Control Engineering Practice 73 (2018) 1–12

Contents lists available at ScienceDirect

Control Engineering Practice

journal homepage: www.elsevier.com/locate/conengprac

Autonomous vehicle control using a kinematic Lyapunov-based technique with LQR-LMI tuning

Eugenio Alcala *, Vicenç Puig, Joseba Quevedo, Teresa Escobet, Ramon Comasolivas

Advanced Control Systems Group, Automatic Control Department, Universitat Politècnica de Catalunya (UPC), Campus de Terrassa, Spain

ARTICLE INFO

Keywords:
Lyapunov control
Autonomous vehicle
LPV control
LMI
LQR

ABSTRACT

This work proposes the control of an autonomous vehicle using a Lyapunov-based technique with a LQR-LMI tuning. Using the kinematic model of the vehicle, a non-linear control strategy based on Lyapunov theory is proposed for solving the control problem of autonomous guidance.

To optimally adjust the parameters of the Lyapunov controller, the closed loop system is reformulated in a linear parameter varying (LPV) form. Then, an optimization algorithm that solves the LQR-LMI problem is used to determine the controller parameters. Furthermore, the tuning process is complemented by adding a pole placement constraint that guarantees that the maximum achievable performance of the kinematic loop could be achieved by the dynamic loop. The obtained controller jointly with a trajectory generation module are in charge of the autonomous vehicle guidance. Finally, the paper illustrates the performance of the autonomous guidance system in a virtual reality environment (SYNTHIA) and in a real scenario achieving the proposed goal: to move autonomously from a starting point to a final point in a comfortable way.

1. Introduction

Autonomous vehicles are gaining a huge popularity in society due to the technological innovation and safety increase with regard to current available vehicles. Between these improvements, one of the most important is the driving control system, which is responsible of generating comfortable and safe vehicle motion. In order to achieve such a right movement, a suitable control technique is needed. Model-based controllers are widely employed in many control applications where in the majority of the cases an elaborated modelling task is required.

Over the past decades, a lot of research effort has been dedicated to develop different vehicle models for control purposes. Kinematic models have been broadly used (Alcalá et al., 2016; Blažič, 2010; Rajamani, 2011) as well as lateral dynamic models (Hahn, Zindler, & Jumar, 2016; Rajamani, 2011; Soualmi, Sentouh, Popieul, & Debernard, 2014) and longitudinal dynamic models (Rajamani, 2011).

As it was expected, due to such model research progress many control techniques appear at the same time for solving the control problem in autonomous guidance. In Rajamani (2011), Hahn et al. (2016), Soualmi et al. (2014), Zhang and Wang (2016), Németh, Gáspár, and Bokor (2016) and Nguyen, Sentouh, and Popieul (2016) different types of lateral controls approaches are presented: PI, LPV (Linear Parameter Varying), T–S (Takagi–Sugeno) and MPC (Model Predictive Control).

In Marino, Scalzi, Orlando, and Netto (2009), a PID control approach is suggested for controlling the kinematic part of a vehicle. Kinematic control is also used in Alcalá et al. (2016), Indiveri (1999) based on Lyapunov approach obtaining promising results in slow velocity scenarios.

In the last decades, Lyapunov theory has become a standard rule for analysing stability of non-linear systems (Dixon, Dawson, Zergeroglu, & Behal, 2001; Freeman & Kokotovic, 2008), but also for obtaining model-based strategies for controlling the studied systems (Alcalá et al., 2016; Blažič, 2010; Dixon et al., 2001). In particular, when working with linear parameter varying (LPV) systems, a linear matrix inequality (LMI) expression can be used for checking Lyapunov stability. Such a LMI formalism has become a standard for analysis and control design in recent years (Duan & Yu, 2013).

LPV paradigm (Shamma, 2012) is nowadays considered a suitable strategy for embedding the system non-linearities inside varying parameters obtaining in this way a linear-like representation of a non-linear system. Such a formalism is appropriate to use linear control schemes for designing the controller.

* Corresponding author.
E-mail address: eugenio.alcala@upc.edu (E. Alcala).

https://doi.org/10.1016/j.conengprac.2017.12.004
Received 4 May 2017; Received in revised form 25 September 2017; Accepted 10 December 2017
Available online 9 January 2018
0967-0661/

In this work, a non-linear kinematic Lyapunov-based control is proposed for solving both, the lateral and longitudinal control problem. An optimization algorithm for adjusting non-linear controller parameters is also proposed. This algorithm is based on formulating the closed-loop system in LPV form. Then, the Lyapunov controller parameters are obtained based on LQR-LMI approach. The idea behind the proposed tuning approach is rooted in the work of Farag and Werner (2004), where an approach for fixed structure controller is proposed splitting the problem into a convex and a non-convex sub-problems. A method for solving the convex sub-problem via LMIs is presented in El Ghaoui and Balakrishnan (1994).

In this paper, the trajectory generation, which uses a map and a global planner to compute the best trajectory for reaching the destination, is briefly presented. This trajectory is coarsely defined by a reduced number of global way-points, which are defined by its GPS coordinates and the vehicle orientation. In order to execute the manoeuvres comfortably, a local planner computes a smooth trajectory by adding intermediate local way-points defined by their GPS position, orientation and the desired linear and angular velocities.

Finally, the proposed techniques for vehicle motion control are first tested in a virtual reality environment (SYNTHIA). Then, a real on-field test scenario using an electric Tazzari vehicle is used for showing effectiveness in real conditions.

The paper is structured as follows: Section 2 presents and describes the electric Tazzari vehicle considered in the real scenarios. Section 3 introduces the vehicle model. The control design approach and its tuning are presented in Section 5. Section 4 describes the trajectory planning task. The simulation and experimental results are shown and commented in Sections 6 and 7. Finally, conclusions are stated in Section 8.

2. Vehicle description

The results presented in this paper are part of the project called *Elektra*[1] that aims to develop an autonomous vehicle. For such purpose, an electric Tazzari zero vehicle (Tazzari electric vehicle, 0000) is used (see Fig. 1). This system is a non-holonomic platform that can move like a normal road vehicle. This platform is composed by a set of sensors and actuators, as well as a PC and an electronic control unit (ECU) that manage all algorithms and communications between them. The diagram of the control architecture is depicted in Fig. 2. On one hand, the vehicle has on board an IMU–GPS and stereo cameras to obtain information about the environment and current state. Proper algorithms have to be employed in order to convert that crude information on convenient data for understanding the environment and localize the vehicle. On the other hand, a set of actuators are employed to perform the motion (steering and driven electric motors) as well as turning on the lights and opening doors. The rest of modules in Fig. 2 (perception, localization, planning and control) compose the software for performing the autonomous guidance task. This paper specially focuses on the non-linear automatic control module. However, the trajectory planning task is introduced for better understanding.

All the algorithms involved run over a trunk PC (6-core i7 5930 K, 32 GB DDR4) running ROS on GNU/Linux (Ubuntu distribution). An NVIDIA GTX Titan X board is used to run GPU-based algorithms for perception-image analysis.

The ECU, based on a Cortex-M4 MCU, runs a custom embedded software which communicates the PC control actions to the different car actuators (steering, throttle, brake, lights, horn), as well as reads the values of the car state sensors (steering, throttle, brake, speed, doors, battery).

The communication net is based on CAN bus protocol. Its cycle is currently set to 100 ms, which is sufficient for running all required algorithms.

[1] http://adas.cvc.uab.es/elektra/.

Fig. 1. Electric Tazzari Zero vehicle.

3. Vehicle modelling

The behaviour of the vehicle presented in Section 2 can be described by using equations that represent the kinematic and dynamic behaviour. In this section the development of the vehicle kinematic model is addressed for designing the control strategy. Kinematic model is based on the velocity vector movement in order to compute longitudinal and lateral velocities referenced to a global inertial frame.

3.1. Kinematic model

The kinematic model for the vehicle has been derived assuming that behaves as a bicycle-like vehicle. This is a quite standard assumption in the literature (Aicardi, Casalino, Bicchi, & Balestrino, 1995). Kinematic based model is widely used for control design because of its low parameter dependency. This model takes into account *yaw*, x and y motion while neglecting *roll*, *pitch* and z movements. Furthermore, its assumes null skidding and considers small lateral force. These two characteristics share the idea of travelling at low speed. The kinematic equations for the bicycle model are introduced below:

$$\begin{cases} \dot{x} = v\cos(\theta) \\ \dot{y} = v\sin(\theta) \\ \dot{\theta} = \omega \end{cases} \tag{1}$$

where x, y and θ represent the current position and orientation of the vehicle in metres and radians respectively, with respect to the global frame; v is the linear velocity and ω represents the angular velocity of the vehicle.

For developing the kinematic-based controller, an error model has been built. It is defined as the difference between real measurements (x, y and θ) and desired values (x_d, y_d and θ_d). However, this set of errors are expressed with respect to the inertial global frame (x, y in Fig. 3). For control purposes is suitable to express the errors with respect to the vehicle, such that lateral error is always measured in the lateral axis of the vehicle. Thus, a rotation over the road orthogonal axis is considered to represent the errors in the body vehicle frame (x_b, y_b):

$$\begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_d - x \\ y_d - y \\ \theta_d - \theta \end{bmatrix} \tag{2}$$

where subindexes d and e refer to desired and error values, respectively. For developing the error model is needed to take into account a non-holonomic constraint of the form:

$$\dot{x}\sin(\theta) = \dot{y}\cos(\theta) \tag{3}$$

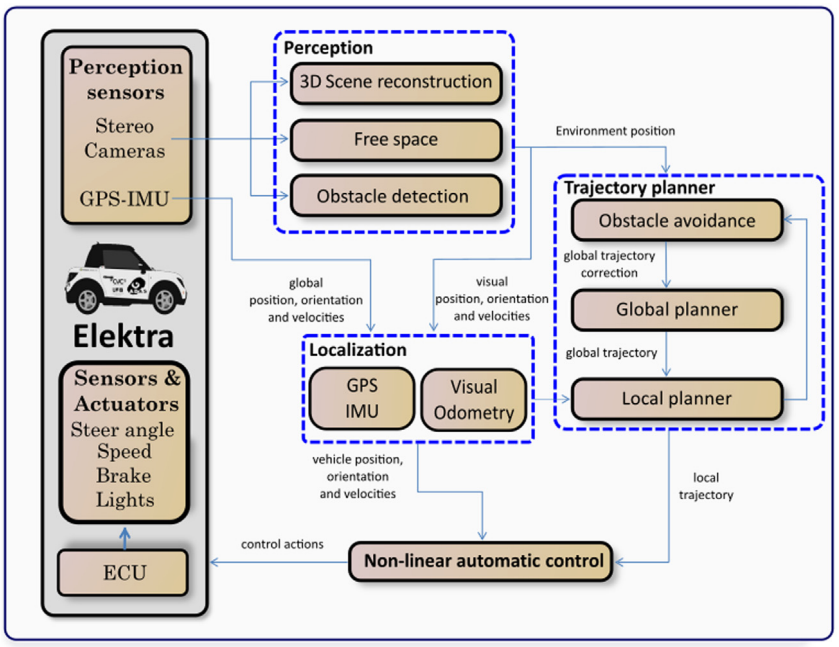

Fig. 2. Elektra control architecture. The large block on the left represents the physical vehicle devices (sensors, actuators and ECU). The rest of the blocks on the image (perception, localization, trajectory planner and automatic control) represent software algorithms running over the PC.

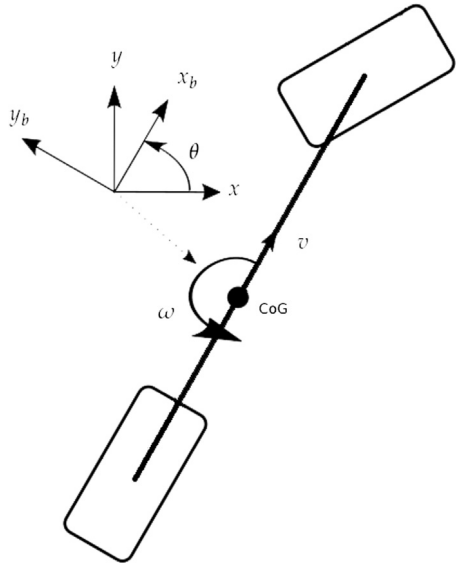

Fig. 3. Two-wheels bicycle model employed for control purposes. x, y frame represents the global inertial frame. The body frame, represented by x_b, y_b, is located in the centre of gravity of the vehicle. Kinematic velocity is represented as v and angular velocity as ω. CoG means centre of gravity.

Hence, computing the time derivative of (2) and using (1) and (3) and some trigonometric identity, we obtain the following open-loop error system (see details in the Appendix):

$$\begin{cases} \dot{x}_e = \omega y_e + v_d \cos\theta_e - v \\ \dot{y}_e = -\omega x_e + v_d \sin\theta_e \\ \dot{\theta}_e = \omega_d - \omega \end{cases} \tag{4}$$

Details about the development of (4) can be found in Chapter 1 of Dixon et al. (2001). In the sequel, the open-loop kinematic error system (4) will be used for the control design purposes.

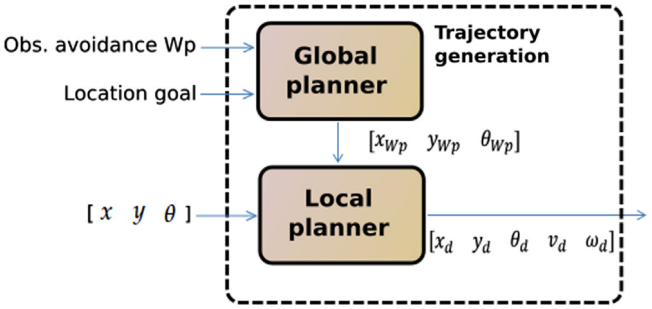

Fig. 4. Trajectory generation module. In case of the perception detects an obstacle the obstacle avoidance module provides a new way-point (Obstacle avoidance way-point) to avoid such an obstacle. Location goal represents the coordinates of the desired final point. x, y and θ are the current states of the vehicle. The sub-index Wp refers to Way-point.

4. Trajectory generation

This section addresses the module responsible of generating the trajectory planning for achieving the desired goal (observe this module in the overall vehicle architecture presented in Fig. 2). Information from other modules, such as obstacle avoidance and localization, is received in order to compute free-collision trajectories. Fig. 4 shows the trajectory generation module and its sub-modules as well as the input and output data. At this point, the vehicle is in charge of managing two planning stages: Global and Local planning. They can be seen as two overlapping and connected layers being the Global planner the upper one. Note that, both planners represent their coordinates and orientation (x, y and θ) with respect to the inertial global frame (x, y).

4.1. Global planner

A human–vehicle interface based on the OpenStreetMap open software is used to introduce the route as a set of way-points along the street (cyan ball in Fig. 5). Moreover, information about close obstacles is introduced with the goal of recomputing the global trajectory. The position of each way-point is provided in the Universal Transverse

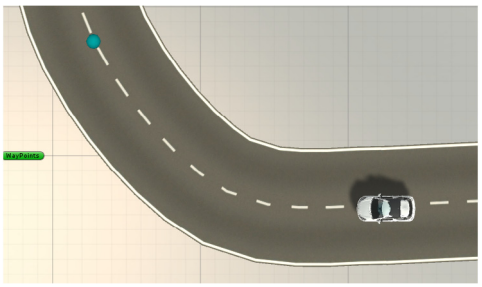

Fig. 5. Global trajectory planning. Cyan point represents one of the global way-points along the route.

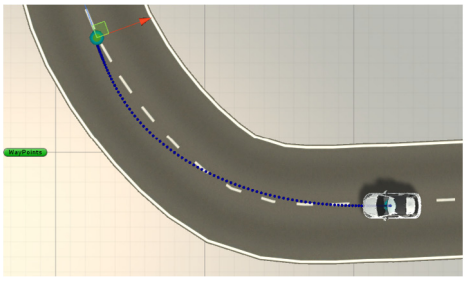

Fig. 6. Local trajectory planning. Small blue points depicts the reference for the controller.

Mercator (UTM) coordinate system. Once the global plan is defined, it provides way-point information (x_{Wp}, y_{Wp}, θ_{Wp}) to the local trajectory planner.

4.2. Local planner

Passenger comfort determines driving quality. The most remarkable variables affecting passenger comfort are the lateral and longitudinal accelerations. High accelerations will annoy passengers, who will find it very difficult to maintain posture. The ISO 2631-1 (International Organization for Standardization, 1997) standard recommends an overall vehicle acceleration (a_w) less than 0.315 $\frac{m}{s^2}$, which is defined as:

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

Following Bianco, Piazzi, and Romano (0000), Solea and Nunes (2007), a quintic spline-based trajectory planner is implemented that generates smooth trajectories with a velocity profile with continuous acceleration and low levels of jerk, ensuring the passenger comfort. Our work adopts a simplified version of such an algorithm: instead of using smooth but variable velocities in straight sections, which is harder for the tracking control task, constant velocity sections are proposed. The algorithm defines three operation modes:

- Acceleration stage: computes a smooth velocity profile under bounded acceleration.
- Constant velocity stage: maintains a constant velocity reference using the control module.
- Deceleration stage: computes a smooth velocity profile under bounded negative acceleration.

This module will provide a set of local way-points to the control module (small blue points in Fig. 6). Each local way-point is defined as a set of desired values (x_d, y_d, θ_d, v_d and ω_d) as it is shown in Fig. 4.

5. Automatic vehicle control

The automatic control strategy tackles the problem of generating an appropriate behaviour as of a desired trajectory. Thus, it is in charge of computing smooth control actions (vehicle speed and steering angle) such that the vehicle is capable of achieving the required speed and orientation at the next local way-point (observe this module in the overall vehicle architecture presented in Fig. 2).

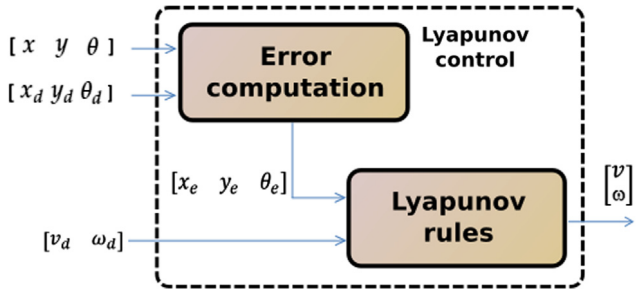

Fig. 7. Lyapunov based controller diagram.

In this section, a nonlinear automatic control strategy based on the Lyapunov theory (Aicardi et al., 1995; Alcalá et al., 2016; Dixon, Dawson, Zergeroglu, & Behal, 2000; Samson & Ait-Abderrahim, 2006) is introduced for trajectory tracking and navigation among way-points as well as a tuning methodology based on LPV LMI based linear–quadratic regulation (LQR) approach.

The idea of the Lyapunov method is to define a control law that ensures the stability and the asymptotic elimination of the error between the real and desired vehicle trajectory. Fig. 7 shows the Lyapunov control sub-modules as well as input and output variables.

The controller receives the set-points of the trajectory provided by the Local planner (x_d, y_d, θ_d, v_d and ω_d) and the current localization of the vehicle (x, y, θ). With all this information, a set of errors (x_e, y_e, θ_e) can be computed by using (2) which will be used by the Lyapunov-based controller.

5.1. Lyapunov control design

The Lyapunov-based controller is designed according to the following Theorem.

Theorem 5.1. *Given the kinematic error model of the vehicle (4), the control law:*

$$\begin{cases} v = k_1 x_e + v_d \cos\theta_e \\ \omega = \omega_d + k_2 v_d \dfrac{\sin\theta_e}{\theta_e} y_e + k_3 \theta_e \end{cases} \tag{6}$$

stabilizes the closed-loop dynamics in the Lyapunov sense if the controller parameters k_1, k_2 and k_3 are positive.

Proof. Following the Lyapunov's stability, the following Lyapunov function candidate is proposed:

$$V(e) = \frac{k_2}{2} x_e^2 + \frac{k_2}{2} y_e^2 + \frac{1}{2}\theta_e^2 \tag{7}$$

Its time derivative is:

$$\dot{V}(e) = k_2 x_e \dot{x}_e + k_2 y_e \dot{y}_e + \theta_e \dot{\theta}_e \tag{8}$$

Now, by substituting the open-loop equations (4) in (8):

$$\dot{V}(e) = k_2 x_e v_d \cos\theta_e - k_2 x_e v + k_2 y_e v_d \sin\theta_e + \theta_e \omega_d - \theta_e \omega \tag{9}$$

By inspection, an expression of the controller in terms of control actions v and ω is determined:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} k_1 x_e + v_d \cos\theta_e \\ \omega_d + k_2 v_d \dfrac{\sin\theta_e}{\theta_e} y_e + k_3 \theta_e \end{bmatrix} \tag{10}$$

such that:

$$\dot{V}(e) = -k_2 k_1 x_e^2 - k_3 \theta_e^2 < 0 \tag{11}$$

fulfilling the Lyapunov's Theorem under the condition that the control parameters satisfy:

$$k_1, k_2, k_3 > 0 \tag{12}$$

Once the control equations have been obtained the closed-loop error system has the following shape by inserting (10) in (4):

$$\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} = \begin{bmatrix} \omega y_e - k_1 x_e \\ -\omega x_e + v_d \sin(\theta_e) \\ -k_2 v_d \dfrac{\sin\theta_e}{\theta_e} y_e - k_3 \theta_e \end{bmatrix} \tag{13}$$

Observe that from (11), the convergence of x_e and θ_e is guaranteed, i.e.

$$\lim_{t\to\infty} x_e(t) = 0 \qquad \lim_{t\to\infty} \theta_e(t) = 0 \tag{14}$$

However, the convergence of y_e is not ensured. In order to demonstrate that $\lim_{t\to\infty} y_e(t) = 0$, the proof relies on the result presented in Theorem 1.2 of Dixon et al. (2000) that shows:

$$\lim_{t\to\infty} \dot{\theta}_e(t) = 0 \tag{15}$$

when using the control law (10). Hence, (15) leads to:

$$\lim_{t\to\infty} k_2 \cdot v_d(t) \cdot \frac{\sin\theta_e(t)}{\theta_e(t)} y_e(t) = 0 \tag{16}$$

considering (14) and (13). And, consequently:

$$\lim_{t\to\infty} y_e(t) = 0 \tag{17}$$

assuming that, when following the desired trajectory, the velocity control action $v_d(t)$ is not null and $\theta_e \cong 0$ such that:

$$\lim_{\theta_e\to 0} \frac{\sin\theta_e}{\theta_e} = 1 \tag{18}$$

Thus, the achievement of the global asymptotic stability can be concluded.

5.2. Lyapunov control adjustment via LQR-LMI

The condition (12) guarantees that the controller is stable, but it does not allow to establish performance specifications. In this section, an iterative algorithm for adjusting the non-linear Lyapunov controller using a LQR-LMI based strategy is proposed.

The method starts by rewriting the closed-loop error system (13) in LPV form, considering the small-angle approximation is used since orientation error remains very close to zero:

$$\frac{\sin\theta_e}{\theta_e} \approx 1 \tag{19}$$

Considering $\omega, v_d \in \mathbb{R}^1$ as the scheduling variables and $\mathbf{K}_s = [k_1\ k_2\ k_3]$:

$$\begin{bmatrix} \dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e \end{bmatrix} = \overbrace{\begin{bmatrix} -k_1 & \omega & 0 \\ -\omega & 0 & v_d \\ 0 & -k_2 v_d & -k_3 \end{bmatrix}}^{\mathbf{A}_{cl}(\mathbf{K}_s,\omega,v_d)} \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} \tag{20}$$

At this point, taken (19) in consideration, the control strategy (6) can be seen as a state feedback control law in the form $\mathbf{u} = \mathbf{K}\mathbf{x} + \mathbf{r}$:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \overbrace{\begin{bmatrix} k_1 & 0 & 0 \\ 0 & k_2 v_d & k_3 \end{bmatrix}}^{\mathbf{K}(\mathbf{K}_s,\omega,v_d)} \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} + \begin{bmatrix} v_d \cos\theta_e \\ \omega_d \end{bmatrix} \tag{21}$$

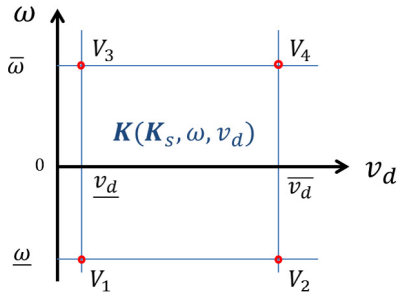

Fig. 8. Bounding box governed by the robust controller $\mathbf{K}(\mathbf{K}_s, \omega, v_d)$. V_i, with $i = 1, \ldots, 4$, represent the vertexes of the bounding box.

The scheduling variables are bounded in a box (see Fig. 8) defined by the operating conditions. The controller is scheduled according to the expression $\mathbf{K}(\mathbf{K}_s, \omega, v_d)$ in (21).

The controller parameters (12) will be determined using the linear–quadratic regulation (LQR) technique via LMI as suggested in Duan and Yu (2013) using the LMI solution for the H_2 problem given by:

$$\begin{aligned}
&\mathbf{A}_{cl}(\mathbf{K}_s,\omega,v_d)\mathbf{P} + (\mathbf{A}_{cl}(\mathbf{K}_s,\omega,v_d)\mathbf{P})^T + 2\alpha\mathbf{P} > 0 \\
&\mathbf{A}_{cl}(\mathbf{K}_s,\omega,v_d)\mathbf{P} + (\mathbf{A}_{cl}(\mathbf{K}_s,\omega,v_d)\mathbf{P})^T < 0 \\
&\begin{bmatrix} -\mathbf{Y} & \mathbf{R}^{\frac{1}{2}}\mathbf{K}(\mathbf{K}_s,\omega,v_d)\mathbf{P} \\ (\mathbf{R}^{\frac{1}{2}}\mathbf{K}(\mathbf{K}_s,\omega,v_d)\mathbf{P})^T & -\mathbf{P} \end{bmatrix} < 0 \\
&trace(\mathbf{Q}^{\frac{1}{2}}\mathbf{P}(\mathbf{Q}^{\frac{1}{2}})^T) + trace(\mathbf{Y}) < \gamma \\
&\mathbf{P} \geq 0,\ \mathbf{Y} = \mathbf{Y}^T > 0 \qquad\qquad for\ v_d \in [\underline{v_d}, \overline{v_d}], \omega \in [\underline{\omega}, \bar{\omega}]
\end{aligned} \tag{22}$$

that is converted to an LMI by means of the following change of variable: $\mathbf{W} = \mathbf{K}(\mathbf{K}_s, \omega, v_d)\mathbf{P}$. However, this procedure would deliver a free structure state feedback controller $\mathbf{K}$, i.e. not keeping the structure of the Lyapunov control law (21). To preserve this structure is not an easy task as discussed in El Ghaoui and Balakrishnan (1994) since leads to a non-convex problem.

Here, to preserve the fixed structure of the control law (21), an optimization problem that has as decision variables the control parameters ($\mathbf{K}_s$) and as objective function the infinity norm of the Lyapunov matrix ($\mathbf{P}$) eigenvalues, is used to maximize the LQR performance as follows:

$$\begin{aligned}
&\min_{\mathbf{K}_s} \quad \|eig(P)\|_\infty \\
&\text{s.t.} \\
&\qquad \mathbf{P} = \text{LQR-LMI-problem}(\{\mathbf{K}_{s,i}\}_{i=1}^4) \\
&\qquad \mathbf{K}_{s,i} \in \left(0, \overline{\mathbf{K}}_{s,i}\right] \qquad\qquad i = 1, \ldots, 4
\end{aligned} \tag{23}$$

where $eig()$ function returns a vector containing the eigenvalues of square matrix P. The function LQR-LMI-problem has as input the set of $\mathbf{K}_s$ vectors and solves the following set of LMIs:

$$\begin{aligned}
&\mathbf{A}_{cl}(\mathbf{K}_{s,i},\omega_i,v_{d_i})\mathbf{P} + (\mathbf{A}_{cl}(\mathbf{K}_{s,i},\omega_i,v_{d_i})\mathbf{P})^T + 2\alpha\mathbf{P} > 0 \qquad for\ i = 1, \ldots, 4 \\
&\mathbf{A}_{cl}(\mathbf{K}_{s,i},\omega_i,v_{d_i})\mathbf{P} + (\mathbf{A}_{cl}(\mathbf{K}_{s,i},\omega_i,v_{d_i})\mathbf{P})^T < 0 \qquad for\ i = 1, \ldots, 4 \\
&\begin{bmatrix} -\mathbf{Y} & \mathbf{R}^{\frac{1}{2}}\mathbf{K}(\mathbf{K}_{s,i},\omega_i,v_{d_i})\mathbf{P} \\ (\mathbf{R}^{\frac{1}{2}}\mathbf{K}(\mathbf{K}_{s,i},\omega_i,v_{d_i})\mathbf{P})^T & -\mathbf{P} \end{bmatrix} < 0 \qquad for\ i = 1, \ldots, 4 \\
&trace(\mathbf{Q}^{\frac{1}{2}}\mathbf{P}(\mathbf{Q}^{\frac{1}{2}})^T) + trace(\mathbf{Y}) < \gamma \\
&\mathbf{P} \geq 0,\ \mathbf{Y} = \mathbf{Y}^T > 0
\end{aligned} \tag{24}$$

Note that, in (23) and (24), i represents each one of the polytope vertexes in Fig. 8, $\mathbf{K}_{s,i}$ represents the i optimization vectors $\mathbf{K}_s \in \mathbb{R}^{1\times 3}$, $\mathbf{K}(\mathbf{K}_{s,i}, \omega_i, v_{d_i})$ is the controller matrix presented in (21), $\mathbf{P} \in \mathbb{R}^{3\times 3}$ is the Lyapunov matrix and the result of the LMI problem, $\mathbf{Q} \in \mathbb{R}^{3\times 3}$, $\mathbf{R} \in \mathbb{R}^{2\times 2}$, $\mathbf{Y} \in \mathbb{R}^{2\times 2}$ and $\gamma \in \mathbb{R}$ are tuning parameters in the LMI-LQR problem. The parameter α represents the boundary for setting the kinematic closed-loop poles (see Fig. 9). Note that, in (23), $\overline{\mathbf{K}}_{s,i}$ is the upper boundary for

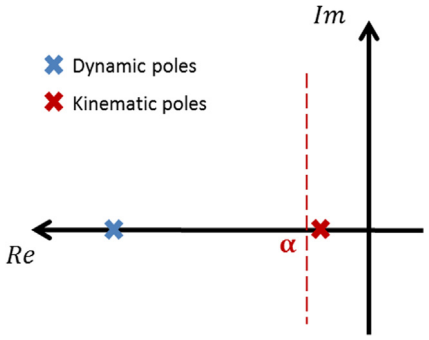

Fig. 9. Representation of the pole placement issue. Dominant kinematic poles (red one) must be four times slower than the dominant dynamic poles (blue one). The hyper-plane is introduced as a LMI in the optimization problem where the parameter α represents its position in the negative real plane. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

the control parameters. Such a boundary has been chosen as an arbitrary very high value in order to ensure the optimal gains are found.

In order to select the constraint for the pole placement α in (24), the consideration that the Lyapunov controller provides the set point to the dynamic loop has to be taken into account. This internal loop has been implemented by using two decoupled PI controllers adjusted by means of the pole placement technique. In order to achieve a good kinematic reference tracking, the dynamic control loop has been considered four times faster than the kinematic one. This leads to locate the kinematic closed loop poles in a specific region between 0 and α (see Fig. 9). Such a restriction is presented in the form of a LMI in the optimization problem (23) as: $\mathbf{A}_{cl}(\mathbf{K}_{s,i}, \omega_i, v_{d_i})\mathbf{P} + (\mathbf{A}_{cl}(\mathbf{K}_{s,i}, \omega_i, v_{d_i})\mathbf{P})^T + 2\alpha\mathbf{P} > 0$.

At this point four controllers ($\mathbf{K}_i$) are considered, one per each vertex of the polytope in Fig. 8. These will be interpolated at every control iteration following the rules shown down below:

$$M_{\underline{v_d}} = \frac{v_d - \underline{v_d}}{\overline{v_d} - \underline{v_d}} \quad M_{\underline{\omega}} = \frac{\omega - \underline{\omega}}{\overline{\omega} - \underline{\omega}} \tag{25a}$$

$$\mu_1(\omega, v_d) = M_{\underline{v_d}} M_{\underline{\omega}} \tag{25b}$$

$$\mu_2(\omega, v_d) = M_{\underline{v_d}}(1 - M_{\underline{\omega}}) \tag{25c}$$

$$\mu_3(\omega, v_d) = (1 - M_{\underline{v_d}})M_{\underline{\omega}} \tag{25d}$$

$$\mu_4(\omega, v_d) = (1 - M_{\underline{v_d}})(1 - M_{\underline{\omega}}) \tag{25e}$$

$$\mathbf{K} = \mu_1\mathbf{K}_1 + \mu_2\mathbf{K}_2 + \mu_3\mathbf{K}_3 + \mu_4\mathbf{K}_4 \tag{25f}$$

It is important to note that due to the high non-convexity of the optimization problem, common gradient-based solvers are not applicable and thus, genetic algorithms have been useful for solving it. This heuristic algorithm does not ensure a global optimal solution but only a local optimal solution.

6. Simulation results

In this section, the behaviour of each module previously introduced is evaluated, i.e. global and local planning, and automatic control. For this purpose the modules have been evaluated in SYNTHIA[2] (Ros, Sellart, Materzynska, Vazquez, & Lopez, 2016). It runs over Unity[3] which is a game development platform.

In this environment, localization is considered ideal and neither noises nor disturbances have been added. However, the point of interest is that vehicle dynamics is modelled in realistic manner considering the complex vehicle physics (Edy's Vehicle Physics, 0000). The global planner defines the route composed with a set of way-points along the scenario (cyan points in Fig. 11). The local trajectory planner has been adjusted and constrains the overall vehicle acceleration as explained in Section 4.2. All algorithms are executed in a regular manner within a period of 0.1 s. The complete diagram for simulation can be seen in Fig. 10.

The section is divided in two simulations tests. On one hand, the Test A is presented. It is composed by a circuit designed for testing the algorithms along the whole range of urban velocities (i.e. 0 – 60 $\frac{km}{h}$). It offers also different curvature curves. On the other hand, Test B is composed by a geometrically simpler circuit in which the vehicle works at lower velocities (i.e. 0 – 18 $\frac{km}{h}$). It is designed with the idea of simulating the real experiment.

Table 1
Test A – control parameters for each vertex of the bounding box (see Fig. 8).

	v_d	ω	k_1	k_2	k_3
$i = 1$	0.1	−1.42	3.9	1.1	1.5
$i = 2$	16.7	−1.42	3.6	1.2	2.1
$i = 3$	0.1	1.42	3.9	1.1	1.5
$i = 4$	16.7	1.42	3.6	1.2	2.1

6.1. Test A

For this test, the scheduling variables have been bounded in the following intervals: $\omega \in [-1.417, 1.417]\frac{rad}{s}$ and $v_d \in [0.1, 16.7]\frac{m}{s}$. The v_d interval starts in $0.1\frac{m}{s}$ since the null velocity is a singular point for the controller and it has to be avoided. Both in simulation and in the real test, the vehicle begins the performance by applying a little force over the rear wheels in order to achieve this low velocity bound and being inside v_d interval. The matrices and variables used for adjusting the LQR optimization are:

$$\mathbf{Q} = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \mathbf{R} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \gamma = 0.0001$$

The control parameters are presented in Table 1 for each bounding box vertex. Note that the controller gains are only function of the linear velocity. This is because of the vehicle geometry is symmetric with respect to its longitudinal axis.

The results of applying the Lyapunov control technique adjusted by means of the LQR-LMI approach are shown in Figs. 12 and 13. Fig. 12 shows the proposed circuit and the trajectory result. Such a circuit has been designed with the idea of offering the vehicle different levels of difficulty. Then, a set of curves with different curvature has been introduced. From Fig. 13a, it can be seen that the velocity profile is divided in two different velocity sections. Observe that it is complex to differentiate reference and response signals. Thus, this means that barely exists velocity error and that the velocity tracking is working correctly. Fig. 13b depicts the evolution of the angular velocity. The reference proposed by the trajectory planner is followed quite well allowing the vehicle to mitigate the possible lateral error that can exist.

Regarding the position errors (see Fig. 13c and d), the longitudinal one reaches a quite low error (around 0.02 m) and the lateral error evolves in the millimetre scale.

An evaluation of the control in terms of quadratic error has been done obtaining 0.0231 m^2 of MSE (Mean Square Error) for the longitudinal position and an amount of 0.0231 m^2 for the lateral position.

6.2. Test B

In this test, due to the differences of the scenario with respect to Test A (i.e. velocity and geometry) the scheduling variables have been bounded in the following intervals: $\omega \in [-1.417, 1.417]\frac{rad}{s}$ and

[2] http://synthia-dataset.net/.
[3] https://unity3d.com/es.

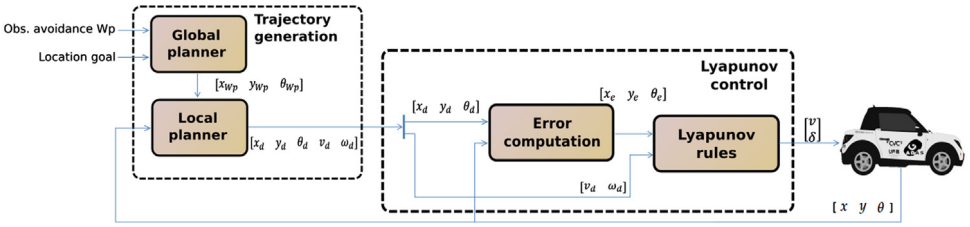

Fig. 10. Complete motion loop diagram composed by the trajectory planning and the Lyapunov control modules. The lower level dynamic control is not depicted in this figure.

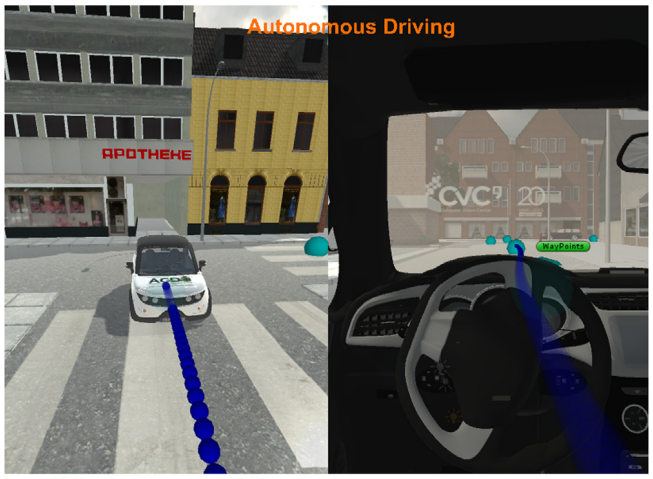

Fig. 11. SYNTHIA scenario: Screen shot of the simulation. The vehicle appears from an external view in the left side and from the driver point of view in the right side.

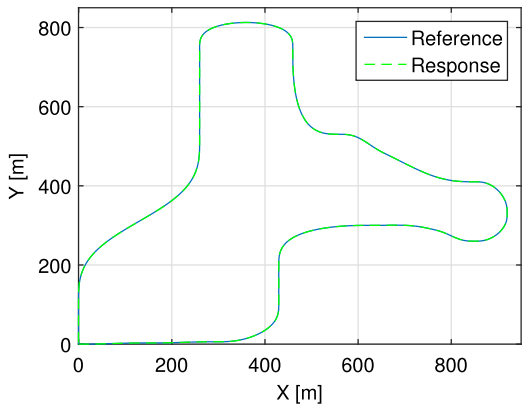

Fig. 12. Results on SYNTHIA (Test A): Desired path and real trajectory followed by means of using the Lyapunov based technique as controller.

$v_d \in [0.1, 5]\frac{m}{s}$. As in Test A, v_d interval starts in $0.1\frac{m}{s}$. The matrices and variables used for adjusting the LQR optimization are:

$$\mathbf{Q} = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \mathbf{R} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \gamma = 0.0001$$

The set of control parameters found for this scenario are presented in Table 2 for each bounding box vertex. The results of applying the Lyapunov control technique adjusted by means of the LQR-LMI approach are shown in Figs. 14 and 15.

Fig. 14 shows the proposed circuit and the trajectory result. This circuit has been proposed with the idea of simulating the real experimental test.

Table 2
Test B – control parameters for each vertex of the bounding box (see Fig. 8).

	v_d	ω	k_1	k_2	k_3
$i = 1$	0.1	−1.42	0.27	0.23	0.31
$i = 2$	5	−1.42	0.78	1.07	1.2
$i = 3$	0.1	1.42	0.27	0.23	0.31
$i = 4$	5	1.42	0.78	1.07	1.2

From Fig. 15a and b, it can be seen that both velocity references are followed presenting low levels of error. However, the relevant signals when performing a trajectory tracking task are the position errors. They are depicted in Fig. 15c and d. Observe that the longitudinal error is in the range of ± 0.05 m and the lateral error evolves in a range of few centimetres.

Furthermore, in order to value the simulation an evaluation of the control performance in terms of quadratic error has been done. The obtained results are 0.0269 m^2 of MSE for the longitudinal position and an amount of 0.0053 m^2 of MSE for the lateral position.

Note that due to the differences in geometry and velocities on both circuits (Tests A and B) they cannot be compared. Test A develops at higher velocity than Test B, but has wider and longer curves than Test B achieving in this manner a lower lateral acceleration in the curves. Hence, we can affirm that lateral error is function of the lateral acceleration that the vehicle suffers when arrives to a curve and that the results are consistent.

Remark (*Programming*). The algorithms in SYNTHIA have been programmed by using *C#* for Unity environment, linking at the same time functions programmed in C++ over Visual Studio 2012.

Remark (*Solvers*). The LMIs have been solved with YALMIP and SeDuMi solvers, while the optimization problem (23) has been solved by using Matlab genetic algorithm "*ga*".

7. Experimental results

Once planning and control systems have been evaluated and the simulations have proved to be satisfactory, a real scenario is used to validate their integrated applicability. The validation results of the complete vehicle behaviour over a real scenario are presented. The test consists in starting from an initial position, reaching a constant velocity while following the desired trajectory and finally stopping in front of a detected pedestrian. The scenario where the test has been performed is a geometrically simple circuit (see Fig. 16.a).

The results of perception, localization and obstacle avoidance modules have been omitted in this work. However, they are used in the real validation (e.g. obstacle avoidance module is in charge of varying the position of the next global way-point if an obstacle is detected). The experimental controller has been adjusted with the same parameters than in simulation (Test B). They can be seen in Table 2.

Remark (*Programming*). For this real scenario, the algorithms have been programmed in C++ over a ROS-Ubuntu platform.

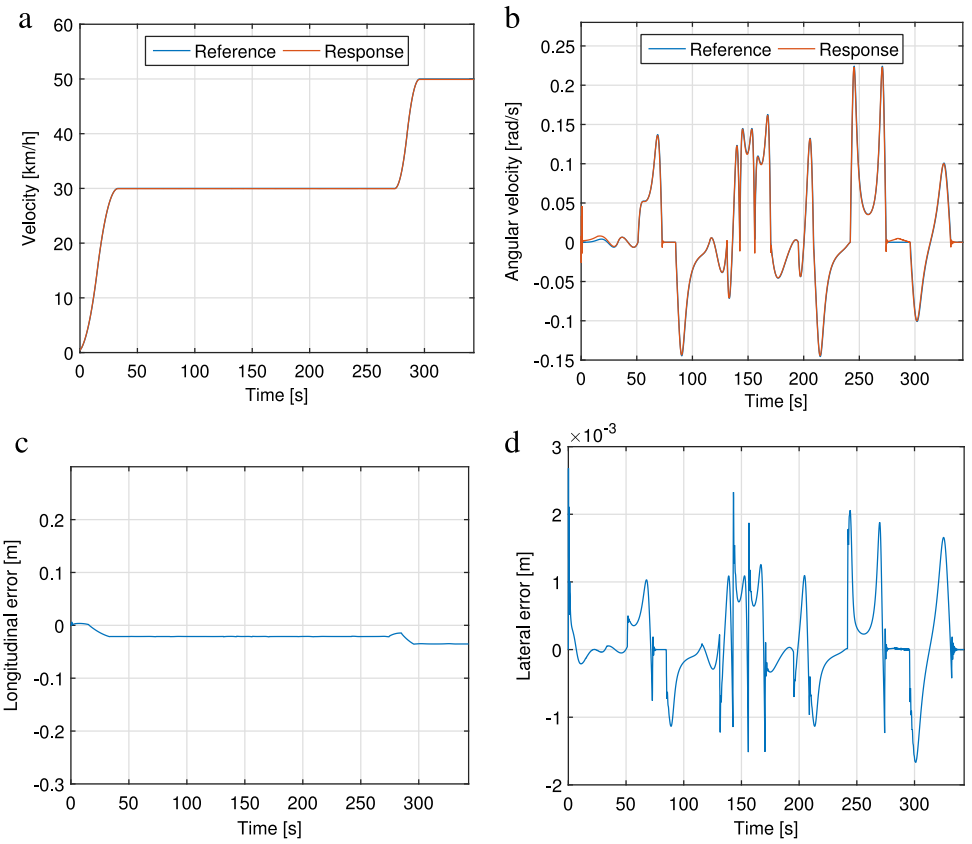

Fig. 13. Results on SYNTHIA (Test A): (a) Linear velocity reference and response. (b) Angular velocity reference and response. (c) Longitudinal error obtained during the simulation. (d) Vehicle lateral error.

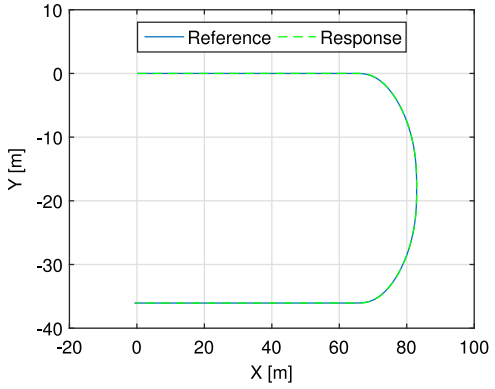

Fig. 14. Results on SYNTHIA (Test B): Desired path and real trajectory followed by means of using the Lyapunov based technique as controller.

The vehicle has two main hardware constraints: the maximum resolution of steering and velocity. On one hand, the steering system has a maximum resolution of two degrees. This is a hard constraint that limits the lateral control. Such limitation produces a nervous steering angle action while trying to achieve the null error. On the other hand, the speed system has a maximum resolution of 1km/h. This issue generates a limitation when controlling the longitudinal behaviour of the vehicle. Thus, it is easy to have an error in the longitudinal speed control and in the longitudinal position.

In Fig. 16b, the real trajectory is shown through rviz ROS tool.[4] It shows multiple data in real time: the stereo visualization, the global way points (green arrow), the completed path (white lines) and the real trajectory (yellow lines).

Fig. 17 shows the resultant behaviour of the system during the experimental test. Fig. 17a depicts how the vehicle follow the velocity reference although being not able to eliminate completely the error in steady state. This issue may be caused by the resolution constraint of 1 km/h and the localization errors as well as the controller adjustment. In this experimental test, the control action measured and sent to the steering actuator is the steering angle. Hence, such a steering variable has been depicted in Fig. 17b. It can also be seen the problem of resolution, i.e. small jumps of 2 degrees, and how the response does not exactly match the reference produced by the trajectory planner. Fig. 17c and d show the longitudinal and lateral errors, respectively. These graphs are used to validate the performance of the vehicle. In this case the response is not exactly the expected one. However, the vehicle is stable and can correct the trajectory at every instant of time in spite of the constraints appointed and localization errors. Finally, Fig. 17e and f represent the signals sent by the controller to the actuators of the vehicle. The Lyapunov controller produces a sharp velocity action trying to reduce the error, while the steering signal is smoother in spite of the huge difference between the reference and the response. Moreover, as in simulation, an evaluation of the control performance in terms of quadratic error has been done. The obtained results are 0.8178 m^2 of MSE for the longitudinal position and an amount of 0.3099 m^2 of MSE for the lateral position. Comparing this results with the ones obtained in simulation Test B, it can be appreciated the huge difference that exists in between of simulation and reality scenarios.

The objective of this experimental test is to follow the trajectory proposed minimizing the lateral and longitudinal errors at the fastest rate possible. The validation is performed graphically and using the Mean Square Error method. We can conclude that the goals have been achieved although with localization and hardware problems. The

[4] http://wiki.ros.org/rviz.

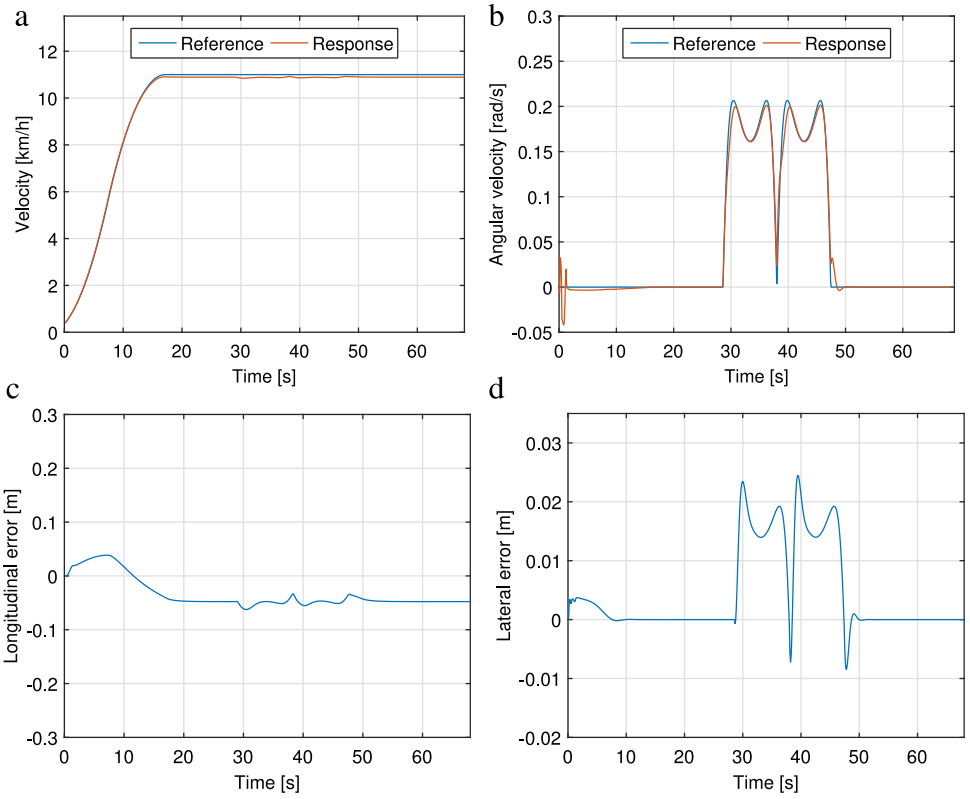

Fig. 15. Results on SYNTHIA (Test B): (a) Linear velocity reference and response. (b) Angular velocity reference and response. (c) Longitudinal error obtained during the simulation. (d) Vehicle lateral error.

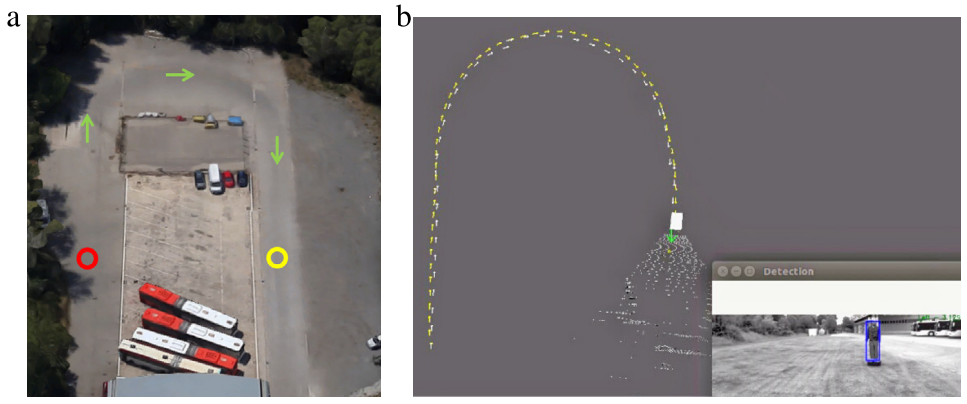

Fig. 16. In field test: (a) Space used to perform tests. Red circle represents the initial position, green arrows are the global way points that define the trajectory and the yellow circle represents the location of a pedestrian. (b) Visualization of test results in rviz tool (ROS) that displays in real time: the cameras visualization, the global way points (green arrows), the desired trajectory (red lines), the completed path (white lines), the real trajectory (yellow lines) and the stereo data (cloud of 3D points). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

vehicle is able to go through the way-points being stable and mitigating the errors. The test was performed 50 times and the goal was achieved in 41 of them. The main problems are due to localization drift. An example video of the vehicle performing in SYNTHIA and real scenarios can be seen in YouTube.[5]

8. Conclusions

In this work, a non-linear control strategy based on Lyapunov theory has been introduced for solving the control problem of autonomous vehicle guidance. This work has also proposed an iterative algorithm for adjusting the parameters of the non-linear controller to achieve not only stability but also performance specifications. This algorithm relies on a LQR-LMI based strategy using a LPV representation of the closed-loop kinematic error model. Furthermore, such an adjustment is fortified by adding a restriction between dominant dynamic poles and dominant kinematic poles for holding a correct physical behaviour. The obtained LPV-Lyapunov controller jointly with a trajectory generation module are in charge of moving the vehicle. It has been presented the performance of the vehicle in simulation obtaining satisfactory results, and it has been achieved the expected goal of moving autonomously from a starting point to a final point in a comfortable way in a real test. As a future work, localization system will be improved to facilitate the mitigation of lateral and longitudinal errors. Planning and control techniques will be enhanced as well as researching other control strategies for autonomous vehicles. Furthermore, different scenarios at different velocities will be

[5] https://www.youtube.com/watch?v=K0omhJXawTo.

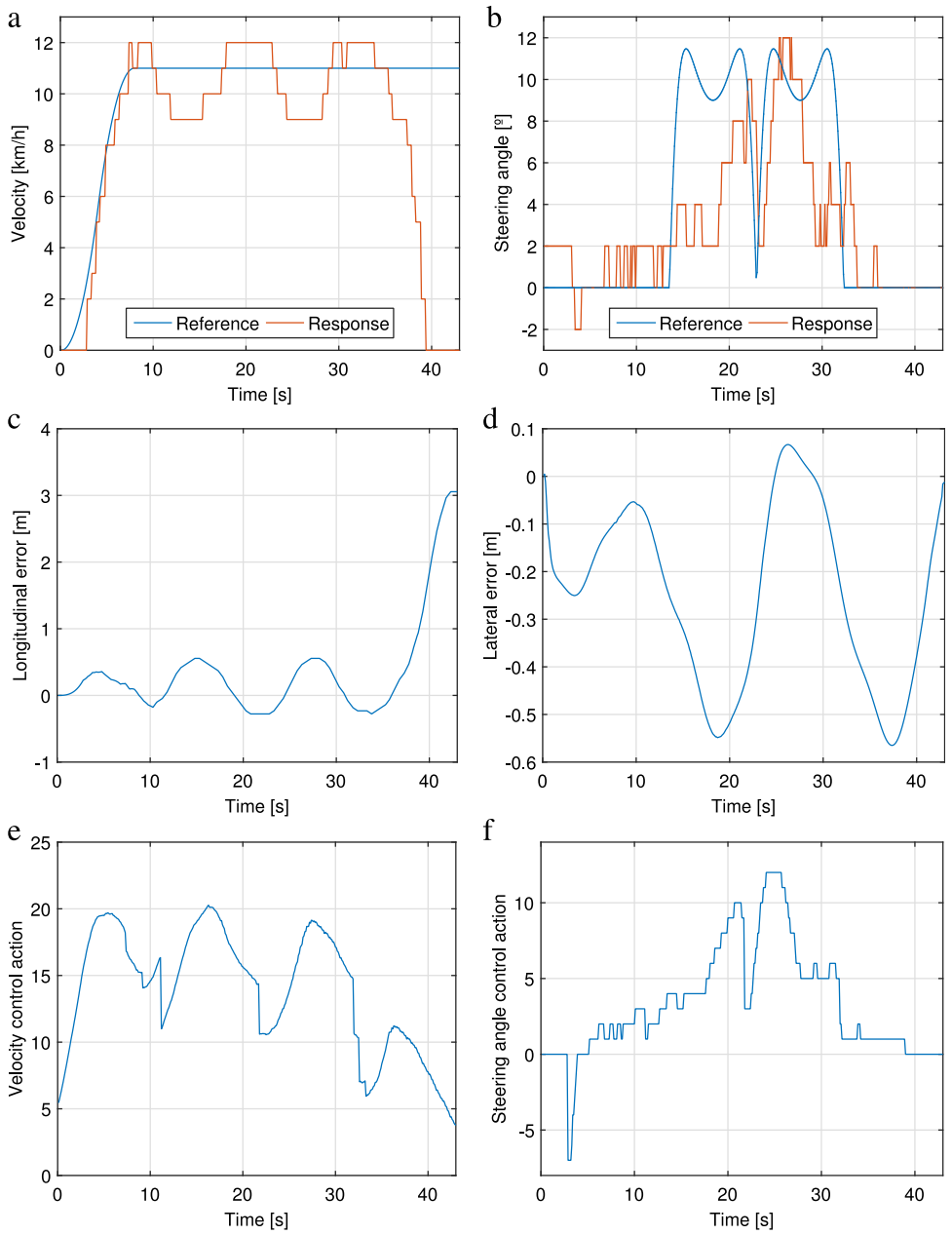

Fig. 17. In field testing results: (a) Velocity reference generated by the trajectory planner and real speed of the vehicle during the test. (b) Steering angle reference and real steering produced after applying the steer angle control action. (c) Longitudinal error achieved during the test. (d) Vehicle lateral error obtained while following the trajectory. (e) Longitudinal velocity control action generated by the controller. (f) Steering angle control action computed by the controller.

tested to extract conclusions about the suitability of the controller for real world application.

Acknowledgements

This work has been funded by the Spanish Ministry of Economy and Competitiveness (MINECO) and FEDER through the projects ECOCIS (Ref. DPI2013-48243. C2-1-R) and HARCRICS (Ref. DPI2014-58104-R). The authors also thank the Computer Vision Centre (CVC), established by the Generalitat de Catalunya and the Universitat Autònoma de Barcelona (UAB), for supporting the real experiments. The author is supported by a FI AGAUR grant (Ref. 2017 FI_B00433).

Appendix. Derivation of the error model

The dynamics of the error posture are what is needed for the trajectory tracking problem. Hence, (2) needs to be differentiated in order to obtain the error model. The equations of motion for each state of the error model are derived here for completeness.

- **Time derivative of x_e.**

The equation that need to be derived is the one correspondent to the first row in (2).

$$\begin{aligned}\dot{x}_e &= (\dot{x}_d - \dot{x})\cos(\theta) + (\dot{y}_d - \dot{y})\sin(\theta) \\ &\quad - (x_d - x)\dot{\theta}\sin(\theta) + (y_d - y)\dot{\theta}\cos(\theta) \\ &= \dot{x}_d\cos(\theta) - \dot{x}\cos(\theta) + \dot{y}_d\sin(\theta) - \dot{y}\sin(\theta) \\ &\quad - (x_d - x)\dot{\theta}\sin(\theta) + (y_d - y)\dot{\theta}\cos(\theta)\end{aligned}$$

Applying the change $\dot{\theta} = \omega$ the last equation can be expressed as

$$\dot{x}_e = \dot{x}_d\cos(\theta) - \dot{x}\cos(\theta) + \dot{y}_d\sin(\theta) - \dot{y}\sin(\theta) - (x_d - x)\omega\sin(\theta) + (y_d - y)\omega\cos(\theta)$$

From the equation of y_e in (2), we know that $y_e = -(x_d - x)\sin(\theta) + (y_d - y)\cos(\theta)$ which appears in the previous equality. Hence

$$\dot{x}_e = \dot{x}_d\cos(\theta) - \dot{x}\cos(\theta) + \dot{y}_d\sin(\theta) - \dot{y}\sin(\theta) + \omega y_e$$

The negative terms of the previous equality, $-\dot{x}\cos(\theta)$ and $-\dot{y}\sin(\theta)$, can be developed using the equations of the model of the vehicle (1)

$$\dot{x}\cos(\theta) + \dot{y}\sin(\theta) = v\cos(\theta)\cos(\theta) + v\sin(\theta)\sin(\theta) = v(\sin^2(\theta) + \cos^2(\theta)) = v$$

Now, using this result

$$\dot{x}_e = \dot{x}_d\cos(\theta) - v + \dot{y}_d\sin(\theta) + \omega y_e$$

By definition: $\theta_e = \theta_d - \theta$, then $\theta = \theta_d - \theta_e$. Replacing θ in $\dot{x}_e$

$$\dot{x}_e = \dot{x}_d\cos(\theta_d - \theta_e) - v + \dot{y}_d\sin(\theta_d - \theta_e) + \omega y_e$$

The trigonometric identities for $\cos(\alpha - \beta)$ y $\sin(\alpha - \beta)$ are used in the next step

$$\begin{aligned}\dot{x}_e &= \dot{x}_d(\cos(\theta_d)\cos(\theta_e) + \sin(\theta_d)\sin(\theta_e)) - v \\ &\quad + \dot{y}_d(\sin(\theta_d)\cos(\theta_e) - \cos(\theta_d)\sin(\theta_e)) + \omega y_e \\ &= \omega y_e - v + (\dot{x}_d\cos(\theta_d) + \dot{y}_d\sin(\theta_d))\cos(\theta_e) \\ &\quad + (\dot{x}_d\sin(\theta_d) - \dot{y}_d\cos(\theta_d))\cos(\theta_e)\end{aligned}$$

The non-holonomic constraint for the real wheels is: $\dot{x}_d\sin(\theta_d) = \dot{y}_d\cos(\theta_d)$. Therefore

$$\dot{x}_e = \omega y_e - v + (\dot{x}_d\cos(\theta_d) + \dot{y}_d\sin(\theta_d))\cos(\theta_e)$$

Following the same procedure that was used before, the terms inside the parenthesis become

$$\dot{x}_d\cos(\theta_d) + \dot{y}_d\sin(\theta_d) = v_d\cos(\theta_d)\cos(\theta_d) + v_d\sin(\theta_d)\sin(\theta_d) = v_d(\sin^2(\theta_d) + \cos^2(\theta_d)) = v_d$$

Finally, using the previous equality, the result for $\dot{x}_e$ is

$$\dot{x}_e = \omega y_e - v + v_d\cos(\theta_e) \tag{A.1}$$

- **Time derivative of y_e.**

The derivation of the $\dot{y}_e$ is similar to the one used for $\dot{x}_e$. The equation that need to be derived is the one correspondent to the second row in (2).

$$\dot{y}_e = -(\dot{x}_d - \dot{x})\sin(\theta) + (\dot{y}_d - \dot{y})\cos(\theta) - (x_d - x)\dot{\theta}\cos(\theta) - (y_d - y)\dot{\theta}\sin(\theta)$$

We had $x_e = (x_d - x)\cos(\theta) + (y_d - y)\sin(\theta)$ from (2) which appears in the previous equality and we also know $\dot{\theta} = \omega$. Hence the last expression can be represented as

$$\begin{aligned}\dot{y}_e &= -(\dot{x}_d - \dot{x})\sin(\theta) + (\dot{y}_d - \dot{y})\cos(\theta) - x_e\omega \\ &= -x_e\omega + \dot{x}\sin(\theta) - \dot{y}\cos(\theta) - \dot{x}_d\sin(\theta) + \dot{y}_d\cos(\theta)\end{aligned}$$

The non-holonomic constraint for the rear wheels is: $\dot{x}\sin(\theta) = \dot{y}\cos(\theta)$. Therefore

$$\dot{y}_e = -x_e\omega - \dot{x}_d\sin(\theta) + \dot{y}_d\cos(\theta_e ur)$$

The error in θ is $\theta_e = \theta_d - \theta$, then $\theta = \theta_d - \theta_e$. Replacing θ in the previous equation

$$\dot{y}_e = -x_e\omega - \dot{x}_d\sin(\theta_d - \theta_e) + \dot{y}_d\cos(\theta_d - \theta_e)$$

The trigonometric identities for $\cos(\alpha - \beta)$ y $\sin(\alpha - \beta)$ are used in the next step

$$\begin{aligned}\dot{y}_e &= -x_e\omega - \dot{x}_d(\sin(\theta_d)\cos(\theta_e) - \cos(\theta_d)\sin(\theta_e)) \\ &\quad + \dot{y}_d(\cos(\theta_d)\cos(\theta_e) + \sin(\theta_d)\sin(\theta_e)) \\ &= -x_e\omega + (\dot{x}_d\cos(\theta_d) + \dot{y}_d\sin(\theta_d))\sin(\theta_e) \\ &\quad + (\dot{y}_d\cos(\theta_d) - \dot{x}_d\sin(\theta_d))\cos(\theta_e)\end{aligned}$$

The same non-holonomic constraint is fulfilled for the reference car: $\dot{x}_d\sin(\theta_d) = \dot{y}_d\cos(\theta_d)$. Using this constraint in $\dot{y}_e$

$$\dot{y}_e = -x_e\omega + (\dot{x}_d\cos(\theta_d) + \dot{y}_d\sin(\theta_d))\sin(\theta_e)$$

Following the same procedure that was used before, the terms inside the parenthesis become

$$\dot{x}_d\cos(\theta_d) + \dot{y}_d\sin(\theta_d) = v_d\cos(\theta_d)\cos(\theta_d) + v_d\sin(\theta_d)\sin(\theta_d) = v_d(\sin^2(\theta_d) + \cos^2(\theta_d)) = v_d$$

Finally, using the previous equality, the result for $\dot{y}_e$ is

$$\dot{y}_e = -x_e\omega + v_d\sin(\theta_e) \tag{A.2}$$

- **Time derivative of θ_e.**

The last state that need to be derivated is θ_e. This one is straightforward, the equation to differentiate is: $\theta_e = \theta_d - \theta$.

The result is:

$$\dot{\theta}_e = \dot{\theta}_d - \dot{\theta} = \omega_d - \omega \tag{A.3}$$

Kinematic error model equations

The obtained result for the error model, Eqs. (A.1) to (A.3), are shown here in matrix form

$$\begin{bmatrix}\dot{x}_e \\ \dot{y}_e \\ \dot{\theta}_e\end{bmatrix} = \begin{bmatrix}\omega y_e - v + v_d\cos(\theta_e) \\ -\omega x_e + v_d\sin(\theta_e) \\ \omega_d - \omega\end{bmatrix} \tag{A.4}$$

References

Aicardi, M., Casalino, G., Bicchi, A., & Balestrino, A. (1995). Closed loop steering of unicycle like vehicles via Lyapunov techniques. *IEEE Robotics & Automation Magazine*, *2*(1), 27–35.

Alcalá, E., Sellart, L., Puig, V., Quevedo, J., Saludes, J., Vázquez, D., et al. (2016). Comparison of two non-linear model-based control strategies for autonomous vehicles. In *2016 24th Mediterranean Conference on Control and Automation (MED)* (pp. 846–851). IEEE.

Bianco, C. G. L., Piazzi, A., & Romano, M. (0000). Velocity planning for autonomous vehicles. *IEEE Intelligent Vehicles Symposium*.

Blažič, S. (2010). Takagi–sugeno vs. Lyapunov-based tracking control for a wheeled mobile robot. *WSEAS Transactions on Systems and Control*, *5*(8), 667–676.

Dixon, W., Dawson, D. M., Zergeroglu, E., & Behal, A. (2001). Nonlinear control of wheeled mobile robots.

Dixon, W. E., Dawson, D. M., Zergeroglu, E., & Behal, A. (2000). *Nonlinear Control of Wheeled Mobile Robots*. Springer.

Duan, G.-R., & Yu, H.-H. (2013). *LMIs in Control Systems: Analysis, Design and Applications*. CRC Press.

Edy's Vehicle Physics (0000). http://www.edy.es/dev/vehicle-physics/.

El Ghaoui, L., & Balakrishnan, V. (1994). Synthesis of fixed-structure controllers via numerical optimization. In *Proceedings of the 33rd IEEE Conference on Decision and Control, 1994*: *Vol. 3*. (pp. 2678–2683). IEEE.

Farag, A., & Werner, H. (2004). A riccati-genetic algorithms approach to fixed-structure controller synthesis. In *Proceedings of the American Control Conference, 2004*: *Vol. 3*. (pp. 2799–2804). IEEE.

Freeman, R., & Kokotovic, P. V. (2008). *Robust Nonlinear Control Design: State-Space and Lyapunov Techniques*. Springer Science & Business Media.

Hahn, S., Zindler, K., & Jumar, U. (2016). Two-degrees-of-freedom lateral vehicle control using nonlinear model based disturbance compensation. *IFAC-PapersOnLine*, *49*(11), 182–189.

Indiveri, G. (1999). Kinematic time-invariant control of a 2d nonholonomic vehicle. In *Proceedings of the 38th IEEE Conference on Decision and Control, 1999*: *Vol. 3*. (pp. 2112–2117). IEEE.

International Organization for Standardization (March 1997). ISO 2631-1, Standard, International standard.

Marino, R., Scalzi, S., Orlando, G., & Netto, M. (2009). A nested pid steering control for lane keeping in vision based autonomous vehicles. In *American Control Conference, 2009. ACC'09* (pp. 2885–2890). IEEE.

Németh, B., Gáspár, P., & Bokor, J. (2016). Lpv-based integrated vehicle control design considering the nonlinear characteristics of the tire. In *American Control Conference (ACC), 2016* (pp. 6893–6898). IEEE.

Nguyen, A.-T., Sentouh, C., & Popieul, J.-C. (2016). Takagi–sugeno model-based steering control for autonomous vehicles with actuator saturation. *IFAC-PapersOnLine*, *49*(5), 206–211.

Rajamani, R. (2011). *Vehicle Dynamics and Control*. Springer Science & Business Media.

Ros, G., Sellart, L., Materzynska, J., Vazquez, D., & Lopez, A. M. (2016). The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *The IEEE Conference on Computer Vision and Pattern Recognition* (CVPR).

Samson, C., & Ait-Abderrahim, K. (24 May 2006). Mobile robot control. part 1: Feedback control of nonholonomic wheeled cart in Cartesian space, HAL-INRIA (pp. 30–34).

Shamma, J. S. (2012). An overview of lpv systems. In *Control of Linear Parameter Varying Systems with Applications* (pp. 3–26). Springer.

Solea, R., & Nunes, U. (2007). Trajectory planning and sliding-mode control based trajectory-tracking for cybercars. *Integrated Computer-Aided Engineering*, *14*, 33–47.

Soualmi, B., Sentouh, C., Popieul, J., & Debernard, S. (2014). Automation-driver cooperative driving in presence of undetected obstacles. *Control Engineering Practice*, *24*, 106–119.

Tazzari electric vehicle (0000). http://www.tazzari-zero.com/es/.

Zhang, H., & Wang, J. (2016). Vehicle lateral dynamics control through afs/dyc and robust gain-scheduling approach. *IEEE Transactions on Vehicular Technology*, *65*(1), 489–494.