2018 European Control Conference (ECC)
June 12-15, 2018. Limassol, Cyprus

Efficient Re–planning for Robotic Cars

Enrico Bertolazzi, Paolo Bevilacqua, Francesco Biral, Daniele Fontanelli, Marco Frego and Luigi Palopoli

Abstract— **We consider the problem of the reactive re-planning of an optimal trajectory for autonomous vehicles subject to geometric and dynamic constraints. Reactive re-planning is used when a vehicle following a planned trajectory encounters an unforeseen obstacle. In such a case, a new local trajectory that avoids the obstacle has to be generated, without violating any constraint and preserving optimality. The solution presented in the paper guarantees that the new trajectory re-joins the previously planned one shortly after the obstacle. Moreover, the transition between old and new trajectory is smooth up to second derivative (curvature), which makes it easy to track an emergency manoeuvre. Finally, our solution is efficient and can be implemented in real–time on lean hardware. In order to validate the approach, we show how the re-planning can be executed in a few milliseconds (on a standard machine) for the realistic example of a racing car.**

I. Introduction

The problem addressed in the paper is how to reactively re-plan a local trajectory to be followed by an autonomous vehicle between two different configurations in minimum time and avoiding an unforeseen obstacle. The vehicle moves along a pre-defined trajectory and the technique proposed in the paper seeks to minimise the amount of the local deviation ensuring that the new trajectory joins the previously planned one shortly after the obstacle.

The considered vehicle is a car–like one, described by its dynamic model. We take into explicit consideration bounds on the longitudinal acceleration, lateral forces and aerodynamic drag. The vehicle is required to move on a track, which restricts the possible trajectories that can be taken. The occasional presence of obstacles and of slower vehicles generates additional geometric constraints. We make the following three assumptions: 1. the geometry of the track is known (i.e. road or lane limits), 2. the vehicle is detected with a sensing system able to reveal obstacles and anomalous conditions in the surroundings (e.g., slippery areas on the road), 3. the trajectory re-planning has to be executed in real–time and adapted every time an unforeseen situation is detected. In addition, we require the use of lean hardware in order to reduce the costs and simplify the system engineering. The potential relevance of the problem is evident in many application domains, ranging from Advanced Driving Assistance to robotic car races. However, this paper focuses only on robotic racing cars.

E. Bertolazzi, F. Biral and D. Fontanelli (daniele.fontanelli@unitn.it) and are with the Department of Industrial Engineering (DII); P. Bevilacqua and and M. Frego and L. Palopoli (luigi.palopoli@unitn.it) and are with the Department of Information Engineering and Computer Science (DISI), University of Trento, Via Sommarive 9, Trento, Italy. This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme - Societal Challenge 1 (DG CONNECT/H) under grant agreement n° 643644 "ACANTO - A CyberphysicAl social NeTwOrk using robot friends".

Related Work. The path planning problem has received a constant attention in the past few years (see [1] for a survey). Some of the solutions adopt geometric approaches [2] and produce sequences of curves that avoid the obstacles. A potential limitation of these solutions is that they neglect the vehicle dynamics and the planned path often presents a discontinuous curvature that makes it difficult to follow.

In his famous work Dubins [3] set the stage for the solution of path planning problems for vehicles moving at constant speed, with limited curvature radii and described by a kinematic model. More recently, Sanfelice et al. [4] applied the Pontryagin Maximum Principle to generalise Dubins results. Despite the recognised importance of these contributions, their applicability is limited in all cases in which dynamic effects, acceleration bounds and slipping constraints cannot be neglected (e.g., high speed vehicles or aggressive manoeuvre).

Dynamic effects can be considered with good results using complex nonlinear optimisation tools [5], [6], [7]. However, the difficult set up of these tools and the lack of any convergence guarantee make their application problematic in dynamic scenarios where the environment changes. An alternative approach is based on the fast generation and selection of feasible kinematic trajectories with direct search and subsequent improvement of the obtained solution, incorporating dynamic properties [8], [9], [10]. The generation of feasible trajectories can be done using stochastic search algorithms such as RRT or RRT* [11], [12], [13].

A different area of research is on threat assessment for semi-autonomous vehicles [14], in which MPC techniques are used to predict the future configuration of the vehicle and therefore forecast possible accidents. However, despite their improvements, MPC based approaches cannot always guarantee convergence (i.e. a solution) in the given time frame. To partially overcome this issue the planning problem can be generally split in a two-level optimisation with hierarchical control structure where the higher level provides a trajectory (possibly with speed profile) and the lower level tracks the computed trajectory, as in [15], [16]. Our work falls within this category, but the high-level planner is based on the computation of semi-analytic time-optimal solutions satisfying the geometric and dynamic constraints of the problem in a computationally efficient way, i.e. in micro-seconds on small embedded processors. Moreover, the algorithm can guaranteed if the solution exists: if not, a fallback solution is used (i.e. stop manoeuvre). Finally, being the computation extremely fast, we can evaluate a large

number of manoeuvres, as in [17], and then pick the best one. The presented work focuses on the high-level planner.

Paper Contribution. In the same way as proposed by different authors [18] and in our previous work [19], we advocate a strategy for trajectory re-planning based on a decomposition of the problem into geometric and dynamic planning, which makes a remarkable difference with respect to similar solutions considering just the geometric re-planning [20]. When the obstacle is detected, the algorithm selects a point P_0 on the optimal trajectory with position (x_0, y_0), angle θ_0, curvature κ_0 and speed v_0 and a point P_2, with position (x_2, y_2), angle θ_2, curvature κ_2. The re-planned trajectory will depart from the old one at point P_0 and will rejoin it at point P_2. Notably, the speed in P_2 cannot be assigned, because, in general, it may not be reachable, e.g. if the global trajectory is time optimal. The algorithm seeks a new point P_1 in the proximity of the obstacle to pivot on to find the best trajectory (see Figure 1). For each candidate point, a geometric optimisation phase is used to find the path and to verify its feasibility. The re-planned spline has to be contained within the track limits. For each candidate path, a dynamic optimisation is executed to find the time optimal manoeuvre, if it exists, using the semi-analytic solution shown in [21], [22], [19]. The connecting curve is a spline of clothoid curves [23] that exhibits C^2 continuity with respect to the global trajectory, and that is very fast to compute. The different choices of point P_1 can be explored via a deterministic search (as done in the paper) or by using stochastic methods [11], [12]. As shown in the numerical validation, our strategy is computationally efficient with a new trajectory produced in a few milliseconds on standard architectures. In most of the reasonable application scenarios, the method reliably produces a solution. In the extreme cases in which it should not work, its efficiency leaves time to slow down the vehicle and back off to different planning solutions.

The paper is organised as follows. Section II presents the track description and the adopted car model with the overview of the planning framework. Section III introduces and solves the re-planning problem. In Section IV we show some numeric examples and, finally, in Section V, we offer our conclusions and discuss future work directions.

II. Background

Modelling. In this section we discuss a convenient choice of curves that approximate the optimal trajectories of a car–like vehicle. The adopted kinematic model in the xy-plane is given by:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \\ \dot{\bar{\delta}} \end{bmatrix} = \begin{bmatrix} \cos\psi \\ \sin\psi \\ \tan\bar{\delta}/l \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \bar{\omega}, \tag{1}$$

where ψ is the orientation (yaw angle) of the vehicle with respect to a right-handed reference frame having the Z axis perpendicular to the $X \times Y$ plane of motion, $\bar{\delta}$ is the steering angle, v is the forward velocity of the vehicle, $\bar{\omega}$ is the normalised angular velocity of the steering wheel and $l > 0$ is the wheelbase. In our problem the vehicle is required to move on a track delimited by two curves avoiding obstacles, which can themselves be modelled as closed curves on the track. When the vehicle moves along a curve, its longitudinal dynamics can be described in terms of a longitudinal abscissa s, with $\dot{s}(t) = v(t)$. The dynamics of the longitudinal velocity $v(t)$ when $v(t) \geq 0$ is governed by a differential equation

$$\dot{v}(t) = a(t) - c_0 v(t) - c_1 v^2(t), \tag{2}$$

which accounts for, normalised with mass, laminar friction $c_0 > 0$ and aerodynamic drag $c_1 > 0$. The term $a(t)$ is a control variable modelling the positive or negative longitudinal acceleration of the car-like vehicle. We require that the later acceleration never exceeds a bound $A_{\max}$. Therefore the constraints on the lateral and longitudinal acceleration are expressed as $a(t) \in [-\underline{a}, \overline{a}]$ and $|k(s(t))|v^2(t) \leq A_{\max}$, where $k(s)$ is the curvature of the trajectory at arc length s.

Clothoids. Our method seeks the optimal trajectory to join two points P_0 and P_2 located on the path. They are characterised by their position, as well as by their angle and the curvature configuration of the global trajectory.

A defining characteristic of the method is that it restricts the search on trajectories composed of sequences of clothoids. The clothoid curve [23] is defined as

$$\begin{aligned} x(s) &= x_0 + sX_0(\kappa' s^2, \kappa s, \theta), \\ y(s) &= y_0 + sY_0(\kappa' s^2, \kappa s, \theta), \end{aligned} \tag{3}$$

where

$$\begin{aligned} X_n(a,b,c) &= \int_0^1 \tau^n \cos\left(\frac{a}{2}\tau^2 + b\tau + c\right) \mathrm{d}\tau, \\ Y_n(a,b,c) &= \int_0^1 \tau^n \sin\left(\frac{a}{2}\tau^2 + b\tau + c\right) \mathrm{d}\tau. \end{aligned} \tag{4}$$

It is worthwhile to note that for a clothoid, the tangent takes the form $\psi(s) = \frac{1}{2}\kappa' s^2 + \kappa s + \theta$. The relation between ψ and the curvature k is differential, i.e., $\psi'(s) = k(s) = \kappa + \kappa' s$, which points out a crucial feature: the curvature is a linear (affine) function of the arc length s. In this environment, this makes the clothoid superior to polynomial splines, which have also parametrisation problems.

Clothoids are a convenient choice since they are natural to follow for a car–like vehicle and they are a close approximation of the optimal solution. Indeed, clothoids are the optimal solution of the minimum time control problem for the model in Equation (1) if the car-like moves at constant velocity, see [2]. When the velocity of the vehicle is not constant, there is no guarantee that the optimal trajectory will be a sequence of clothoids. However, it is known [19], [24] that a carefully chosen sequence of clothoids is typically a good approximation of the optimum for the model (1). In particular, [24] guarantees that the clothoid-based synthesis satisfies the maximum lateral acceleration limit.

Selection of entry and exit points. In principle, the algorithm presented below operates with any pair of entry and exit points P_0–P_2 on the global trajectory. An obvious requirement is that P_0 and P_2 be located before and after

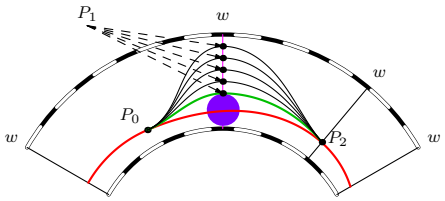

Fig. 1. Structure of the track with a sketch of the used notation. The way lines w divide the track into sectors, in red the global trajectory that is no more feasible because of the obstacle (purple circle). In green the optimal escaping manoeuvre, in black feasible candidates for different choices of P_1.

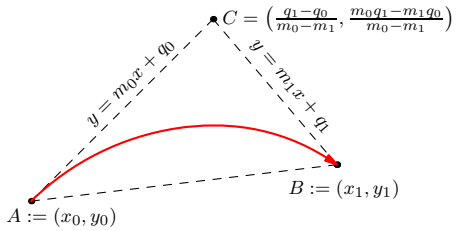

Fig. 2. Example of construction of the triangle that contains a clothoid in the xy plane. The tangents in A and in B with the segment AB form the triangle (dashed) used to check the intersection of the (red) clothoid with other objects.

the obstacle. The low computational cost of the algorithm allows us to test different possible choices or back off to an emergency strategy if the spline identified by the algorithm fails to satisfy the geometric or the dynamic constraints. However, the application of reasonable heuristics on the selection of P_0 and P_2 limits the occurrence of this anomaly. It is useful to observe that if the obstacle is very close to the vehicle, no feasible solution is likely to be found. On the other hand, if the obstacle is far away, we are in condition to generate a new global plan. The local re-planning is applied whenever the obstacle is detected since we have in mind minimum time manoeuvre (i.e. overtake if possible).

In this case, an intuitive and straightforward way to produce the points P_0 and P_2 is to identify them as the intersections of the global trajectory with a finite set of way lines superimposed to the track, see Figure 1. The distance between two subsequent way lines can be tailored to the geometric features of the track and to the dynamic properties of the vehicle. For instance, way lines can be closer in curved sectors of the track and farther in straight sectors. When an obstacle is detected, P_0 can be chosen coincident with the current position of the vehicle. P_2 can be chosen on the intersection between the global trajectory and the first way line beyond the obstacle. As a final remark, if a multicore architecture is available, multiple choices for P_0 and P_2 could be explored in parallel.

III. The Reactive Re-planning Problem

We assume that the sensors or other devices provide the information of an obstacle on the track such that the execution of the previously computed trajectory will cause a collision.

A. Obstacle and Lane Models

In this paper, we assume that the on-line detected obstacle is static, however a (slowly) moving obstacle could be considered as static for a certain time interval if we extend its size along its moving direction. The obstacles and the boundary of the track are modelled as clothoids (where segments and circular arcs are particular cases), even though the assumption could easily be released using any kind of smooth analytical curve. Thereby, validating a path amounts to find if two clothoids intersect. The application of standard methods (e.g., Newton method) to find the intersection is not always a viable option because of potential singular configurations (e.g., when the clothoids are close to be tangent). Along the line of our previous work [19], [25], we advocate a different approach based on the computation of the bounding triangles that inscribe the two curves followed by the check of their potential intersection (which for triangles is rather quick). To summarise this approach, consider a clothoid of parameters κ', κ, θ in the interval $(0, L)$ such that the variation of the angle is less than $\pi/2$, intuitively this corresponds to "c" shaped curves. Then the clothoid is contained in the triangle ABC, where $A = (x_0, y_0)$ is the initial point of the clothoid arc, B is the point at the end, e.g. for $s = L$ and C is given by the intersection of the tangents to the clothoid in A and B (Figure 2). An additional advantage of the approach is that the intersection of a triangle with a large number of other triangles (e.g., resulting from the decomposition of a "long" clothoid spline) can be hierarchically organised and executed in a very efficient way using a tree data structure. This leads to a fine granularity for the triangles.

B. Reactive Re-planning

In this framework, where we already have the optimal trajectory, we compute the local re-planning that has to avoid the obstacle and connect with C^2 continuity to the points P_0 and P_2 (e.g. Figure 1 or Figure 4 for the case study). For safety reasons, we first compute if it is possible to stop the vehicle before crashing into the obstacle while travelling on the assigned trajectory; after that we compute the reactive re-planning that tries to avoid the obstacle as in Figure 1. We require the following tools to compute the reactive re-planning, most of which are available in literature: a function for computing the time optimal speed profile for a given sequence of clothoids [21], [22], [19], and two functions that solve the G^1 and G^2 Hermite Interpolation Problem for clothoid curves (HIP). Those interpolation problems ask to find the parameters of a clothoid that interpolates the assigned initial and final points and angles, producing a spline that has geometric continuity up to the first derivative (G^1 case). If also the curvature at the extremal points is specified, we speak of G^2 HIP. Notice that in case of clothoids, which are parametrised by arc length, the geometric continuity (G^1 and G^2) is the same of the usual continuity (respectively C^1 and C^2). The notation with G is kept because it is traditional in this kind of problems. The G^1 problem has

been solved efficiently in [23], whereas the solution to the G^2 problem is discussed in [26]. For our scope it is sufficient to consider those solutions as two functions g1Hip and g2Hip, the latter corresponding to the solution of a nonlinear system of 16 equations that represent the continuity constraints. For instance, the G^2 continuity between the segments connecting P_i to P_{i+1} requires that the following relations must hold: $H_{i,j}(\theta_i, \kappa_i, \kappa_i', L_i, \theta_{i+1}, \kappa_{i+1}) = 0$ for $j = 0, 1, 2, 3$, where

$$H_{i,0} := x_i + L_i X_0\big(\kappa_i' L_i^2, \kappa_i L_i, \theta_i\big) - x_{i+1}, \quad (5)$$

$$H_{i,1} := y_i + L_i Y_0\big(\kappa_i' L_i^2, \kappa_i L_i, \theta_i\big) - y_{i+1}, \quad (6)$$

$$H_{i,2} := (1/2)\kappa_i' L_i^2 + \kappa_i L_i + \theta_i - \theta_{i+1}, \quad (7)$$

$$H_{i,3} := \kappa_i' L_i + \kappa_i - \kappa_{i+1}. \quad (8)$$

Equations (5) and (6) ensure point-wise continuity, whereas (7) and (8) stand for the angle and curvature, X_0 and Y_0 are the functions defined in (4). The function g2Hip has to solve 4 of such blocks and can be evaluated relatively easily but only a limited number of times to keep the whole algorithm fast enough for real time applications.

We present now the main contribution of the present paper, the geometric smooth connection of the escape manoeuvre to the optimal trajectory. The setting is the following: in P_0 we know position, angle and curvature, as well as in P_2, in the middle point P_1 we know the position only, suitable angles and curvature should be found in order to connect P_0 to P_1 and P_1 to P_2 via the G^2 HIP. For the sake of computational efficiency, we wish to call the g2Hip only twice, when the final interpolation is performed, and prefer instead to rely on the very fast g1Hip function to find the missing parameters, which costs 4 Newton iterations in the worst case (2 on average, see [23]). The idea is to construct two clothoid arcs neglecting the curvature at P_0 and P_2 but with G^2 continuity at P_1. We use then the retrieved angle and curvature in P_1 to interpolate from P_0 to P_1 and next to P_2 maintaining G^2 continuity with the assigned curvatures κ_0 and κ_2 by means of the function g2Hip.

We start connecting P_0 to P_1 with the function g1Hip, the G^1 algorithm takes as input the points and angles to be interpolated and gives as output the curvatures κ_0', $\overline{\kappa}_0$ and the length L_0 of the clothoid. In our case g1Hip can thus be seen as a function of one unknown variable θ_1 because the other parameters are fixed, hence we obtain $\kappa_0'(\theta_1)$, $\overline{\kappa}_0(\theta_1)$ and $L_0(\theta_1)$. Analogously we perform the same calculations between P_1 and P_2 obtaining $\kappa_1'(\theta_1), \kappa_1(\theta_1)$ and $L_1(\theta_1)$. Because θ_1 is the end angle of the first segment and the initial angle of the second segment, the resulting arcs match with G^1 continuity in P_1. Thus, we have a family of splines made by two clothoids parametrised with θ_1 and we can vary this value in order to satisfy the G^2 constraint, i.e. the curvatures of the two arcs in P_1 must match. This condition is

$$h(\theta_1) := \kappa_0'(\theta_1) L_0(\theta_1) + \kappa_0(\theta_1) - \kappa_1(\theta_1) = 0, \quad (9)$$

which is simply equation (8) specialised to our scope. To find an angle θ_1 that satisfies $h(\theta_1) = 0$ a standard Newton–Raphson scheme suffices, which works well in practice, and only 2-3 (cheap) iterations are required. The derivative of $h(\theta_1)$ can be also explicitly written and is an expression that depends on already computed values. It can be easily evaluated by differentiating with respect to the angles θ_i and θ_{i+1} the three equations (5), (6), (7); the resulting linear relations for the desired derivatives are then solved straightforwardly. Even tough the proof is simple, the explicit expressions of the results require some space and are herein omitted.

We wish to remark that the computed $\overline{\kappa}_0(\theta_1)$ is in general not equal to the value κ_0 of the original trajectory and is used in the estimation of θ_1 and consequently $\kappa_1(\theta_1)$ only. Once these values are retrieved, we can apply g2Hip to compute the real G^2 spline that satisfies the smooth G^2 connection with the original trajectory.

Once one of such escaping manoeuvres is computed, we check if it is collision free and in such a case we evaluate its time optimal speed profile with the function minTime, which solves the optimal control problem for the velocity profile [24]. Because the whole algorithm is very fast, it is possible to compute more of such escaping manoeuvres and select the one with minimum time, as it is shown in Figure 1, where the green curve is selected among the black curves. It is worthwhile to note that if an additional obstacle is detected along a re-planned trajectory, the re-planning can be similarly iterated, e.g. analogously using the approach proposed in [27] for service robots.

C. Algorithmic Version of the Re-planning

The compact algorithmic form of the previous section can be summarised in Algorithm 1.

Algorithm 1: Find a collision free partial path

Function *AvoidObstacle*
- **Input :** conf_0, initial config. of the vehicle (x_0, y_0, θ_0, κ_0);
  conf_2, final config. of the vehicle (x_2, y_2, θ_2, κ_2);
  v_0, initial velocity of the vehicle;
  waypoints, intermediate waypoints.
- **Output:** {time, traj}, collision free traj. with lowest cost.

```
time ← ∞; traj ← [ ];
foreach wp ∈ waypoints do
    [θ1, κ1] ← findThetaKappa(conf0,conf2,wp);
    spline1 ← g2Hip(conf0,{wp, θ1, κ1});
    if not collisionFree(spline1) then continue;
    spline2 ← g2Hip({wp, θ1, κ1},conf2);
    if not collisionFree(spline2) then continue;
    currTraj ← [spline1, spline2];
    {ok, currMan} ← minTime(currTraj,v0);
    if not ok then continue;
    [currTime] ← time(currMan);
    if currTime < time then
        time ← currTime; traj ← currTraj
    end
end
return {time, traj};
```

The input is given by the initial and final configurations of the vehicle, thus position, angle and curvature, and by the initial velocity. The final velocity is neglected, since it is almost impossible to reconnect to the global trajectory with the optimal speed. Moreover, the re-computation of

the speed profile for the whole lap (if it is modelled as a sequence of clothoids) is computationally very cheap (less than a millisecond, see [19]). Since the algorithm loops on all the available waypoints, they are also part of the input. The output is given by the best escaping manoeuvre computed by the reactive re-planning (if one exists). Notice that Algorithm 1 can be used also to safely stop the vehicle if an overtaking manoeuvre does not exist. The first part of the algorithm involves the search of the missing angle and curvature at P_1, namely θ_1 and κ_1, and is done by solving equation (9) (see line 1 and Algorithm 2). This computation requires to solve the G^1 HIP for the two clothoids that connect P_0 to P_2 passing through P_1 with G^1 continuity. The resulting values for θ_1 and κ_1 are then used by the g2Hip function to compute the final smooth spline that connects P_0 to P_2 with G^2 continuity, e.g. without jumps in the curvature. The spline is also checked for collisions with the obstacle and the track limits. If the spline is valid, the time optimal speed profile is computed (fixing the initial velocity to the current velocity of the vehicle). When all the possible trajectories have been processed, the algorithm terminates and returns the solution with the lowest travel time.

Algorithm 2: Find intermediate angle and curvature

Function ***FindThetaKappa***

Input : conf_0, initial config. of the vehicle (x_0, y_0, θ_0, κ_0);
conf_2, final config. of the vehicle (x_2, y_2, θ_2, κ_2);
wp, intermediate waypoint (x_1, y_1);
$\{\max_{\text{iter}}, \text{tol}\}$, termination criteria

Output: $\{\theta_1, \kappa_1\}$

$\theta_1 \leftarrow (\theta_0 + \theta_2)/2$;
for iter *from* 1 *to* $\max_{\text{iter}}$ **do**
  $[\bar{\kappa}, \bar{\kappa}', L, \bar{\kappa}_{\theta_1}, \bar{\kappa}'_{\theta_1}, L_{\theta_1}]_0 \leftarrow$ g1Hip(x_0,y_0,θ_0,x_1,y_1,θ_1);
  $[\kappa, \kappa', L, \kappa_{\theta_1}, \kappa'_{\theta_1}, L_{\theta_1}]_1 \leftarrow$ g1Hip(x_1,y_1,θ_1,x_2,y_2,θ_2);
  $h \leftarrow \bar{\kappa}_0 + \bar{\kappa}'_0 L_0 - \kappa_1$;
  $h' \leftarrow \bar{\kappa}_{0\theta_1} + \bar{\kappa}'_{0\theta_1} L_0 + \bar{\kappa}'_0 L_{0\theta_1} - \kappa_{1\theta_1}$;
  if $|h| < \text{tol}$ **then return** $\{\theta_1, \kappa_1\}$;
  $\theta_1 \leftarrow \theta_1 - h/h'$;
end
return $\{\}$

As a final remark, we point out that it is possible to employ more curves by adding more intermediate points between P_0 and P_2, but at the price of a (slightly) higher computational cost. In fact, the key point of the present algorithm is the root finding of a nonlinear system of the form (5)–(8), that, for only one middle point P_1, boils down to the single equation (9), that can be solved with few Newton–Raphson iterations.

IV. Application Example

In this section we discuss an application example of the present algorithm: the reactive re-planning of a time optimal trajectory on a lap of the Formula 1 track of Silverstone, UK. The track is shown in Figure 3 along with the global plan computed as in [19]. In our scenario we simulated the unpredicted presence of an obstacle in the area denoted by the red rectangle. We applied the approach presented in the paper to compute a local modification of the trajectory. The result is shown in Figure 4, where a small deviation was

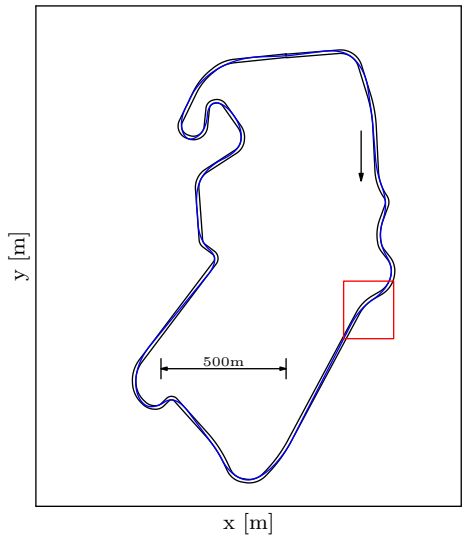

Fig. 3. The circuit track of Silverstone, UK. In red the portion of the track that is zoomed in Figure 4.

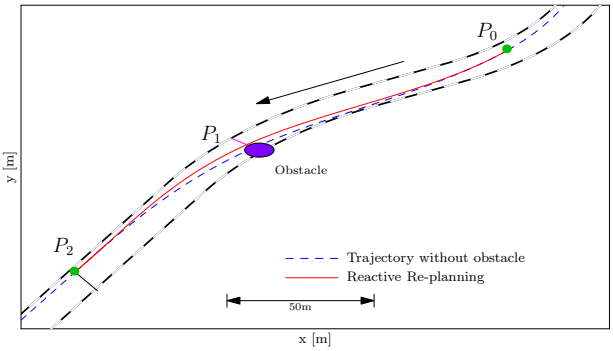

Fig. 4. Re-planning in the presence of an obstacle. The original trajectory (in blue dashed) is no more feasible because of the obstacle. The re-planning (solid red line) can be done between two points on the way lines P_0 and P_2, because before P_0 and after P_2 the trajectory is close to the old one without the obstacle and allows a smooth reconnection.

found to avoid the obstacle and then rejoin the global path on point P_2 chosen on the way line right after the obstacle. This local re-planning is very fast (in the order of the milliseconds on a Core I5 2.3 GHz machine) as it involves only few way lines. We tested the escaping manoeuvre posing a circular obstacle of radius 5 m on the track. We introduced a way line in its proximity and performed two simulations: in the first we sampled 15 waypoints uniformly distributed, in the second 30. The execution time of the algorithm was of 3.94 ms in the first test and 7.80 ms in the second case. Due to stringent space limits, we reported only one example, even though the algorithms have been tested placing the obstacles in different positions as well as using different tracks. In all the tests, the computation times obtained by state of the art solutions based on A*, RRT* or MPC are at least one order of magnitude greater than the proposed solution.

V. Conclusions and Future Work

We presented an algorithm to find a minimum time deviation from a planned trajectory to connect an initial to a

final configuration for a robotic car avoiding an obstacle. Our strategy splits the problem into three subtasks: the generation of a geometric path made of clothoids, its validation by checking the absence of collisions and the evaluation of its quality in terms of the travel time, taking into account the dynamic characteristics of the vehicle. The algorithm works performing a deterministic search of the best trajectory between sampled points on a way line collocated in proximity of the obstacle. The algorithm is very efficient and can be realistically implemented on an embedded platform. Several topics remain open to future investigations. We mention below two of the most promising.

Parallelisation The algorithm has been conceived to make the most of a multi–core architecture. Indeed, during the processing of each block, a number of independent trajectories must be generated, and for each of them the optimal time has to be computed. All these operations are inherently independent and can be executed in parallel. Moreover, the throughput of the proposed algorithm, in terms of number of blocks processed per unit time, can be highly increased using thread pipelining. This possibility is the result of the modular structure of our approach and is shared with classic monolithic approaches for motion planning. Due to the simplicity of the computations involved, a further step will employ embedded GPU platforms, which are now brought to embedded systems for the automotive field, to speed up the execution time by orders of magnitude.

Iterative Refinement An important aspect of the proposed algorithm is the possibility to run it iteratively multiple times in order to refine and improve the final solution, opening to the possibility of considering highly dynamic obstacles. This is possible by selecting, at the refinement iteration, only locations of interest for each way line. Those are in the neighbourhood of the optimal ones found so far. The described action corresponds to the application of a "zoom" on the track, focusing only on promising locations subsets, and hence softening the effect of location quantisation. Of course, the core of the algorithm does not change at each iteration.

REFERENCES

[1] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1258–1276, 2013.

[2] T. Fraichard and A. Scheuer, "From Reeds and Shepp's to continuous-curvature paths," *Robotics, IEEE Transactions on*, vol. 20, no. 6, pp. 1025–1035, Dec 2004.

[3] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *American Journal of mathematics*, pp. 497–516, 1957.

[4] R. Sanfelice, S. Yong, and E. Frazzoli, "On Minimum-time Paths of Bounded Curvature with Position-dependent Constraints," *Automatica*, vol. 50, no. 2, pp. 537–546, 2014.

[5] F. Biral, R. Lot, S. Rota, M. Fontana, and V. Huth, "Intersection support system for powered two-wheeled vehicles: Threat assessment based on a receding horizon approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 805–816, 2012.

[6] E. Bertolazzi, F. Biral, M. Da Lio, A. Saroldi, and F. Tango, "Supporting drivers in keeping safe speed and safe distance: The SASPENCE subproject within the european framework programme 6 integrating project PReVENT," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 525–538, 2010.

[7] J. V. Frasch, A. Gray, M. Zanon, H. J. Ferreau, S. Sager, F. Borrelli, and M. Diehl, "An auto-generated nonlinear mpc algorithm for real-time obstacle avoidance of ground vehicles," in *Control Conference (ECC), 2013 European*. IEEE, 2013, pp. 4136–4141.

[8] C. Urmson, J. Anhalt, H. Bae, J. A. Bagnell, and et. al., "Autonomous driving in urban environments: Boss and the Urban Challenge," *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.

[9] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path Planning for Autonomous Vehicles in Unknown Semi-structured Environments," *The International Journal of Robotics Research*, vol. 29, no. 5, pp. 485–501, 2010.

[10] Y. Kuwata, S. Karaman, J. Teo, E. Frazzoli, J. How, and G. Fiore, "Real-Time Motion Planning With Applications to Autonomous Urban Driving," *IEEE Trans. on Control Systems Technology*, vol. 17, no. 5, pp. 1105–1118, Sept 2009.

[11] J. J. Kuffner and S. M. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 2. IEEE, 2000, pp. 995–1001.

[12] S. Karaman, M. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime motion planning using the RRT*," in *Robotics and Automation (ICRA), 2011 IEEE Int. Conf.*, May 2011, pp. 1478–1483.

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

[14] S. J. Anderson, S. C. Peters, T. E. Pilutti, and K. Iagnemma, "An optimal-control-based framework for trajectory planning, threat assessment, and semi-autonomous control of passenger vehicles in hazard avoidance scenarios," *International Journal of Vehicle Autonomous Systems*, vol. 8, no. 2-4, pp. 190–216, 2010.

[15] A. Gray, Y. Gao, T. Lin, J. K. Hedrick, H. E. Tseng, and F. Borrelli, "Predictive control for agile semi-autonomous ground vehicles using motion primitives," in *2012 American Control Conference (ACC)*, June 2012, pp. 4239–4244.

[16] K. Berntorp and F. Magnusson, "Hierarchical predictive control for ground-vehicle maneuvering," in *Proceedings of the 2015 American Control Conference*, 2015, pp. 2771–2776.

[17] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1:43 scale rc cars," *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 628–647, 2015. [Online]. Available: http://dx.doi.org/10.1002/oca.2123

[18] E. Velenis and P. Tsiotras, "Minimum-Time Travel for a Vehicle with Acceleration Limits: Theoretical Analysis and Receding-Horizon Implementation," *Journal of Optimization Theory and Applications*, vol. 138, no. 2, pp. 275–296, 2008.

[19] M. Frego, P. Bevilacqua, E. Bertolazzi, F. Biral, D. Fontanelli, and L. Palopoli, "Trajectory planning for car-like vehicles: A modular approach," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, Dec 2016, pp. 203–209.

[20] N. Nagasaka and M. Harada, "Towards safe, smooth, and stable path planning for on-road autonomous driving under uncertainty," in *Proc. IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, Nov 2016, pp. 795–801.

[21] M. Frego, E. Bertolazzi, F. Biral, D. Fontanelli, and L. Palopoli, "Semi-analytical minimum time solutions for a vehicle following clothoid-based trajectory subject to velocity constraints," in *2016 European Control Conference (ECC)*, June 2016, pp. 2221–2227.

[22] E. Bertolazzi and M. Frego, "Semianalytical minimum-time solution for the optimal control of a vehicle subject to limited acceleration," *Optimal Control Applications and Methods*, 2018.

[23] ——, "G1 fitting with clothoids," *Mathematical Methods in the Applied Sciences*, vol. 38, no. 5, pp. 881–897, 2015.

[24] M. Frego, E. Bertolazzi, F. Biral, D. Fontanelli, and L. Palopoli, "Semi-analytical minimum time solutions with velocity constraints for trajectory following of vehicles," *Automatica*, vol. 86, pp. 18–28, 2017.

[25] P. Bevilacqua, M. Frego, E. Bertolazzi, D. Fontanelli, L. Palopoli, and F. Biral, "Path planning maximising human comfort for assistive robots," in *2016 IEEE Conference on Control Applications (CCA)*. IEEE, 2016, pp. 1421–1427.

[26] E. Bertolazzi and M. Frego, "Interpolating clothoid splines with curvature continuity," *Mathematical Methods in the Applied Sciences*, vol. 41, no. 4, pp. 1723–1737, 2018.

[27] P. Bevilacqua, M. Frego, D. Fontanelli, and L. Palopoli, "Reactive Planning for Assistive Robots," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1276–1283, April 2018.