Automatic Perpendicular and Diagonal Unparking Using a Multi-Sensor-Based Control Approach

David Pérez-Morales, Olivier Kermorgant, Salvador Dominguez-Quijada, Philippe Martinet

Automatic Perpendicular and Diagonal Unparking Using a Multi-Sensor-Based Control Approach

David Pérez-Morales, Olivier Kermorgant, Salvador Domínguez-Quijada and Philippe Martinet

***Abstract*— This paper explores the feasibility of a Multi-Sensor-Based Control (MSBC) approach for addressing forward nonparallel (perpendicular and diagonal) unparking problems of car-like vehicles as an alternative to classical approaches (e.g. path planning based, etc.). The results of individual cases are presented to illustrate the behavior and performance of the proposed approach as well as results from exhaustive simulations to evaluate the convergence and stability. The results presented in this work increase the versatility and validity of our MSBC approach towards a fully autonomous parking system.**

I. INTRODUCTION

Even for experienced drivers, parking and unparking can be a difficult tasks, especially in big cities where the parking spots are often narrow. The search for an increase in comfort and safety when parking and unparking has led to a quite extensive literature [1], having explored many different approaches to automate this bothersome task.

Despite the fact that the automobile industry has already started to roll out some commercial active parking assistants capable of actively controlling acceleration, braking and steering [2], the quantity of systems able to unpark is limited.

Path planning approaches have been heavily investigated in recent years. Even if the focus of most works in the literature is on the parking task, if a path is found it can be used as well for unparking. Among the different planning techniques it is possible to distinguish between geometric approaches, with either constant turning radius [3], [4] using saturated feedback controllers, or continuous-curvature planning using clothoids [5], [6]; heuristic approaches [7] and machine learning techniques [8].

A well-known drawback of path planning is that it is necessary to have the knowledge on the free and occupied space of the whole environment beforehand if online replanning is not feasible, potentially leading to costly infrastructure requirements. Furthermore, the tracking performance of a given path is highly dependent on the localization performance which might get degraded on certain environments (e.g. underground parking lots without any special infrastructure) or after a few maneuvers leading to non-negligible differences between the planned path and the performed one [5], [6].

D. Pérez-Morales, O. Kermorgant and S. Domínguez-Quijada are with École Centrale Nantes and Laboratoire des Sciences du Numérique de Nantes(LS2N UMR 6004), France. P. Martinet is with Inria Sophia Antipolis, École Centrale de Nantes and LS2N, France
David.Perez-Morales@eleves.ec-nantes.fr
Olivier.Kermorgant@ec-nantes.fr
Salvador.DominguezQuijada@ls2n.fr
Philippe.Martinet@inria.fr

An interesting alternative is the use of a sensor-based control approach. It has been proven to be valid for navigation [9], dynamic obstacle avoidance [10] and for parking applications [11], [12] but has yet to be investigated for unparking tasks - where an acceptable final pose for the vehicle might not be as clearly defined when compared to parking tasks. Thus, it is worth to study the behavior and performance of unparking maneuvers using a sensor-based control approach.

Assuming that the vehicle is capable of perceiving its own parking spot, it should possible to unpark without any path planning nor prediction using a Multi-Sensor-Based Control (MSBC) approach by minimizing the error between the current value of a certain set of sensor features (i.e. a line collinear to the parking spot main axis and another collinear to the rear boundary of the parking spot) and its desired value while avoiding collision by imposing certain constraints on another set of sensor features (lines defining the boundaries of the parking spot, points at the corners of said spot, etc.).

The contribution of this paper is the exploration of a MSBC approach for forward perpendicular and diagonal unparking in one maneuver. It should be noted that, in order to decouple the performance of the controller from the perception, the sensory data is generated virtually and assumed to be available all the time.

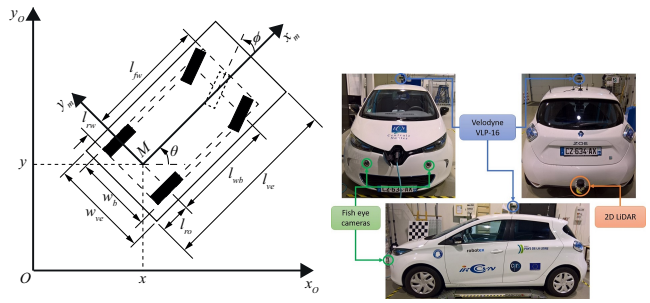
In the next section the kinematic model of the vehicle and the multi-sensor modeling are presented. Section III describes the interaction model allowing to formalize the unparking task and the constraints for collision avoidance. Afterwards, the controller is presented in Section IV. The obtained results are presented in Section V: several cases in two different simulation environments are presented as well as exhaustive simulations results for assessing the convergence performance of the presented approach for the two different types of parking maneuvers addressed are shown. Finally, some conclusions are given in Section VI.

II. MODELING AND NOTATION

A. Car-like robot model and notation

Given that unparking maneuvers are low-speed motions, a kinematic model can be considered as accurate enough. Thus, in the present work the kinematic model for a car with rear-wheel driving is considered:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ \tan\phi/l_{wb} \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \dot{\phi}, \qquad (1)$$



(a) (b)

Fig. 1. (a) Kinematic model diagram for a car-like rear-wheel driving robot. (b) Robotized Renault ZOE used for real experimentation

where v and $\dot{\phi}$ are the longitudinal and steering velocities.

The point M is located at the mid-distance between the passive fixed wheels (rear) axle and the distance between the rear and the front axle is described by l_{wb}. The generalized coordinates are $\mathbf{q} = [x, y, \theta, \phi]^\mathsf{T}$ where x and y are the Cartesian coordinates of the point M, θ is the orientation of the platform with respect to the x_0 axis and the steering angle of the steerable wheel(s) is denoted by ϕ (Fig. 1a).

The turning radius ρ_m around the instantaneous center of rotation (ICR) can be defined as:

$$\rho_m = \frac{l_{wb}}{\tan\phi} \tag{2}$$

The vehicle used for experimentation and simulation, represented by its bounding rectangle in Fig. 1a, is a Renault ZOE (Fig. 1b). Its relevant dimensional parameters are presented in Table I.

TABLE I

DIMENSIONAL VEHICLE PARAMETERS

Parameters	*Notation*	*Value*
Wheelbase: Distance between the front and rear wheel axles	l_{wb}	2.588 m
Rear overhang: Distance between the rear wheel axle and the rear bumper	l_{ro}	0.657 m
Total length of the vehicle	l_{ve}	4.084 m
Total width of the vehicle	w_{ve}	1.945 m

B. Multi-sensor modeling

The considered multi-sensor modeling is recalled in this subsection.

1) Kinematic model: Let us consider a robotic system equipped with k sensors (Fig. 2) that provide data about the environment. Each sensor S_i gives a signal (sensor feature) $\mathbf{s}_i$ of dimension $\mathbb{d}_i$ with $\sum_{i=1}^{k} \mathbb{d}_i = \mathbb{d}$.

In a static environment, the sensor feature derivative can be expressed as follows:

$$\dot{\mathbf{s}}_i = \breve{\mathbf{L}}_i \breve{\mathbf{v}}_i = \breve{\mathbf{L}}_i {}^i\breve{\mathbf{T}}_m \breve{\mathbf{v}}_m \tag{3}$$

where $\breve{\mathbf{L}}_i$ is the interaction matrix [13] of $\mathbf{s}_i$ ($\dim(\breve{\mathbf{L}}_i) = \mathbb{d}_i \times 6$) and ${}^i\breve{\mathbf{T}}_m$ is the 3D screw transformation matrix that allows expressing the sensor twist $\breve{\mathbf{v}}_i$ (which is expressed in its corresponding frame $\mathcal{F}_i$) with respect to the robot twist $\breve{\mathbf{v}}_m$ (expressed in the control frame $\mathcal{F}_m$).

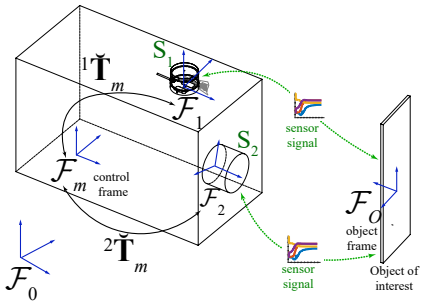


Fig. 2. Multi-sensor model

Denoting $\mathbf{s} = (\mathbf{s}_1, \ldots, \mathbf{s}_k)$ the $\mathbb{d}$-dimensional signal of the multi-sensor system, the signal variation over time can be linked to the moving vehicle twist:

$$\dot{\mathbf{s}} = \breve{\mathbf{L}}_\mathbf{s} \breve{\mathbf{v}}_m \tag{4}$$

with:

$$\breve{\mathbf{L}}_\mathbf{s} = \breve{\mathbf{L}} \breve{\mathbf{T}}_m \tag{5}$$

where $\breve{\mathbf{L}}$ and $\breve{\mathbf{T}}_m$ are obtained by concatenating either diagonally or vertically, respectively, matrices $\breve{\mathbf{L}}_i$ and ${}^i\breve{\mathbf{T}}_m$ $\forall\, i \in [1 \ \ldots \ k]$.

Planar world assumption: Assuming that the vehicle to which the sensors are rigidly attached evolves in a plane and that the sensors and vehicle have vertical parallel z axes, all the twists are reduced to $[v_{x_i}, v_{y_i}, \dot{\theta}_i]^\mathsf{T}$ hence the reduced forms $\breve{\mathbf{L}}$, $\breve{\mathbf{L}}_\mathbf{s}$, $\breve{\mathbf{L}}_i$, $\breve{\mathbf{v}}_m$ and ${}^i\breve{\mathbf{T}}_m$ of, respectively, $\breve{\mathbf{L}}$, $\breve{\mathbf{L}}_\mathbf{s}$, $\breve{\mathbf{L}}_i$, $\breve{\mathbf{v}}_m$ and ${}^i\breve{\mathbf{T}}_m$ are considered.

$\breve{\mathbf{L}}_i$ is of dimension $\mathbb{d}_i \times 3$, $\breve{\mathbf{v}}_m = [v_{x_m}, v_{y_m}, \dot{\theta}_m]^\mathsf{T}$ and ${}^i\breve{\mathbf{T}}_m$ is defined as:

$${}^i\breve{\mathbf{T}}_m = \begin{bmatrix} \cos({}^m\theta_i) & \sin({}^m\theta_i) & x_i \sin({}^m\theta_i) - y_i \cos({}^m\theta_i) \\ -\sin({}^m\theta_i) & \cos({}^m\theta_i) & x_i \cos({}^m\theta_i) + y_i \sin({}^m\theta_i) \\ 0 & 0 & 1 \end{bmatrix} \tag{6}$$

where ${}^m\mathbf{t}_i = [x_i, y_i]^\mathsf{T}$ and ${}^m\theta_i$ are, respectively, the position and orientation of S_i (frame $\mathcal{F}_i$) with respect to $\mathcal{F}_m$ expressed in $\mathcal{F}_m$.

Furthermore, since in the considered model the control frame $\mathcal{F}_m$ is attached to the vehicle rear axis with origin at the point M (Fig. 1a), it is not possible to generate a velocity along y_m on the vehicle frame and assuming that there is no slipping nor skidding (i.e. $v_{y_m} = 0$), the robot twist $\breve{\mathbf{v}}_m$ can be further reduced to:

$$\mathbf{v}_m = [v_{x_m}, \dot{\theta}_m]^\mathsf{T} \tag{7}$$

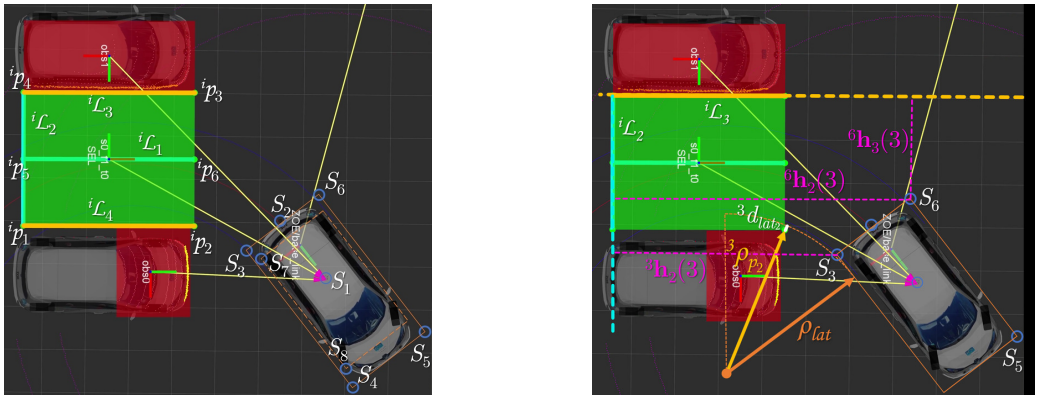
with $v_{x_m} = v$ and $\dot{\theta}_m = \dot{\theta}$ according to the model (1), thus it is possible to write:

$$\dot{\mathbf{s}} = \mathbf{L}_\mathbf{s} \mathbf{v}_m \tag{8}$$

where $\mathbf{L}_\mathbf{s}$ is composed of the first and third columns of $\breve{\mathbf{L}}_\mathbf{s}$.

III. INTERACTION MODEL

For the interaction model, we rely on the perception of several lines $\mathcal{L}_j$ and points from several (virtual) sensors



(a) Sensors' configuration and sensor features

(b) Constrainted features illustration

Fig. 3. (a) General sensors' configuration and sensor features. (b) Example of the constrained sensor features

placed at in convenient frames in order to simplify the sensor features definitions and their interaction matrices.

The sensors placement can be seen in Fig. 3a. S_1 corresponds to the VLP-16 while S_2 to the 2D LiDAR placed on the rear (LMS151). S_3 to S_6 are placed on the corners of the car bounding rectangle and have the same orientation as the control frame.

As it can be seen in Fig. 3a, points p_1 to p_4 correspond to the corners of the parking spot while p_5 and p_6 are, respectively, the midpoints between (p_1, p_4) and (p_2, p_3). $\mathcal{L}_1$ is a line that passes through p_5 and p_6, i.e. it passes through the center of the parking spot. $\mathcal{L}_2$ is a line that passes through p_1 and p_4 thus corresponding to the rear boundary of the parking spot. $\mathcal{L}_3$ is a line that passes through p_3 and p_4. All the lines are parametrized using normalized Plücker coordinates.

A. Line parametrization

Given two distinct 3D points ^ip_f and ^ip_g in homogeneous coordinates, with

$$^ip_f = [^iX_f, ^iY_f, ^iZ_f, ^iW_f]^\mathsf{T} \tag{9a}$$

$$^ip_g = [^iX_g, ^iY_g, ^iZ_g, ^iW_g]^\mathsf{T}, \tag{9b}$$

a line passing through them can be represented using normalized Plücker coordinates as a couple of 3-vectors [14]:

$$^i\mathcal{L}_j = [^i\underline{\mathbf{u}}_j, ^i\mathbf{h}_j]^\mathsf{T} \tag{10}$$

where $^i\underline{\mathbf{u}}_j = {}^i\mathbf{u}_j/||^i\mathbf{u}_j||$ (with $^i\mathbf{u}_j \neq 0$) describes the orientation of the line and $^i\mathbf{h}_j = {}^i\mathbf{r}_j/||^i\mathbf{u}_j||$ where $^i\mathbf{r}_j$ encodes the plane containing the line and the origin (*interpretation plane*) and the distance from the origin to the line. The two 3-vectors $^i\mathbf{u}_j$ and $^i\mathbf{r}_j$ are defined as [15]:

$$^i\mathbf{u}_j^\mathsf{T} = {}^iW_f[^iX_g, ^iY_g, ^iZ_g] - {}^iW_g[^iX_f, ^iY_f, ^iZ_f] \tag{11a}$$

$$^i\mathbf{r}_j^\mathsf{T} = [^iX_f, ^iY_f, ^iZ_f] \times [^iX_g, ^iY_g, ^iZ_g] \tag{11b}$$

Due to the planar world assumption considered in this paper, the third element of $^i\underline{\mathbf{u}}_j$ and the first and second elements of $^i\mathbf{h}_j$ are equal to zero, i.e. $^i\underline{\mathbf{u}}_j(3) = {}^i\mathbf{h}_j(1) = {}^i\mathbf{h}_j(2) = 0$, therefore the sensor signal $\mathbf{s}_{i_{\mathcal{L}_j}}$ and interaction matrix $\check{\mathbf{L}}_{i_{\mathcal{L}_j}}$ for the line $^i\mathcal{L}_j$ observed by S_i are defined respectively as:

$$\mathbf{s}_{i_{\mathcal{L}_j}} = \left[{}^i\underline{\mathbf{u}}_j(1), {}^i\underline{\mathbf{u}}_j(2), {}^i\mathbf{h}_j(3)\right]^\mathsf{T} \tag{12}$$

$$\check{\mathbf{L}}_{i_{\mathcal{L}_j}} = \begin{bmatrix} 0 & 0 & {}^i\underline{\mathbf{u}}_j(2) \\ 0 & 0 & -{}^i\underline{\mathbf{u}}_j(1) \\ -{}^i\underline{\mathbf{u}}_j(2) & {}^i\underline{\mathbf{u}}_j(1) & 0 \end{bmatrix} \tag{13}$$

B. Task sensor features

The set of task sensor features $\mathbf{s}^t$ is defined as:

$$\mathbf{s}^t = [s_1^t, \ldots, s_6^t]^\mathsf{T} = [\mathbf{s}_{1_{\mathcal{L}_1}}, \mathbf{s}_{1_{\mathcal{L}_2}}]^\mathsf{T} \tag{14}$$

The corresponding interaction matrix $\check{\mathbf{L}}^t$ is computed by a 2nd order approximation [16] of the form:

$$\check{\mathbf{L}}^t = \frac{\check{\mathbf{L}}_\mathcal{L} + \check{\mathbf{L}}_\mathcal{L}^*}{2} \tag{15}$$

where $\check{\mathbf{L}}_\mathcal{L} = [\check{\mathbf{L}}_{i_{\mathcal{L}_1}}, \check{\mathbf{L}}_{i_{\mathcal{L}_2}}]^\mathsf{T}$ and $\check{\mathbf{L}}_\mathcal{L}^*$ is equal to the value of $\check{\mathbf{L}}_\mathcal{L}$ at the desired pose.

Sensor-based control laws are usually defined so that the system reaches desired sensor features. In our case, the features are built on $^i\mathcal{L}_1$ and $^i\mathcal{L}_2$. While the desired values for those two lines are clearly defined when parking, multiple choices may be available when unparking. Indeed, it depends on how we want the car to be positioned at the end of the unparking maneuver. A sensible choice would be for $^i\mathcal{L}_1^*$ to be perpendicular to the vehicle longitudinal axis (x_m-axis) and $^i\mathcal{L}_2^*$ to be parallel to y_m-axis, with each line being at a certain distance to the vehicle in order to be able to drive it outside of the parking spot.

C. Constraints sensor features

The set of constrained sensor features (Fig. 3b) used for collision avoidance $\mathbf{s}^c$ is defined as:

$$\mathbf{s}^c = [s_1^c, \ldots, s_6^c]^\mathsf{T} = [\mathbf{s}_3, \mathbf{s}_6]^\mathsf{T} \tag{16}$$

with

$$\mathbf{s}_3 = [^3\mathbf{h}_2(3), {}^3Y_2, {}^3d_{lat_2}]^\mathsf{T} \tag{17a}$$

$$\mathbf{s}_6 = [{}^6\mathbf{h}_2(3), {}^6\mathbf{h}_3(3), {}^6Y_3]^\mathsf{T}. \tag{17b}$$

where the difference of radii ${}^i d_{lat_a}$ is defined as:

$${}^i d_{lat_a} = {}^i\rho_{p_a} - \rho_{lat}, \tag{18}$$

with:

$$\begin{cases} {}^i\rho_{p_a} & = \sqrt{({}^iX_a + x_i)^2 + ({}^iY_a + y_i - \rho_m)^2} \\ \rho_{lat} & = |\rho_m| - \frac{w_{ve}}{2} \end{cases} \tag{19}$$

The interaction matrix $\check{\mathbf{L}}_{i_{Y_a}}$ associated to ${}^iY_a$ is:

$$\check{\mathbf{L}}_{i_{Y_a}} = \begin{bmatrix} 0 & -1 & -{}^iX_a \end{bmatrix} \tag{20}$$

while interaction matrix associated to ${}^i d_{lat_a}$ is defined as:

$$\check{\mathbf{L}}_{i_d} = \begin{bmatrix} 0 & \frac{{}^i\varrho_y}{{}^i\rho^2_{p_a}} & \frac{{}^iX_a\,{}^i\varrho_y}{{}^i\rho^2_{p_a}} \end{bmatrix} \tag{21}$$

with ${}^i\varrho_y = -|{}^iY_a + y_i - \rho_m|$. The interaction matrices associated to the rest of the features used as constraints can be deduced from the third row of (13). The corresponding interaction matrix $\check{\mathbf{L}}^c_\mathbf{s}$ is computed at each iteration.

It should be noted that some constraints must be deactivated under certain conditions as the scene topology evolves. These conditions used to obtain the results presented in this work are detailed in the following table:

TABLE II
CONSTRAINTS DEACTIVATION CONDITIONS

Constraint	*Deactivate if*
${}^3d_{lat_2}$	$\phi \geq 0$ or ${}^3X_2 > -x_i$
${}^6\mathbf{h}_3(3)$	${}^6X_3 < 0$
${}^6Y_3$	${}^6X_3 < 0$

IV. CONTROL

The controller presented in [12] used for parking tasks is now used as well for unparking. For the sake of clarity we recall its definition and main properties. The control input of the robotized vehicle is defined as:

$$\mathbf{v}_r = [v, \phi]^\mathsf{T} \tag{22}$$

with ϕ, considering (1) and (2), being mapped to $\dot{\theta}$ by

$$\dot{\theta} = \frac{v}{\rho_m}. \tag{23}$$

The steering angle ϕ is bounded by its maximum value $\phi_{\max}$ with

$$|\phi| < \phi_{\max} \tag{24}$$

while the longitudinal velocity of the vehicle is saturated by

$$|v| < v_{\max} \tag{25}$$

where $v_{\max}$ is an adaptive saturation value imposing a deceleration profile based on the velocity profile shown in [4] as the vehicle approaches the final pose. Furthermore, to avoid large changes in the control signals at the current iteration n that may cause uncomfortable sensations for the passengers or surrounding witnesses and, to consider to some extent the dynamic limitations of the vehicle, the control signals are saturated as well by some increments with respect to the previous control signals (at iteration $n-1$) as shown below:

$$(v_{n-1} - \Delta_{dec}) \leq v_n \leq (v_{n-1} + \Delta_{acc}) \tag{26a}$$

$$(\phi_{n-1} - \Delta_\phi) \leq \phi_n \leq (\phi_{n-1} + \Delta_\phi). \tag{26b}$$

In order to automatically adapt the influence of each task feature, we use a weighting version of the feature error (14):

$$\mathbf{e}^t_\mathbf{W} = \mathbf{W}\mathbf{e}^t = \mathbf{W}(\mathbf{s}^t - \mathbf{s}^{t*}) \tag{27}$$

where $\mathbf{W}$ is a diagonal weighting matrix of dimension 6, with constant value for w_3 and w_6 but where $(w_i, i \in \{1,2,4,5\})$ are computed using a smooth weighting function (Fig. 4) based on the one presented in [17].

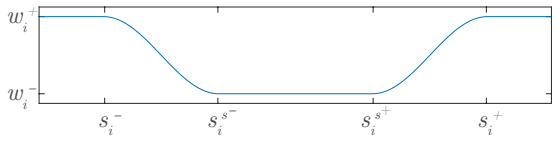


Fig. 4. Weighting function $w(s)$

The task interaction matrix corresponding to (27) is similarly weighted:

$$\mathbf{L}^t_\mathbf{W} = \mathbf{W}\mathbf{L}^t_\mathbf{s} \tag{28}$$

The core of the control law is formulated in a quadratic programming form [18] with only inequality constraints. It expresses that we want $\mathbf{s}^t$ to approach its desired value at each iteration, but only in the vehicle velocity subspace compatible with the various constraints (obstacles, velocity and steering limits):

$$\begin{aligned} \mathbf{v}_m = \text{argmin} &||\mathbf{L}^t_\mathbf{W}\mathbf{v}_m + \lambda\mathbf{e}^t_\mathbf{W}||^2 \\ \text{s.t. } &\mathbf{A}\mathbf{v}_m \leq \mathbf{b} \end{aligned} \tag{29}$$

with:

$$\mathbf{A} = [\mathbf{L}^c_\mathbf{s}, -\mathbf{L}^c_\mathbf{s}]^\mathsf{T} \tag{30}$$

$$\mathbf{b} = [\alpha(\mathbf{s}^{c^+} - \mathbf{s}^c), -\alpha(\mathbf{s}^{c^-} - \mathbf{s}^c)]^\mathsf{T} \tag{31}$$

where α is a gain constant, λ is the control gain and $[\mathbf{s}^{c^-}, \mathbf{s}^{c^+}]$ is the interval in which $\mathbf{s}^c$ should remain. Since the constraints are used for collision avoidance, only one side of the interval $[s_i^{c^-}, s_i^{c^+}]$ is considered for each feature.

We now present simulation results illustrating the validity of this framework.

V. RESULTS

For the results shown in this section, the parameters in Table III are considered. The value of $\phi_{\max}$ corresponds to the maximum steering angle of the real vehicle while the rest of the parameters were determined by empirical testing. For the MATLAB implementations, the sampling time T_s is equal to 0.1 s while for ROS, T_s = 0.01 s.

The nonlinear solver used for MATLAB implementations is fmincon with a Sequential Quadratic Programming (SQP) algorithm while for C++ implementations the solver NLopt [19] with a Sequential Least Squares Programming (SLSQP) algorithm is used.

TABLE III

CONTROL-RELATED VEHICLE PARAMETERS

Parameteres	*Notation*	*Value*
Maximum steering angle	$\phi_{\max}$	30°
Maximum longitudinal velocity	$v_{\max}$	$\leq$ 2 km/h
Maximum acceleration increment	Δ_{acc}	$\text{sign}(v_{n-1})\ 0.2\ T_s$
Maximum deceleration increment	Δ_{dec}	$\text{sign}(v_{n-1})\ 2.5\ T_s$
Maximum ϕ increment	Δ_{ϕ}	$2^\circ\ T_s$

A. MATLAB simulations

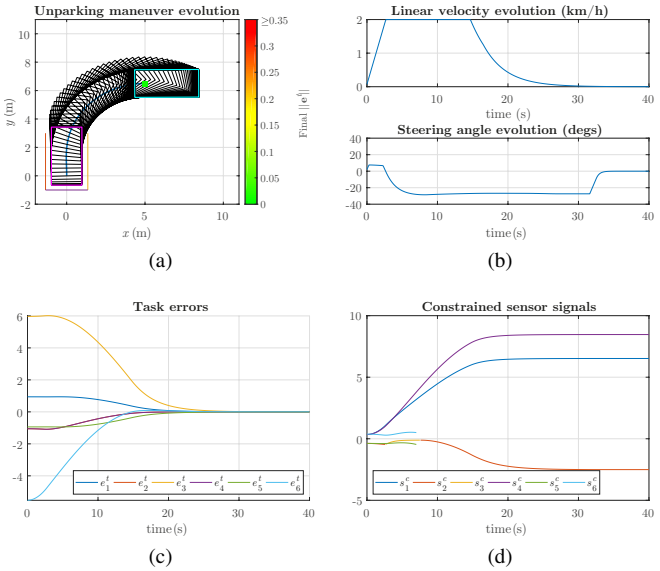


Fig. 5. Constrained forward $\perp$ unparking maneuver signals: (a) performed maneuver (desired pose = (5m, 6.5m, 0°)), (b) control signals, (c) task error signal, (d) constrained sensor signals

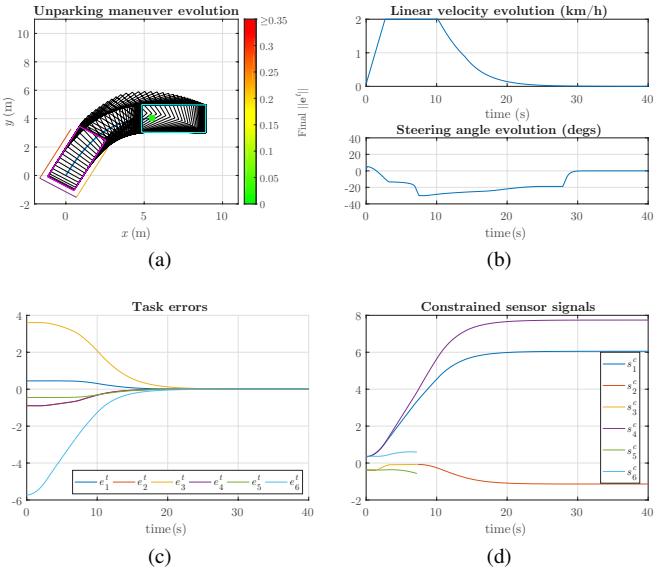
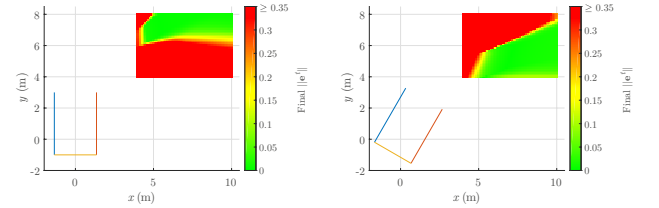


Fig. 6. Constrained forward diagonal unparking maneuver signals: (a) performed maneuver (desired pose = (5.5m, 4m, 0°)), (b) control signals, (c) task error signal, (d) constrained sensor signals



(a) Backward perpendicular case. (b) Backward diagonal case.

Fig. 7. Exhaustive simulations. Final orientation = 0°. Parking spot length = 4m and width = 2.7m

To illustrate the behavior of the MSBC approach, perpendicular (Fig. 5) and diagonal (Fig. 6) maneuvers are shown. The final position is marked with a colored square whose color depends on the final value of $||\mathbf{e}^t||$. It can be clearly seen that, for both cases, the car is able to unpark successfully. The control signals have a generally smooth evolution (Figs. 5b, 6b) thanks to the considered bounds and saturation values. The (active) constraints imposed on $\mathbf{s}^c$ are satisfied at each time instant (Fig. 5d) ensuring a collision-free maneuver.

B. Exhaustive simulations

It is well known that the stability of sensor-based control laws such as (29) are very difficult to analyze. That is why various convergence analyses for the two unparking cases were conducted by means of exhaustive simulations. Due to paper length constraints, we assume that the desired final orientation of the vehicle is 0° for the two considered cases (Figs. 7a-7b).

Since the exhaustive simulations are an aggregation of the results obtained from several simulations (like those shown in Figs. 5 and 6), each figure consists of a parking spot (represented by 3 lines) adapted to each case and a scatter plot of the final position of the vehicle (with a sampling step of 10cm), whose color depends on the final value of $||\mathbf{e}^t||$. The green portion of each scatter plot corresponds to the region of attraction (ROA) and the red one represents the initial positions that are outside of the ROA. It should be noted that in the red regions the car ends outside of the parking spot but, due to the constrained sensor features and non-holonomic constraints, the car is not capable of reaching the desired pose.

Furthermore, it was expected for the ROA to be smaller for perpendicular maneuvers compared to the diagonals ones since, as most experienced drivers know, more space is needed to maneuver when unparking from a perpendicular spot.

C. Fast prototyping environment

A homemade fast prototyping environment using the same software architecture as the one embedded inside the car is used for simulation purposes. In addition to behaving nearly identically (from a software architecture point of view) to the real vehicle, this fast prototyping environment simulates as well the dynamics of the vehicle, leading to more realistic simulations than the MATLAB environment used for the results presented in the previous subsections.

As it can be seen in Figs. 8-9, the car is able to unpark successfully from the parking spot (represented by a green rectangle) in one motion while satisfying the constraints during the whole maneuver, with the evolution of the many different signals being very similar to the MATLAB cases.

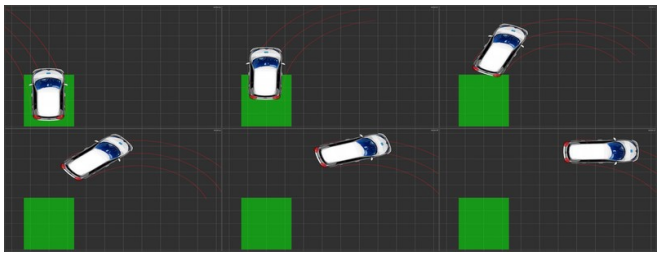
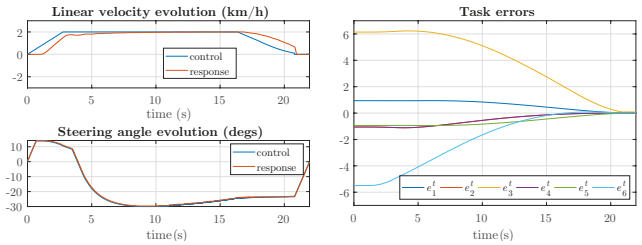
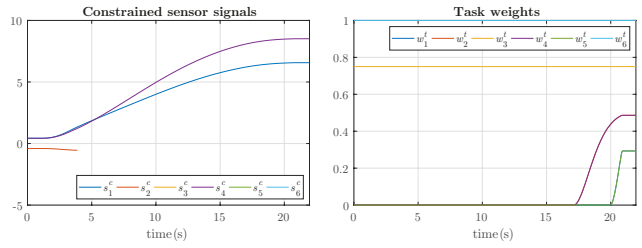


Fig. 8. Forward perpendicular unparking maneuver in simulation using a homemade fast prototyping environment



(a) Control signals (b) Task error signal: $\mathbf{e}_2^t$



(c) Constrained sensor signals (d) Variable weighting signals

Fig. 9. Backward perpendicular parking maneuver signals

VI. CONCLUSIONS

Following our previous work [12], we've shown how, with some minor changes (simplifications) on the interaction model, it is possible to perform unparking tasks using the same control approach as for parking tasks. These new results increase the, already important, versatility and validity of our MSBC approach towards a fully autonomous parking system. The main considered (future) improvement is on the automated definition of the desired features, that are not uniquely defined when unparking. A region-reaching control law may be suitable, as in a fully autonomous framework the car could reach the most feasible position on the trajectory to be tracked after unparking.

ACKNOWLEDGMENT

This work was supported by the Mexican National Council for Science and Technology (CONACYT). This paper describes work carried out in the framework of the Valet project, reference ANR-15-CE22-0013-02.

REFERENCES

[1] W. Wang, Y. Song, J. Zhang, and H. Deng, "Automatic parking of vehicles: A review of literatures," *International Journal of Automotive Technology*, 2014.

[2] Y. Song and C. Liao, "Analysis and Review of State-of-the-Art Automatic Parking Assist System," in *2016 IEEE Int. Conf. on Vehicular Electronics and Safety*, Beijing, China, 2016.

[3] P. Petrov, F. Nashashibi, and M. Marouf, "Path Planning and Steering control for an Automatic Perpendicular Parking Assist System," in *7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles, PPNIV'15*, Hamburg, Germany, 2015.

[4] P. Petrov and F. Nashashibi, "Saturated Feedback Control for an Automated Parallel Parking Assist System," in *13th Int. Conf. on Control, Automation, Robotics and Vision (ICARCV'14)*, Marina Bay Sands, Singapore, 2014.

[5] H. Vorobieva, N. Minoiu-Enache, S. Glaser, and S. Mammar, "Geometric Continuous-Curvature Path Planning for Automatic Parallel Parking," in *2013 10th IEEE Int. Conf. on Networking, Sensing and Control (ICNSC)*, Evry, France, 2013.

[6] Y. Yi, Z. Lu, Q. Xin, L. Jinzhou, L. Yijin, and W. Jianhang, "Smooth path planning for autonomous parking system," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017.

[7] C. Chen, M. Rickert, and A. Knoll, "Path planning with orientation-aware space exploration guided heuristic search for autonomous parking and maneuvering," in *2015 IEEE Intelligent Vehicles Symposium*, Seoul, Korea, 2015.

[8] G. Notomista and M. Botsch, "Maneuver segmentation for autonomous parking based on ensemble learning," in *2015 Int. Joint Conf. on Neural Networks (IJCNN)*, Killarney, Ireland, 2015.

[9] D. A. de Lima and A. C. Victorino, "Sensor-Based Control with Digital Maps Association for Global Navigation: A Real Application for Autonomous Vehicles," in *2015 IEEE 18th Int. Conf. on Intelligent Transportation Systems*, Las Palmas, Spain, 2015.

[10] Y. Kang, D. A. de Lima, and A. C. Victorino, "Dynamic obstacles avoidance based on image-based dynamic window approach for human-vehicle interaction," in *2015 IEEE Intelligent Vehicles Symposium (IV)*, Seoul, South Korea, jun 2015.

[11] D. Pérez Morales, S. Domínguez Quijada, O. Kermorgant, and P. Martinet, "Autonomous parking using a sensor based approach," in *8th Workshop on Planning, Perception and Navigation for Intelligent Vehicles, PPNIV'16 at 19th IEEE ITSC 2016*, Rio de Janeiro, Brazil, 2016.

[12] D. Pérez-Morales, O. Kermorgant, S. Domínguez-Quijada, and P. Martinet, "Laser-Based Control Law For Autonomous Parallel And Perpendicular Parking," in *Second IEEE Int. Conf. on Robotic Computing*, Laguna Hills, CA, 2018.

[13] F. Chaumette and S. Hutchinson, "Visual servo control, part I : Basic Approaches," *IEEE Robotics Automation Magazine*, 2006.

[14] N. Andreff, B. Espiau, and R. Horaud, "Visual Servoing from Lines," *International Journal of Robotics Research*, 2002.

[15] B. Přibyl, P. Zemčík, and M. Čadik, "Camera Pose Estimation from Lines using Plücker Coordinates," in *Procedings of the British Machine Vision Conf. 2015*. British Machine Vision Association, 2015.

[16] O. Thari and Y. Mezouar, "On the efficient second order minimization and image-based visual servoing," in *2008 IEEE Int. Conf. on Robotics and Automation*. IEEE, may 2008.

[17] V. K. Narayanan, F. Pasteau, M. Marchal, A. Krupa, and M. Babel, "Vision-based adaptive assistance and haptic guidance for safe wheelchair corridor following," *Computer Vision and Image Understanding*, 2016.

[18] O. Kanoun, F. Lamiraux, and P.-B. Wieber, "Kinematic control of redundant manipulators: generalizing the task priority framework to inequality tasks," *IEEE Trans. on Robotics*, 2011.

[19] S. G. Johnson, "The NLopt nonlinear-optimization package." [Online]. Available: http://ab-initio.mit.edu/nlopt