2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Sensor-Based, Task-Constrained Motion Generation Under Uncertainty

Arne Sieverling Nicolas Kuhnen Oliver Brock[1]

***Abstract*— Mobile manipulation targets applications in dynamic and unstructured environments. Motion generation methods suitable for these applications must account for end-effector task constraints, must reason about environment uncertainty, i.e. the fact that the exact state of the dynamic environment cannot be known to the robot, and should do so only using their on-board sensors. We present the Expected-Shortest-Path Elastic Roadmap (ESPER) planner as a motion generation method suitable for mobile manipulation. It integrates task-constrained, whole-body, reactive motion generation in high-dimensional configuration space with reasoning about uncertainty. In our experiments, we generate task-consistent motion in uncertain environments on a real-world mobile manipulator only relying on on-board sensors.**

I. Introduction

A successful motion generation system for mobile manipulation in unstructured environments must—in addition to generating global, task-consistent motion—also address issues of uncertainty and perception. Each of these problems has received significant attention in the robotics community, however, little work in mobile manipulation attempts to address all three at the same time in practical, real-world scenarios. This is the goal of this paper.

We will start from a seemingly restrictive assumption, namely, that the mobile manipulator will not have a complete model of the world and can only rely on high-quality on-board sensors for perception. As we will see, this assumption enables us to address issues of uncertainty efficiently, by adapting a polynomial-time algorithm for the expected-shortest-path problem [1]. Relying on our previous work on elastic roadmaps [2], which achieves global, task-consistent motion in dynamic environments, we develop an approach to global, task-consistent motion under uncertainty, informed solely by on-board sensing.

We establish the following requirements for motion generation under uncertainty in the context of mobile manipulation:

- R1 *Motion:* The robot must perform global, collision-free motion while maintaining task constraints, e.g., move to the kitchen while holding a glass of water.
- R2 *Uncertainty:* The robot must reason about the uncertainty associated with unstructured environments.
- R3 *Perception:* The robot may only derive information about the world from on-board sensors.

We gratefully acknowledge the funding provided by the Alexander von Humboldt foundation and the Federal Ministry of Education and Research (BMBF) and through the First-MM project (European Commission, FP7-ICT-248258). We thank SimLab for their support.

[1]All authors are members of the Robotics and Biology Laboratory at the Technische Universität Berlin, Germany.

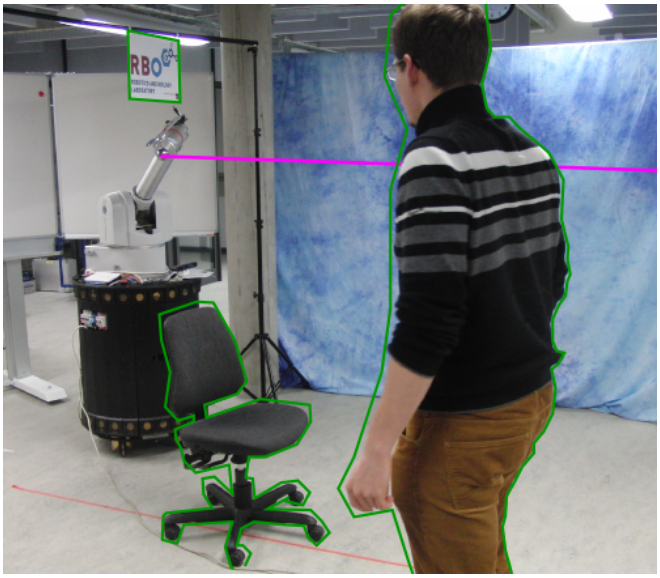


Fig. 1. A motion generation problem for a mobile manipulator in an uncertain environment. The robot has to move the end-effector on a virtual pink line. Additionally, it has to perceive and avoid three obstacles with its onboard sensors: a rolling chair, a sign hanging from the ceiling, and a moving person.

Satisfying any single one of these requirements already is challenging. Motion planning (R1) was proven to be PSPACE-complete [3]. POMDPs are a general framework for addressing uncertainty (R2), but the computation of exact solutions is intractable, even for small state spaces [4]. And complete perception of the world (R3) with only local sensors is physically impossible. We therefore know that we cannot hope for complete solutions but instead must resort to approximations and heuristics to be able to address real-world problems.

In the remainder of this paper, we will present a practical approach to motion generation under uncertainty for mobile manipulation, and we will demonstrate the effectiveness of this approach on a real-world robotic platform. The key contributions of this paper are (i) the formulation of a practical approach for combining task constrained whole-body motion with efficient reasoning about uncertainty and (ii) to demonstrate the effectiveness of this approach in simulation and in real-world experiments.

II. Related Work

We first revisit the motion and perception components of current mobile manipulation systems and analyse how they handle uncertainty. Then, we introduce planning methods under uncertainty and discuss their real-world applications.

A. Mobile manipulation planning systems

First we analyze the motion generation capabilities of two state-of-the-art mobile manipulation systems [5, 6]. Both systems handle unstructured environments by constructing an environmental representation during task execution. To account for unforeseen obstacles, they monitor motion execution and replan in case of invalidated solutions. Apart from grasp planning (which is not covered in this paper), the generation of robot motion does not actively consider uncertainty. In addition, unlike our approach, these systems generate motion for arm and base separately. For base navigation, three-dimensional obstacles were taken into account [7].

Belief space HPN (hierarchical task and motion planning in the now) [8] is an integrated task and motion planning method under uncertainty. The method plans sensing and motion actions for complex interactions with the environment by planning on symbolic abstractions of the belief space. Belief space planning is in general of high computational complexity; as a result, this method is not well-suited for applications in dynamic environments, where new plans must be generated several times per second. Our method is complementary to this approach. We generate robot motion that satisfies the constraints associated with manipulation tasks without the necessity to reason in the high-dimensional belief spaces that arise in manipulation planning.

B. Planning under uncertainty

Now we will introduce planning methods that reason about uncertainty and discuss successful applications to robotics. In this discussion, we will distinguish four kinds of uncertainty: sensing uncertainty (sensor measurements are noisy), action uncertainty (actions are stochastic), state uncertainty (the state is not known exactly), and environment uncertainty (the world model is not known exactly). Most approaches address only some of these; in contrast, the method presented in his paper addresses all types of uncertainty while at the same time remaining computationally efficient.

Planning with feedback: Feedback plans map each state of the robot to an action [9, 10]. As a result, feedback plans accommodate action uncertainty: no matter what the outcome of an action, the plan specifies a next step to make progress towards the goal. FIRM [11] generates a feedback motion plan as a roadmap of controllers over the belief space, addressing the robot's state uncertainty. Feedback motion planners rely on known and static environments. Our method instantiates a similar graph-based composition of controllers based on the Elastic Roadmap framework [2] for whole-body, task-consistent motion in dynamic environments. We do not plan in belief space but we will continuously employ feedback to adapt the plan relative to sensed obstacles.

Planning with Markov Decision Processes: Markov Decision Processes (MDP) [12] model actions with stochastic outcomes. Several approaches use this formulation to compute motion policies under action uncertainty [13, 14], or environment uncertainty [15, 16]. Value iteration minimizes expected costs efficiently but does not explicitly account for the robots sensing limitations.

The Expected Shortest Path (ESP) problem [1] is of particular relevance in the context of this paper. It is also formulated as an MDP but instead of action uncertainty it handles environment uncertainty. The ESP computes a policy that returns the action with minimal expected costs for each sensory input. This differs from the other MDP approaches, which do not consider the robot's ability to gather information during motion when computing the expected costs.

Planning with partial observability: The partially observable MDP (POMDP) extends the MDP formulation with uncertainty about the current and future states of the robot. Solving POMDPs requires solving an MDP over an exponentially large belief space and is in general intractable [4]. Approximate solutions were successfully applied to 2D mobile robot navigation [17] and motion planning under sensing and action uncertainty [18].

A related, exponentially complex approach models the state of each edge in the MDP as a Markov Chain so as to address environment uncertainty. Policies for 2D navigation were found by planning in the belief space over a reduced graph [19], or by iterating over discrete time-steps [20]. We also model the state of an edge in the MDP as a continuous time Markov Chain, but we only consider a class of environments for which a polynomial-time solution exists.

III. Expected Shortest Path Elastic Roadmap

Now we will present the core concepts of the Expected Shortest Path Elastic Roadmap planner (ESPER). First, we will recapitulate the main ideas of the Elastic Roadmap framework to show how it addresses requirement R1 and addresses state uncertainty. We then extend the Elastic Roadmap with an efficient planning algorithm in uncertain environments to address requirement R2. Finally, we show how ESPER addresses requirement R3 by instantiating an elastic roadmap based on local sensor data.

A. The Elastic Roadmap

The previously published Elastic Roadmap framework [2] is an efficient, incomplete feedback motion planning approach that generates reactive, task-constrained, whole-body motion for mobile manipulation tasks. It captures the connectivity of the workspace in a roadmap that is incrementally modified in response to perceived changes in the environment. As these changes in the environment move milestones and edges of the roadmap, it appears to be "elastic", hence the name.

An Elastic Roadmap is a graph $G = (V,E)$, consisting of a set of milestones (vertices) $V = \{v_1, v_2, \ldots\}$ and edges $e_{ij} = (v_i, v_j) \in E$. An edge e_{ij} indicates that the milestone v_i is in the region of attraction of the controller associated with v_j, i.e. invoking v_j's controller while the robot is in state v_i will cause the robot to move to v_j. The controllers used in the Elastic Roadmap are task-consistent, whole-body, operational-space controllers. Task constraints are therefore maintained automatically as the robot moves through the

roadmap. This roadmap captures important, task-relevant aspects of the workspace connectivity.

To achieve the elastic effect, every milestone v_i is maintained by a separate task-consistent, whole-body controller, whose attractor is associated with environmental features. As a result, each milestone represents a task-consistent via point in configuration space. When obstacles move, nearby milestones will move with them while maintaining task constraints. As milestones move, the planner updates the connectivity between milestones continuously, maintaining the roadmap.

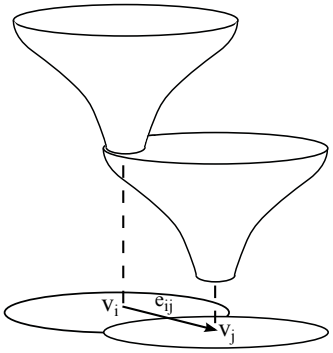


Fig. 2. Two milestones v_i and v_j of an Elastic Roadmap and an edge e_{ij} connecting them: the funnels visualize the controllers' domain of attraction; the exit of the funnel is the controller's attractor; a roadmap represents a feedback plan as a network of funnels.

The Elastic Roadmap planner covers large parts of the configuration space with the regions of attractions of milestones, using workspace information to identify task-relevant configuration space regions. The roadmap G define a feedback plan over the state space (see Fig. 2). Coverage of configuration space will never be complete, and hence the Elastic Roadmap provides no completeness guarantees. It consciously gives up completeness to satisfy the real-time requirements for motion in dynamic environments.[1]

The Elastic Roadmap planner addresses current and future state uncertainty, action uncertainty, and sensing uncertainty through the use of feedback control. The controllers associated with milestones continuously transfer states in their regions of attraction into the attractor states. The attractor state itself also adapts to the uncertain or dynamic environment and keeps milestones valid for long periods of time. As the robot moves from attractor to attractor, we assume that every underlying controller uses feedback on environmental features. We assume no uncertainty about the robots position with respect to the next intermediate goal. This assumption is justified, if the environment provides sufficient features for each controller to uniquely identify its attractor from sensor data. This assumption is mostly likely valid in the cluttered environments of mobile manipulation applications. Effectively, controllers continuously reduce state uncertainty as the robot moves through the roadmap. We exploit this assumption and do not reason about robot state uncertainty during planning.

[1]Completeness guarantees can only be useful when the state of the world is perfectly known; this will never be the case in our application domain.

B. Planning under environment uncertainty

In this section, we extend the Elastic Roadmap planner to incorporate environment uncertainty. Planning in uncertain environments is a hard problem. To solve it optimally, a planner would need to plan on belief space capturing all possible states of the world. This is computationally infeasible. We reduce the complexity of the problem by making a simple, appropriate, and justified assumption about the sensor capabilities of the robot: we assume that a mobile manipulator can only obtain information about the world through its onboard sensors. We also assume that the robot possesses some kind of global map about the environment, containing only stationary obstacles. (This map could also be acquired by the robot online; for the sake of clarity, we will assume the availability of a static map.)

Based on these two assumptions, we decompose planning under uncertainty in two sub-problems: *1)* reasoning about uncertainty within the local, perceivable region, and *2)* reasoning about uncertainty in the remainder of the space. The local region around the robot is bounded by a "horizon of relevance", influenced by the sensor range, acceleration capabilities, and the environment. In this local region, we rely on sensor feedback to address uncertainty, as explained in the previous section. In the global region, we explicitly reason about environment uncertainty, as will be explained in this section.

Local region: In the local region, the robot can decide if a given motion alternative will violate task-constraints, result in collision, or will be valid. We call this decision a *local decision*. The robot can perform local decisions efficiently based on timely sensor data. The decisions affect only ongoing actions. We execute these local decisions continuously with the sensor update rate of 10 Hz. The constant use of feedback eliminates the need to explicitly reason about state, action, and sensing uncertainty.

Global region (Time-Dependent Expected Shortest Path): What does it mean to reason about uncertainty in the global region? We have a global map of the static part of the environment. Within this map, we have an elastic roadmap, capturing the connectivity of the space. Uncertainty stems from the possibility of any of the edges of the roadmap being blocked by unobserved, moving obstacles. We associate with each edge in the roadmap information about it's "blocking characteristics" (defined in detail below).

Reasoning about uncertainty during the planning process can now be viewed as follows (please refer to Figure 3): In the local region around milestone n, we use sensor information to make local decisions. The cost of going to milestones a, b, and c is known, as the costs l_{ni} between milestones n and i can be determined from sensor data. However, the overall expected cost to the goal, which our algorithm seeks to minimize, also depends on the remainder of the path to the goal milestone g. The key now is to reason about the expected cost E_{ig} to travel through the roadmap

from each of the three milestone a, b, c to the goal g. The parts of the elastic roadmap that have to be traversed after the local decision has been made, are shown as clouds in Figure 3).

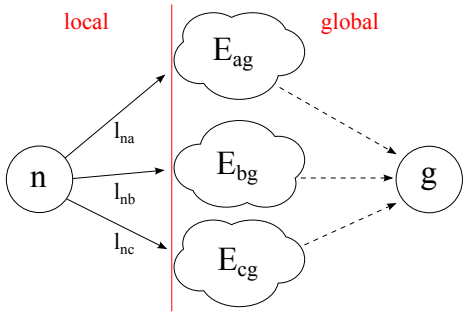


Fig. 3. Local decisions for a problem with three alternatives. The robot is at node n. The clouds represent the multitude of paths to the goal, after reaching a, b, or c. Knowledge about the expected outcome of future actions is reflected in E_{ig}.

The expected cost E_{ig} depends on the graph structure of the roadmap and the blocking probabilities of the edges contained in it. It is easy to see how blocking probabilities affect the cost, as blocked edges cannot be traversed and may lead to detours. Figure 4 illustrates the effect of the graph structure on the expected cost. Assuming equal blocking probabilities for all edges, the route from v_s to v_g through v_1 has lower expected cost than the route through v_2, as the probability of both routes following v_1 being blocked is lower than the probability of the the edge following v_2 being blocked.

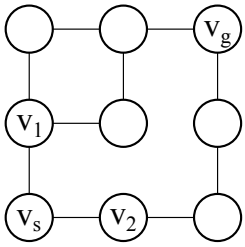


Fig. 4. Effect of graph structure on expected costs: The path from v_s to v_g via v_1 provides two alternative paths to g with equal length, while the path via v_2 provides only one. The expected cost of the path via v_1 is lower because the robot can avoid one possibly blocked edge following v_1.

In our setting, we use time as the cost, seeking to minimize the time to reach the goal. To estimate the expected travel duration along the entire path, we derive a model for the blocking probability of edges from sensor observations. We model the unknown state of an unobserved edge as a continuous-time Markov Chain with two states: *blocked* and *free*. Three parameters characterize every edge $e = (i, j)$:

- l_{ij}: the expected time it takes to traverse e,
- t^B_{ij}: the expected duration the edge is *blocked*, and
- t^F_{ij}: the expected duration the edge is *free*.

In the current setting, determining the expected cost requires long-term reasoning about the state of edges. That means that we have to reason about all possible blocked/unblocked states of all edges based on the robots observation history. This forces us into a POMDP scenario [19], making the solution too computationally complex to be applied to dynamic environments. However, our assumption that the world consists of a static map and moving obstacles allows us to devise a more effective solution. We assume that edges are never blocked for time periods much longer than the robot requires to move along the edge, i.e. $t^B_{ik} < l_{ij}$ for all pairs of edges $(i, j), (i, k)$. And we remove edges that are blocked for very long time periods from the roadmap. Additionally, we remove edges which are blocked so frequently that they can not be traversed safely ($t^F_{ij} < l_{ij}$). Also, once the robot observes the state of an edge, it is not considered to change again. Based on these assumptions, it suffices to reason about the currently known state of the world, as opposed to all possible states. This is the key step to addressing environment uncertainty efficiently.

Given this assumption, the probability p_{ij} that an edge is traversable is given by the stationary distribution of the Markov chain:

$$p_{ij} = \frac{t^F_{ij}}{t^F_{ij} + t^B_{ij}}. \tag{1}$$

Now p_{ij} and l_{ij} are known, and we can apply the polynomial-time Expected Shortest Path (ESP) algorithm, which is based on dynamic programming, to compute the expected cost E_{ig} for every node i [1].

We can add an additional source of information to improve the policies of the Expected Shortest Path algorithm: in our time-dependent model, the robot can estimate the time each blocked edge (i, j) takes to become unblocked by t^B_{ij}. This time estimate enables an additional motion alternative: wait for a promising edge to become unblocked. For each edge the robot has two options:

- If the edge is *free*, the robot can move along it right away. This action is possible with probability p_{ij} and takes time l_{ij}.
- If the edge is *blocked*, the robot can wait until it becomes unblocked and then move along it.[2] This is possible with probability 1 and we estimate the time by $l_{ij} + t^B_{ij}$.

The optimal policy follows immediately: The robot executes the controller associated with milestone k, where

$$k = \underset{i \in V}{\operatorname{argmin}} \begin{cases} l_{ni} + E_{ig} & \text{and } (n,i) \text{ is } \textit{free} \\ l_{ni} + t^B_{ni} + E_{ig} & \text{and } (n,i) \text{ is } \textit{blocked} \end{cases} \tag{2}$$

In the second case, the robot will wait until the edge state changes from blocked to free.

The ESP without the new waiting actions can be solved in $O(|E| \cdot \log(|V|))$ with a label-correcting algorithm [21]. Adding the waiting actions doubles the amount of edges, but does not increase the runtime.

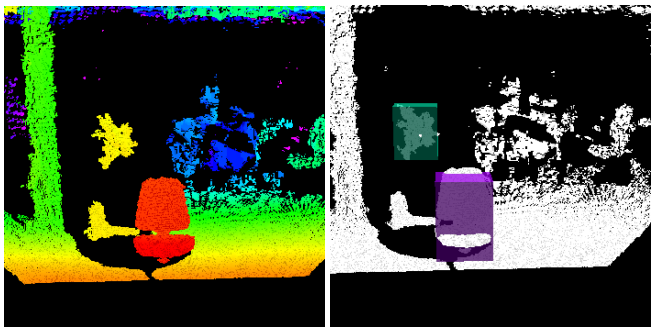
[2]These actions replace the waiting edges (i, i) of the original expected shortest path (ESP) algorithm. They capture a new option for the robot that is not possible with the original ESP formulation: it might be useful to wait a short time for an edge to become unblocked instead of taking the next best free edge.

C. Perception

An important part of the proposed algorithm is the integration with real-world sensing. To be suitable for mobile manipulation applications, a motion generation method should only rely on onboard sensor information during execution.

ESPER uses onboard sensor information to place and adjust milestones in the roadmap. To place milestones efficiently, it needs to be able to recognize obstacle boundaries. To use the feedback capabilities of the milestones it also needs to track dynamic obstacles with high frequencies and update the milestone controllers accordingly.

ESPER instantiates the obstacle-based roadmap with a simple and fast obstacle tracking method based on RGB-D data. For background subtraction (including the floor) ESPER uses a pre-recorded 3D-occupancy grid of the static part of the environment. Then it aligns the depth data from a RGB-D sensor to this occupancy grid. To align with the global grid it uses a laser range finder and Adaptive Monte Carlo Localization [22]. ESPER then detects all 3D points of the sensor point cloud that do not appear in the occupancy grid and clusters them into point cloud regions (see Fig. 5). Each of these point clusters is considered to be an an obstacle. Note that the global localization is only used for two parts in the ESPER framework: for background subtraction and for the execution of globally defined tasks. In general, ESPER does not depend on global localization because milestones are only defined relative to environmental features and not with respect to global coordinates.



(a) Point cloud (b) Bounding boxes

Fig. 5. a) Point cloud obtained with the robot's on-board sensor, colored by depth (b) two bounding boxes returned by the segmentation algorithm around dynamic obstacles not contained in the map

The algorithm tracks dynamic obstacles by comparing the centroids of the point cloud regions to the position of regions in the previous frame. The segmentation processes 10 frames per second and is able to track obstacles moving at 1 m/s. The ESPER planner computes an axis-aligned bounding-box around the point cloud region and places milestones at the corners of each box. We instantiate each milestone with a multi-priority, whole-body controller to obtain task-consistent intermediate configurations that dynamically avoid obstacles (see Section IV for details). Using these milestones, we can then estimate the edge parameters t^B_{mn} and t^F_{mn}. To estimate if an edge is traversable, we observe the amount of free workspace between two milestones. In our world representation, we cast rays between sampled points on the milestone configurations and check for collisions with the bounding boxes. We do these tests continuously for each edge whenever it is inside the sensor range and store the duration that each edge is consecutively blocked or free. We compute the expected durations t^B_{mn} and t^F_{mn} by averaging over all of these time measurements.

IV. Experiments

Now we will present three experiments to show that ESPER satisfies the three initially stated main requirements. The first experiment concerns requirement R1: task-consistent whole-body motion in dynamic environments. The second experiment will show how learning a probabilistic transition model and reasoning about uncertainty generates faster and safer motion, showing that requirement R2 is addressed. The third experiment shows how all necessary information for ESPER can be extracted from on-board sensor data (requirement R3) and how motion is executed on a real mobile manipulator.

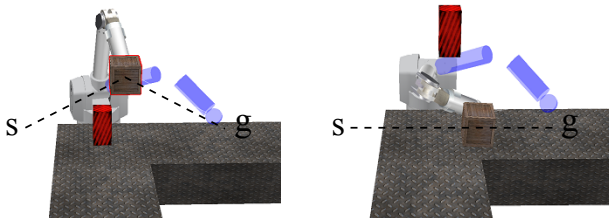
All experiments are based on a C++ implementation of the ESPER planner on a standard Windows desktop PC. For the first two experiments, the motion is executed in simulation (the dynamic simulation runs on a different PC). In the third experiment, we use a mobile manipulation platform, consisting of holonomic base (Nomadic XR4000) and a seven degree-of-freedom manipulator (Barrett WAM). The planner continuously plans at 1 Hz. The majority of runtime is spent on milestone maintenance, the time for computing the Expected Shortest Path never exceeds 100 ms.

Milestone maintenance is performed using multi-objective control at acceleration level [23]. For motion execution, we employ control primitives that realize four objectives each on a different level of priority: the end-effector task is the highest priority, obstacle avoidance, sensor orientation towards the goal, and null space posture on lower priorities. We implement task constraints, obstacle avoidance, and sensor orientation as attractive or repulsive potentials with velocity saturation. The line task, used in the third experiment, is a 2D controller acting on the end-effector. Obstacle avoidance is realized with repulsive potentials acting on base and elbow of the robot. The sensor orientation is maintained using a potential function acting on the rotational joint of the base. We realized the posture constraint as an interpolated joint space controller to achieve smooth behavior in the nullspace of all other objectives. We obtain the desired intermediate postures from the converged milestone controllers. All controllers and the dynamic simulation are implemented in RoboticsLab, a robotic tool kit by SimLab.

A. Reactive task-consistent whole-body motion

In this simulated experiment, we demonstrates that ESPER generates task-consistent whole-body motion (requirement R1). A stationary seven-DOF manipulator has to move a box mounted to its end-effector from left to right over a moving conveyor belt containing red obstacles. The task

also requires the orientation of the box to remain fixed. We deliberately use a stationary manipulator in this experiment to show that it has to use all of its seven degrees of freedom to avoid obstacles while executing the task. It also demonstrates that ESPER can successfully be applied to stationary scenarios.



(a) Motion via intermediate milestones above the obstacle

(b) Motion below the obstacle while avoiding it with the elbow

Fig. 6. Motion execution for a stationary manipulator moving a wooden box among dynamic red obstacles using ESPER: the robot has to move the box from its initial position s to the final position g; the milestone at the goal configuration is shown in blue; ESPER generates different motions in response to changes in the environment; the task constraint is maintained throughout the entire motion.

Figures 6(a) and 6(b) show how the whole kinematic chain avoids the dynamic obstacles. When the obstacles block the lower path, the global plan guides the arm above the obstacles to avoid collisions. When the obstacles move to the upper level, the motion instantly adapts. Now the robot reactively avoids colliding with the boxes with its elbow using all of its seven degrees of freedom to execute the task. Still, during the complete motion, the end-effector orientation remains constant.

This experiment shows that ESPER generates whole-body motion under task constraints in scenarios without uncertainty. Similar results can be achieved with mobile manipulators [2].

B. Reasoning about uncertain environments

In this experiment, we will show how requirement R2—explicit handling of uncertainty—reduces execution time and collision rates compared to uncertainty-unaware planning. A ten-DOF mobile manipulator moves in a dynamic environment (Fig. 7(a)), while holding a tray with objects level. Two obstacles, O_1 and O_2, move from wall to wall but always keep a fixed distance d. In this scenario, both the paths above (referring to the figure) O_1 and below O_2 are invalid about half of the time.

ESPER places milestones around the two dynamic obstacles. It observes the moving obstacles for 10 seconds and then computes a higher collision probability for the upper and lower edges shown in yellow. The path between the two obstacles is valid all the time and thus has a lower collision probability (shown by the red lines). Fig. 7(b) shows a snapshot of the roadmap created with ESPER during execution. The edge probabilities qualitatively match the expected behavior from Fig. 7(a).

We compared ESPER to an uncertainty-unaware implementation of the Elastic Roadmap (*ER*). This planner uses the shortest path over the milestones it believes to be passable at planning time and recomputes a new path every second. We executed the motion in 50 simulation experiments with randomized obstacle positions.

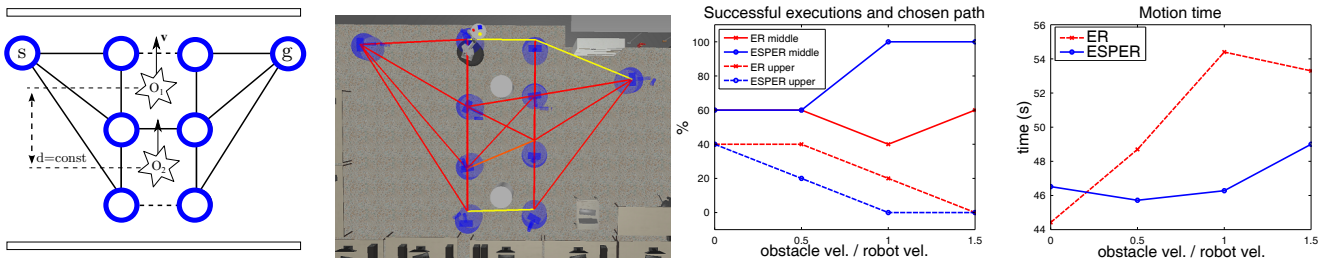
For low obstacle velocities, both planners perform roughly equal and both planners choose paths above O_1 and between O_1 and O_2. For higher obstacle velocities, the path above O_1 becomes blocked often. The uncertainty-unaware elastic roadmap planner still chooses both paths, depending on the exact obstacle positions. (Fig. 7(c)). In 20% of the runs with high velocities, the robot gets stuck between the upper wall and O_1, leading to a collision. In contrast, the ESPER planner always chooses the path between the obstacles, which is safe even at high obstacle velocities (no collisions). The overall motion time for the non-colliding runs is also lower for the uncertainty-unaware planner (see Fig. 7(d)).

This experiment shows that explicit handling of environment uncertainty reduces motion time and can even help to reduce the probability of collisions.

C. Sensor-based planning and execution

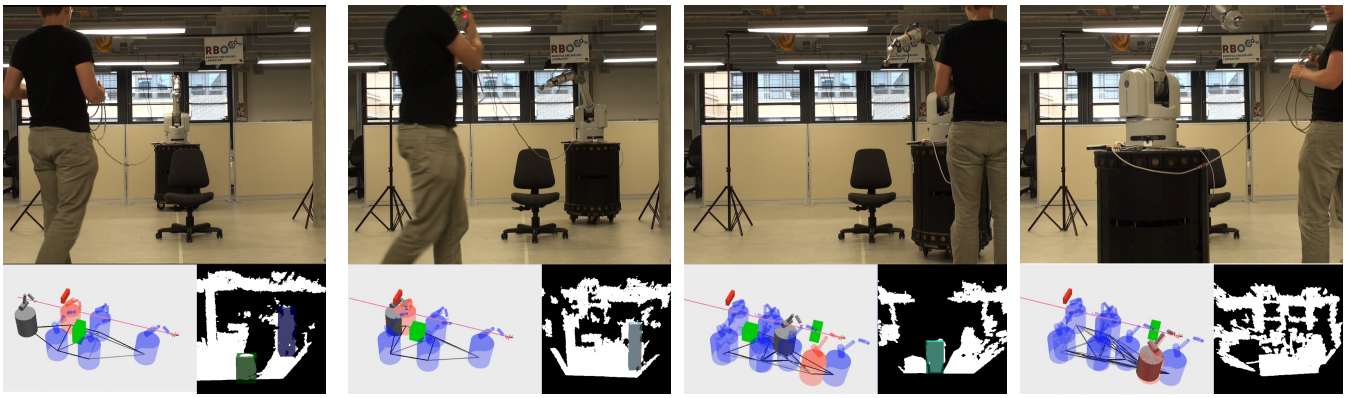
In the last experiment, we will combine the established requirements R1 and R2 with the requirement for perceiving the world only with on-board sensors. Our experimental platform is a 7-DOF Barrett WAM robotics arm mounted on a Nomadic XR-4000 holonomic mobile base. The platform has two onboard sensors: a SICK laser range finder and an Asus Xtion RGB-D sensor. The robot is placed in the starting configuration shown in Figure 8(a) and receives the task to move 4 meters forwards while keeping its end-effector on a virtual line.

The robot creates a static map from its sensor data consisting of the floor, the sign on the ceiling, and the columns for localization and background filtering. Afterwards two unknown obstacles: a chair and a person, enter the environment. The rolling chair is in front of the robot and static for the rest of the experiment. The ESPER planner generates task-consistent intermediate milestones on both sides of the chair. In front of the chair the person moves continuously on the right side of the line. Initially, the robot observes this scene for 20 seconds. During this time, the robot observes the person. The planner estimates the edge parameters t^F and t^B, which it uses in the computation of the expected shortest path. The resulting roadmap from Figure 8(a) shows a higher value for t^B on the edges that were blocked by the person (visible by the lighter shade). Then we start computing the expected shortest path with the observed parameters. Within the first iteration (under 1 second), it computes a path which guides the robot to the other side of the chair, avoiding the edges that were previously blocked by the person (see Fig. 8(b)). On this path, the intermediate milestone guides the robot to lower its elbow to avoid collision with the sign. This shows the whole-body capabilities of the controllers. After crossing the chair and avoiding the sign, the person unexpectedly crosses the line to the other side (Fig. 8(c)). The robot avoids the collision by moving the milestones, replanning, and adapting its path (Fig. 8(d)). Fig. 9(b) shows



(a) Expected behavior (b) ESPER during execution (c) Chosen path percentage (d) Average motion time

Fig. 7. (a) Illustration of an elastic roadmap in a scene with two dynamic obstacles (only a subset of roadmap edges is shown): d remains constant throughout the up-and-down motion of obstacles O_1 and O_2; the dashed edges have high collision probabilities because the obstacles move near the upper and lower walls; the solid edges have low collision probability; depending on the probabilities the expected shortest path is either above O_1 or between O_1 and O_2. (b) Screenshot of the execution of the ESPER planner. (c) and (d): In both graphs the x-axis represents the obstacle to robot velocity ratio; the y-axis in (c) represents the success rates for different paths (numbers do not add up to 100% for uncertainty-unaware planning due to collisions) and in (d) the average execution time.



(a) The initial robot configuration. The robot has to move its end-effector on a line towards the camera.

(b) ESP computes the expected shortest path which guides the robot on the path underneath the sign.

(c) The person changes the side and blocks the current path to the goal.

(d) the robot replans fast enough to avoid the person without slowing down.

Fig. 8. a) Execution on a mobile manipulation platform. Shown below the screenshots from the execution are on the left: the current state of the ESPER planner. The task is shown as a pink line; The bounding boxes of the observed obstacles are shown green; The blue robot shapes are all task-consistent milestones; Shown in red is the current milestone with lowest expected cost. On the right we show the matching point cloud view from the robots perspective. The segmented obstacles are shown in color.

the executed trajectory of the robot. During motion the end-effector error with respect to the base localization never exceeds 6 *cm* (see Fig. 9(a)).

All requirements R1 to R3 were present in this experiment. The robot motion was subject to task constraints and whole-body motion was required to avoid the sign (R1). The robot reasoned about the uncertain occurrence of the person, which let it move on a path that avoids the person early on, avoiding collision and reducing the expected motion duration (R2). Finally, the motion was generated based on on-board sensor data using a both an RGB-D sensor and a laser range finder (R3).

V. Conclusion

We presented the Expected Shortest Path Elastic Roadmap (ESPER) planner, an approach for the sensor-based generation of task-constrained, whole-body motion under uncertainty. In simulated experiments with stationary and mobile manipulation platforms and in a real-world experiment with a mobile manipulator, we demonstrated the planner's ability to execute collision-free mobile manipulation tasks while reasoning about environmental uncertainty. In our real-world experiment, the robot can only perceive its environment using on-board sensors. It is this restriction to on-board sensors that facilitates the treatment of uncertainty by the ESPER planner. We show that in the region inaccessible to on-board sensors, uncertainty can be handled using a polynomial-time expected shortest path algorithm. The presented experiments demonstrate that—in the context of mobile manipulation—the ESPER planner satisfies the requirements of motion generation, handling of uncertainty, and perception.

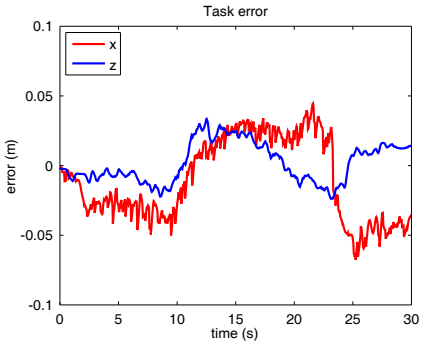
References

[1] A. J. Briggs, C. Detweiler, D. Scharstein, and A. Vandenberg-Rodes, "Expected shortest paths for landmark-based robot navigation," *The International Journal of Robotics Research*, vol. 23, no. 7-8, pp. 717–728, Aug. 2004.

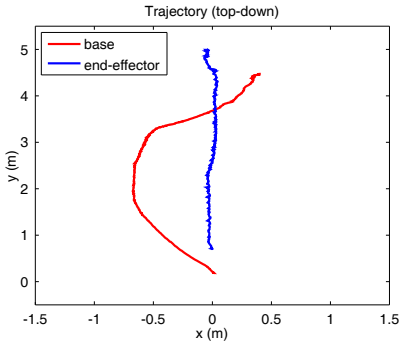
[2] Y. Yang and O. Brock, "Elastic roadmaps—motion generation for autonomous mobile manipulation," *Autonomous Robots*, vol. 28, no. 1, pp. 113–130, Jan. 2010.

[3] J. F. Canny, *Complexity of robot motion planning*. MIT press, 1988.

[4] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of markov



(a) Task error



(b) Base and end-effector trajectory

Fig. 9. Task error and base trajectory for the motion of Fig. 8. The end-effector position is constrained to be constant in x and z directions. The task error never exceeds 6 cm.

decision processes," *Mathematics of operations research*, vol. 12, no. 3, pp. 441–450, 1987.

[5] S. S. Srinivasa, D. Ferguson, C. J. Helfrich, D. Berenson, A. Collet, R. Diankov, G. Gallagher, G. Hollinger, J. Kuffner, and M. V. Weghe, "Herb: a home exploring robotic butler," *Autonomous Robots*, vol. 28, no. 1, pp. 5–20, 2010.

[6] S. Chitta, E. G. Jones, M. Ciocarlie, and K. Hsiao, "Mobile manipulation in unstructured environments: Perception, planning, and execution," *Robotics & Automation Magazine, IEEE*, vol. 19, no. 2, pp. 58–71, 2012.

[7] A. Hornung, M. Phillips, E. Jones, M. Bennewitz, M. Likhachev, and S. Chitta, "Navigation in three-dimensional cluttered environments for mobile manipulation," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, 2012, pp. 423–429.

[8] L. P. Kaelbling and T. Lozano-Pérez, "Integrated task and motion planning in belief space," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1194–1227, 2013.

[9] R. R. Burridge, A. A. Rizzi, and D. E. Koditschek, "Sequential composition of dynamically dexterous robot behaviors," *The International Journal of Robotics Research*, vol. 18, no. 6, pp. 534–555, 1999.

[10] L. Yang and S. LaValle, "The sampling-based neighborhood graph: an approach to computing and executing feedback motion strategies," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 3, pp. 419–432, June 2004.

[11] A.-a. Agha-mohammadi, S. Chakravorty, and N. M. Amato, "Firm: Sampling-based feedback motion planning under motion uncertainty and imperfect measurements," *The International Journal of Robotics Research*, 2013.

[12] D. P. Bertsekas, *Dynamic programming and optimal control*. Athena Scientific Belmont, MA, 1995.

[13] R. Alterovitz, T. Siméon, and K. Goldberg, "The stochastic motion roadmap: A sampling framework for planning with markov motion uncertainty," in *Robotics: Science and Systems*, 2007, pp. 246–253.

[14] I. A. Sucan and L. E. Kavraki, "Accounting for uncertainty in simultaneous task and motion planning using task motion multigraphs," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, 2012, pp. 4822–4828.

[15] P. Missiuro and N. Roy, "Adapting probabilistic roadmaps to handle uncertain maps," in *Robotics and Automation, 2006 IEEE International Conference on*, may 2006, pp. 1261 –1267.

[16] B. Burns and O. Brock, "Sampling-based motion planning with sensing uncertainty," in *Robotics and Automation (ICRA), 2007 IEEE International Conference on*. IEEE, Apr. 2007, pp. 3313–3318.

[17] R. Simmons and S. Koenig, "Probabilistic navigation in partially observable environments," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 1995, pp. 1080–1087.

[18] H. Kurniawati, T. Bandyopadhyay, and N. M. Patrikalakis, "Global motion planning under uncertain motion, sensing, and environment map," *Autonomous Robots*, vol. 33, no. 3, pp. 255–272, 2012.

[19] B. Marthi, "Robust navigation execution by planning in belief space," in *Robotics: Science and Systems*, Sydney, Australia, July 2012.

[20] S. Loibl, D. Meyer-Delius, and P. Pfaff, "Probabilistic time-dependent models for mobile robot path planning in changing environments," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 5545–5550.

[21] A. Bar-Noy and B. Schieber, "The canadian traveller problem," in *Proceedings of the second annual ACM-SIAM symposium on Discrete algorithms*, 1991, pp. 261–270.

[22] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[23] H. Sadeghian, L. Villani, M. Keshmiri, and B. Siciliano, "Dynamic multi-priority control in redundant robotic systems," *Robotica*, vol. 31, no. 07, pp. 1155–1167, 2013.