# Near Time-optimal Constrained Trajectory Planning on Outdoor Terrain

Marin B. Kobilarov and Gaurav S. Sukhatme

Robotic Embedded Systems Laboratory Center for Robotics and Embedded Systems University of Southern California, Los Angeles, USA mkobilar gaurav@robotics.usc.edu

Abstract—We present an outdoor terrain planner that finds near optimal trajectories under dynamic and kinematic constraints. The planner can find solutions in close to real time by relaxing some of the assumptions associated with costly rigid body simulation and complex terrain surface interactions. Our system is based on control-driven Probabilistic Roadmaps and can efficiently find and optimize a near time-minimum trajectory. We present simulated results with artificial environments, as well as a real robot experiment using Segway Robotic Mobile Platform.

## I. INTRODUCTION

The ability to move optimally on non-flat terrain is central to outdoor mobile robotics. An autonomous vehicle operating outdoors should be able to quickly plan a minimum time trajectory to a given location in the environment. This paper presents a practical implementation of an outdoor planner for differentially driven ground vehicles under dynamic constraints. The planner is demonstrated in numerous simulated environments as well as in the real world. A Segway Robotic Mobile Platform (RMP) is used to demonstrate the planner effectiveness. We choose the RMP because it works well outdoors and has unconventional dynamics that are interesting to analyze.

Our goal is to implement a fast and efficient time-optimal outdoor planner. Such a planner requires several components: an accurate geometric 3-D environment map that has high resolution and encodes real-world surface properties; an accurate physical-dynamic simulator accounting both for the robot behavior and its interaction with the surface under different environment conditions; and a trajectory generation component that computes an optimal and feasible trajectory satisfying the robot kinematic and dynamic constraints. These are the main requirements for a successful planner. Ideally, even if these required components were available, the time required for accurate simulation and planning would be far from close to real time and the system would not be suitable for an agile autonomous robot. Although there have been recent advances in the relevant fields (Sec II), in reality, systems that completely meet each of the listed requirements are not available. Generating accurate 3-D maps of outdoor terrains is hard and still an open problem. Fast and realistic rigid body simulation is often available only in a specific environment domain and complex surface interactions are usually unaccounted for. Kinodynamic planning in real-time is also difficult and currently only randomized approaches are shown to find near time-optimal paths in close to real time.

Clearly, implementing an ideal planner with the above requirements is not feasible. In order to create a practical real-world implementation one would have to solve the problem by making simplifying assumptions that relax these requirements. This paper presents such a practical implementation that is able to plan near-optimal trajectories in close to real time. We use an outdoor map that is an interpolated surface from a coarse elevation grid. Because the map is imprecise, we choose to represent the robot kinematically as a point instead of a full rigid body. Thus, time consuming collision detection can be avoided since it is unnecessary in case of a low resolution (and most likely erroneous) map. The dynamics of the robot are compressed in a model that depends only on the normal of the traversed terrain patch and the vehicle velocities and accelerations. Although the robot is modeled as a point, the computed trajectories account for its steering non-holonomic and dynamic constraints. These assumptions enable fast simulation that is critical for efficient kinodynamic planning. The planner is based on control-driven Probabilistic Roadmaps (PRM).



Fig. 1. Segway RMP on uneven terrain

The implemented system is applicable to differentially driven wheeled mobile robots with imposed dynamic bounds. We demonstrate the system by specifically encoding the dynamics and constraints of the two-wheeled differentially driven Segway RMP, shown in Figure 1. The RMP is the robotic version of the popular Segway Human Transporter. It can self-balance and move with commanded linear and angular velocities. The robot pitches forward (leans) when accelerating and backward when decelerating. It can pitch up to 40 degrees in either direction depending on acceleration. Higher platform tilt causes the RMP to lose balance and fall. The RMP can develop a top speed of 3.5 m/s, can sustain a maximum acceleration of 2 m/s<sup>2</sup>, can carry up to 50 kg, and has approximately 15 km travel range [1]. It has zero turning radius, but at high forward speed the turning rate is limited. Based on empirical tests carried out in our lab, it can climb slopes up to 17 degrees without payload. It can move on different types of terrain, and easily traverse small obstacles. Given its good mobility the Segway RMP can be used for a number of outdoor tasks e.g. as a robotic mule to transport precious cargo during time-critical missions, operations in hazardous sites, or as a personal assistant carrying luggage or executing chores that require quick autonomous travel between two locations.

Our goal is to find a minimum time path from a start location to a goal location given an elevation map of the terrain. For the RMP, this means maximizing acceleration whenever possible. High accelerations result in higher pitch and less stability. Motivated by safety considerations (e.g. the cargo may be sensitive to too much tilt) we impose an upper bound on the allowed maximum pitch of the robot. This is a dynamic constraint not encountered with conventional wheeled robots and its effect on time-optimal planning is considered in our analysis<sup>1</sup>. In addition to the constraint on pitch, the robot is also subjected to the 'standard' dynamic constraints: bounded maximum velocity, acceleration, and roll (lateral tilt).

The planner we built can compute near time-optimal trajectories which satisfy these dynamic constraints. It is verified in simulation using numerous environments of different complexity. The trajectories are computed in close to real time. The trajectories are not necessary feasible since the full rigid body kinematics is not considered during planning. But we emphasize that because high quality 3-D maps with accurate terrain material modeling are still extremely difficult to generate, the planned trajectories are appropriate for real-time global planning in the real world. Finally, the planner is applied on a real Segway RMP on a small uneven terrain whose map was generated from laser scans.

## II. RELATED WORKS

## A. Terrain Planning

Path planning on terrain is a small part of the broad motion planning [2] field. While most works consider the kinematics of the robot only a few address planning under dynamic constraints that is time optimal. Some authors use grid-based search to find shortest paths using A\* [3], [4],

[7]. Such paths are not time-optimal and usually do not account for dynamic constraints. Some of these works as well as others use a geometric model of the robot [8], [9], [5]. [10] focuses on terrain parameter estimation in terms of cohesion and friction angle, [11] models the terrain surface as connected particles. Some researchers address planning based on the traversability of the terrain: in terms of roll, pitch, and height of the robot placement [3] or by using fuzzy logic to classify the terrain [12]. [13] presents an efficient algorithm for finding minimum energy paths accounting for overturn danger and power limitations. Such paths are near-optimal in distance and energy but not in terms of time. In [8] and [16] Cherif uses a twolevel planner by generating a set of subgoals on a discrete grid in the 2D projected configuration space. The planner then finds feasible motions between these points on the terrain surface. This technique finds paths that are safely executable but since they are precomputed without initially accounting for dynamics they are not time optimal.

[9] incorporates uncertainty in the planning method by characterizing the soil/tire interaction and the path following error. [4] also proposes a method for dealing with uncertainty in terrain profile by attaching error intervals on each grid elevation and by imposing a free channel around the computed trajectory. [6] adds that planning can be made more robust by attaching uncertainty to the computed trajectories based on the visibility of natural landmarks used for localization. Most works dealing with dynamics use constraints such as terrain slope and roughness, engine torque, sliding, surface contact, tip-over, and velocity and acceleration limits [7], [6], [18]. [18] models the terrain surface and paths using B-spline interpolation. They present a time-optimal planner under dynamic constraints using global search (optimized with "branch and bound") and local trajectory optimization for time. Their kinodynamic planner has many features, but grid based search might not be scalable to larger environments.

## B. Randomized Kinodynamic Planning

In order to reduce the hardness of complete deterministic planning Kavraki and Svestka ([14], [15]) introduced Probabilistic RoadMaps(PRM). This early version had a preprocessing construction phase and an on-line multi-query phase and did not account for vehicle dynamics. Rapidly-exploring Random Trees(RRT) are due to LaValle et al. ([26], [27], [29], [28]) to create a single query planner that outputs dynamically executable paths. Hsu et al. [30] propose a kinodynamic planning method based on a single query Probabilistic Roadmap. Feron et al. [31] introduce the idea of real-time motion planning using PRM and an a priori available obstacle-free guidance system based on a maneuver model.

[19] and [20] mention the use of rapidly exploring random trees (RRTs) for terrain planning. To our knowledge, only [21] have implemented an RRT-based planner for non-flat terrain. They work with local sensor range data only and without a priori known maps and do not focus on globally optimal solutions.

<sup>&</sup>lt;sup>1</sup>Although the planner uses a constraint specific to the Segway RMP, the work presented is applicable to any other type of two-wheeled or car-like robot by removing the pitch constraint.

In contrast to previous works, we apply an entirely randomized approach to handle the complexity of kinodynamic planning on non-flat terrain. With the help of control-driven randomized methods the planner considers dynamics while exploring the environment efficiently. Although not optimal, the paths can be computed in near real-time which is not guaranteed in discrete search methods. In addition, we consider the novel situation where a dynamic constraint is closely related to the stability of the robot.

## III. PROBLEM STATEMENT

A 3-D rigid body dynamics is conventionally represented by the ODE  $\dot{x} = f(x, u)$  where  $x \in \mathcal{X}$  is the robot state,  $\mathcal{X}$  is 12 or larger dimensional state space (position, orientation, their first derivatives, and possibly other state variables), and u is the control input, usually accelerations. Dynamic bounds are represented in the form F(x) < 0. The response of a robot can be simulated based on this formulation with given initial conditions and a sequence of control inputs in an obstacle-free setting. The problem with simulation on an uneven terrain is that there is constant collision (external force) which changes the system dynamics. Although it is possible to use an efficient full rigid-body simulator that computes the robot motion on a non-flat surface it will be very time consuming and impractical for on-line global planning in arbitrary large terrain. Thus we make two important assumptions about the motion of the robot and the terrain. This section states the planning problem, describes the assumptions, and defines the constraints.

## A. Problem formulation

The robot operates on a terrain surface  $\mathbf{G} = \{(x,y,g(x,y))|g(x,y)\text{is height at grid location}(x,y)\}$  which is interpolated from an elevation grid of cell size  $c_g$ . Its configuration space is defined as  $\mathcal{C} \subset (\mathbf{G} \times \mathbf{SO}(3))$  with reachable space  $\mathcal{C}_{free} \subset \mathcal{C}$ . Start and goal configurations  $s_0 \in \mathcal{C}_{free}$  and  $s_f \in \mathcal{C}_{free}$  are given. A set of possible controls  $\mathbf{U} = \{u = (\dot{v}, \dot{\omega}) | |\dot{v}| \in [0, \dot{v}_{max}], |\dot{\omega}| \in [0, \dot{\omega}_{max}]\}$  is provided where  $\dot{v}$  and  $\dot{\omega}$  are linear and angular accelerations. One state transition equation  $\dot{\mathbf{s}} = f(\mathbf{s}, u)$  defines the robot control response. The goal is to compute an admissible trajectory  $\tau : [0,T] \to \mathcal{C}_{free}$  such that  $\tau(0) = s_0, \ \tau(T) = s_f,$  and a set of controls  $U_\tau$  to achieve this trajectory such that the total travel time T is minimized.

# B. Planar Discretization Assumption

The planner assumes that at each instant for a very short distance  $\delta_g$  the robot is moving on a tilted planar patch extracted from the surface (similar to [22]) (Fig 2). Thus, the 3D trajectory is composed of  $\delta_g$ -long transformed planar 2D segments. This assumption is valid for a point robot and determines the trajectory discretization. In case of elevation grids,  $\delta_g$  should be chosen as a fraction of the grid cell size  $c_g$ . Assume that the robot is moving on planar patch  $\Pi$  with normal  $\vec{N}$  along tangent  $\vec{T} \in \Pi$ . We can define a local patch reference frame  $\mathcal{F}_{\Pi} = (O, \vec{T}_o, \vec{N} \times$ 

 $\vec{T_o}, \vec{N}$ ) that corresponds to  $\Pi$ .  $O \in \Pi$  is the point where the robot transitions to  $\Pi$  with tangent  $\vec{T_o}$ . We assume that robot transitions between planes with continuous velocity.

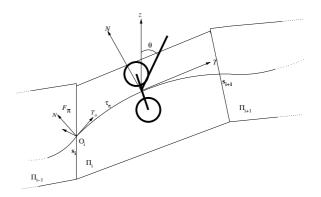


Fig. 2. The robot's discretized path.

#### C. Dynamics Assumption

The dynamic model of the robot can now be simplified so that the only external force affecting the system dynamics arises from the slope along the tangent of motion of the planar patch being traversed. It is assumed that the robot wheels roll without slipping and have constant traction with the ground. Thus, the dynamic model of a planar vehicle can be modified to incorporate the slope of the patch. The control response can then be computed by integrating the new dynamics equations forward in time. We model the dynamics of Segway RMP using this assumption. The RMP has a balancing controller with full state feedback and decoupled PID controllers for steering and velocity. The robot is operated with linear and angular velocity commands. Its closed loop low-level controller guarantees that commanded velocities are achieved quickly as long as such velocities and the required accelerations are within allowable bounds. These bounds are determined by the maximum torque of the robot motors and depend on the slope of the terrain and its surface properties. Some authors (sec. II) derive dynamic equations for a general rigid body robot moving on terrain and use them to determine dynamic constraints such as torque limits, sliding, tipping over, etc... In practice, the effect of outdoor surfaces such sand, grass, water, is difficult to model. We choose to derive the dynamic bounds of the Segway experimentally rather then using formal modeling.

We have built a complete geometric and dynamic model of the Segway in the 3D kinodynamic simulator Gazebo [23], but to achieve real-time planning efficiency we choose to use a point robot and simplify the computation of dynamics by using lookup functions. Assuming a linear time-invariant model we recorded the Segway's velocity and acceleration bounds at different slopes, as well as its pitch response at a range of accelerations and slopes. Such a model does not represent the full system dynamics but is approximately correct (excluding transient behavior) and convenient for fast simulation. Formal mechanical

models of two-wheel balancing robots can be found in [33], [34]. Such models can be modified to include terrain properties and complex surface interactions but, again, such methods would be too complex and the simulation too costly for real-time planning.

## D. Robot Simplified Model and Constraints

Based on the planar discretization assumption, robot displacement can now be viewed as a planar motion starting at reference  $\mathcal{F}_{\Pi}$  fixed at the point where the robot enters tilted plane  $\Pi$ . Let the robot state be defined as  $\mathbf{s} = (\mathcal{F}_{\Pi}, x, y, \psi, v, \omega, \theta)$ , where:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \\ \dot{v} \\ \dot{\omega} \end{pmatrix} = \begin{pmatrix} v \cos \psi \\ v \sin \psi \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \mathbf{u}$$

x, y, and  $\psi$  are the robot location and orientation defined in the reference frame  $\mathcal{F}_{\Pi}$ . The frame  $\mathcal{F}_{\Pi}$  does not change until the robot transitions to a new plane. v and  $\omega$  are the linear and angular velocities, and  $\theta$  is pitch computed directly as described in the preceding subsection. From this state formulation a standard 3-D rigid body (e.g. including roll and its derivative, as well as global reference frame variables) state can be easily computed. The planner generates a trajectory and corresponding linear and angular acceleration controls that track it. Since the RMP is controlled with velocities, these accelerations are integrated at high frequency to compute velocities v and  $\omega$  and send them to the robot closed loop controller. Thus the commanded velocity profile curve is close to continuous. Since the robot moves with continuous velocity its trajectory must have continuous curvature. The curvature derivative must be constrained to reflect a bound on the maximum velocity with which the robot can change heading [25].

$$|\kappa| < \kappa_{max}, \, \hat{\kappa}_{max}(v) = 1/\hat{R}_{min}(v)$$

$$|\dot{\kappa}| < \sigma_{max}$$

where  $\hat{R}_{min}(v)$  is the minimum turning radius as a function of translational velocity and  $\sigma_{max}$  is maximum rate of change in curvature. These bounds create the following steering constraints:

$$\omega < v\kappa_{max}(v)$$
  $\dot{\omega} \le v\sigma_{max} + \dot{v}\kappa$  (1)

Constraints of this type are especially important in timeoptimal planning since the vehicle might operate at high speeds.

In addition to these bounds, we can define the standard non-holonomic constraint,

$$-\dot{x}\sin(\psi) + \dot{y}\cos(\psi) = 0. \tag{2}$$

and the following dynamic constraints (either from the vehicle envelope or further constrained by the user):

Pitch: 
$$|\theta| < \theta_{max}$$
 (3)

Roll:
$$|\phi| < \phi_{max}$$
 (4)

Velocity:
$$v < v_{max}$$
 (5)

Acceleration: 
$$|\dot{v}| < a_{max}$$
 (6)

#### IV. RANDOMIZED KINODYNAMIC SOLUTION

Kinodynamic planning is known to be NP-hard [17]. Complete and deterministic algorithms require at least exponential time in the dimension of the state space. In case of multidimensional state spaces, such as the one considered in this paper, approximation methods are used to compute near optimal solutions. While there has been an extensive body of literature on this problem the most successful approaches are randomized motion-planning algorithms.

We base our implementation on a control-driven Probabilistic RoadMap. PRM is an appropriate choice for terrain planning because it can expand efficiently in the high-dimensional state space while satisfying the problem constraints. The method creates a tree of nodes which explores the environment until the vicinity of the goal is reached. After one feasible path has been found the algorithm can continue optimizing the best solution by expanding and/or pruning the tree further. An appropriately chosen cost metric determines the "distance" between nodes. Thus, the algorithm tries to optimize the total cost from start(root) to goal location. PRM relies on a control system of the type  $\dot{s} = f(s,u)$  that automatically accounts for constraints.

## A. Cost Metric

The cost metric is time. Each node stores the accumulated cost from the tree root node. Thus, each node in the tree has correct measure of time to all other nodes along the path to the root node. Correct measure to other nodes or new random states is not available and computing it is as hard as solving the original problem. It is possible though to compute an upper and lower bound on the time required to move to any other state. Lower bounds are computed by assuming that the environment between the two nodes is obstacle-free and it allows maximum acceleration/deceleration permitted by the vehicle dynamics. The costs are efficiently computed using bang-bang control. Upper bounds are computed only after a feasible trajectory has been found from the current node to the goal, until then this cost is infinity.

The cost of trajectory from  $s_i$  to  $s_j$  is simply:

$$L(s_i, s_j) = \sum_{k=i+1}^{j} l(s_k, u_k)$$

where  $l(s_k,u_k)$  is the cost of moving from state  $s_{k-1}$  to  $s_k$  after executing control  $u_k$ , which is simply the duration of the control. The accumulated cost at each state s is  $L(s_0,s)$ . The lower and upper bounds  $L^-$  and  $L^+$  of the the cost-to-go to state s' are:

 $L^-(s,s')=L^*(s,s')$ , where  $L^*$  is the optimal cost in an assumed obstacle-free environment where maximum allowed velocity can be achieved.

$$L^+(s,s') = \begin{cases} L(s,s') & \text{if a feasible trajectory from} \\ s \text{ to } s' \text{ has been found;} \\ \inf & \text{otherwise.} \end{cases}$$

This type of cost representation is not new in kinodynamic planning. Fraichard [24], for example, use this type of lower bounds as heuristic in a deterministic statespace search. Feron et. al. [31] introduce an "obstacle-free guidance system" which serves the role of computing cost-to-go estimates in the obstacle-free case. The authors use the cost in computing lower bounds. We extend parts of our PRM implementation from their work.

In addition, we employ the cost bounds to discard nodes that cannot lead to a better solution. At any point of time  $L^+(s_0,s_f)$  represents the cost of the currently best found trajectory from start to the goal. If no trajectory has been found this value is infinity. Thus, once a new  $s_r$  is selected and a feasible trajectory from any of the tree nodes has been found,  $s_r$  is added to the tree only if the following holds:

$$L(s_0, s_r) + L^-(s_r, s_f) < L^+(s_0, s_f)$$

In other words we make sure that the lower bound on the total cost of the resulting path from start to goal through  $s_r$  is less than the currently best existing one. If this criterion is not satisfied the trajectory is discarded as if it were infeasible. This pruning step reduces significantly the number of newly added nodes during trajectory optimization.

# B. PRM

Control system based PRM operates in (state x time) space by sampling "milestones" from that space and connecting them to existing PRM nodes using admissible trajectories. If a solution exists the probability that the algorithm finds the solution converges exponentially to one in the number of random PRM nodes [30]. The algorithm performance depends on the uniform sampling of PRM milestones. We base our implementation on the approach of Feron et. al. [31]. When selecting nearest neighbor, this method checks all existing nodes (sequenced by ascending  $L^-(s,s_r)$ ) until a feasible trajectory reaching  $s_r$  is found. We implement the expansion heuristics and pruning techniques described by the authors.

## C. Steering Method

The system assumes that the robot can be steered from any existing tree state to a new random state if a feasible trajectory between the two exists. But how do we select the controls that will result in the minimum cost local trajectory? The controls can be selected at random or, for example, a control theoretic method can be used to guide the motion locally. The trajectory is then computed by applying the controls and integrating the state equations

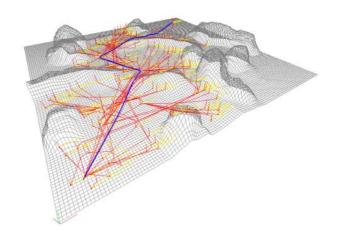


Fig. 3. PRM

forward in time. Given the trajectory planar discretization assumption we choose the following strategy. When trying to get to a new state the robot first changes heading as quickly as possible towards that state and then continues in a straight line. The turn is made by first applying maximum possible angular acceleration followed by minimum such until the angular velocity is zero and robot moves in a straight line. Such steering should satisfy constraints 1 and 2. The choice of smooth trajectory selection is based on Scheuer and Fraichard's method described in [25]. The authors present a technique for finding a smooth local nearoptimal path in 2D between two configurations of position, heading, and zero curvature. We modify their algorithm to include arbitrary starting curvature. The modified version can be found in [32].

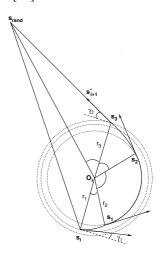


Fig. 4. Left turn during PRM extension

Using this method, for example, a left turn consists of up to four pieces (Figure 4):

- Clothoid with curvature varying linearly from initial curvature  $\kappa_{max}(v_i)$ . The curve starts at  $\mathbf{s}_i$  and ends at  $\mathbf{s}_1$ .
- Optional arc with constant curvature  $\hat{\kappa}_{max}(v_i)$  from  $\mathbf{s}_1$  to  $\mathbf{s}_2$ .

- Anticlothoid with curvature varying linearly from  $\kappa_{max}(v_i)$  to 0. The curve stretches from  $s_2$  to  $s_3$ .
- Straight line from  $s_3$  toward  $s_{rand}$  ending at  $s_{i+1}^*$ .

The method provides a curvature profile that can be used to compute maximum angular acceleration within the constraints that will result in a shortest length turn given the current velocity. Such turn might not be shortest time but it is an appropriate choice locally.

Since the generated trajectory is discretized on a series of connected planes we iteratively apply these 2D methods in the local robot reference frame on each plane. For example, a single tree node of total length  $d_{max}$  can consist of  $\lceil d_{max}/\delta_g \rceil$  connected planar segments each of length  $\delta_g$  or less.

When the robot transition to a new plane, the trajectory is transformed using the transformation associated the change in the two planes reference frames. The transformation method as well as the method for deriving the reference frame of each new plane is described in [32].

### V. EXPERIMENTS

#### A. Simulations

The planner is simulated on grids ranging form 128x128 to 1048x1048 cells. The cell size is 0.25 meters. The maps simulate uneven surfaces that represent environments with hills and valleys, narrow corridors, obstacles, and dead ends. The PRM in each run is initialized randomly. We show that the planner can efficiently compute trajectories in various settings. We focus our experimental analysis on the trajectory optimality and the computational time required. The analysis is based on two main criteria: environment size and environment expansiveness.

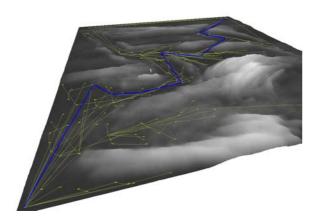


Fig. 5. Large terrain and a trajectory generated in 0.8 seconds.

Fig 6 shows the trajectory costs (total execution times, which the planner minimizes) on the left y-axis time scale and computation time on the right y-axis time scale as a function of the environment size in square meters. The computation time shown is the time it took to reach an initial solution. It took less than a tenth of a second (on a modern PC) to compute an initial trajectory for all ten environments shown. The "Optimized Trajectory Cost" plot shows the improved cost after exactly ten seconds of

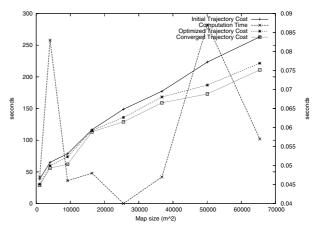


Fig. 6. Trajectory costs and computation times vs. environment size.

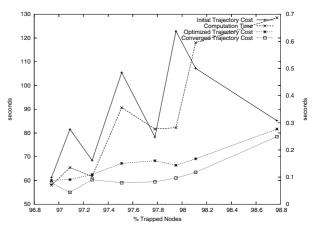


Fig. 7. Trajectory costs and computation times vs. percentage of trapped nodes during expansion.

execution. This additional plot suggests that the algorithm has the ability to find a "sensible" initial solution and quickly optimize it. A converged solution is also plotted for reference.

The notion of expansiveness cannot be exactly determined numerically. We use the fraction of "trapped" nodes of all node extension attempts during PRM expansion as a metric for expansiveness. Naturally, if the environment has more inaccessible parts, contains narrow passages, or many steep elevations, the percentage of trapped nodes is higher. Fig 7 shows the same metrics evaluated above but now as a function of the percentage of trapped nodes. The size of the environments in this experiment is the same but they differ in expansiveness. The environments are deliberately created to model rough terrain with steep hills and numerous obstacles. The graph shows that initial trajectories can be computed in less than 0.7 seconds even in such adverse conditions. In this scenario, it is interesting to observe that the "10-seconds" optimized trajectories have substantially improved from the initial solutions.

While we do not attempt to formally prove the performance of our planner, we use the experimental analysis based on environment size and expansiveness to show that the planner can compute initial solutions in close to real

time and optimize them efficiently.

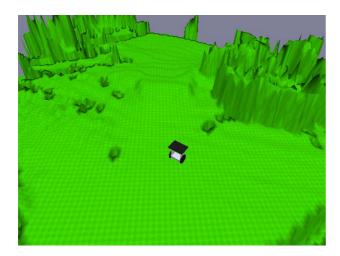


Fig. 8. The terrain verified in Gazebo

### B. Real robot experiment

The planner is demonstrated on the real Segway RMP on a small uneven terrain. The terrain elevation map is built using a data collection system that consisting of laser, inertial measurement unit (IMU) and GPS. The map is roughly 50 by 50 meters and the Segway is asked to plan a trajectory between two locations separated by a few small hillocks. The grid cell size is 0.25 meters and the grid heights are averaged over raw sensor readings. Thus the map is not absolutely accurate and has missing data (appearing as holes). Fig 8 shows the resulting map and robot in Gazebo, where we verified the planner operation before the real world test.

The RMP is equipped with GPS and IMU and is able to determine its initial pose and plot the traversed path. The robot plans a trajectory (Fig 9) and executes the resulting linear (v(t)) and angular  $(\omega(t))$  velocity profile (Fig 10). We do not attempt to localize the robot or to track the computed trajectory exactly. In this experiment we only try to show that the planned path is executable within the robot dynamic constraints. The commanded controls indeed guided the robot along a path close to the planned one and the robot was able to safely reach the vicinity of the goal by avoiding the steepest hill. A dynamic constraint that the planner did not satisfy was the pitch constraint. The slippery surface in the steeper part of the environment caused the robotic platform (Fig 1) to slip and tilt more than the planner had anticipated. At that point, towards the end of the path (Fig 9), the robot diverged from the planned trajectory. This shows the inability of the simple dynamical model to capture complex surface interactions. We leave the solution of this problem to our future work.

## VI. CONCLUSION

The two main problems with planning on terrains under dynamic and kinematic constraints are 1) handling the kinodynamic interaction between the robot and the terrain

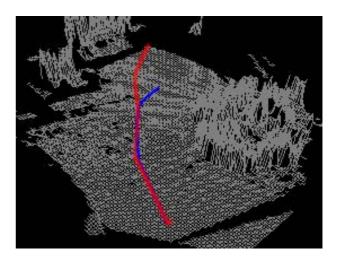


Fig. 9. Computed path(blue) and executed path(red)

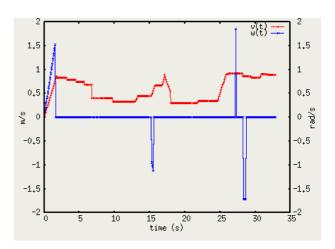


Fig. 10. Velocity profile of the trajectory

surface, and 2) quickly exploring the multidimensional state space in order to find optimal trajectory. The approach in this paper relaxes 1) by making simplifying assumption about the robot dynamics and surface interaction. Under these assumptions, we have presented an efficient solution to 2). The paths computed by our planner are not necessarily feasible but they are sufficiently "good" for real-world applications because generating precise terrain maps is still an open problem.

The simulation tests presented show that the randomized PRM approach is appropriate for terrain planning. Although we have implemented the planner on a real robot and shown that it can execute a given trajectory, we were not able to satisfy all dynamic constraints. Using more realistic dynamical modeling without sacrificing the near real-time performance would be a topic for our future work.

#### REFERENCES

- [1] Segway Robotic Mobile Platform(RMP), Instructions for DARPA Users V1.2. Segway LLC.
- [2] J.C. Latombe, Robot Motion Planning, Kluwer Academic Publishers, Norwell, MA, 1991.

- [3] T. Kubota, Y. Kuroda, Y. Kunii, T. Yoshimitsu, Path planning for newly developed microrover, *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation*, 2001, vol. 4, pp. 3710–3715
- [4] A. Hait and T. Simeon, Motion planning on rough terrain for an articulated vehicle in presence of uncertainties, *Proceedings of the* 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems, 4-8 Nov. 1996, vol. 3, pp. 1126–1133.
- [5] D. Bonnafous, S. Lacroix, and T. Simeon, Motion generation for a rover on rough terrains, *Proceedings. 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001, 29 Oct.-3 Nov. 2001, vol. 2, pp.784–789.
- [6] A. Hait, T. Simeon, and M. Taix, Robust motion planning for rough terrain navigation, *Proceedings. 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 17-21 Oct. 1999, vol. 1, pp. 11–16.
- [7] Y. Guo, L. E. Parker, D. L. Jung and Z. Dong, Performance-based rough terrain navigation for nonholonomic mobile robots, *The 29th Annual Conference of the IEEE Industrial Electronics Society (IECON'03)*, November 2003.
- [8] M. Cherif, J. Ibanez-Guzman, C. Laugier, and T. Goh, Motion planning for an all-terrain autonomous vehicle, *Int. Conf. on Field and Service Robotics*, Pittsburgh, PA, USA, August, 1999.
- [9] K. Iagnemma, F. Genot, S. Dubowsky, Rapid physics-based roughterrain rover planning with sensor and control uncertainty, Proceedings. 1999 IEEE International Conference on Robotics and Automation, 10-15 May 1999, vol. 3, pp. 2286–2291.
- [10] K. Iagnemma, H. Shibly, S. Dubowsky, On-line terrain parameter estimation for planetary rovers, *Proceedings*. 2002 IEEE International Conference on Robotics and Automation, 11-15 May 2002, vol. 3, pp. 3142–3147
- [11] M. Cherif, Motion planning for all-terrain vehicles: a physical modeling approach for coping with dynamic and contact interaction constraints, *IEEE Transactions on Robotics and Automation*, April 1999, vol. 15, issue 2, pp. 202–218.
- [12] A. Howard, H. Seraji, and B. Werger, Fuzzy terrain-based path planning for planetary rovers, *Proceedings of the 2002 IEEE International Conference on Fuzzy Systems*, 12-17 May 2002, vol. 1, pp. 316–320.
- [13] Z. Sun and J. Reif, On energy-minimizing paths on terrains for a mobile robot, *Proceedings. IEEE International Conference on Robotics and Automation*, Sept.14-19, 2003, vol. 3, pp. 3782–3788.
- [14] Kavraki, L. E., Kolountzakis, M. N., and Latombe, J.-C. Analysis of Probabilistic Roadmaps for Path Planning, *IEEE Transactions on Robotics and Automation*, 14(1):166-171, 1998.
- [15] Mark H Overmars, Petr Svestka A Paradigm for Probabilistic Path Planning, *Technical Report*, Department of Computer Science, Utrecht University, March 1996.
- [16] M. Cherif, Kinodynamic motion planning for all-terrain wheeled vehicles, *Proceedings*. 1999 IEEE International Conference on Robotics and Automation, 10-15 May 1999, vol. 1,pp. 317–322.
- [17] Z. Shiller, Y. R. Gwo, Dynamic motion planning of autonomous vehicles, *IEEE Trans. Robot. Automat.*, vol. 7, Apr. 1991.
- [18] Z. Shiller, Motion Planning for Mars Rover, Proceedings of the First Workshop on Robot Motion and Control(RoMoCo), 28-29 June 1999, pp. 257–262.
- [19] C. Urmson, Locally Randomized Kinodynamic Motion Planning for Robots in Extreme Terrain, Thesis Proposal, CMU, 2002.
- [20] R. Jarvis, An Articulated Six Wheel Drive Robot for Very Rough Terrain Navigation, *Proceedings. 2002 Australian Conference on Robotics and Automation*, Auckland, 27-29 November 2002
- [21] D.J. Spero and R.A Jarvis, Path Planning for a Mobile Robot in Rough Terrain Environment, Proceedings of the Third International Workshop on Robot Motion and Control(RoMoCo), 9-11 Nov. 2002, pp. 417–422.
- [22] T. Simeon and B. Dacre-Wright, A practical motion planner for all-terrain mobile robots, *Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 26-30 July 1993, vol. 2, pp. 1357–1363.
- [23] N. Koenig and A. Howard, Design and use paradigms for gazebo, an open-source multi-robot simulator, Proceedings. 2004 IEEE International Conference on Robotics and Automation
- [24] Th. Fraichard Dynamic trajectory planning with dynamic constraints: a 'state-time space' approach, Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Jul.26-30, 1993. Yokohama, Japan. pp.1393-1400.

- [25] A. Scheuer and Th. Fraichard, Continuous-curvature path planning for car-like vehicles, Intelligent Robots and Systems, 1997. IROS '97., Proceedings of the 1997 IEEE/RSJ International Conference on Volume: 2, 7-11 Sept. 1997 Pages:997 - 1003 vol.2
- [26] S. LaValle and J. Kuffner, Jr, Randomized kinodynamic planning, Proceedings. 1999 IEEE International Conference on Robotics and Automation, 10-15 May 1999, vol. 1, pp. 473–479.
- [27] P. Cheng and S. LaValle, Reducing metric sensitivity in randomized trajectory design, *Proceedings. 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 29 Oct.-3 Nov. 2001 vol. 1, pp. 43–48.
- [28] S. LaValle, ¿From Dynamic Programming to RRTs: Algorithmic Design of Feasible Trajectories, In A. Bicchi, H. I. Christensen, and D. Prattichizzo, editors, *Control Problems in Robotics*, 2002 Springer-Verlag, Berlin, pp. 19–37.
- [29] P. Cheng, Z. Shen, and S. M. LaValle, RRT-based trajectory design for autonomous automobiles and spacecraft, Archives of Control Sciences, 11(3-4):167–194, 2001.
- [30] D. Hsu, R. Kindel, J.-C. Latombe, and S. Rock, Randomized kinodynamic motion planning with moving obstacles, *International Workshop on Algorithmic Foundations of Robotics*, pages 233–255, 2000
- [31] E. Feron, E. Frazzoli, M.A. Dahleh, Real-time motion planning for agile autonomous vehicles, AIAA Conf. on Guidance, Navigation and Control, Denver, August 2000.
- [32] M. Kobilarov, G. Sukhatme, Near time-optimal outdoor terrain path planning under dynamic constraints for Segway RMP, *Technical Report*, University of Southern California, March 2004.
- [33] M. Baloh, M. Parent, Modeling and model verification of an intelligent self-balancing two-wheeled vehicle for an autonomous urban transportation system, *The Conference on Computational Intelligence*, Robotics, and Autonomous Systems, Dec. 15 2003, Singapore
- [34] F. Grasser, A. D'Arrigo, S. Colombi, A. Rufer, JOE: A mobile, inverted pendulum, *IEEE Transactions on Industrial Electronics*, vol. 49, No. 1, Feb. 2002