Gradient-Informed Path Smoothing for Wheeled Mobile Robots

Eric Heiden[1], Luigi Palmieri[2], Sven Koenig[1], Kai O. Arras[2] and Gaurav S. Sukhatme[1]

***Abstract*— Planning smooth trajectories is important for the safe, efficient and comfortable operation of mobile robots, such as wheeled robots moving in crowded environments or cars moving at high speed. Asymptotically optimal sampling-based motion planners can be used to generate such trajectories. However, to achieve the necessary efficiency for the real-time operation of robots, one often uses their initial feasible trajectories or the trajectories of non-optimal motion planners instead, typically after a post-smoothing step. We propose a gradient-informed post-smoothing algorithm, called GRIPS, that deforms given trajectories by locally optimizing the placement of vertices while satisfying the system's kinodynamic constraints. We show experimentally that GRIPS typically produces trajectories of significantly smaller length and higher smoothness than several existing post-smoothing algorithms.**

I. Introduction

The generation of smooth robot motion is important in robotics, such as to achieve legible navigation in crowded environments [1], [2] and comfortable robot and car motion that respects kinodynamic constraints [3]. In recent years, several asymptotically optimal sampling-based motion planners have been introduced (e.g., RRT* [4], PRM* [4], RRT$^\#$ [5], FMT* [6] and SORRT* [7]) which, given enough time, yield high-quality (that is, minimum-cost) paths. A *de facto* standard technique to generate smooth paths in less time than asymptotically optimal motion planners is the combination of a sampling-based or discrete motion planner (e.g., Theta* [8] and A* [9]) with a post-smoothing algorithm [10], [11], [12] that improves the feasible path returned by the motion planner. Common smoothing operations utilized by such algorithms are short-cutting [10], removal of redundant vertices or solving of an optimization problem initialized with the feasible path [13], [14], [15].

Following this line of work, we introduce a new path-smoothing technique called **Gr**adient-**I**nformed **P**ath **S**moothing (GRIPS). Unlike previous techniques, GRIPS combines the advantages of gradient-based optimization and short-cutting. Instead of only skipping redundant vertices on the feasible path, GRIPS also deforms its shape by locally optimizing the placement of the vertices. After this gradient-based path-deformation phase, GRIPS prunes the path in a cost-aware short-cutting phase that ensures that the path remains sufficiently far away from obstacles (see Fig. 1) while adhering to the kinodynamics of the system.

[1]Eric Heiden, Sven Koenig and Gaurav S. Sukhatme are with the Department of Computer Science, University of Southern California, Los Angeles, USA {heiden, skoenig, gaurav}@usc.edu,

[2]Luigi Palmieri and Kai O. Arras are with Robert Bosch GmbH, Corporate Research, Germany {Luigi.Palmieri, KaiOliver.Arras}@de.bosch.com.

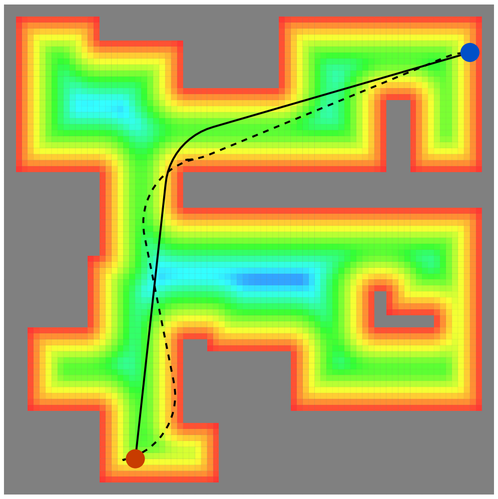


Fig. 1: An example path generated by Theta* with Reeds-Shepp steering using **Gr**adient-**I**nformed **P**ath **S**moothing (GRIPS) (solid line). GRIPS generates a shorter and smoother path than the initial path (dashed line) by dynamically deforming it over an obstacle distance field (red and blue represent closest and furthest distances, respectively).

We evaluate GRIPS experimentally on paths provided by different motion planners (RRT [16], PRM [17], RRT*, RRT$^\#$ and Theta*) for kinodynamic systems realized by different steer functions (POSQ [18], Reeds-Shepp [19], G^1 clothoids [20] and Dubins [21]). Our results indicate that, on average, GRIPS generates shorter (smaller arc length) and smoother (lower maximum curvature) paths than several state-of-the-art path-smoothing techniques, while finding feasible paths in more cases and being as fast as them.

This paper is structured as follows: We first review related work in Sec. II and then describe GRIPS in Sec. III. Finally, we show our experiments and results in Sec. IV and Sec. V, respectively.

II. Related Work

Several approaches have been introduced in recent years for improving the quality of feasible paths generated by motion planners.

Hsu et al. [10] describe a short-cutting technique that removes redundant motions between nearby vertices on the given path by repeatedly selecting two non-consecutive vertices on the path and trying to connect them directly via a steer function. Geraerts et al. [11] describe an extension of this technique that repeatedly applies a shortcut to a randomly chosen system dimension. GRIPS, instead of only

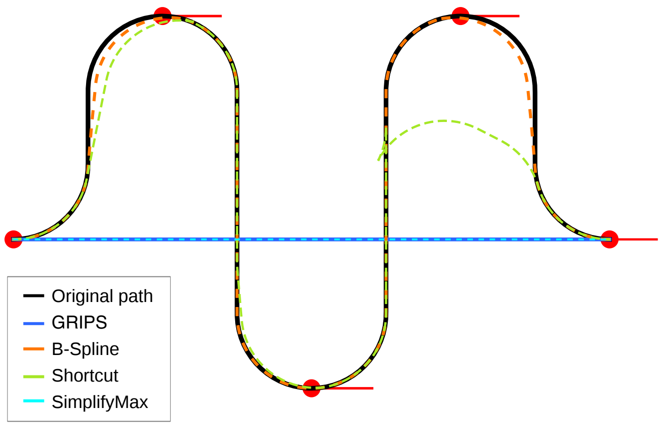


Fig. 2: Post-smoothing of a hand-crafted path (black) with GRIPS (blue solid line) and other path-smoothing techniques (dashed lines). Red circles with outgoing red lines indicate the $\mathrm{SE}(2)$ vertices on the given path with their respective yaw angles.

removing redundant vertices on the path, also deforms its shape by locally optimizing the placement of the vertices, taking nearby obstacles and the kinodynamic constraints of the system into account to generate shorter and smoother paths that remain sufficiently far away from obstacles.

Raveh et al. [22] describe a hybridization technique that operates on several given paths instead of a single one. It first finds feasible connections between the paths and then identifies a new (hybrid) path of high quality via classical search on the graph given by all paths and their connections. Luna et al. [12] describe a variant of this hybridization technique that converts a non-optimal motion planner to one with any-time properties. Their method, when given more planning time, first uses the motion planner to generate an additional path and then alternates between short-cutting and hybridization of the best k previously generated paths. GRIPS, on the other hand, does not require a motion planner to be called repeatedly. It directly operates on a single given path to run fast.

Yang et al. [23] describe a spline-based interpolation technique that repeatedly fits a G^1- and G^2-continuous cubic Bézier curve through a sequence of vertices on the given path and then replaces the corresponding path segment with the curve if the curve does not collide with obstacles. GRIPS, on the other hand, takes nearby obstacles into account to ensure that the path remains sufficiently far away from obstacles.

Several motion-optimization techniques optimize the shape of the given path to improve its quality with respect to a given cost function that typically includes the length and smoothness of the path. For example, Quinlan et al. [15] optimize the shape of the path with gradient descent based on an elastic force in the configuration space and tend to generate paths that are very close to obstacles. Ratliff et al. [13] optimize the shape of the path with covariant gradient descent. Their `CHOMP` technique works well as local optimizer for holonomic systems but does not handle the inequality constraints common for nonholonomic systems. Schulman et al. [14] optimize the shape of the path with sequential quadratic programming after linearizing the dynamics of the system. The latter two motion-optimization techniques use a term in the cost function to penalize collisions with obstacles. However, they may not find feasible paths in very cluttered environments. Furthermore, they work only on paths whose number of vertices is fixed. GRIPS, on the other hand, also inserts, removes and moves vertices on the path to generate shorter and smoother paths that remain sufficiently far away from obstacles even in very cluttered environments.

III. Our Approach

We first present our terminology and notation in Sec. III-A and then describe GRIPS in Sec. III-B.

A. *Terminology and Notation*

For wheeled mobile robots, a path σ is defined as a sequence of $\mathrm{SE}(2)$-vertices connecting a given start state $\mathbf{x}_s$ to a given goal state $\mathbf{x}_g$, $\sigma = \{\mathbf{x}_0 = \mathbf{x}_s, \mathbf{x}_2, \ldots, \mathbf{x}_{N-2}, \mathbf{x}_{N-1} = \mathbf{x}_g\}$ via N interpolated states. During their search in the configuration space $\mathcal{X}$, motion planners try to connect the start state $\mathbf{x}_s \in \mathcal{X}$ and goal state $\mathbf{x}_g \in \mathcal{X}$ by finding several intermediary states that lead to the goal, while ensuring the geometric and kinematic feasibility of these connections. States can be randomly sampled (as done by sampling-based motion planners [4], [17], [16]) or generated from deterministic tessellations of the configuration space (as done by grid- and lattice-based searches [8], [9]). To ensure kinodynamic feasibility, the connection of two adjacent states $\mathbf{x}$ and $\mathbf{x}'$ is often formulated as a *two-point boundary value problem* (2P-BVP), which is commonly solved by a steer function $s(\mathbf{x}, \mathbf{x}')$. A steer function returns a sequence of vertices (path or trajectory) σ' as numerical solution to the 2P-BVP. The environment is defined by a rectangular grid consisting of square-shaped cells with $1\,\mathrm{m}$ side length. A grid cell can either be occupied or free and is indexed by a pair of coordinates x, y denoting the horizontal and vertical position on the grid, respectively.

B. *Gradient-Informed Path Smoothing*

The objective of a post-smoothing algorithm is to improve the quality of a given feasible path σ generated by a motion planner. GRIPS is inspired by the split-and-merge algorithm presented in [24], which linearly approximates curves by subdividing line segments at vertices of higher error and subsequently merging adjacent segments while maintaining a defined error bound. Given a path σ and a steer function s, GRIPS (Alg. 1) works in two similar stages. In the first phase, it deforms the path by moving and inserting vertices (Alg. 2) such that the distance from obstacles is increased and enough vertices are sampled in areas closer to obstacles. In the second phase, it removes vertices from the path using a cost-aware short-cutting mechanism (Alg. 3).

1) Gradient-Based Path Deformation (Alg. 2): In the first phase, GRIPS moves and inserts vertices via *gradient descent* on the obstacle distance field D (see Fig. 1 for an example) where the obstacle distance is computed as follows:

$$D[p] = \min_{o \in \mathcal{O}} ||o - p||_2,$$

where p is the continuous vertex position (assumed to be two-dimensional in this work), and $\mathcal{O}$ is the set of coordinates of occupied grid cells. Once an array of grid cell distances to the static obstacles has been precomputed, $D[p]$ can be determined in constant time by linearly interpolating between the grid cells adjacent to p.

The gradient ∇D of the distance field is approximated as follows:

$$\nabla D[p] = \begin{pmatrix} \dfrac{D[p.x-\epsilon,\ p.y] - D[p.x+\epsilon,\ p.y]}{2\epsilon} \\ \dfrac{D[p.x,\ p.y-\epsilon] - D[p.x,\ p.y+\epsilon]}{2\epsilon} \end{pmatrix}$$

for sufficiently small $\epsilon > 0$.

Gradient descent moves the vertices p by $-\eta \nabla D[p]/D[p]$ where η is the step size (η_0 being the initial value) which is multiplied in every gradient-descent round by discount factor $\gamma \in (0, 1]$. $-\eta \nabla D[p]$ defines the direction on the distance field pointing away from closest obstacles. Dividing this direction by $D[p]$ alters the magnitude of the gradient-descent step to move vertices which are close to obstacles away from them by a larger degree compared to vertices that are further away from obstacles. The number of gradient-descent rounds K in combination with η_0 and γ further affects how far the vertices will be moved away from obstacles. During each gradient-descent round, a new vertex is inserted at positions of the path which are close to obstacles. If many vertices are placed close to each other, interpolating the path becomes increasingly more expensive due to the high number of 2P-BVP instances to be solved. Therefore, a minimum distance d_{min} between consecutive vertices on the path is maintained. The resulting path provides the subsequent cost-aware short-cutting phase (Alg. 3) with more opportunity to prune the path as vertices further away from obstacles are more likely to be directly connectable by a steer function compared to vertices which are "trapped" near obstacles.

2) Cost-Aware Path Short-Cutting (Alg. 3): In the second phase, a short-cutting technique is used to remove vertices that cause unnecessary turns and thus high arc lengths and curvatures. It first identifies vertices which cannot be skipped by directly steering from their predecessors to their successors. Then it constructs, for every pair of such consecutive irremovable vertices $\sigma[a]$ and $\sigma[b]$, a directed acyclic graph whose vertices $\sigma[a : b]$ are the (possibly removable) vertices of the path segment and whose edges are collision-free steering connections between these vertices, as shown in Fig. 3b. The best path σ^*_{ab} from vertex $\sigma[a]$ to vertex $\sigma[b]$ is found and all vertices of path segment $\sigma[a:b]$ which are not on path σ^*_{ab} are removed from path σ. The best path is the one with the shortest arc length – using the more time-consuming evaluation of maximum curvature as cost function has shown no improvement in our experiments. The cost-aware short-cutting phase iterates over the whole path until no more vertices can be removed or the maximum number of pruning rounds L has been reached.

During the gradient-based path deformation and cost-aware short-cutting phases, the headings of the SE(2) vertices are updated (in UPDATEANGLES(σ)). This function computes, for every vertex $\sigma[i]$, the heading of the incoming line from predecessor $\sigma[i-1]$ and the heading of the outgoing line to successor $\sigma[i+1]$. The heading of $\sigma[i]$ is set to the average of the headings of $\sigma[i-1]$ and $\sigma[i+1]$ if this change does not cause any collisions in the steering, i.e. $s(\sigma[i-1], \sigma[i])$ and $s(\sigma[i], \sigma[i+1])$ do not collide with obstacles.

Algorithm 1 GRIPS

```
function GRIPS($\sigma$)
    $\sigma \leftarrow$ MOVEANDINSERTVERTICES($\sigma$)
    $\sigma \leftarrow$ PRUNEVERTICES($\sigma$)
    return $\sigma$
```

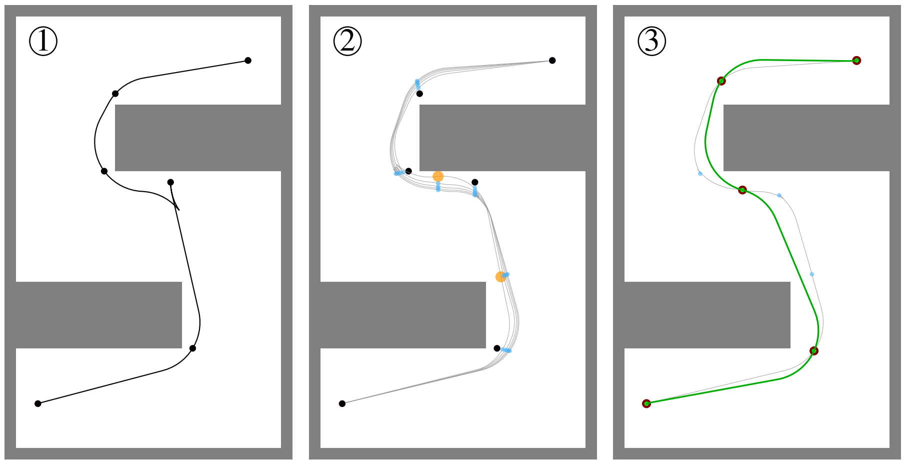
Algorithm 2 Gradient-Based Path Deformation Phase

```
function MOVEANDINSERTVERTICES($\sigma$)
    $\eta \leftarrow \eta_0$
    for $k = 1 \dots K$ do
        ▷ Gradient descent on obstacle distance field
        for $i = 1 \dots |\sigma|-2$ do
            $\sigma[i] \leftarrow \sigma[i] - \eta \frac{\nabla D[\sigma[i]]}{D[\sigma[i]]}$
        $\eta \leftarrow \gamma\eta$
        $\sigma \leftarrow$ UPDATEANGLES($\sigma$)
        ▷ Insert vertices at local minima in distance field
        $\sigma' \leftarrow \emptyset$
        for $i \in 0 \dots |\sigma|-2$ do
            $\sigma' \leftarrow$ APPEND($\sigma', \sigma[i]$)
            for $q \in s(\sigma[i], \sigma[i+1])$ do
                if $D[q]$ is local minimum
                        $\wedge ||q-\sigma[i]||_2 \geq d_{\min}$
                        $\wedge ||q-\sigma[i+1]||_2 \geq d_{\min}$ then
                    $\sigma' \leftarrow$ APPEND($\sigma', q$)
        $\sigma' \leftarrow$ APPEND($\sigma', \sigma[|\sigma|-1]$)
        $\sigma \leftarrow \sigma'$
        $\sigma \leftarrow$ UPDATEANGLES($\sigma$)
    return $\sigma$
```

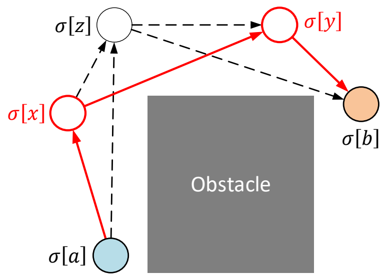
IV. EXPERIMENTS

In this section, we introduce a set of experiments to evaluate GRIPS[1] in terms of efficiency and quality of the smoothing process. We compare GRIPS to four other baselines: a *B-spline interpolation* technique similar to [23], *SimplifyMax*, a *short-cutting* method [10] and *Anytime Path Shortening* (AnytimePS) [12]. The SimplifyMax method

[1]Our implementation is available at https://github.com/eric-heiden/grips.



(a)



(b)

Fig. 3: **(a)** Post-smoothing of a path generated by Theta* with Reeds-Shepp steering: ① Initial path with SE(2)-vertices (black dots), ② gradient-based path deformation phase (Alg. 2) (gray lines with light-blue vertices indicate the path deformed during gradient descent rounds, yellow vertices are inserted during gradient descent), ③ cost-aware short-cutting phase (Alg. 3) (dark-red vertices are irremovable, the green line is the final path). **(b)** Path segment $\{\sigma[a], \sigma[x], \sigma[z], \sigma[y], \sigma[b]\}$ avoiding an obstacle. Vertices $\sigma[a]$ and $\sigma[b]$ have been determined to be irremovable s.t. they form a directed acyclic graph with vertices $\sigma[a], \sigma[x], \sigma[z], \sigma[y]$ and $\sigma[b]$, whose edges represent collision-free steering connections between the vertices (dashed and solid lines). For simplicity, the steer function is assumed to be a linear interpolator in this example. The red solid path $\{\sigma[a], \sigma[x], \sigma[y], \sigma[b]\}$ replaces the path segment $\sigma[a\!:\!b]$ in Alg. 3.

Algorithm 3 Cost-Aware Short-Cutting Phase

```
function PRUNEVERTICES($\sigma$)
    for $l = 1 \ldots L$ do
        $N \leftarrow |\sigma|$
        $\mathfrak{u} \leftarrow \{i \in \{1 \ldots |\sigma|-2\} \mid$
                COLLIDES$(s(\sigma[i-1], \sigma[i+1]))\}$
        for subsequent indices $a, b \in \mathfrak{u}$ do
            ▷ Build directed acyclic graph $G = (V, E)$:
            $V \leftarrow \{\sigma[j] \mid j \in \{a \ldots b\}\}$
            $E \leftarrow \{(\sigma[i], \sigma[j]) \mid i, j \in \{a \ldots b\} \wedge i < j \wedge$
                    $\neg$COLLIDES$(s(\sigma[i], \sigma[j]))\}$
            $\sigma[a\!:\!b] \leftarrow$ BESTPATH$(G, \sigma[a], \sigma[b])$
        $\sigma \leftarrow$ UPDATEANGLES$(\sigma)$
        if $|\sigma| = N$ then
            break
    return $\sigma$
```

first collapses close vertices to remove redundancies, then attempts to short-cut the remaining path and, in the last step, to connect adjacent vertices with a B-spline interpolation. We compare all algorithms by applying them to paths generated by various motion planners (namely RRT, RRT*, SORRT*, RRT#, PRM and Theta*). All motion planners use the same collision checker (which ensures that the distance to the nearest obstacle is at least $0.05\,\mathrm{m}$) and uniform sampling with 5% goal biasing. As soon as any path has been found by the motion planner, the post-smoothing algorithm is executed on this initial solution. We adopt a kinodynamic version of the Theta* planner where the EXTEND-method is realized by the steer function instead of a straight-line connection. We use the implementations of the sampling-based motion planners and post-smoothing algorithms from the Open Motion Planning Library (OMPL) [25].

The behavior of GRIPS can be configured by the parameters η_0 (the initial gradient-descent step size), γ (the discount factor), K (the number of gradient-descent steps), d_{min} (the minimum vertex distance) and L (the maximum number of pruning rounds). Throughout all our experiments, we set $d_{min} = 3\,\mathrm{m}$ as shorter vertex distances would cause numerical instabilities in some steer functions and $L = 100$, although the cost-aware short-cutting phase typically completed after at most five pruning rounds in our experiments. As shown in Tab. II, we analyze the influence of $\eta_0 \in \{0.25, 0.5, 0.75\}$, $\gamma \in \{0.5, 0.8, 0.95\}$ and $K \in \{3, 5, 10\}$ on the path length l, the maximum curvature κ_{max}, the number of vertices N on the path, the obstacle D and the computation time T. Larger values of η_0, γ and K tend to increase the path length and mean obstacle distance since the gradient-based path-deformation rounds move vertices away from obstacles.

Resulting from this empirical validation, we use $\eta_0 = 0.5$, $\gamma = 0.8$ and $K = 5$ throughout this paper as these parameters provide a good trade-off between a low number of gradient-based path deformation steps (and thus low computation time) and a high path quality in terms of path length and maximum curvature. Furthermore, by relying on a single parameter configuration, we can show that GRIPS robustly yields high performance for different motion planners with several steer functions in different environments.

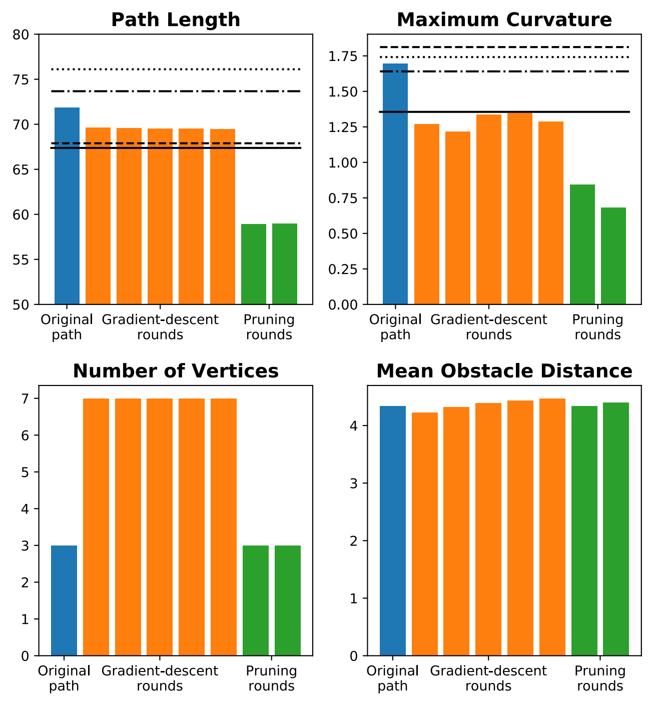


Fig. 4: Evolution of path length (top left), maximum curvature (top right), number of vertices on the path (bottom left) and mean obstacle distance (bottom right) of a path generated by RRT# with POSQ steering in a 50×50 corridor environment before post-smoothing (blue bar), during the gradient-descent rounds (orange bars) and during the pruning rounds (green bars). For comparison, the quality metrics for B-Spline (—), SimplifyMax (- -), Shortcut (—·) and AnytimePS (· · ·) are shown as well.

A. *Environments and Metrics*

As shown in Fig. 5, we use indoor-like ("corridor") environments of sizes $50\,\mathrm{m} \times 50\,\mathrm{m}$ and $150\,\mathrm{m} \times 150\,\mathrm{m}$, and outdoor-like ("random") environments of size $50\,\mathrm{m} \times 50\,\mathrm{m}$. Corridor environments feature wider rectangular areas resembling rooms and narrower bottlenecks as small as $6\,\mathrm{m}$ in width representing hallways (Figs. 1, 5a and 5b), while the occupancy of each grid cell in random environments (Fig. 5c) independently follows a uniform distribution such that 5% of grid cells (excluding the border) are occupied. For each environment, the start and goal vertices were sampled randomly among pairs of far-apart but connected points.

We evaluate the performance of the post-smoothing algorithms based on several performance metrics, namely the computation time T of planning and post-smoothing combined, the total number of colliding paths C and quality metrics for the resulting paths. The quality metrics are the path length l, the maximum curvature κ_{max} and the obstacle distance D. The first two quality metrics directly affect the possible traversal time of a path: it takes more time to follow longer paths with higher curvature (yielding sharp turns) given that robots have physical limits on their accelerations and velocities. For each pair of motion planner and post-smoothing algorithm, we perform runs on 20 different environments and compute the mean and standard deviation of the performance metrics. We approximate the curvature $\kappa_t = 1/r_t$ along a path σ by considering three consecutive points $\sigma'[t-1], \sigma'[t], \sigma'[t+1]$ on the corresponding sequence of vertices σ' returned by the steer function and computing the radius r_t of the circle which passes through them.

B. *Nonholonomic Systems*

We investigate how GRIPS and the baselines perform on several nonholonomic systems. Each motion planner is executed on 20 environments for each of the following steer functions: POSQ [18], Reeds-Shepp [19], G^1 clothoids [20] and Dubins [21].

Reeds-Shepp and Dubins generate shortest paths for car-like kinematic systems considering a constant tangential velocity: a Dubins car is allowed to move only forward, while a Reeds-Shepp car can also move backward. Since Dubins and Reeds-Shepp present curvature profiles with discontinuities, we also compare the steer functions with G^1 clothoids: G^1 clothoids interpolate two given points in a plane with assigned unit tangent vectors, generating a curve with linearly time-varying curvature. Reeds-Shepp, Dubins and G^1 clothoids generate trajectories for the following kinematic system (with respect to arc length l):

$$\begin{pmatrix} x' \\ y' \\ \theta' \\ \kappa' \end{pmatrix} = \begin{pmatrix} \cos\theta \\ \sin\theta \\ \kappa \\ 0 \end{pmatrix} d + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} d\kappa, \tag{1}$$

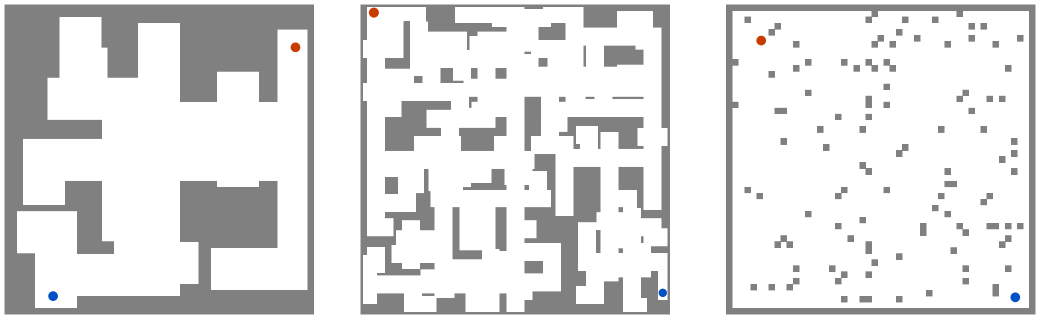
where the position of the midpoint of the rear axle is denoted by (x, y), the heading of the car by θ and the curvature by κ. The driving direction d and the change of curvature $d\kappa$ (or sharpness) are the inputs to the system. The operator $(.)'$ denotes the derivative with respect to the arc length l. POSQ generates trajectories for wheeled mobile robots modeled as differential drive robots with the following kinematic equations in a polar-coordinate representation (time-dependency is omitted):

$$\begin{pmatrix} \dot\rho \\ \dot\alpha \\ \dot\phi \end{pmatrix} = \begin{pmatrix} \cos\alpha \\ \frac{\sin\alpha}{\rho} \\ 0 \end{pmatrix} v + \begin{pmatrix} 0 \\ -1 \\ -1 \end{pmatrix} \omega, \tag{2}$$

where ρ is the Euclidean distance between the Cartesian coordinates of the robot state (x, y, θ) and the goal state in operation space, ϕ is the angle between the x-axis of the robot reference frame $\{X_r\}$ and the x-axis of the goal state frame $\{X_g\}$, α is the angle between the y-axis of the robot reference frame and the vector connecting the robot with the goal position, v is the translational robot velocity and ω is the angular robot velocity. We denote the time derivative of a as $\dot a$.

V. Results

Table I summarizes the performance metrics of the post-smoothing methods averaged over the six motion planners with four different steer functions in 20 environments per



(a) 50×50 corridor (b) 150×150 corridor (c) 50×50 random

Fig. 5: Examples of the three different environments types with start states (red) and goal states (blue).

Configuration	Metric	Before	GRIPS	B-Spline	SimplifyMax	Shortcut	AnytimePS
POSQ 50×50 corridor	l $[m]$	69.15 ± 22.36	**57.03 ± 4.14**	63.89 ± 13.09	60.28 ± 7.05	66.17 ± 14.07	67.31 ± 18.87
	κ_{max}	1.80 ± 0.90	**0.26 ± 0.23**	1.40 ± 0.67	1.47 ± 0.74	1.70 ± 0.85	1.80 ± 0.96
	T $[s]$	**1.50 ± 4.70**	**1.51 ± 4.70**	**1.51 ± 4.70**	**1.51 ± 4.70**	**1.50 ± 4.70**	**1.50 ± 4.70**
	C	24	**10**	30	32	32	24
POSQ 150×150 corridor	l $[m]$	265.89 ± 75.93	**206.85 ± 14.76**	234.56 ± 48.76	208.48 ± 11.54	252.41 ± 59.32	265.89 ± 75.93
	κ_{max}	2.35 ± 1.22	**0.61 ± 0.92**	1.37 ± 0.41	1.24 ± 0.47	2.33 ± 1.30	2.32 ± 1.25
	T $[s]$	**1.47 ± 2.61**	1.49 ± 2.61	**1.48 ± 2.61**	**1.48 ± 2.61**	**1.47 ± 2.61**	**1.47 ± 2.61**
	C	37	**21**	49	54	46	38
POSQ 50×50 random	l $[m]$	69.58 ± 9.20	**64.21 ± 4.30**	66.47 ± 6.39	65.49 ± 5.42	68.22 ± 7.30	69.58 ± 9.20
	κ_{max}	1.65 ± 0.85	**0.68 ± 0.24**	1.03 ± 0.62	0.93 ± 0.55	1.41 ± 0.88	1.62 ± 0.86
	T $[s]$	**3.73 ± 6.41**	3.74 ± 6.41	3.74 ± 6.41	3.76 ± 6.41	**3.73 ± 6.41**	**3.73 ± 6.41**
	C	36	**31**	65	67	50	39
Reeds-Shepp 50×50 corridor	l $[m]$	69.10 ± 14.39	**56.78 ± 4.96**	67.24 ± 12.51	58.68 ± 5.95	65.23 ± 10.21	63.95 ± 8.51
	κ_{max}	1.80 ± 1.83	**0.12 ± 0.14**	1.98 ± 2.11	0.75 ± 1.13	1.31 ± 1.60	1.39 ± 1.60
	T $[s]$	**0.05 ± 0.11**	**0.05 ± 0.11**	**0.05 ± 0.11**	**0.05 ± 0.11**	**0.05 ± 0.11**	**0.05 ± 0.11**
	C	7	**0**	9	5	6	5
Reeds-Shepp 150×150 corridor	l $[m]$	233.36 ± 26.62	**209.17 ± 11.47**	223.85 ± 20.38	215.00 ± 13.71	223.92 ± 19.15	221.81 ± 16.95
	κ_{max}	1.05 ± 1.04	**0.40 ± 0.70**	0.99 ± 1.34	1.03 ± 1.04	1.03 ± 0.91	0.83 ± 0.88
	T $[s]$	**0.08 ± 0.15**	**0.09 ± 0.15**	**0.08 ± 0.15**	**0.08 ± 0.15**	**0.08 ± 0.15**	**0.08 ± 0.15**
	C	47	**12**	43	56	41	35
Reeds-Shepp 50×50 random	l $[m]$	74.95 ± 10.79	**63.89 ± 3.88**	72.74 ± 9.38	67.53 ± 6.18	71.63 ± 7.87	70.08 ± 6.94
	κ_{max}	2.54 ± 2.01	**0.63 ± 1.21**	2.37 ± 2.30	1.88 ± 1.69	2.53 ± 1.88	1.51 ± 1.86
	T $[s]$	**0.11 ± 0.22**	**0.11 ± 0.22**	**0.11 ± 0.22**	**0.11 ± 0.22**	**0.11 ± 0.22**	**0.11 ± 0.22**
	C	39	**28**	58	57	43	46
Clothoid 50×50 corridor	l $[m]$	74.42 ± 18.16	**59.19 ± 4.08**	73.35 ± 17.15	63.82 ± 5.81	69.32 ± 10.02	67.10 ± 7.54
	κ_{max}	0.64 ± 0.59	0.25 ± 0.43	0.72 ± 0.63	**0.21 ± 0.21**	0.51 ± 0.51	0.46 ± 0.40
	T $[s]$	**0.86 ± 2.45**	1.18 ± 2.66	1.08 ± 2.51	0.89 ± 2.45	**0.87 ± 2.45**	**0.86 ± 2.45**
	C	31	**8**	30	20	33	24
Clothoid 150×150 corridor	l $[m]$	227.92 ± 16.51	**202.23 ± 6.80**	223.92 ± 14.13	208.27 ± 4.90	222.32 ± 16.26	222.74 ± 16.46
	κ_{max}	0.45 ± 0.43	**0.17 ± 0.17**	0.29 ± 0.08	0.34 ± 0.49	0.20 ± 0.13	0.31 ± 0.18
	T $[s]$	**0.41 ± 0.23**	6.62 ± 10.03	0.60 ± 0.28	0.48 ± 0.17	**0.41 ± 0.23**	**0.41 ± 0.23**
	C	108	**66**	99	71	93	82
Clothoid 50×50 random	l $[m]$	69.21 ± 6.23	64.02 ± 3.46	68.64 ± 5.90	**63.10 ± 1.82**	67.79 ± 5.49	67.52 ± 6.13
	κ_{max}	0.30 ± 0.16	0.23 ± 0.17	0.38 ± 0.24	**0.12 ± 0.08**	0.26 ± 0.14	0.35 ± 0.20
	T $[s]$	**2.76 ± 4.74**	4.89 ± 9.47	2.97 ± 4.73	2.79 ± 4.74	**2.77 ± 4.74**	**2.76 ± 4.74**
	C	104	**48**	95	66	96	82
Dubins 50×50 corridor	l $[m]$	68.26 ± 15.70	**58.85 ± 5.72**	79.28 ± 32.75	61.59 ± 6.57	64.05 ± 9.10	63.31 ± 7.91
	κ_{max}	0.66 ± 0.04	**0.36 ± 0.30**	0.66 ± 0.03	0.64 ± 0.06	0.66 ± 0.03	0.65 ± 0.04
	T $[s]$	**0.05 ± 0.11**	**0.06 ± 0.11**	**0.05 ± 0.11**	**0.05 ± 0.11**	**0.05 ± 0.11**	**0.05 ± 0.11**
	C	29	**10**	63	27	33	33
Dubins 150×150 corridor	l $[m]$	223.97 ± 29.12	**205.91 ± 10.98**	220.05 ± 25.17	215.05 ± 15.89	213.06 ± 14.44	214.48 ± 14.56
	κ_{max}	0.61 ± 0.14	0.54 ± 0.21	0.65 ± 0.06	**0.53 ± 0.21**	0.56 ± 0.18	0.55 ± 0.19
	T $[s]$	**0.10 ± 0.14**	0.12 ± 0.14	**0.10 ± 0.14**	**0.10 ± 0.14**	**0.10 ± 0.14**	**0.10 ± 0.14**
	C	58	**21**	71	51	61	53
Dubins 50×50 random	l $[m]$	68.93 ± 6.65	**63.74 ± 2.14**	69.56 ± 7.12	65.10 ± 3.28	68.09 ± 5.54	66.33 ± 4.30
	κ_{max}	0.66 ± 0.01	**0.60 ± 0.16**	0.67 ± 0.00	0.64 ± 0.09	0.67 ± 0.00	0.66 ± 0.01
	T $[s]$	**0.01 ± 0.02**	**0.01 ± 0.02**	**0.01 ± 0.02**	**0.01 ± 0.02**	**0.01 ± 0.02**	**0.01 ± 0.02**
	C	54	**39**	91	45	66	62

TABLE I: Post-smoothing results averaged over the six motion planners (Sec. IV) with different steer functions in 20 randomly generated maps per environment type. Metrics are shown in the format *mean* $\pm$ *std deviation*.

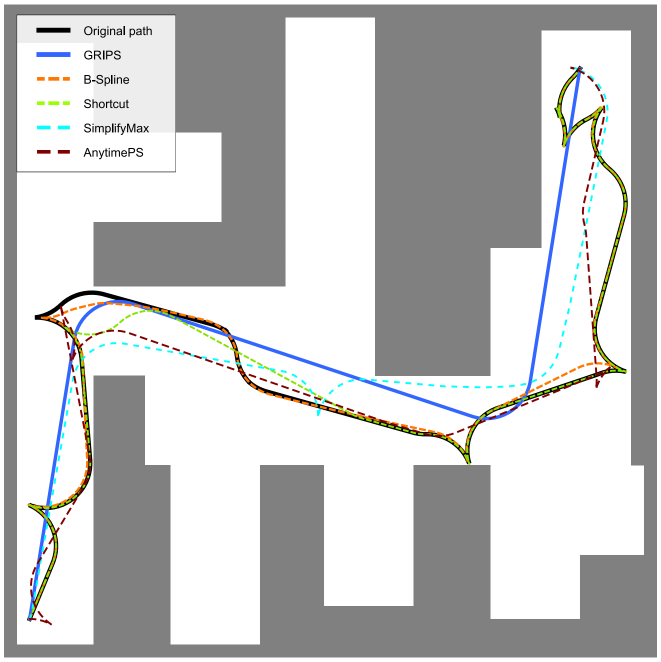


Fig. 6: Visual comparison of post-smoothing results for a path generated by RRT with Reeds-Shepp steering (black) in a 50×50 corridor environment.

GRIPS Parameters			**Quality Metrics**				
η_0	γ	K	l $[m]$	κ_{max}	N	D $[m]$	T $[ms]$
0.25	0.5	3	79.62	0.86	5	2.32 ± 1.38	6
0.25	0.5	5	79.70	0.93	5	2.33 ± 1.38	9
0.25	0.5	10	79.73	0.931	5	2.33 ± 1.38	15
0.25	0.8	3	79.93	0.935	5	2.35 ± 1.37	7
0.25	0.8	5	80.30	1.21	5	2.39 ± 1.36	9
0.25	0.8	10	80.70	1.31	5	2.43 ± 1.34	15
0.25	0.95	3	80.12	1.2	5	2.37 ± 1.36	6
0.25	0.95	5	80.75	1.31	5	2.44 ± 1.34	9
0.25	0.95	10	81.84	0.598	5	2.53 ± 1.31	15
0.5	0.5	3	80.5	1.22	5	2.41 ± 1.35	6
0.5	0.5	5	80.64	1.32	5	2.43 ± 1.34	9
0.5	0.5	10	80.68	1.31	5	2.43 ± 1.34	15
0.5	0.8	3	81.02	1.21	5	2.46 ± 1.33	7
0.5	0.8	5	81.61	0.679	5	2.51 ± 1.31	9
0.5	0.8	10	82.18	0.983	5	2.55 ± 1.30	15
0.5	0.95	3	81.32	1	5	2.49 ± 1.32	6
0.5	0.95	5	82.27	1.26	5	2.56 ± 1.29	9
0.5	0.95	10	83.51	0.984	6	2.63 ± 1.29	16
0.75	0.5	3	81.34	0.984	5	2.49 ± 1.32	7
0.75	0.5	5	81.51	0.94	5	2.50 ± 1.32	9
0.75	0.5	10	81.56	0.931	5	2.50 ± 1.31	16
0.75	0.8	3	81.99	0.979	5	2.54 ± 1.30	6
0.75	0.8	5	82.64	1.28	5	2.59 ± 1.29	9
0.75	0.8	10	83.2	1.27	5	2.64 ± 1.29	16
0.75	0.95	3	82.35	1.26	5	2.57 ± 1.29	7
0.75	0.95	5	83.31	1.26	5	2.64 ± 1.29	9
0.75	0.95	10	84.39	1.29	6	2.84 ± 1.17	17

TABLE II: Quality metrics for GRIPS applied with various parameter configurations on a path generated by Theta* with POSQ steering in the environment depicted in Fig. 6. The obstacle distance D is shown in the format *mean* $\pm$ *std deviation*.

environment type. On average, GRIPS outperforms the baselines in terms of path length, maximum curvature and number of collisions, while its computation time is on par with the baselines. In many cases, GRIPS is able to correct paths that collide with obstacles via the gradient-based path-deformation phase. The curvatures for Reeds-Shepp and Dubins steering do not vary significantly because of the fixed turning radii ($3.5\,\mathrm{m}$ and $1.5\,\mathrm{m}$, respectively) these steer functions use.

Fig. 6 demonstrates that paths found by sampling-based motion planners (such as RRT in this case) with Reeds-Shepp steering can contain vertices at which the car would need to move backward in order to navigate narrow passages of the environment. All baselines retained these motions in the example, resulting in larger traversal times of the paths. GRIPS, on the other hand, was able to find a path with a minimum number of turns. This example shows the potential of combining fast motion planners with efficient post-smoothing methods to obtain high-quality paths within short computation times.

The scatter plots in Fig. 7 visualize the performance metrics for different combinations of motion planners and post-smoothing methods. Although these plots show results generated with POSQ, similar trends were observed for the other steer functions as well. Independent of the motion planner which generated the initial path, GRIPS yielded the highest quality on these metrics in most cases, as shown in Tab. I.

Fig. 4 shows the evolution of the quality metrics over the phases of GRIPS: before post-smoothing (blue bar), during the gradient-descent rounds (orange bars) and during the subsequent pruning rounds (green bars). As visualized by the black lines, GRIPS outperforms the baselines in terms of the path length and maximum curvature. Similar trends were observed for the other combinations of motion planners, steer functions and environment types. The gradient-descent rounds tend to decrease the path length and the maximum curvature but tend to increase the number of vertices and the mean obstacle distance. Thus, the gradient-based path deformation phase provides the subsequent cost-aware short-cutting phase with more opportunity to shorten the path as more vertices are distributed in areas further away from obstacles. The pruning rounds continue to decrease the path length and the maximum curvature but decrease also the number of vertices and the mean obstacle distance, until they reach or drop below the corresponding values of the initial path.

VI. Conclusion

In this paper, we introduced a new methodology for path smoothing, called Gradient-Informed Path Smoothing (GRIPS). Unlike previous techniques, GRIPS combines the advantages of gradient-based optimization and short-cutting in two phases. First, it inserts and moves vertices away from obstacles according to the gradient of the obstacle distance field. Then, it uses cost-aware short-cutting to remove unnecessary vertices. Our experiments indicate that, on average, GRIPS generates shorter (smaller arc length) and smoother

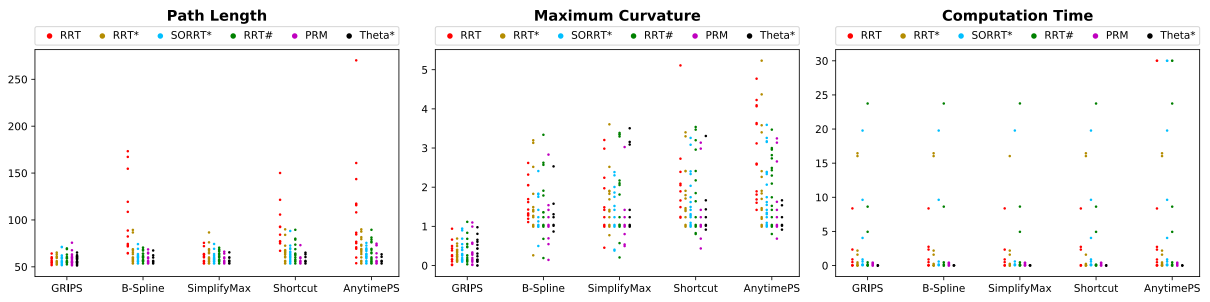


Fig. 7: Post-smoothing quality metrics for trajectories generated by various motion planners with POSQ steering (cf. Table I) in a 50×50 corridor environment (Fig. 5a).

(smaller maximum curvature) paths than several state-of-the-art path-smoothing techniques, while finding feasible paths in more cases and being as fast as them.

In the future, we want to apply GRIPS to high-dimensional systems (for example, manipulators) and prove its completeness. We also want to use GRIPS as part of any-angle path planners, such as Theta*, to generate kinodynamic paths by interleaving GRIPS with heuristic search.

VII. Acknowledgements

The research at the University of Southern California was supported by the National Science Foundation (NSF) under grant numbers 1724392, 1409987 and 1319966. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the sponsoring organizations, agencies or the U.S. government.

Luigi Palmieri's work has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 732737 (ILIAD).

References

[1] A. D. Dragan, K. C. Lee, and S. S. Srinivasa, "Legibility and predictability of robot motion," in *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2013, pp. 301–308.

[2] R. Triebel, K. Arras, R. Alami, L. Beyer, S. Breuers, R. Chatila, M. Chetouani, D. Cremers, V. Evers, M. Fiore, *et al.*, "Spencer: A socially aware service robot for passenger guidance and help in busy airports," in *Field and Service Robotics (FSR)*, vol. 14, 2015, pp. 607–622.

[3] S. Gulati, C. Jhurani, B. Kuipers, and R. Longoria, "A framework for planning comfortable and customizable motion of an assistive mobile robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2009, pp. 367–393.

[4] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research (IJRR)*, vol. 30, no. 7, pp. 846–894, 2011.

[5] O. Arslan and P. Tsiotras, "Use of relaxation methods in sampling-based algorithms for optimal motion planning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013, pp. 2421–2428.

[6] L. Janson, E. Schmerling, A. Clark, and M. Pavone, "Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions," *International Journal of Robotics Research (IJRR)*, vol. 34, no. 7, pp. 883–921, 2015.

[7] J. D. Gammell, "Informed anytime search for continuous planning problems," Ph.D. dissertation, University of Toronto, 2017.

[8] A. Nash, K. Daniel, S. Koenig, and A. Felner, "Theta*: Any-angle path planning on grids," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2007, p. 11771183.

[9] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[10] D. Hsu, J.-C. Latombe, and S. Sorkin, "Placing a robot manipulator amid obstacles for optimized execution," in *Proceedings of the International Symposium on Assembly and Task Planning (ISATP)*, 1999, pp. 280–285.

[11] R. Geraerts and M. H. Overmars, "Creating high-quality paths for motion planning," *International Journal of Robotics Research (IJRR)*, vol. 26, no. 8, pp. 845–863, 2007.

[12] R. Luna, I. A. Şucan, M. Moll, and L. E. Kavraki, "Anytime solution optimization for sampling-based motion planning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013, pp. 5068–5074.

[13] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: Gradient optimization techniques for efficient motion planning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2009, pp. 489–494.

[14] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *International Journal of Robotics Research (IJRR)*, vol. 33, no. 9, pp. 1251–1270, 2014.

[15] S. Quinlan and O. Khatib, "Elastic bands: Connecting path planning and control," in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 2, 1993, pp. 802–807.

[16] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 2, 2000, pp. 995–1001.

[17] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[18] L. Palmieri and K. O. Arras, "A novel RRT extend function for efficient and smooth mobile robot motion planning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014, pp. 205–211.

[19] J. Reeds and L. Shepp, "Optimal paths for a car that goes both forwards and backwards," *Pacific Journal of Mathematics*, vol. 145, no. 2, pp. 367–393, 1990.

[20] E. Bertolazzi and M. Frego, "G1 fitting with clothoids," *Mathematical Methods in the Applied Sciences*, vol. 38, no. 5, 2015.

[21] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *American Journal of Mathematics*, vol. 79, no. 3, pp. 497–516, 1957.

[22] B. Raveh, A. Enosh, and D. Halperin, "A little more, a lot better: Improving path quality by a path-merging algorithm," *IEEE Transactions on Robotics*, vol. 27, no. 2, pp. 365–371, 2011.

[23] K. Yang and S. Sukkarieh, "An analytical continuous-curvature path-smoothing algorithm," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 561–568, 2010.

[24] T. Pavlidis and S. L. Horowitz, "Segmentation of plane curves," *IEEE Transactions on Computers*, vol. 23, no. 8, pp. 860–870, 1974.

[25] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, 2012, http://ompl.kavrakilab.org.