Timed-Elastic Smooth Curve Optimization for Mobile-Base Motion Planning

Jérémie Deray[1], Bence Magyar[2], Joan Solà[1] and Juan Andrade-Cetto[1]

***Abstract*— This paper proposes the use of piecewise C^n smooth curve for mobile-base motion planning and control, coined *Timed-Elastic Smooth Curve* (TESC) planner. Based on a *Timed-Elastic Band*, the problem is defined so that the trajectory lies on a spline in SE(2) with non-vanishing *n-th* derivatives at every point. Formulated as a multi-objective nonlinear optimization problem, it allows imposing soft constraints such as collision-avoidance, velocity, acceleration and jerk limits, and more. The planning process is realtime-capable allowing the robot to navigate in dynamic complex scenarios.**

The proposed method is compared against the state-of-the-art in various scenarios. Results show that trajectories generated by the TESC planner have smaller average acceleration and are more efficient in terms of total curvature and pseudo-kinetic energy while being produced with more consistency than state-of-the-art planners do.

I. Introduction

The capacity of planning a safe trajectory is of fundamental importance for autonomous mobile-base to enable high-level applications in service robotics or autonomous transportation. Not only should a robot be able to move toward a desired configuration, but it shall do it optimally, and most importantly safely. Such planning must happen online in order to react to the dynamic aspects of the environment. Splines, specifically with non-vanishing *n-th* derivatives allow to define constraints such as velocity, acceleration but also jerk limits, which is desirable *e.g.* for autonomous people transportation vehicles in order to improve comfort and prevent motion sickness [1].

While there exist many approaches solving *A-to-B* type planning, often at the cost of additional requirements, they generally lack in speed, complexity or guarantees.

Initially designed for obstacle avoidance, the method of *Potential Field* (PF) proved to be very valuable for trajectory planning [2]. Two artificial potential fields are superimposed, one repulses from obstacles while the second attracts toward the desired goal. While being simple and efficient, their main drawbacks are the presence of local-minima in which the robot gets stuck and typically failing to find a path between close obstacles.

Rapidly-exploring Random Trees (RRTs)-based methods were first presented in [3], [4]. They originally aimed at solving general nonholonomic and kinodynamic planning problems which earlier randomizer approaches struggled with - *e.g. Probabilistic Road Maps* (PRM) [5]. By subsequently growing a tree randomly from the initial configuration, RRT methods offer good exploration of the search space with relative ease. However they are by nature non-guided therefore non-optimal, and have to be instrumented to provide time guarantees.

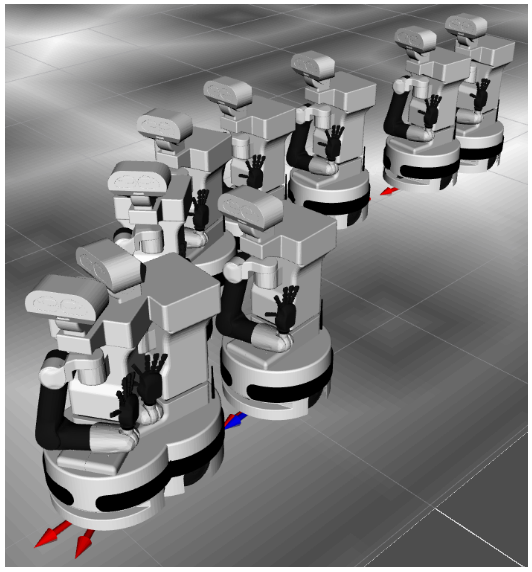


Fig. 1: Decomposition of TIAGo's motion traversing through a scene with obstacles marked by dark regions on the floor

Differential geometry on manifold for trajectory optimization has often been used in robotics, especially on Lie groups, with *e.g.* a smooth rigid-body motion on SE(3) [6] presented as early as twenty years ago. Most methods rely on estimating piecewise-smooth curves, based on piecewise polynomial function [7]–[9] or B-spline [10], [11] or *Non-Uniform Rational B-Splines* (NURBS) [12], [13]. Other methods model the trajectory using Bezier Curves geometrically constructed on Manifold [14]. More recently, [15] presented an extension of the method to construct splines. For either of the later two works, the derivatives do not exhibit the same properties as the Euclidean case such as continuous smoothness at the splines knots.

The *Timed Elastic Bands* (TEB) planner [16], [17] rapidly became one of the most popular local planners for mobile-bases in the *Robot Operating System* (ROS) [18] community. Based on a TEB formulation (please refer to Sec. II-B.1) which originates in [19], the TEB-planner allows

This work was partially supported by the EU H2020 project LOGIMATIC (H2020-Galileo-2015-1-687534), and by the Spanish State Research Agency through projects EB-SLAM (DPI2017-89564-P) and the María de Maeztu Seal of Excellence to IRI (MDM-2016-0656).

[1] Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain. `jderay,jsola,cetto@iri.upc.edu`. Jérémie Deray is also with PAL Robotics, Barcelona, Spain.

[2] Heriot-Watt University, Edinburgh Centre for Robotics, Scotland. `bence.magyar@hw.ac.uk`

to efficiently and rapidly estimate a discretized trajectory in the plan. Formulated as a multi-objective optimization problem, it incorporates constraints such as the trajectory feasibility from a robot's kinematics point of view, together with velocity limits and obstacle avoidance.

This paper proposes a *Timed-Elastic Smooth Curve* (TESC) planner for mobile-base motion planning and control. Building upon the work of Rösmann *et al.* [16], [17] we formulate a sparse trajectory planning scheme whose discretization lies on piecewise C^n curve on a Lie manifold. Unlike the work of [16], [17] whose TEB's connectedness is ensured by kinematic constraints, our trajectory's connectedness is maintained by a revisited formulation of smooth interpolation on Lie manifolds. This ensures non-vanishing *n-th* derivatives at any point along the TEB. Moreover, our proposal differs from the traditional polynomial or spline-based trajectory estimation in that we do not aim at estimating coefficients [10] nor control points which encompass and define the curve [14], [15] but rather a discrete collection of points which themselves lie on the piecewise curve in SE(2) that forms the trajectory. The formulation of the problem allows for a seamless calculus of a pose at any time along the TEB.

The remainder of the paper is as follows. Section II details the TESC planner formulation. Section III describes the experiments and results. Section IV draws a conclusion and proposes further developments.

II. TESC Planning

A. Notation and Definitions

We consider the trajectory as a sequence of points lying on a Lie manifold SE(n) of rigid motions. Although our formulation can be expressed abstractly for each dimension n, in this paper we focus on the algebraic realization for the Lie manifold SE(2) of rigid motions in the plane (translation in the plan, rotation over the z-axis), which is where we performed all the benchmarking experiments. While the mathematical tools used in this work rely on linear algebra, rotations are not part of a *vector space*. This motivates the use of Lie groups and their associated Lie algebra, which *is* a vector space. The reader may refer to [20] for further details about the Lie theory of the most common Lie groups in robotics.

1) The SE(2) Lie group: We note generic elements $\left[\begin{smallmatrix}\mathbf{R} & \mathbf{p}\\ \mathbf{0} & 1\end{smallmatrix}\right] \in \mathrm{SE}(2)$ with $\mathbf{x}$, the translation with $\mathbf{p}$ and a 2D rotation matrix of θ rad with $\mathbf{R} = \mathbf{R}(\theta)$. For the sake of brevity we express $\mathbf{x}$ directly as $\mathbf{x} = (x, y, \theta) \triangleq (\mathbf{p}, \theta)$. Composition $\cdot$ and inversion $^{-1}$ are performed respectively with

$$\mathbf{x}_a \cdot \mathbf{x}_b = \begin{bmatrix}\mathbf{p}_a + \mathbf{R}_a\mathbf{p}_b \\ \theta_a + \theta_b\end{bmatrix} \quad , \qquad \mathbf{x}^{-1} = \begin{bmatrix}-\mathbf{R}^\top\mathbf{p} \\ -\theta\end{bmatrix} \quad . \tag{1}$$

2) Exponential map: Associated to any point of SE(2) is a tangent space. The tangent space at the identity is named the Lie algebra and noted $\mathfrak{se}(2)$. This space has elements $\left[\begin{smallmatrix}[\theta]_\times & \mathbf{u}\\ \mathbf{0} & 0\end{smallmatrix}\right]$ with $\mathbf{u} = [u, v]^T$ and $[\theta]_\times \triangleq \left[\begin{smallmatrix}0 & -\theta\\ \theta & 0\end{smallmatrix}\right]$, and is isomorphic to the Cartesian space $\mathbb{R}^3$. We thus express

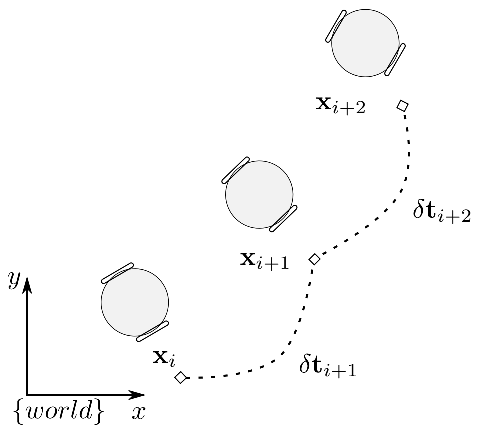


Fig. 2: The TEB is a sequence of robot poses forming a trajectory. Consecutive poses are tied to one another by a time interval.

tangent elements as $\boldsymbol{\tau} = (\mathbf{u}, \theta) \in \mathbb{R}^3 \simeq \mathfrak{se}(2)$. The exponential map relating $\mathbb{R}^3$ and SE(2) is as follows,

$$\boldsymbol{\tau} \in \mathbb{R}^3 \simeq \mathfrak{se}(2) \quad \underset{\log(\cdot)}{\overset{\exp(\cdot)}{\rightleftharpoons}} \quad \mathbf{x} \in \mathrm{SE}(2) \ . \tag{2}$$

Given $\mathbf{x} = (\mathbf{p}, \theta)$ and $\boldsymbol{\tau} = (\mathbf{u}, \theta)$, we have [21],

$$\exp(\boldsymbol{\tau}) \triangleq \begin{bmatrix}\mathbf{R}(\theta) & \mathbf{A}\cdot\mathbf{u} \\ \mathbf{0} & 1\end{bmatrix} \simeq \begin{bmatrix}\mathbf{A}\cdot\mathbf{u} \\ \theta\end{bmatrix} \tag{3}$$

$$\log(\mathbf{x}) \triangleq \begin{bmatrix}\mathbf{A}^{-1}\cdot\mathbf{p} \\ \theta\end{bmatrix} \ . \tag{4}$$

with

$$\mathbf{A} = \frac{1}{\theta}\begin{bmatrix}\sin(\theta) & -(1-\cos(\theta)) \\ 1-\cos(\theta) & \sin(\theta)\end{bmatrix} \ . \tag{5}$$

3) Plus and Minus: The operators $\oplus$ and $\ominus$ allow us to express variations around the manifold elements $\mathbf{x}$ as vectors $\boldsymbol{\tau}$ in its tangent space. They are defined such that, for $\mathbf{x}_a, \mathbf{x}_b \in \mathrm{SE}(2)$ and $\boldsymbol{\tau} = \mathbb{R}^3$,

$$\mathbf{x}_b = \mathbf{x}_a \oplus \boldsymbol{\tau} \ \triangleq \mathbf{x}_a \cdot \exp(\boldsymbol{\tau}) \quad \in \mathrm{SE}(2) \ , \tag{6}$$

$$\boldsymbol{\tau} = \mathbf{x}_b \ominus \mathbf{x}_a \triangleq \log(\mathbf{x}_a^{-1}\cdot\mathbf{x}_b) \ \in \mathbb{R}^3 \ . \tag{7}$$

These operators are respectively called right- plus and minus (see [20] for a more elaborate discussion).

B. TESC Problem Formulation

1) Timed Elastic Band: The Timed-Elastic-Band is described as a sequence of *n+1* robot poses $\in \mathrm{SE}(2)$ linking together an initial and a final configuration:

$$\mathbf{Q} = \{\mathbf{x}_i\}_{i=0\ldots n} \ , \tag{8}$$

with $\mathbf{x}_0$ fixed at the origin. All consecutive poses are tied to one another by n time intervals δt,

$$\Delta\mathbf{T} = \{\delta t_i\}_{i=1\ldots n} \ , \tag{9}$$

with δt_i denoting the time interval required for the robot to move from a pose $\mathbf{x}_{i-1}$ to $\mathbf{x}_i$ along the trajectory. The TEB is illustrated in Fig. 2.

2) Multi-objective Problem: Defining the pairs,

$$\mathbf{P} = \{(\mathbf{x}_i, \delta t_i)\}_{i=1\ldots n} \ , \tag{10}$$

the TEB is formulated as a multi-objective optimization problem:

$$\mathbf{P}^* = \arg\min_{\mathbf{P}} \sum_{k,i} w_k c_{ki}(\mathbf{Q}_i, \Delta\mathbf{T}_i) \ , \tag{11}$$

which can be solved by means of a least-squares nonlinear solver. The terms w_k are weight factors used to balance the different cost contributions $c_{ki}(\mathbf{Q}_i, \Delta\mathbf{T}_i)$. These costs depend on $\mathbf{Q}_i$ and $\Delta\mathbf{T}_i$, which are subsets of respectively $\mathbf{Q}$ and $\Delta\mathbf{T}$ in the neighborhood of δt_i. The cost functions are built simply with $c_k = \mathbf{e}_k^\top \mathbf{e}_k$, with $\mathbf{e}_k$ an error measure. The index $k \in \{s, v, a, j, l, g, o\}$ indicates the nature of each objective error during the interval δt_i. These are described hereafter with the time index i dropped.

3) C^n Smooth Curve: Considering the TEB as a collection of discrete points, we aim at enforcing consecutive points to lie on a smooth curve in SE(2), and the different pieces to form a continous, smooth, spline. Jakubiak *et al.* [22] propose a geometric two-step algorithm to generate smooth splines on Riemannian manifolds, in particular Lie groups. Given two configurations $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$, the algorithm allows for interpolating a configuration $\mathbf{x}_t$ so that it lies on a smooth curve connecting $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$. The smoothness constraint proposed here results from a revisited formulation of the interpolation algorithm of [22]. Given three consecutive poses $\mathbf{x}_{i-1}$, $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$, and their associated time intervals δt_i and δt_{i+1}, we proceed as follows. First compute the tangent vectors $\boldsymbol{\tau}_{i-1}$ and $\boldsymbol{\tau}_{i+1}$, which are approximated with backward differences, and the interpolation factor t,

$$\boldsymbol{\tau}_i = \mathbf{x}_i \ominus \mathbf{x}_{i-1} \tag{12}$$
$$t = \delta t_i/(\delta t_i + \delta t_{i+1}) \in [0, 1] \tag{13}$$

Then compute the desired interpolated point $\hat{\mathbf{x}}_i$ with

$$\mathbf{l}(t) = \mathbf{x}_{i-1} \oplus (t \cdot \ \boldsymbol{\tau}_{i-1}) \ , \tag{14}$$
$$\mathbf{r}(t) = \mathbf{x}_{i+1} \oplus ((t-1) \cdot \ \boldsymbol{\tau}_{i+1}) \ , \tag{15}$$
$$\boldsymbol{\beta}(t) = \mathbf{r}(t) \ominus \mathbf{l}(t) \ , \tag{16}$$
$$\hat{\mathbf{x}}_i = \mathbf{l}(t) \oplus (\phi(t) \cdot \boldsymbol{\beta}(t)) \ . \tag{17}$$

The resulting error is then:

$$\mathbf{e}_s = \hat{\mathbf{x}}_i \ominus \mathbf{x}_i \ . \tag{18}$$

While the definition of the real valued smoothing function $\phi(t)$ in (17) is given hereafter, the reader can refer to [22] for more detailed reference of its critical guarantees:

$$\phi(t) = \gamma \sum_{j=0}^{m} \frac{a_{m+1+j}}{m+1+j} t^{m+1+j} \ , \tag{19}$$

with

$$a_{m+1+j} = (-1)^j \binom{m}{j} t^{m+j} \ , \tag{20}$$

$$\gamma^{-1} = \sum_{j=0}^{m} \frac{a_{m+1+j}}{m+1+j} \ , \tag{21}$$

where m is the smoothness degree (C^m).

Figure 3 illustrates (14–18) and how the intermediate (middle) pose $\mathbf{x}_i$ is attracted towards the smooth curve defined by $\mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \boldsymbol{\tau}_{i-1}, \boldsymbol{\tau}_{i+1}$ and t.

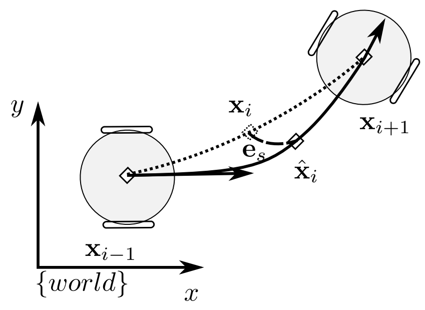


Fig. 3: The pose $\mathbf{x}_i$ is constrained towards the smooth curve defined by $\mathbf{x}_{i-1}, \mathbf{x}_{i+1}, \boldsymbol{\tau}_{i-1}$ and $\boldsymbol{\tau}_{i+1}$ (tangents are illustrated by the arrows).

The trajectory smoothness at segments junctions (knots) is implicitly ensured by design as the final tangent of the i-th segment is to be equal to the initial tangent of the $i+1$-th consecutive segment. We therefore do not require to explicitly impose equality constraints on knots, unlike other spline-based frameworks such as [23].

4) Curve's Derivatives Boudaries Constraints: The formulation from Sec. II-B.3 ensures non-vanishing *n-th* derivatives at every point, allowing to enforce upper and lower boundaries (*i.e.* inequality constraints) on the curve's derivatives. The derivatives subject to inequality constraints are:

- $\mathbf{v}_k$ the average velocity over a δt
- $\mathbf{a}_k$ the average acceleration over a δt
- $\mathbf{j}_k$ the average jerk over a δt .

They are approximated using backward finite differencing through a sliding window over the TEB of, respectively, the past two, three and four configurations,

$$\mathbf{v}_i = \frac{\mathbf{x}_i \ominus \mathbf{x}_{i-1}}{\delta t_i} \ , \tag{22}$$
$$\mathbf{a}_i = 2 \cdot \frac{\mathbf{v}_i - \mathbf{v}_{i-1}}{\delta t_i + \delta t_{i-1}} \ , \tag{23}$$
$$\mathbf{j}_i = 6 \cdot \frac{\mathbf{a}_i - \mathbf{a}_{i-1}}{\delta t_i + \delta t_{i-1} + \delta t_{i-2}} \ . \tag{24}$$

Similarly to [17] we use a nonlinear least-squares solver to optimize the TESC problem. Inequality constraints are approximated by two-sided quadratic penalties so that the error functions of an inequality constraint are,

$$e_\nu = \begin{cases} -\nu + \nu_L, & \text{if } \nu < \nu_L \\ +\nu - \nu_U, & \text{if } \nu > \nu_U \\ 0, & \text{otherwise} \ , \end{cases} \tag{25}$$

with ν a constrained variable, ν_L and ν_U respectively ν's lower and upper bounds, and $<$, $>$ are element-wise comparisons. From (22–25) results the following error functions:

$$\mathbf{e}_v \quad \text{subject to} \quad \mathbf{v}_L < \mathbf{v}_i < \mathbf{v}_U \ , \tag{26}$$

$$\mathbf{e}_a \quad \text{subject to} \quad \mathbf{a}_L < \mathbf{a}_i < \mathbf{a}_U \ , \tag{27}$$

$$\mathbf{e}_j \quad \text{subject to} \quad \mathbf{j}_L < \mathbf{j}_i < \mathbf{j}_U \ . \tag{28}$$

5) Non-holonomic Constraints: Since the trajectory optimized is that of a mobile-base, one has to take the kinematic constraint into account so that the trajectory is physically feasible. The first of such kinematic constraints is the non-holonomic constraint since differential or bicycle-like base cannot move sideways. It is imposed as,

$$e_h \quad \text{subject to} \quad \mathbf{v}_{yi} = 0 \ . \tag{29}$$

with $\mathbf{v}_{yi}$ the y-component of the velocity vector computed from (22). Given our current optimization framework, equality constraints are obtained by setting both the lower and the upper bounds to the same value.

6) Minimum Turning Radius Constraints: If the mobile-base considered is a bicycle-like model (*e.g.* car-like), one has to ensure a minimum turning radius. We implement the equivalent condition on the inverse radius, since unlike R, $1/R$ crosses zero continuously as the robot transitions from a left turn to a right turn. We have,

$$1/R = \mathbf{v}_{\omega i}/\mathbf{v}_{xi} \ , \tag{30}$$

$$e_r \quad \text{subject to} \quad -1/R_{min} < 1/R < 1/R_{min} \ , \tag{31}$$

with $\mathbf{v}_{xi}$ and $\mathbf{v}_{\omega i}$ respectively the x- and the angular-components of the velocity vector (22). For $\mathbf{v}_{xi} \rightarrow 0$, we constrain $\mathbf{v}_{\omega i}$ to zero in a way akin to (29) to avoid turning in place.

7) Minimizing Time and Trajectory Length: Not only shall a trajectory be feasible but also should it be as short as possible, both in terms of execution time and travelled distance. While a double objective of distance and time, *i.e.*, $c = w_l \langle \boldsymbol{\tau}, \boldsymbol{\tau} \rangle + w_t \, \delta t^2$ seems natural, we have found that having them enforce each other gives better results in terms of smoothness and stability of the solutions in the optimization framework used here. We define the joint length-time error as,

$$e_l = \langle \boldsymbol{\tau}_i \delta t, \boldsymbol{\tau}_i \delta t \rangle = \langle \boldsymbol{\tau}_i, \boldsymbol{\tau}_i \rangle \delta t^2 \ , \tag{32}$$

where $\langle \, , \rangle$ denotes the inner product[1] and $\langle \boldsymbol{\tau}_i, \boldsymbol{\tau}_i \rangle$ is the arc-length of the i-th segment squared. While this cost function has no particular physical meaning, it is preferable for a numerical optimization process. Having two different cost functions for time and length, each having their own weight, leads to optimizing $2(n-1)$ unique cost functions, each one having a fairly large residual with respect to the other cost functions. This is so since, *e.g.*, time will not reduce as much as the distance to the final target. Instead, by merging those two cost function into (32), we only have $n-1$ unique cost functions to minimize, a single weight to tune and a residual of a similar order as the others.

[1]On $\mathfrak{se}(2)$ the weigthed Euclidean inner product is defined as $\langle \boldsymbol{\tau}, \boldsymbol{\tau} \rangle = \boldsymbol{\tau}^T \cdot \mathbf{W} \cdot \boldsymbol{\tau}$, with $\text{diag}(\mathbf{W}) = [1, 1, 2]$ the weight matrix relative to the space basis.

8) Distance to the Target Configuration: While the previous cost functions constrain the overall shape of the trajectory, this one pulls the TEB's tip, $\mathbf{x}_n$, toward the desired configuration or *goal*, $\mathbf{x}_g$. It is defined as follows:

$$\mathbf{e}_g = \mathbf{x}_g \ominus \mathbf{x}_n \ . \tag{33}$$

9) Obstacle Avoidance: When dealing with mobile-base navigation, the robot's surrounding environment is often represented by a 2D *Occupancy Grid* (OG). In its simplest ternary form, the OG has three distinct values denoting whether a cell is free (no obstacle), occupied (obstacle) or unknown. In this work, the environment is represented by means of an *Euclidean Distance Grid* (EDG) which stores in each cell the distance to the closest obstacle in the grid. Cells representing an obstacle have a zero value. Such representation allows to efficiently evaluate whether a configuration is in collision with an obstacle or not. Given a 2D-OG, an EDG is computed using the distance transform algorithm described in [24]. This algorithm is of first choice as it is fast, efficient and computes an exact Euclidean distance. Given a distance grid and two consecutive poses $\mathbf{x}_i$ and $\mathbf{x}_j$ along the TEB, the obstacle avoidance constraint is computed as follows.

First, k poses are interpolated between $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$ as per [22], with k chosen in adequacy with the grid resolution. Then the EDG cells' distance corresponding to each of the $k+2$ poses are evaluated. The resulting error functions is

$$e_o = \begin{cases} r - d, & \text{if } d \leq r \\ 0, & \text{otherwise} \end{cases} \ , \tag{34}$$

with d the smallest distance evaluated over the $k+2$ poses and r the radius of the circle encompassing the robot footprint. Notice that complex footprint shapes can be considered and is left here to further work. Evaluating intermediate interpolated configurations allows to assert that a segment as a whole is obstacle free avoiding the common problem in discrete trajectory planning of having consecutive poses lying on each side of an obstacle.

III. Experiments

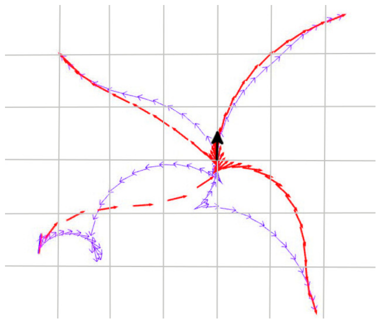
This section describes the experimental setup along with results from simulations in three different scenarios using the *TIAGo* mobile-manipulator robot's simulation[2].

We compare the proposed TESC approach against the state-of-the-art TEB planner [17].

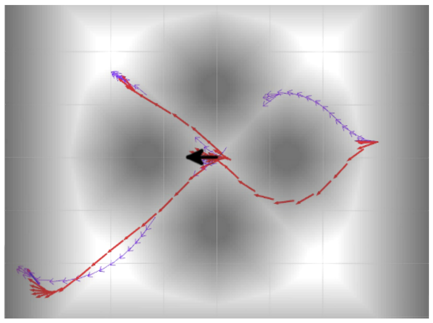
Specifically, each planner is executed a 1000 times for each scenario and results were compiled from these trials. To achieve a fair comparison TESC and TEB are initialized with the same velocity and acceleration limits.

The evaluation covers eight metrics. First is the success rate, which indicates whether the planner has found a collision-free trajectory or not. The optimization time, measures how much time the planner took to find a trajectory. As the robot operates in the real environment it is important to be able to find plans in a timely manner, especially

[2]http://wiki.ros.org/Robots/TIAGo



(a) Obstacle-free scenario, collage of four different goal.



(b) Four obstacles scenario, collage of three different goals.

Fig. 4: Experiment setups. TEB and TESC are depicted respectively with violet and red arrows. The initial pose at the center of the grid is depicted with a larger black arrow.

TABLE I: Metrics used in our experiments.

Metric	Description
Success rate in %	$100 \cdot \frac{success}{success+failure}$
Planning time in s	
Trajectory arc length	$\sum \|\|\mathbf{x}_i \ominus \mathbf{x}_i \;\; 1\|\|$
Trajectory time (s)	$\sum \delta_{t_i}$
Average velocity	mean($\|\|\mathbf{v}_i\|\|$)
Average acceleration	mean($\|\|\mathbf{a}_i\|\|$)
Energy	See (35)
Trajectory curvature	See (36)

for reactive control applications. The trajectory arc-length and the trajectory time show how much the robot has to move to reach the goal and the time it takes to do so. The average velocity and acceleration metrics encompass both their linear and angular components. Finally the energy is an approximation of the kinetic-energy, while the trajectory curvature highlights the smoothness of the trajectory's curve. Smoother trajectories require less acceleration/deceleration, therefore putting less stress on the mechanical parts of the robot. Moreover they allow people to feel safer in the robot's vicinity due to a more predictable behavior. The aforementioned height metrics are listed in Table I.

The energy metrics reads,

$$\sum \frac{||\log(\mathbf{x}_i^{-1} \cdot \mathbf{x}_{i-1})||}{2 \cdot \delta t_i^2} \tag{35}$$

The curvature is approximated as the sum of acceleration's norm in the global frame,

$$\sum ||2 \cdot \frac{\log(\mathbf{x}_i^{-1} \cdot \mathbf{x}_{i-1}) - \log(\mathbf{x}_{i-1}^{-1} \cdot \mathbf{x}_{i-2})}{\delta t_i + \delta t_{i-1} + \delta t_{i-2}}|| \tag{36}$$

The weight factors w_k for the TESC planner are empirically determined so that the costs $w_k c_{ki}(\mathbf{Q}_i, \Delta\mathbf{T}_i)$ are all of the same order of magnitude. Similarly to the observations made in [17], we found that the weights associated to both the kinematics (Section II-B.5) and the goal (Section II-B.8) must be an order of magnitude higher than other weights. The weight factors used for the TEB planner are those presented as optimal in the original paper.

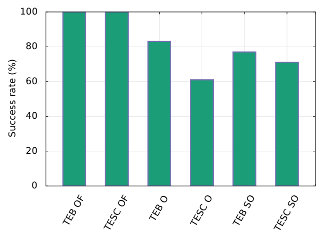
A. *Obstacle-free planning*

In the first scenario, the robot is located at the center of an 8×8m. grid with no obstacles and has to plan a trajectory toward a randomly generated pose. It is illustrated in Fig. 4a for four different goals. Statistics of the several metrics were summarized in Fig. 5, columns associated with this experiment are marked with the **OF** suffix. As Figs. 5a–5b show, the optimization time of both planners in this environment is well within realtime-capable requirements. Both planners performed very similar trajectory arc length, with a short advantage for TESC (Fig. 5c). Trajectory time as shown in Fig. 5d are also very similar. For the average velocity metric shown in Fig. 5e, TESC has a slightly higher values than TEB but has a much lower acceleration average (Fig. 5f). Finally, TESC outperformed TEB for both the total energy cost and the trajectory curvature shown respectively in Fig. 5g and Fig. 5h. These first results highlight that the smooth properties of the TESC approach improve the quality of the generated trajectories.

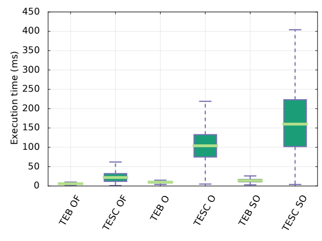
B. *Synthetic obstacles scenario*

The second environment is a 8×8m. grid with four circular obstacles surrounding the robot. Goals are randomly generated and their feasibility is verified. Fig. 4b depicts this scenario for three different goals. Statistics of the several metrics were summarized in Fig. 5, columns associated with this experiment are marked with **O** suffix.

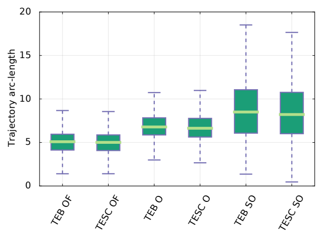
With the increase in challenge, the difficulty is reflected by both a decrease of success rates as well as an increase of execution time. As Fig. 5a depicts, TEB and TESC only succeeded planning in 83% and 61% of the time respectively. An example of failure - discontinued trajectory - is illustrated for TEB on the right side of Fig. 4b. TESC optimization time increases largely while TEB's remains fairly stable as visible in Fig. 5b. Both planners performed in a similar manner in terms of trajectory time. Observations for the trajectory arc-length and time are similar to those in Section III-A. With a lower average velocity (Fig. 5e), a much smaller acceleration (Fig. 5f) and overall curvature (Fig. 5g) TESC produced trajectories not only smoother but less prone to cause motion sickness and wear out vehicle hardware while consuming



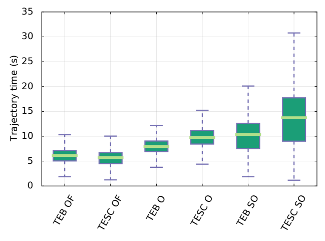
(a) Success rate (%)



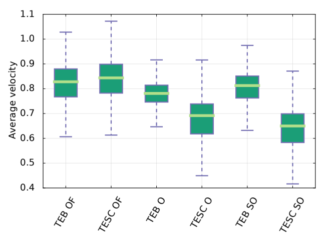
(b) Execution time (ms)



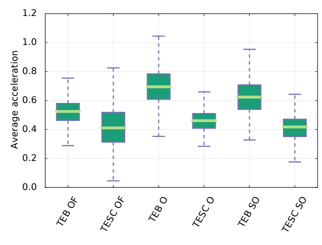
(c) Trajectory arc length (m)



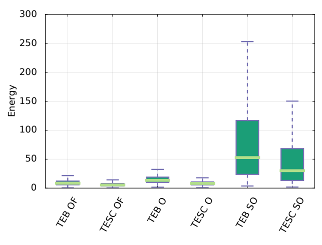
(d) Trajectory time (s)



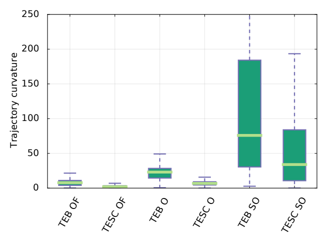
(e) Average velocity ($\frac{m}{s}$)



(f) Average acceleration ($\frac{m}{s2}$)



(g) Energy



(h) Trajectory curvature

Fig. 5: Experiment metrics

less energy (Fig. 5g). Note how even the deviation depicted in both Figs. 5g–5h is much smaller for TESC than TEB, signifying consistent better results.

C. *Complex obstacle scenario*

The third experiment considers a more realistic scenario relying on the OG generated from a simulated small office. The environment is of size $10,2 \times 14,85$m, constituted of two distinct rooms connected by an open door, both filled with furniture such as shelves and tables. Once again goals are generated randomly while their feasibility is verified. Statistics are presented in Fig. 5, columns associated with this experiment are marked with **SO** suffix.

Unlike experiment in Section III-B, the success rate of both planners is much closer to one another with 77% and 71% for respectively TEB and TESC. However the optimization time of TESC slightly increased again while TEB's remained fairly stable. Trajectory length and time for both planners show the same trend as for previous experiments. The same manner TESC shows once again a smaller average velocity (Fig. 5e) and a much smaller acceleration (Fig. 5f) than TEB, but also smaller and more consistent energy (Fig. 5g) and curvature (Fig. 5h).

D. *Implementation Details*

The TESC planner has been implemented in C++ using the least-squares solver Ceres [25] as it is flexible and offers automatic-differentiation. It also relies on our recently released `manif` library [20], a Lie-theory library for state-estimation. All scenarios are simulated using ROS [18], TESC has been integrated within the navigation stack and the output of these planners are directly applicable to real robots using `ros_control` [26]. The source code of the planner is available online. [3]

[3]https://github.com/artivis/tesc_planner

IV. Conclusion

We presented a novel formulation, *Timed-Elastic Smooth Curve* (TESC), for C^n smooth trajectory optimization. The generated trajectory's curve has non-vanishing *n-th* derivatives, allowing to constrain velocity, acceleration, jerk, *etc.* with ease. Moreover, the continuity of the curve at its knots is ensured by design, which relieves from the addition of extra cost functions to do so. While relying on a discrete set of points, its formulation allows for interpolating points that do belong to the trajectory curve. This property allows *e.g.* to ensure that the whole trajectory is collision-free. We benchmarked TESC in a series of mobile base motion planning scenarios and shown that it prevails or matches the performance of the TEB planner in most presented metrics. TESC has also proven to be more consistent in the quality of the generated trajectories.

However challenging, these experimental scenarios put both planners to the test. While TESC could not uniformly prevail in all metrics, we believe it is due to *Euclidean Distance Grid* (EDG) discretization and unsigned-ness, leading to discrete or non-existing gradients, thus to optimization issues.

Our experiments have shown that TESC is planning more energy-efficient and smoother trajectories of the same length as TEB but with a smaller average velocity and acceleration at the cost of increasing the optimization time.

Further work includes extending the framework to other Lie manifolds, specifically SE(3). This is feasible with little modification, since (12–18) are group-agnostic, simply employing the appropriate $\exp(\cdot)$ and $\log(\cdot)$ functions. Such $3D$ planner would typically be interesting for optimizing the trajectory of *e.g.* an Unmanned Aerial Vehicle, or the end-effector of a robotics arm in robotics manipulation tasks. A second extension will be the use of a *Signed Euclidean Distance Transform* for our collision avoidance cost, allowing for the existence of discrete gradient throughout the grid and thus to recover from a collision.

References

[1] M. Elbanhawi, M. Simic, and R. Jazar, "In the passenger seat: investigating ride comfort measures in autonomous cars," *IEEE Intelligent Transportation Systems Magazine*, vol. 7, no. 3, pp. 4–17, 2015.

[2] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *IEEE International Conference on Robotics and Automation*, vol. 2, March 1985, pp. 500–505.

[3] J. Kuffner and S. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *IEEE International Conference on Robotics and Automation*, vol. 2, 2002, pp. 995–1001.

[4] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[5] 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, 1996.

[6] M. Zefran, V. Kumar, and C. B. Croke, "On the generation of smooth three-dimensional rigid body motions," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 4, pp. 576–589, Aug 1998.

[7] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *IEEE International Conference on Robotics and Automation*, May 2011, pp. 2520–2525.

[8] H. Oleynikova, M. Burri, Z. Taylor, J. Nieto, R. Siegwart, and E. Galceran, "Continuous-time trajectory optimization for online UAV replanning," *IEEE International Conference on Intelligent Robots and Systems*, pp. 5332–5339, Sep 2016.

[9] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-D complex environments," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1688–1695, July 2017.

[10] S. Lovegrove, A. Patron-Perez, and G. Sibley, "Spline Fusion: A continuous-time representation for visual-inertial fusion with application to rolling shutter cameras," in *British Machine Vision Conference*. BMVA Press, 2014, pp. 93.1–93.11.

[11] T. Mercy, R. V. Parys, and G. Pipeleers, "Spline-based motion planning for autonomous guided vehicles in a dynamic environment," *IEEE Transactions on Control Systems Technology*, pp. 1–8, 2017.

[12] S. Jalel, P. Marthon, and A. Hamouda, "NURBS based multi-objective path planning," in *Mexican Conference on Pattern Recognition. Lecture Notes in Computer Science*, vol. 9116, 2015, pp. 190–199.

[13] ——, "A new path generation algorithm based on accurate NURBS curves," *International Journal of Advanced Robotic Systems*, vol. 13, no. 2, p. 75, 2016.

[14] F. C. Park and B. Ravani, "Bézier curves on Riemannian manifolds and Lie groups with kinematics applications," *Journal of Mechanical Design*, vol. 117, no. 1, p. 36, 03 1995.

[15] T. Popiel and L. Noakes, "Bézier curves and C^2 interpolation in Riemannian manifolds," *Journal of Approximation Theory*, vol. 148, no. 2, pp. 111 – 127, 2007.

[16] C. Rösmann, W. Feiten, T. Wsch, F. Hoffmann, and T. Bertram, "Efficient trajectory optimization using a sparse model," in *2013 European Conference on Mobile Robots*, Sept 2013, pp. 138–143.

[17] C. Rösmann, F. Hoffmann, and T. Bertram, "Kinodynamic trajectory optimization and control for car-like robots," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2017, pp. 5681–5686.

[18] M. Quigley, K. Conley, B. 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*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

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

[20] J. Solà, J. Deray, and D. Atchuthan, "A micro lie theory for state estimation in robotics," Institut de Robòtica i Informàtica Industrial, Barcelona, Tech. Rep. IRI-TR-18-01, 2018.

[21] E. Eade, "Lie groups for 2d and 3d transformations," Tech. Rep., 2013.

[22] J. Jakubiak, F. S. Leite, and R. C. Rodrigues, "A two-step algorithm of smooth spline generation on riemannian manifolds," *Journal of Computational and Applied Mathematics*, vol. 194, no. 2, pp. 177 – 191, 2006.

[23] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *16th International Symposium on Robotics Research*, ser. Springer Tracts in Advanced Robotics. Springer, 2016, vol. 114, pp. 649–666.

[24] P. F. Felzenszwalb and D. P. Huttenlocher, "Distance transforms of sampled functions," *Theory of Computing*, vol. 8, no. 1, pp. 415–428, 2012.

[25] S. Agarwal, K. Mierle, and Others, "Ceres solver," http://ceres-solver.org.

[26] S. Chitta, E. Marder-Eppstein, W. Meeussen, V. Pradeep, A. R. Tsouroukdissian12, J. Bohren, D. Coleman, B. Magyar, G. Raiola, M. Lüdtke, *et al.*, "ros_control: A generic and simple control framework for ROS," *The Journal of Open Source Software*, vol. 2, p. 456, 2017.