TRAJECTORY PLANNING METHODS FOR AUTONOMOUS CAR-LIKE VEHICLES

Larissa Labakhua[1)], Urbano Nunes[2)], Rui Rodrigues[2)], Fatima Leite[2)]

[1)]*University of Algarve - Escola Superior de Tecnologia/ADEE, Faro, Portugal*
[2)]*Institute of Systems and Robotics, University of Coimbra, Coimbra, Portugal*

Abstract: A fleet of small individual electric vehicles, which can run automatically without direct human intervention, could be a solution for the downtown transportation. For this kind of vehicles it will be necessary an effective trajectory planning method, to ensure the mobility with safety and comfort.
In this paper three trajectory planning methods are analysed from the point of view of passengers' comfort, implementation facility, and trajectory tracking.

Keywords: Trajectory planning, Splines, Curves, Controller, Vehicle simulation.

1. INTRODUCTION

The car traffic enlargement in downtown cities has harmful effects to their inhabitants. The major problems are air pollution, accidents, and parking place congestion. A public transportation system with a pool of small electric vehicles could solve the mobility problem downtown.

A fleet of such vehicles could be an important element in a novel individual, door-to-door, urban transportation system available in closed areas, as city centres, tourist resorts, campuses, etc.

To open utilization to different users, not having necessarily driving licence, these vehicles must be friendly, easy to handle and secure, not only for the driver but also for the pedestrians walking in the same area.
The progresses of computing, electronics, robotics, and automatic control allow envisaging that these vehicles could run automatically without direct human intervention (Lisowsky et al., 1997; Corridori et al. 2004; Nunes et al., 2005). For automatic running, a trajectory planning method is necessary.

Motion planning of mobile robots has been thoroughly studied over the past 25 years. However, results obtained on these platforms can not be directly applied to car-like vehicles (Sekhavat et al., 2000), since its steering capability, dimensions, speed, and human transport utilization introduces further complexity in the configuration and in the time derivatives of its variables. Some of these different parameters are shown in Table 1.

Table 1 Car-Like Vehicle Characteristics (Lisowsky et al., 1997)

Symbol	Quantity	Value
L	Overall length:	1.8 m
e	Overall width:	1.2 m
V_{max}	Maximal speed:	30 km/h
	Actuated wheels:	4
	Steering	Bi-steerable

In this paper, three off-line methods of trajectory planning are studied: cubic spline interpolation, trigonometric spline interpolation, and a combination of clothoids, circles and straight lines. These off-line planning methods are necessary for the generation of smooth trajectories passing through a set of prescribed points, called way points.

A trajectory controller must ensure that the vehicle follows the planned trajectory, which acts as a reference for the real vehicle position feedback. For the trajectory planning evaluation, we can use and adapt one of the well-known trajectory controllers, developed for mobile robots. The controller in Kanayama (1990) seems adequate for this purpose.

The evaluation of trajectory planning methods can be done from several points of view: program complexity, number of way points, processing time, trajectory length and errors, and passengers comfort.

Table 2 Overall r.m.s. Acceleration Consequence - I.S.O. 2631-1

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

The standard ISO 2631-1 relates the comfort with the overall root mean square (r.m.s.) acceleration, acting in the human body, as shown in Table 2. In the standard, the overall r.m.s. acceleration is defined as

$$a_w = \sqrt{k_x^2 a_{wx}^2 + k_y^2 a_{wy}^2 + k_z^2 a_{wz}^2} \qquad (1)$$

where a_{wx}, a_{wy}, a_{wz}, are the components of the r.m.s. accelerations (with respect to the x,y,z axes respectively), and k_x, k_y, k_z, are multiplying factors. For a seated person k_x, = k_y = 1.4, k_z = 1. For the trajectory planning on x,y-plane, $a_{wz} = 0$.

This local coordinates system is chosen so that the x-axis is the longitudinal trajectory direction, and y-axis is the lateral trajectory direction. The longitudinal acceleration is related to the vehicle speed variation along the trajectory. The lateral acceleration is related to (smoothing corners) smoothest of cornering.

A kinematics model of a car like vehicle is used to verify the studied trajectory planning methods for an urban route.

2. KINEMATIC MODEL

Autonomous car-like vehicles (cybercars (Nunes et al.,2003)) are expected to be widely used in urban areas, airport terminals, pedestrian zones, etc, i.e, in places where the vehicle must run at relatively low velocity. Therefore, kinematics-based trajectory control can be considered. This kind of robots has less controls then configuration variables. For a car-like vehicle moving in the x,y-plane, the two controls are the linear and the angular velocities of the steering wheels, although the configuration is described by 3 coordinates $\{x,y,\theta\}$, the first 2 for the position and the last for the orientation.

A representation of the kinematics model of bi-steerable, four-wheels actuated vehicle is shown in the Fig. 1. 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, represented by the point G, can be deduced.

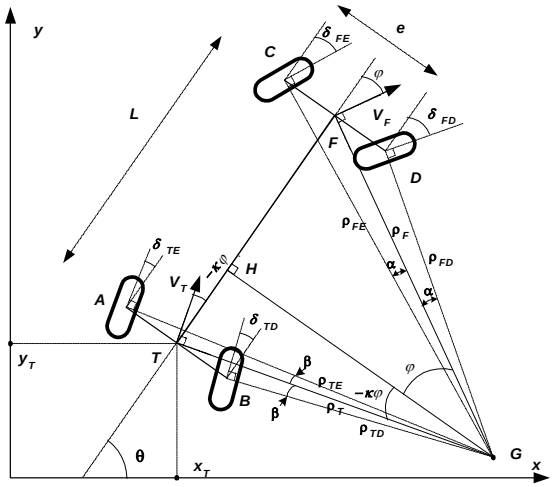

Fig. 1. Kinematics model of a four-wheels car-like vehicle with rear steering possibility.

The following two equations can be obtained from the kinematics model represented in Fig. 1 (see (Sekhavat et al., 2000)), where L is the vehicle length.

$$\overline{TH} = \overline{HF} \cdot \frac{\tan(k\varphi)}{\tan(\varphi)} \qquad (2)$$

$$\overline{TH} + \overline{HF} = L\,. \qquad (3)$$

So,

$$\overline{TH} = L \cdot \frac{\cos(\varphi)\sin(k\varphi)}{\sin(\varphi + k\varphi)} \qquad (4)$$

$$\overline{HF} = L \cdot \frac{\cos(k\varphi)\sin(\varphi)}{\sin(\varphi + k\varphi)}\,. \qquad (5)$$

Hence, the rotation radius around point G, ρ_F and ρ_T can be calculated from (4) and (5).

$$\rho_F = L \cdot \frac{\cos(k\varphi)}{\sin(\varphi + k\varphi)} \qquad (6)$$

$$\rho_T = L \cdot \frac{\cos(\varphi)}{\sin(\varphi + k\varphi)}. \quad (7)$$

All points of the vehicle turn around *G* with the same angular velocity $\dot{\theta}$, so

$$\dot{\theta} = \frac{v_T}{\rho_T} = \frac{v_F}{\rho_F} \Rightarrow v_T = v_F \frac{\rho_T}{\rho_F}, \quad (8)$$

where v_F and v_T are the linear velocities at the points *F* and *T*.

From (6) to (8)

$$v_T = v_F \frac{\cos\varphi}{\cos(k\varphi)} \quad (9)$$

$$\dot{\theta} = v_T \frac{\sin(\varphi + k\varphi)}{L \cdot \cos(\varphi)}. \quad (10)$$

From the projections of v_T on the axis, the point T and the components of the velocity, $\dot{x}_T$ and $\dot{y}_T$, are obtained. Previous expressions allow writing (11), which represents the nonholonomic model of the car-like vehicle, with the possibility of steering both the rear and front wheels. The steering velocity of the front wheel is the angular velocity v_2 of the wheel, around a vertical axis. The rear wheel steering angular velocity is $-kv_2$.

$$\dot{\mathbf{q}} = \begin{bmatrix} \dot{x}_T \\ \dot{y}_T \\ \dot{\theta} \\ \dot{\varphi}_F \\ \dot{\varphi}_T \end{bmatrix} = \begin{bmatrix} \cos(\theta - k\varphi_F) \\ \sin(\theta - k\varphi_F) \\ \dfrac{\sin(\varphi_F + k\varphi_F)}{L \cdot \cos(\varphi_F)} \\ 0 \\ 0 \end{bmatrix} \cdot v_T + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ -k \end{bmatrix} \cdot v_2 . \quad (11)$$

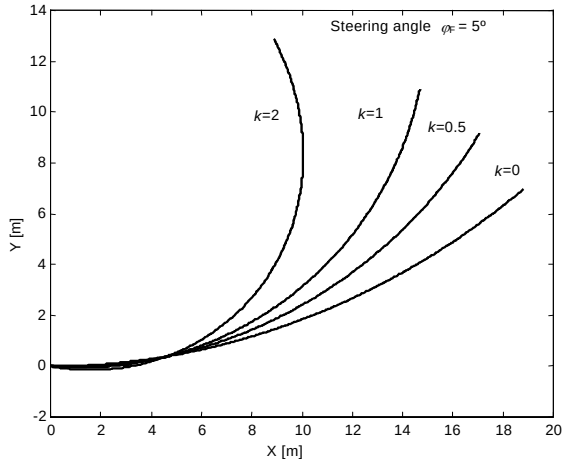

Fig. 2. Car-like vehicle trajectories, using the same front wheel steering angle φ_F = 5º and different values of the rear steering angle, given by the coefficient *k*.

The results showed in the graphics of Fig. 2 were calculated, using the nonholonomic model (11). Here, for the same front angle steering φ, the effect of the rear wheel steering is shown.

Autonomous vehicles are expected to be used in places such as city centers 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 given by (11), and knowing the vehicle length *L* and its width *e*, it is possible to derive an output equation for the wheels' positions.

3. TRAJECTORY PLANNING METHODS

1.1 Cubic splines

A cubic spline is a curve that results from the smooth concatenation of cubic polynomial curves. Here we assume that $t_0 < t_1 < \cdots < t_m$ is a given partition of the time interval $[t_0, t_m]$, $p_0, p_1, \ldots, p_m$ are points in $\Re^2$ used as way points to define a trajectory at those instants of time, and v_0, v_m are the inicial and final velocities.

A function $S : [t_0, t_m] \rightarrow \Re^2$ is a cubic spline in $\Re^2$ if it fulfils simultaneously the following:

1) S is a C^2–smooth function, that is, S and its derivatives S' and S'' are continuous in the time interval $[t_0, t_m]$;

2) S is in each subinterval $[t_k, t_{k+1}]$ defined by

$$S_k(t) = \begin{pmatrix} S_{kx}(t) \\ S_{ky}(t) \end{pmatrix} = \begin{pmatrix} a_{1k} + b_{1k}t + c_{1k}t^2 + d_{1k}t^3 \\ a_{2k} + b_{2k}t + c_{2k}t^2 + d_{2k}t^3 \end{pmatrix}, \quad (12)$$

where a_{ik}, b_{ik}, c_{ik} and d_{ik}, $i = 1,2$, are real constants to be found;

3) S satisfies the interpolation conditions $S(t_k) = p_k$, $k = 0,1,\cdots,m$;

4) S satisfies the boundary conditions $S(t_0) = v_0$, $S(t_m) = v_m$.

The computation of the real constants a_{ik}, b_{ik}, c_{ik} and d_{ik}, $i = 1,2$, $k = 0,1,\cdots,m-1$, follows from solving a set of linear equations. The cubic spline curve is uniquely determined and can be seen as a smooth concatenation of each spline segment $S_k(t)$ (Gerald et al.,1984).

1.2 Trigonometric splines

Another alternative to build a C^2–smooth trajectory in a two-dimensional environment is based on the construction of a trigonometric interpolating curve, described in (Nagy et al.,2000), (Rodrigues et al.,2003). The explicit formulas involve trigonometric functions and, for that reason, such interpolating functions are called trigonometric splines. As before, let $t_0 < t_1 < \cdots < t_m$ be a given

partition of the time interval $[t_0, t_m]$ and $p_0, p_1, \ldots, p_m$ given distinct points in $\Re^2$. The trigonometric spline curve, denoted by S, satisfies the interpolation conditions, i.e., $S(t_k) = p_k$, $k = 0, \ldots, m$, and results from the smooth concatenation of m spline segments.

A particular feature of this spline is that each segment can be calculated separately, and consequently, when computing each segment one may simplify notations and always work in the time interval $[0,1]$. Let S_k be the segment connecting point p_k (at $t = 0$) to point p_{k+1} (at $t = 1$). S_k is defined by the following convex combination of two vector functions L_k and R_k:

$$S_k(t) = \cos^2(\pi t/2)\, L_k(t) + \sin^2(\pi t/2)\, R_k(t)\,. \quad (13)$$

The functions L_k and R_k are denoted respectively as the left component and the right component of the spline segment and will be computed from the local data as follows.

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 doesn't contain p_{k-1}.

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

The algorithm described above is also implemented to compute the right component, but it uses instead the points p_k, p_{k+1} and p_{k+2}.

The computation of the left component L_0 of the segment S_0 and the right component R_m of the 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 (Rodrigues et al.,2003).

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 time instant 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.

1.3 Clothoids

Using clothoid curves it is also possible to produce smooth trajectories also with smooth changes in curvature. Clothoids allow smooth transitions from a straight line to a circle arc or *vice versa*. The clothoid curvature can be defined as in (Leao et al.,2002) by:

$$k(s) = \sigma s + k_0\,, \quad (14)$$

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 (16):

$$\theta(s) = \int_0^s k(u)du = \frac{\sigma}{2}s^2 + k_0 s + \theta_0 \quad (15)$$

where θ_0 is the initial orientation angle.

The parametric equations of a clothoid in the *xy*-plane are given by:

$$\begin{bmatrix} x \\ y \end{bmatrix}(s) = r_l \sqrt{2\pi\theta_l}\, R(\theta_0 - \theta_0^{'}) * \left(\begin{bmatrix} CF\left(\sqrt{\frac{2\theta^{'}(s)}{\pi}}\right) \\ SF\left(\sqrt{\frac{2\theta^{'}(s)}{\pi}}\right) \end{bmatrix} - \begin{bmatrix} CF\left(\sqrt{\frac{2\theta_0^{'}}{\pi}}\right) \\ SF\left(\sqrt{\frac{2\theta_0^{'}}{\pi}}\right) \end{bmatrix} \right) + \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}, \quad (16)$$

where θ_l and r_l are respectively the orientation angle and the radius of the clothoid at the point $s=l$, R is a plane rotation with angle $(\theta_0 - \theta_0^{'})$, 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

$$CF(x) = \int_0^x \cos\left(\frac{\pi}{2}u^2\right)du\,, \quad (17)$$

$$SF(x) = \int_0^x sen\left(\frac{\pi}{2}u^2\right)du\ ,\quad \forall x \in \Re \qquad (18)$$

and

$$\theta'(s) = \frac{\sigma}{2}s^2 + k_0 s + \theta_0' \quad \text{and} \quad \theta_0' = \frac{k_0^2}{2\sigma}\ . \qquad (19)$$

Solving the equations, one can find the clothoid parameters σ and l

$$\sigma = \frac{1}{2 r_i^2 \theta_l} \qquad \text{and} \qquad l = 2 r_i \theta_l\ . \qquad (20)$$

4. SIMULATION MODEL

A simulation numerical model was developed using the MATLAB/SIMULINK programming environment.

The first step consisted on calculating the trajectories, from a set of points $p_i=(x_i, y_i)$, i=1, 2 …n, using cubic splines, trigonometric splines and clothoids. These calculations give the reference positions x and y. A time vector is obtained from the desired trajectory velocities v_{ri} wich is used to define the time depending reference variables $x(t)$, $y(t)$, $\theta(t)$ and $v(t)$, as shown in Fig. 3.

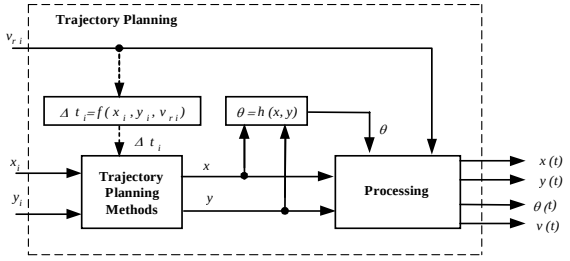

Fig. 3. Time depending reference variables processing.

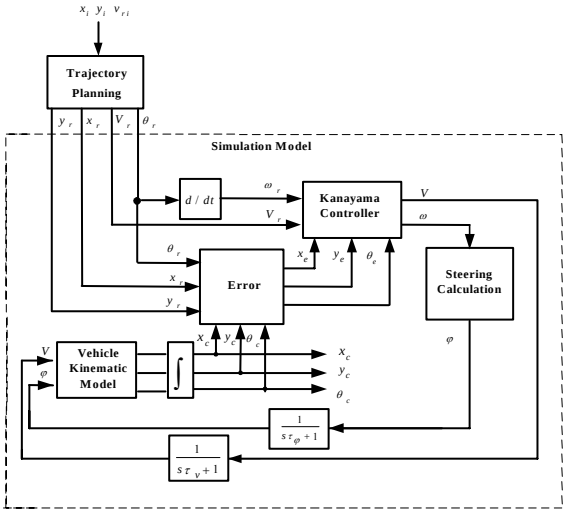

Fig. 4. Simulation model block diagram.

The simulation model block diagram is shown in the Fig.4. A trajectory controller must ensure that the vehicle follows the planned trajectory as a reference, compared with the real vehicle position feedback.

The trajectory error is obtained comparing the reference position with the kinematics model output position. The target velocities v and ω are obtained from a Kanayama controller (Kanayama *et al*., 1990; Leâo *et al*., 2002). The angle φ is calculated in order to model the steering input of a car-like vehicle. For a front wheels only steering, the steering angle calculation is relatively simple. From (11), for k=0,

$$\varphi = \arctan(\omega \cdot L / v)\ . \qquad (21)$$

For both front and rear wheels steering, and for equal front and rear angles, k=1, resulting

$$\varphi = \arcsin(\omega \cdot L / 2v)\ . \qquad (22)$$

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 (11) is related to the velocity v_T. This velocity is in the direction of the rear wheels, as shown in Fig. 1. On the other hand, the target velocity is in the direction of the vehicle axis. Hence,

$$v_T = v / \cos(k\varphi) \qquad (23)$$

and the kinematics model becomes

$$\begin{bmatrix} \dot{x}_T \\ \dot{y}_T \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ \dfrac{\sin(\varphi_F + k\varphi_F)}{L \cdot \cos^2 \varphi_F} \end{bmatrix} \cdot v \qquad (26)$$

A more realistic evaluation of the steering and of the trajectory tracking can be made, using time constants τ_φ and τ_v between the reference and the targets angle φ and velocity v.

5. RESULTS

Three trajectory planning methods were applied to a set of prescribed way points (points defined by stars in Fig.5). 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. 5. Fig. 6 shows in the graphic form, the used reference velocities. Figs. 7 to 12 show behaviour results of the trajectory planning and vehicle's path-following for the three trajectories obtained using the planning methods described in Section 3.

Fig. 7, 9 and 11 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 trajectory. The acceleration results allow an evaluation of the trajectory comfort. However, the accelerations also depend of the longitudinal velocity variation. So, using a different velocity vector other results would be obtained.

Subsequently, the planned trajectories were applied to the simulation model for path-following, using a Kanayama controller. The tracking errors obtained from the simulation are shown in Fig. 8, 10 and 12. The angle, longitudinal and lateral errors are shown for cubic splines, trigonometric splines and clothoid curves planned trajectories tracking.

Table 3 summarises results of the applied trajectory planning methods.

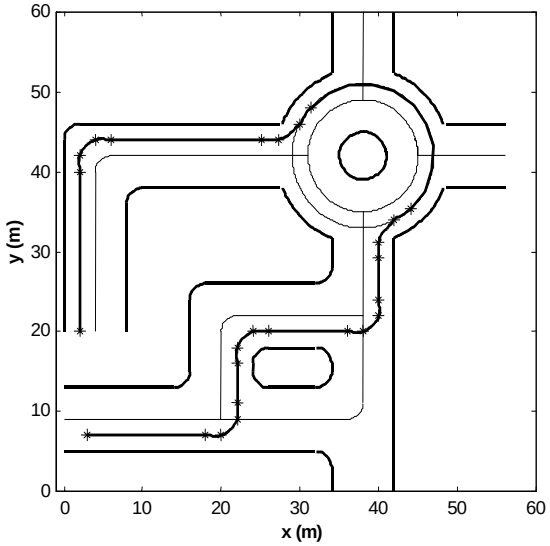

Fig. 5. Planned trajectory using trigonometric splines.

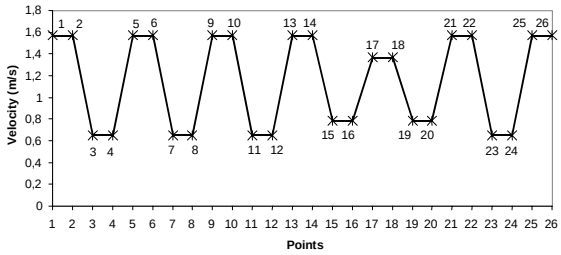

Fig. 6. Reference velocities v_{ri}, i=1, 2,…,26.

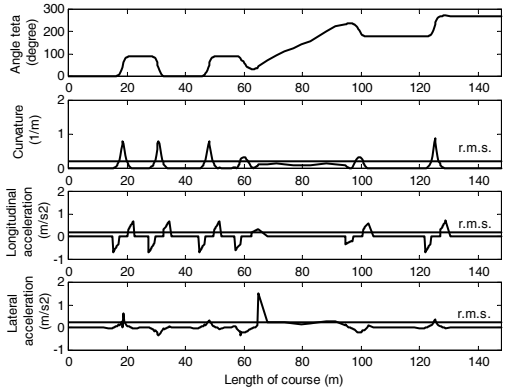

Fig. 7. Orientation angle θ, curvature, longitudinal and lateral acceleration behaviour along the course for the given reference velocity vector, using cubic splines trajectory planning.

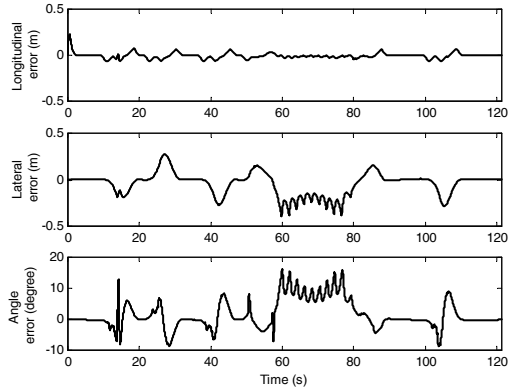

Fig. 8. Angle, longitudinal and lateral tracking errors, using a Kanayama controller and the vehicle kinematics model to follow cubic splines planned trajectory.

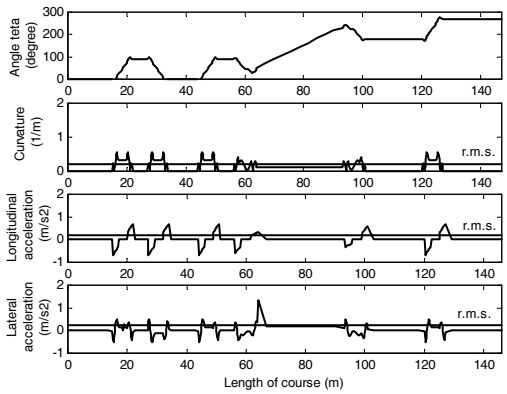

Fig. 9. Orientation angle θ, curvature, longitudinal and lateral acceleration behaviour along the course for the given reference velocity vector, using trigonometric splines trajectory planning.

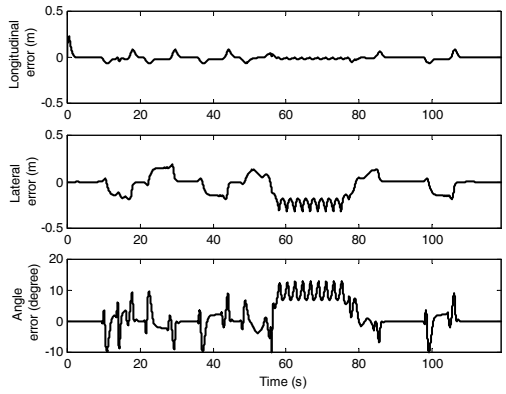

Fig. 10. Angle, longitudinal and lateral tracking errors, using a Kanayama controller and the vehicle kinematics model to follow cubic trigonometric planned trajectory.

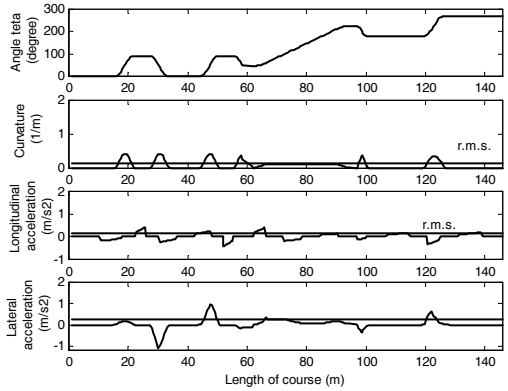

Fig. 11. Orientation angle θ, curvature, longitudinal and lateral acceleration behaviour along the course for the given reference velocity vector, using clothoid curves trajectory planning.

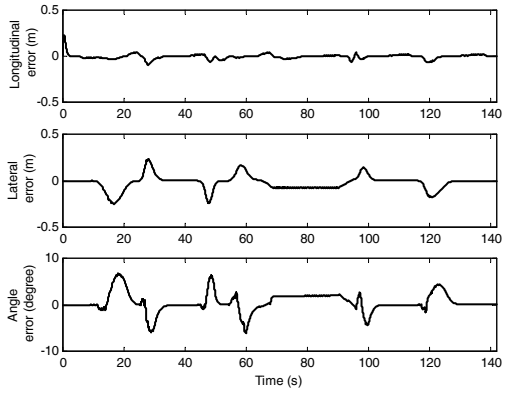

Fig. 12. Angle, longitudinal and lateral tracking errors, using a Kanayama controller and the vehicle kinematics model to follow clothoid curves planned trajectory.

Table 3 Planning Methods Results

Quantity	Cub. Splin.	Trig. Splin.	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

6. CONCLUSIONS

In this paper were analysed three trajectory planning methods using cubic splines, trigonometric splines and clothoid curves. The generated trajectories were applied to a numeric model for path-following, 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.

Finally, the trigonometric splines showed to be a very adequate method to the given utilisation, because of its flexibility in local trajectory planning. However, further simulations and real experiments should be performed to come up with a more complete set of conclusions.

ACKNOWLEDGEMENTS

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

REFERENCES

Bastow, D. (1990). *Car Suspension and Handling*, ISBN-0-7273-0316-3, Billing & Sons Limited, London.

Broch, J. (1984). *Mechanical Vibration and Shock Measurements*, ISBN-87-87355-361, Larsen & Son, Glostrup.

Corridori, C. and M. Zanin (2004). High Curvature Two-Clothoid Road Model Estimation, *IEEE Intelligent Transportation Systems Conference*, Washington, D.C..

Cybercars (2001). Cybernetic technologies for the car in the city. [online], www.cybercars.org.

Dyllong, E. and A. Visioli (2003). Planing and real-time modification of a trajectory using spline techniques, *Robotica, Cambridge University Press*, **Vol. 21**, pp. 475-482.

Gerald, C. and P. Wheatley (1984). *Applied Numerical Analysis*, Menlo Park, California, Addison-Wesley, p. 579.

Gillespie, T. (1992). *Fundamentals of Vehicle Dynamics*, ISBN-56091-199-9, Society of Automotive Engineers, Inc., Warrendale.

Kanayama, Y., Y. Kimura, F. Miyazaky and T. Noguchy (1991). A stable tracking control method for a non-holonomic mobile robot, *IEEE/RSJ International Conference on Intelligent Robots and System*.

Kanayama,Y, Y. Kimura, F. Miyazaky and T. Noguchy (1990). A stable tracking control method for an autonomous mobile robot, *IEEE/RSJ International Conference on Intelligent Robots and System*.

Leao, D., T. Pereira, P. Lima and L. Custódio (2002). Trajectory Planing Using Continuos Curvature Paths , *Journal DETUA*, **Vol.3**, nº6, (in Portuguese).

Lisowsky, L., and G. Baille (1997). Specification of a small electric vehicle: modular and distributed approach, *Proceeding of the IEEE IROS*.

Nagy, M. S. and T.P. Vendel (2000). Generating Curves and Swept Surfaces by Blended Circles, Computer Aided Geometric Design, **Vol. 17**, 197-206.

Nunes, U. and M. Parent (2005). (Guest Editors) Special Issue on Robotics Technologies for Intelligent Road Vehicles, *Autonomous Robots*, Springer-Verlag, **vol.19**(12).

Nunes, U. and M. Parent (2003). (Guest Editors) Special Issue on Robotics Technologies for Intelligent Vehicles, *Machine Intelligence and Robotic Control*, **vol.5 (3)**, Cyber Scientific.

Rodrigues, R., F. Leite and S. Rosa (2003). On the generation of a trigonometric interpolating curve in $\Re^3$, *Proceeding of ICAR 2003*, *The 11th International Conference on Advanced Robotics*, Coimbra, Portugal.

Sekhavat, S. and J. Hermosillo (2000). The cycab robot: a differentially flat system, *IEEE/RSJ International Conference on Intelligent Robots and System*, pp. 312-317.