Kinodynamic Trajectory Optimization and Control for Car-Like Robots

Christoph Rösmann, Frank Hoffmann and Torsten Bertram

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 2017

Kinodynamic Trajectory Optimization and Control for Car-Like Robots

Christoph Rösmann[1], Frank Hoffmann[1] and Torsten Bertram[1]

***Abstract*— This paper presents a novel generic formulation of Timed-Elastic-Bands for efficient online motion planning of car-like robots. The planning problem is defined in terms of a finite-dimensional and sparse optimization problem subject to the robots kinodynamic constraints and obstacle avoidance. Control actions are implicitly included in the optimized trajectory. Reliable navigation in dynamic environments is accomplished by augmenting the inner optimization loop with state feedback. The predictive control scheme is real-time capable and responds to obstacles within the robot's perceptual field. Navigation in large and complex environments is achieved in a pure pursuit fashion by requesting intermediate goals from a global planner. Requirements on the initial global path are fairly mild, compliance with the robot kinematics is not required. A comparative analysis with Reeds and Shepp curves and investigation of prototypical car maneuvers illustrate the advantages of the approach.**

I. INTRODUCTION

In the context of mobile robot navigation trajectory planning and control constitutes a fundamental task in applications such as service robotics and autonomous transportation systems. Online planning is preferred over offline solutions since the former integrate planning with state feedback and respond to dynamic environments and perturbations at runtime. The *elastic band* (EB) approach is well known for online path deformation [1]. Predefined internal forces contract the path while external forces maintain a separation from obstacles. However, conventional path planning does not explicitly incorporate temporal and (kino-)dynamic constraints. An extension of the EB approach to online deformation of trajectories rather than paths is presented in [2]. Discrete trajectory waypoints are repelled from obstacles. Their connectedness w.r.t. a dynamic motion model is restored afterward. Delsart et al. combine both stages into a single operation [3]. However, online trajectory optimization based approaches are often limited by the computational burden to converge to a feasible and optimal solution under real-time constraints. Sampling-based approaches, such as the dynamic window approach (DWA) [4], address this issue of computational efficiency. Simulated trajectories are sampled and repeatedly scored from a velocity search space restricted by a set of feasible velocities w.r.t. the remaining distance to the goal, the velocity and separation from obstacles. Lau et al. [5] optimize trajectories represented by splines according to kinodynamic constraints of the robot.

Most planners only consider non-holonomic constraints of a differential-drive robot, which however does not capture the minimum turning radius of a car-like robot. An overview about feedback control techniques is provided in [6]. Lamiraux et al. present a path deformation based on potential gradients [7]. The approach rests upon smooth path representations and velocity limits are not covered explicitly. This substantially constrains its direct application to parking maneuvers. Vendittelli et al. provide collision free paths for point-shaped robots based on the well-known Reeds and Shepp (RS) curves [8]. RS curves provide analytic solutions of the minimum-time optimal control problem w.r.t. the kinematic model [9]. Also, the classical EB has been adopted to car-like robots [10]. Hereby, consecutive path points on which internal and external forces act, are connected by RS curves. Furthermore, Bézier polynomials are utilized for smoother transitions. However, the deformation is not subject to arbitrary velocity and acceleration limits so that time scaling is required to determine a feasible albeit non-optimal trajectory. Gu et al. [11] present a multi-stage planning approach that utilizes an optimization-free EB to generate paths followed by a speed planning stage. In the context of direct trajectory optimization, the DWA is extended to support car-like robots in [12] as it restricts the search space of rotational velocities to the set of feasible solutions. However, the assumption of constant velocities in prediction prohibits motion reversals required to navigate in confined spaces. A non-realtime-capable, but complete optimal control formulation that considers kinodynamic constraints and traveling time is presented in [13]. Other approaches rest upon search based strategies, e.g. using cell decomposition in [14] or by applying smoothing filters to well-known (global) *rapidly exploring random trees* [15].

The *Timed-Elastic-Band* (TEB) approach provides a real-time capable online trajectory planner for differential-drive robots [16], [17]. It is inspired by the idea of the EB method but reformulates planning of trajectory and controls as a sparse optimization problem. The TEB mimics a predictive controller and efficiently optimizes the trajectory w.r.t. (kino-)dynamic constraints while explicitly incorporating temporal information to achieve a time-optimal solution.

This contribution presents a novel and more generic formulation of the original TEB approach which is among others the support of motion reversals and generic obstacle representations. Furthermore, it is extended to generate trajectories and control actions for car-like robots during runtime. The TEB is intended as a local planner in a hierarchical navigation architecture. The TEB planning horizon coincides with the robots perceptual range, the remote path based on a static map is entrusted to the global planner. The TEB approach is available as open-source C++ code and integrated into ROS [18] and provides the first local planner for the

[1]Faculty of Electrical Engineering & Information Technology, Institute of Control Theory and Systems Engineering, TU Dortmund, 44227 Dortmund, Germany {forename.surename}@tu-dortmund.de

navigation stack in ROS that supports car-like robots.

The paper is organized as follows: Section II details the kinematic model and reformulates the TEB approach for car-like robots. Section III presents simulations and experiments. It shows that the TEB solution coincides with RS curves in the obstacle-free case. It analyzes parallel parking scenarios of a car like robot. Section IV summarizes the results.

II. THEORETICAL BACKGROUND

A. Kinematic Model of a Car-Like Robot

This section recapitulates the kinematic model of a car-like robot shown in Fig. 1a w.r.t. the world frame (x_w, y_w). The car exhibits the planar motion of a rigid body in which $\mathbf{s} = [x, y, \beta]^\intercal \in \mathbb{R}^2 \times S^1$ denotes the current vehicle configuration. The base frame is located at the center of the rear axle and its x-axis is aligned with the main axis of the robot. The steering angle is limited to $\phi \in (-\phi_{\max}, \phi_{\max})$ with $\phi_{\max} \in (0, \frac{\pi}{2})$. The term $v \in \mathbb{R}$ denotes the signed translational velocity w.r.t. the robot's x-axis. The motion of the two rear wheels and their translational velocities $v_r \in \mathbb{R}$ and $v_l \in \mathbb{R}$ are decoupled by a differential gear such that $v = 0.5(v_l + v_r)$. $L \in \mathbb{R}^+$ denotes the wheelbase in terms of distance between front and rear axles.

The motion of the car-like robot is described by nonlinear ordinary differential equations [9]:

$$\dot{\mathbf{s}}(t) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\beta}(t) \end{bmatrix} = \begin{bmatrix} v(t)\cos\left(\beta(t)\right) \\ v(t)\sin\left(\beta(t)\right) \\ \frac{v(t)}{L}\tan\left(\phi(t)\right) \end{bmatrix}. \tag{1}$$

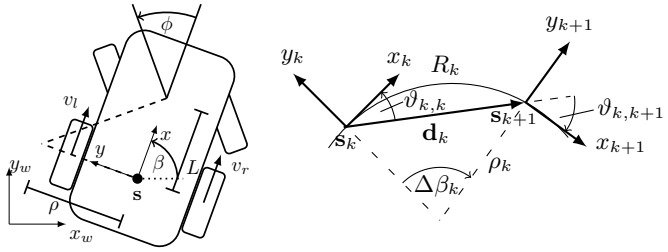
The robot pose at time t is denoted by $\mathbf{s}(t)$ and $\mathbf{u}(t) = [v(t), \phi(t)]^\intercal$ defines the control. $\mathbf{s}_s$ denotes the initial state at time $t = 0\,\mathrm{s}$. We assume that the robot motor controllers accurately track the commanded reference speed and steering angle. Notice, that the model does not consider robot dynamics in terms of forces, torques, and inertia of the car. The actual feasible turning radius at high speeds might be lower than the geometric minimum turning radius. Nevertheless, the kino-dynamic model accounts for bounds on velocities as well as accelerations.

The optimal control input sequence $\mathbf{u}^*(t)$ that minimizes an objective function while it adheres to the car's kinematic (1) is retrieved by means of a (direct) optimal control framework. The underlying optimization problem is discretized and minimized either w.r.t. the controls only (sequential approach) or w.r.t. to both configurations and controls (simultaneous approach). Although the latter formulation leads to a larger solution vector it achieves in general better convergence due to the local and sparse structure of the optimization problem.

The control input $\mathbf{u}(t)$ is directly inferred from the trajectory $\mathbf{s}(t)$ respectively its temporal derivative according to the inverse of (1):

$$\mathbf{u}(t) = \begin{bmatrix} v(t) \\ \phi(t) \end{bmatrix} = \begin{bmatrix} \gamma(\cdot)\sqrt{\dot{x}^2(t) + \dot{y}^2(t)} \\ \tan^{-1}\left(Lv^{-1}(t)\dot{\beta}(t)\right) \end{bmatrix}. \tag{2}$$

Function $\gamma(\cdot)$ retains the direction of velocity and is described in section II-B.1. In case of $v(t) = 0$ in (2), $\phi(t)$ is merely the last known valid steering angle or a constant reference. The closed form solution for $\mathbf{u}(t)$ defines the optimization problem in terms of robot configurations only. This reduces the dimensions of the solution vector while it maintains the sparse structure of the original simultaneous formulation. The elimination of $\mathbf{u}(t)$ allows a purely geometric formulation of the condition imposed on consecutive states for car-like kinematics in (1).



(a) (b)

Fig. 1: Geometry of the car-like robot and poses.

According to (1), the robot performs straight line motions in case $\dot{\beta}(t) = 0$ respectively $\phi(t) = 0$. For a constant steering angle ϕ, the robot describes a circular path with radius $\rho = L/\tan(\phi)$. The maximum steering angle $\phi_{\max}$ causes a minimum turning radius $\rho_{\min}$ of the robot. Let $(\mathbf{s}_k)_{k=1,2,\ldots,n}$ denote a sequence of discretized robot configurations sampled from $\mathbf{s}(t)$ by applying constant control inputs $\mathbf{u}_k$ between two consecutive time instances k.

Straight line and circular motions require two consecutive poses $\mathbf{s}_k$ and $\mathbf{s}_{k+1}$ to be located on a common arc of constant curvature. Let $\vartheta_{k,k}$ denote the angle between configuration $\mathbf{s}_k$ and the direction $\mathbf{d}_k = [x_{k+1} - x_k, y_{k+1} - y_k, 0]^\intercal$ at time k (refer to Fig. 1b). Similarly, $\vartheta_{k,k+1}$ defines the angle between $\mathbf{d}_k$ and configuration $\mathbf{s}_{k+1}$ at time $k+1$. A common arc of constant curvature is obtained if and only if $\vartheta_{k,k}$ and $\vartheta_{k,k+1}$ are equal:

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

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

Constraint (4) is compliant with the robot turning in place ($\mathbf{d}_k = \mathbf{0}$). An auxiliary condition accounts for the minimum turning radius $\phi_{\max}$. According to Fig. 1b, the arc length R_k is defined by $R_k = \rho_k \Delta\beta_k$ with turning radius ρ_k. For non-holonomic robots, the angle $\Delta\beta_k \in S^1$ equals the change of robot orientation in the transition between $\mathbf{s}_k$ and $\mathbf{s}_{k+1}$ and hence is computed according to $\Delta\beta_k = \beta_{k+1} - \beta_k$. Changes of orientation are mapped to the interval $(-\pi, \pi]$. The turning radius ρ_k is described by the trigonometry of circular segments:

$$\rho_k = \frac{\|\mathbf{d}_k\|_2}{\left|2\sin\left(\frac{\Delta\beta_k}{2}\right)\right|} \overset{\Delta\beta_k \ll 1}{\approx} \frac{\|\mathbf{d}_k\|_2}{|\Delta\beta_k|}. \tag{5}$$

The robot motion has to comply with (4) and $\rho_k \geq \rho_{\min}$.

B. Timed-Elastic-Band (TEB) Optimization Problem

This section introduces the TEB optimization formulation. The overall objective is to steer the car-like robot from a start $\mathbf{s}_s$ to a desired goal configuration $\mathbf{s}_f$ in minimum time. The underlying optimization problem is defined in terms of a finite-dimensional parameter vector composed of the discretized sequence of n robot configurations $(\mathbf{s}_k)_{k=1,2,\dots,n}$ from section II-A. The TEB approach incorporates temporal information directly into the optimization problem and thus accounts for the minimization of transition time under kinodynamic constraints. Let $(\Delta T_k)_{k=1,2,\dots,n-1}$ denote a sequence of strictly positive time intervals $\Delta T_k \in \mathbb{R}^+$. Each ΔT_k describes the time required to transit from $\mathbf{s}_k$ to $\mathbf{s}_{k+1}$. The set of parameters subject to optimization is defined by:

$$\mathcal{B} := \{\mathbf{s}_1, \Delta T_1, \mathbf{s}_2, \Delta T_2, \dots, \mathbf{s}_{n-1}, \Delta T_{n-1}, \mathbf{s}_n\} \quad (6)$$

The TEB optimization problem is formulated as a nonlinear program:

$$\min_{\mathcal{B}} \quad \sum_{k=1}^{n-1} \Delta T_k^2 \qquad \text{(NLP)}$$

subject to

$$\begin{aligned}
&\mathbf{s}_1 = \mathbf{s}_c, \quad \mathbf{s}_n = \mathbf{s}_f, \quad 0 \leq \Delta T_k \leq \Delta T_{\text{max}},\\
&\mathbf{h}_k(\mathbf{s}_{k+1}, \mathbf{s}_k) = \mathbf{0}, \quad \tilde{r}_k(\mathbf{s}_{k+1}, \mathbf{s}_k) \geq 0,\\
&\mathbf{o}_k(\mathbf{s}_k) \geq \mathbf{0},\\
&\boldsymbol{\nu}_k(\mathbf{s}_{k+1}, \mathbf{s}_k, \Delta T_k) \geq \mathbf{0}, \qquad (k = 1, 2, \dots, n-1)\\
&\alpha_k(\mathbf{s}_{k+2}, \mathbf{s}_{k+1}, \mathbf{s}_k, \Delta T_{k+1}, \Delta T_k) \geq 0, (k = 2, 3, \dots, n-2)\\
&\alpha_1(\mathbf{s}_2, \mathbf{s}_1, \Delta T_1) \geq 0, \quad \alpha_n(\mathbf{s}_n, \mathbf{s}_{n-1}, \Delta T_{n-1}) \geq 0.
\end{aligned}$$

Initial and final configurations, $\mathbf{s}_1$ and $\mathbf{s}_n$, are tied with the current robot state $\mathbf{s}_c$ obtained from robot localization and goal state $\mathbf{s}_f$. The strictly positive time interval ΔT is bounded from above to ΔT_{max} to accomplish an appropriate discretization of the continuous time motion in terms of the discrete model provided in section II-A. Minimizing $\sum_k \Delta T_k^2$ tends to obtain uniform time intervals (proof follows by Lagrange multiplier method). The equality constraint $\mathbf{h}_k(\cdot)$ satisfies the kinematics equation (4) while inequality $\tilde{r}_k(\cdot)$ imposes a minimum turning radius with $\tilde{r}_k(\cdot) = r_k - \rho_{\text{min}}$. The approximation of the turning radius for small angles in (5) is valid in case of small ΔT_{max}. Notice, that for $\Delta\beta_k \to 0$ the arc length R_k tends to infinity. This case is considered separately in the particular solver implementation. In the constraint approximation from section II-C, $\tilde{r}_k(\cdot)$ is set to 0 for $\Delta\beta_k = 0$ without violating C^1 differentiability.

The remaining equality and inequality constraints reflect kino-dynamic bounds and clearance from obstacles.

1) Limited Velocity and Acceleration: According to section II-A, the translational velocity of configuration $\mathbf{s}_k$ at discrete time step k is denoted as v_k. The robot covers a distance R_k in between $\mathbf{s}_k$ to $\mathbf{s}_{k+1}$ given by (5) and $R_k = \rho_k \Delta\beta_k$. Therefore, its velocity v_k is defined by:

$$v_k = \frac{\rho_k \Delta\beta_k}{\Delta T_k} \gamma(\mathbf{s}_k, \mathbf{s}_{k+1}) \overset{\Delta\beta_k \ll 1}{\approx} \frac{\|\mathbf{d}_k\|_2}{\Delta T_k} \gamma(\mathbf{s}_k, \mathbf{s}_{k+1}). \quad (7)$$

$\gamma(\cdot)$ denotes the sign of the robot velocity. Since the motion of the robot is restricted to sequential linear and circular movements, the projection of the orientation vector $\mathbf{q}_k = [\cos\beta_k, \sin\beta_k, 0]^\intercal$ onto the distance vector $\mathbf{d}_k$ results in

$$\gamma(\mathbf{s}_k, \mathbf{s}_{k+1}) = \text{sign}(\langle \mathbf{q}_k, \mathbf{d}_k \rangle) \approx \frac{\kappa \langle \mathbf{q}_k, \mathbf{d}_k \rangle}{1 + |\kappa \langle \mathbf{q}_k, \mathbf{d}_k \rangle|} \quad (8)$$

with scalar product operator $\langle \cdot, \cdot \rangle$. A sigmoidal approximation of (8) with $\gamma(\cdot) \in [-1, 1]$ is employed as most of the common optimization algorithms are not suited for non-continuous functions. Parameter $\kappa \in \mathbb{R}^+$ denotes a scaling factor that defines the slope (e.g. $\kappa = 10^2$). Both terms cause incorrect velocities at $\langle \mathbf{q}_k, \mathbf{d}_k \rangle = 0$ or $\langle \mathbf{q}_k, \mathbf{d}_k \rangle \approx 0$ respectively. Figuratively, this situation occurs if pose $\mathbf{s}_{k+1}$ is orthogonal to pose $\mathbf{s}_k$. However, such a configuration is not within the feasible set given by constraint (4) and for the special case in which the position parts of both poses coincide it is $\mathbf{d}_k = \mathbf{0} \implies v_k = 0$. Both the actual and the Euclidean distance approximation of the arc segment in (7) are applicable. In the remainder, only the latter, approximative description is investigated.

The angular velocity $\omega_k = \frac{\Delta\beta_k}{\Delta T_k}$ of configuration $\mathbf{s}_k$ is inherently bounded to $|\omega_k| \leq \omega_{\text{max}}$ with $\omega_{\text{max}} = v_{\text{max}} \rho_{\text{min}}^{-1}$. Limits $\pm v_{\text{max}}$ and $\pm\omega_{\text{max}}$ are enforced by inequality constraint $\boldsymbol{\nu}_k(\mathbf{s}_{k+1}, \mathbf{s}_k, \Delta T_k) = [v_{max} - |v_k|, \omega_{max} - |\omega_k|]^\intercal$.

Similarly, translational accelerations a_k are limited to $\pm a_{\text{max}}$. In particular, a_k is defined by finite differences:

$$a_k = \frac{2(v_{k+1} - v_k)}{\Delta T_k + \Delta T_{k+1}}. \quad (9)$$

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 resulting inequality is $\alpha_k(\mathbf{s}_{k+2}, \mathbf{s}_{k+1}, \mathbf{s}_k, \Delta T_{k+1}, \Delta T_k) = a_{max} - |a_k|$. Special cases occur at $k = 1$ and $k = n-1$ for which v_1 and v_{n-1} are substituted by the desired start and final velocities (v_s, ω_s) and (v_f, ω_f) respectively.

2) Clearance from obstacles: The robot trajectory is supposed to reach the goal without any collision with obstacles. An obstacle is modeled as a simply-connected region in $\mathbb{R}^2$ and is denoted as $\mathcal{O}$. In the presence of R obstacles $\mathcal{O}_l$, $l = 1, 2, \dots, R$, the subscript l is added. The distance between configuration $\mathbf{s}_k$ and the obstacle perimeter is quantified in a continuous metric space, e.g. the Euclidean metric. Let $\delta(\mathbf{s}_k, \mathcal{O}) : \mathbb{R}^2 \times S^1 \times \mathcal{O} \to \mathbb{R}$ describes the minimal Euclidean distance between obstacle $\mathcal{O}$ and pose $\mathbf{s}_k$. A minimum separation δ_{min} between all obstacles and configuration $\mathbf{s}_k$ is defined by the inequality constraint:

$$\begin{aligned}
\mathbf{o}_k(\mathbf{s}_k) = &[\delta(\mathbf{s}_k, \mathcal{O}_1), \delta(\mathbf{s}_k, \mathcal{O}_2), \dots, \delta(\mathbf{s}_k, \mathcal{O}_R)]^\intercal\\
&- [\delta_{\text{min}}, \delta_{\text{min}}, \dots, \delta_{\text{min}}]^\intercal.
\end{aligned} \quad (10)$$

The obstacle set $\mathcal{O}$ is updated online to account for dynamic environments. Additionally, prediction models for dynamic obstacles might be included in $\mathbf{o}_k(\cdot)$.

C. Approximative Least-Squares Optimization

Solving nonlinear programs with hard constraints is computational expensive. Therefore, improving the efficiency of fast online solvers has been a focus of research in nonlinear optimization over the last decade. The TEB approach rests

upon unconstrained optimization techniques since they are well studied and mature implementations in open-source packages are widely available.

The exact nonlinear program (NLP) is transformed into an approximative nonlinear squared optimization problem that is solved efficiently as the solver approximates the Hessian by first order derivatives while it exploits the sparsity pattern of the problem. Constraints are incorporated into the objective function as additional penalty terms. In the following the 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:

$$\phi(\mathbf{h}_k, \sigma_h) = \sigma_h \mathbf{h}_k^T \mathbf{I} \mathbf{h}_k = \sigma_h \|\mathbf{h}_k\|_2^2. \tag{11}$$

Inequalities are approximated by weighted one-sided quadratic penalties:

$$\chi(\boldsymbol{\nu}_k, \sigma_\nu) = \sigma_\nu \| \min\{\mathbf{0}, \boldsymbol{\nu}_k\} \|_2^2. \tag{12}$$

The min-operator is applied row-wise. Additional 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. The overall unconstrained optimization problem with objective function $\tilde{V}(\mathcal{B})$ that approximates (NLP) is given by:

$$\mathcal{B}^* = \arg \min_{\mathcal{B} \setminus \{\mathbf{s}_1, \mathbf{s}_n\}} \tilde{V}(\mathcal{B}), \tag{13}$$

$$\begin{aligned} \tilde{V}(\mathcal{B}) = \sum_{k=1}^{n-1} \big[\Delta T_k^2 + \phi(\mathbf{h}_k, \sigma_h) + \chi(\tilde{r}_k, \sigma_r) + \chi(\boldsymbol{\nu}_k, \sigma_\nu) + \\ + \chi(\mathbf{o}_k, \sigma_o) + \chi(\alpha_k, \sigma_\alpha) \big] + \chi(\alpha_n, \sigma_\alpha). \end{aligned} \tag{14}$$

$\mathcal{B}^*$ denotes the optimal solution vector. The theory of quadratic penalties [19] states that $\mathcal{B}^*$ only coincides with the actual minimizer of the nonlinear program (NLP) in case all weights tend towards infinity $\sigma \to \infty$. Unfortunately, large weights introduce ill-conditioning 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 obtainable 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 reasonable point of departure, except for the weight σ_h associated with the equality constraint of the non-holonomic kinematics which is a few orders of magnitude larger (≈ 1000).

For solving (14), the TEB approach utilizes the Levenberg-Marquardt (LM) method due to its proper balance between robustness and efficiency. The graph optimization framework *g2o* [20] implements a highly efficient sparse variant of LM that solves (13). Since terms of (NLP) depend only on a small subset of parameters, the underlying Hessian is sparse and banded.

D. Closed-loop Predictive Control

The TEB approach defines a predictive control strategy in order to account for disturbances, map, and model uncertainties and dynamic environments that guides the robot from its current pose $\mathbf{s}_c$ towards a goal pose $\mathbf{s}_f$.

Problem (NLP) is solved repeatedly at a rate faster than the robot control cycle rate. At each sampling instance, the control action $\mathbf{u}_1$ is inferred from the optimized trajectory $\mathcal{B}^*$ by applying (2) and finite differences (e.g. refer to (7)). The overall algorithm is given by:

1: **procedure** TIMEDELASTICBAND($\mathcal{B}, \mathbf{s}_c, \mathbf{s}_f, \mathcal{O}$)
2: Initialize or update trajectory
3: **for all** Iterations 1 to I_{teb} **do**
4: Adjust length n
5: Update obstacle constraints from set $\mathcal{O}$
6: $\mathcal{B}^* \leftarrow$ SOLVENLP($\mathcal{B}$) ▷ solve (NLP)
7: Check feasibility
8: $\mathbf{u}_1^* \leftarrow$ Map $\mathcal{B}^*$ to $\mathbf{u}_1^*$ ▷ obtain $[v_1, \phi_1]^\intercal$ with (2)
return (sub-) optimal $\mathbf{u}_1^*$

The initial solution $\mathcal{B}$ is obtained from a sparse set of collision free linear path segments generated by a global coarse planner. The initial band is composed of robot states $\mathbf{s_k}$ located on the coarse path equidistant in space and time.

Obviously, the initialization strategy effects which the convergence to competing local optimal solutions and is discussed in the analysis section. The set $\mathcal{B}^*$ is fed back in subsequent sampling intervals to warm-start consecutive optimizations. Since the proposed optimization problem minimizes transition time towards a fixed final configuration, the strategy operates with a shrinking horizon.

An outer optimization loop with I_{teb} iterations maintains the length respectively resolution of the current trajectory.

The number of configurations n is updated depending on the temporal discretizations ΔT_k:

- If $\Delta T_k < \Delta T_{\text{ref}} - \Delta T_{\text{hyst}} \wedge n > n_{\min}$, remove $\mathbf{s}_k$.
- If $\Delta T_k > \Delta T_{\text{ref}} + \Delta T_{\text{hyst}} \wedge n < n_{\max}$, insert new configuration between $\mathbf{s}_k$ and $\mathbf{s}_{k+1}$.

$\Delta T_{ref} \in (0, \Delta T_{\max}]$ denotes the desired temporal resolution and ΔT_{hyst} introduces a hysteresis to avoid oscillations. The number of samples is further limited to $n_{\min} \leq n \leq n_{\max}$. Novel configurations are inserted by linear interpolation between successor and predecessor pose.

The adaption of the length of the horizon for such a shrinking horizon problem offers the major advantage of decoupling path length and overall transition time: The number of poses as well as the number of motion reversals that are required to satisfy (4) is unknown and changes with dynamic obstacles. Insertion and deletion of states at runtime allows the overall path length to contract or extend, while the deformed trajectory still complies with the imposed temporal discretization.

Under some preliminaries, predictive controllers with final equality constraints, such as $\mathbf{s}_n = \mathbf{s}_f$, can be shown to enforce stability as long as the subsequent solutions of the underlying optimization problem are feasible. However, a stability analysis is beyond the scope of this paper. We assume the feasibility of the (approximative) optimization problem which is sufficient for the investigated scenarios. On the other hand, there are obstacle scenarios for which

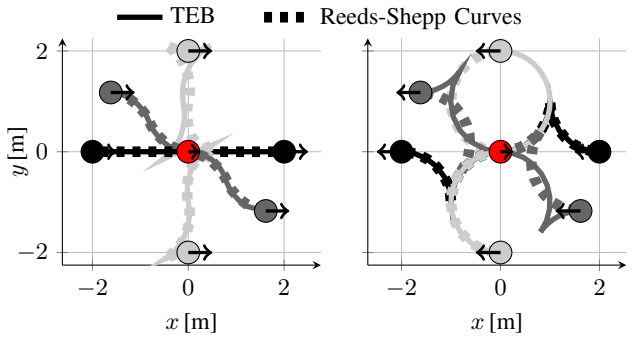


Fig. 2: TEB trajectories and RS curves for different goal configurations, left: $\beta_f = 0\,\text{rad}$, right: $\beta_f = \pi\,\text{rad}$

no solution exist in terms of satisfying all constraints. Hence feasibility is confirmed by an external collision check at each sampling interval before applying the motion command.

III. ANALYSIS AND EXPERIMENTS

This section compares the trajectories generated by the TEB approach in the absence of obstacles with the optimal analytical solution in terms of Reeds and Shepp curves. It investigates the TEB for car like robots in typical parallel parking scenarios. Simulations are performed in C++ (PC: 3,4 GHz Intel i7 CPU).

A. Comparison with Reeds and Shepp Curves

Reeds- and Shepp (RS) curves describe the complete set of paths of a car like robot with kinematics (1) from start to goal in the absence of obstacles [9]. Under limited controls $v(t) \in \{-1, 1\}$ and $\phi(t) \in (-\phi_{\text{max}}, \phi_{\text{max}})$ the shortest path corresponds to the minimum time solution and provides our reference in the following analysis. Since RS curves do not consider dynamics, no TEB acceleration constraints in (NLP) are imposed. The minimum turning radius is set to $\rho_{\text{min}} = 1.0\,\text{m}$. The desired temporal reference is $\Delta T_{\text{ref}} = 0.2\,\text{s}$. Outer loop iterations (I_{teb}) of the TEB planner in section II-D are repeated upon convergence. For the solution of (13) five iterations of Levenberg-Marquardt are performed with the g2o-framework. Fig. 2 illustrates the planned trajectories for a fixed start $\mathbf{s}_s = \mathbf{0}$ and multiple goal poses. The TEB waypoints are initialized along the straight line connecting the start and goal pose with no a-priori reversal of driving direction. In eight of twelve cases, the path obtained from the TEB approach coincides with the optimal RS curve. In the remaining cases, the LM solver converges to a local minimum closest to the initial solution. The four trajectories are symmetric to the RS curve and are of equal length and transition time. In the general case of environments with obstacles global optimality of the trajectory can not be guaranteed as the initial path is locally optimized.

The following analysis investigates the trajectory resolution, traveled distance and computational effort w.r.t. to the lower bound on the turning radius ρ_{min}. The robot reverses its orientation in a cuspidal motion from $\mathbf{s}_s = [2, 0, 0]^\intercal$ to $\mathbf{s}_f = [-2, 0, \pi]^\intercal$. The initial TEB is composed of merely $n = 5$ equidistant waypoints along a straight line from start to goal. Table I reports the traveled distances (path lengths) and computation time upon convergence to the optimal RS curve. During optimization, the number of TEB waypoints n increases to comply with the spatio-temporal resolution. Even for a path length of $25\,\text{m}$ and more than 100 waypoints the LM solver converges within less than $40\,\text{ms}$.

B. Parking Scenarios

In addition to velocity bounds the maximum acceleration of the robot is limited $a_{\text{max}} = 1.5\,\text{m/s}^2$. The obstacle constraint in (10) inflates the Euclidean distance between the rectangular car footprint (length: $0.6\,\text{m}$, width: $0.2\,\text{m}$, $L = 0.4\,\text{m}$) and polygon obstacles with $\delta_{\text{min}} = 0.1\,\text{m}$.

Fig. 3 depicts parallel parking scenario with one or two obstacles and initial biases for forward and reverse parking. In case of a bias for reverse parking 3b and 3d, the initial path (dashed lines) is composed of a straight forward segment that extends beyond the gap succeeded by a back up along a straight line. In case of two obstacles, the TEB converges to distinct local optimal trajectories for either forward 3a or reverse parking 3b. Fig. 4 illustrates velocity and steering angle profile for trajectories 3a and 3b. The time optimal solution exhibits maximum acceleration between the velocity bounds $v_{\text{max}} = 1\,\text{m/s}$ and $-v_{\text{max}}$ and extremal steering angle $\phi_{\text{max}} = \text{atan}^{-1}(L/\rho_{\text{min}}) = 0.38\,\text{rad}$. The forward maneuver 3d is slightly faster than the more common reverse parking option but becomes infeasible for curb-side parking as the car veers beyond the curb to the right. In the case of a single obstacle, forward parking 3c is faster and even the initial trajectory biased towards reverse parking 3d converges to the same global time optimal maneuver. If reverse parking is strictly preferred, the global planner should impose an intermediate way-point behind the gap. For comparison, the paths resulting from the EB approach for car-like robots [10] are depicted in Fig. 3. EB paths in 3a and 3b collide with obstacles. The approach requires internal bubbles to have a radius of at least $\sqrt{2}\rho$ which substantially limits their usage for parking maneuvers. Nevertheless, their minimum size is reduced here, resulting in colliding RS curves. EB paths in 3c and 3d are collision-free but not optimized w.r.t. acceleration limits. Hence, time-scaling leads to non-time-optimal trajectories.

C. Experimental Results

Fig. 5 illustrates the TEB planner on a Lego EV3 car-like robot. Intermediate goals are obtained from a global planner within a perceptual range of $1\,\text{m}$ and hence the environment

TABLE I: Trajectories between $\mathbf{s}_s = [2, 0, 0]^\intercal$ to $\mathbf{s}_f = [-2, 0, \pi]^\intercal$ for different minimal turning radii

ρ_{min} [m]	n	Length [m] / RS Length [m]	CPU Time [ms]
0.75	27	4.86 / 4.85	2.6
1.75	28	5.98 / 5.99	5.9
3.00	49	9.42 / 9.41	21.5
4.25	75	13.35 / 13.33	25.8
6.75	102	21.55 / 21.17	28.1
8.00	104	24.96 / 25.10	35.4

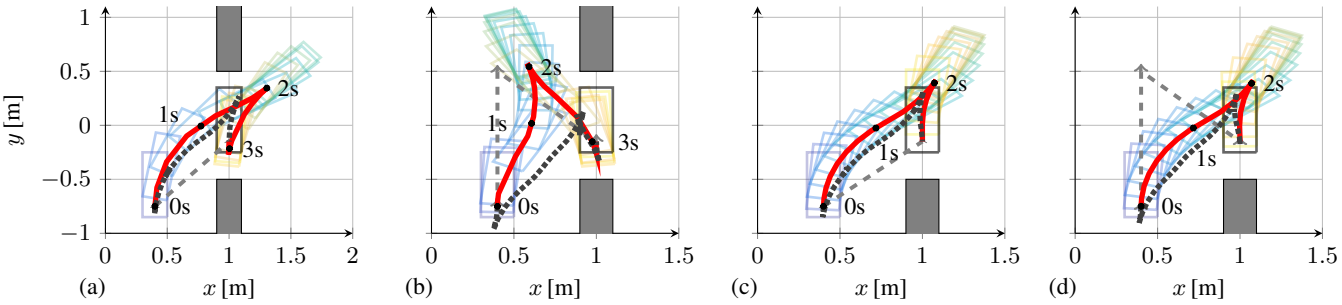


Fig. 3: Parking trajectories with the initial paths (dashed lines) and obstacles (gray), forward (a), backward (b) two obstacles, (c) forward, (d) backward single obstacle. Time stamps are highlighted with black circles. Paths of EB approach [10] are shown with dotted lines (non-time-optimal since acceleration limits are not explicitly incorporated, infeasible in (a) and (b)).

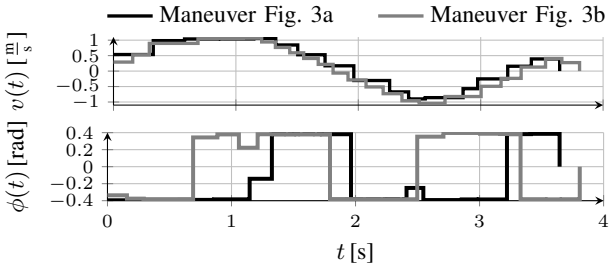


Fig. 4: Velocity and steering angle profile

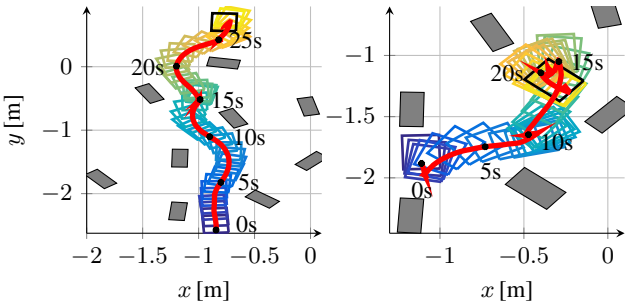


Fig. 5: TEB motion control of Lego EV3 Robot

changes online from a planning point of view. Localization is performed with a motion capture system. Both collision-free closed-loop trajectories guide the robot towards the goal pose despite substantial backlash in steering and model mismatch of the kinematics. In Fig. 5 (right) the TEB reconfigures its initial path at approx. 10 s as it becomes aware of the obstacle on the right only as it enters the robots perceptual range[1] (width: 0.2 m, length: 0.28 m, $\phi_{\max} = 0.6$ rad, $L = 0.19$ m, $v_{\max} = 0.15$ m/s).

IV. CONCLUSIONS

The generic TEB approach for local trajectory planning and control offers substantial advantages for navigation of car-like robots namely computationally efficiency, unsupervised emergence of motion reversals and minimal requirements on the global planner. TEB reliably and efficiently generates feasible minimum-time trajectories that constitute not necessarily a global but in any case local optimum.

[1]Video available online: `https://youtu.be/3FNPSld6Lrg`

REFERENCES

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

[2] H. Kurniawati and T. Fraichard, "From path to trajectory deformation," in *IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, 2007.

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

[4] 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.

[5] B. Lau, C. Sprunk, and W. Burgard, "Kinodynamic motion planning for mobile robots using splines," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, USA, 2009, pp. 2427–2433.

[6] A. D. Luca, G. Oriolo, and C. Samson, *Feedback control of a nonholonomic car-like robot*. Springer, 1998, pp. 171–253.

[7] F. Lamiraux, D. Bonnafous, and O. Lefebvre, "Reactive path deformation for nonholonomic mobile robots," *IEEE Transactions on Robotics*, vol. 20, no. 6, pp. 967–977, 2004.

[8] M. Vendittelli, J. P. Laumond, and C. Nissoux, "Obstacle distance for car-like robots," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 4, pp. 678–691, 1999.

[9] S. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[10] M. Khatib, H. Jaouni, R. Chatila, and J. P. Laumond, "Dynamic path modification for car-like nonholonomic mobile robots," in *Intl. Conf on Robotics and Automation*, vol. 4, 1997, pp. 2920–2925.

[11] T. Gu, J. Atwood, C. Dong, J. M. Dolan, and J.-W. Lee, "Tunable and stable real-time trajectory planning for urban autonomous driving," in *IEEE International Conference on Intelligent Robots and Systems*, 2015, pp. 250–256.

[12] K. Rebai, O. Azouaoui, M. Benmami, and A. Larabi, "Car-like robot navigation at high speed," in *IEEE Intl. Conf. on Robotics and Biomimetics*, 2007, pp. 2053–2057.

[13] S. Gulati, "A framework for characterization and planning of safe, comfortable, and customizable motion of assistive mobile robots." Ph.D. dissertation, 2011.

[14] N. Ghita and M. Kloetzer, "Trajectory planning for a car-like robot by environment abstraction," *Robotics and Autonomous Systems*, vol. 60, no. 4, pp. 609–619, 2012.

[15] W. Xu and J. M. Dolan, "A real-time motion planner with trajectory optimization for autonomous vehicles," in *Proceedings of the Intl. Conf. on Robotics and Automation*, 2012, pp. 2061–2067.

[16] 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*, 2012, pp. 74–79.

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

[18] C. Rösmann, "teb_local_planner ROS Package [Online]," 2015. [Online]. Available: http://wiki.ros.org/teb_local_planner

[19] J. Nocedal and S. J. Wright, *Numerical optimization*, ser. Springer series in operations research. New York: Springer, 1999.

[20] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: A general framework for graph optimization," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2011, pp. 3607–3613.