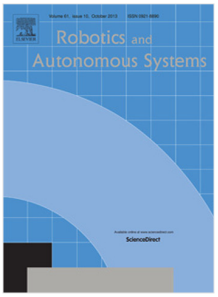
Accepted Manuscript



Integrated online trajectory planning and optimization in distinctive topologies

Christoph Rösmann, Frank Hoffmann, Torsten Bertram

PII: S0921-8890(16)30049-5
DOI: http://dx.doi.org/10.1016/j.robot.2016.11.007
Reference: ROBOT 2743

To appear in: *Robotics and Autonomous Systems*

Received date : 31 January 2016
Accepted date : 10 November 2016

Please cite this article as: C. Rösmann, et al., Integrated online trajectory planning and optimization in distinctive topologies, *Robotics and Autonomous Systems* (2016), http://dx.doi.org/10.1016/j.robot.2016.11.007

Integrated Online Trajectory Planning and Optimization in Distinctive Topologies

Christoph Rösmann[a,*], Frank Hoffmann[a], Torsten Bertram[a]

[a] *Institute of Control Theory and Systems Engineering*
Technical University of Dortmund
44227 Dortmund, Germany

Abstract

This paper presents a novel integrated approach for efficient optimization based online trajectory planning of topologically distinctive mobile robot trajectories. Online trajectory optimization deforms an initial coarse path generated by a global planner by minimizing objectives such as path length, transition time or control effort. Kinodynamic motion properties of mobile robots and clearance from obstacles impose additional equality and inequality constraints on the trajectory optimization. Local planners account for efficiency by restricting the search space to locally optimal solutions only. However, the objective function is usually non-convex as the presence of obstacles generates multiple distinctive local optima.

The proposed method maintains and simultaneously optimizes a subset of admissible candidate trajectories of distinctive topologies and thus seeking the overall best candidate among the set of alternative local solutions. Time-optimal trajectories for differential-drive and carlike robots are obtained efficiently by adopting the Timed-Elastic-Band approach for the underlying trajectory optimization problem. The investigation of various example scenarios and a comparative analysis with conventional local planners confirm the advantages of integrated exploration, maintenance and optimization of topologically distinctive trajectories.

* Corresponding author
Email address: christoph.roesmann@tu-dortmund.de (Christoph Rösmann)

Keywords: Online trajectory optimization, mobile robot motion planning, distinctive topologies, homology classes

1. Introduction

In the context of service robotics and autonomous transportation systems, mobile robots are required to navigate in highly dynamic environments while accomplishing complex tasks. On this occasion, one of the fundamental challenges in mobile robotics is concerned with the development of universally applicable motion planning strategies. Online planning is preferred over offline approaches since they immediately respond to changes in the environment or perturbations of the robot motion at runtime. The well known *elastic band* approach [1] locally deforms a path online. Predefined internal forces contract the path while external forces maintain a separation from obstacles. An alternative established path planning approach based on an optimization technique is presented in [2]. However, conventional path planning does not explicitly incorporate temporal and (kino-)dynamic aspects of motion, therefore ignoring constraints imposed by kinematic or dynamic motion models with bounded velocities and accelerations.

Kurniawati et al. extend the *elastic band* approach to the online deformation of trajectories rather than paths [3]. The approach consists of two stages that at first repel discrete trajectory points from obstacles and secondly enforce connectedness w.r.t. a dynamic motion model. Delsart et al. combine both stages into a single operation [4].

Trajectory optimization usually attempts to minimize either control effort, control error or the transition time between start and goal pose. However, the computational burden required to find an optimal solution is high, which limits its direct online integration with feedback control. Due to this observation many researchers are focusing on obtaining solutions or even approximations of the underlying trajectory optimization problem efficiently. The dynamic window approach (DWA) constitutes a widely applied method for mobile robot

navigation [5]. Simulated trajectories are sampled repeatedly from a velocity search space restricted by a set of feasible velocities. A cost function evaluates each candidate trajectory w.r.t. the remaining distance to the goal, the forward velocity and separation from obstacles. The lowest-cost solution is selected for controlling the robot. The approach accounts for efficiency by restricting the feasible set of valid trajectories to a subset of sampled candidates with segments of constant velocity leading to merely suboptimal trajectories. Lau et al. [6] and Sprunk et al. [7] optimize trajectories represented by splines continuously according to kinodynamic constraints of the robot. An online planning algorithm that relies on a covariant gradient descent method is presented in [8]. In a previous work the authors present a further extension to the *elastic band* called *Timed-Elastic-Band* (TEB) approach [9, 10]. The TEB efficiently optimizes the robot trajectory w.r.t. (kino-)dynamic constraints and non-holonomic kinematics while explicitly incorporating temporal information in order to reach the goal pose in minimal time. The approach accounts for efficiency by exploiting the sparsity structure of the underlying problem formulation. It has been generalized to an efficient time-optimal model predictive control approach for dynamic systems [11].

In practice, due to limited computational resources online optimization is usually performed using local optimization techniques for which the discovery of the global optimal trajectory is not guaranteed. Especially in mobile robot navigation local minima often emerge due to the presence of obstacles. Kalakrishnan et al. apply a stochastic descent method to partially overcome these limitations [12]. However, the approach requires extensive sampling of trajectories in order to estimate the true gradient w.r.t. the global optimum. A two stage local optimization approach that generates a set of alternative, topologically distinctive trajectories is presented by Kuderer et al. [13]. The proposed method extracts multiple candidate trajectories from a modified *Voronoi diagram* that often coincide with the local minima of the optimization problem. Paths that belong to the same *equivalence class* are grouped by an equivalence relation based on the winding number at each obstacle. The paper at hand

pursues a related approach for filtering distinctive candidate trajectories in the context of online trajectory planning. It mainly differs from [13] w.r.t. sampling strategy, equivalence relation and the trajectory optimization technique.

The idea of exploring topologically distinctive paths and trajectories is not novel. However, past approaches mainly focus on global offline path and trajectory planning. Probabilistic roadmap (PRM) methods that operate with different equivalence relations are presented in [14] and [15]. The proposed algorithms are in principle able to identify complicated paths in large, complex environments but rely upon an algorithmic rather than closed form computation of the equivalence relations. Our approach employs a computationally more efficient sampling strategy and equivalence relation. It is intended to operate in a local subregion of the environment and is thus suited for online trajectory optimization. Obviously, this does not replace the need for global planning in large environments. Knepper et al. propose a local planner based on path sampling and the *Hausdorff* metric as equivalence relation [16]. The planner performs a discrete path selection rather than deforming continuous trajectories. An equivalence relation originating from the field of complex analysis is presented by Bhattacharya et al. [17]. The closed form solution motivates its application within our approach for trajectory filtering. Pokorny et al. presents a graph free sampling based approach based on filtrations of simplicial complexes [18].

This contribution presents an integrated online trajectory planning approach that combines the exploration and simultaneous optimization of multiple admissible topologically distinctive trajectories during runtime. As an extended version of [19] the online exploration strategy is fully integrated with the TEB approach providing the underlying trajectory optimization. Hence the TEB approach is entirely reformulated w.r.t. its original description [9] and further extended to a more generic obstacle representation, supporting forward and backward robot motion and to accomplish navigation tasks for carlike robots beyond the differential-drive robots considered in the original proposal. The combined and integrated approach preserves the locally optimal trajectory of each equivalence class to facilitate warm starting from previous solutions. Can-

didate trajectories are sampled repeatedly and evolve not only over space but also time which significantly reduces the number of samples. The sample based approach is compared with the systematic generation of waypoints from Voronoi diagrams regarding completeness of candidate trajectories and computational efficiency. The presented approach is available as open-source C++ code and integrated into ROS [20].

The paper is organized as follows: Section 2 formulates the local TEB optimization approach followed by the exploration of trajectories in distinctive topologies in section 3. The overall integration of exploration, planning and maintenance of trajectories is described in section 4. Section 5 demonstrates and evaluates results obtained from realistic simulations with a differential drive and a carlike mobile robot. The integrated planning approach is compared with the original locally operating TEB and the DWA as mentioned above. Finally, section 6 summarizes the results and provides an outlook on future work.

2. Timed-Elastic-Band

This section describes the TEB approach which performs the actual trajectory optimization in the overall planning task. The approach is entirely reformulated and extended w.r.t the formulation originally presented in [9].

2.1. Definition and Representation

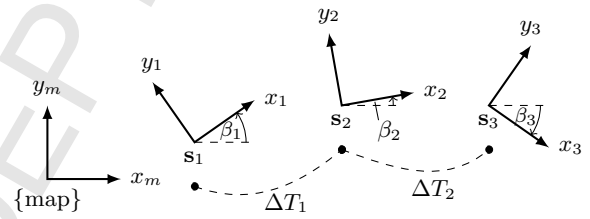


Figure 1: TEB trajectory representation with $n = 3$ poses

Let $\mathbf{s}_k = [x_k, y_k, \beta_k]^\intercal \in \mathbb{R}^2 \times S^1$ denote a robot pose at a discrete point in time k. $x_k \in \mathbb{R}$ and $y_k \in \mathbb{R}$ represent the planar position and $\beta_k \in S^1$

the orientation of the robot. A discretized trajectory is described in terms of a sequence $\mathcal{S} = \{\mathbf{s}_k | k = 1, 2, \ldots, n\}$ of robot poses. The TEB augments the trajectory representation with strictly positive time intervals $\Delta T_k \in \mathbb{R}^+, k = 1, 2, \ldots, n-1$. Each time interval denotes the time that the robot requires to transit from the pose $\mathbf{s}_k$ to its subsequent pose $\mathbf{s}_{k+1}$. An exemplary trajectory with three poses is depicted in Fig. 1. Poses and time intervals are joined into the parameter vector $\mathbf{b} \in \mathcal{B}$:

$$\mathbf{b} = [\mathbf{s}_1, \Delta T_1, \mathbf{s}_2, \Delta T_2, \mathbf{s}_3, \ldots, \Delta T_{n-1}, \mathbf{s}_n]^\intercal \tag{1}$$

2.2. Open-loop optimization problem

The TEB open-loop optimization task is to find controls in order to transit the robot from an initial pose $\mathbf{s}_s$ to a final pose $\mathbf{s}_f$ in minimal time while satisfying kinodynamic constraints and maintaining a safe separation from obstacles. The overall task is formulated as a nonlinear program:

$$V^*(\mathbf{b}) = \min_{\mathbf{b}} \sum_{k=1}^{n-1} \Delta T_k^2 \tag{2}$$

subject to

$$\begin{aligned}
&\mathbf{s}_1 = \mathbf{s}_s, \quad \mathbf{s}_n = \mathbf{s}_f, \quad \Delta T_k > 0 \\
&\mathbf{h}_k(\mathbf{s}_{k+1}, \mathbf{s}_k) = \mathbf{0} \\
&r_k(\mathbf{s}_{k+1}, \mathbf{s}_k) \geq \mathbf{0} \\
&\mathbf{o}_k(\mathbf{s}_k) \geq \mathbf{0} \\
&\boldsymbol{\nu}_k(\mathbf{s}_{k+1}, \mathbf{s}_k, \Delta T_k) \geq \mathbf{0} && (k = 1, 2, \ldots, n-1) \\
&\boldsymbol{\alpha}_k(\mathbf{s}_{k+2}, \mathbf{s}_{k+1}, \mathbf{s}_k, \Delta T_{k+1}, \Delta T_k) \geq \mathbf{0} && (k = 2, 3, \ldots, n-2) \\
&\boldsymbol{\alpha}_1(\mathbf{s}_2, \mathbf{s}_1, \Delta T_1) \geq \mathbf{0}, \quad \boldsymbol{\alpha}_n(\mathbf{s}_n, \mathbf{s}_{n-1}, \Delta T_{n-1}) \geq \mathbf{0}
\end{aligned}$$

According to the definition of the trajectory (1), the total transition time is approximated by $T \approx \sum_{k=1}^{n-1} \Delta T_k$. The minimization of T is ill-posed since individual ΔT_k are unconstrained. Instead, temporal homogeneity is achieved by minimizing the sum of squared ΔT_k which enforces uniform time intervals

$\Delta T_k = \frac{T}{n}$. Initial $\mathbf{s}_1$ and final pose $\mathbf{s}_n$ are constrained by $\mathbf{s}_s$ and $\mathbf{s}_f$ respectively. Kinematic constraints between two consecutive poses $\mathbf{s}_k$ and $\mathbf{s}_{k+1}$ are incorporated by equality constraints $\mathbf{h}_k(\mathbf{s}_{k+1}, \mathbf{s}_k)$. Furthermore, for carlike robots a minimum turning radius must be satisfied which is captured by $r_k(s_{k+1}, s_k)$. The robot's velocity and acceleration are limited by inequalities $\boldsymbol{\nu}_k(\cdot)$ and $\boldsymbol{\alpha}_k(\cdot)$ respectively. The inequality $\mathbf{o}_k(\mathbf{s}_k)$ sustains a minimum separation from obstacles. Each term is described in more detail in the following paragraphs.

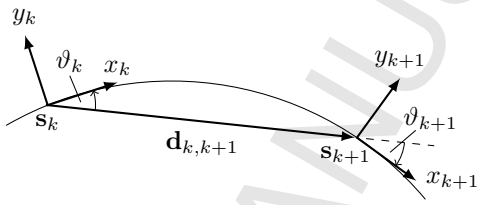


Figure 2: Geometric interpretation of the non-holonomic constraint

Non-holonomic kinematics. The equality constraint $\boldsymbol{\nu}_k(\cdot)$ enforces compliance with the kinematic constraints of the mobile robot. This paper investigates in particular differential drive and carlike robots with only two local degrees of freedom, thus the robot can only execute a smooth path that is composed of linear and arc segments. According to [9], the non-holonomic constraint is defined by a geometric interpretation. Two consecutive poses $\mathbf{s}_k$ and $\mathbf{s}_{k+1}$ are required to be located on a common arc of constant curvature as shown in Fig. 2. The angle ϑ_k between pose $\mathbf{s}_k$ and the direction $\mathbf{d}_{k,k+1} = [x_{k+1} - x_k, y_{k+1} - y_k, 0]^\intercal$ has to be equal to the corresponding angle ϑ_{k+1} at the consecutive pose $\mathbf{s}_{k+1}$:

$$\vartheta_k = \vartheta_{k+1} \tag{3}$$

$$\begin{bmatrix} \cos(\beta_k) \\ \sin(\beta_k) \\ 0 \end{bmatrix} \times \mathbf{d}_{k,k+1} = \mathbf{d}_{k,k+1} \times \begin{bmatrix} \cos(\beta_{k+1}) \\ \sin(\beta_{k+1}) \\ 0 \end{bmatrix} \tag{4}$$

The resulting equality constraint is given by:

$$\mathbf{h}_k(\mathbf{s}_{k+1}, \mathbf{s}_k) = \left(\begin{bmatrix} \cos(\beta_k) \\ \sin(\beta_k) \\ 0 \end{bmatrix} + \begin{bmatrix} \cos(\beta_{k+1}) \\ \sin(\beta_{k+1}) \\ 0 \end{bmatrix} \right) \times \mathbf{d}_{k,k+1} \tag{5}$$

Equation (5) is suitable for differential drive robots that are able to rotate in place. For carlike robots and Ackermann drives respectively, the robot motion is further restricted by a minimal turning radius between consecutive poses. The absolute turning radius $\tilde{r}$ is given by:

$$\tilde{r}(\mathbf{s}_{k+1}, \mathbf{s}_k) = \left| \frac{v_k}{\omega_k} \right| = \frac{\|\mathbf{d}_{k,k+1}\|}{|\beta_{k+1} - \beta_k|} \tag{6}$$

In order to satisfy a minimal turning radius of carlike robots, an inequality constraint with $r_k(s_{k+1}, s_k) = \tilde{r}(\cdot) - r_{min}$ is introduced. r_{min} denotes the lower bound on the turning radius. For differential drive robots it becomes $r_{min} = 0$.

Limited velocity and acceleration. Limiting the velocity is crucial for the success in practical applications. Without any bound, the time-optimal solution according to (2) implies $\Delta T_k \to 0, \forall k$ as $v_k \to \infty$. For the sake of simplicity, translational and rotational velocities at each pose v_k and ω_k (e.g. in the center of the robot) are considered rather than individual wheel velocities. This simplification is sufficient for most practical applications. However, the approach allows the limitation of arbitrary velocities that can be expressed in terms of v_k, ω_k and dedicated robot design parameters. Translational and rotational velocities are approximated using finite differences according to the Euclidean resp. angular distance between two consecutive poses $\mathbf{s}_k$ and $\mathbf{s}_{k+1}$:

$$v_k = \Delta T_k^{-1} \|[x_{k+1} - x_k,\, y_{k+1} - y_k]^\intercal\| \gamma(\mathbf{s}_k, \mathbf{s}_{k+1}) \tag{7}$$

$$\omega_k = \Delta T_k^{-1} (\beta_{k+1} - \beta_k) \tag{8}$$

Note, subtracting and adding angles in domain S^1 requires a normalization in practical implementations. $\gamma(\mathbf{s}_k, \mathbf{s}_{k+1})$ denotes a function that extracts the sign of the translational velocity, whether the robot moves forwards or backwards.

For non-holonomic robots that are restricted to transitions along linear and arc segments respectively, the sign of the projection of orientation vector $\mathbf{q}_k = [\cos\beta_k, \sin\beta_k, 0]^\intercal$ onto the distance vector $\mathbf{d}_{k,k+1}$ is utilized:

$$\gamma(\mathbf{s}_k, \mathbf{s}_{k+1}) = \text{sign}(\langle \mathbf{q}_k, \mathbf{d}_{k,k+1} \rangle) \tag{9}$$

Hereby, operator $\langle \cdot, \cdot \rangle$ applies the scalar product. Since many common optimization algorithms are not suited for non-smooth functions such as (9), a sigmoidal approximation maps the projection to the interval $[-1, 1]$. Our implementation employs:

$$\gamma(\mathbf{s}_k, \mathbf{s}_{k+1}) \approx \frac{\kappa \langle \mathbf{q}_k, \mathbf{d}_{k,k+1} \rangle}{1 + |\kappa \langle \mathbf{q}_k, \mathbf{d}_{k,k+1} \rangle|} \tag{10}$$

Variable $\kappa \in \mathbb{R}^+$ denotes a scaling factor that changes the slope (e.g. $\kappa = 10^2$). Eq. (9) and (10) result in vanishing and incorrect velocities at $\langle \mathbf{q}_k, \mathbf{d}_{k,k+1} \rangle = 0$ or $\langle \mathbf{q}_k, \mathbf{d}_{k,k+1} \rangle \approx 0$ respectively. Figuratively, this occurs if pose $\mathbf{s}_{k+1}$ is placed orthogonal to pose $\mathbf{s}_k$. However, such configuration is not part of the feasible set given by the non-holonomic constraint as mentioned before and for the special case in which position parts of both poses coincide it is $\mathbf{d}_{k,k+1} = \mathbf{0} \implies v_k = 0$.

Limiting velocities to $\pm v_{max}$ and $\pm \omega_{max}$ is obtained by the inequality constraint $\boldsymbol{\nu}_k(\mathbf{s}_{k+1}, \mathbf{s}_k, \Delta T_k) = [v_{max} - |v_k|,\, \omega_{max} - |\omega_k|]^\intercal$. The equations might be adapted in cases in which the bound on negative velocities should differ, which is neglected here for simplicity, but which is often preferred in practical applications. A similar procedure is applied for limiting translational and rotational accelerations a_k and $\dot{\omega}_k$ respectively. In particular for a_k it is:

$$a_k = \frac{2(v_{k+1} - v_k)}{\Delta T_k + \Delta T_{k+1}} \tag{11}$$

For the sake of clarity, $\mathbf{s}_{k+2}$, $\mathbf{s}_{k+1}$ and $\mathbf{s}_k$ are substituted by their related velocities (7). The limitation of the acceleration is obtained by inequality $\boldsymbol{\alpha}_k(\mathbf{s}_{k+2}, \mathbf{s}_{k+1}, \mathbf{s}_k, \Delta T_{k+1}, \Delta T_k) = [a_{max} - |a_k|,\, \dot{\omega}_{max} - |\dot{\omega}_k|]^\intercal$. Special cases occur at $k = 1$ and $k = n - 1$ for which v_1 and v_{n-1} are substituted by desired start and final velocities (v_s, ω_s) and (v_f, ω_f) respectively.

Obstacle avoidance. The robot is supposed to reach the goal without any collision with obstacles. The approach is suited for a broad range of obstacle repre-

sentations. The underlying optimization method is feasible and computationally efficient under the assumption that the minimal Euclidean distance between the pose $\mathbf{s}_k$ and the set of points on the obstacle perimeter is described by a continuous function. This paper considers obstacle representations in terms of set of points, circles, lines and polygons. An obstacle is represented as a simply-connected region in $\mathbb{R}^2$ and is denoted as $\mathcal{O}$. In the presence of R obstacles $\mathcal{O}_l$, $l = 1, 2, \ldots, R$, the subscript l is added. Let $\rho(\mathbf{s}_k, \mathcal{O}) : \mathbb{R}^2 \times S^1 \times \mathcal{O} \to \mathbb{R}$ denote the minimal Euclidean distance between obstacle $\mathcal{O}$ and the pose $\mathbf{s}_k$. The orientation part β_k of $\mathbf{s}_k$ can be neglected in case of a circular shape robot. The inequality constraint maintains a minimum separation ρ_{min} between all obstacles and pose $\mathbf{s}_k$ according to:

$$\mathbf{o}_k(\mathbf{s}_k) = [\rho(\mathbf{s}_k, \mathcal{O}_1), \rho(\mathbf{s}_k, \mathcal{O}_2), \ldots, \rho(\mathbf{s}_k, \mathcal{O}_R)]^\intercal - [\rho_{min}, \rho_{min}, \ldots, \rho_{min}]^\intercal \quad (12)$$

The equality constraint (12) in itself restricts the feasible set of robot positions (x_k, y_k) to $\{\mathbb{R}^2 \setminus (\bigcup_{l=1}^{R} \tilde{\mathcal{O}}_l)\}$ in which $\tilde{\mathcal{O}}_l$ denotes the obstacle region $\mathcal{O}_l$ inflated by ρ_{min}. Obviously, this set is non-convex such that the corresponding nonlinear program (2) exhibits multiple local minima. The occurrence of local minima caused by the presence of obstacles is a common phenomenon in trajectory and path planning. The number of obstacles l has a significant influence on the number of local minima. Due to limited computational resources local optimization methods are preferred for online optimziation, therefore the solution strongly depends on the globally planned trajectory $\mathbf{b}$ which serves as an initial solution for the local optimizer. Note, for most practical applications the number of distance calculations required during the optimization might be reduced such that each obstacle $\mathcal{O}_l$ only effects a subset of nearby poses, rather than the entire sequence of poses $\mathbf{s}_k, \forall k$. The association between obstacles and nearby poses is updated between subsequent sampling intervals to refine suboptimal solutions during runtime.

2.3. Approximative least-squares optimization

Solving nonlinear programs with hard constraints is computational expensive. Therefore, improving the efficiency of fast online solvers has become an important research topic over the last decade. The TEB approach further investigates the application of unconstrained optimization techniques since they are well studied and mature implementations in open-source packages are widely available. Unconstrained optimization avoids Lagrange resp. Karush-Kuhn-Tucker multipliers (dual variables) causing the dimension of the Hessian matrices to become identical with the number of primal variables in $\mathbf{b}$.

The exact nonlinear program (2) is transformed into an approximative nonlinear least-squares optimization problem that is solved efficiently as the solver approximates the Hessian by first order derivatives and exploits the sparsity pattern of the problem. Constraints are incorporated into the objective function as additional penalty terms. Since the unconstrained objective function is composed of squared nonlinear terms only, quadratic penalty functions are applied according to [21]. Note, that other unconstrained approximations such as log-barrier, augmented Lagrangian or exact penalty methods [22] exist, which however contain non-squared terms and therefore are not applicable to unconstrained least-squares solvers.

In the following arguments of constraints are omitted for better readability. The equality constraint $\mathbf{h}$ is expressed in terms of a quadratic penalty with a scalar weight σ_h and identity $\mathbf{I}$ by:

$$\phi(\mathbf{h}_k, \sigma_h) = \sigma_h \mathbf{h}_k^T \mathbf{I} \mathbf{h}_k = \sigma_h \|\mathbf{h}_k\|_2^2 \tag{13}$$

Inequalities are approximated by weighted one-sided quadratic penalties:

$$\chi(\boldsymbol{\nu}_k, \sigma_\nu) = \sigma_\nu \| \min\{\mathbf{0}, \boldsymbol{\nu}_k\} \|_2^2 \tag{14}$$

The min-operator is applied row-wise. Further inequalities $\boldsymbol{\alpha}_k$ and $\mathbf{o}_k$ are approximated in a similar fashion. Initial and final constraint, $\mathbf{s}_s$ and $\mathbf{s}_f$ respectively, are eliminated by substitution and are therefore not subject to the optimization. $\Delta T_k > 0$ is implicitly incorporated by the difference quotients in (7)

and (8) since the quotients diverge for $\Delta T_k \to 0$ (initial ΔT_k must be positive). The overall unconstrained optimization problem with objective function $\tilde{V}(\mathbf{b})$ that approximates (2) is given by:

$$\mathbf{b}^* = \arg \min_{\mathcal{B} \setminus \{\mathbf{s}_1, \mathbf{s}_n\}} \tilde{V}(\mathbf{b}) \tag{15}$$

$$\tilde{V}(\mathbf{b}) = \sum_{k=1}^{n-1} \big[\Delta T_k^2 + \phi(\mathbf{h}_k, \sigma_h) + \chi(r_k, \sigma_\nu) + \chi(\boldsymbol{\nu}_k, \sigma_\nu) + \chi(\mathbf{o}_k, \sigma_o) + \cdots \tag{16}$$

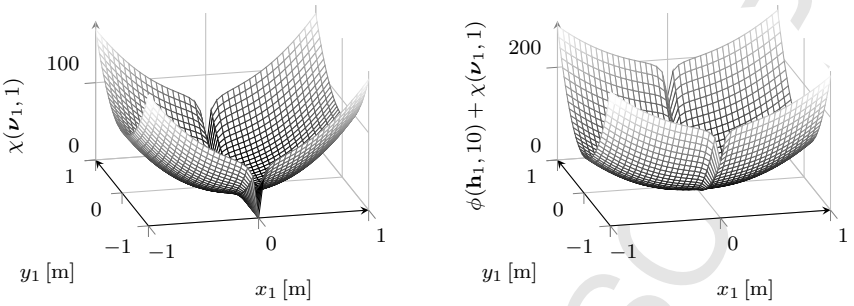
$$+ \chi(\boldsymbol{\alpha}_k, \sigma_\alpha)\big] + \chi(\boldsymbol{\alpha}_n, \sigma_\alpha) \tag{17}$$

Variable $\mathbf{b}^*$ denotes the optimal parameter vector. From the theory of quadratic penalties [21] it is well known that $\mathbf{b}^*$ only coincides with the actual minimizer of the nonlinear program (2) in case all weights tend towards infinity $\sigma \to \infty$. Unfortunately, large weights introduce ill-conditioned characteristics of the problem such that the underlying solver does not converge properly due to inadequate step sizes. The TEB approach abandons the true minimizer in favor of a suboptimal but computationally more efficiently obtained solution with user defined weights. For small to medium sized cluttered environments within the robots local field of perception our experiments reveals that unit weights of 1 provide a reasonably point of departure, except for the weight σ_h associated with the equality constraint of the non-holonomic kinematics which should be chosen a few magnitudes higher (≈ 1000). This setting is recommended as the penalty function (13) is small compared to the inequality constraints and a near-perfect compliance of the trajectory with the robot kinematics is crucial.

The transformation of the non-convex signed velocity constraint (10) onto a quadratic penalty term introduces local minima (see Fig. 3a). However, the Fig. 3b reveals that these local minima are canceled in the overall objective function by the conflicting non-holonomic-constraint (5).

2.4. Solution of the approximative optimization problem

The literature proposes an abundance of algorithms for solving nonlinear least-squares problems such as (15). Popular solvers are Gauss-Newton, Dogleg or Levenberg-Marquardt (LM) algorithm [21]. The TEB approach utilizes LM



(a) Limiting velocity penalty

(b) Superposed velocity and non-holonomic constraint penalties

Figure 3: Limiting velocity and non-holonomic constraint penalty values for a varying position $(x_1,\, y_1)$. Variables $x_2 = y_2 = 0\,\mathrm{m}$, $\beta_1 = \beta_2 = 0\,\mathrm{rad}$, $\Delta T_1 = 0.1\,\mathrm{s}$ and $v_{max} = 1.0\,\frac{\mathrm{m}}{\mathrm{s}}$ are fixed.

due to its proper balance between robustness and efficiency. LM constitutes a trust-region strategy that only accepts step sizes that decrease the overall cost. The optimization problem is regularized implicitly in case it becomes singular. Applying the approach requires the solution of a sparse linear system for which $(\mathbf{H} + \lambda \mathbf{I})^{-1}$ is computed with a damping factor λ. $\mathbf{H} = \mathbf{J}^\intercal \mathbf{J}$ denotes the Hessian that itself depends on the Jacobian $\mathbf{J}$. The open-source graph optimization framework *g2o* [23] implements a highly efficient sparse variant of LM which is employed for solving (15). Notice, that the ordering of poses and time intervals in $\mathbf{b}$, see (1), affects the structure of the optimization problem. Since terms of the objective function and individual constraints in (2) depend only on a small subset of parameters, the resulting Hessian is sparse and banded. The formulation of the minimum time objective presented here differs from [9] in which $V(\mathbf{b}) = (\sum_{k=1}^{n-1} \Delta T_k)^2 = T^2$ is minimized which leads to a dense block of size $(n-1) \times (n-1)$ in the Hessian. Furthermore, according to the discussion in section 2.2 the solution for individual ΔT_k in the previous strategy in [9] implies a non-zero null space. In that case the convergence significantly relies on a proper regularization and it is highly sensitive with respect to cost function weights and problem dimension.

2.5. Closed-loop control

The optimization problem presented in the previous sections is solved repeatedly during runtime. Within each sampling interval only the first control input of the planned trajectory is commanded to the robot. This is a common procedure in model predictive control to account for disturbances and changes in the environment by feedback. In our case mobile robots are controlled by a translational and rotational reference velocity w.r.t. their center of rotation. These control inputs, in particular v_1 and ω_1, are calculated according to (7) and (8) respectively. Furthermore, the TEB approach supports warm-starting to efficiently refine previously obtained solutions. Within each sampling interval, start and final poses are updated and the previous trajectory is resampled w.r.t. its length n to account for changing magnitudes of ΔT_k and to guide the planner towards a desired temporal discretization ΔT_{ref} of the trajectory. A new sample is inserted between $\mathbf{s}_k$ and $\mathbf{s}_{k+1}$ if ΔT_k is larger than the reference step size, otherwise $\mathbf{s}_{k+1}$ is removed [9]. A small hysteresis ΔT_{hyst} in regulating the number of samples avoids excessive oscillations. The experimental results are obtained with $\Delta T_{ref} = 0.3\,\mathrm{s}$ and $\Delta T_{hyst} = 0.1\Delta T_{ref}$. The initial trajectory length n is obtained from a uniform partition of the initial trajectory into piecewise linear segments with equidistant spatial separation of 0.3 m. In the following the approach is extended further, such that the focus on closed-loop control is resumed in section 4.

3. Exploring distinctive topologies

The following sections address the problem of finding the global minimum of (2) resp. (15). As described in section 2.2, the presence of obstacles introduces multiple local minima since the resulting set of feasible robot poses is non-convex. Therefore, finding local minima coincides with the extraction of distinctive topologies. The developed approach aims to identify N relevant topologies implicated by occupied obstacle regions and provides an initial trajectory $\mathbf{b}_j, j = 1, 2, \ldots, N$ for each topology. These initial trajectories are intended

to be optimized in parallel by the TEB approach as described in section 2. The least-cost trajectory is selected from the set of alternatives to reveal the global minimizer. Furthermore, the developed approach is integrated in the state feedback of the closed-loop control system to explore and update the set of local solutions during runtime.

3.1. Equivalence relations: Homotopy classes and homology classes

Under the reasonable assumption that local minima are caused by obstacles, the search space for identifying local candidates is reduced to the 2D plane containing only position parts $\mathbf{z}_k = (x_k, y_k) \in \mathbb{R}^2$ of $\mathbf{s}_k$, since they are not allowed to be part of $\bigcup_{l=1}^{R} \tilde{\mathcal{O}}_l$. Parameters β_k and ΔT_k are not affected directly. The path in terms of a sequence of robot positions is defined as $\tau = (\mathbf{z}_k)_{k=1,2,\ldots,n}$. $\mathbf{b} = \mathbf{h}(\tau)$ denotes the mapping from the (initial) path to the optimization parameter $\mathbf{b}$. Notice, that the global cost function might posses multiple local minima due to symmetry of trajectories or solutions with initial robot forward or backward motion. In our case the initial local solution is dictated by the path generated by the global planner.

Without loss of generality, in the following a 2D position $\mathbf{z}_k \in \mathbb{R}^2$ of a path τ is represented in the complex domain by $z_k = x_k + \mathrm{i}y_k \in \mathbb{C}$. The same applies for obstacles $\mathcal{O}_l$ that are embedded in $\mathbb{C}$ rather than $\mathbb{R}^2$. In the following boldface script is omitted whenever the complex domain representation is preferred.

The presented approach is based on the theory of homotopy classes. First homotopic paths are defined according to [24]:

Definition 1 (Homotopic paths)**.** *Two paths τ_1 and τ_2 connecting the same start and goal points $\mathbf{z}_s$ and $\mathbf{z}_f$ respectively, are homotopic if and only if one can be continuously deformed into the other without intersecting any obstacles. The set of all paths that are homotopic to each other is denoted as homotopy class.*

Based on the above Definition 1 local optimization methods for dynamic problems with warm-starting establish a homotopy between the solution paths

τ^* and τ of subsequent optimization steps. Let $\mathcal{A} = \{\mathbf{b}_\epsilon \in \mathcal{B} \mid \tilde{V}(\mathbf{b}^*) \leq \tilde{V}(\mathbf{b}_\epsilon), \mathbf{b}_\epsilon = \mathbf{b}^* + \boldsymbol{\epsilon}, \boldsymbol{\epsilon} > \mathbf{0}\}$ denote the (attractive) vicinity of a local minimum $\mathbf{b}^*$ with the objective function $\tilde{V}$ of (17). Figuratively, all paths τ_ϵ with $\mathbf{b}_\epsilon = \mathbf{h}(\tau_\epsilon) \in \mathcal{A}$ that converge to the same local minimum are homotopic and therefore relate to the same homotopy class. A single representative $\mathbf{b}_\epsilon^{(i)} \in \mathcal{A}^{(i)}$ of each homotopy class $\mathcal{A}^{(i)}$ provides a valid initial solution of problem (15) and is sufficient to identify the local minimum $\mathbf{b}^{*(i)}$ of the i-th homotopy class.

The closed form generic computation of homotopy classes is difficult. [24] suggests substituting the homotopy with homology classes as they are easier to compute. A homology class defines a set of homologous paths in which elements are homologous to each other.

Definition 2 (Homologous paths)**.** *Two paths τ_1 and τ_2 connecting the same start and goal points $\mathbf{z}_s$ and $\mathbf{z}_f$ respectively, are homologous if and only if $\tau_1 \sqcup -\tau_2$ forms the complete boundary of a 2D manifold embedded in $\mathbb{C}$ not containing and intersecting any obstacle.*

Homotopy implies homology, but the reverse implication does not hold. However, for most practical mobile robot planning scenarios, both definitions can be considered as equivalent. Figure 4 shows an example in which paths τ_1 and τ_2 belong to two different homotopy classes, since they cannot be transformed continuously into each other without intersecting any obstacle. On the other hand τ_1 and τ_2 are homologue to each other, since the area $A = A_1 \cup A_2$ spanned by the disjoint union $\tau_1 \sqcup -\tau_2$ does not contain obstacles. They belong to the same homology class. Notice, homologous paths might also be defined by vanishing winding numbers evaluated at each obstacle within the cycle $\tau_1 \sqcup -\tau_2$.

[24] and [17] present an analytical approach to determine homology classes based on complex analysis. In summary, the homology invariant termed H-signature constitutes an equivalence relation that assigns a unique complex number to paths of the same homology class. The H-signature is originally defined for continuous paths between start and goal points z_s and z_f respectively. Let $\tilde{\tau}(t)$ denote a continuous path such that $\tilde{\tau}(t = 0) = z_s$ and $\tilde{\tau}(t = T) = z_f$,

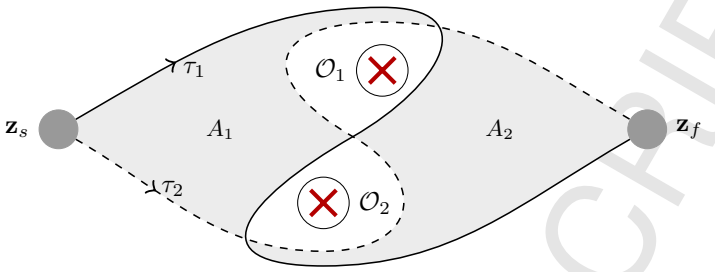


Figure 4: Illustrative example of homologue but not homotopic paths borrowed from [24].

the complex homology invariant is defined by:

$$\mathcal{H}(\tilde{\tau}) = \int_{\tilde{\tau}} \mathcal{F}(z)dz \tag{18}$$

Equation (18) follows immediately from the definition of the Cauchy's integral formula [17]. Obviously, $\mathcal{F}(z)$ depends on the obstacle positions. [17] suggest the following function $\mathcal{F}(z)$, referred to as obstacle marker function:

$$\mathcal{F}(z) = \frac{f_0(z)}{(z-\xi_1)(z-\xi_2)\cdots(z-\xi_R)} \tag{19}$$

$(\xi_l \subseteq \mathcal{O}_l) \in \mathbb{C}, \forall l = 1, 2, \ldots, R$ denote representative points of R obstacles. Each representative point ξ_l is arbitrarily chosen from the interior of the obstacle region respectively obstacle shape $\mathcal{O}_l$. f_0 denotes an arbitrary analytic function over $\mathbb{C}$. In our experiments we chose $f_0(z) = ab(z - BL)(z - TR)$ with $a = \text{ceil}(\frac{R}{2})$ and $b = R - a$. Operator ceil$(\cdot)$ maps to the smallest following integer. Parameters $BL \in \mathbb{C}$ and $TL \in \mathbb{C}$ denote the bottom left and upper right corner of the environment. This formulation slightly differs from the one presented in [17], as it improves the scalability w.r.t. number of obstacles ($R = 0 - 500$ in our experiments with typical H-signature values of less than 10^{10}). The original formulation tends to large H-signatures ($> 10^{100}$) even for hundreds of obstacles for which numerically instabilities could occur.

In order to calculate the H-signature of the discrete path τ, the analytic solution of (18) for line segments is utilized. [17] derives an analytic expression

for a line segment connecting two points z_k and z_{k+1}:

$$\mathcal{H}_s(z_k, z_{k+1}) = \sum_{l=1}^{R} A_l(\ln(z_{k+1} - \xi_l) - \ln(z_k - \xi_l)) \tag{20}$$

with $A_l = f_0(\xi_l)\left[\prod_{j=1,j\neq l}^{R}(\xi_l - \xi_j)\right]^{-1}$. The H-signature of a discrete path (composed of line segments) is calculated by:

$$\mathcal{H}(\tau) = \sum_{k=1}^{N-1} \mathcal{H}_s(z_k, z_{k+1}) \tag{21}$$

Note that the actual implementation of (20) requires the theory of complex logarithm. [24] suggests selecting the branch that minimizes the angle between $z_{k+1} - \xi_l$ and $z_k - \xi_l$ (by testing some values $\alpha \in \mathbb{Z}$ close to zero):

$$\begin{aligned}\mathcal{H}_s(z_k, z_{k+1}) = \sum_{l=1}^{R} A_l \Big[&\ln(|z_{k+1} - \xi_l|) - \ln(|z_k - \xi_l|) + \dots \\ &+\mathrm{i}\min_{\alpha\in\mathbb{Z}}\Big(|\arg(z_{k+1} - \xi_l) - \arg(z_k - \xi_l) + 2\alpha\pi|\Big)\Big]\end{aligned} \tag{22}$$

The proposed H-signature determines whether multiple paths belong to the same homology class which is fulfilled if all H-signatures are identical up to numerical precision.

3.2. Discovery of Homology Classes

Based on the homology invariant proposed in the previous section, an algorithm for exploring relevant homology classes is developed. [17] introduces a search graph that is augmented by H-signatures in order to restrict trajectories to a given admissible set of homology classes while invoking an A^*-search for the optimal trajectory. In contrast, our approach does not attempt to directly solve the planning problem with graph search. Instead, it solves (15) with a local online optimization method. This modification requires the ongoing maintenance of current and discovery of novel homology classes in conjunction with the underlying trajectory optimization. The required computation time is of particular importance since solving nonlinear problems such as (15) still requires a substantial computational effort. Therefore, the proposed approach

gathers coarse, collision free candidate trajectories which consecutive waypoints are arranged in forward direction only. The H-signature is applied as a filter to eliminate all but one trajectory for each homology class.

Given the robot's current position z_s, goal z_f (both in $\mathbb{C}$ notation) and the set of obstacle regions $\mathcal{O} = \{\mathcal{O}_l \mid l = 1, 2, \ldots, R\}$, an exploration graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ is constructed in order to gather an initial subset of admissible paths. The set of vertices is defined by $\mathcal{V} = \{z_s, \zeta_i, z_f \in \mathbb{C} \mid \forall \zeta_i \notin \mathcal{O}, i = 1, 2, \ldots, I\}$. $\zeta_i \in \mathbb{C}$ are waypoint samples that later may become a part of the trajectory.

3.2.1. Exploration based on Voronoi Diagrams

A complete exploration graph with all feasible candidate trajectories is generated from a Voronoi diagram of the environment. Given a distance metric (e.g. Euclidean distance), the Voronoi diagram partitions the 2d plane into multiple regions $\mathcal{R}_l \subseteq \mathbb{C}$ according to the number of obstacles $l = 1, 2, \ldots, R$. Each region $\mathcal{R}_l$ defines the set of points which are (strictly) closer to the corresponding obstacle $\mathcal{O}_l$ than to all other obstacles $\mathcal{O} \setminus \mathcal{O}_l$. Formally, the Voronoi diagram $\mathcal{G}_{vd}$ is defined as the set of all region boundaries $\mathcal{G}_{vd} = \mathbb{C} \setminus \cup_{l=1}^{R} \mathcal{R}_l$. In order to transform $\mathcal{G}_{vd}$ into an exploration graph $\mathcal{G}$, the set $\mathcal{G}_{vd}$ is discretized spatially such that discretized points $\zeta_i, i = 1, 2, \ldots, I$ are considered as vertices and the corresponding part of the boundary between two consecutive vertices is included as a bidirectional edge. Furthermore, start and goal positions, z_s and z_f respectively, are connected to the vertices obtained from the Voronoi diagram. A reasonable strategy for both start and goal is to find the closest point ζ_i each for which an edge in terms of collision-free line segment can be generated. An exploration graph $\mathcal{G}$ obtained from the Voronoi diagram for an example environment is illustrated in Fig. 5.

Based on the generated graph $\mathcal{G}$, its simple paths between z_s and z_f are extracted by a depth-first search augmented by a visited list. The H-signature for each path is calculated according to (21) and is added to the set of known signatures in case it is not a member yet. Path candidates with duplicate H-signature are discarded. Even though the Voronoi diagram already provides

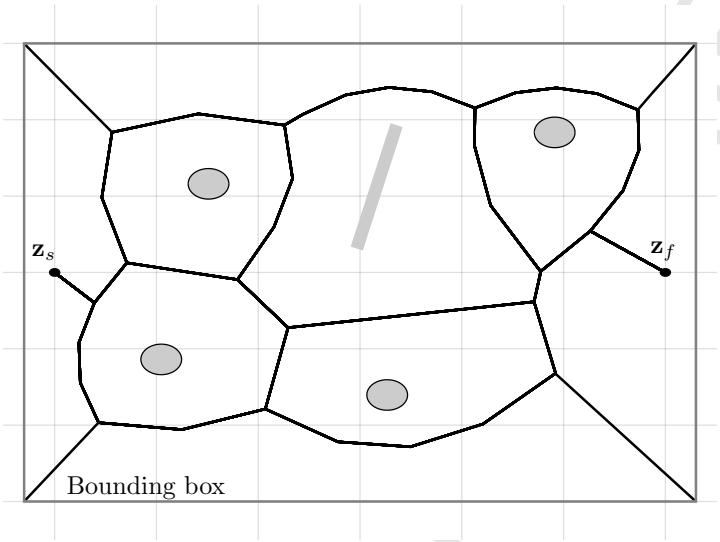


Figure 5: Example of a discretized Voronoi diagram with connected start $\mathbf{z}_s$ and goal $\mathbf{z}_f$.

distinctive topologies, the H-signature is required to determine previously found and optimized trajectories as described in the following section 4.

The two steps of finding all simple paths and filtering homologue paths is combined into a single search algorithm to improve efficiency. Algorithm 1 constitutes a modified recursive depth-first search. The set L contains those vertices already visited such that after reaching the goal, L consists of the complete path candidate from z_s to z_f. This path candidate is matched with potential homologue duplicates in H in line 10 and 11. If its homology class is novel, the corresponding trajectory for the underlying optimization problem is initialized from the path L. The coarse path defined by $\{z_s, \zeta_i, z_f\}$ is subsampled and the orientation parts β_k of the poses are initialized according to the direction among subsequent positions. Rather than storing the 2D position part τ of the trajectories, the complete optimization parameter $\mathbf{b}$ is stored for subsequent warm-starts of the optimization. The number of maximum distinctive topologies is specified at line 2 in order to limit the computation time.

Algorithm 1 Find paths in alternative homology classes

Input: $\mathcal{G}$ - reference to the acyclic graph; L - reference to an ordered visited list containing only z_s; z_f - goal vertex; T - reference to the trajectory set; H - reference to the set of H-signatures

Output: Updated set of trajectories and H-signatures

function DEPTHFIRST($\mathcal{G}, L, z_f, T, H$)
  if max. size of T is reached **then**
    return
  $l \leftarrow L$.back() ▷ Get last visited vertex
  for each adjacent vertex w of vertex l in $\mathcal{G}$ **do**
    if $w \in L$ **then** ▷ Already visited
      continue
    if $w == z_f$ **then** ▷ Goal reached
      L.append(w) ▷ Finalize trajectory
      h $\leftarrow$ CALCHSIG(L) ▷ See Eq. (22)
      if $h \notin H$ **then**
        $b \leftarrow$ INITTRAJECTORY(L) ▷ $\mathbf{b} = \mathbf{h}(\tau)$
        T.append(b) ▷ Save complete trajectory
        H.append(h) ▷ Store H-signature
      break
  for each adjacent vertex w of vertex l in G **do**
    if $w \in B$ **or** $w == z_f$ **then** ▷ Already visited or goal reached
      continue
    L.append(w)
    DEPTHFIRST($\mathcal{G}, L, z_f, T, H$) ▷ Recursion step
    L.pop(w)
return

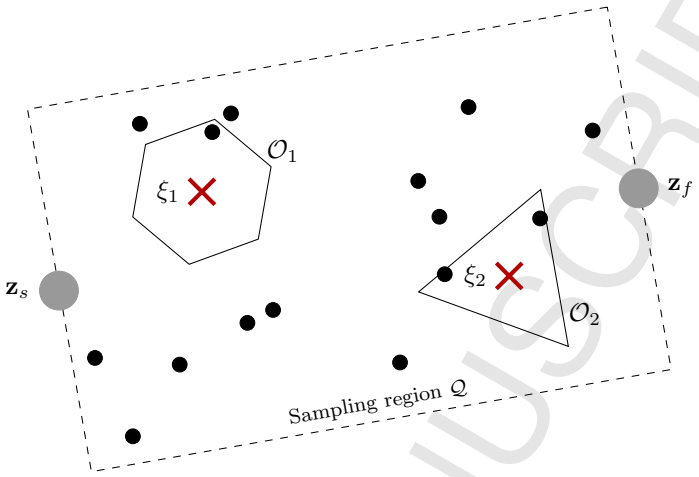


Figure 6: Example of sampled exploration graph

3.2.2. Sampling-based Exploration Strategy

As an alternative to a complete exploration we propose a sampling strategy to generate the exploration graph $\mathcal{G}$. The previously presented approach based on Voronoi diagrams is complete, but the computation time for generating the candidate trajectories does not scale well with the number of obstacles. In the context of online trajectory planning under limited computational resources one favors faster computations to identify a subset of most promising candidate trajectories rather than exploring the complete set.

Waypoint sampling follows the probabilistic roadmaps (PRM) approach [25]. Hence, waypoints ζ_l are sampled uniformly from a predefined region $\mathcal{Q} \subseteq \mathbb{C}$. Sampled waypoints that intersect with any obstacle region are rejected until a desired number of samples is obtained. The set of edges $\mathcal{E}$ is constructed from the waypoint seeds. An edge connects a pair of vertices $w_1 \in \mathcal{V}$ and $w_2 \in \mathcal{V}$ if the following conditions apply:

- Direction is forward oriented with respect to the goal heading, such that $\Re[\overline{(w_2 - w_1)}(z_f - z_s)]\,(|w_2 - w_1||z_f - z_s|)^{-1} > \theta$ with $\theta \in [0, 1]$.
- Line segment $\mathcal{L} = \{w_1 + t(w_2 - w_1) \mid \forall t \in [0, 1]\}$ does not intersect with

any obstacle $\mathcal{O}_l$ such that $\mathcal{L} \cap \mathcal{O} = \emptyset$.

Obviously, the first condition eliminates those paths which Euclidean distance to the goal does not decrease monotonically. However, online trajectory planners usually obtain their intermediate goals from a global planner. Assuming that these subgoals are properly arranged and ordered the restriction to acyclic graphs is justified. This condition significantly reduces the number of candidate trajectories and computational effort of graph search. The threshold θ for the angular width of the forward direction can be increased to further narrow the line of sight. The remaining steps are identical to the algorithm outlined above. See Fig. 6 for a figurative example, samples inside $\mathcal{O}_1$ or $\mathcal{O}_2$ are rejected. Obviously, the computation time increases exponentially with an increasing number of samples. Sampling is repeated at each iteration such that novel homology classes might still be discovered at a later stage. It is possible to construct artificial environments that violate the assumption. However in most practical applications the assumption is valid for the robots local environment considered for trajectory optimization. Refer to section 5 for the corresponding analysis.

4. Integrated online trajectory planning in distinctive topologies

The following section describes the integration of the homology class discovery with the TEB approach within the robot control feedback loop. Algorithm 2 performs the principal planning step invoked at each sampling interval of the mobile robot. During the initial invocation the set of admissible trajectories Γ is empty (in practical implementations Γ may contain an initial trajectory provided by the global planner). A new graph is created in lines 7-9 according to section 3.2 by seeding random samples in a region of interest (typically a rectangular or a semicircle connecting z_s and z_f). Afterwards the modified depth first search (see Alg. 1) is applied in order to discover distinctive homology classes. A single representative trajectory featuring a unique H-Signature is initialized for each class in Γ. Candidate trajectories $\forall \mathbf{b}_i \in \Gamma$ are optimized simultaneously in lines 11-14 by applying I_{teb} iterations to minimize (17). Additionally, each

Algorithm 2 Online planning step which is invoked repeatedly

Input: $s_s = (z_s, \beta_s)$ - start pose; $s_f = (z_f, \beta_f)$ - final pose; (v_s, ω_s) - initial velocities; Γ - reference to the trajectory set; H - reference to the set of H-signatures; $\mathcal{O}$ - set of obstacles

Output: (Sub-)optimal control $\mathbf{u}^*$

1: **function** PLANTRAJECTORIES($s_s, s_f, \Gamma, H, \mathcal{O}$)
2: **if** Γ is not empty **then** ▷ Γ.size() == H.size()
3: $\Gamma \leftarrow$ UPDATETRAJECTORIES($\Gamma, s_s, s_f, v_s, \omega_s$)
4: **if** size(Γ) > 1 **then**
5: $(\Gamma, H) \leftarrow$ DELETEDETOURS($\Gamma, s_s, s_f, \mathcal{O}$)
6: $H \leftarrow$ RENEWHC($T, H, \mathcal{O}$)
7: $L \leftarrow$ allocate empty visited list
8: L.append(z_s)
9: $\mathcal{G} \leftarrow$ CREATEGRAPH($\mathcal{O}$)
10: DEPTHFIRST($\mathcal{G}, L, z_f, \Gamma, H$)
11: **for** each trajectory $b \in \Gamma$ **do** ▷ parallelizable
12: **for** all Iterations 1 to I_{teb} **do**
13: Adjust length n of the TEB
14: $b \leftarrow$ CALLOPTIMIZER($b, \mathcal{O}$) ▷ solve Eq. (15)
15: $b^* \leftarrow$ SELECTGLOBALOPTIMALTRAJECTORY($T, \mathcal{O}$)
return control input (v_1, ω_1) according to (7), (8) and b^*

trajectory is resampled according to a reference temporal discretization as mentioned in section 2.5 (line 13). The globally optimal trajectory $\mathbf{b}^*$ is selected according to the minimal objective function value (17). Control variables v_1 and ω_1 are obtained from $\mathbf{b}^*$ by applying (7) and (8) respectively.

In subsequent planning step invocations, Γ and H already contain candidate trajectories optimized in previous iterations and hence lines 3-6 are executed. The current start pose $\mathbf{s}_s$, initial velocities (v_s, ω_s) and final pose $\mathbf{s}_f$ are updated according to the novel robot state and perceptions. Additionally, the current set

of trajectories is validated for admissibility if multiple alternatives are available (lines 5-6). In particular the edge conditions of section 3.2.2 are verified. In case of a violation the trajectory and the corresponding H-signature are eliminated. Keeping at least a single trajectory which does not fulfill all edge requirements (line 5) ensures that a trajectory in which the robot must head backwards is not rejected (e.g. by preserving the global plan as candidate). The current set of H-signatures is updated in case obstacle configuration changes (line 6).

5. Experimental Results and Examples

The following examples and scenarios demonstrate the capabilities of the proposed online trajectory planning approach. Algorithms are executed on a machine running Ubuntu with a 3.4 Ghz Intel i7 CPU. The C++ source code is available online [20].

Firstly, the process of finding alternative trajectories in a medium sized environment ($10\,\mathrm{m} \times 10\,\mathrm{m}$) with randomly generated obstacles and start respectively goal positions of the robot is investigated. Exploration results of each generated scenario are analyzed and compared for the proposed exploration strategies, in particular the Voronoi diagram approach and the sampling based approach with a varying number of samples I. Hereby, three different types of obstacle representations are considered: i) point-shaped obstacles, ii) line-shaped obstacles and iii) a mixture of both but with discretized lines (0.1 m steps) in order to simulate map representations based on laser range data which often arises in practical navigation scenarios. For each type 100 environments are generated such that the minimum distance between obstacles is at least 1.5 m and that line segments are not longer than 4 m. Additionally, each exploration strategy is invoked multiple times for each environment by maintaining the set of previously found candidates. Hereby, the capabilities of revealing new candidate trajectories during online exploration is investigated. The sampling region $\mathcal{Q}$ is chosen as oriented rectangle between the robot's current start z_s and goal z_f. It's width is 6 m and the length is expanded by 10 % of $|z_f - z_s|$ in order to

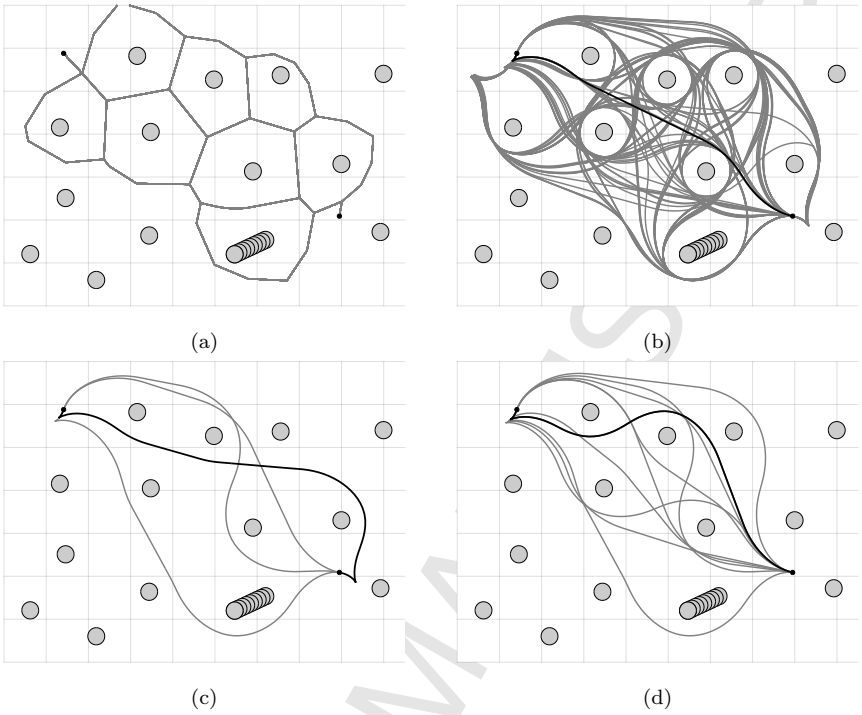


Figure 7: Application of Alg. 2 with an internal bounding box (width: 6 m) between start and goal. (a) Simple paths found with the Voronoi based approach and (b) the corresponding TEB optimization result, the black solid line corresponds to the selected candidate. (c) Found and optimized trajectories using the sample based approach (with $I = 10$) after a single invocation and (d) after ten invocations.

take further samples close to the start and goal into account. The maximum line of sight for each edge with respect to the goal heading is set to $\theta = \pi/3$. In comparison, the Voronoi diagram is limited to the same rectangle as bounding box but without restricting line of sight. The Voronoi diagram is generated based on an efficient implementation of the algorithm in [26] as part of the *boost* open-source software library. Edges that separate two close obstacles (separation smaller than the robot's expanse) are rejected before invoking Algorithm 1 in order to prevent the planner from generating infeasible trajectories and reducing computation time for scenarios with laser range data and point clouds as obstacle representations.

Table 1: Exploration and computation time analysis for randomly generated environments. Abbreviation "S." denotes the sampling based strategy along with I samples. Hit corresponds to the percentage of the correctly selected trajectory in the 1. step.

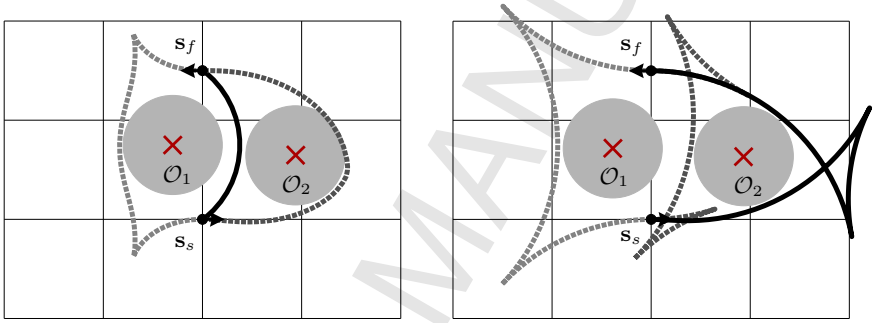
	Strategy	Path coverage [%]				CPU time [ms]	Hit [%]
		Step 1	Step 3	Step 6	Step 10		
Points only	Voronoi	100	100	100	100	189 ± 381	100
	S. ($I = 5$)	36.8	41.4	47.0	48.6	6 ± 9	90
	S. ($I = 10$)	40.2	48.2	49.9	52.0	11 ± 19	92
	S. ($I = 15$)	42.7	50.6	52.1	55.2	18 ± 32	93
	S. ($I = 20$)	45.6	51.8	54.9	57.2	36 ± 76	96
Lines only	Voronoi	100	100	100	100	11 ± 34	100
	S. ($I = 5$)	57.6	69.3	75.6	76.8	1 ± 2	89
	S. ($I = 10$)	66.4	78.1	79.2	81.3	2 ± 3	93
	S. ($I = 15$)	70.1	81.2	81.5	82.5	3 ± 4	93
	S. ($I = 20$)	76.7	80.6	83.3	84.5	7 ± 17	93
Points & discr. lines	Voronoi	100	100	100	100	2011 ± 4466	100
	S. ($I = 5$)	57.6	69.3	75.6	76.8	65 ± 83	92
	S. ($I = 10$)	66.4	78.1	79.2	81.3	108 ± 145	95
	S. ($I = 15$)	70.1	81.2	81.5	82.5	166 ± 243	95
	S. ($I = 20$)	76.7	80.5	83.3	84.5	332 ± 691	96

Table 1 depicts the statistical results including path coverage with respect to the complete list of explored candidates obtained on the Voronoi diagram. Furthermore, it presents the percentage of correctly selected globally optimal trajectories among the subset of candidates (correct hit) and finally the mean and standard deviation of the required computation time (CPU time). Path coverage is provided for multiple exploration invocations ($1, 3, 6$ and 10) as mentioned before. Figure 7 depicts an example environment with randomized points and discretized line segments and illustrates results for both strategies. Since the Voronoi strategy is considered as complete within the configured bounding

box, its candidate set and the corresponding selected trajectory are considered as reference. The analysis reveals that the sampling based strategy achieves an average path coverage of approximately 60%. Notably, the least cost trajectory is among the explored set in approximately $90 - 95\%$ after the first exploration invocation even by seeding only a few samples $(5 - 10)$ and by requiring only a fraction of the computation time compared to our Voronoi implementation. This observation supports the hypotheses that the globally optimal trajectory does not differ substantially from the line of sight between the current start and goal frame. Obviously, this does not apply for arbitrary large environments, but holds for scenarios in which the local optimal planner is confronted with multiple small static and dynamic obstacles within the local perception of the robot while following intermediate goal points provided by the global planner. The global planner in that case is assumed to be aware of large (static) obstacles and barriers in the environment. Hence it is worth considering the sampling based approach in applications in which a full path coverage for finding the globally optimal trajectory is not crucial in exchange for faster online planning and still being able to reveal alternative trajectories around obstacles. The Voronoi based approach is preferable if environments are large and the planner is utilized in a global planning context. Notice, for preprocessed or structured obstacle representations such as line segments or polygons, the Voronoi strategy handles the complete exploration even in a few milliseconds for the investigated environments. In the following, the sampling based approach with 15 samples is selected for local planning applications.

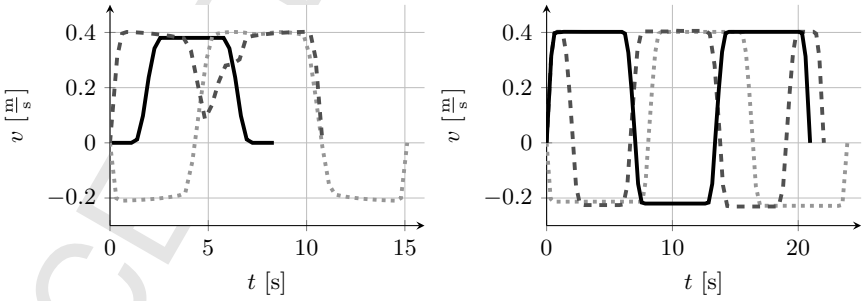
The second example demonstrates the extension of the TEB approach to support reversals of driving direction and kinematics of a carlike robot. Furthermore it investigates whether the globally optimal trajectory is correctly selected among the set of alternative maneuvers. The final pose $\mathbf{s}_f$ is located 1.5 m next to the start pose however with opposite final orientation. Figure 8a depicts three distinctive trajectories for a differential drive robot. The translational velocity profile is bounded to the interval $[-0.2\,\frac{\mathrm{m}}{\mathrm{s}},\, 0.4\,\frac{\mathrm{m}}{\mathrm{s}}]$ and are shown in Fig. 9a. As expected the trajectory that achieves the shortest transition time (black solid

line) is selected as the global optimal solution. The same scenario is performed for a carlike robot with a lower limit on the turning radius (6) to $r_{min} = 2\,\mathrm{m}$ (see Fig. 8b). Here, also the fastest candidate trajectory is commanded to the robot as depicted in Fig. 9b. In this scenario, the number of solver and resizing invocations I_{teb} is set to 4 (see Alg. 2) and the number of LM iterations as part of the g2o framework is set to 5. The average CPU time measured for each call to Alg. 2 is 45 ms including exploration, maintenance and optimization of all 3 alternatives. Hereby, all 3 alternatives are found, initialized and converged within the first 500 ms.



(a) Differential-drive robot (b) Carlike robot with $r_{min} = 2\,\mathrm{m}$

Figure 8: Moving from $\mathbf{s}_s = [0\,\mathrm{m}, 0\,\mathrm{m}, 0\,\mathrm{rad}]^\intercal$ to $\mathbf{s}_f = [0\,\mathrm{m}, 1.5\,\mathrm{m}, \pi\,\mathrm{rad}]^\intercal$. Cell size: $1\,\mathrm{m}^2$. The solid trajectory represents the globally optimal solution among the set of alternatives.



(a) Differential-drive robot (b) Carlike robot

Figure 9: Translational velocity profile according to scenarios in Fig. 8 (globally optimal solution: solid line, alternatives: dashed and dotted lines).

The following two scenarios demonstrate the application of the proposed algorithm to closed-loop control of a Pioneer 3DX differential drive mobile robot in environments with dynamics obstacles. Humans are referred to as dynamic obstacles and walls represent static obstacles. Simulations are performed in Gazebo (as part of ROS) using a dynamic model of the Pioneer 3DX and constant velocity models for humans. The robot is equipped with a laser scanner that generates range readings for a 2D horizontal slice of the environment. Each intersection with an obstacle marks the corresponding cell ($15\,\mathrm{cm} \cdot 15\,\mathrm{cm}$) in a costmap as occupied. Furthermore, the TEB treats each cell as individual point obstacle. Translational and angular velocities are limited to $v_{max} = 0.4\,\frac{\mathrm{m}}{\mathrm{s}}$ and $\omega_{max} = 0.3\,\frac{\mathrm{rad}}{\mathrm{s}}$ respectively. The desired minimal separation from obstacles is $0.5\,\mathrm{m}$.

The proposed integrated method is compared with the classical TEB approach (without homology class exploration) and DWA (available in ROS, refer to section 1 for details). The DWA forward simulation time for each sample is $6\,\mathrm{s}$ in order to obtain a trajectory length comparable to the other methods. The first scenario evaluates the closed-loop behavior of the planner in case the

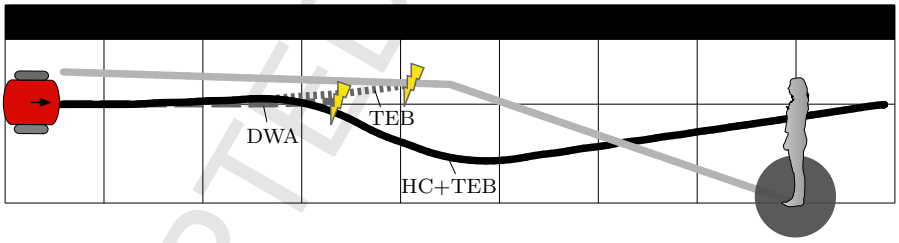


Figure 10: Scenario 1: Traces of the robot's odometry and the human's movement (gray line). Cell size: $1\,\mathrm{m}^2$.

optimal trajectories transit from the left to right or vice versa right to left side of a moving obstacle. In this scenario a human who ignores the approaching robot moves towards a wall. The initial optimal trajectory passes between the wall and the human and becomes inadmissible as the human cuts of the gap for the robot to pass in between the human and the wall. Figure 10 shows the

closed loop trajectory of the robot in response to the evolution of the human motion. At the beginning all planners prefer the fastest trajectory from the homology class along the gap between the human and the wall. The classical TEB gets stuck at the local minimum and collides with the obstacle. The DWA first slows down the robot before switching to the opposite side. The transition is initiated too late such that a collision can no longer be avoided. The TEB optimization with distinctive topologies discovers the opportunity to pass on the opposite side early on, as it considers the alternative path from the beginning. It switches to the alternative path as it becomes more attractive as the original gap closes.

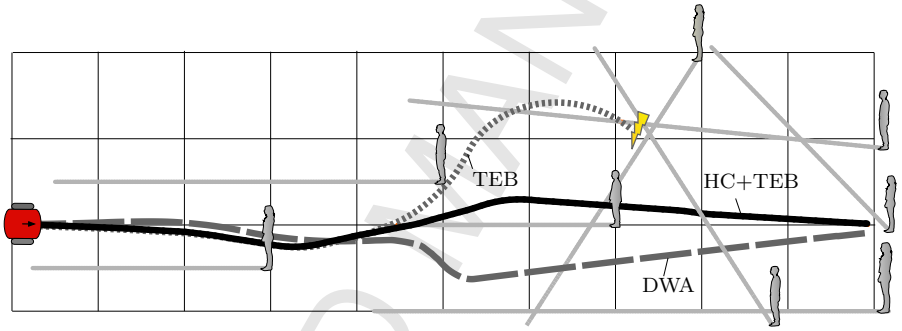


Figure 11: Scenario 2: Traces of the robot's odometry and the humans' movements (solid gray lines). Cell size: $1\,\mathrm{m}^2$.

The second scenario demonstrates the online planning in an open space environment with eight dynamic ignorant obstacles which traces are shown in Fig. 11. The classic TEB collides halfway through the transit of the group, as its preferred trajectory is elongated by the diagonal motion of the human in blue. In addition, the human in dark green causes the robot to collide, since the local planner is unable to switch the homology class. The proposed planner and the DWA are both able to find a collision free path to the goal despite the obstruction caused by the dynamic obstacles. The closed loop trajetory of our approach stays close to the ideal straight line between start and goal. Figure 12 shows the corresponding translational velocity profile of the robot for the different

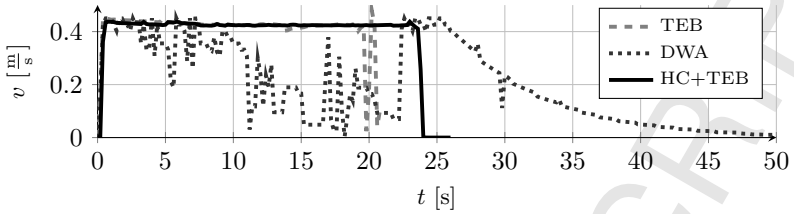


Figure 12: Transl. velocity profile of the robot in scenario 2

planners. The DWA often reduces the robot speed in order to avoid imminent collisions. In contrast, the ability of our approach to consider alternative future evolutions of a scenario enable the robot to navigate at near maximum speed towards the goal. Note, that the DWA implementation reduces speed while approaching the goal, therefore the actual travel time is incomparable.

6. Conclusions and Future Work

The combined approach of homology class exploration and online trajectory optimization for closed-loop planning and control offers the advantage to account for alternative evolutions of scenarios of dynamic obstacles. The comparative analysis of a sampling based exploration strategy with an approach based on Voronoi diagrams reveals application specific advantages of each method. Whereas the latter provides the complete set of alternative trajectories and hence the globally optimal solution, the former generates a sufficient subset for small to medium size environments for limited computational budgets. Examples demonstrate that the *Timed-Elastic-Band* (TEB) approach constitutes a suitable method to efficiently optimize the set of alternative trajectories during runtime while providing time-optimal solutions. With the proposed extensions the TEB approach enables the planning in distinctive topologies even for carlike robots that are restricted to a limited turning radius. The comparison with two planners that do not consider distinctive topological trajectories in simulation illustrates the benefits of multiple trajectory planning in general.

Future work is concerned with extending the search for alternative topolo-

gies to not only the spatial but also the temporal domain. In that predicted trajectories of dynamic obstacles can be incorporated in order to decide whether slowing down or speeding up the robot might be preferred over avoiding obstacles by spatial deformations of the path. Furthermore, in order to establish the practical usefulness of homology based planners it is of interest to analyze the computation time of alternative equivalence relations proposed in the literature.

References

[1] S. Quinlan, Real-Time Modification of Collision-Free Paths, Stanford University, Stanford, CA, USA, 1995.

[2] F. Lamiraux, D. Bonnafous, O. Lefebvre, Reactive path deformation for nonholonomic mobile robots, IEEE Transactions on Robotics 20 (6) (2004) 967–977.

[3] H. Kurniawati, T. Fraichard, From path to trajectory deformation, in: IEEE-RSJ Int. Conf. on Intelligent Robots and Systems, San Diego, États-Unis, 2007.

[4] V. Delsart, T. Fraichard, Reactive Trajectory Deformation to Navigate Dynamic Environments, in: European Robotics Symposium, Czech Republic, 2008.

[5] D. Fox, W. Burgard, S. Thrun, The dynamic window approach to collision avoidance, IEEE Robotics & Automation Magazine 4 (1) (1997) 23–33.

[6] B. Lau, C. Sprunk, W. Burgard, Kinodynamic motion planning for mobile robots using splines, in: Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), St. Louis, MO, USA, 2009, pp. 2427–2433.

[7] C. Sprunk, B. Lau, P. Pfaff, W. Burgard, Online generation of kinodynamic trajectories for non-circular omnidirectional robots, in: Proc. of the IEEE

International Conference on Robotics and Automation (ICRA), Shanghai, China, 2011, pp. 72–77.

[8] N. Ratliff, M. Zucker, J. A. Bagnell, S. Srinivasa, Chomp: Gradient optimization techniques for efficient motion planning, in: IEEE Int. Conference on Robotics and Automation, 2009.

[9] C. Rösmann, W. Feiten, T. Wösch, F. Hoffmann, T. Bertram, Trajectory modification considering dynamic constraints of autonomous robots, in: 7th German Conference on Robotics (ROBOTIK), 2012, pp. 74–79.

[10] C. Rösmann, W. Feiten, T. Wösch, F. Hoffmann, T. Bertram, Efficient trajectory optimization using a sparse model, in: 6th European Conference on Mobile Robots (ECMR), 2013, pp. 138–143.

[11] C. Rösmann, F. Hoffmann, T. Bertram, Timed-elastic-bands for time-optimal point-to-point nonlinear model predictive control, in: European Control Conference (ECC), 2015, pp. 3352–3357.

[12] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, S. Schaal, Stomp: Stochastic trajectory optimization for motion planning, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, 2011.

[13] M. Kuderer, C. Sprunk, H. Kretzschmar, W. Burgard, Online generation of homotopically distinct navigation paths, in: IEEE International Conference on Robotics and Automation, Hong Kong, China, 2014, pp. 6462–6467.

[14] E. Schmitzberger, J. Bouchet, M. Dufaut, D. Wolf, R. Husson, Capture of homotopy classes with probabilistic road map, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, Vol. 3, 2002, pp. 2317–2322.

[15] L. Jaillet, T. Simeon, Path deformation roadmaps: Compact graphs with useful cycles for motion planning, The International Journal of Robotics Research 27 (11-12) (2008) 1175–1188.

[16] R. A. Knepper, S. Srinivasa, M. T. Mason , Toward a deeper understanding of motion alternatives via an equivalence relation on local paths, International Journal of Robotics Research 31 (2) (2012) 168–187.

[17] S. Bhattacharya, V. Kumar, M. Likhachev, Search-based path planning with homotopy class constraints, in: Proc. National Conference on Artificial Intelligence, 2010.

[18] F. T. Pokorny, M. Hawasly, S. Ramamoorthy, Multiscale topological trajectory classification with persistent homology, in: Proceedings of Robotics: Science and Systems, Berkeley, USA, 2014.

[19] C. Rösmann, F. Hoffmann, T. Bertram, Planning of multiple robot trajectories in distinctive topologies, in: IEEE European Conference on Mobile Robots, 2015, pp. 1–6.

[20] C. Rösmann, teb_local_planner ROS Package [Online] (2015).
URL `http://wiki.ros.org/teb_local_planner`

[21] J. Nocedal, S. J. Wright, Numerical optimization, Springer series in operations research, Springer, New York, 1999.

[22] E. C. Kerrigan, J. M. Maciejowski, Soft constraints and exact penalty functions in model predictive control, in: UKACC International Conference, Cambridge and UK, 2000.

[23] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, W. Burgard, G2o: A general framework for graph optimization, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2011, pp. 3607–3613.

[24] S. Bhattacharya, Topological and geometric techniques in graph-search based robot planning, Ph.D. thesis, University of Pennsylvania (January 2012).

[25] L. Kavraki, P. Svestka, J.-C. Latombe, M. Overmars, Probabilistic roadmaps for path planning in high-dimensional configuration spaces, IEEE Transactions on Robotics and Automation 12 (4) (1996) 566–580.

[26] S. Fortune, A sweepline algorithm for voronoi diagrams, in: Proceedings of the Second Annual Symposium on Computational Geometry, USA, 1986, pp. 313–322.

Author Information

Christoph Rösmann was born in Münster, Germany, on December 8, 1988. He received the B.Sc. and M.Sc. degree in electrical engineering and information technology from the Technische Unversität Dortmund, Germany, in 2011 and 2013 respectively. He is currently working towards the Dr.-Ing. degree at the Institute of Control Theory and Systems Engineering, Technische Universität Dortmund, Germany. His research interests include nonlinear model predictive control, mobile robot navigation and fast optimization techniques.

Frank Hoffmann received the Diploma and Ph.D. degrees in physics from Christian-Albrechts University of Kiel, Germany. He was a Postdoctoral Researcher at the University of California, Berkeley from 1996-1999. From 2000 to 2003, he was a lecturer in computer science at the Royal Institute of Technology, Stockholm, Sweden. He is currently a professor at TU Dortmund and affiliated with the Chair for Control and Systems Engineering. His research interests are in the areas of robotics, computer vision, computational intelligence, and control system design.

Torsten Bertram received the Dipl.-Ing. and Dr.-Ing. degrees in mechanical engineering from the Gerhard Mercator Universität Duisburg, Duisburg, Germany, in 1990 and 1995, respectively. In 1990, he joined the Gerhard Mercator Universität Duisburg, Duisburg, Germany, in the Department of Mechanical Engineering, as a Research Associate. During 1995-1998, he was a Subject Specialist with the Corporate Research Division, Bosch Group, Stuttgart, Germany, In 1998, he returned to Gerhard Mercator Universität Duisburg as an Assistant Professor. In 2002, he became a Professor with the Department of Mechanical Engineering, Technische Universität Ilmenau, Ilmenau, Germany, and, since 2005, he has been a member of the Department of Electrical Engineering and Information Technology, Technische Universität Dortmund, Dortmund, Germany, as a Professor of systems and control engineering. His research fields are control theory and computational intelligence and their application to mechatronics, service robotics, and automotive systems.

Highlights

- An integrated online trajectory optimization approach is proposed.
- Maintaining and optimization of admissible candidate trajectories of distinctive topologies in order to seek the overall best candidate.
- An exploration strategy based on Voronoi diagrams provides the complete set of alternative trajectories.
- An alternative proposed sampling based strategy generates a sufficient subset for small to medium sized environments for limited computational budgets.
- The underlying trajectory optimization method considers non-holonomic kinematics of differential-drive and carlike robots.