State Space Sampling of Feasible Motions for High-Performance Mobile Robot Navigation in Complex Environments

Thomas M. Howard, Colin J. Green, and Alonzo Kelly
Robotics Institute
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, Pennsylvania 15213
e-mail: thoward@ri.cmu.edu, cjgreen@ri.cmu.edu, alonzo@ri.cmu.edu

Dave Ferguson
Intel Research Pittsburgh
4720 Forbes Avenue
Pittsburgh, Pennsylvania 15213
e-mail: dave.ferguson@intel.com

Received 21 December 2007; accepted 3 April 2008

Sampling in the space of controls or actions is a well-established method for ensuring feasible local motion plans. However, as mobile robots advance in performance and competence in complex environments, this classical motion-planning technique ceases to be effective. When environmental constraints severely limit the space of acceptable motions or when global motion planning expresses strong preferences, a state space sampling strategy is more effective. Although this has been evident for some time, the practical question is how to achieve it while also satisfying the severe constraints of vehicle dynamic feasibility. The paper presents an effective algorithm for state space sampling utilizing a model-based trajectory generation approach. This method enables high-speed navigation in highly constrained and/or partially known environments such as trails, roadways, and dense off-road obstacle fields. © 2008 Wiley Periodicals, Inc.

1. INTRODUCTION

Outdoor mobile robot navigation is a challenging problem because environments are often complex and only partially known, dynamics can be difficult to predict accurately, and both planning time and computational resources are limited. We can generally model the dynamics of a vehicle by a nonlinear differential equation of the form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}). \quad (1)$$

The input or control vector $\mathbf{u}$ and the state vector $\mathbf{x}$ are both time-varying points in input and state space, respectively. The complexity of such accurate models of mobility combined with the scale of outdoor

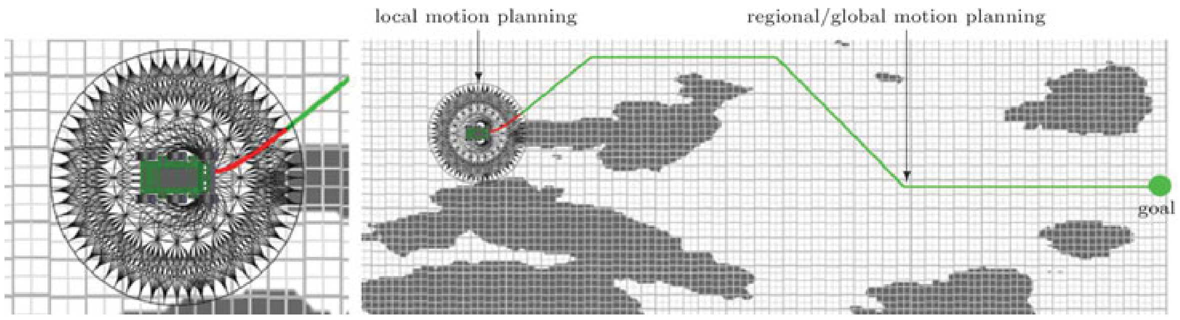


Figure 1. Hierarchical motion planning. The planner uses a search space that accounts for vehicle mobility constraints in the area close to the vehicle, and it uses a grid that ignores vehicle mobility constraints in the area far from the vehicle.

mobile robot navigation leads to a difficult trade-off between the computational demands of perceptive intelligence at the local level and deliberative intelligence at the global level. It is difficult to be both smart and fast when computation is limited.

A common approach to this problem is to employ a hierarchical motion planning architecture to generate behaviors that are both intelligent and responsive. Figure 1 illustrates a case in which the environment is a continuous cost field. Cases based on a more topological environment representation, such as a road network, are shown in Figure 2.

The architecture is hierarchical in the sense that two levels of detail are used for the modeling of both the vehicle and the environment. The higher-level motion planner (global planner) is responsible for directing the vehicle to achieve mission goals. It produces a large-scale, long-term motion plan based on simplified vehicle models and coarse representations of the environment. Conversely, the lower-level motion planner (local planner) is used to keep the vehicle safe. It generates a finer-scale, short-term motion plan based on higher-fidelity vehicle models and finer-resolution representations of the environment.

Whereas it is a common approach to use higher-fidelity models to subsequently smooth the trajectories produced by lower-fidelity planners, the resulting smooth path may be reduced in optimality and it may no longer avoid obstacles. For a continuously moving vehicle, such a planner failure leads at best to the inefficiencies associated with stopping and at worst to a damaging collision. Such considerations lead to a desire for the smoothing algorithm to interpret the environmental model in its computations and hence become a planner in its own right. Once

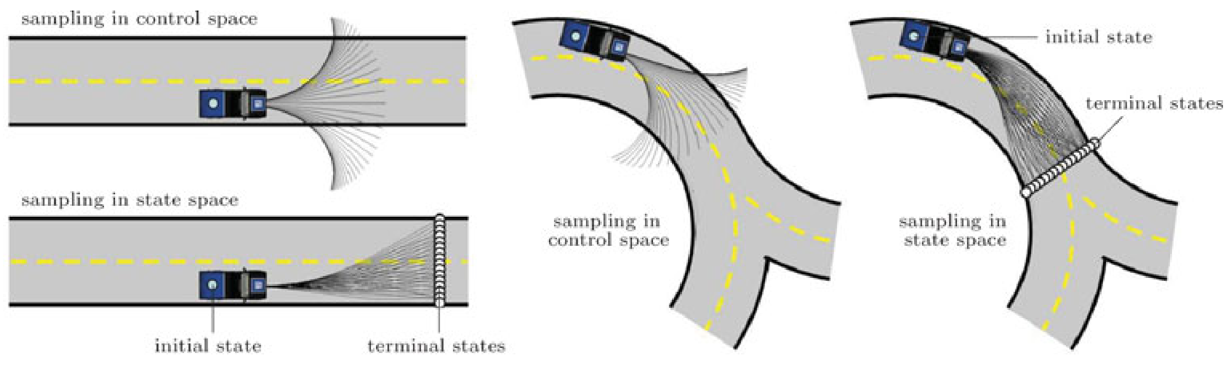


Figure 2. Search spaces generated by sampling in control space vs. state space are shown in environments that are highly constrained (e.g., road networks). The majority of the options generated by sampling in control space leave the lane or are oriented to do so shortly, whereas those generated by sampling in state space remain within the lane.

the smoother becomes a planner, any short-term portion of the path produced by the global planner becomes somewhat redundant because it will always be replaced by the results of this planner, which is more competent on the local scale.

The models used by each of these planners are also typically associated with mutually exclusive spatial regions. Local planner models are used in a small region centered around the vehicle that extends outward as far as useful perception data are available. Global planner models are used outside the boundary of the local region. The local region moves with the vehicle so that decisions are based on models that are, at least locally, as accurate as computation and sensing will allow.

1.1. Motivation

The capacity of a vehicle to move in the environment depends on both the characteristics of the vehicle and the environment. Typical vehicle mobility constraints include, for example, a restriction that wheels roll with no slip or with specified or terrain-dependent slip profiles. Others include the dynamics of the locomotion and steering actuators and of the propulsion system. Motions that satisfy such constraints are said to be "feasible." Environmental constraints may include terrain geometric or soil characteristics that modulate tractive forces, body-contacting obstacles that actively oppose motion, or contextual information such as the requirement that the vehicle remain in its assigned lane of the road most of the time.

In the context of the architecture described previously, the opportunity for a degree of optimization arises regularly when many candidate paths in the local region are obstacle free. For any locally generated path, the characteristics of a path to the goal that starts where the local solution ends can be used to define the best local solution and select it for execution. In this sense, the global planner can serve as guidance to the selection process.

Furthermore, as we will show, different sets of local paths comprising the local search space may be fundamentally better than others in terms of the quality of the solutions they produce. This paper deals with the problem of improving the quality of local motion-planning search spaces that satisfy feasibility and environmental constraints while attempting to maximally exploit global guidance information.

This paper also addresses a difficult issue that arises in the context of hierarchical planning architectures and in differentially constrained motion planning in general. The formation of the local planning problem involves constraints and utilities that are most conveniently expressed in two spaces:

- State space: Those arising from the environment
- Control space: Those arising from vehicle mobility

Environmental constraints are easily satisfied when planned trajectories are expressed in state space, but their dynamic feasibility cannot be easily guaranteed. Conversely, when trajectories are expressed in control (input) space, feasibility constraints can be trivially satisfied but the satisfaction of environmental constraints cannot be easily guaranteed. When most feasible motions are likely to satisfy environmental constraints, control space sampling is an effective approach. However, this traditional approach suffers if the environment imposes significant limits on acceptable motions (Figure 2).

A second consideration is path-sampling efficiency. All approaches to motion planning fundamentally select one trajectory from a set of alternatives. The problem is the design of a search space that will best utilize the computational resources in the time available. Separation of trajectories matters because nearly identical trajectories will likely intersect the same obstacles and are therefore inefficient. We define a well-separated set of trajectories to be one that covers a majority of the reachable state space with a minimum of overlap.

For example, consider a control space sampling consisting of a set of uniformly sampled constant curvature arcs (Figure 3) applied to different initial vehicle states. The trajectories are denser in the direction of the initial angular velocity because the vehicle's limited maximum turning rate ($d\omega/dt$) leads to very similar outputs for several distinct inputs. Conversely, sampling in state space would permit direct control over the spacing of the endpoints of these trajectories.

1.2. Related Work

There has been substantial research in the generation of expressive and complete trajectory sets for local motion-planning search spaces. Some early work in this appears in Kelly and Stentz (1998), where the local motion-planning search space is generated

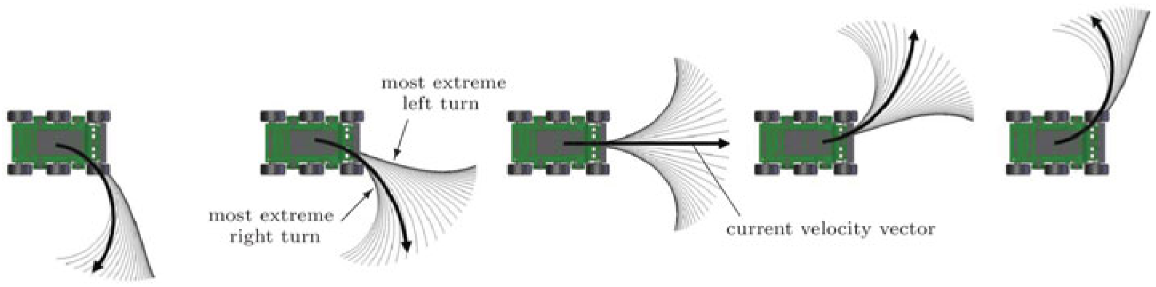


Figure 3. Irregular mapping from control space to state space. Accurate dynamic simulations of a set of uniformly sampled constant curvature arcs (control space sampling) are shown for a variety of different initial vehicle states. Notice that the responses are not uniformly separated despite the uniform separation of controls.

by sampling in the control space of curvature. Each control is passed through a vehicle dynamics model to estimate the response of the vehicle to the control. The shape of the response is highly dependent on the vehicle model and the initial vehicle state (curvatures and velocities). Similar approaches have been adapted in a variety of other unmanned ground vehicles (Kelly et al., 2006; Wettergreen, Tompkins, Urmson, Wagner, & Whittaker, 2005). Egographs (Lacze, Moscovitz, DeClaris, & Murphy, 1998) represent another method for generating local motion-planning search spaces. This approach generates, offline, a well-separated dynamically feasible search space for a limited set of initial states. Online adaptation to changes in terrain or vehicle models are not considered. Precomputed arcs and point turns comprised the control primitive sets that were used to autonomously drive Spirit and Opportunity during the Mars Exploration Rover mission (Besiadecki, Leger, & Maimone, 2007). Trajectory selection was based on a convolution on a cost or "goodness" map. This approach was an extension of Morphin, an arc-planner variant in which terrain shape was considered in the trajectory selection process (Simmons et al., 1995; Singh et al., 2000). Another closely related algorithm is the one presented in Bonnafous, Lacroix, and Siméon (2001), where an arc-based search space is evaluated based on considering risk and interest.

Rapidly exploring random trees (RRTs) have recently been applied to the problem of generating dynamically feasible controls through complex environments. This method is well suited to the problem of navigating complex environments because of its ability to search high-dimensional input spaces and can consider vehicle dynamics and terrain shape in its solution (Berg, Ferguson, & Kuffner, 2006; Melchior, Kwak, & Simmons, 2007). Frazzoli, Dahleh, and Feron (2001) use RRTs to navigate among static and dynamic obstacles, where tree expansions are done in state space rather than input space. A controller is applied to determine the feasible action, if one exists, that connects the current state to the new branch state. Chen and Fraichard (2007) applied partial motion planning, an iterative technique based on RRT that considers vehicle model constraints such as acceleration, steering velocity, and steering angle bounds and the real-time operation constraint of the system for navigation in urban environments.

Potential fields (Haddad, Khatib, Lacroix, & Chatila, 1998; Koren & Borenstein, 1991) have regularly been applied to obstacle avoidance and navigation, where attractive forces represent goals and repulsive forces represent obstacles. In Shimoda, Kuroda, and Iagnemma (2005), potential fields generate actions that considered dynamic hazards such as rollover and terrain shape in their evaluation. Lacroix et al. (2002) applied potential fields for navigation in simple terrain. For more difficult environments, they applied arcs and arc-trees that consider terrain shape in their forward simulation to generate the navigation search space.

Another important group of work involves navigators that solve for an optimal or near-optimal trajectory in the continuum. These methods typically are fast and solve for the local, but not global, optimal solution. A navigator based on optimizing a utility function in the continuum is described in Cremean et al. (2006). Horizon-limited trajectories are generated by minimizing the steering and acceleration control effort subject to a vehicle dynamics model that considers limited steering rates and rollover constraints. Obstacles are avoided by representing them as very low-speed regions and are naturally avoided in this framework. The local minima problem is handled by seeding the optimization function with a coarse spatial path. The *dynamic window* approach (Fox, Burgand, & Thrun, 1997) selects translational and rotational velocities by maximizing an objective function based on the heading to the target position, distance to the closest obstacle, and velocity of the robot. The search space is generated using arcs and is restricted to reachable and safe velocities at its current state.

The satisfaction of environmental constraints in local motion-planning search spaces is related to lane following, a problem that has generated a great deal of interest with the recent DARPA Grand Challenge competitions. In Miller et al. (2006), Bézier splines are constructed with terminal points defined by a lateral offset and the road shape. Dynamically infeasible trajectories are culled through a postprocessing technique, and an optimal navigation strategy is determined through minimizing a weighted combination of integrated path cost and steering effort. A related approach involves Bézier splines whose control points are located and adjusted by the location of obstacles (Berglund, Jonsson, & Soderkvist, 2003; Trepagnier, Nagel, Kinney, Koutsougeras, & Dooner, 2006). In Urmson et al. (2006), a geometric planner that conforms environmental constraints (a trail or road network) is applied to the problem of navigating in desert environments. Dynamic feasibility is passed to a path tracker, and a speed planner ensures safety by slowing the vehicle during aggressive maneuvers. Another similar system is described in Leedy et al (2006), where the deliberative planner generates a path by searching a cost map using A* and following the trajectory using pure pursuit (Coulter, 1992) and then checking it for safety. This method trades off inherent dynamic feasibility for global guidance. A reactive approach is also presented in Leedy et al. (2006) that uses a fuzzy logic controller to follow roads and avoid local obstacles. Nudges and swerves (Thrun et al., 2006), representing smooth and aggressive lateral offsets around a base trajectory, have proven effective for generating expressive local motion-planning search spaces for obstacle avoidance and lane following. Search in trajectory space (κ,v) bounded by hazards and dynamic effects including steering limits, wheel–terrain interaction, rollover, and sideslip constraints is presented in Spenko, Kuroda, Dubowsky, and Iagnemma (2006) and demonstrated to be an effective approach at speeds up to 9 m/s.

This research is related to another body of work concerning the need for expressive motion sets. Motion planning has been concerned with the expressiveness of sequences of motion primitives since Dubins (1957), and work continues in this area today (Frazzoli, Dahleh, & Feron 2003). The importance of separation in a local planning search space has been discussed in Green and Kelly (2007), where it was shown that the mutual separation of a set of paths is related to the relative completeness of the motion set.

This paper differentiates itself from the prior art through applying a state-based sampling strategy to generate path sets that consider global guidance and satisfy environmental constraints while also guaranteeing dynamic feasibility through the use of a model-predictive trajectory generator. Controlling the state-based sampling increases the expressiveness and completeness of the search space. The model-predictive trajectory generator guarantees dynamic feasibility of the actions (to the fidelity of the motion model that describes the dynamics) and requires no postprocessing of the trajectory set. The explicit requirement that trajectories terminate at a regular and predefined horizon provides a better basis for a trajectory arbiter function to select optimal control inputs.

1.3. Technical Approach

An effective local planning search space would ideally be optimal, efficient, and robust. The search space would be optimal if it could maximally exploit global guidance, efficient if it could control path separation, and robust if it searched only feasible motions. Recent advances in real-time, model-based trajectory generation have provided the capability to make progress toward achieving these goals. In Howard and Kelly (2007) a general method is presented that computes control inputs that satisfy a pair of boundary states subject to the vehicle dynamics model, and in Howard, Knepper, and Kelly (2006) we applied it to the problem of path following in the absence of obstacles.

This paper improves on Howard et al. (2006) by generating a set of feasible actions derived by sampling in the surrounding state space. Motions are generated to alternative terminal states based on global guidance—in the form of a global planner or knowledge of road lane configuration. The technique is superior to control space sampling in its efficiency (mean separation of trajectories) and its satisfaction of environmental constraints by construction.

The approach in this paper differs from that of all of the prior work cited in its capacity to generate expressive local motion-planning search spaces using a state space sampling approach while enforcing arbitrary dynamic constraints. Section 2 describes the model-predictive trajectory generation technique used to generate dynamically feasible paths between arbitrary boundary state constraints. Section 3 describes ideas related to the expressiveness and completeness of search spaces. In Section 3, the arguments for why expressiveness and completeness are important in search space generation are reviewed. Section 4 describes the methodology for generating adaptive search spaces that sample in state space subject to vehicle dynamic and environmental constraints, and Section 5 discusses some of the practical considerations for how such a system would be deployed on a real field robot. Simulation and field results for two mobile robot platforms are presented and discussed in Section 6.

2. MODEL-PREDICTIVE TRAJECTORY GENERATION

Our state-based sampling approach requires a method for generating the action between specified pairs of vehicle states. This was achieved by applying the real-time model-predictive trajectory generator from Howard and Kelly (2007) that numerically linearizes and inverts simulated models of vehicle motion. The continuum optimization method efficiently modifies parameterized control inputs to minimize boundary state constraint error.

Explicitly requiring that the actions in our search space satisfy kinodynamic constraints of the vehicle is important for two reasons. First, it allows the motion planner to reason about the feasibility and true cost of actions in the search space. Sampling in the space of controls ensures feasibility of the predicted motion. In a state-based sampling search space generation approach, an algorithm (such as the model-predictive trajectory generator) is needed to invert the vehicle dynamics to provide the input that connects the states. The navigation function (the method by which the local motion planner selects the best trajectory from the set of candidate actions) must have the best information possible to operate effectively. Paths that violate the kinodynamic constraints may not represent the actual vehicle path and yet can influence the navigation function, resulting in suboptimal behaviors. The second reason for embedding a sophisticated model of dynamics in the trajectory generator is that the following error of such paths can be minimized. Predicting some disturbances well for feedforward purposes reduces the difficulty of path following.

The model-predictive trajectory generator can be used to solve for parameterized controls specified by arbitrary boundary state constraints (positions, headings, curvatures, rates of curvature, velocities, etc.). In this paper we will consider boundary state constraints only on the position x, y and heading ψ at the terminal state $\mathbf{x_F}$. Curvature continuity is not considered in this example because the vehicle will likely never execute the entire selected motion between replanning cycles and because the global motion planner places no constraints on curvature. In contrast to the terminal state, the full initial vehicle state $\mathbf{x_I}$ is necessary to initialize the vehicle dynamics model used by the motion simulation.

Other considerations for the algorithm include the controls parameterization and the vehicle dynamics model. The number of freedoms in the parameterized controls is typically a function of the number of state constraints, how well the action represents all feasible motions over that distance, and whether it is desired to optimize something over the path. For skid-steered or Ackerman-steered mobile robots required to satisfy position and heading state constraints, second-order curvature splines (a spline function with three knot points) and simple linear velocity functions (constant, linear, trapezoidal, etc.) have proven to be effective. The second consideration, the motion model, is itself a trade-off between speed (dynamics are computed tens of thousands of times per second) and fidelity (how well it can accurately predict the response to the inputs). We have found first-order models of linear and angular velocity response, along with a rigid body motion simulation, to be an effective trade-off that encodes the major constraints of motion feasibility.

For example, consider the trajectory planning problem exhibited in Figure 4. A simulated vehicle is attempting to reach a target terminal state on top of a hill by executing an action with no path error servo to compensate reactively for path-following error. Initially, a trajectory that does not consider the terrain shape and model dynamics is generated. Using a

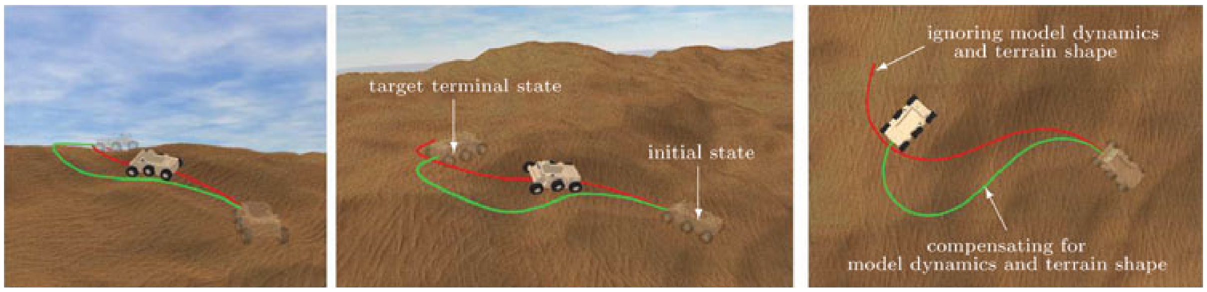

target terminal state
initial state
ignoring model dynamics and terrain shape
compensating for model dynamics and terrain shape

Figure 4. Model-predictive trajectory generation. Two trajectories are generated and executed with and without predictive models of motion that consider terrain shape and simple model dynamics. By incorporating a predictive motion model that understands kinodynamic constraints and wheel–terrain interaction into the trajectory generator, an action that compensates for these effects can be found.

more sophisticated motion model in the simulation, we see that the action terminates at a point that is not the target terminal state. By folding a more accurate model of motion into the vehicle model used by the model-predictive trajectory generator, an action can be generated that reaches the target terminal state.

Typically, feedback control is applied to compensate for discrepancies in the motion model, but in general it cannot be assumed that the generated trajectory is feasible or easily followable in the first place. By applying a method that integrates a more sophisticated vehicle model at the trajectory-planning stage, we can reduce the amount of error that a feedback controller must compensate while having a more accurate prediction of cost for candidate action. If the forward predictive model is in fact good enough, then we can execute controls directly from the model-predictive trajectory generator.

3. SEARCH SPACE SEPARATION

In Green and Kelly (2007) it was shown that one important characteristic of a search space is how well separated its encoded paths are. A portion of that discussion follows. Given a limited amount of computation time, the number of paths that form a search space is necessarily limited and should be selected to maximize the probability of finding a solution when one exists, which we define as *relative completeness*. Additionally, the incremental benefit of finding multiple solutions is small compared to the penalty for failing to find any solutions. So we would like to maximize the probability of finding at least one solution.

Consider next two nonempty differential regions dR_1 and dR_2. If the two regions do not overlap [Figure 5(a)] and obstacles are considered to have zero width, then the probability that dR_1 intersects an obstacle is independent of the probability that dR_2 intersects an obstacle. But if the two regions do overlap [Figure 5(b)] then the probability that both regions intersect an obstacle increases with the area of the overlap. Additionally, if obstacles have some size associated with them, then increasing the separation of two regions that do not overlap decreases the probability that one individual obstacle could intersect both regions at the same time.

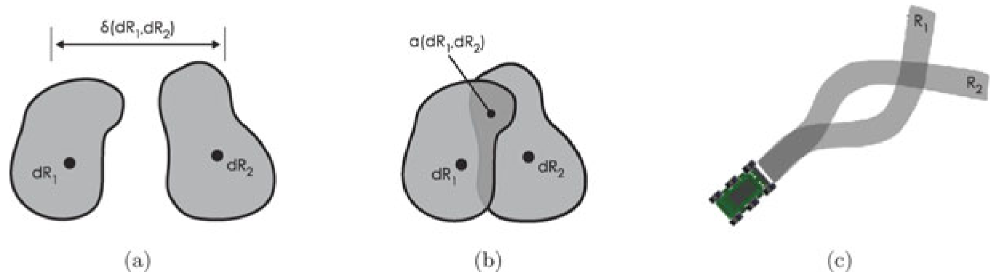


Figure 5. Regions that may contain obstacles: (a) two separated differential regions, (b) two overlapping differential regions, and (c) two finite regions (swept volumes) that overlap.

For two finite regions R_1 and R_2 that could represent the swept volume of a vehicle following a path [Figure 5(c)], the correlation depends on the influence of every element of R_2 on every element of R_1. For such sets we approximate the effect of the separation of each element by relating the probability that two paths will intersect the same obstacle to the area between the two paths.

Based on the above argument, it would be a poor choice if two paths in a search space were unnecessarily close to each other because if one were in collision, the other would also be highly likely to be in collision. Furthermore, if one were not in collision, the other would likely be a redundant solution whose presence is probabilistically of little value from the perspective of completeness—because only one safe path is required. So a search space in which the encoded paths are close to each other is probabilistically less likely to solve a planning query, and has a lower relative completeness, compared to a search space with better separated paths.

4. ADAPTIVE SEARCH SPACES

The algorithm in Howard and Kelly (2007) creates an opportunity to produce controlled distributions of sampled terminal states, in a local search space, that satisfy environmental and separation constraints while also producing feasible trajectories that adhere to a nontrivial dynamics model. Section 4.1 outlines our general approach to structuring the search space adaptation algorithm, and Sections 4.2 and 4.3 discuss and provide examples for exploiting global guidance information and satisfying environmental constraints, respectively.

4.1. Adaptive Search Space Set Design

A search space for any motion-planning problem can be considered to be simply a set of candidate paths through state (or less generally, configuration) space. The paths may be specified implicitly or explicitly in any sufficiently expressive system of coordinates. One form of state space sampling technique generates a set of actions (encoding paths indirectly) by solving for trajectories between n boundary state pairs $\mathbf{x_N}$. The first state in each pair is the initial or current state of the vehicle $\mathbf{x_I}$, and the second state is the target terminal state $\mathbf{x_F}$, which is to be reached at the end of the trajectory:

$$\mathbf{x_N} = \begin{bmatrix} \mathbf{x_I} \\ \mathbf{x_F} \end{bmatrix} = \begin{bmatrix} \mathbf{x_{I,0}} & \mathbf{x_{I,1}} & \cdots & \mathbf{x_{I,n}} \\ \mathbf{x_{F,0}} & \mathbf{x_{F,1}} & \cdots & \mathbf{x_{F,n}} \end{bmatrix}, \qquad (2)$$

where

$$\mathbf{x} = \begin{bmatrix} x & y & \psi & \kappa & v & \ldots \end{bmatrix}^T .$$

The problem becomes that of determining the proper input or control $\mathbf{u}(\mathbf{p}, \mathbf{x})$ that satisfies both the boundary state constraint pair $\mathbf{x_I}$, $\mathbf{x_F}$ and the system dynamics $\mathbf{f}(\mathbf{x}, \mathbf{u})$:

$$\mathbf{u_N}(\mathbf{p}, \mathbf{x}) = \begin{bmatrix} \mathbf{u}_0(\mathbf{p_0}, \mathbf{x}) \\ \mathbf{u}_1(\mathbf{p_1}, \mathbf{x}) \\ \vdots \\ \mathbf{u}_n(\mathbf{p_n}, \mathbf{x}) \end{bmatrix}. \qquad (3)$$

Because the navigation function typically relies on the convolution of the trajectory with a "goodness" or cost map and not simply the input itself, a cost c_i is computed for each valid control that connects a boundary state pair in the set:

$$\mathbf{c_N} = \begin{bmatrix} c_0 & c_1 & \ldots & c_n \end{bmatrix}^T . \qquad (4)$$

Note that this definition is independent of the method of generating the search space because all trajectories start and end somewhere. The basic outline of the adaptive search space generation method is shown in Algorithm 1, which takes the initial state of the vehicle, a model of the system dynamics, and a data structure containing information about the desired shape of the search space. The first step in the algorithm is to generate the boundary state pairs, which in this version is accomplished through Algorithm 2. Once the set of boundary state pairs is established, the method calls the GENERATETRAJECTORY() method to determine the control input and trajectory that satisfies the pair of state boundary constraints. In our implementation we utilize the model-predictive trajectory generation algorithm described in Section 2 because of its ability to generate feasible actions between boundary state pairs in real time subject to arbitrary models of dynamics, although in general other techniques could be substituted. Our approach is based on designing rules and parameters, such as the ones presented in Algorithms 2–4, that define the shape of the boundary state constraint set based on expressiveness and completeness of trajectory sets, global guidance, environmental constraints, and initial state information.

An array of search space shape parameters $\mathbf{p_{ss}}$ defines the shape of the search space. These variables are typically tuned based on the reachable search space of the vehicle, the desired density of the search

Algorithm 1. GENERATESEARCHSPACE ($\mathbf{x_I}$,$\mathbf{f}(\mathbf{x}, \mathbf{u}(\mathbf{p}, \mathbf{x}))$)

```
Input: $\mathbf{p_{ss}}$ (shape parameter vector), $\mathbf{x_I}$ (initial state),
    $\mathbf{f}(\mathbf{x}, \mathbf{u}(\mathbf{p}, \mathbf{x}))$ (vehicle motion model)
Output: control set ($\mathbf{u_N}(\mathbf{p_N}, \mathbf{x})$), cost set ($\mathbf{c_N}$)
1 $\mathbf{x_N} \leftarrow$ GENERATEUNIFORMBOUNDARYSTATES($\mathbf{p_{ss}}$,$\mathbf{x_I}$)
2 for $i \in n$ do
3     $\mathbf{u_i}(\mathbf{p_i}, \mathbf{x}) \leftarrow$ GENERATETRAJECTORY($\mathbf{x}_{\mathbf{I},i}$,$\mathbf{x}_{\mathbf{F},i}$,$\mathbf{f}(\mathbf{x}, \mathbf{u_i}(\mathbf{p}_i, \mathbf{x}))$)
4     if $\mathbf{u_i}(\mathbf{p_i}, \mathbf{x})$ exists then
5         $c_i \leftarrow$ COMPUTETRAJECTORYCOST($\mathbf{x_I}$,$\mathbf{u_i}(\mathbf{p_i}, \mathbf{x})$)
6     end
7 end
8 return $\mathbf{u_N}(\mathbf{p_N}, \mathbf{x})$,$\mathbf{c_N}$
```

Algorithm 2. GENERATEUNIFORMBOUNDARYSTATES ($\mathbf{p_{ss}}$,$\mathbf{x_I}$)

```
Input: $\mathbf{p_{ss}}$ (shape parameters), $\mathbf{x_I}$ (initial vehicle state)
Output: boundary state pair set ($\mathbf{x_N}$)
1 for $i = 0$ to $(n_p - 1)$ do
2     for $j = 0$ to $(n_h - 1)$ do
3         $n \leftarrow i \cdot n_h + j$
4         $\mathbf{x_{I,n}} \leftarrow \mathbf{x_I}$
5         $\alpha \leftarrow \alpha_{\min} + (\alpha_{\max} - \alpha_{\min}) \cdot i/(n_p - 1)$
6         $x_{F,n} \leftarrow x_I + d \cdot \cos(\alpha + \psi_I)$
7         $y_{F,n} \leftarrow y_I + d \cdot \sin(\alpha + \psi_I)$
8         $\psi_{F,n} \leftarrow \psi_I + \psi_{\min} + (\psi_{\max} - \psi_{\min}) \cdot j/(n_h - 1) + \alpha$
9     end
10 end
11 return $\mathbf{x_N}$
```

space, and the number of boundary state constraints we wish to satisfy. In the example shown in Figure 6, the parameter array contains seven constants: the number of samples in terminal state position and heading n_p,n_h, the terminal position horizon d, the angular range of the terminal position sampling $\alpha_{\min}$,$\alpha_{\max}$, and the angular range of the terminal heading angle offsets ($\psi_{\min}$,$\psi_{\max}$):

$$\mathbf{p_{ss}} = \begin{bmatrix} n_p \\ n_h \\ d \\ \alpha_{\min} \\ \alpha_{\max} \\ \psi_{\min} \\ \psi_{\max} \end{bmatrix}. \tag{5}$$

In the scenarios considered in Figure 6, search spaces are generated with the following shape parameter vector values, with varying initial curvature values:

$$\mathbf{p_{ss}} = \begin{bmatrix} n_p = 30 \\ n_h = 3 \\ d = 5.0\ \text{m} \\ \alpha_{\min}, \alpha_{\max} = \pm 45\ \text{deg} \\ \psi_{\min}, \psi_{\max} = \pm 45\ \text{deg} \end{bmatrix}. \tag{6}$$

The result of this approach is a set of sophisticated maneuvers that adapt automatically to varied initial conditions. Furthermore, the trajectories span most of the feasible set while remaining roughly equidistant from each other at their terminal states. This result contrasts dramatically with the results of the control space space sampling techniques from Figure 3.

4.2. Utilization of Global Guidance

The global planner may produce a preferred path to follow or a cost field or navigation function $\varphi(x, y)$ encoding the optimal distance to the goal from every state. Both cases are referred to here more generically as global guidance. On the assumption that deviation

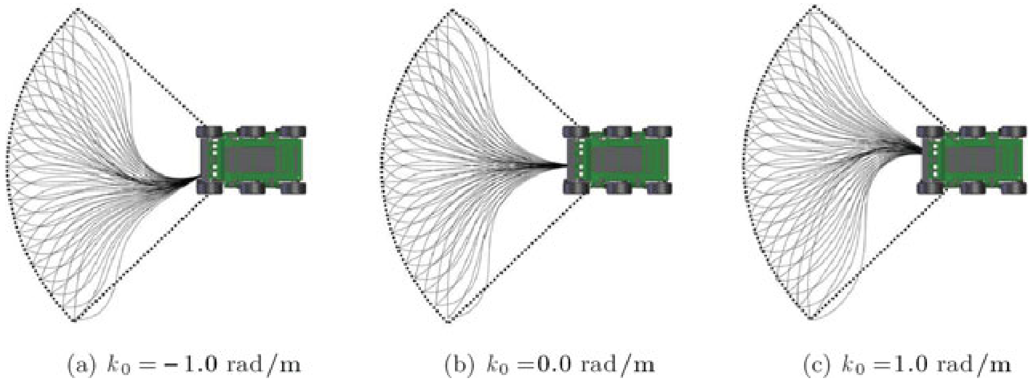


Figure 6. Uniform terminal state sampling for adaptive search space generation with varied initial states. The ability to control the shape of the search space for three initial states: (a) turning left, (b) straight, and (c) turning right.

from global guidance is less likely to lead efficiently to the goal, a local motion-planning search space will improve if it biases its search to be most consistent with global guidance. We typically use a global planner that continuously provides a navigation function (consisting of (an infeasible) path cost from any point to the goal) to the local planner. Given such information, it is better on average to sample terminal states at a higher density in lower-global-cost regions and at a lower density in higher-cost regions, as shown in Figure 7. Some samples are retained in higher-cost regions because the low-cost regions produced by the global planner may not reflect actual dynamic constraints of the vehicle, or an accurate map of the surroundings, and the global planner may not be able to react quickly enough to recently perceived obstacles.

The shape parameters for the navigation function–influenced search space include all of the parameters from the uniformly sampled search space with the addition of the number of navigation function samples n_s:

$$\mathbf{p_{ss}} = \begin{bmatrix} n_s \\ n_p \\ n_h \\ d \\ \alpha_{\min} \\ \alpha_{\max} \\ \psi_{\min} \\ \psi_{\max} \end{bmatrix}. \tag{7}$$

Algorithm 3 is applied to bias the search space with respect to global guidance in the example shown in Figure 7. It is similar to Algorithm 2 except that the α parameter is not sampled uniformly; it is biased to sample more densely in regions where the navigation function $\varphi(x, y)$ is near a minimum. This is accomplished by sampling the navigation function at a fixed horizon uniformly and creating a distribution by taking the difference between the maximum and the sampled values and dividing by the sum. Then the integral of this conditional distribution is sampled uniformly, generating a nonuniform sampling of the angular value of the terminal position α. This generates denser sampling in the low-cost regions of the navigation function, corresponding to a denser search space in those regions. The number of samples taken of the navigation function n_s can be dynamically modified based on the complexity of the environment and how long each sample takes to compute. We use linear interpolation to approximate the navigation function values between sampled states. All of the shared shape parameters used in Figure 7 were the same as in Figure 6 with $n_s = 100$.

4.3. Satisfaction of Environmental Constraints

It is valuable for mobile robots operating in structured environments such as road networks and forest trails to consider environmental constraints in the design of their local motion-planning search space. Figure 2 exhibited the effects of ignoring such constraints, leading to utterly ineffective planning because many actions lie outside of the acceptable navigation region. In situations in which

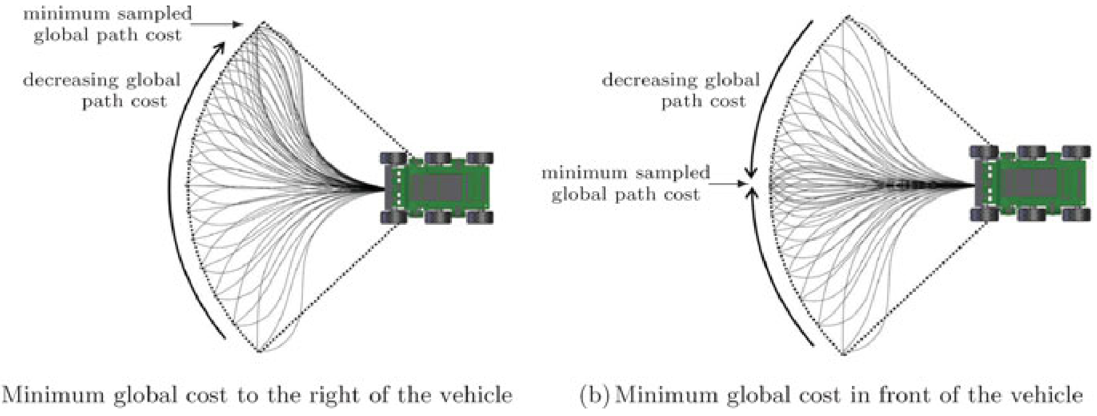

minimum sampled global path cost
decreasing global path cost
(a) Minimum global cost to the right of the vehicle
decreasing global path cost
minimum sampled global path cost
(b) Minimum global cost in front of the vehicle

Figure 7. Focused terminal state sampling for adaptive search space generation. The ability to exploit global guidance via state space sampling generates local motion-planning search spaces that are denser in the direction of minimum global cost (and therefore more likely to reduce the cost to the goal). Examples (a) and (b) show the same setup as in Figure 6 but focused in the direction of minimum global cost.

environmental constraints are important, it is desirable to adjust the sampling strategies to be (at least partially) based on these constraints. For example, consider the search spaces shown in Figure 8. The state-based sampling strategy here is parameterized on the road shape at some forward distance along the path and sampled along the perpendicular of the centerline of the road. The alternative (noncenterline) actions exhibited here are generated for obstacle avoidance and lane-switching capabilities.

Algorithm 3. GENERATEGLOBALLYGUIDEDBOUNDARYSTATES ($\mathbf{p_{ss}}$,$\mathbf{x_I}$,$\varphi(x, y)$)

```
Input: $\mathbf{p_{ss}}$ (shape parameters), $\mathbf{x_I}$ (initial vehicle state),
       $\varphi(x, y)$ (global navigation function)
Output: boundary state pair set ($\mathbf{x_N}$)
1  for $i = 0$ to $(n_s - 1)$ do
2      $\alpha_{\mathbf{s}i} \leftarrow \alpha_{\min} + (\alpha_{\max} - \alpha_{\min}) \cdot i/(n_s - 1)$
3      $\mathbf{cnav}_i \leftarrow \varphi(x_I + d \cdot \cos(\psi_I + \alpha_{\mathbf{s}i}), y_I + d \cdot \sin(\psi_I + \alpha_{\mathbf{s}i}))$
4  end
5  $cnav_{\text{sum}} \leftarrow \sum_{i=0}^{n-1} \mathbf{cnav}_i$
6  $cnav_{\max} \leftarrow \max(\mathbf{cnav})$
7  for $i = 1$ to $(n_s - 1)$ do
8      $\mathbf{cnav}_i \leftarrow (cnav_{\max} - \mathbf{cnav}_i)/(cnav_{\max} \cdot n_s - cnav_{\text{sum}})$
9  end
10 for $i = 0$ to $(n_p - 1)$ do
11     for $j = 0$ to $(n_h - 1)$ do
12         $n \leftarrow i \cdot n_h + j$
13         $\mathbf{x_{I,n}} \leftarrow \mathbf{x_I}$
14         $\alpha \leftarrow$ FINDINTERSECTION ($\int_{\alpha_{\min}}^{\alpha} \mathbf{cnav}, i/(n_p - 1)$)
15         $x_{F,n} \leftarrow x_I + d \cdot \cos(\alpha + \psi_I)$
16         $y_{F,n} \leftarrow y_I + d \cdot \sin(\alpha + \psi_I)$
17         $\psi_{F,n} \leftarrow \psi_I + \psi_{\min} + (\psi_{\max} - \psi_{\min}) \cdot j/(n_h - 1) + \alpha$
18     end
19 end
20 return $\mathbf{x_N}$
```

To generate the search spaces exhibited in Figure 8, we apply an algorithm similar to Algorithms 2 and 3 except that the sampling strategy is defined by some environmental constraints, such as centerlanes and road boundaries. As shown in Figure 8, in the presence of multiple valid lanes, we can easily generate single or full sets of trajectories to evaluate the relative cost of a lane change maneuver. The search space parameter array that corresponds to the provided example consists of road centerlines l_{center}, lane headings l_{heading}, lane widths l_{width}, vehicle width v_{width}, lane horizons d, the

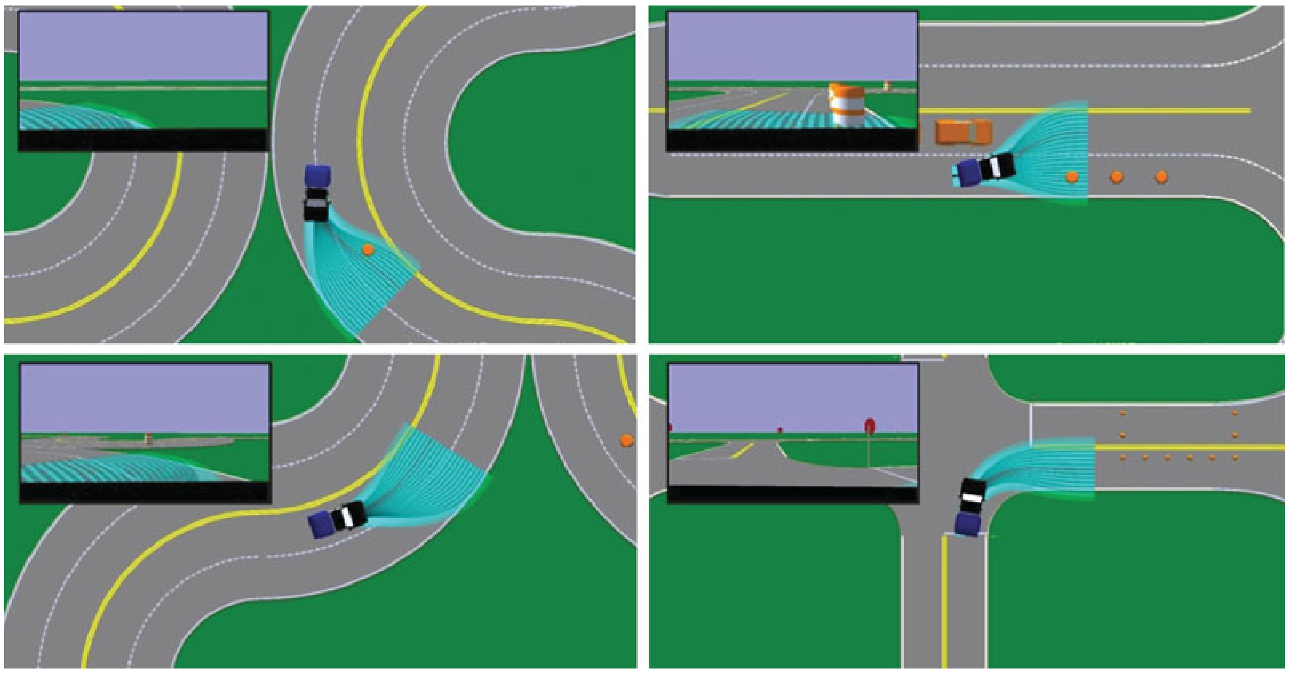


Figure 8. State-based sampling applied to navigation in road networks, an example of exploitation and satisfaction of environmental constraints. In each of these examples, a local motion-planning search space is generated that adapts to the shape of the road, inherently satisfying the environmental constraint of staying within or oriented along the lanes in the direction of travel.

number of lateral offsets to generate for each lane n_p, and the number of lanes itself n_l:

$$\mathbf{p_{ss}} = \begin{bmatrix} l_{\text{center}} \\ l_{\text{heading}} \\ l_{\text{width}} \\ v_{\text{width}} \\ d \\ n_p \\ n_l \end{bmatrix}. \tag{8}$$

Algorithm 4 generally operates by finding the state at some distance d along each lane defined in our search space parameter array. In this particular example, the terminal positions are sampled with a uniform lateral offset δ from a lane centerline at a fixed distance down the lane. Although it is not required, all of the terminal headings are constrained to be aligned with the tangent of the centerline.

Algorithm 4. GENERATELANEGUIDEDBOUNDARY STATES ($\mathbf{p_{ss}}$,$\mathbf{x_I}$)

```
   Input: $\mathbf{p_{ss}}$ (shape parameters), $\mathbf{x_I}$ (initial vehicle state)
   Output: boundary state pair set ($\mathbf{x_N}$)
1  for $i = 0$ to $(n_p - 1)$ do
2      for $j = 0$ to $(n_l - 1)$ do
3          $\mathbf{x_{center}} \leftarrow$ COMPUTESTATEATDISTANCEALONGLANE ($\mathbf{l_{center,j}}, d$)
4          $n \leftarrow i \cdot n_l + j$
5          $\mathbf{x_{I,n}} \leftarrow \mathbf{x_I}$
6          $\delta \leftarrow -0.5(\mathbf{l_{width,j}} - v_{\text{width}}) + (l_{\text{width}} - v_{\text{width}}) \cdot i / (n_p - 1)$
7          $\psi_{F,n} \leftarrow \mathbf{l_{heading,j}}$
8          $x_{F,n} \leftarrow x_{\text{center}} - \delta \sin(\psi_{\text{center}})$
9          $y_{F,n} \leftarrow y_{\text{center}} + \delta \cos(\psi_{\text{center}})$
10     end
11 end
12 return $\mathbf{x_N}$
```

This is important for two reasons. The first is that because all trajectories terminate at an equal distance along the path, the trajectory selection function is easier to design because they match better with the global guidance heuristic. The penalty for deviating from the centerline of the road shape is accounted for when all paths terminate at an equal distance along the road. Control-sampled search spaces that consider a constant distance for each trajectory do not have this feature, and this is why in part such approaches fail to navigate effectively in road networks. The second reason is in the control of the terminal heading at points along the path. It is important to consider whether an evasive action will inevitably lead the vehicle out of the lane, which would be unacceptable in a real-world application. The ability to control terminal heading and curvature ensures that the vehicle will stay in the lane after an evasive maneuver.

5. PRACTICAL CONSIDERATIONS

The ability to apply an arbitrary state-based sampling strategy is powerful, but actual implementation requires solutions to a few additional practical problems. This section will describe potential and applied solutions to several application challenges, including search space envelope determination, state sampling strategies, trajectory selection function, and model identification.

5.1. Adaptive Search Envelope Determination in Unconstrained Environments

The envelope of the adaptive search space depends heavily on the dynamic limitations of the mobile robot and the initial conditions. The shape and position of the search space horizon should adapt to the current speed because of braking distance and obstacle avoidance limitations. Curvature and angular acceleration limits may decrease the range of terminal headings.

An effective way to determine the envelope of the reachable set is simply to exhaustively and densely sample control space and record the extremes achieved in state space. Such a method would have to be employed offline, and it could not be used in rough terrain or for vehicles whose models change over time. Another approach is to exploit the continuity of the dynamic model and evaluate several aggressive maneuvers (such as max-turn right and left) to form rough bounds on reachable positions and headings at the horizon. This approach is simple and fast, and it can be used for adaptive vehicle models (whose motion model can change over time), and so we have preferred it over the offline method.

5.2. Sampling the Search Envelope

Whereas it is important to have an accurate representation of the envelope of the reachable set to constrain the search in regions where trajectories are likely to be feasible, another equally important consideration involves sampling of boundary states in this search envelope. Although it has already been mentioned that it is beneficial to focus the terminal state sampling in the direction of global guidance, it is also important to generate a search space that is reasonably well separated. Given the high number of freedoms still available in the definition of the boundary state pairs (including initial state curvature and velocity commands, terminal state positions, headings and curvatures, and velocities), it is important to generate rules based on the expressiveness of trajectories that adhere to the model dynamics to generate well-separated search spaces.

5.3. Trajectory Selection Function

In any approach in which an optimal trajectory must be determined from a local motion-planning search space, generating the correct optimization function for this procedure is important and often difficult. It is important to generate a cost function that correctly considers some mixture of risk, energy consumption, slope dwell (time spent on slopes), global guidance, smoothness, aggressiveness, and other "costs" that can be associated with a trajectory. Although most current approaches are hand tuned by robot operators, a promising approach is to apply machine learning techniques to discover the proper mapping from action and environment to cost, but this method still is human or self-supervised and requires a wealth of useful feature inputs and representative data for the training process. We have used a similar approach applied by other research projects by minimizing some weighted combination of integrated cost and steering activity.

5.4. Model Identification

It is important in the application of these model-predictive techniques to have an accurate trajectory prediction, as the validity of the motion plans is directly related to the fidelity of the forward vehicle model. If a high-fidelity vehicle model is available, we have the ability to directly execute commands from the local planner without the need for a path follower. In applications in which no model is available, the dynamics are too complex, or the model may degrade over time, it may be necessary to apply some feedback control to attempt to follow the selected trajectory. The use of predictive motion models in motion planning and navigation simplifies the obstacle avoidance problem by considering only dynamically feasible actions in its search space.

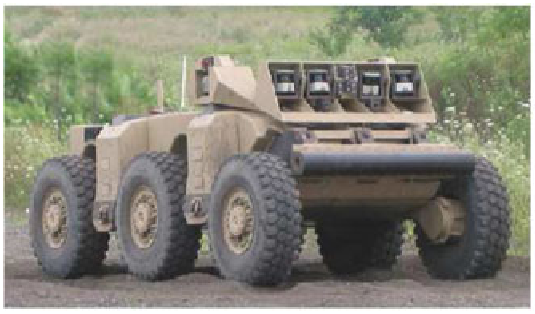
Methods applied for model identification on the two projects detailed in Section 6 consist of offline learning and parameterized model tuning from prior data. Neural networks have proven to be effective for offline learning of the forward predictive model (Bode, 2007), but it can be difficult to provide comprehensive training data. We have implemented parameterized models that represent the underlying physics tuned for accuracy from prior data logs, which has proven to be an effective, however inefficient, method. More sophisticated approaches involving online learning and real-time system identification will inevitably lead to better motion prediction and higher-performance navigators for complex environments.

6. EXPERIMENTS AND EXPERIMENTAL RESULTS

To evaluate the performance of our approach, the adaptive search space algorithm presented in Section 4 was tested in a series of simulation and field experiments. The simulation experiments are comparisons, in a series of randomized worlds, of our approach against constant curvature control based search spaces. The field results consist of two separate field robotic programs operating in complex and distinct environments: the off-road DARPA UGCV-PerceptOR Integrated (UPI) project and the Tartan Racing project that produced Carnegie Mellon's winning entry to the DARPA Urban Grand Challenge in 2007. The vehicles used on each of these programs are shown in Figure 9. Section 6.1 describes and details the simulation experiments, and Section 6.2 demonstrates new capabilities for the two field robotics projects.

6.1. Simulation Experiments

This section describes the set of simulation experiments performed to assess the value of the proposed methods in a setting where experiments can be controlled and a statistically significant number of them can be performed. Simulation is the most cost-effective way to judge the relative performance of the proposed algorithms because a sufficient quantity of field results would be prohibitively expensive to collect. Also, field experiments spanning long periods of time may be too difficult to control because conditions such as weather, visibility, and vehicle health will not be sufficiently controllable.


(a) Crusher

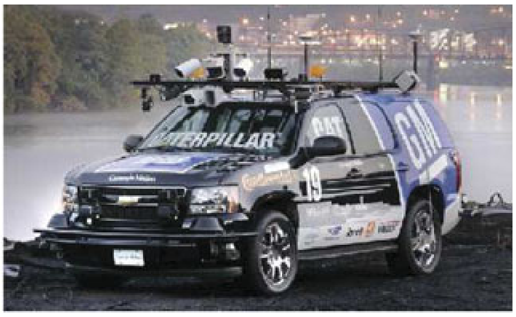

(b) Boss

Figure 9. The two mobile robot platforms used to perform the field experiments. (a) Crusher is a six-wheeled, skid-steered field robot for the DARPA UPI program. (b) Boss, a converted Chevrolet Tahoe, was the platform for Tartan Racing and was Carnegie Mellon's winning entry in the 2007 DARPA Urban Grand Challenge.

6.1.1. Simulation Setup

The experiment consisted of a vehicle driving through a randomly generated world populated with randomly sized circular obstacles, simulating a forest or boulder field environment. Obstacle positions and sizes are drawn from bounded uniform distributions. A region of points within a fixed radius from the vehicle represents the portion of the world known to the vehicle from its limited horizon perception system. This region is updated in each planning cycle with all obstacles. The vehicle model used for these simulations consists of a simple first-order (change in velocity is proportional to velocity error) approximation for linear and angular velocity based on observations from the Crusher platform. There is no uncertainty in this simulation; both vehicle model and obstacle observations are perfect.

During each planning cycle, the vehicle follows the path that was determined to be obstacle free for the greatest length. That selected path is followed for a distance corresponding to how far the vehicle would travel during the time required for planning and perception calculations. The global planner used is Field D* (Ferguson & Stentz, 2006). In the case of a tie, the path with the lowest Field D* distance to the goal (based on the portion of the world that is known at that point) is selected. In the simulation the goal is 1 km away and the local search space path lengths are 17 m. The method of evaluating a constant set of paths has been applied to the obstacle avoidance problem at least as far back as Daily et al. (1988). A view of the two search spaces, the environment, and a comparison of two runs can be found in Figure 10.

For simplicity, the vehicle cannot adjust its speed during a simulation. Instead, if the vehicle is in a position where is cannot avoid an obstacle at its fixed speed, then the vehicle heading is changed to point along the Field D* path to goal to simulate a stop-and-turn correction.

6.1.2. Simulation Results

One hundred simulated runs were performed for each of the search spaces. For each pair of runs, one simulated world was generated and both the constant curvature arc-based and adaptive search spaces were tested on that world. The two search spaces had equal numbers of trajectories. Simulations that ended without the vehicle reaching the goal were not used in the final results. A representative run from the comprehensive set is shown in Figure 10(c). Note that there were no successful simulations for both search spaces at 5 m/s for the higher obstacle densities, and so no data are provided in that region. On average, the overall path length was between 25% and 60% shorter in high-obstacle-density worlds for the adaptive search space as shown in Figure 11. This demonstrates that the improved flexibility and efficiency of the adaptive search space provides a performance advantage over constant-curvature search spaces in complex environments.

6.2. Field Experiments

As previously mentioned, the field experiments were conducted on two mobile robot platforms designed

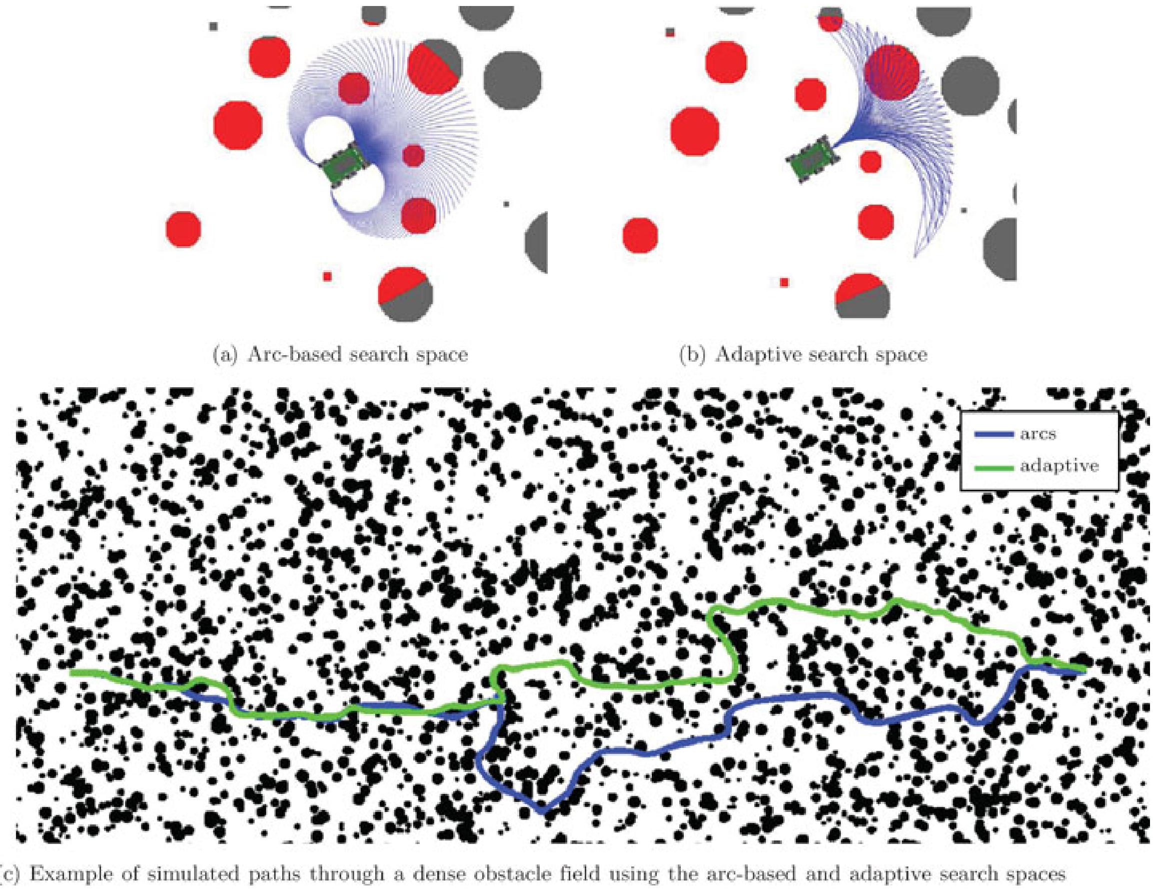

(a) Arc-based search space
(b) Adaptive search space
arcs
adaptive
(c) Example of simulated paths through a dense obstacle field using the arc-based and adaptive search spaces

Figure 10. Examples of the arc-based (a) and adaptive search space (b) driving through a simulated random world with a limited perceptual horizon. (c) Plots of the resulting paths for each search space for one simulated world.

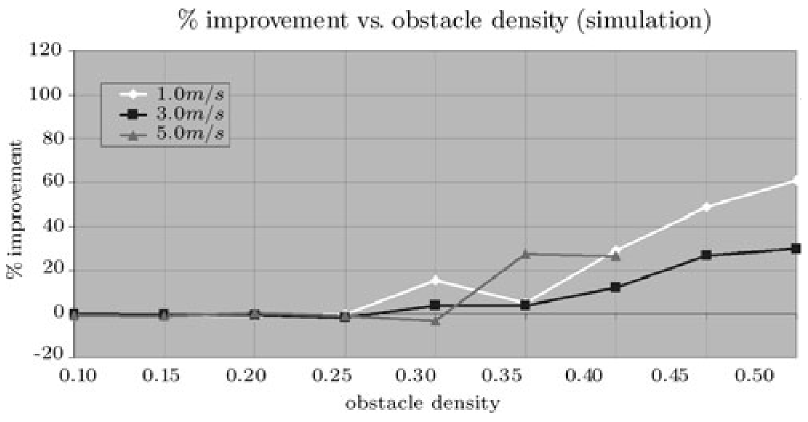


Figure 11. Simulation results. The percent improvement in path length provided by the adaptive search space compared to simple constant curvature controls for various densities and vehicle speeds.

for operation in different environments. The first set of experiments was performed on Crusher, an autonomous six-wheeled, skid-steered field robot designed to operate in the most difficult outdoor environments. The second set of field experiments was performed on Boss, a converted Chevrolet Tahoe designed for autonomous operation in urban environments. The Crusher experiments are a series of comparative tests to an arc-based search space, and the Boss experiments exhibit new capabilities gained and difficult problems made easy by applying this approach to highly constrained environments considering dynamic obstacles.

Boss and Crusher operated with similar navigation architectures. A global motion planner (Field D* for Crusher, a lane-based planner for Boss) provided global guidance to a local motion planner based on the presented adaptive search space algorithm to navigate complex environments. A testament to the value of model-predictive motion planners was that neither system relied on a path tracker; the controls generated by the model-predictive trajectory generator were accurate enough when executed without path-relative error feedback to ensure safe and effective navigation.

6.2.1. UPI/Crusher

Expressiveness of maneuvers is important for competent operation of mobile robots when obstacles are dense, speeds are high, or stopping and reorienting of the robot is undesirable or unacceptable. The autonomy system on Crusher has proven the capability to traverse more than 30 km/day under such circumstances with an average distance of more than 10 km between human interventions.

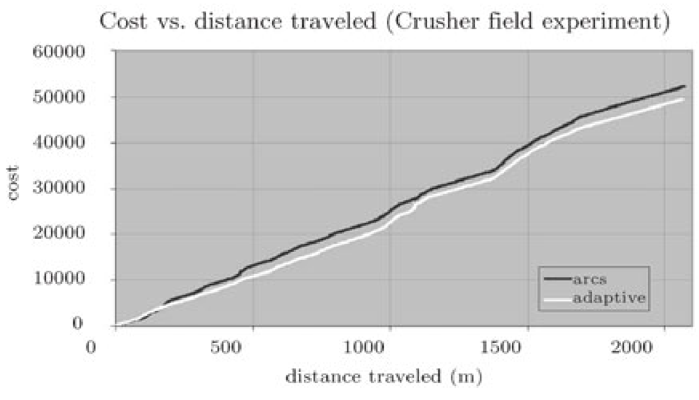
This adaptive search space has been used extensively on the Crusher platform during UPI field tests. For these field tests the adaptive search space was inserted into a full planning system. This system includes Field D* to provide global guidance and a series of reactive behaviors that are called to help the vehicle navigate through difficult situations that involve backing up. The adaptive search operates in the main behavior, which is responsible for forward navigation when there are no error conditions. These new trajectories replace a set of constant curvature controls that were used in that behavior for many years. Online, trajectories are selected to minimize the sum of the convolved cost along the trajectory, the Field D* distance to goal from the end of the trajectory, and a set of costs that penalize large curvature commands and deviation from the previously planned path.

It would be difficult and expensive to perform statistically significant tests of our hypothesis on this mobile robot. Routine software upgrades are performed several times per week in order to enhance the performance of the system in DARPA controlled field tests, and a freeze on all other development activities for our purposes for the times required is simply not feasible. Nonetheless, a small number of comparisons were completed. Two runs were completed on a short, 2-km course during which the only parameter changed was the set of trajectories used in the forward navigation behavior. The path traversed by the vehicle using each trajectory set is shown in Figure 12(b). As shown in Figure 12(a), the adaptive search space achieved a lower path cost early in the runs and maintained that advantage throughout the course. This set of runs was repeated three times. Averaging the results of these three runs, the adaptive search space had a 3.5% lower total path cost compared to constant curvature controls, where cost can be related to the roughness of the terrain or probability of mission failure.

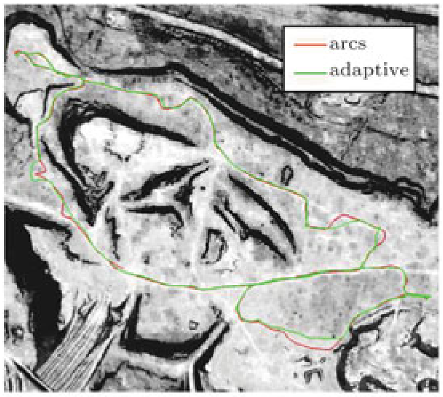
In addition to these specific comparisons, many subjective observations led the field team to believe that this search space offered some improvements over our legacy constant curvature controls. Because of these factors, this adaptive search space has been successfully used on the UPI program for 5 months with more than 250 autonomous kilometers driven during field tests.

6.2.2. Tartan Racing/Boss

The second platform for experiments in adaptive search spaces, Boss, was designed to compete in the 2007 DARPA Urban Grand Challenge, a 60-mile (96.5 km), 6-hour race through an urban environment. To date Boss has driven more than 2,000 autonomous miles using the described adaptive search space algorithm. This algorithm was chosen for key parts of the local motion planner design based on its ability to conform the search space to highly constrained road environments and its generation of trajectories that satisfy a representative dynamic model. The latter was especially important for navigation among movable obstacles, as we needed an accurate prediction of vehicle motion to evaluate the safety of each action. The application of motion models that do not accurately represent reality (instantaneous

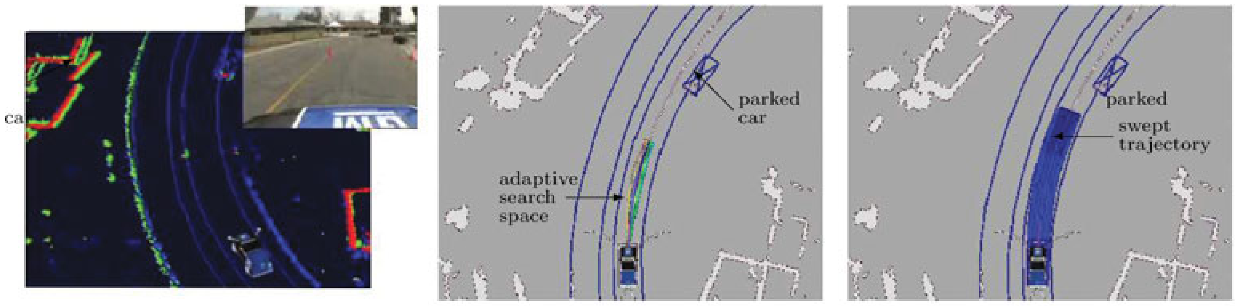


(a) Cost vs. distance for selected Crusher field experiment

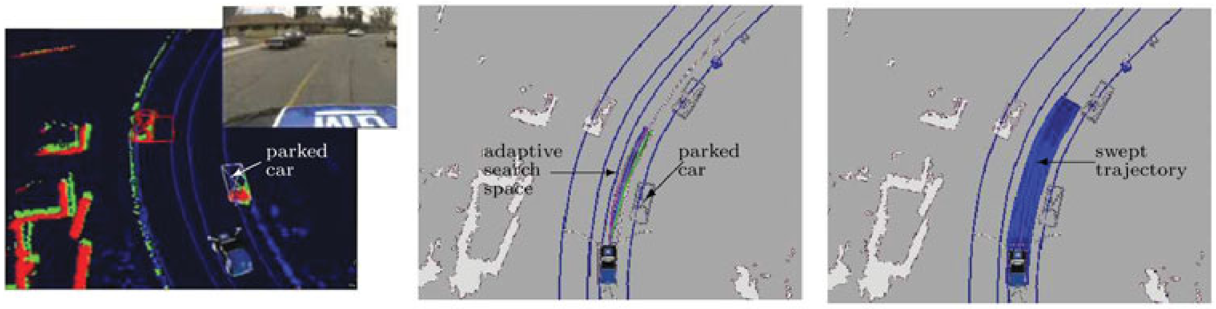


(b) Actual paths overlaid on prior map data

Figure 12. Comparative experiment between arc-based and adaptive search space local motion planners on the Crusher mobile robot. (a) Graph showing the total path cost as it accumulates along the paths followed by the vehicle for each search space. (b) The two paths overlaid on a cost map of the site.

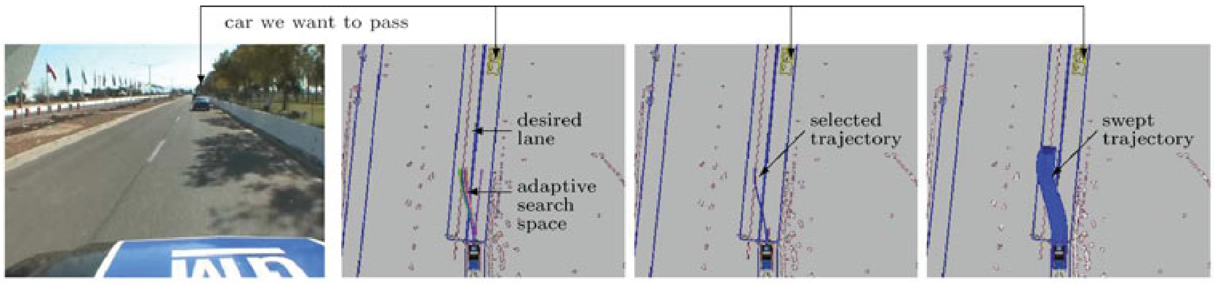


(a) Selecting centered trajectory (no obstacles in local search space)

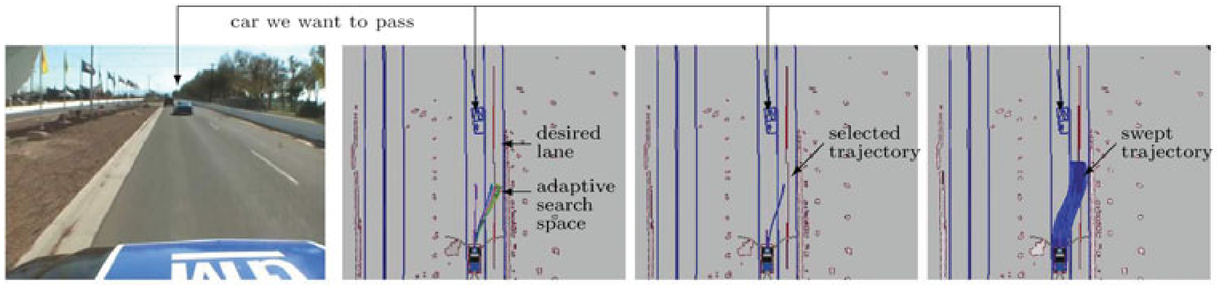


(b) Selecting noncentered trajectory to avoid parked car

Figure 13. In-lane navigation. These images show the local motion-planning search space of Boss navigating in a single-lane road. (a) The trajectory that acquires the centerline path of the lane is selected because of the absence of obstacles. (b) The vehicle selects an alternative trajectory to the left of the centerline to avoid the parked car obstructing the right-hand side of the lane.



(a) Double-lane-change maneuver (passing on the left)



(b) Double-lane-change maneuver (passing on the right)

Figure 14. Generation of search spaces for passing. The ability to choose an arbitrary state sampling strategy allows simple generation of local motion-planning search spaces for lane switching. Boss senses that it would like to pass a slower-moving car, so it generates and evaluates candidate actions for switching lanes.

velocity, infinite acceleration, the absence of angular velocity or angular acceleration limits) could lead to selecting suboptimal and/or dangerous maneuvers in the presence of both static and dynamic obstacles. This section will highlight two capabilities of applying this method to on-road navigation from the Urban Grand Challenge event that took place in Victorville, California, on November 3, 2007.

6.2.2.1. *In-Lane Navigation*

The majority of urban driving involves navigating in a single lane and adhering to local traffic laws. Typically, it is optimal to drive in the center of these road lanes as this provides the most distance between the vehicle and other obstacles in the environment. In some situations, it is necessary to make small adjustments inside the lane to avoid cones, potholes, pedestrians, parked cars, or other typical hazards. It is therefore necessary to not only generate trajectories that will drive the vehicle to the center of the road lane but to generate other candidate actions that offset the vehicle within the lane. Figure 13 shows two situations where the vehicle plans to drive down the center of the lane [Figure 13(a)] and the vehicle selects an action that drives the vehicle to the left (but still within the lane) to avoid the parked car that is obstructing the right-hand side of the lane [Figure 13(b)].

6.2.2.2. *Lane Switching*

When lanes can be blocked in a multilane road or when driving as fast as possible is important, the capacity to switch lanes is necessary. When a higher-level planner in the motion-planning hierarchy suggests to the local motion planner that it should switch lanes, the state space sampling strategy for the search space adapts to the desired lane and leverages the model-based trajectory generator to determine the actions necessary to reach those states. For example, consider Figure 14, where the vehicle is attempting to pass a slower-moving vehicle on a two-lane road. When Boss realizes that it is necessary to change lanes, a series of candidate actions to make the s-turn maneuver to change lanes are evaluated. Notice that a single trajectory to a forward point of the current lane is generated to provide an alternative action in the situation in which all trajectories that drive to the desired lane collide with obstacles, are infeasible, or are less optimal than driving in the current lane. Alternatively, a control space sampling technique that considered actions more sophisticated than arcs (e.g., clothoids) could also be used to generate the trajectories for lane switching. This approach would, however, be considerably less efficient as it could not guarantee that all of the actions generated satisfy the environmental constraints (staying within the road network) and thus would incur the increased computational burden of generating the large set of actions needed to densely search the more complex control space. Limiting the number of sampled actions negates the added computation requirements but sacrifices completeness as the control space is sampled sparsely. Using the state space sampling framework for navigating in constrained environments works well because it allows the local motion planner to focus its search in the direction of interest using more sophisticated control inputs.

7. CONCLUSIONS

We have leveraged our own recent work in model-predictive trajectory generation to create the capacity to efficiently generate state-based sampled, dynamically feasible search spaces. This approach generates search spaces that are more expressive and outperform contemporary approaches in input or control space when navigating in complex environments. This is important because the number of trajectories that we can evaluate is limited (because computational resources and replanning time are constrained) and more complete search spaces are more likely to find the optimal solution. We have demonstrated that this approach produces search spaces that are less influenced by the initial state of the vehicle, can exploit global guidance, and can satisfy environmental constraints, all within a simple unified framework. When navigating in an unconstrained environment, the terminal boundary states in the search space can be focused in the direction of minimum global cost. In a constrained environment, such as in lane following or navigating in traffic, difficult problems such as lane switching or in-lane obstacle avoidance become simple. The inherent feasibility of the search space provides the navigation function with better information to represent the actual result of a selected action. Because the actions consider predictive dynamic models, they can be applied directly without the explicit need for a path follower.

The provided simulation and field experiments demonstrate that the algorithm performs better when directly compared to a contemporary constant curvature input space sampling technique. We also advocate that this approach simplifies several relevant problems for mobile robot navigation in road networks, including lane changes and in-lane navigation. This algorithm has been used to generate local motion-planning search spaces for two outdoor mobile robots at speeds up to 30 mph (13.4 m/s), has been used to navigate thousands of autonomous kilometers in both constrained on-road and complex off-road environments, and was generally used as the search space generation component for Carnegie Mellon's winning entry in the 2007 Urban Grand Challenge.

ACKNOWLEDGMENTS

This work was sponsored by the Defense Advanced Research Projects Agency (DARPA) under contract Unmanned Ground Combat Vehicle PerceptOR Integration (contract number MDA972-01-9-0005). Additional field results are presented from Tartan Racing, with the generous support of our sponsors including General Motors, Caterpillar, and Continental and further supported by DARPA under contract HR0011-06-C-0142. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government.

REFERENCES

Berg, J., Ferguson, D., & Kuffner, J. (2006). Anytime path planning and replanning in dynamic environments (pp. 2366–2371). In Proceedings of the IEEE Conference on Robotics and Automation, Gaithersburg, MD. IEEE.

Berglund, T., Jonsson, H., & Soderkvist, I. (2003). An obstacle-avoiding minimum variation b-spline problem (pp. 156–161). In Proceedings of the 2003 International Conference on Geometric Modeling and Graphics, Los Alamitos, CA. Thousand Oaks, CA: Sage.

Besiadecki, J., Leger, P., & Maimone, M. (2007). Tradeoffs between directed and autonomous driving on the Mars exploration rovers. International Journal of Robotics Research, 26(91), 91–104.

Bode, M. (2007). Learning the forward predictive model for an off-road skid-steer vehicle (Tech. Rep. CMU-RI-TR-07-32). Pittsburgh, PA: Carnegie Mellon University.

Bonnafous, D., Lacroix, S., & Siméon, T. (2001). Motion generation for a rover on rough terrains (pp. 784–789). In Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems, Maui, HI. IEEE.

Chen, G., & Fraichard, T. (2007). A real-time navigation architecture for automated vehicles in urban environments (pp. 1223–1228). In Proceedings of the 2007 IEEE Intelligent Vehicles Symposium, Istanbul, Turkey. IEEE.

Coulter, R. (1992). Implementation of the pure pursuit path tracking algorithm (Tech. Rep. CMU-RI-TR-92-01). Pittsburgh, PA: Carnegie Mellon University.

Cremean, L., Foote, T., Gillula, J., Hunes, G., Kogan, D., Kriechbaum, K., Lamb, J., Leibs, J., Lindzey, L., Rasmussen, C., Stewart, A., Burdick, J., & Murray, R. (2006). Alice: An information-rich autonomous vehicle for high-speed desert navigation. Journal of Field Robotics, 9(23), 777–810.

Daily, M., Harris, J., Keirsey, D., Olin, K., Payton, D., Reiser, K., Rosenblatt, J., Tseng, D., & Wong, V. (1988). Autonomous cross-country navigation with the ALV (volume 2, pp. 718–726). In Proceedings of the IEEE International Conference on Robotics and Automation, Leuven, Belgium. IEEE.

Dubins, L. E. (1957). On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents. American Journal of Mathematics, 79, 497–516.

Ferguson, D., & Stentz, A. (2006). Using interpolation to improve path planning: The Field D* algorithm. Journal of Field Robotics, 23(2), 79–101.

Fox, D., Burgand, W., & Thrun, S. (1997). The dynamic window approach to collision avoidance. IEEE Robotics and Automation Magazine, 4, 22–33.

Frazzoli, E., Dahleh, M., & Feron, E. (2001). Real-time motion planning for agile autonomous vehicles (pp. 43–49). In Proceedings of the American Control Conference, Arlington, VA. IEEE.

Frazzoli, E., Dahleh, M. A., & Feron, E. (2003). A maneuver-based hybrid control architecture for autonomous vehicle motion planning (pp. 299–323). In Software enabled control: Information technology for dynamical systems. Piscataway, NJ: IEEE Press.

Green, C., & Kelly, A. (2007). Towards optimal sampling in space of paths. To appear in Proceedings of the International Symposium on Robotics Research, Hiroshima, Japan.

Haddad, H., Khatib, M., Lacroix, S., & Chatila, R. (1998). Reactive navigation in outdoor environments using potential fields (volume 2, pp. 1232–1237). In Proceedings of the 1998 IEEE Conference on Robotics and Automation, Leuven, Belgium. IEEE.

Howard, T., & Kelly, A. (2007). Optimal rough terrain trajectory generation for wheeled mobile robots. International Journal of Robotics Research, 26(2), 141–166.

Howard, T., Knepper, R., & Kelly, A. (2006). Constrained optimization path following of wheeled mobile robots in natural terrain (pp. 343–352). In Proceedings of the International Symposium on Experimental Robotics, Rio de Janeiro, Brazil. Berlin: Springer-Verlag.

Kelly, A., & Stentz, T. (1998). Rough terrain autonomous mobility—Part 2: An active vision and predictive control approach. Autonomous Robots, 5, 163–198.

Kelly, A., Stentz, T., Amidi, O., Bode, M., Bradley, D., Diaz-Calderon, A., Happold, M., Herman, H., Mandelbaum, R., Pilarski, T., Rander, P., Thayer, S., Vallidis, N., & Warner, R. (2006). Toward reliable off road autonomous vehicles operating in challenging environments. International Journal of Robotics Research, 25(5).

Koren, Y., & Borenstein, J. (1991). Potential field methods and their inherent limitations for mobile robot navigation (pp. 1398–1404). In Proceedings of the IEEE Conference on Robotics and Automation, Sacramento, CA.

Lacroix, S., Mallet, A., Bonnafous, D., Bauzil, G., Fleury, S., Herrb, M., & Chatila, R. (2002). Autonomous rover navigation on unknown terrains functions and integration. International Journal of Robotics Research, 21(10–11), 917–942.

Lacze, A., Moscovitz, Y., DeClaris, N., & Murphy, K. (1998). Path planning for autonomous vehicles driving over rough terrain (pp. 50–55). In Proceedings of the 1998 IEEE ISIC/CIRA/ISAS Joint Conference, Gaithersburg, MD.

Leedy, B., Putney, J., Bauman, C., Cacciola, S., Webster, J., & Reinholtz, C. (2006). Virginia Tech's twin contenders: A comparative study of reactive and deliberative navigation. Journal of Field Robotics, 23(9), 709–727.

Melchior, N., Kwak, J., & Simmons, R. (2007). Particle rrt for path planning in very rough terrain. In Proceedings of the NASA Science Technology Conference 2007.

Miller, I., Lupashin, S., Zych, N., Moran, P., Schimpf, B., Nathan, A., & Garcia, E. (2006). Cornell University's 2005 DARPA Grand Challenge entry. Journal of Field Robotics, 23(8), 625–652.

Shimoda, S., Kuroda, Y., & Iagnemma, K. (2005). Potential field navigation of high speed unmanned ground vehicles on uneven terrain (pp. 2828–2833). In Proceedings of the 2005 IEEE Conference on Robotics and Automation, Barcelona, Spain.

Simmons, R., Krotkov, E., Chrisman, L., Cozman, F., Goodwin, R., Hebert, M., Katragadda, L., Koenig, S., Krishnaswamy, G., Shinoda, Y., Whittaker, W., & Klarer, P. (1995). Experience with rover navigation for lunar-like terrains (volume 1, pp. 441–446.). In Proceedings of the 1995 IEEE Conference on Intelligent Robots and Systems.

Singh, S., Simmons, R., Smith, T., Stentz, T., Verma, V., Yahja, A., & Schwehr, K. (2000). Recent progress in local and global traversability for planetary rovers (pp. 1194–1200). In Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, CA.

Spenko, M., Kuroda, Y., Dubowsky, S., & Iagnemma, K. (2006). Hazard avoidance for high-speed mobile robots in rough terrain. Journal of Field Robotics, 23(5), 311–331.

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Goffmann, G., Lau, K., Oakley, C., Palatucci, M., Pratt, V., & Stang, P. (2006). Stanley: The robot that won the

DARPA Grand Challenge. Journal of Field Robotics, 23(9), 661–692.

Trepagnier, P., Nagel, J., Kinney, P., Koutsougeras, C., & Dooner, M. (2006). Kat-5: Robust systems for autonomous vehicle navigation in challenging and unknown terrain. Journal of Field Robotics, 23(8), 509–526.

Urmson, C., Ragusa, C., Ray, D., Anhalt, J., Bartz, D., Galatali, T., Gutierrez, A., Johnston, J., Harbaugh, S., Kato, H., Messner, W., Miller, N., Peterson, K., Smith, B., Snider, J., Spiker, S., Ziglar, J., Whittaker, W., Clark, M., Koon, P., Mosher, A., & Struble, J. (2006). A robust approach to high-speed navigation for unrehearsed desert terrain. Journal of Field Robotics, 23(8), 467–508.

Wettergreen, D., Tompkins, P., Urmson, C., Wagner, M., & Whittaker, W. (2005). Sun-synchronous robotics exploration: Technical description and field experimentation. International Journal of Robotics Research, 24(1), 3–30.