Planning of Multiple Robot Trajectories in Distinctive Topologies

Christoph Rösmann, Frank Hoffmann and Torsten Bertram
Institute of Control Theory and Systems Engineering, Technical University of Dortmund
44227 Dortmund, Germany
Email: christoph.roesmann@tu-dortmund.de

***Abstract*—This contribution presents a novel approach for efficient online planning of topologically distinctive mobile robot trajectories. Trajectory optimization deforms an initial coarse path drafted by a global planner with respect to robot motion related objectives and constraints. The primary objective is to reach a goal state in minimal time on a collision free path that adheres to the kinematic and dynamic constraints of the mobile robot. Conventional local planners get often stuck in a local optimal trajectory as they are unable to transit across obstacles.**

Our approach seeks the globally optimal trajectory as it maintains and optimizes a subset of admissible candidate trajectories of distinctive topologies in parallel. In case of dynamic environments the planner switches to the current globally optimal trajectory among the candidate set. The online trajectory planning with timed elastic bands is tightly integrated with the robot motion feedback control. The comparative analysis with conventional local planners confirms the advantages of maintaining distinctive topologies to circumnavigate dynamic obstacles.

I. INTRODUCTION

Trajectory planning is a fundamental problem in robotic research and application. The challenges increase with the complexity of the dynamic environment and motion task which favors trajectory planning over path planning as the former explicitly considers dynamic aspects of robot motion. On the other hand, planning trajectories is computational more demanding which limits its close integration with motion control. However, online generated trajectories respond to changes in the environment or perturbations of the robot motion. For an overview on established approaches to path and trajectory planning the reader is referred to [1].

Delsart et al. extend the original *elastic band* approach by Quinlan et al. [2] to the online deformation of trajectories rather than paths [3]. Rösmann et al. [4] and [5] present a further extension to the *elastic band*, so called *Timed Elastic Band* (TEB), in terms of optimization based trajectory deformation with a sparse structure and the explicit incorporation of time dependent states. The optimization allows for but is not limited to kinodynamic constraints and nonholonomic kinematics. Lau et al. [6] and Sprunk et al. [7] optimize trajectories represented by splines. Ratliff et al. [8] present an online planning algorithm that relies on a covariant gradient descent method.

The majority of optimization based online planning approaches locally refine an initial coarse path recursively. Such a local planner continuously deforms the trajectory and is therefore unable to transform it across obstacles. [9] partially overcomes these limitations by applying a stochastic descent method. However, the approach requires extensive sampling of trajectories in order to estimate the true gradient w.r.t. the global optimum. Kuderer et al. develop a two stage approach that generates a set of alternative, topologically distinctive trajectories subject to local optimization [10]. These alternatives often coincide with local minima of the cost function and are extracted from a modified *Voronoi diagram*. It groups paths that belong to the same *equivalence class* defined by an equivalence relation based on the circumvolution of obstacles. This paper pursues a related approach for filtering distinctive candidate trajectories in the context of online trajectory planning. It mainly differs from [10] w.r.t. to the path generation and sampling strategy and the underlying equivalence relation.

The idea of utilizing topologically distinctive paths and trajectories for planning and navigation is not novel. However, past approaches mainly focus on global offline path and trajectory planning. [11] and [12] present probabilistic roadmap (PRM) methods that both operate with albeit different equivalence relations to group trajectories into equivalence classes. 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 and is thus suited for online trajectory optimization. Knepper et al. [13] propose a local planner based on path sampling. The equivalence relation utilizes the *Hausdorff* metric and ensures continuity between subsequent iterations. The planner performs a discrete path selection rather than deforming continuous trajectories. Bhattacharya et al. [14] present an equivalence relation from the domain of complex analysis. The closed form solution motivates its usage in our approach for trajectory filtering. A related graph free sampling based approach constructs a multi-scale approximation that operates with filtrations of simplicial complexes [15].

This contribution presents an integrated online approach that combines the exploration, filtering and simultaneous optimization of a few admissible topologically distinctive trajectories. The approach maintains a set of candidate trajectories to accomplish their coherence among subsequent iterations of optimization. The locally optimal trajectory of an equivalence class is preserved to allow a hot start from previous solutions. Way point sampling advances not only over space but also time which significantly reduces the number of samples.

Section II introduces the integrated planning approach.

978-1-4673-9163-4/15/$31.00

Section III presents and evaluates results obtained in a realistic simulation of a differential drive mobile robot, its sensors and environment. The TEB approach borrowed from previous work is utilized for the underlying trajectory optimization. The integrated planning approach is compared with the original TEB and the *Dynamic Window Approach* (DWA) [16]. Section IV summarizes the results and provides an outlook on further work.

II. ONLINE TRAJECTORY PLANNING

A. Problem Definition

The general trajectory optimization problem is concerned with the computation of the optimal controls $\mathbf{u}^* \in \mathbb{R}^m$ and is defined by:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} J(\mathbf{x}), \quad \mathbf{x} \in \mathcal{X} \tag{1}$$

$$\mathbf{u}^* = \mathbf{k}(\mathbf{x}^*) \tag{2}$$

in which $J(\mathbf{x})$ denotes a nonlinear cost function depending on the optimization parameter $\mathbf{x}$. The domain $\mathcal{X} \subseteq \mathbb{R}^n$ is often referred to as search space described by a set of equality and inequality constraints. Typically, for unconstrained problems $\mathcal{X}$ represents the euclidean space $\mathcal{X} = \mathbb{R}^n$. The function $\mathbf{k} : \mathcal{X} \rightarrow \mathbb{R}^m$ denotes the mapping from the optimization parameters $\mathbf{x}$ to the controls $\mathbf{u}$. The choice of $\mathbf{k}(\mathbf{x})$ determines which of the following three most common variants is realized: 1) The sequential approach minimizes J merely w.r.t. the controls such that $\mathbf{u} = \mathbf{I}\mathbf{x}$, the robot states are implicitly included through forward simulations or substitutions. 2) The simultaneous approach minimizes J w.r.t. to both the controls and robot states (typically resulting in a sparse structure). 3) The optimization parameter $\mathbf{x}$ is identical with the robot states, typically defined by a sequence of robot poses augmented with time. In this case $\mathbf{u}$ is afterwards inferred from pairs of subsequent states by inverse kinematics or dynamics $\mathbf{u}^* = \mathbf{k}(\mathbf{x}^*)$.

Online optimization of (1) is preferred in order to cope with dynamic environments and perturbations. Planning of the optimal trajectory is closely integrated with state feedback motion control. At each sampling interval, the first entry of the optimal control sequence controls the robot. In subsequent sampling intervals the trajectory optimization is repeated w.r.t. to updated robot states and perceptions. This concept is also known as *Model Predictive Control*. Efficiently solving (1) is usually performed with local optimization techniques such that $J(\mathbf{x}^*) \leq J(\mathbf{x})$ for all $\mathbf{x}$ within a vicinity of $\mathbf{x}^*$. Under the assumption that $J(\mathbf{x})$ is non-convex, the resulting optimal solution $\mathbf{x}^*$ strongly depends on the initial parameter $\mathbf{x}_0$.

In mobile robot navigation $J(\mathbf{x})$ is usually composed of terms that smooth and contract the trajectory to either minimize the time or distance between the current and goal pose. Additional terms guarantee a minimal separation from obstacles. In case of an obstacle free environment, the cost function $J(\mathbf{x})$ for the shortest or fastest trajectory is convex or at least possesses a unique minimum $\mathbf{x}^*$. In case of obstacles, $J(\mathbf{x})$ contains terms that depend on the minimal euclidean distance between the trajectory and obstacles. These cost terms are concave as they contain a maximum centered at the obstacle. The cost function $J(\mathbf{x})$ that results from the superposition of convex and concave terms is non-convex such that under local deformations trajectories do not transit to the opposite side of an obstacle. In the worst case a moving obstacle progressively elongates the trajectory along its direction of motion without the ability to rather pass the obstacle on its opposite side. In case the robot encounters two obstacles approaching from opposite directions the separation constraint is violated and the robot is forced to stop in order to avoid a collision. This failure is related to the Freezing Robot Problem in stochastic motion planning [17]. Even in case of feasible locally optimal trajectories a global minimum of (1).

Hot starting utilizes the previously planned trajectory to initialize the next iteration of the solver. However, hot starting is unable to respond to the above mentioned dynamic scenarios. Maintaining and simultaneous optimization of multiple topological distinctive plans conciliates the advantages of hot starts with global optimality.

B. Homotopy Classes and Homology Classes

The following sections address the problem of finding the global minimum of (1). The underlying planning problem is restricted to trajectories of a planar mobile robot. It makes no particular assumptions on the representation or shape of obstacles. The path in terms of a sequence of robot positions, ignoring its orientation, is defined as $\tau = \{\mathbf{z}_k \in \mathbb{R}^2 \mid k = 1, 2, \ldots, N\}$. $\mathbf{x} = \mathbf{h}(\tau)$ denotes the mapping from the (initial) path to the optimization parameter $\mathbf{x}$. Notice, trajectories may contain poses as elements of the $SE(2)$ Lie subgroup and may include temporal information. However, the following discussion is only concerned with the robot position.

Our approach is based on the theory of homotopy classes. According to [18] homotopic trajectories are defined by:

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

Based on the above definition 1 local optimization methods for dynamic problems with hot-starting establish a homotopy between the solution trajectories τ^* and τ of subsequent optimization steps. Let $\mathcal{A} = \{\mathbf{x}_\epsilon \in \mathcal{X} \mid J(\mathbf{x}^*) \leq J(\mathbf{x}_\epsilon), \mathbf{x}_\epsilon = \mathbf{x}^* + \boldsymbol{\epsilon}, \boldsymbol{\epsilon} > \mathbf{0}\}$ denote the (attractive) vicinity of a local minimum $\mathbf{x}^*$. Figuratively, all trajectories τ_ϵ with $\mathbf{x}_\epsilon = \mathbf{h}(\tau_\epsilon) \in \mathcal{A}$ that converge to the same local minimum are homotopic and therefore relate to the same homotopy class. It is reasonable to assume that the cost terms for obstacle clearance are the only cause for the existence of multiple local minima. In that case a single representative $\mathbf{x}_\epsilon^{(i)} \in \mathcal{A}^{(i)}$ of each homotopy class $\mathcal{A}^{(i)}$ provides a valid initial solution of problem (1) and is sufficient to identify the local minimum $\mathbf{x}^{*(i)}$ of the i-th homotopy class.

The closed form generic computation of homotopy classes is difficult. [18] suggests substituting the homotopy with homology classes as they are easier to compute. A homology class defines a set of homologous trajectories in which elements are homologous to each other.

Definition 2 (Homologous trajectories): Two trajectories τ_1 and τ_2 connecting the same start and goal points $\mathbf{z}_s$ and

$\mathbf{z}_g$ respectively, are homologous if and only if $\tau_1 \sqcup -\tau_2$ forms the complete boundary of a 2D manifold embedded in $\mathbb{R}^2$.

Homotopy implies homology, but the reverse implication does not hold. However, for most practicable mobile robot planning scenarios, both definitions can be considered as equivalent. See [18] for further details on the distinction between homology and homotopy including examples in which both definitions differ from each other.

[18] and [14] 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 trajectories of the same homology class. Without loss of generality, a 2D position $\mathbf{z}_k$ of a trajectory τ is represented in the complex domain by $z_k = x_k + \mathrm{i}y_k \in \mathbb{C}$ (boldface omitted). The *H*-signature is originally defined for continuous trajectories between start and goal points z_s and z_g respectively. Let $\tilde{\tau}(t)$ denote a continuous trajectory such that $\tilde{\tau}(t=0) = z_s$ and $\tilde{\tau}(t=T) = z_g$, the complex homology invariant is defined by:

$$\mathcal{H}(\tilde{\tau}) = \int_{\tilde{\tau}} \mathcal{F}(z)dz \tag{3}$$

Equation (3) follows immediately from the definition of the Cauchy's integral formula [14]. Obviously, $\mathcal{F}(z)$ depends on the obstacle positions according to the remarks of section II-A. [14] suggest the following function $\mathcal{F}(z)$, referred to as obstacle marker function:

$$\mathcal{F}(z) = \frac{f_0(z)}{(z-\xi_1)(z-\xi_2)\cdots(z-\xi_R)} \tag{4}$$

$\xi_l \in \mathcal{O}_l, \forall l = 1, 2, \ldots, R$ denote representative points of R obstacles. f_0 denotes an arbitrary analytic function over $\mathbb{C}$. Each representative point is arbitrarily chosen from the interior of the obstacle region respectively obstacle shape $\mathcal{O}_l \subset \mathbb{C}$. A particular and suitable choice f_0 is suggested in [14].

In order to calculate the *H*-signature of the discrete trajectory τ, the analytic solution of (3) for line segments is utilized. [14] derives an analytic expression for a line segment connecting two points z_k and z_{k+1}:

$$\mathcal{H}_s(z_k, z_{k+1}) = \sum_{l=1}^{R} A_l(\ln(z_{k+1}-\xi_l) - \ln(z_k - \xi_l)) \tag{5}$$

with $A_l = \frac{f_0(\xi_l)}{\prod_{j=1, j\neq 1}^{R}(\xi_l - \xi_j)}$. The *H*-signature of a discrete trajectory (composed of line segments) is calculated by:

$$\mathcal{H}(\tau) = \sum_{k=1}^{N-1} \mathcal{H}_s(z_k, z_{k+1}) \tag{6}$$

Note that the actual implementation of (5) requires the theory of complex logarithm. [18] suggest choosing 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):

$$\mathcal{H}_s(z_k, z_{k+1}) = \sum_{l=1}^{R} A_l \Big[\ln(|z_{k+1}-\xi_l|) - \ln(|z_k - \xi_l|) + \ldots + \mathrm{i}\min_{\alpha\in\mathbb{Z}}\Big(|\arg(z_{k+1}-\xi_l) - \arg(z_k-\xi_l) + 2\alpha\pi|\Big)\Big] \tag{7}$$

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

C. Discovery of Homology Classes

Based on the homology invariant proposed in the previous section, an algorithm for exploring relevant homology classes is developed. [14] 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 (1) 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 (1) still requires a substantial computational effort. Therefore, the proposed approach gathers coarse, collision free candidate trajectories which waypoints are located in forward direction. The *H*-signature is applied as a filter to eliminate all but one trajectory for each homology class.

Given the robots current position z_s, goal z_g (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 $\mathcal{V} = \{z_s, \zeta_i, z_g \in \mathbb{C} \mid \forall \zeta_i \notin \mathcal{O}, i = 1, 2, \ldots, I\}$. $\zeta_i \in \mathbb{C}$ are waypoint samples that later may become part of the trajectory. The first exploration stage seeds waypoints between z_s and z_g. For the sake of simplicity, it is assumed that obstacle shapes are circular of limited radius. A pair of candidate waypoints is placed to the left and right of each obstacle orthogonal to the line spanned by z_s and z_g. This initialization suffices to generate the subset of distinctive admissible paths. The configuration of obstacles and associated waypoint candidates is illustrated for an example environment in Fig. 1. $\xi_l \in \mathcal{O}_l$ denotes the representative point of each obstacle marked with a cross. The circle at ξ_l denotes the obstacle region $\mathcal{O}_l$.

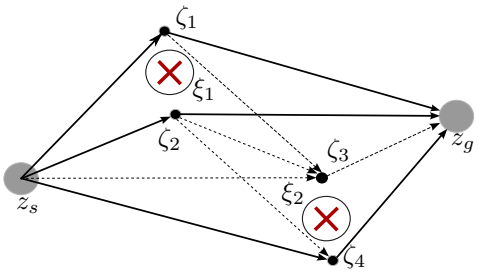


Fig. 1: Example of an exploration graph

The set of edges $\mathcal{E}$ is constructed from the waypoint seeds. An edge connects a pair of vertices $v_1 \in \mathcal{V}$ and $v_2 \in \mathcal{V}$ if the following conditions hold:

- Direction is forward oriented with respect to the goal heading, such that $\frac{\Re[(v_2-v_1)(z_g-z_s)]}{|v_2-v_1||z_g-z_s|} > \theta$ with $\theta \in [0, 1]$.
- Line segment $\mathcal{L} = \{v_1 + t(v_2 - v_1) \mid \forall t \in [0, 1]\}$ 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. Edges for the example above are shown in Fig. 1 as arrows.

Based on the generated graph $\mathcal{G}$, its simple paths between z_s and z_g are extracted by a depth-first search augmented by a visited list. The *H*-signature for each path is calculated according to (6) 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. The filtered graph for the example above is shown in Fig. 1 with bold solid lines.

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 B contains those vertices already visited such that after reaching the goal, B consists of the complete path candidate from z_s to z_g. 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 B. The coarse path defined by $\{z_s, \zeta_i, z_g\}$ is subsampled and the orientation parts 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 $\mathbf{x}$ is stored for subsequent hot-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; B - reference to an ordered visited list containing only z_s; z_g - goal vertex; T - reference to the trajectory set; H - reference to the set of *H*-signatures
Output: Updated set of trajectories and *H*-signatures

```
1: function DEPTHFIRST(G, B, z_g, T, H)
2:     if max. size of T is reached then
3:         return
4:     b ← B.back()                          ▷ Get last visited vertex
5:     for each adjacent vertex v of vertex b in G do
6:         if v ∈ B then                      ▷ Already visited
7:             continue
8:         if v == z_g then                   ▷ Goal reached
9:             B.append(v)                    ▷ Finalize trajectory
10:            h ← CALCHSIG(B)                ▷ See Eq. (7)
11:            if h ∉ H then
12:                x ← INITTRAJECTORY(B)      ▷ x = h(τ)
13:                T.append(x)                ▷ Save complete trajectory
14:                H.append(h)                ▷ Store H-signature
15:            break
16:    for each adjacent vertex v of vertex b in G do
17:        if v ∈ B or v == z_g then          ▷ Already visited or goal reached
18:            continue
19:        B.append(v)
20:        DEPTHFIRST(G, B, z_g, T, H)        ▷ Recursion step
21:        B.pop(v)
    return
```

For arbitrary obstacle shapes $\mathcal{O}_l$, the above simply sampling strategy for candidate waypoints to the left and right side is not applicable. Instead, waypoint sampling follows the probabilistic roadmaps (PRM) approach [19]. Waypoints ζ_l are sampled uniformly from a predefined region $\mathcal{S} \subseteq \mathbb{C}$. The remaining steps are identical to the algorithm outlined above. Obviously, the computation time increases exponentially with an increasing number of samples. However, our experience suggest that a few samples $(10 - 20)$ are sufficient to discover the admissible paths. The sampling is repeated at each iteration such that novel homology classes might still be discovered at a later stage. The proposed approach is based on the assumption, that the global minimum (1) is among a subset of few admissible trajectories that do not detour substantially from the straight line connection between start and goal. It is highly likely that the globally optimal trajectory is among the first two to five distinctive homology classes. Fig. 2a and 2b show examples for the discovery of homology classes (30 waypoints and line of sight parameter for edges $\theta = 0.4$). The mean number of subsequent iterations required to discover 4 and 7 homology classes as well as the mean computation time for varying number of samples are shown in table I (3.4 Ghz Intel i7).

TABLE I: Exploration and runtime analysis of the example shown in Fig. 2. All values are mean values that are subject to 100 repetitions.

No. samples	Iter. up to 4$^{\text{th}}$ class	Iter. up to 7$^{\text{th}}$ class	CPU time
3	12.2	–	0.05 ms
8	3.8	13.7	0.2 ms
15	1.7	7.0	1.1 ms
25	1	2.4	12 ms

D. Trajectory Discovery & Online Optimization

The following section describes the integration of the homology class discovery with the trajectory optimization inside the robot control feedback loop. Algorithm 2 lists the principal planning step invoked at each control cycle of the mobile robot. The first invocation ($T = \emptyset$) creates a new graph (line 8) according to section II-C by seeding random samples in a region of interest (typically a rectangular or a semicircle connecting z_s and z_g). Afterwards the modified depth first search is invoked according to Alg. 1 that discovers the distinctive homology classes and initializes a single representative trajectory for each class in T. The trajectories $\forall \mathbf{x} \in T$ (line 10-11) are optimized simultaneously according to (1). The globally optimal trajectory $\mathbf{x}^*$ is selected according to the cost function: $\mathbf{x}^* = \arg\min_{\mathbf{x} \in T} J(\mathbf{x})$ and the (sub) optimal control $\mathbf{u}^* = \mathbf{k}(\mathbf{x}^*)$ is returned.

In subsequent iterations of the control cycle, T and H already contain candidate trajectories optimized in previous iterations. The current start z_0 and goal z_g are updated according to the novel robot state and perceptions. The current

Algorithm 2 Perform a single local planning step

Input: z_s - start point; z_g - goal point; T - reference to the trajectory set; H - reference to the set of H-signatures; $\mathcal{O}$ - set of obstacles
Output: (Sub-)optimal control $\mathbf{u}^*$

function PLANTRAJECTORIES($z_s, z_g, T, H, \mathcal{O}$)
  if T is not empty **then** ▷ T.size() == H.size()
    $T \leftarrow$ UPDATETRAJECTORIES(T, z_s, z_g)
    $(T, H) \leftarrow$ DELETEDETOURS($T, z_s, z_g, \mathcal{O}$)
    $H \leftarrow$ RENEWHC($T, H, \mathcal{O}$)
  $B \leftarrow$ allocate empty visited list
  B.append(z_s)
  $\mathcal{G} \leftarrow$ CREATEGRAPH($\mathcal{O}$)
  DEPTHFIRST($\mathcal{G}, B, z, T, H$)
  for each trajectory $x \in T$ **do** ▷ parallelizable
    $x \leftarrow$ CALLOPTIMIZER($x, \mathcal{O}$) ▷ solve Eq. (1)
  $x^* \leftarrow$ SELECTGLOBALOPTIMALTRAJECTORY($T, \mathcal{O}$)
return corresponding control $\mathbf{u}^*$ according to Eq. (2)

set of trajectories is validated for admissibility. In particular the edge conditions of section II-C are verified. In case of a violation the trajectory and the corresponding H-signature are eliminated. The current H-signatures are updated in case the obstacle configuration changes.

III. Experimental Results and Examples

The proposed planning algorithm 2 is integrated with the Timed Elastic Band (TEB) sparse online trajectory planning approach for non-holonomic mobile robots [4], [5]. The integrated architecture and functions are introduced and two different mobile robot scenarios are presented and evaluated. Detailing the complete TEB approach is beyond the scope of this paper. Fig. 2 demonstrates the proposed algorithm in combination with the TEB optimization for two subsequent online planning steps.

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. Algorithms run on a PC with a 3.4 Ghz Intel i7 CPU. The current position of obstacles (humans and walls) is tracked with a laser scanner. Translational and angular velocities are limited to $v_{max} = 0.4\,\frac{\text{m}}{\text{s}}$ and $\omega_{max} = 0.3\,\frac{\text{rad}}{\text{s}}$ respectively. The desired minimal separation from obstacles is 0.5 m.

The proposed method is compared with the classical TEB approach (without homology class exploration) and the dynamic window approach (DWA) [16]. The DWA is a state-of-the-art method for mobile robot navigation. Trajectories are sampled at each control cycle from a velocity search space restricted by a feasible set of velocities. The least cost trajectory controls the robot. Due to the sampling based strategy the DWA easily detects multiple local minima in contrast to continuous optimization techniques with hot-starts. The DWA forward simulation time for each sample is 6 s in order to obtain a trajectory length comparable to the other methods. The implementation of Both TEB approaches[1] are implemented in C++ and ROS [20]. The DWA is already available in ROS.

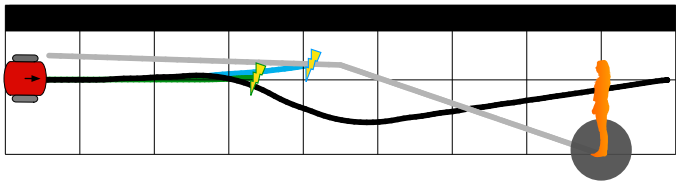


Fig. 3: Scenario 1: Traces of the robot's odometry and the human's movement (grey line). Legend: black (proposed method); blue (TEB without homology class exploration), green (dynamic window approach). Cell size: 1 m^2.

The first scenario evaluates the closed-loop behavior of the planner in case the optimal trajectories transits from one to the opposite side of a moving obstacle. In this scenario a human ignoring the approaching robot moves towards a wall. The initially optimal trajectory passes between the wall and the human and becomes inadmissible as the human together with the wall forms a barrier. Fig. 3 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 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 sides. The transition is initiated too late such a collision can not 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.

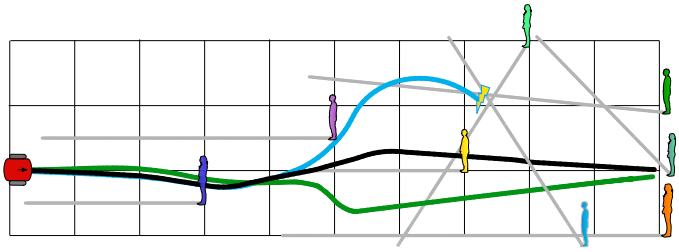
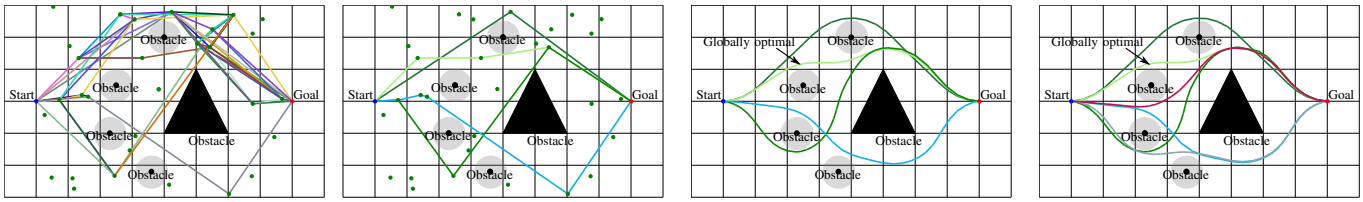


Fig. 4: Scenario 2: Traces of the robot's odometry and the humans' movements (grey lines). Legend: black (proposed method); blue (TEB without homology class exploration), green (dynamic window approach). Cell size: 1 m^2.

The second scenario demonstrates the online planning in an open space environment with eight dynamic ignorant obstacles which traces are shown in Fig. 4. The classic TEB collides halfway through the transition 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. Fig. 5 shows the corresponding translational velocity profile of the robot for the different planners. The DWA often reduces the robot speed in order to avoid imminent collisions. In contrast, the ability of our approach to

[1] Sourcecode of both algorithms and a video are available online [21].



(a) (b) (c) (d)

Fig. 2: Actual application of Alg. 2. Planning is performed using the TEB approach. (a) Simple paths found without homology filtering (line of sight restricted to $\pi/3$). (b) Simple paths found with homology filtering activated. (c) Optimization result (TEB). (d) Subsequent call of Alg. 2: existing homology classes are kept, but two new ones have been explored and added.

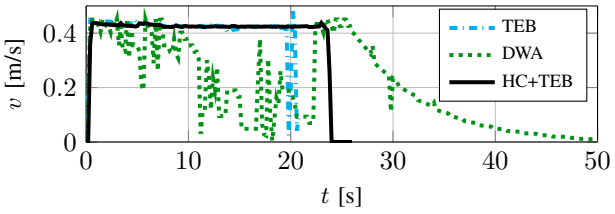


Fig. 5: Transl. velocity profile of the robot in scenario 2

consider alternative future evolutions of a scenario is able to navigate the robot at maximum speed towards the goal. Note, the DWA implementation reduces speed while approaching the goal, therefore the actual travel time is incomparable.

IV. CONCLUSIONS AND FUTURE WORK

The integrated approach of homology class exploration and trajectory optimization for closed-loop planning and control offers the advantage to account for alternative evolutions of scenarios of dynamic obstacles. The comparison with two planners that do not consider distinctive topological trajectories in simulation illustrates the benefits of multiple trajectory planning.

Future work is concerned with extending the search for alternative topologies to not only spatial but also temporal domain. 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. M. LaValle, *Planning Algorithms*. New York, USA: Cambridge University Press, 2006.

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

[3] V. Delsart and T. Fraichard, "Reactive Trajectory Deformation to Navigate Dynamic Environments," in *European Robotics Symposium*, Czech Republic, 2008.

[4] C. Rösmann, W. Feiten, T. Wösch, F. Hoffmann, and T. Bertram, "Trajectory modification considering dynamic constraints of autonomous robots," in *7th German Conference on Robotics*, May 2012, pp. 74–79.

[5] C. Rösmann, W. Feiten, T. Wösch, F. Hoffmann, and T. Bertram, "Efficient trajectory optimization using a sparse model," in *6th European Conference on Mobile Robots (ECMR)*, September 2013.

[6] B. Lau, C. Sprunk, and 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, and W. Burgard, "Improved non-linear spline fitting for teaching trajectories to mobile robots," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, St. Paul, MN, USA, Mai 2012, pp. 2068–2073.

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

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

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

[11] E. Schmitzberger, J. Bouchet, M. Dufaut, D. Wolf, and 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.

[12] L. Jaillet and T. Simeon, "Path deformation roadmaps: Compact graphs with useful cycles for motion planning," *The International Journal of Robotics Research*, vol. 27, no. 11-12, pp. 1175–1188, 2008.

[13] R. A. Knepper, S. Srinivasa, and M. T. Mason , "Toward a deeper understanding of motion alternatives via an equivalence relation on local paths," *International Journal of Robotics Research*, vol. 31, no. 2, pp. 168–187, March 2012.

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

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

[16] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, March 1997.

[17] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010.

[18] S. Bhattacharya, "Topological and geometric techniques in graph-search based robot planning," Ph.D. dissertation, University of Pennsylvania, January 2012.

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

[20] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.

[21] C. Rösmann. (2015, May). teb_local_planner ROS Package [Online]. Available: http://wiki.ros.org/teb_local_planner