Smooth Trajectory Planning for Fully Automated Passengers Vehicles: Spline and Clothoid Based Methods and its Simulation

Larissa Labakhua[1], Urbano Nunes[2], Rui Rodrigues[2] and Fátima S. Leite[2]

[1] University of Algarve, Escola Superior de Tecnologia/ADEE, Faro, Portugal
llabak@ualg.pt
[2] Institute of Systems and Robotics, University of Coimbra, Coimbra, Portugal
urbano@isr.uc.pt, ruicr@isec.pt, fleite@mat.uc.pt

Abstract. A new approach for mobility, providing an alternative to the private passenger car, by offering the same flexibility but with much less nuisances, is emerging, based on fully automated electric vehicles. A fleet of such vehicles might be an important element in a novel individual, door-to-door, transportation system to the city of tomorrow. For fully automated operation, trajectory planning methods that produce smooth trajectories, with low associated accelerations and jerk, for providing passenger's comfort, are required. This chapter addresses this problem proposing an approach that consists of introducing a velocity planning stage to generate adequate time sequences for usage in the interpolating curve planners. Moreover, the generated speed profile can be merged into the trajectory for usage in trajectory-tracking tasks like it is described in this chapter, or it can be used separately (from the generated 2D curve) for usage in path-following tasks. Three trajectory planning methods, aided by the speed profile planning, are analysed from the point of view of passengers' comfort, implementation easiness, and trajectory tracking.

Keywords. Trajectory planning, splines, clothoids, trajectory tracking, fully-automated vehicles, passenger comfort.

1 Introduction

Negative side effects of car use in build-up areas jeopardise the quality of life. Technology driven inventions like cybernetic transport systems may contribute to sustainable urban mobility. In this context, a new approach for mobility providing an alternative to the private passenger car, by offering the same flexibility but with much less nuisances, is emerging, based on fully automated electric vehicles, named cybercars [2] [7]. A fleet of such vehicles might be an important element in a novel individual, door-to-door, transportation system to the city of tomorrow. These vehicles must be user-friendly, easy to handle and safe, not only for passengers but also for the other road users. These vehicles are already in operation in specific environments featuring short trips at low speed [1] [7].

For fully automated operation, trajectory planning methods that produce smooth trajectories, with low associated accelerations and jerk, are required. Although motion

planning of mobile robots has been thoroughly studied in the last decades, the requisite of producing trajectories with minimum accelerations and jerk (integrating both lateral and longitudinal accelerations) has not been traceable in the technical literature. A global minimum-jerk trajectory planning approach is proposed in [8] but in the context of joint space trajectories of robot manipulators.

This chapter addresses the problem of generating smooth trajectories with low associated accelerations, proposing an approach that consists of introducing a velocity planning stage to generate adequate time sequences to be fed into the interpolating curve planners. The generated speed profile can be merged into the trajectory for usage in trajectory-tracking tasks like it is described in this chapter, or it can be used separately (from the generated 2D curve) for usage in path-following tasks [9]. Three trajectory planning methods used to generate smooth trajectories from a set of waypoints, embedding a given speed profile, are analysed from the point of view of passengers' comfort, easiness of implementation, and trajectory tracking performance. The trajectory-planning methods studied are the following ones: cubic spline interpolation, trigonometric spline interpolation, and a combination of clothoids, circles and straight lines. For its evaluation, the well-known Kanayama trajectory-tracking controller was used [4]. The kinematics model of a Robucar vehicle (platform used in our autonomous navigation experiments) is used to evaluate through simulations the studied trajectory planning methods.

2 Acceleration Effects on the Human Body

For a vehicle following a trajectory at speed v, accelerations are induced on the passengers, which can be expressed as

$$\boldsymbol{a} = \frac{d\boldsymbol{v}}{dt} = \frac{dv}{dt}\boldsymbol{e}_T + v\frac{d\theta}{dt}\boldsymbol{e}_N \tag{1}$$

where $\boldsymbol{v}$ denotes the longitudinal velocity (tangent to the trajectory), θ is the vehicle orientation, and $\boldsymbol{e}_T$ and $\boldsymbol{e}_N$ are unit vectors in the tangent and normal trajectory directions, respectively. Moreover

$$\frac{d\theta}{dt} = \frac{1}{\rho}v \tag{2}$$

where ρ is the curvature radius. From (1) and (2) one gets the longitudinal acceleration (tangential component), induced by variations in speed,

$$a_T = \frac{dv}{dt} \tag{3}$$

and lateral accelerations (normal component), originated by changes in vehicle's orientation, whose values are also affected by the vehicle speed:

$$a_L = \frac{d\theta}{dt}v = \frac{1}{\rho}v^2 \tag{4}$$

The lateral acceleration is function of the trajectory curvature and speed. Assuming constant speed, the smaller is the curvature the smaller is the induced lateral acceleration, and therefore less harmful effects on the passengers. The ISO 2631-1 standard (Table 1) relates comfort with the overall r.m.s. acceleration, acting on the human body, defined as

$$a_w = \sqrt{k_x^2 a_{wx}^2 + k_y^2 a_{wy}^2 + k_z^2 a_{wz}^2} \tag{5}$$

where a_{wx}, a_{wy}, a_{wz}, are the components of the r.m.s. acceleration w.r.t. x, y, z axes and k_x, k_y, k_z, are multiplying factors. For a seated person $k_x = k_y = 1.4$, $k_z = 1$. For motion on the xy-plane, $a_{wz} = 0$. The local coordinate system is chosen so that its x-axis is aligned with the longitudinal axis of the vehicle, and it's y-axis defines the trajectory lateral direction.

2.1 Speed Profile

Trajectory planning for passenger's transport vehicles must generate smooth trajectories with low associated accelerations and jerk. As expressed by (3) and (4), lateral and longitudinal accelerations depend on the vehicle's speed. Thus, the trajectory planner should not only generate a smooth curve (spatial dimension) but also its associated speed profile (temporal dimension).

Table 1 ISO 2631-1 Standard

Overall acceleration	Consequence
$a_w < 0.315$ m/s^2	Not uncomfortable
$0.315 < a_w < 0.63$ m/s^2	A little uncomfortable
$0.5 < a_w < 1$ m/s^2	Fairy uncomfortable
$0.8 < a_w < 1.6$ m/s^2	Uncomfortable
$1.25 < a_w < 2.5$ m/s^2	Very uncomfortable
$a_w > 2.5$ m/s^2	Extremely uncomfortable

Using Table 1 and (5), for "not uncomfortable" accelerations, the longitudinal and lateral r.m.s. accelerations must be less than 0.21 m/s^2. Speed profiles can be calculated under this constraint, and consequently appropriate time-interval values sequences obtained to be used by the curve planners. Assuming a constant speed and a perfect arc cornering with a radius r, the reference speed in corners (segment between waypoints i and j) is

$$v_{ij} \leq \sqrt{a_T \cdot r},\ a_T \leq 0.21 \text{ m/s}^2 \tag{6}$$

It makes sense to consider a straight course segment just before each corner for reducing speed, and others after corners for increasing speed. So, the reference speed on the straight segments begin (end), designated by the waypoint k, can be calculated as

$$v_k = v_i \pm a_L \Delta t, \quad a_L \leq 0.21 \text{ m/s}^2 \tag{7}$$

$$\Delta t = \sqrt{2l / a_L} \tag{8}$$

where the waypoint *i* designates the corner begin (end), and is the straight segment length.

Figure 1 shows an urban road way with very close corners, a roundabout, and a set of waypoints defined by stars.

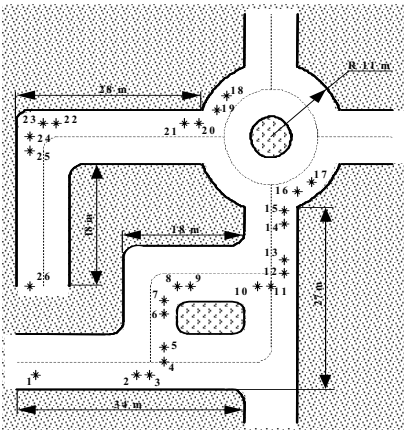

Fig. 1 Urban road way with very close corners, a roundabout, and waypoints defined by stars.

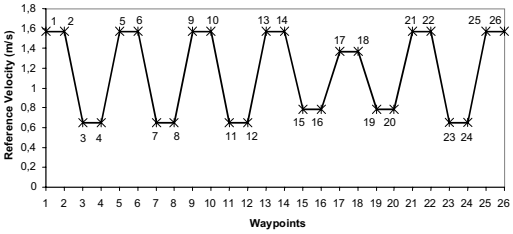

Fig. 2 Speed profile defined by speed values at the waypoints specified in Fig. 1: $v_{ri}, i = 1, 2, \ldots, 26.$

For the purpose of comparison of the three trajectory planning algorithms, applied to the scenario depicted in Fig. 1, and somehow to observe the above acceleration constraints, it was empirically defined the speed profile shown in Fig. 2. A simple algorithm to generate a C^1 speed profile curve using a second order polynomial is presented in [9] which is a step in an iterative trajectory planning method that generates smooth curves with bounded associated accelerations.

3 Kinematics Model

Cybercars are expected to be used in urban areas, airport terminals, pedestrian zones, etc., i.e. in places where the vehicle will move at relatively low speed. Therefore, kinematics-based trajectory control can be considered.

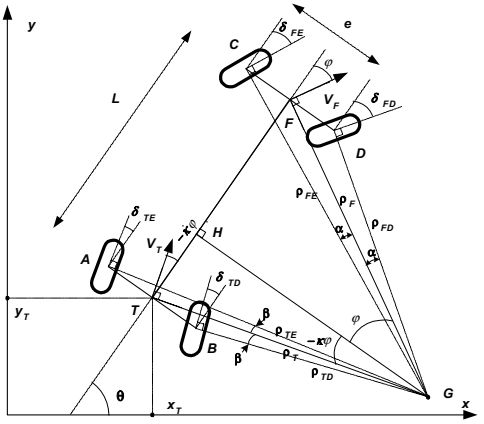

Fig. 3 Kinematics model of a 4-wheel car-like vehicle with front and rear steering capability.

These vehicles are under-actuated systems with two controls, speed and steering angle, but evolving in a $3D$ configuration space $\{x, y, \theta\}$, the first 2 coordinates for the $2D$ position and the third for the vehicle orientation. A representation of the kinematics model of Robucar (bi-steerable, 4-wheels actuated vehicle manufactured by Robosoft) is shown in Fig. 3. The model shows the possibility to steer both the rear and front pairs of wheels. The rear steering angle is proportional by a factor $-k$ to the front steering angle. If the angle φ represents the front wheels' steering command, the back wheels will be deflected from the central axis of the vehicle by an angle –kφ. Assuming that the wheels roll without slipping, the rear and front steering angles give the directions of the velocities at points *F* and *T*, respectively. Hence, the position of the instantaneous turning centre of the solid, point *G* in Fig. 3 can be deduced. Using the geometrical model of Fig. 3, the kinematics model of the vehicle, with the possibility of steering both the rear and front wheels, can be derived [11]:

$$\dot{q} = \begin{bmatrix} \dot{x}_T \\ \dot{y}_T \\ \dot{\theta} \\ \dot{\varphi}_F \\ \dot{\varphi}_T \end{bmatrix} = \begin{bmatrix} \cos(\theta - k\varphi_F) \\ \sin(\theta - k\varphi_F) \\ \dfrac{\sin(\varphi_F + k\varphi_F)}{L \cdot \cos(\varphi_F)} \\ 0 \\ 0 \end{bmatrix} \cdot v_T + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ -k \end{bmatrix} \cdot v_2 \tag{9}$$

where L is the vehicle length and v_2 defines the front wheels steering angular speed. The rear wheels steering angular speed is kv_2.

The results shown in Fig. 4 were calculated using model (9). For a given front steering angle φ, the effect of the rear steering angle is shown.

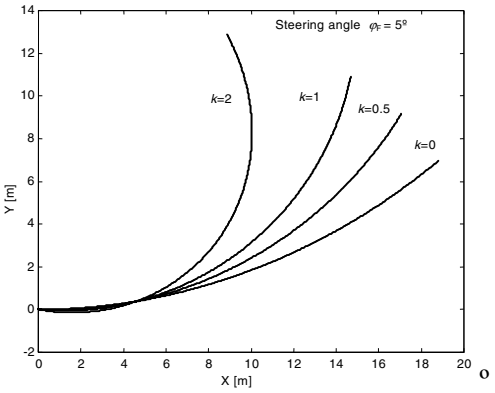

Fig. 4 Car-like vehicle trajectories, using the same front wheel steering angle $\varphi_F = 5^o$ and different values of the rear steering angle, given by the coefficient *k*.

Autonomous vehicles are expected to be used in places such as city centres with narrow areas and wherever it is needed to share the space with pedestrians. So, it is also important to know the position of each wheel, in order to avoid any kind of casualties, sidewalks, etc. Solving the kinematics model (9), and knowing the vehicle length L and its width e, it is possible to derive an output equation for the wheels' positions.

4 Trajectory Planning Methods

4.1 Cubic Splines

We assume that $t_0 < t_1 < \cdots < t_m$ is a chosen partition of the time interval $[t_0, t_m]$, and that $p_0, p_1, \ldots, p_m$ are given distinct points in $\Re^2$. We are interested in the construction of a smooth curve in $\Re^2$ which goes through the point p_k at time t_k, for all $k = 0, 1, \ldots, m$, with prescribed initial and final velocities (v_0 and v_m respectively). The instants of time are chosen in order that the trajectory satisfies a reasonable criterion of performance. Typically, this interpolation problem can be solved by a cubic spline, which is roughly a smooth concatenation of simple cubic polynomial curves. More precisely, a curve $S(t)$, $t \in [t_0, t_m]$, is a cubic spline in $\Re^2$ if it fulfils simultaneously the following:

1. $S(t)$ is defined in each subinterval $[t_k, t_{k+1}]$ by:

$$S_k(t) = \begin{pmatrix} a_{1k} + b_{1k}t + c_{1k}t^2 + d_{1k}t^3 \\ a_{2k} + b_{2k}t + c_{2k}t^2 + d_{2k}t^3 \end{pmatrix} \tag{10}$$

2. $S(t)$ is C^2–smooth in $[t_0, t_m]$, i.e., $S(t)$, $S'(t)$, $S''(t)$ are continuous functions in $[t_0, t_m]$;

3. $S(t_k) = p_k$, $k = 0, \ldots, m$ (interpolation conditions);

4. $S'(t_0) = v_0$ and $S'(t_m) = v_m$ (boundary conditions).

All the coefficients in (10) are uniquely determined by solving a set of linear algebraic equations arising from conditions 2–4. The cubic spline $S(t)$ is a smooth concatenation of each spline segment $S_k(t)$ and thus uniquely computed [3].

4.2 Trigonometric Splines

An alternative to build a C^2-smooth trajectory in a two-dimensional environment satisfying all the requirements at the beginning of this section is based on the construction of a trigonometric interpolating curve, described in [6] and [10]. This curve is again obtained by putting together smaller pieces (spline segments). However, one particular but important feature of this construction is that each piece can be computed separately. As a consequence, one may reduce the computations of each spline segment to the time interval $[0,1]$, thus simplifying notations.

The piece connecting point p_k (at $t = 0$) to point p_{k+1} (at $t = 1$) is denoted by $S_k(t)$ and given by the following convex combination of two other curves, $L_k(t)$ and $R_k(t)$:

$$S_k(t) = \cos^2(\pi t/2)\, L_k(t) + \sin^2(\pi t/2)\, R_k(t).$$

The curves L_k and R_k are called respectively the left component and the right component of the spline segment and will be computed from the local data as follows. The name "trigonometric spline" is suggested by the expression which defines the spline components.

- Computation of L_k ($k \neq 0$):

If the points p_k, p_{k+1} and p_{k-1} define a straight line, then $L_k(t)$ is the line segment connecting p_k (at $t = 0$) to p_{k+1} (at $t = 1$). Otherwise, consider the circle defined by the 3 points and let $L_k(t)$ be the circular arc joining p_k (at $t = 0$) and p_{k+1} (at $t = 1$) that does not contain p_{k-1}.

- Computation of R_k ($k \neq m$):

The previous algorithm (for the left component) is also implemented to compute the right component, but uses instead the points p_k, p_{k+1} and p_{k+2}.

The computation of the left component L_0 of the spline segment S_0 and the right component R_m of the spline segment S_m is slightly different. The computation of L_0 requires the use of the prescribed initial direction (at time t_0) in addition to the points p_0 and p_1. For R_m it is required to use the prescribed final direction (at time t_m) besides the points p_{m-1} and p_m. More details can be found in [10]. Properties of the trigonometric spline:

a. The final curve is guaranteed to be C^2-smooth;

b. The procedure used to compute L_0 and R_m shows how to compute a trigonometric spline when directions are prescribed at each instant of time t_k. This is

an important issue in trajectory planning in a real environment. However, in this case S will no longer be C^2-smooth;

c. Another important property is due to the fact that only four data points are used to compute each spline segment. This is of particular importance in real trajectory planning. Indeed, under the presence of an unpredictable change of a data point (resulting, for instance, from the appearance of a sudden obstacle), at most the two previous and the two following segments of the spline, have to be recalculated. This contrasts with the classical cubic spline, mentioned previously, which would have to be entirely recalculated.

4.3 Clothoids

Using clothoid curves it is also possible to produce smooth trajectories with smooth changes in curvature (see Fig. 5). Clothoids allow smooth transitions from a straight line to a circle arc or *vice versa*. The clothoid curvature can be defined as in [5] by:

$$k(s)=\sigma s+k_0\,, \tag{11}$$

where σ is the curvature derivative, k_0 the initial curvature, s the position variable $s\in[0,\,l]$, and l the curve length. The orientation angle at any clothoid point is obtained integrating (11):

$$\theta(s)=\int_0^s k(u)du=\frac{\sigma}{2}s^2+k_0 s+\theta_0 \tag{12}$$

where θ_0 is the initial orientation angle. The parametric equations of a clothoid in the xy-plane are given by:

$$\begin{bmatrix}x\\ y\end{bmatrix}(s)=r_l\sqrt{2\pi\theta_l}\,R(\theta_0-\theta_0')*\left(\begin{bmatrix}CF\left(\sqrt{\frac{2\theta'(s)}{\pi}}\right)\\ SF\left(\sqrt{\frac{2\theta'(s)}{\pi}}\right)\end{bmatrix}-\begin{bmatrix}CF\left(\sqrt{\frac{2\theta_0'}{\pi}}\right)\\ SF\left(\sqrt{\frac{2\theta_0'}{\pi}}\right)\end{bmatrix}\right)+\begin{bmatrix}x_0\\ y_0\end{bmatrix}, \tag{13}$$

where θ_1 and r_l are respectively the orientation angle and the radius of the clothoid at the point $s=l$, R is a $2D$ rotation matrix, x_0 and y_0 are the co-ordinates of the clothoid at $s=0$, CF and SF denote respectively the cosine and sine Fresnel integrals

$$CF(x)=\int_0^x\cos\left(\frac{\pi}{2}u^2\right)du\,,\quad SF(x)=\int_0^x sen\left(\frac{\pi}{2}u^2\right)du$$

and

$$\theta'(s)=\frac{\sigma}{2}s^2+k_0 s+\theta_0'\,,\ \text{and}\ \ \theta_0'=\frac{k_0^2}{2\sigma} \tag{14}$$

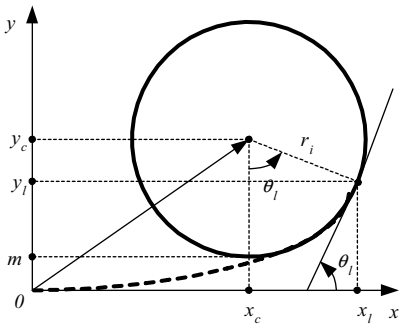

Fig. 5 Transition from a straight line to a circumference arc using a clothoid curve. *m* represents the distance between the straight line and the circumference.

The smooth transition from a straight line to a circumference is shown in Fig. 5, where the x-axis represents a straight line tangent to the clothoid trajectory. The dashed line clothoid curve should handle the smooth transition between the straight line and the circumference arc with centre at (x_c, y_c), radius η and curvature $k=1/r_l$. Solving the equations, one can find the clothoid parameters σ and l:

$$\sigma=\frac{1}{2\eta^2\theta_l} \text{ and } l=2\eta\theta_l \tag{15}$$

The trajectory planning using clothoids is not an interpolation method. The trajectory results from the concatenation of straight line segments, clothoid curves, and circumference arcs.

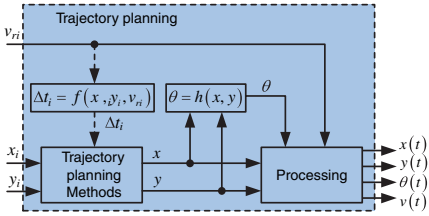

Fig. 6 Trajectory planning module.

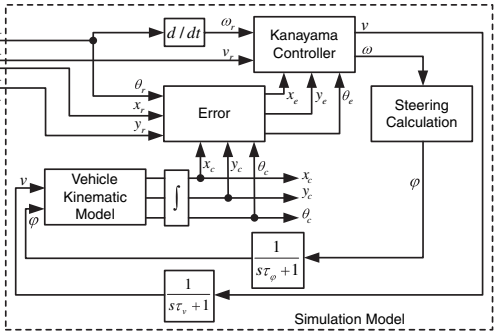

Fig. 7 Simulation model block diagram: trajectory planning and trajectory-tracking modules.

Thus, the trajectory is obtained by means of a geometric construction, and it is not possible to use the prescribed points in the same way as in interpolation methods. A previous processing is needed for assigning new points, circumference arcs radius, and the distance between the straight line segments and circumference arcs.

5 Simulation Model

A simulation numerical model was developed using the MATLAB/SIMULINK programming environment (see Figs. 6 and 7). The first step consisted on calculating the trajectories, from a set of points $p_i = (x_i, y_i)$, $i = 1,2,\ldots,n$, using cubic splines, trigonometric splines and clothoids. These calculations give the reference positions x and y. A time-dependent vector is obtained from the desired trajectory speed values v_{ri} which is used to define the reference variables $x(t), y(t), \theta(t)$ and $v(t)$, as shown in Fig. 6. A trajectory controller must ensure that the vehicle follows the planned reference trajectory. Errors are obtained comparing the reference position with vehicle's position, and a Kanayama controller [4] is used to calculate velocity commands v and ω. The angle φ is calculated in order to model the steering input of a car-like vehicle. For a front wheels only steering, $k = 0$,

$$\varphi = \arctan(\omega \cdot L / v) \tag{16}$$

while for both front and rear wheels steering, and for equal front and rear angles, k=1, results:

$$\varphi = \arcsin(\omega \cdot L / 2v) \tag{17}$$

For other values of k it is more complicated to find the value of angle φ. One possible way is to expand the sine and cosine in Taylor series and solve the resulting equation. The kinematics model (9) is related to the velocity v_T this velocity is in the direction of the rear wheels, as shown in Fig. 3. On the other hand, the target velocity is in the direction of the vehicle axis. Hence,

$$v_T = v / \cos(k\varphi) \tag{18}$$

and the kinematics model becomes

$$\begin{bmatrix} \dot{x}_T \\ \dot{y}_T \\ \theta \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & \dfrac{\sin(\varphi_F + k\varphi_F)}{L.\cos^2\varphi_F} \end{bmatrix}^T v \tag{19}$$

Simple first order steering and speed vehicle's model were used in simulations, using time constants τ_φ and τ_v between the reference and the targets angle φ and velocity v (see Fig. 7).

6 Results

Three trajectory planning methods were applied to a set of prescribed waypoints (points defined by stars in Fig. 4). These point locations represent an urban road way with very close corners and a roundabout. As an example, a planned trajectory using trigonometric splines is depicted in Fig. 8. Figures 9 to 14 show results of the trajectory planning and vehicle's path-following for the three trajectories obtained using the planning methods described in Sect. 4.

Figures 9, 11 and 13 show the orientation angle, curvature, and longitudinal and lateral accelerations behaviour. The curvature is a non time-depending parameter, which shows the smoothness of the planned curve. The acceleration results allow an

evaluation of the trajectory comfort. However, the accelerations also depend on linear speed variation. So, using a different speed profile other results would be obtained. Subsequently, the planned trajectories were applied to the simulation model for trajectory tracking, using a Kanayama controller. The tracking errors obtained from the simulation are shown in Figs. 10, 12 and 14. The angle, longitudinal and lateral errors are shown for cubic splines, trigonometric splines and clothoid curves planned trajectories tracking. Table 2 summarises results of the applied trajectory planning methods.

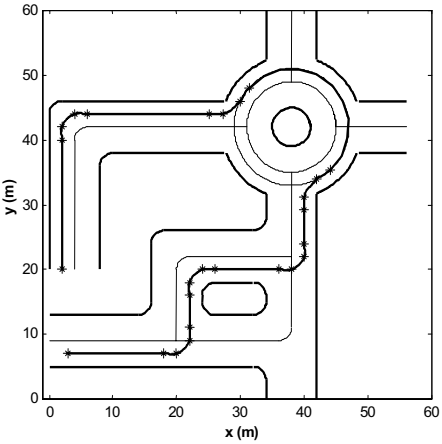

Fig. 8 Generated trajectory using trigonometric splines.

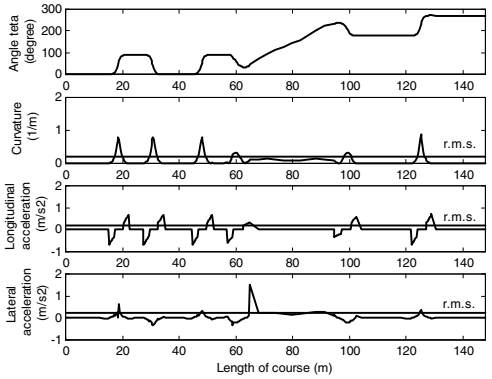

Fig. 9 Orientation angle θ, curvature, longitudinal and lateral acceleration behaviour along the course for the given reference velocity vector, using cubic splines trajectory planning.

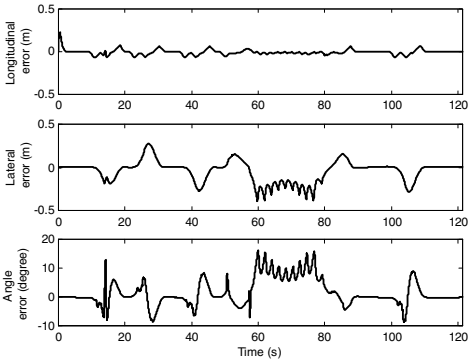

Fig. 10 Angle, longitudinal and lateral tracking errors, using a Kanayama controller and the vehicle kinematics model to follow cubic splines planned trajectory.

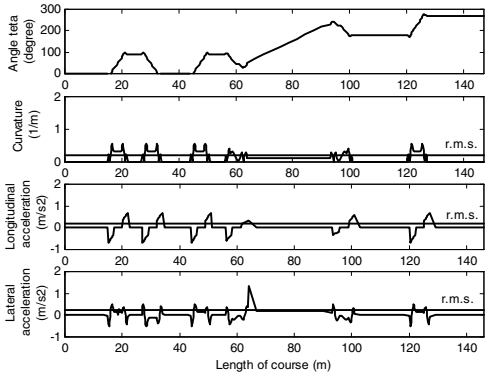

Fig. 11 Orientation angle θ, curvature, longitudinal and lateral acceleration behaviour along the course for the given reference velocity vector, using trigonometric splines trajectory planning.

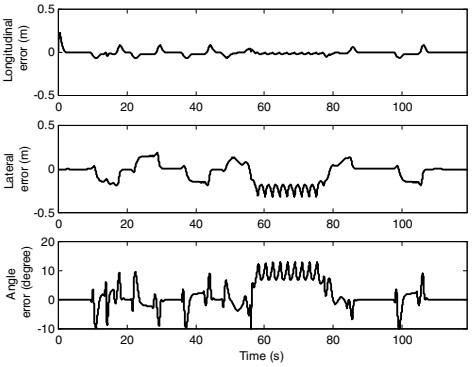

Fig. 12 Angle, longitudinal and lateral tracking errors, using a Kanayama controller and the vehicle kinematics model to follow cubic trigonometric planned trajectory.

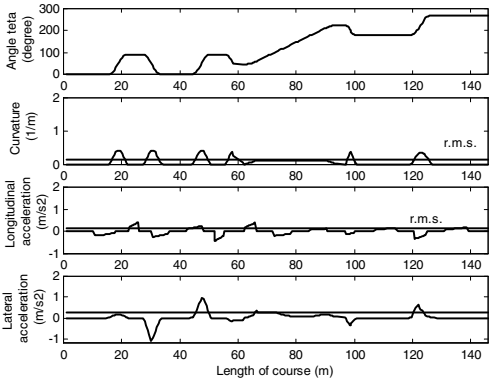

Fig. 13 Orientation angle θ, curvature, longitudinal and lateral acceleration behaviour along the course for the given reference speed profile, using clothoid curves trajectory planning.

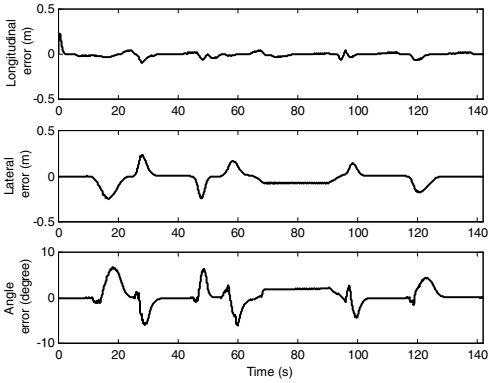

Fig. 14 Angle, longitudinal and lateral tracking errors, using a Kanayama controller and the vehicle kinematics model to follow clothoid curves planned trajectory.

Table 2 Planning methods results

Quantity	Cub.	Trig.	Clothoid
Max. Curvature (1/m)	0.87	0.56	0.41
r.m.s. Curvature (1/m)	0.21	0.20	0.16
Max. Long. Accel. (m/s^2)	0.69	0.69	0.42
r.m.s. Long. Accel. (m/s^2)	0.21	0.21	0.15
Max. Lateral Accel. (m/s^2)	1.50	1.32	0.95
r.m.s. Lateral Accel. (m/s^2)	0.24	0.25	0.25
Overall Acceleration (m/s^2)	0.43	0.46	0.40

7 Conclusions

In this chapter, three trajectories planning methods using cubic splines, trigonometric splines and clothoid curves, were analysed. The integration of a speed profile planner was proposed, with the goal of calculating the time-intervals sequence that lead to low level of accelerations and jerk. Further research is being carried out in this direction [9]. The generated trajectories were applied to a numeric model for trajectory-tracking, using a Kanayama controller. The first conclusion is related to the use of methods easiness. In spite of the relatively good results, the use of clothoid curves is complex and without flexibility in case of trajectory change. On the other hand, all methods showed to be adequate from the point of view of passengers' comfort and tracking.

Acknowledgements

This work was supported in part by ISR-UC and FCT (Fundação para a Ciência e Tecnologia), under contract NCT04: POSC/EEA/SRI/58016/2004.

References

1. Bishop, R., 2005. *Intelligent Vehicle Technology and Trends*, Artech House, London, UK.
2. Cybercars 2001. Cybernetic technologies for the car in the city [online], www.cybercars.org.
3. Gerald, C. and P. Wheatley, 1984. *Applied Numerical Analysis*, Menlo Park, California, Addison-Wesley.
4. Kanayama, Y., Y. Kimura, F. Miyazaky, and T. Noguchy, 1991. A stable tracking control method for a non-holonomic mobile robot. *IEEE/RSJ Int. Conference on Intelligent Robots and Systems (IROS 1991)*.
5. Leao, D., T. Pereira, P. Lima, and L. Custódio, 2002. Trajectory planning using continuous curvature paths. *Journal DETUA*, Vol. 3, no. 6 (in Portuguese), 557–564.
6. Nagy, M. and T. Vendel, 2000. Generating curves and swept surfaces by blended circles. *Computer Aided Geometric Design*, Vol. 17, 197–206.
7. Parent, M., G. Gallais, A. Alessandrini, and T.Chanard, 2003. CyberCars: review of first projects. *Int. Conference on People Movers APM 03*, Singapore.
8. Piazzi, A. and A. Visioli, 2000. Global minimum-jerk trajectory of robot manipulators. *IEEE Transactions on Industrial Electronics*, Vol. 47, no. 1, 140–149.
9. Solea, R. and U. Nunes, 2006. Trajectory planning with velocity planner for fully-automated passenger vehicles. *IEEE 9th Intelligent Transportation Systems Conference*, Toronto, Canada.
10. Rodrigues, R., F. Leite, and S. Rosa, 2003. On the generation of a trigonometric interpolating curve in $\Re^3$. *11th Int. Conference on Advanced Robotics*, Coimbra, Portugal.
11. Sekhavat, S. and J. Hermosillo, 2000. The Cycab robot: a differentially flat system. *IEEE/RSJ Int. Conf. on Intelligent Robots and System (IROS 2000).*