation framework which is well-suited for motion generation in the context of autonomous mobile manipulation.

Acknowledgements We gratefully acknowledge support by the Alexander von Humboldt foundation through the Alexander von Humboldt professorship, which is financed by the German Federal Ministry of Education and Research (BMBF). We equally gratefully acknowledge support by the National Science Foundation (NSF) under grants CNS-0454074 and IIS-0545934.

References

Amato, N., Bayazit, B., Dale, L., Jones, C., & Vallejo, D. (1998). OBPRM: An obstacle-based PRM for 3D workspaces. In *Robotics: The algorithmic perspective*. Wellesley: AK Peters.

Barraquand, J., & Latombe, J.-C. (1991). Robot motion planning: A distributed representation approach. *International Journal of Robotics Research*, *10*(6), 628–649.

Brock, O., & Grupen, R. (2005). Final report for the NSF/NASA Workshop on Autonomous Mobile Manipulation (AMM), November 2005. http://rbo.cs.umass.edu/amm/results.html.

Brock, O., & Kavraki, L. E. (2001). Decomposition-based motion planning: A framework for real-time motion planning in high-dimensional configuration spaces. In *Proc. int. conf. on robotics and automation*.

Brock, O., & Khatib, O. (2002). Elastic strips: A framework for motion generation in human environments. *International Journal of Robotics Research*, *21*(12), 1031–1052.

Burns, B., & Brock, O. (2005). Toward optimal configuration space sampling. In *Proceedings of robotics: Science and systems (RSS)*.

Burridge, R. R., Rizzi, A. A., & Koditschek, D. E. (1999). Sequential composition of dynamically dexterous robot behaviors. *International Journal of Robotics Research*, *18*(6), 534–555.

Chang, K.-S., & Khatib, O. (2000). Operational space dynamics: Efficient algorithms for modelling and control of branching mechanisms. In *Proc. int. conf. on robotics and automation* (pp. 850–856).

Chen, P. C., & Hwang, Y. K. (1998). SANDROS: A dynamic graph search algorithm for motion planning. *IEEE Transactions on Robotics and Automation*, *14*(3), 390–403.

Choi, W., & Latombe, J.-C. (1991). A reactive architecture for planning and executing robot motions with incomplete knowledge. In *Proc. int. conf. on intelligent robots and systems* (Vol. 1, pp. 24–29).

Choset, H., Lynch, K. M., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L. E., & Thrun, S. (2005). *Principles of robot motion*. Cambridge: MIT Press.

Conner, D. C., Rizzi, A. A., & Choset, H. (2003). Composition of local potential functions for global robot control and navigation. In *Proc. int. conf. on intelligent robots and systems* (pp. 3546–3551).

Conner, D., Choset, H., & Rizzi, A. (2006). Integrated planning and control for convex-bodied nonholonomic systems using local feedback control policies. In *Proceedings of robotics: Science and systems*, Philadelphia, USA, August 2006.

Connolly, C. I., & Grupen, R. A. (1993). One the applications of harmonic functions to robotics. *Journal of Robotic Systems*, *10*(7), 931–946.

Cortés, J., Jaillet, L., & Siméon, T. (2008). Disassembly path planning for complex articulated objects. *IEEE Transactions on Robotics*.

Diankov, R., Ratliff, N., Ferguson, D., Srinivasa, S., & Kuffner, J. (2008). Bispace planning: Concurrent multi-space exploration. In *Proceedings of robotics: Science and systems (RSS)*, ETH Zurich, June 2008.

Franklin, G. F., Powell, J. D., & Emami-Naeini, A. (1994). *Feedback control of dynamic systems*. Reading: Addison-Wesley.

Garber, M., & Lin, M. C. (2002). Constraint-based motion planning using Voronoi diagrams. In *Proc. of the workshop on the algorithmic foundations of robotics*.

Gayle, R., Klingler, K. R., & Xavier, P. G. (2007). Lazy reconfiguration forest (LRF): An approach for motion planning with mulitple tasks in dynamic environments. In *Proc. int. conf. on robotics and automation*.

Hauser, K., Bretl, T., Harada, K., & Latombe, J.-C. (2006). Using motion primitives in probabilistic sample-based planning for humanoid robots. In *Proc. of the workshop on the algorithmic foundations of robotics* (pp. 507–522).

Hsu, D., Kindel, R., Latombe, J.-C., & Rock, S. (2000). Randomized kinodynamic motion planning with moving obstacles. In *Proc. of the workshop on the algorithmic foundations of robotics* (pp. 247–264).

Hsu, D., Latombe, J.-C., & Kurniawati, H. (2005a). On the probabilistic foundations of probabilistic roadmap planning. In *Proceedings of the international symposium of robotics research*.

Hsu, D., Sánchez-Ante, G., & Sun, Z. (2005b). Hybrid prm sampling with a cost-sensitive adaptive strategy. In *Proc. int. conf. on robotics and automation*.

Huber, M., & Grupen, R. A. (1997). A feedback control structure for on-line learning tasks. *Robotics and Autonomous Systems*, *22*(3–4), 303–315.

Jaillet, L., & Siméon, T. (2004). A PRM-based motion planner for dynamically changing environments. In *Proc. int. conf. on intelligent robots and systems*.

Jaillet, L., & Siméon, T. (2008). Path deformation roadmaps: compact graphs with useful cycles for motion planning. *International Journal of Robotics Research*.

Kallmann, M., & Matarić, M. (2004). Motion planning using dynamic roadmaps. In *Proc. int. conf. on robotics and automation* (pp. 4399–4404).

Kavraki, L. E., Švestka, P., Latombe, J.-C., & Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, *12*(4), 566–580.

Khatib, O. (1987). A unified approach to motion and force control of robot manipulators: The operational space formulation. *International Journal of Robotics and Automation*, *3*(1), 43–53.

Kuffner, J., & LaValle, S. M. (2000). RRT-connect: An efficient approach to single-query path planning. In *Proc. int. conf. on robotics and automation* (Vol. 2, pp. 995–1001).

Kurniawati, H., & Hsu, D. (2004). Workspace importance sampling for probabilistic roadmap planning. In *Proc. int. conf. on intelligent robots and systems* (pp. 1618–1623).

LaValle, S. M. (2006). *Planning algorithms*. Cambridge: Cambridge University Press.

Leven, P., & Hutchinson, S. (2002). A framework for real-time path planning in changing environments. *International Journal of Robotics Research*, *21*(12), 999–1030.

Lindemann, S. R., Hussein, I. I., & LaValle, S. M. (2006). Real time feedback control for nonholonomic mobile robots with obstacles. In *Proceedings of the IEEE conference on decision and control* (pp. 2406–2411). San Diego, USA, December 2006.

Morales, M., Tapia, L., Pearce, R., Rodriguez, S., & Amato, N. M. (2004). A machine learning approach for feature-sensitive motion planning. In *Proc. of the workshop on the algorithmic foundations of robotics*, Utrecht/Zeist, The Netherlands.

Oriolo, G., & Mongillo, C. (2005). Motion planning for mobile manipulators along given ed-effector paths. In *Proc. int. conf. on robotics and automation* (pp. 2166–2172). Barcelona, Spain.

Plaku, E., Kavraki, L. E., & Vardi, M. Y. (2007). Discrete search leading continuous exploration for kinodynamic motion planning. In *Proceedings of robotics: Science and systems (RSS)* (pp. 313–320). Atlanta, USA, June 2007.

Quinlan, S., & Khatib, O. (1993). Elastic bands: Connecting path planning and control. In *Proc. int. conf. on robotics and automation* (Vol. 2, pp. 802–807). Atlanta, USA.

Rimon, E., & Koditschek, D. E. (1992). Exact robot navigation using artificial potential fields. *IEEE Transactions on Robotics and Automation*, *8*(5), 501–518.

Sato, K. (1987). Collision avoidance in multi-dimensional space using Laplace potential. In *Proceedings of the 15th conference of the robotics society of Japan* (pp. 155–156).

Sentis, L., & Khatib, O. (2005). Synthesis of whole-body behaviors through hierarchical control of behavioral primitives. *International Journal of Humanoid Robots*, *2*(4), 505–518.

Siméon, T., Laumond, J.-P., & Nissoux, C. (2000). Visibility-based probabilistic roadmaps for motion planning. *Journal of Advanced Robotics*, *14*(6), 477–494.

Siméon, T., Laumonde, J.-P., Cortés, J., & Sahbani, A. (2004). Manipulation planning with probabilistic roadmaps. *International Journal of Robotics Research*, *23*(7–8), 729–746.

Stilman, M. (2007). Task constrained motion planning in robot joint space. In *Proc. int. conf. on intelligent robots and systems*, October 2007.

Tang, X., Thomas, S., & Amato, N. M. (2005). Planning with reachable distances: Fast enforcement of closure constraints. In *Proc. int. conf. on robotics and automation* (pp. 2694–2699). Rome, Italy.

van den Berg, J. P., & Overmars, M. H. (2004). Using workspace information as a guide to non-uniform sampling in probabilistic roadmap planners. In *Proc. int. conf. on robotics and automation* (pp. 453–460).

van den Berg, J. P., & Overmars, M. H. (2005). Roadmap-based motion planning in dynamic environments. *IEEE Transactions on Robotics and Automation*, *21*(5), 885–897.

Vannoy, J., & Xiao, J. (2004). Real-time adaptive and trajectory-optimized manipulator motion planning. In *Proc. int. conf. on intelligent robots and systems* (Vol. 1, pp. 497–502).

Yang, Y., & Brock, O. (2004). Adapting the sampling distribution in PRM planners based on an approximated medial axis. In *Proc. int. conf. on robotics and automation* (pp. 4405–4410).

Yang, Y., & Brock, O. (2005). Efficient motion planning based on disassembly. In *Proceedings of robotics: Science and systems (RSS)*.

Yang, Y., & Brock, O. (2006). Elastic roadmaps: Globally task-consistent motion for autonomous mobile manipulation in dynamic environments. In *Proceedings of robotics: Science and systems (RSS)*.

Yang, L., & LaValle, S. M. (2003). The sampling-based neighborhood graph: A framework for planning and executing feedback motion strategies. In *Proc. int. conf. on robotics and automation*.

Yao, Z., & Gupta, K. (2007). Path planning with general end-effector constraints. *Robotics and Autonomous Systems*, *55*(4), 315–327.

Zucker, M., Kuffner, J., & Bagnell, J. (2008). Adaptive workspace biasing for sampling-based planners. In *Proc. int. conf. on robotics and automation* (pp. 3757–3762). Pasadena, CA, May 2008.

Zucker, M., Kuffner, J., & Branicky, M. (2007). Multipartite RRTs for rapid replanning in dynamic environments. In *Proc. int. conf. on robotics and automation* (pp. 1603–1609). Roma, Italy, April 2007.

Yuandong Yang is a Ph.D. candidate in Computer Science at the University of Massachusetts Amherst. He received his B.Sc. from the Automation Department of Tsinghua University in 1999 and a Master's degree from the Computer Science and Technology Department of Tsinghua University in 2001. He currently works for Microsoft. His research focuses on motion generation for Autonomous Mobile Manipulation.

Oliver Brock is a Professor and member of the Faculty of Electrical Engineering and Computer Science at the Technische Universität Berlin in Germany. He received his Diploma in Computer Science from the Technische Universität Berlin and his Master's and Ph.D. in Computer Science from Stanford University. He also held post-doc positions at Rice University and Stanford University. He was an Assistant Professor and Associate Professor in the Department of Computer Science at the University of Massachusetts Amherst, prior to moving back to the Technische Universität Berlin. The research of Brock's lab, the Robotics and Biology Laboratory, focuses on autonomous mobile manipulation and the application of algorithms and concepts from robotics to problems in structural molecular biology.