Improving the Performance of Sampling-Based Motion Planning With Symmetry-Based Gap Reduction

Peng Cheng, Emilio Frazzoli, and Steven LaValle

***Abstract*—Sampling-based nonholonomic and kinodynamic planning iteratively constructs solutions with sampled controls. A constructed trajectory is returned as an acceptable solution if its "gaps," including discontinuities within the trajectory and mismatches between the terminal and goal states, are within a given gap tolerance. For a given coarseness in the sampling of the control space, finding a trajectory with a small gap tolerance might be either impossible or extremely expensive. In this paper, we propose an efficient trajectory perturbation method, which complements existing steering and perturbation methods, enabling these sampling-based algorithms to quickly obtain solutions by reducing large gaps in constructed trajectories. Our method uses system symmetry, e.g., invariance of dynamics with respect to certain state transformations, to achieve efficient gap reduction by evaluating trajectory final state with a constant-time operation, and, naturally, generating the admissible perturbed trajectories. Simulation results demonstrate dramatic performance improvement for unidirectional, bidirectional, and PRM-based sampling-based algorithms with the proposed enhancement with respect to their basic counterparts on different systems: one with the second-order dynamics, one with nonholonomic constraints, and one with two different modes.**

***Index Terms*—Kinodynamic planning, motion planning, nonholonomic planning, symmetry.**

I. Introduction

Nonholonomic planning [1] and kinodynamic planning [2], called *motion planning with differential constraints* (MPD) [3], [4] together, attract significant research efforts because of their extensive applications in robotics, manufacturing, transportation, as well as nontraditional domains such as realistic computer animation [5], medical instrumentation [6], and verification [7]–[9]. While exact algorithms and two-stage methods [10]–[12] are, computationally, very expensive and limited to special systems, alternative methods [13]–[17] use sample controls to quickly construct approximate solutions for general systems, including those neither nilpotent nor differentially flat [18] while sacrificing completeness.

A shortcoming of these sampling-based algorithms is the fact that they construct trajectories that present a finite number of discontinuities, called "gaps" in this paper. A constructed trajectory is returned as a solution if its gaps are less than a given gap tolerance. Gaps, especially those in the middle of the trajectory, can severely degrade the quality of solutions by offsetting the final state away from the goal state. (In Fig. 3, a gap between x_{new} and x_e offsets the final state from x_{goal} to x_f.) Just as it is difficult for path planning to find a solution for problems with narrow passages [19], returning a solution satisfying a small gap tolerance tends to be extremely expensive or even impossible for MPD. For example, in [20], the set of points reachable by a class of systems subject to control inputs sampled from a discrete set has the structure of a lattice, thereby preventing the exact matching of the trajectory endpoint with the goal state. In cases in which the set of reachable states is everywhere dense, or even continuous [21], it is possible, in principle, to add a sequence of sampled controls to move the final state arbitrarily close to the goal state, but this is done at the expense of the efficiency of the trajectory and longer running time, since there will be fewer solutions and the search depth tends to increase.

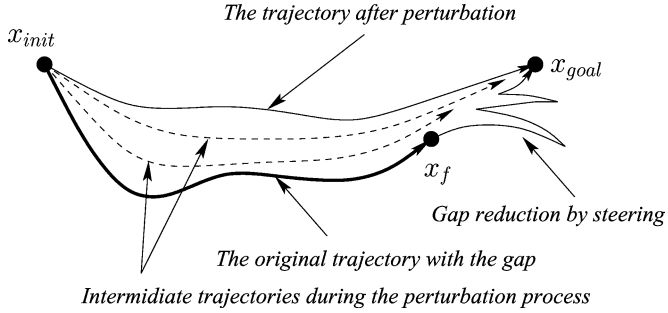

Fig. 1. Comparison of the gap reduction by perturbation and steering.

Noticing that trajectories with large gaps can be found much easier, a natural enhancement for sampling-based MPD is to use an additional gap-reduction method to reduce large gaps in candidate solution trajectories. If the large gaps can efficiently be reduced to satisfy the given tolerance, a solution is quickly constructed. The gap reduction is essentially a challenging two-point boundary value problem. Two methods are sketched in Fig. 1. One approach is to analytically construct a trajectory to steer the system across the gap. The method is efficient, but is only available for a limited class of systems [18], [22]–[26]. The other method is to minimize gaps by numerically perturbing the trajectory [27]–[29]. It is applicable for general systems, but needs expensive numerical integration in computation of the final state of the perturbed trajectory, and is difficult to incorporate constraints on the trajectory during the perturbation.

Our methods [30], [31] are similar to the latter approach, but dramatically increase efficiency of gap reduction using system symmetries, i.e., invariance of system dynamics with respect to a group of special transformation (called group action), to evaluate final state with a constant-time (with respect to integration accuracy) operation and to naturally maintain constraints along the perturbed trajectories. The cost of such symmetry-based gap reduction is that the application of the proposed method in MPD requires that: 1) the system has symmetry (might be neither nilpotent nor differentially flat), including most of mobile vehicle systems; 2) the trajectory with gaps includes special type of states (called coasting states), such that special type of trajectories (called coasting trajectories) can be inserted and perturbed; and 3) if two end states of a gap differ by not just a group action, a user-provided system-specific preprocessing procedure is able to make them so. Note that the second requirement is on the trajectory, for which symmetry is apparently a necessary condition, but we did not find sufficient conditions on system structures. The proposed approach was incorporated into existing sampling-based MPD, including the unidirectional and bidirectional Rapidly-exploring Random Tree (RRT) based planner [14] and a PRM-based planner [32]. Simulation results of the improved planners on different systems demonstrated the performance improvement with respect to their basic counterparts. Besides our work, another perturbation method is presented in [33] by tailoring the reactive path deformation method [34].

Manuscript received September 26, 2006; revised May 14, 2007. This paper was recommended by Associate Editor P. Rives and Editor L. Parker upon evaluation of the reviewers' comments. This work was supported by the National Science Foundation under CAREER Grant 9875304 (LaValle), Grant 0208891 (Frazzoli and LaValle), and Grant 0118146 (Bullo and LaValle).

P. Cheng is with the General Robotics, Automation, Sensing, and Perception (GRASP) Lab, University of Pennsylvania, Philadelphia, PA 19104-6228 USA (e-mail: chpeng@seas.upenn.edu).

E. Frazzoli is with the Aeronautics and Astronautics Department, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: frazzoli@mit.edu).

S. LaValle is with the Department of Computer Science, University of Illinois, Urbana, IL 61801 USA (e-mail:lavalle@cs.uiuc.edu).

Digital Object Identifier 10.1109/TRO.2007.913993

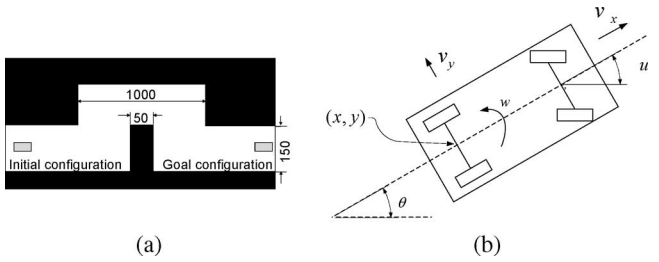

Fig. 2. Example of the MPD problem.

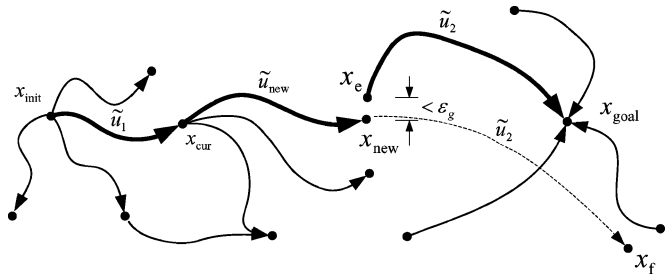

Fig. 3. Gap in a bidirectional sampling-based algorithm.

This paper is organized as follows. Section II describes the gap problems after a brief review of MPD and its sampling-based algorithm template. Section III presents the gap-reduction algorithm and its integration with sampling-based algorithms. Simulation results are provided in Section IV.

II. Gaps in Sampling-Based Planning

To be self-contained, we briefly review MPD problems and the sampling-based MPD template in [3], [4].

A. MPD Problems

An MPD problem is characterized by state space $X \subset \mathbb{R}^n$, input space $U \subset \mathbb{R}^m$ with $m \leq n$, state space obstacle as a closed set $X_{\text{obs}} \subset X$, initial state x_{init}, goal state x_{goal}, and motion equation as time invariant ODEs, $\dot{x} = f(x, u)$. Set $X_{\text{free}} = X \setminus X_{\text{obs}}$ is the *violation-free* open set. A *control*, denoted as $\tilde{u}$, represents a piecewise-continuous vector-valued function from $[0, t]$ to U for some $t > 0$. The *control space* $\mathcal{U}$ contains all possible controls for the system. The concatenation of two controls, $\tilde{u}_1$ and $\tilde{u}_2$, in $\mathcal{U}$ is defined as

$$(\tilde{u}_1 \tilde{u}_2)(t) = \begin{cases} \tilde{u}_1(t) & t \in [0, \bar{t}(\tilde{u}_1)) \\ \tilde{u}_2(t - t_1) & t \in [\bar{t}(\tilde{u}_1), \bar{t}(\tilde{u}_1) + \bar{t}(\tilde{u}_2)] \end{cases} \tag{1}$$

in which $\bar{t}(\tilde{u})$ gives the time duration of control $\tilde{u} \in \mathcal{U}$. The trajectory of the system subject to control $\tilde{u}$ from x_0 is $\Phi(\tilde{u}, x_0, t) = x_0 + \int_0^t f(\tilde{x}(\tau), \tilde{u}(\tau)) d\tau$. A control $\tilde{u}$ is an exact solution if: 1) $\Phi(\tilde{u}, x_{\text{init}}, t) \in X_{\text{free}}$ for all $t \in [0, \bar{t}(\tilde{u})]$ and 2) $\Phi(\tilde{u}, x_{\text{init}}, \bar{t}(\tilde{u})) = x_{\text{goal}}$.

An MPD problem is shown in Fig. 2(a), in which a car with realistic second-order dynamics is required to move from the initial to the goal configuration with 60 mi/h constant forward speed while avoiding obstacles. The state of the car [see Fig. 2(b)] is $(x, y, \theta, v_y, \omega)$, in which x, y, and θ represent position and orientation, ω is the angular velocity, and v_y is translational velocity perpendicular to the forward direction. The input u is the steering angle. The motion equation is

$$\begin{aligned} \dot{x} &= v_x \cos(\theta) - v_y \sin(\theta) \qquad & \dot{\theta} &= \omega \\ \dot{y} &= v_x \sin(\theta) + v_y \cos(\theta) \qquad & \dot{\omega} &= \frac{(f_{\text{yf}} a - f_{\text{yr}} b)}{I} \\ \dot{v}_y &= -v_x \omega + \frac{(f_{\text{yf}} + f_{\text{yr}})}{M} \end{aligned} \tag{2}$$

in which $f_{\text{yf}} = -C_f((v_y + a\omega)/v_x - u)$ and $f_{\text{yr}} = -C_r(v_y - b\omega)/v_x$ are forces on tires, v_x is the constant forward velocity, and a, b, C_f, C_r, M, I are constant parameters (see [35] for details). The initial and goal configuration, both at rest ($v_y = \omega = 0$ be zero), are, respectively, shown in Fig. 2(a).

B. Gaps Generated in Sampling-Based Algorithms

Sampling-based MPD incrementally constructs a search graph to search for a solution. A node in the graph represents a state, and an edge represents a sample control. For a given MPD problem and a gap tolerance ϵ_g, a sampling-based planning template [3], [4] is as follows.

Step 1: The search graph is initialized with one, two, or more starting states, which are, respectively, called unidirectional, bidirectional, and PRM-based methods. In Fig. 3, the search graph of a bidirectional method is initialized with the initial state x_{init} and goal state x_{goal}.

Step 2: Use a given global search strategy to select a state in the search graph, e.g., x_{cur} in Fig. 3.

Step 3: Use a given local planner to generate a new trajectory segment from the selected state with a sample control, e.g., the segment of $\tilde{u}_{\text{new}}$ from x_{cur} in Fig. 3.

Step 4: Use a given updating policy to update the search graph with respect to the new trajectory segment.

Step 5: Use a given solution checking criterion to determine whether a constructed trajectory from x_{init} to x_{goal} exists to satisfy the gap tolerance. If so, the sample controls along the trajectory are concatenated as the solution. In Fig. 3, the thick curves satisfy the gap tolerance, and the returned solution is $\tilde{u} = \tilde{u}_1 \tilde{u}_{\text{new}} \tilde{u}_2$.

Step 6: Go to Step 2 until a given termination condition is satisfied.

Gaps could be induced in Step 1 between the trajectory starting state and x_{init} if the starting states do not include x_{init}, or in Step 4 between two consecutive trajectory segments if the control of a segment (e.g., $\tilde{u}_{\text{new}}$ in Fig. 3) does not exactly connect the starting states of these two segments, or in Step 5 between the trajectory final state and x_{goal} when the trajectory final state does not equal to x_{goal}.

III. Motion Planning With Gap Reduction

In this section, we present the gap-reduction method after reviewing the notion of symmetry and coasting trajectories for dynamical systems. Finally, the proposed method is incorporated into sampling-based MPD algorithms.

A. Coasting Trajectories of Systems With Symmetry

Let $\mathcal{G}$ be a Lie group, with identity element e. A (left) group action of $\mathcal{G}$ on the state is a smooth map $\Psi : \mathcal{G} \times X \to X$ such that: 1) $\Psi(e, x) = x$, for all $x \in X$ and 2) $\Psi(g, \Psi(h, x)) = \Psi(gh, x)$, for all $g, h \in \mathcal{G}$, and $x \in X$. We will often use the shorthand $\Psi_g(x)$ to indicate $\Psi(g, x)$. A group $\mathcal{G}$ is a symmetry group for a control system if the dynamics of the latter is invariant with respect to the action of $\mathcal{G}$, i.e., if $\Psi_g \circ \Phi(\tilde{u}, \cdot, \bar{t}(\tilde{u})) = \Phi(\tilde{u}, \cdot, \bar{t}(\tilde{u})) \circ \Psi_g$. Typically, man-made vehicles and mobile robots are invariant with respect to certain classes of rigid-body motions, i.e., to actions of subgroups of SE(3). While

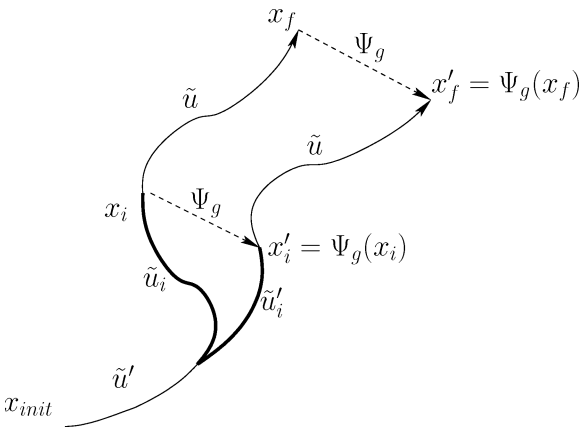

Fig. 4. Perturbing $\tilde{u}_i$ to avoid reintegration of $\tilde{u}$ in calculating x'_f.

there is no systematic way to deduce invariance properties of a control system, these can often be deduced from observation and first-principle arguments (see [36]).

State $x_0 \in X$ and input $u_0 \in U$ are, respectively, called the *coasting state* and *coasting input* if we can obtain the *coasting trajectory* $\Phi(\tilde{u}, x_0, t) = \Psi(\exp(\xi t), x_0)$ with $\tilde{u}(t) = u_0$, in which ξ is an element of the Lie algebra of $\mathcal{G}$. Two boundary states of a coasting trajectory differ by a group action, which is called the *group displacement* of the coasting trajectory and can be parameterized by the trajectory duration.

The car system (2) is invariant with respect to rigid-body motions in the horizontal plane, i.e., to actions of SE(2). Its coasting state and input satisfy $aC_r(v_y - b\omega) + av_x^2 M\omega + bC_r(v_y - b\omega) = 0$ and $u = [C_r(v_y - b\omega) + v_x^2 M\omega + C_f(v_y + a\omega)]/C_f v_x$.

B. Efficient Gap Reduction With Symmetry

For a given MPD problem, a control $\tilde{u}$, a gap tolerance ϵ_g, and a distance function,[1] we will show how to use symmetry to efficiently perturb $\tilde{u}$ into $\tilde{u}'$ such that the gap distance between the final and goal states is less than ϵ_g.

1) Efficient Final State Evaluation: Constant-time (with respect to integration accuracy) calculation of the final state using symmetry is one of the key factors for efficiency of our method, because such calculation is extensively used in gap reduction. The idea is shown in Fig. 4. By symmetry, if perturbing $\tilde{u}_i$ into $\tilde{u}'_i$ makes x_i differ from x'_i by a group action Ψ_g, then, the new final state x'_f equals $\Psi_g \circ x_f$ without reintegrating $\tilde{u}$ from x'_i. If a given trajectory has multiple coasting states, we can insert coasting trajectories after these states, and the final state can be adjusted by perturbing durations of these inserted trajectories. In Fig. 5, the trajectory of $\tilde{u}$ has three coasting state and input pairs, denoted as (x_i, u_i) for $i = 1, 2$, and 3 (see top picture). These states divide $\tilde{u}$ into four controls $\tilde{u}_1, \tilde{u}_2, \tilde{u}_3$, and $\tilde{u}_4$ with durations t_1, t_2, t_3, and t_4, respectively. Let $\Phi_{\tilde{u}}^{\bar{t}(\tilde{u})} = \Phi(\tilde{u}, \cdot, \bar{t}(\tilde{u})) : X \to X$. We have $x_f = \Phi_{\tilde{u}_4}^{t_4} \circ \Phi_{\tilde{u}_3}^{t_3} \circ \Phi_{\tilde{u}_2}^{t_2} \circ \Phi_{\tilde{u}_1}^{t_1}(x_{\text{init}})$. Coasting trajectories (thick curves) are inserted after these coasting states. These coasting controls, denoted as $\tilde{u}'_i$, have constant inputs u_i and changeable durations δt_i. The new final state is

$$\begin{aligned} x'_f &= \Phi_{\tilde{u}_4}^{t_4} \circ \Phi_{\tilde{u}'_3}^{\delta t_3} \circ \Phi_{\tilde{u}_3}^{t_3} \circ \Phi_{\tilde{u}'_2}^{\delta t_2} \circ \Phi_{\tilde{u}_2}^{t_2} \circ \Phi_{\tilde{u}'_1}^{\delta t_1} \circ \Phi_{\tilde{u}_1}^{t_1}(x_{\text{init}}) \\ &= \Phi_{\tilde{u}_4}^{t_4} \circ \Psi_{h_3} \circ \Phi_{\tilde{u}_3}^{t_3} \circ \Psi_{h_2} \circ \Phi_{\tilde{u}_2}^{t_2} \circ \Psi_{h_1} \circ \Phi_{\tilde{u}_1}^{t_1}(x_{\text{init}}) \\ &= \Psi_{h_3} \circ \Psi_{h_2} \circ \Psi_{h_1}(x_f) \end{aligned} \tag{3}$$

in which Ψ_{h_i} is the group displacement of the i-th coasting trajectory.

[1]The distance function is required to be continuous. Its value is nonnegative and will be zero if and only if the two states are equal.

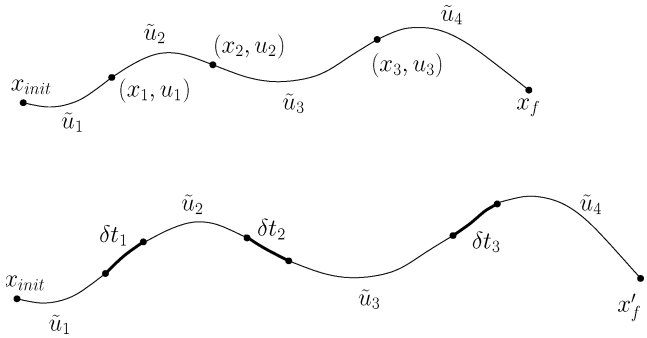

Fig. 5. Trajectory perturbation by inserting coasting trajectories.

Note that the calculation of the new final state only needs to evaluate Ψ_{h_i} without reintegrating the original controls. These group actions need, at most, the numerical integration of the coasting trajectories, and could even be evaluated analytically when the symmetry group is the finite product of subgroups of SE(3), e.g., $\mathbb{R}$, S^1, and SE(2). Calculating the durations of the coasting trajectories to achieve the desired group action is also called *trajectory planning via inverse kinematics* in [23]. Since analytical solutions are only available for few cases [37], gradient-based optimization is used in this paper to compute the coasting durations.

2) A Heuristic to Select Coasting Trajectories for Optimization: Generally, if the symmetry group is n_g-dimensional, perturbing durations of n_g coasting trajectories could eliminate the gap. If the number of coasting trajectories is larger than n_g, then, the following heuristic is used to perturb the trajectory iteratively to avoid expensive evaluation of a high-dimensional gradient vector. In each iteration, only a subset of all coasting trajectories are perturbed to reduce the gap.

Observing that most gradient-based optimization techniques employ the steepest descent method as their starting step, it is expected that a nonlinear program with better convergence rate at the starting point for the steepest descent method might have a good chance to converge faster. Therefore, we calculate the convergence rate $1/\alpha$ of the steepest descent method [38] for the nonlinear program at the starting point as follows

$$\alpha^2 = 1 - \frac{\left(\sum_i \varsigma_i^2 \lambda_i^2\right)^2}{\left(\sum_i \varsigma_i^2 \lambda_i^3\right)\left(\sum_i \varsigma_i^2 \lambda_i\right)} \tag{4}$$

in which $\{\lambda_i\}$ are eigenvalues of Jacobian matrix J and $\{\varsigma_i\}$ are coefficients of linear decomposition of $x_f - x_{\text{goal}}$ with respect to eigenvectors of J. In each iteration, multiple subsets of coasting trajectories are chosen, and the subset with the highest convergence rate is used to reduce the gap.

The intuition of (4) is schematically illustrated in Fig. 6(f). Assume that the trajectory of a system moving in a plane has three coasting trajectories. Vectors v_1, v_2, and v_3 are the partial derivatives of the final state with respect to durations of coasting trajectories, respectively. Perturbing durations of two coasting trajectories tend to move the final state along some direction in the cone between their partial derivative vectors. Thus, perturbing coasting trajectories of v_2 and v_3 is less likely to reduce the gap than perturbing those of v_1 and v_3. Equation (4) shows that the convergence rate of v_1 and v_3 is larger.

C. Incorporating Gap Reduction With Sampling-Based Planning

For bidirectional methods (see Fig. 3), the following procedure is used before checking whether the gap distance between x_{new} and x_e is less than gap tolerance ϵ_g.

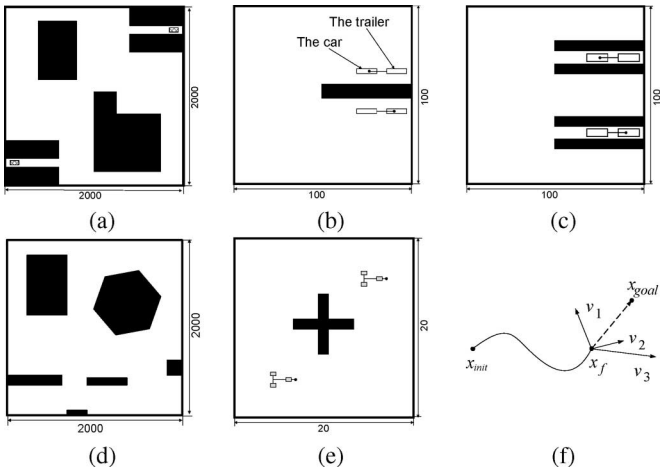

Fig. 6. (a)–(e) MPD problems used in the simulation. (f) Intuition of selecting coasting trajectories.

1) *Detect Solution Candidate:* If the gap distance is less than a given large gap tolerance, a solution candidate is found; otherwise, report that no solution is detected. The large gap tolerance is normally chosen empirically for different problems.
2) *Make Two Gap States Differ Only by a Group Action:* Retrieve the control $\tilde{u}$ of the trajectory from x_{init} to x_{new}. Check whether $\tilde{u}$ can be changed into $\tilde{u}'$ with a given preprocessing procedure such that the new final state x'_{new} differs from x_e by only a group action. If not, report that no solution is detected. If such a given preprocessing procedure is not available, an extra small gap tolerance could be enforced in the previous step besides the large gap tolerance. A solution candidate is detected when there exists a group action such that the distance from the goal state to the final state under the group action is less than this extra gap tolerance.
3) *Eliminate Gap:* Find the coasting states along the trajectory of $\tilde{u}'$. If there are no coasting states, report that no solution is detected. Otherwise, use an optimization solver to perturb the trajectory to eliminate the gap for a given number of iterations. In each iteration, a subset of coasting trajectories are selected to setup an optimization problem using (4) in Section III-B.2. When the optimization terminates, the perturbed trajectory is kept if it satisfies all constraints or discarded otherwise. If the final gap distance is less than ϵ_g, a solution is returned, or report no solution is detected otherwise.

For unidirectional methods, the search graph is built only from x_{init}. A solution is detected if the final state of a trajectory from x_{init} is in the ϵ_g neighborhood of a goal state. Therefore, the earlier technique is called before checking whether the new state of a trajectory from x_{init} is in the ϵ_g neighborhood of a goal state.

For PRM-based planning methods, connecting two states is a critical component. For path planning problems, the connection of two states is simple and efficient. However, for MPD problems, it is very difficult. When analytical connection methods are not available, sampling-based algorithms can be used as the connection methods, and therefore, the gap problem is induced. A constructed trajectory from x_{init} to x_{goal} can have multiple gaps. We can simply use the unidirectional and bidirectional algorithms improved with the proposed gap reduction as connection methods to solve the gap problem.

IV. Simulation Studies

We did extensive simulation to compare the performance of various sampling-based MPD algorithms with or without gap reduction on problems with different systems. To evaluate randomized sampling-based algorithms, simulation results were based on 20 trials of each planner and problem pair. Due to limited computational resources, each trial terminates if no solution is returned after 400000 iterations if no other termination condition is specified. All simulations were done on a 2.0 GHz PC running Linux. The Numerical Algorithms Group (NAG) library is used to solve the optimization in the gap reduction.

The gap distance is calculated with $\sum_{i=1}^{n} \|x_i, x'_i\|^2 w_i$, in which $\{x_i\}$ and $\{x'_i\}$ are the state variables and w_i is the associated weight. The function $\|x_i, x'_i\|$ equals $\min(|x_i - x'_i|, 2\pi - |x_i - x'_i|)$ if x_i denotes the orientation, or $|x_i - x'_i|$ otherwise. All distances are measured in feet and time in seconds. The large gap tolerance to detect solution candidates is 100.0, and the default solution gap tolerance is 0.1. Assume that there are N_c coasting states along a trajectory, our experiences showed that the improved planners achieved better performance when the number of chosen subsets equals $\lceil N_c/6 \rceil$ or 30 if $N_c/6 > 30$. Each subset includes $3n_g$ coasting trajectories.

A. Systems in the Simulation

Three systems are used in our simulation. Specially, the finite friction model of the roller racer [39] is a switch system with two different modes for which there are no existing results, showing that it is either nilpotent or differentially flat up to now to the best of our knowledge.

The first system in (2) considers the second-order dynamics with significant drift. The preprocessing procedure is to eliminate difference in $z = [v_y w]^T$ between two states, which is to solve a two-point boundary value problem for the linear system, $\dot{z} = Az + Bu$, in which

$$A = \begin{bmatrix} -\dfrac{C_f + C_r}{v_x M} & \dfrac{C_r b - C_f a}{v_x M} - v_x \\ \dfrac{C_r b - C_f a}{v_x I} & \dfrac{b^2 C_r + a^2 C_f}{v_x I} \end{bmatrix} \qquad B = \begin{bmatrix} -\dfrac{C_f}{M} \\ \dfrac{C_f a}{I} \end{bmatrix}. \tag{5}$$

It can be verified that using two constant controls with values c_1 and c_2 and the same duration δt will convert the previous linear ODE into linear discrete equation such that the controls to eliminate difference in z can be computed. The gap distance weights are $1, 1, 1, 1$, and 100, respectively, for v_y, ω, x, y, and θ. The weight for θ is specially chosen to be 100, since a small variation in orientation could greatly change the final state of a trajectory.

The second system is a variation of the nonholonomic car-and-trailer system [25] such that the existing analytical steering method is not directly applicable (explained in the model description). Its state is $(x, y, \theta_1, \beta, \theta_2)$, in which x, y, θ_1, and β are the configuration and steering angle of the car and θ_2 is the orientation of the trailer. The system must satisfy

$$|\theta_1 - \theta_2| < \frac{\pi}{2}. \tag{6}$$

The system input is (u_1, u_2), in which $u_1 \in [0, 2.0]$ is the forward velocity, and $u_2 \in [-0.24, 0.24]$ is the changing rate of the steering angle. Note that u_1 is restricted to be nonnegative, so that the system is not small time locally controllable (STLC), and the gap cannot be reduced by trivially moving along the direction of vector fields generated by the Lie bracket. The motion equation is

$$\begin{aligned} \dot{x} &= u_1 \cos(\theta_1)\ \dot{y} = u_1 \sin(\theta_1) \\ \dot{\theta}_1 &= \frac{u_1 \tan(\beta)}{L_1}\ \dot{\beta} = u_2 \\ \dot{\theta}_2 &= \frac{u_1 \sin(\theta_1 - \theta_2)}{L_2} \end{aligned} \tag{7}$$

TABLE I
RUNNING TIME WITH DIFFERENT GAP TOLERANCES

ϵ_g	Max. Time	Min. Time	Avg. Time	Succ.
100.0	158.7	2.6	29.7105	20
10.0	7113.7	2.6	1992.7	20
1.0	60736.2	2075.8	30576.3	11
0.1	61785.7	60168.1	60736.2	0

in which $L_1 = 2.0$ is the length of the car, and $L_2 = 10.0$ is the length of the hitch. The system has symmetry group SE(2). With transformation $\theta_d = \theta_1 - \theta_2$, the state is transformed into (β, θ_d), and the last equation in (7) is changed to $\dot{\theta}_d = u_1 \tan(\beta)/L_1 - u_1 \sin(\theta_d)/L_2$. Then, it is easy to see that the coasting input and state satisfy $u_2 = 0, \tan(\beta)/L_1 - \sin(\theta_d)/L_2 = 0$.

Assume that two gap states, respectively, have (β, θ_d) and (β', θ'_d) with $\theta'_d > \theta_d$, the preprocessing procedure consists of three steps: 1) set u_2 to be the maximal value and u_1 zero, to increase β to its maximal value while keeping θ_d constant; 2) set u_2 to be zero and u_1 maximal, to increase θ_d to θ'_d while keeping β constant; 3) set u_2 to be the minimal value and u_1 zero, to decease β to β' while keeping θ_d constant. The gap distance weights are $1, 1, 10, 1$, and 10, respectively, for x, y, θ_1, β, and θ_2.

The last system is a finite friction model for the roller racer [39]. The model considers the second-order dynamics and switches between two modes. One is the sticking mode, in which the rolling without slipping constraints are held on both front and back wheels, and the other is the slipping mode, in which the front wheel slips on the ground. When the friction between the wheel and ground cannot provide enough constraint force to refrain the wheel from slipping, the system will switch into the slipping mode. By controlling the steering angle, the system will be able to move forward and stop. A planning method is provided in [39] to design motion between any two points in SE(2) with zero velocities when no obstacles exist. Here, we use the proposed method to plan motions while considering obstacles. The system has symmetry group SE(2). Coasting states include all states in the sticking mode with zero steering angle velocity. The weight for position and velocity is 1 and for the orientation is 50. To include coasting states in the constructed trajectories, the trajectory segment in Step 3 in Section II-B is generated with sinusoidal controls, which is conditionally followed by a constant steering angle control if the system is not in the sticking mode at the end of the sinusoidal control. The stop control in [39] works as a preprocessing procedure.

B. Computational Cost of Sampling-Based MPD for Small Gap Tolerance

We ran several experiments using the bidirectional RRT-based planner [14] to solve the problem in Fig. 6(a) for different values of the gap tolerance; the results are reported in Table I. It can be seen that the computational cost is very expensive when the gap tolerance is small. Note that with gap tolerances 1.0 and 0.1, the planner, respectively, only found 11 and 0 solutions over 20 trials. It is expected that the average time for gap tolerance 1.0 and 0.1 would be much bigger if the algorithm finds 20 solutions. These data strongly suggest the necessity of planning with gap reduction when a small gap tolerance is required.

C. Comparisons With a Classical Numerical Method

We also implemented a classical numerical method to reduce the gaps, in which the final states were calculated by numerically integrating the perturbed controls. Both the proposed method and this classical one were incorporated with a unidirectional RRT-based planner [14]. Each sample control is constant and parameterized by an m-dimensional vector for the input and a real number for the duration. A constructed control, which consists of k sample controls, has $(m+1)k$ parameters. The classical method reduces the gaps by perturbing these $(m+1)k$ parameters. The 4th order Runge–Kutta numerical integration is used, and each integration step is over 0.01 sec. duration. Simulation results from respective improved planners on problems in Figs. 2 and 6(b) are reported in Table II, in which T_{all} is the overall running time, $N_{\int}$ is the number of numerical integrations, `Ob.` denotes whether obstacles are considered, `Symm.` denotes whether symmetry is used, `Pl.` shows whether the planner is unidirectional (`uni.`) or bidirectional (`bi.`), `Sel.` shows whether selecting coasting trajectories with (4) is used, and `Succ.` shows the number of returned solutions over 20 trials. The running time of the algorithm using the proposed method is, respectively, about 3 and 1000 times less than the classical method. In the last row in the table, no results are reported since the classical method failed to return a solution in four days in the first trial.

TABLE II
COMPARISON OF PLANNERS WITH OR WITHOUT USING SYMMETRY

	Ob.	Sy.	Pl.	Sel.	T_{all}	$N_{\int}$	Succ.
Car	N	Y	uni.	Y	7.5e2	8.2e7	20
Car	N	N	uni.	Y	2.3e4	4.5e9	20
Car	Y	Y	uni.	Y	3.4e4	7.9e7	20
Car	Y	N	uni.	Y	9.1e4	9.5e9	20
Trailer	N	Y	uni.	Y	2.2e2	9.4e6	20
Trailer	N	N	uni.	Y	2.4e5	4.7e10	20
Trailer	Y	Y	uni.	Y	3.3e2	1.1e7	20
Trailer	Y	N	uni.	Y	-	-	-

One reason of the dramatic performance improvement is that the algorithm using symmetry generated about 100 times less number of numerical integrations than that of the classical method. Another important reason is that the proposed symmetry-based perturbation method naturally generates admissible trajectories, while the classical method does not. For the problem in Fig. 6(b), a large amount of time of the classical method was wasted to generate perturbed trajectories that violated the constraint in (6) even though their gaps were eliminated. This means that the proposed symmetry-based method is more suitable than the classical method for problems with systems that have stringent constraints on the trajectory.

D. Effects of Selecting Coasting Trajectories

The unidirectional planners using gap reduction with or without using (4) in selecting coasting trajectories were used to solve the problems in Figs. 2 and 6(b). To compare only the effects of coasting trajectory selection, obstacles in the problems are ignored. Similar results as Table II are reported in Table III except that N_{opt}, instead of $N_{\int}$, is provided to compare the number of the NAG optimization function calls. From the table, observe that the overall time and number of calls for the planner with coasting trajectory selection are smaller. This demonstrates the effectiveness of the proposed heuristic in selecting coasting trajectories.

E. Performance Improvements of the Improved Uni- and Bidirectional Planners

The proposed gap reduction was used to improve both uni- and bidirectional planners. Simulation results of the unidirectional planner

TABLE III
EFFECTS OF COASTING TRAJECTORY SELECTION

	Ob.	Sy.	Pl.	Sel.	T_{all}	$N_{\int}$	Succ.
Car	N	Y	uni.	Y	2.4e3	1.4e3	20
Car	N	Y	uni.	N	3.7e3	5.3e3	20
Trailer	N	Y	uni.	Y	4.6e2	3.2e3	20
Trailer	N	Y	uni.	N	6.1e2	1.3e4	20

TABLE IV
SIMULATION RESULTS OF THE IMPROVED UNI- AND BIDIRECTIONAL PLANNERS

	Ob.	Sy.	Pl.	Sel.	T_{all}	$N_{\int}$	Succ.
Car	Y	Y	bi.	Y	4.4e3	n/a	20
Trailer	Y	Y	bi.	Y	1.4e4	n/a	20
Roller Racer	Y	Y	uni.	Y	2.2e3	n/a	20

TABLE V
SIMULATION RESULTS OF THE IMPROVED PRM-BASED PLANNER

V.N.	E.N.	C. T.	Q. T.	Succ.
50	36	2.1e3	4.6e4	4
100	158	7.9e3	5.7e4	9
150	371	1.8e4	6.4e4	15
200	593	2.6e4	9.0e4	20
250	783	4.1e4	1.4e5	34

for problems in Figs. 2 and 6(b) are already shown in Table II. Results from the bidirectional planner for problems in Fig. 6(a) and (c) and from the improved unidirectional planner for the problem in Fig. 6(e) with the roller racer system are shown in Table IV. The basic counterparts of these improved planners did not return a solution for these problems over 20 trials for average 1.5 h running time.

F. PRM-Based Planner With Gap Reduction

The improved bidirectional planner was used in the basic PRM-based method as the connection method to solve the problem in Fig. 6(d) with the car system. The construction and query processes alternated as follows. Every time, after the construction process inserted 50 new vertices into the road map, we will query solutions for 40 randomly chosen initial and goal state pairs. In the construction phase, a sampling point tries to connect to, at most, 20 neighbors, and each connection runs for 2000 iterations. In the query phase, a sampling point tries to connect to, at most, 40 neighbors, and each connection runs for 20000 iterations to fully utilize the constructed road map. The results are shown in Table V, in which `V.N.` and `E.N.`, respectively, denote the number of vertices and edges in the road map, `Q.T.` and `C.T.` are, respectively, the overall query time and construction time, and `Succ.` is the number of returned solutions over 40 queries. We can see that with the proposed gap reduction, the PRM-based method could be used to solve a kinodynamic planning problem without using an analytical steering method.

V. CONCLUSION

In this paper, we proposed a symmetry-based gap-reduction method to reduce the sensitivity of the performance of sampling-based MPD with respect to the gap tolerance. Simulation results demonstrated that the performance of different sampling-based MPD algorithms, including unidirectional, bidirectional, and PRM-based methods, was dramatically improved with the proposed approach to solve MPD problems with systems which do not have analytical steering solutions. There are many places for future improvement, e.g., systematic ways to design the preprocessing procedures and how to reduce gaps even when the candidate trajectory does not have coasting states, which will make the proposed gap reduction have more general applications.

ACKNOWLEDGMENT

The authors would like to thank F. Lamiraux, J.-P. Laumond, S. Lindemann, and anonymous reviewers for helpful discussions and comments.

REFERENCES

[1] J.-P. Laumond, "Trajectories for mobile robots with kinematic and environment constraints," in *Proc. Int. Conf. Intell. Auton. Syst.*, 1986, pp. 346–354.
[2] B. R. Donald, P. G. Xavier, J. Canny, and J. Reif, "Kinodynamic planning," *J. ACM*, vol. 40, pp. 1048–66, Nov. 1993.
[3] P. Cheng, "Sampling-based motion planning with differential constraints" Ph.D. dissertation, Univ. Illinois, Urbana, IL, Aug. 2005.
[4] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2006. [Online]. Available: http://planning.cs.uiuc.edu/
[5] M. Lau and J. J. Kuffner, "Behavior planning for character animation," in *Proc. Eurographics/SIGGRAPH Symp. Comput. Anim.*, 2005, pp. 271–280.
[6] R. Alterovitz, M. Branicky, and K. Goldberg, "Constant-curvature motion planning under uncertainty with applications in image-guided medical needle steering," in *Algorithmic Found. Robot. VII*, S. Akella, N. Amato, W. Huang, and B. Misha, Eds. Berlin, Germany: Springer-Verlag, Jul. 2006.
[7] A. Bhatia and E. Frazzoli, "Incremental search methods for reachability analysis of continuous and hybrid systems," in *Hybrid Systems: Computation and Control*, R. Alur and G. J. Pappas, Eds. Berlin, Germany: Springer-Verlag, 2004, pp. 67–78, Lecture Notes in Computer Science, 2993.
[8] P. Cheng and V. Kumar, "Sampling-based falsification and verification of controllers for continuous dynamic systems," in *Algorithmic Found. Robot. VII*, S. Akella, N. Amato, W. Huang, and B. Misha, Eds. Berlin, Germany: Springer-Verlag, 2006.
[9] J. Esposito, J. W. Kim, and V. Kumar, "Adaptive RRTs for validating hybrid robotic control systems,". (2004, Jul.). *presented at the Workshop Algorithmic Found. Robot.,* Zeist, The Netherlands, Jul. 2004, 2004.
[10] P. Cheng, G. Pappas, and V. Kumar, "Decidability of motion planning with differential constraints," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr., Apr. 2007, pp. 1826–1831.
[11] K. G. Shin and N. D. McKay, "Minimum-time control of robot manipulators with geometric path constraints," *IEEE Trans. Autom. Control*, vol. 30, no. 6, pp. 531–541, Jun. 1985.
[12] S. Sekhavat and J.-P. Laumond, "Topological property for collision-free nonholonomic motion planning: The case of sinusoidal inputs for chained-form systems," *IEEE Trans. Robot. Autom.*, vol. 14, no. 5, pp. 671–680, Oct. 1998.
[13] J. Barraquand and J.-C. Latombe, "Nonholonomic multibody mobile robots: Controllability and motion planning in the presence of obstacles," *Algorithmica*, vol. 10, pp. 121–155, 1993.
[14] P. Cheng and S. M. LaValle, "Resolution complete rapidly-exploring random trees," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2002, pp. 267–272.
[15] E. Frazzoli, M. A. Dahleh, and E. Feron, "Real-time motion planning for agile autonomous vehicles," *AIAA J. Guid. Control*, vol. 25, no. 1, pp. 116–129, 2002.
[16] A. M. Ladd and L. E. Kavraki, "Motion planning in the presence of drift, underactuation and discrete system changes," in *Robot.: Sci. Syst.*, 2005.
[17] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *Int. J. Robot. Res.*, vol. 20, no. 5, pp. 378–400, May 2001.
[18] M. Fliess, J. Lévine, P. Martin, and P. Rouchon, "Flatness and defect of non-linear systems: Introductory theory and examples," *Int. J. Control*, vol. 61, no. 6, pp. 1327–1361, 1995.
[19] Z. Sun, D. Hsu, T. Jiang, H. Kurniawati, and J. Reif, "Narrow passage sampling for probabilistic roadmap planners," *IEEE Trans. Robot.*, vol. 21, no. 6, pp. 1105–1115, Dec. 2005.
[20] A. Bicchi, A. Marigo, and B. Piccoli, "On the reachability of quantized control systems," *IEEE Trans. Autom. Control*, vol. 47, no. 4, pp. 546–563, Apr. 2002.

[21] E. Frazzoli, M. A. Dahleh, and E. Feron, "Maneuver-based motion planning for nonlinear systems with symmetries," *IEEE Trans. Robot.*, vol. 21, no. 6, pp. 1077–1091, Dec. 2005.
[22] D. J. Balkcom and M. T. Mason, "Time optimal trajectories for bounded velocity differential drive vehicles," *Int. J. Robot. Res.*, vol. 21, no. 3, pp. 199–217, 2002.
[23] F. Bullo and K. M. Lynch, "Kinematic controllability for decoupled trajectory planning in underactuated mechanical systems," *IEEE Trans. Robot. Autom.*, vol. 17, no. 4, pp. 402–412, Aug. 2001.
[24] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *Am. J. Math.*, vol. 79, pp. 497–516, 1957.
[25] R. M. Murray and S. Sastry, "Nonholonomic motion planning: Steering using sinusoids," *IEEE Trans. Autom. Control*, vol. 38, no. 5, pp. 700–716, 1993.
[26] J. A. Reeds and L. A. Shepp, "Optimal paths for a car that goes both forwards and backwards," *Pacific J. Math.*, vol. 145, no. 2, pp. 367–393, 1990.
[27] A. W. Divelbiss and J. T. Wen, "A path-space approach to nonholonomic planning in the presence of obstacles," *IEEE Trans. Robot. Autom.*, vol. 13, no. 3, pp. 443–451, Jun. 1997.
[28] H. B. Keller, *Numerical Methods for Two Point Boundary Value Problems*. New York: Dover, 1992.
[29] G. Oriolo, S. Panzieri, and G. Ulivi, "Learning optimal trajectories for nonholonomic systems," *Int. J. Control*, vol. 73, no. 10, pp. 980–991, 2000.
[30] P. Cheng, E. Frazzoli, and S. M. LaValle, "Exploiting group symmetries to improve precision in kinodynamic and nonholonomic planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct., Oct. 2003, pp. 631–636.
[31] P. Cheng, E. Frazzoli, and S. M. LaValle, "Improving the performance of sampling-based planners by using a symmetry-exploiting gap reduction algorithm," in *Proc. IEEE Int. Conf. Robot. Autom.*, May, May 2004, pp. 4362–4368.
[32] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Autom.*, vol. 12, no. 4, pp. 566–580, Jun. 1996.
[33] F. Lamiraux, E. Ferre, and E. Vallee, "Kinodynamic motion planning: Connecting exploration trees using trajectory optimization methods," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2004, pp. 3987–3992.
[34] F. Lamiraux, D. Bonnafous, and O. Lefebvre, "Reactive path deformation for non-holonomic mobile robots," *IEEE Trans. Robot.*, vol. 20, no. 6, pp. 967–977, Dec. 2004.
[35] P. Cheng and S. M. LaValle, "Reducing metric sensitivity in randomized trajectory design," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2001, pp. 43–48.
[36] F. Bullo and A. D. Lewis, *Geometric Control of Mechanical Systems*. Berlin: Springer-Verlag, 2004.
[37] S. Martinez, J. Cortes, and F. Bullo, "A catalog of inverse-kinematics planners for underactuated systems on matrix lie groups," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2003, pp. 625–630.
[38] J. R. Shewchuk. (1994). "An introduction to the conjugate gradient method without the agonizing pain," [Online]. Available: "http://www-2.cs.cmu.edu/∼jrs/jrspapers.html".
[39] P. Cheng, E. Frazzoli, and V. Kumar, "Motion planning for the roller racer with a sticking/slipping switching model," in *Proc. IEEE Int. Conf. Robot. Autom.*, MayMay 15–19, 2006, pp. 1637–1642.

Localization and Matching Using the Planar Trifocal Tensor With Bearing-Only Data

J. J. Guerrero, A. C. Murillo, and C. Sagüés

***Abstract*—This paper addresses the robot and landmark localization problem from bearing-only data in three views, simultaneously to the robust association of this data. The localization algorithm is based on the 1-D trifocal tensor, which relates linearly the observed data and the robot localization parameters. The aim of this work is to bring this useful geometric construction from computer vision closer to robotic applications. One contribution is the evaluation of two linear approaches of estimating the 1-D tensor: the commonly used approach that needs seven bearing-only correspondences and another one that uses only five correspondences plus two calibration constraints. The results in this paper show that the inclusion of these constraints provides a simpler and faster solution and better estimation of robot and landmark locations in the presence of noise. Moreover, a new method that makes use of scene planes and requires only four correspondences is presented. This proposal improves the performance of the two previously mentioned methods in typical man-made scenarios with dominant planes, while it gives similar results in other cases. The three methods are evaluated with simulation tests as well as with experiments that perform automatic real data matching in conventional and omnidirectional images. The results show sufficient accuracy and stability to be used in robotic tasks such as navigation, global localization or initialization of simultaneous localization and mapping (SLAM) algorithms.**

***Index Terms*—Bearing-only data, 1-D trifocal tensor, global localization, robot vision, robust matching, SLAM initialization.**

I. Introduction

When an unknown scene is observed from multiple unknown positions, a complex but well-known geometric problem appears. The goal is to associate the observations and to recover the robot and landmark locations. Let us focus on the case of 1-D bearing-only observations while performing planar robot motion, which is typical for robots working in man-made environments. Robot localization based on bearing-only data has been considered in autonomous guided vehicles using landmarks of known location [1] and also in simultaneous localization and mapping (SLAM), where the initialization and the data matching are difficult problems [2].

This geometric problem is similar to the structure from motion problem studied in computer vision [3], but specialized for 1-D observations and 2-D locations. In computer vision, it is usual to relate different views with an initial matching of relevant features, followed by a robust estimation of a projective transformation or a tensor [4]. These tensors provide general constraints between the bearing-only observations, and allow us to recover the camera localization from them. For example, the fundamental matrix has been extensively used for two-view robust matching [5], since it provides a general constraint for 2-D bearing-only observations. Recently, it has been applied to help loop closing methods in SLAM [6].

However, using 1-D bearing-only data, two views do not provide a constraint between observations; at least three views are required.

Manuscript received December 20, 2006; revised October 22, 2007. This paper was recommended for publication by Associate Editor J. Santos-Victor and Editor L. Parker upon evaluation of the reviewers' comments. This work was supported in part by project MEC DPI2006-07928 and in part by project IST-1-045062-URUS-STP. This paper was presented in part at the IEEE ICRA 2006, Orlando, FL, and in part at the IEEE IROS 2006, Beijing, China.

The authors are with the DIIS-I3A, University of Zaragoza, 50018 Zaragoza, Spain (e-mail: jguerrer@unizar.es, acm@unizar.es, csagues@unizar.es).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2008.918043