

Autonomous car driving by a humanoid robot

Antonio Paolillo, Pierre Gergondet, Andrea Cherubini, Marilena Vendittelli, Abderrahmane Kheddar

▶ **To cite this version:**

Antonio Paolillo, Pierre Gergondet, Andrea Cherubini, Marilena Vendittelli, Abderrahmane Kheddar. Autonomous car driving by a humanoid robot. Journal of Field Robotics, 2018, 35 (2), pp.169-186. 10.1002/rob.21731 . hal-01485736

HAL Id: hal-01485736
https://hal.science/hal-01485736

Submitted on 9 Mar 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Autonomous car driving by a humanoid robot

Antonio Paolillo
CNRS-UM LIRMM
161 Rue Ada, 34090 Montpellier, France
CNRS-AIST JRL UMI3218/RL
1-1-1 Umezono, 305-8560 Tsukuba, Japan.
paolillo@lirmm.fr

Pierre Gergondet
CNRS-AIST JRL UMI3218/RL
1-1-1 Umezono, 305-8560 Tsukuba, Japan.
pierre.gergondet@aist.go.jp

Andrea Cherubini
CNRS-UM LIRMM
161 Rue Ada, 34090 Montpellier, France
cherubini@lirmm.fr

Marilena Vendittelli
DIAG, Sapienza Università di Roma
via Ariosto 25, 00185 Roma, Italy
vendittelli@diag.uniroma.it

Abderrahmane Kheddar
CNRS-AIST JRL UMI3218/RL
1-1-1 Umezono, 305-8560 Tsukuba, Japan.
CNRS-UM LIRMM
161 Rue Ada, 34090 Montpellier, France.
kheddar@lirmm.fr

Abstract

Enabling a humanoid robot to drive a car, requires the development of a set of basic primitive actions. These include: walking to the vehicle, manually controlling its commands (e.g., ignition, gas pedal and steering), and moving with the whole-body, to ingress/egress the car. In this paper, we present a sensor-based reactive framework for realizing the central part of the complete task, consisting in driving the car along unknown roads. The proposed framework provides three driving strategies by which a human supervisor can teleoperate the car, ask for assistive driving, or give the robot full control of the car. A visual servoing scheme uses features of the road image to provide the reference angle for the steering wheel to drive the car at the center of the road. Simultaneously, a Kalman filter merges optical flow and accelerometer measurements, to estimate the car linear velocity and correspondingly compute the gas pedal command for driving at a desired speed. The steering wheel and gas pedal reference are sent to the robot control to achieve the driving task with the humanoid. We present results from a driving experience with a real car and the humanoid robot HRP-2Kai. Part of the framework has been used to perform the driving task at the DARPA Robotics Challenge.

1 Introduction

The potential of humanoid robots in the context of disaster has been exhibited recently at the DARPA Robotics Challenge (DRC), where robots performed complex locomotion and manipulation tasks (DARPA Robotics Challenge, 2015). The DRC has shown that humanoids should be capable of operating machinery, originally designed for humans. The DRC utility car driving task is a good illustration of the complexity of such tasks.

Worldwide, to have the right to drive a vehicle, one needs to be delivered a license, requiring months of practice, followed by an examination test. To make a robot drive in similar conditions, the perception and control algorithms should reproduce the human driving skills.

If the vehicle can neither be customized nor automated, it is more convenient to think of a robot in terms of anthropomorphic design. A driving robot must have motion capabilities for operations such as: reaching the vehicle, entering it, sitting in a stable posture, controlling its commands (e.g., ignition, steering wheel, pedals), and finally egressing it. All these skills can be seen as action templates, to be tailored to each vehicle and robot, and, more importantly, to be properly combined and sequenced to achieve driving tasks.

Noticeable research is currently made, to automate the driving operation of unmanned vehicles, with the ultimate goal of reproducing the tasks usually performed by human drivers (Nunes et al., 2009; Zhang et al., 2008; Hentschel and Wagner, 2010), by relying on visual sensors (Newman et al., 2009; Broggi et al., 2010; Cherubini et al., 2014). The success of the DARPA Urban Challenges (Buehler et al., 2008; Thrun et al., 2006), and the impressive demonstrations made by Google (Google, 2015), have heightened expectations that autonomous cars will very soon be able to operate in urban environments. Considering this, why bother making a robot drive a car, if the car can make its way without a robot? Although both approaches are not exclusive, this is certainly a legitimate question.

One possible answer springs from the complexity of autonomous cars, which host a *distributed robot*, with various sensors and actuators controlling the different tasks. With a *centralized robot*, such embedded devices can be removed from the car. The reader may also wonder when should a *centralized* robot be preferred to a *distributed* one, i.e., a fully automated car?

We answer this question through concrete application examples. In the DRC (Pratt and Manzo, 2013), one of the eight tasks that robot must overtake is driving a utility vehicle. The reason is that in disaster situations, the intervention robot must operate vehicles – usually driven by humans – to transport tools, debris, etc. Once the vehicle reaches the intervention area, the robot should execute other tasks, (e.g., turning a valve, operating a drill). Without a humanoid, these tasks can be hardly achieved by a unique system. Moreover, the robot should operate cranks or other tools attached to the vehicle (Hasunuma et al., 2003; Yokoi et al., 2006). A second demand comes from the car manufacturing industry (Hirata et al., 2015). In fact, current crash-tests dummies are passive and non-actuated. Instead, in crash situations, real humans perform protective motions and stiffen their body, all behaviors that are programmable on humanoid robots. Therefore, robotic crash-test dummies would be

more realistic in reproducing typical human behaviors.

These applications, along with the DRC itself, and with the related algorithmic questions, motivate the interest for developing a robot driver. However, this requires the solution of an unprecedented "humanoid-in-the-loop" control problem. In our work, we successfully address this, and demonstrate the capability of a humanoid robot to drive a real car. This work is based on preliminary results carried out with the HRP-4 robot, driving a simulated car (Paolillo et al., 2014). Here, we add new features to that framework, and present experiments with humanoid HRP-2Kai driving a real car outdoor on an unknown road.

The proposed framework presents the following main features:

- car steering control, to keep the car at a defined center of the road;
- car velocity control, to drive the car at a desired speed;
- admittance control, to ensure safe manipulation of the steering wheel;
- three different driving strategies, allowing intervention or supervision of a human operator, in a smooth shared autonomy manner.

The modularity of the approach allows to easily enable or disable each of the modules that compose the framework. Furthermore, to achieve the driving task, we propose to use only standard sensors for a common full-size humanoid robot, i.e., a monocular camera mounted on the head of the robot, the Inertial Measurement Unit (IMU) in the chest, and the force sensors at the wrists. Finally, the approach being purely reactive, it does not need any a priori knowledge of the environment. As a result, the framework allows - under certain assumptions - to make the robot drive along a previously unknown road.

The paper organization reflects the schematic description of the approach given in the next Sect. 2, at the end of which we also provide a short description of the paper sections.

2 Problem formulation and proposed approach

The objective of this work is to enable a humanoid robot to autonomously drive a car at the center of an unknown road, at a desired velocity. More specifically, we focus on the driving task and, therefore, consider the robot sitting in the car, already in a correct driving posture.

Most of the existing approaches have achieved this goal by relying on teleoperation (DRC-Teams, 2015; Kim et al., 2015; McGill et al., 2015). Atkeson and colleagues (Atkeson et al., 2015) propose an hybrid solution, with teleoperated steering and autonomous speed control. The velocity of the car, estimated with stereo cameras, is fed back to a PI controller, while LIDAR, IMU and visual odometry data support the operator during the steering procedures. In (Kumagai et al., 2015), the gas pedal is teleoperated and a local planner, using robot

kinematics for vehicle path estimation, and point cloud data for obstacle detection, enables autonomous steering. An impedance system is used to ensure safe manipulation of the steering wheel.

Other researchers have proposed fully autonomous solutions. For instance, in (Jeong et al., 2015), autonomous robot driving is achieved by following the proper trajectory among obstacles, detected with laser measurements. LIDAR scans are used in (Rasmussen et al., 2014) to plan a path for the car, while the velocity is estimated with a visual odometry module. The operation of the steering wheel and gas pedal is realized with simple controllers.

We propose a reactive approach for autonomous driving that relies solely on standard humanoids sensor equipment, thus making it independent from the vehicle sensorial capabilities, and does not require expensive data elaboration for building local representations of the environment and planning safe paths. In particular, we use data from the robot on-board camera and IMU, to close the autonomous driver feedback loop. The force measured on the robot wrists is exploited to operate the car steering wheel.

In designing the proposed solution, some simplifying assumptions have been introduced, to capture the conceptual structure of the problem, without losing generality:

1. The car brake and clutch pedals are not considered, and the driving speed is assumed to be positive and independently controlled through the gas pedal. Hence, the steering wheel and the gas pedals are the only vehicle controls used by the robot for driving.

2. The robot is already in its driving posture on the seat, with one hand on the steering wheel, the foot on the pedal, and the camera pointing the road, with focal axis aligned with the car sagittal plane. The hand grasping configuration is unchanged during operation.

3. The road is assumed to be locally flat, horizontal, straight, and delimited by parallel borders[1]. Although global convergence can be proved only for straight roads, turns with admissible curvature bounds are also feasible, as shown in the Experimental section. Instead, crossings, traffic lights, and pedestrians are not negotiated, and road signs are not interpreted.

Given these assumptions, we propose the control architecture in Fig. 1. The robot sits in the car, with its camera pointing to the road. The acquired images and IMU data are used by two branches of the framework running in parallel: car steering and velocity control. These are described hereby.

The car steering algorithm guarantees that the car is maintained at the center of the road. To this end, the IMU is used to get the camera orientation with respect to the road, while an image processing algorithm detects the road borders (*road detection*). These borders

[1]The assumption on parallel road borders can be relaxed, as proved in (Paolillo et al., 2016). We maintain the assumption here to keep the description of the controller simpler, as will be shown in Sect. 5.1.

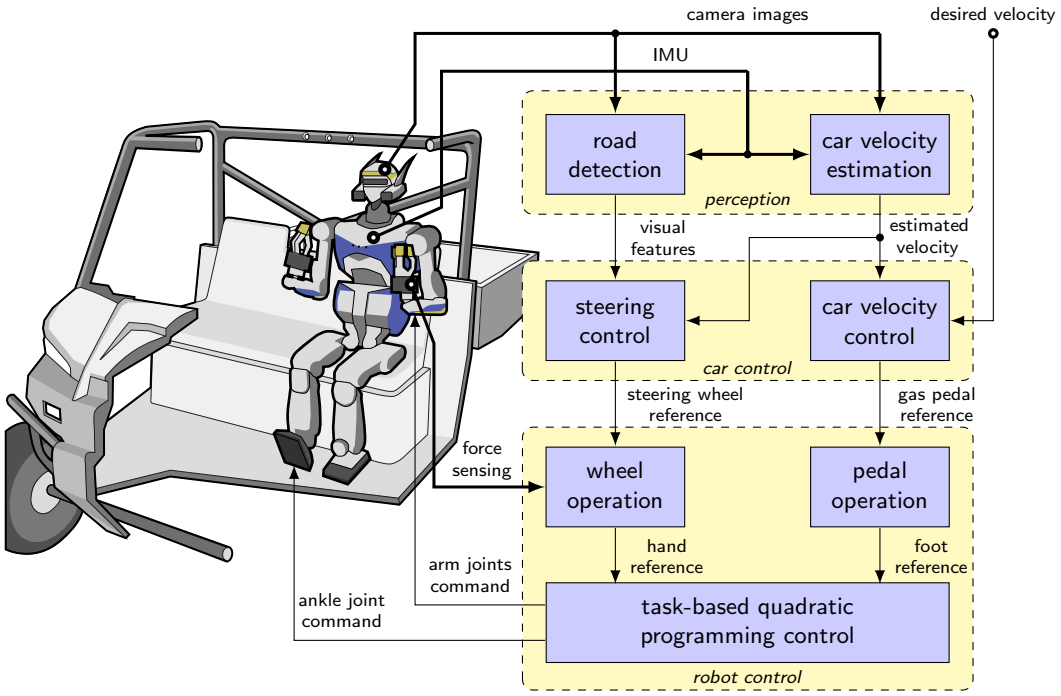


Figure 1: Conceptual block diagram of the driving framework.

are used to compute the visual features feeding the *steering control* block. Finally, the computed steering wheel reference angle is transformed by the *wheel operation* block into a desired trajectory for the robot hand that is operating the steering wheel. This trajectory can be adjusted by an admittance system, depending on the force exchanged between the robot hand and the steering wheel.

The car velocity control branch aims at making the car progress at a desired speed, through the gas pedal operation by the robot foot. A Kalman Filter (KF) fuses visual and inertial data to estimate the velocity of the vehicle (*car velocity estimation*) sent as feedback to the *car velocity control*, which provides the gas pedal reference angle for obtaining the desired velocity. The *pedal operation* block transforms this signal into a reference for the robot foot.

Finally, the reference trajectories for the hand and the foot respectively operating the steering wheel and the pedal, are converted into robot postural tasks, by the *task-based quadratic programming controller*.

The driving framework, as described above, allows a humanoid robot to autonomously drive a car along an unknown road, at a desired velocity. We further extend the versatility of our framework by implementing three different "driving modes", in order to ease human supervision and eventual intervention if needed:

- *Autonomous*. Car steering and velocity control are both enabled, as indicated above, and the robot autonomously drives the car without any human aid.

Table 1: Driving modes. For each mode, the steering and the car velocity control are properly enabled or disabled.

Driving mode	**Steering control**	**Car velocity control**
Autonomous	enabled	enabled
Assisted	enabled*	disabled
Teleoperated	disabled	disabled

* Road detection is assisted by the human.

- *Assisted*. A human takes care of the *road detection*, the *car velocity estimation*, and the *control*, by teleoperating the robot ankle, and manually selecting the visual features (road borders). These are then used by the steering controller to compute the robot arm command.
- *Teleoperated*. Both the robot hand and foot are teleoperated for steering the wheel and the gas pedal operation, respectively. The reference signals are sent to the task-based quadratic programming control through a keyboard or joystick. The human uses the robot camera images as visual feedback for driving.

For each of the driving modes, the car steering and velocity controllers are enabled or disabled, as described in Table 1. The human user/supervisor can intervene at any moment during the execution of the driving task, to select one of the three driving modes. The selection, as well as the switching between modes, is done by pushing proper joystick (or keyboard) buttons.

The framework has a modular structure, as presented in Fig. 1. In the following Sections, we detail the primitive functionalities required by the autonomous mode, since the assisted and teleoperation modes use a subset of such functionalities.

The rest of paper is organized as follows. Section 3 describes the model used for the car-robot system. Then, the main components of the proposed framework are detailed. Sect. 4 presents the perception part, i.e., the algorithms used to detect the road and to estimate the car velocity. Section 5 deals with car control, i.e., how the feedback signals are transformed into references for the steering wheel and for the gas pedal, while Sect. 6 focuses on humanoid control, i.e., on the computation of the commands for the robot hand and foot. The experiments carried out with HRP-2Kai are presented in Sect. 7. Finally, Sect. 8 concludes the paper and outlines future research perspectives.

3 Modelling

The design of the steering controller is based on the car kinematic model. This is a reasonable choice since, for nonholonomic systems, it is possible to cancel the dynamic parameters via

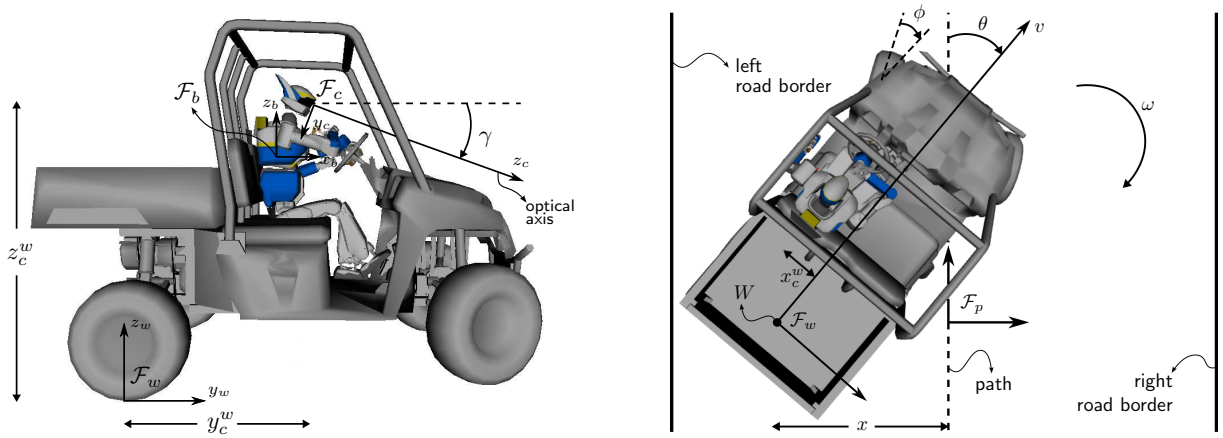


Figure 2: Side (a) and top view (b) of a humanoid robot driving a car with relevant variables.

feedback, and to solve the control problem at the velocity level, provided that the velocity issued by the controller is differentiable (De Luca and Oriolo, 1995). To recover the dynamic system control input, it is however necessary to know the exact dynamic model, which is in general not available. Although some approximations are therefore necessary, these do not affect the controller in the considered scenario (low accelerations, flat and horizontal road). On-line car dynamic parameter identification could be envisaged, and seamlessly integrated in our framework, whenever the above assumptions are not valid. Note, however, that the proposed kinematic controller would remain valid, since it captures the theoretic challenge of driving in the presence of nonholonomic constraints.

To derive the car control model, consider the reference frame $\mathcal{F}_w$ placed on the car rear axle midpoint W, with the y-axis pointing forward, the z-axis upward and the x-axis completing the right handed frame (see Fig. 2a). The path to be followed is defined as the set of points that maximize the distance from both the left and right road borders. On this path, we consider a tangent Frenet Frame $\mathcal{F}_p$, with origin on the normal projection of W on the path. Then, the car configuration with respect to the path is defined by x, the Cartesian abscissa of W in $\mathcal{F}_p$, and by θ, the car orientation with respect to the path tangent (see Fig. 2b). Describing the car motion through the model of a unicycle, with an upper curvature bound $c_M \in \mathbb{R}^+$, x and θ evolve according to:

$$\begin{aligned} \dot{x} &= v \sin\theta \\ \dot{\theta} &= \omega \end{aligned} \qquad \left|\frac{\omega}{v}\right| < c_M, \tag{1}$$

where v and ω represent respectively the linear and angular velocity of the unicycle. The front wheel orientation ϕ can be approximately related to v and ω through:

$$\phi = \arctan\left(\frac{\omega l}{v}\right), \tag{2}$$

with l the constant distance between the rear and front wheel axes[2].

[2]Bounds on the front wheels orientation characterizing common service cars induce the maximum curva-

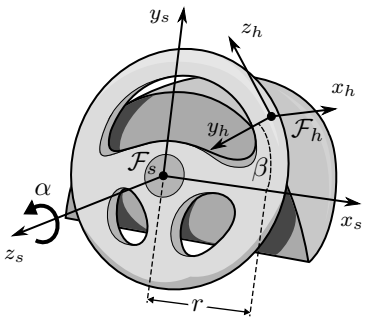


Figure 3: The steering wheel, with rotation angle α, hand and steering frames, $\mathcal{F}_h$ and $\mathcal{F}_s$. The parameters r, the radius of the wheel, and β, characterizing the grasp configuration, are also shown here.

Note that a complete car-like model could have been used, for control design purposes, by considering the front wheels orientation derivative as the control input. The unicycle stabilizing controller adopted in this paper can in fact be easily extended to include the dynamics of the front wheels orientation, for example through backstepping techniques. However, in this case, a feedback from wheel orientation would have been required by the controller, but is, generally, not available. A far more practical solution is to neglect the front wheels orientation dynamics, usually faster than that of the car, and consider a static relationship between the front wheels orientation and the car angular velocity. This will only require a rough guess on the value of the parameter l, since the developed controller shows some robustness with respect to model parameters uncertainties as will be shown in Sect. 5.

The steering wheel is shown in Fig. 3, where we indicate, respectively with $\mathcal{F}_h$ and $\mathcal{F}_s$, the hand and steering wheel reference frames. The origin of $\mathcal{F}_s$ is placed at the center of the wheel, and α is the rotation around its z-axis, that points upward. Thus, positive values of α make the car turn left (i.e., lead to negative ω).

Neglecting the dynamics of the steering mechanism (Mohellebi et al., 2009), assuming the front wheels orientation ϕ to be proportional to the steering wheel angle α, controlled by the driver hands, and finally assuming small angles $\omega l/v$ in (2), leads to:

$$\alpha = k_\alpha \frac{\omega}{v}, \tag{3}$$

with k_α a negative[3] scalar, characteristic of the car, accounting also for l.

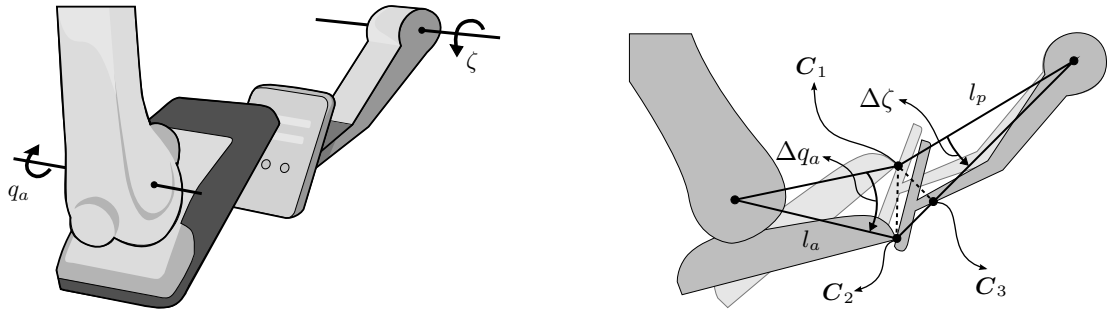
The gas pedal is modeled by its inclination angle ζ, that yields a given car acceleration $a = dv/dt$. According to experimental observations, at low velocities, the relationship between the pedal inclination and the car acceleration is linear:

$$\zeta = k_\zeta a. \tag{4}$$

The pedal is actuated by the motion of the robot foot, that is pushing it (see Fig. 4a).

ture constraint in (1).

[3]Because of the chosen angular conventions.



(a) (b)

Figure 4: (a) The robot foot operates the gas pedal by regulating the joint angle at the ankle q_a, to set a pedal angle ζ, and yield car acceleration a. (b) Geometric relationship between the ankle and the gas pedal angles.

Assuming small values of Δq_a and $\Delta\zeta$, the point of contact between the foot and the pedal can be considered fixed on both the foot and the pedal, i.e., the length of the segment $\overline{C_2C_3}$ in Fig. 4b can be considered close to zero[4]. Hence, the relationship between Δq_a and $\Delta\zeta$ is easily found to be

$$\Delta\zeta = \frac{l_a}{l_p}\Delta q_a, \tag{5}$$

where l_a (l_p) is the distance of the ankle (pedal) rotation axis from the contact point of the foot with the pedal.

The robot body reference frame $\mathcal{F}_b$ is placed on the robot chest, with x-axis pointing forward, and z-axis upward. Both the accelerations measured by the IMU, and the humanoid tasks, are expressed in this frame. We also indicate with $\mathcal{F}_c$ the robot camera frame (see Fig. 2). Its origin is in the optical center of the camera, with z-axis coincident with the focal axis. The y-axis points downwards, and the x-axis completes the right-handed frame. $\mathcal{F}_c$ is tilted by an angle γ (taken positive downwards) with respect to the frame $\mathcal{F}_w$, whereas the vector $\boldsymbol{p_c^w} = (x_c^w, y_c^w, z_c^w)^T$ indicates the position vector of the camera frame expressed in the car reference frame.

Now, the driving task can be formulated. It consists in leading the car on the path, and aligning it with the path tangent:

$$(x, \theta) \rightarrow (0, 0)\,, \tag{6}$$

while driving at a desired velocity:

$$v \rightarrow v^*. \tag{7}$$

Task (6) is achieved by the steering control that uses the kinematic model (1), and is realized by the robot hand according to the steering angle α. Concurrently, (7) is achieved by the car

[4]For the sake of clarity, in Fig. 4b the length of the segment $\overline{C_2C_3}$ is much bigger than zero. However, this length, along with angles Δq_a and $\Delta\zeta$, is almost null.

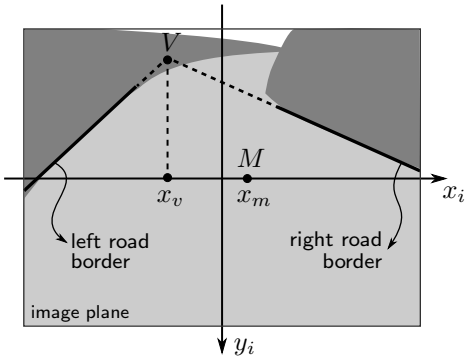


Figure 5: The images of the road borders define the middle and vanishing point, respectively M and V. Their abscissa values are denoted with x_m and x_v.

velocity control realized by the robot foot that sets a proper angle ζ for the gas pedal. The computation of α and ζ rely on the perception module, that is detailed in the next Section.

4 Perception

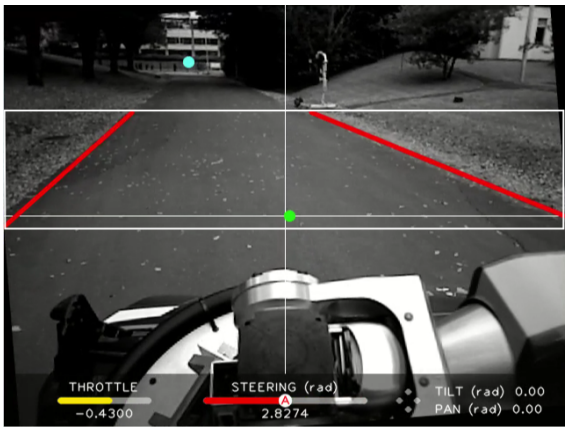
The block diagram of Fig. 1 shows our perception-action approach. At a higher level, the perception block, whose details are described in this Section, provides the feedback signals for the car and robot control.

4.1 Road detection

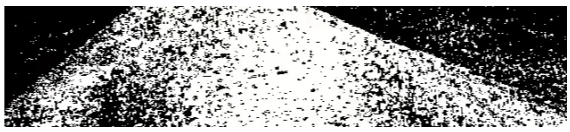
This Section describes the procedure used to derive the road visual features, required to control the steering wheel. These visual features are: (i) the *vanishing point* (V), i.e., the intersection of the two borders, and (ii) the *middle point* (M), i.e., the midpoint of the segment connecting the intersections of the borders with the image horizontal axis. Both are shown in Fig. 5.

Hence, *road detection* consists of extracting the road borders from the robot camera images. After this operation, deriving the vanishing and middle point is trivial. Since the focus of this work is not to advance the state-of-the-art on road/lane detection, but rather to propose a control architecture for humanoid car driving, we develop a simple image processing algorithm for road border extraction. More complex algorithms can be used to improve the detection and tracking of the road (Liu et al., 2008; Lim et al., 2009; Meuter et al., 2009; Nieto et al., 2012), or even to detect road markings (Vacek et al., 2007). However, our method has the advantage of being based solely on vision, avoiding the complexity induced by integration of other sensors (Dahlkamp et al., 2006; Ma et al., 2000). Note that more advanced software is owned by car industries, and therefore hard to find in open-code source or binary.

Part of the road borders extraction procedure follows standard techniques used in the field



(a) On-board camera image with the red detected road borders. The vanishing and middle point are shown respectively in cyan and green.



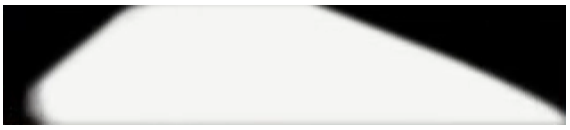
(b) First color detection.



(c) Second color detection.



(d) Mask obtained after dilation and erosion.



(e) Convex hull after Gaussian filtering.



(f) Canny edge detection.



(g) Hough transform.



(h) Merged segments.

Figure 6: Main steps of the road detection algorithm. Although the acquired robot image (a) is shown in gray-scale here, the proposed road detection algorithm processes color images.

of computer vision (Laganière, 2011) and is based on the OpenCV library (Bradski, 2000) that provides ready-to-use methods for our vision-based algorithm. More in detail, the steps used for the detection of the road borders on the currently acquired image are described below, with reference to Fig. 6.

- From the image, a Region Of Interest (ROI), shown with white borders in Fig. 6a, is manually selected at the initialization, and kept constant during the driving experiment. Then, at each cycle of the image processing, we compute the average and standard deviation of hue and saturation channels of the HSV (Hue, Saturation and Value) color space on two central rectangular areas in the ROI. These values are considered for the thresholding operations described in the next step.
- Two binary images (Fig. 6b and 6c) are obtained by discerning the pixels in the ROI, whose hue and saturation value are in the ranges (average ± standard deviation) defined in the previous step. This operation allows to detect the road, while being adaptive to color variation. The HSV value channel is not considered, in order to be

robust to luminosity changes.

- To remove "salt and pepper noise", the dilation and erosion operators are applied to the binary images. Then, the two images are merged by using the OR logic operator to obtain a mask of the road (Fig. 6d).
- The convex hull is computed with areas greater than a given threshold on the mask found in the previous step; then, a Gaussian filter is applied for smoothing. The result is shown in Fig. 6e.
- The Canny edge detector (Fig. 6f), followed by Hough transform (Fig. 6g) are applied, to detect the line segments on the image.
- Similar segments are merged[5], as depicted in Fig. 6h.

This procedure gives two lines corresponding to the image projection of the road borders. However, in real working conditions, it may happen that one or both the borders are not detectable because of noise on the image, or failures in the detection process. For this reason, we added a recovery strategy, as well as a tracking procedure, to the pipeline. The recovery strategy consists in substituting the borders, that are not detected, with artificial ones, defined offline as oblique lines that, according to the geometry of the road and to the configuration of the camera, most likely correspond to the road borders. This allows the computation of the vanishing and middle point even when one (or both) real road borders are not correctly detected. On the other hand, the tracking procedure gives continuity and robustness to the detection process, by taking into account the borders detected on the previous image. It consists of a simple KF, with state composed of the slope and intercept of the two borders[6]. In the prediction step, the KF models the position of lines on the image plane as constant (a reasonable design choice, under Assumption 3, of locally flat and straight road), whereas the measurement step uses the road borders as detected in the current image.

From the obtained road borders (shown in red in Fig.6a), the vanishing and middle point are derived, with simple geometrical computations. Their values are then smoothed with a low-pass frequency filter, and finally fed to the steering control, that will be described in Sect. 5.1.

4.2 Car velocity estimation

To keep the proposed framework independent from the car characteristics, we propose to estimate the car speed v, by using only the robot sensors, and avoiding information coming from the car equipment, such as GPS, or speedometer. To this end, we use the robot camera,

[5]For details on this step, refer to (Paolillo et al., 2016).

[6]Although 3 parameters are sufficient if the borders are parallel, a 4-dimensional state vector will cover all cases, while guaranteeing robustness to image processing noise.

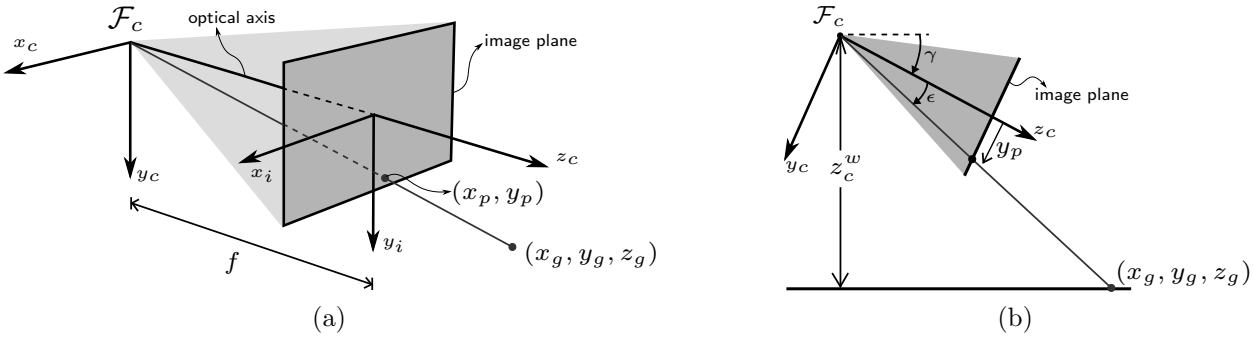


Figure 7: Schematic representation of the robot camera looking at the road. (a) Any visible cartesian point (x_g, y_g, z_g) on the ground has a projection on the camera image plane, whose coordinates expressed in pixels are (x_p, y_p). (b) The measurement of this point on the image plane, together with the camera configuration parameters, can be used to estimate the depth z_g of the point.

to measure the optical flow, i.e. the apparent motion of selected visual features, due to the relative motion between camera and scene.

The literature in the field of autonomous car control provides numerous methods for estimating the car speed by means of optical flow (Giachetti et al., 1998; Barbosa et al., 2007). To improve the velocity estimate, the optical flow can be fused with inertial measurements, as done in the case of aerial robots, in (Grabe et al., 2012). Inspired by that approach, we design a KF, fusing the acceleration measured by the robot IMU and the velocity measured with optical flow.

Considering the linear velocity and acceleration along the forward car axis y_w as state $\boldsymbol{\xi} = (v\ a)^T$ of the KF, we use a simple discrete-time stochastic model to describe the car motion:

$$\boldsymbol{\xi}_{k+1} = \begin{pmatrix} 1 & \Delta T \\ 0 & 1 \end{pmatrix} \boldsymbol{\xi}_k + \boldsymbol{n}_k, \tag{8}$$

with ΔT the sampling time, and $\boldsymbol{n}_k$ the zero-mean white gaussian noise. The corresponding output of the KF is modeled as:

$$\boldsymbol{\eta}_k = \boldsymbol{\xi}_k + \boldsymbol{m}_k, \tag{9}$$

where $\boldsymbol{m}_k$ indicates the zero-mean white gaussian noise associated to the measurement process. The state estimate is corrected, thanks to the computation of the residual, i.e., the difference between measured and predicted outputs. The measurement is based on both the optical flow (v_{OF}), and the output of the IMU accelerometers (a_{IMU}). Then, the estimation of the car velocity v will correspond to the first element of state vector $\boldsymbol{\xi}$. The process to obtain v_{OF} and a_{IMU} is detailed below.

4.2.1 Measure of the car speed with optical flow

To measure the car velocity v_{OF} in the KF, we use optical flow. Optical flow can be used to reconstruct the motion of the camera, and from that, assuming that the transformation from the robot camera frame to the car frame is known, it is straightforward to derive the vehicle velocity.

More in detail, the 6D velocity vector $\boldsymbol{v}_c$ of the frame $\mathcal{F}_c$ can be related to the velocity of the point tracked in the image $\dot{\boldsymbol{x}}_p$ through the following relation:

$$\dot{\boldsymbol{x}}_p = \boldsymbol{L}\boldsymbol{v}_c, \tag{10}$$

where the interaction matrix $\boldsymbol{L}$ is expressed as follows (Chaumette and Hutchinson, 2006):

$$\boldsymbol{L} = \begin{pmatrix} -\frac{S_x}{z_g} & 0 & \frac{x_p}{z_g} & \frac{x_p y_p}{S_y} & -(S_x + \frac{x_p^2}{S_x}) & \frac{y_p S_x}{S_y} \\ 0 & -\frac{S_y}{z_g} & \frac{y_p}{z_g} & S_y + \frac{y_p^2}{S_y} & -\frac{x_p y_p}{S_y} & -\frac{x_p S_y}{S_x} \end{pmatrix}. \tag{11}$$

Here, (x_p, y_p) are the image coordinates (in pixels) of the point on the ground, expressed as (x_g, y_g, z_g) in the camera frame (see Fig. 7). Furthermore, it is $S_{x,y} = f\alpha_{x,y}$, where f is the camera focal length and α_x/α_y the pixel aspect ratio. In the computation of $\boldsymbol{L}$, we consider that the image principal point coincides with the image center. As shown in Fig. 7b, the point depth z_g can be reconstructed through the image point ordinate y_p and the camera configuration (tilt angle γ and height z_c^w):

$$z_g = \frac{z_c^w \cos\epsilon}{\sin(\gamma + \epsilon)}, \quad \epsilon = \arctan\left(\frac{y_p}{S_y}\right). \tag{12}$$

Actually, the camera velocity $\boldsymbol{v}_c$ is computed by taking into account n tracked points, i.e., in (10), we consider respectively $\hat{\boldsymbol{L}} = (\boldsymbol{L}_1 \cdots \boldsymbol{L}_n)^T$ and $\hat{\dot{\boldsymbol{x}}}_p = (\dot{\boldsymbol{x}}_{p,1} \cdots \dot{\boldsymbol{x}}_{p,n})^T$, instead of $\boldsymbol{L}$ and $\dot{\boldsymbol{x}}_p$. Then, $\boldsymbol{v}_c$ is obtained by solving a least-squares problem[7]:

$$\boldsymbol{v}_c = \arg\min_{\boldsymbol{\chi}} ||\hat{\boldsymbol{L}}\boldsymbol{\chi} - \hat{\dot{\boldsymbol{x}}}_p||_2. \tag{13}$$

The reconstruction of $\hat{\dot{\boldsymbol{x}}}_p$ in (13) is based on the computation of the optical flow. However, during the navigation of the car, the vibration of the engine, poor textured views and other un-modeled effects add noise to the measurement process (Giachetti et al., 1998). Furthermore, other factors, such as variable light conditions, shadows, and repetitive textures, can jeopardize feature tracking. Therefore, raw optical flow, as provided by off-the-shelf algorithms –e.g., from the OpenCV library (Bradski, 2000), gives noisy data that are insufficient for accurate velocity estimation; so filtering and outlier rejection techniques must be added.

[7]To solve the least-square problem, $n \geq 3$ points are necessary. In our implementation, we used the openCV *solve* function, and in order to filter the noise due to few contributions, we set $n \geq 25$. If $n < 25$, we set $\boldsymbol{v}_c = 0$.

Since the roads are generally poor in features, we use a *dense* optical flow algorithm, that differs from *sparse* algorithms, in that it computes the apparent motion of all the pixels of the image plane. Then, we filter the dense optical flow, first according to geometric rationales, and then with an outlier rejection method (Barbosa et al., 2007). The whole procedure is described below, step-by-step:

- Take two consecutive images from the robot on-board camera.
- Consider only the pixels in a ROI that includes the area of the image plane corresponding to the road. This ROI is kept constant along all the experiment and, thus, identical for the two consecutive frames.
- Covert the frames to gray scale, apply a Gaussian filter, and equalize with respect to the histogram. This operation reduces the measurement noise, and robustifies the method with respect to light changes.
- Compute the dense optical flow, using the Farnebäck algorithm (Farnebäck, 2003) implemented in OpenCV.
- Since the car is supposed to move forward, in the dense optical flow vector, consider only those elements pointing downwards on the image plane, and discard those not having a significant centrifugal motion from the principal point. Furthermore, consider only contributions with length between an upper and a lower threshold, and whose origin is on an image edge (detected applying Canny operator).
- Reject the outliers, i.e., the contributions $(\dot{x}_{p,i}, \dot{y}_{p,i})$, $i \in \{1, \ldots, n\}$, such that $\dot{x}_{p,i} \notin [\bar{\dot{x}}_p \pm \sigma_x]$ and $\dot{y}_{p,i} \notin [\bar{\dot{y}}_p \pm \sigma_y]$, where $\bar{\dot{x}}_p$ ($\bar{\dot{y}}_p$) and σ_x (σ_y) are the average and standard deviation of the optical flow horizontal (vertical) contributions. This operation is made separately for the contributions of the right and left side of the image, where the module and the direction of the optical flow vectors can be quite different (e.g., on turns).

The final output of this procedure, $\hat{\dot{\boldsymbol{x}}}_p$, is fed to (13), to obtain $\boldsymbol{v}_c$, that is then low-pass filtered. To transform the velocity $\boldsymbol{v}_c$ in frame $\mathcal{F}_c$, obtained from (13), into velocity $\boldsymbol{v}_w$ in the car frame $\mathcal{F}_w$, we apply:

$$\boldsymbol{v}_w = \boldsymbol{W}_c^w \boldsymbol{v}_c, \tag{14}$$

with $\boldsymbol{W}_c^w$ the twist transformation matrix

$$\boldsymbol{W}_c^w = \begin{pmatrix} \boldsymbol{R}_c^w & \boldsymbol{S}_c^w \boldsymbol{R}_c^w \\ \boldsymbol{0}_{3\times 3} & \boldsymbol{R}_c^w \end{pmatrix}, \tag{15}$$

$\boldsymbol{R}_c^w$ the rotation matrix from car to camera frame, and $\boldsymbol{S}_c^w$ the skew symmetric matrix associated to the position $\boldsymbol{p}_c^w$ of the origin of $\mathcal{F}_c$ in $\mathcal{F}_w$.

Finally, the speed of the car is set as the y-component of $\boldsymbol{v}_w$: $v_{OF} = v_{w,y}$. This will constitute the first component of the KF measurement vector.

4.2.2 Measure of the car acceleration with robot accelerometers

The IMU mounted on-board the humanoid robot is used to measure acceleration, in order to improve the car velocity estimation through the KF. In particular, given the raw accelerometer data, we first compensate the gravity component, with a calibration executed at the beginning of each experiment[8]. This gives $\boldsymbol{a}_b$, the 3D robot acceleration, expressed in the robot frame $\mathcal{F}_b$. Then, we transform $\boldsymbol{a}_b$ in the car frame $\mathcal{F}_w$, to obtain:

$$\boldsymbol{a}_w = \boldsymbol{R}_b^w \boldsymbol{a}_b, \tag{16}$$

where $\boldsymbol{R}_b^w$ is the rotation matrix relative to the robot body - vehicle transformation. Finally, a_{IMU} is obtained by selecting the y-component of $\boldsymbol{a}_w$. This will constitute the second component of the KF measurement vector.

5 Car control

The objective of car control is (i) to drive the rear wheel axis center W along the curvilinear path that is equally distant from the left and right road borders (see Fig. 2b), while aligning the car with the tangent to this path, and (ii) to track desired vehicle velocity v^*. Basically, car control consists in achieving tasks (6) and (7), with the steering and car velocity controllers described in the following subsections.

5.1 Steering control

Given the visual features extracted from the images of the robot on-board camera, the vision-based steering controller generates the car angular velocity input ω to regulate both x and θ to zero. This reference input is eventually translated in motion commands for the robot hands.

The controller is based on the algorithm introduced by (Toibero et al., 2009) for unicycle corridor following, and recently extended to the navigation of humanoids in environments with corridors connected through curves and T-junctions (Paolillo et al., 2016). In view of Assumption 3 in Sect. 2, the same algorithm can be applied here. For the sake of completeness, in the following, we briefly recall the derivation of the features model (that can be found, for example, also in (Vassallo et al., 2000)) and the control law originally presented by (Toibero et al., 2009). In doing so, we illustrate the adaptations needed to deal with the specificity of our problem.

The projection matrix transforming the homogeneous coordinates of a point, expressed in $\mathcal{F}_p$, to its homogeneous coordinates in the image, is:

$$\boldsymbol{P} = \boldsymbol{K}\ \boldsymbol{T}_w^c\ \boldsymbol{T}_p^w, \tag{17}$$

[8]The assumption on horizontal road in Sect. 3 avoids the need for repeating this calibration.

where $\boldsymbol{K}$ is the camera calibration matrix (Ma et al., 2003), $\boldsymbol{T}_w^c$ the transformation from the car frame $\mathcal{F}_w$ to $\mathcal{F}_c$, and $\boldsymbol{T}_p^w$ from the path frame $\mathcal{F}_p$ to $\mathcal{F}_w$.

As intuitive from Fig. 2, the projection matrix depends on both the car coordinates, and the camera intrinsic and extrinsic parameters. Here, we assume that the camera principal point coincides with the image center, and we neglect image distortion. Furthermore, $\boldsymbol{P}$ has been computed neglecting the z-coordinates of the features, since they do not affect the control task. Under these assumptions, using $\boldsymbol{P}$, the abscissas of the vanishing and middle point, respectively denoted by x_v and x_m, can be expressed as (Toibero et al., 2009; Vassallo et al., 2000):

$$\begin{aligned} x_v &= k_1 \tan\theta \\ x_m &= k_2 \frac{x}{c_\theta} + k_3 \tan\theta + k_4, \end{aligned} \tag{18}$$

where

$$\begin{aligned} k_1 &= -S_x / c_\gamma \\ k_2 &= -S_x s_\gamma / z_c^w \\ k_3 &= -S_x c_\gamma - S_x s_\gamma y_c^w / z_c^w \\ k_4 &= -S_x s_\gamma x_c^w / z_c^w. \end{aligned}$$

We denote $\cos(*)$ and $\sin(*)$ with c_* and s_*, respectively. Note that with respect to the visual features model in (Toibero et al., 2009; Vassallo et al., 2000), the expression of the middle point changes, due to the introduction of the lateral and longitudinal displacement, x_c^w and y_c^w respectively, of the camera frame with respect to the car frame. As a consequence, to regulate the car position to the road center, we must define a new visual feature $\bar{x}_m = x_m - k_4$. Then, the navigation task (6) is equivalent to the following visual task:

$$(\bar{x}_m, x_v) \to (0, 0)\,. \tag{19}$$

In fact, according to (18), asymptotic convergence of x_v and $\bar{x}_m$ to zero implies convergence of x and θ to zero, achieving the desired path following task.

Feedback stabilization of the dynamics of $\bar{x}_m$, is given by the following angular velocity controller (Toibero et al., 2009):

$$\omega = \frac{k_1}{k_1 k_3 + \bar{x}_m x_v} \left(-\frac{k_2}{k_1} v x_v - k_p \bar{x}_m \right), \tag{20}$$

with k_p a positive scalar gain. This controller guarantees asymptotic convergence of both $\bar{x}_m$ and x_v to zero, under the conditions that $v > 0$, and that k_2 and k_3 have the same sign, which is always true if (i) $\gamma \in (0, \pi/2)$ and (ii) $y_c^w > -z_c^w / \tan\gamma$, two conditions always verified with the proposed setup.

Note that this controller has been obtained considering the assumption of parallel road borders. Nevertheless, this assumption can be easily relaxed since we showed in (Paolillo

et al., 2016) that the presence of non-parallel borders does not jeopardize the controller's local convergence.

To realize the desired ω in (20), the steering wheel must be turned according to (3):

$$\alpha = \frac{k_\alpha k_1}{k_1 k_3 + \bar{x}_m x_v} \left(-\frac{k_2}{k_1} x_v - k_p \frac{\bar{x}_m}{v} \right), \tag{21}$$

where $\bar{x}_m$ and x_v are obtained by the image processing algorithm of Sect. 4.1, while the value of v is estimated through the velocity estimation module presented in Sect. 4.2.

5.2 Car velocity control

In view of the assumption of low acceleration, and by virtue of the linear relationship between the car acceleration and the pedal angle (eq. (4)), to track a desired car linear velocity v^* we designed a PID feedback controller to compute the gas pedal command:

$$\zeta = k_{v,p}\ e_v + k_{v,i} \int e_v + k_{v,d}\ \frac{d}{dt} e_v. \tag{22}$$

Here, $e_v = (v^* - v)$ is the difference between the desired and current value of the velocity, as computed by the car velocity estimation block, while $k_{v,p}$, $k_{v,i}$ and $k_{v,d}$ are the positive proportional, integral and derivative gains, respectively. In the design of the velocity control law, we decided to insert an integral action to compensate for constant disturbances (like, e.g., the effect of a small road slope) at steady state. The derivative term helped achieving a damped control action. The desired velocity v^* is set constant here.

6 Robot control

This section presents the lower level of our controller, which enables the humanoid robot to turn the driving wheel by α, and push the pedal by ζ.

6.1 Wheel operation

The reference steering angle α is converted to the reference pose of the hand grasping the wheel, through the rigid transformation

$$\boldsymbol{T}_h^{b*} = \boldsymbol{T}_s^b(\alpha)\, \boldsymbol{T}_h^s(r, \beta).$$

Here, $\boldsymbol{T}_h^{b*}$ and $\boldsymbol{T}_s^b$ are the transformation matrices expressing respectively the poses of frames $\mathcal{F}_h$ and $\mathcal{F}_s$ in Fig. 3 with respect to $\mathcal{F}_b$ in Fig. 2a. Constant matrix $\boldsymbol{T}_h^s$ expresses the pose of $\mathcal{F}_h$ with respect to $\mathcal{F}_s$, and depends on the steering wheel radius r, and on the angle β parameterizing the hand position on the wheel.

For a safe interaction between the robot hand and the steering wheel, it is obvious to think of an admittance or impedance controller, rather than solely a force or position controller (Hogan, 1985). We choose to use the following admittance scheme:

$$\boldsymbol{f} - \boldsymbol{f}^* = \boldsymbol{M}\Delta\ddot{\boldsymbol{x}} + \boldsymbol{B}\Delta\dot{\boldsymbol{x}} + \boldsymbol{K}\Delta\boldsymbol{x}, \tag{23}$$

where $\boldsymbol{f}$ and $\boldsymbol{f}^*$ are respectively the sensed and desired generalized interaction forces in $\mathcal{F}_h$; $\boldsymbol{M}$, $\boldsymbol{B}$ and $\boldsymbol{K} \in \mathbb{R}^{6\times6}$ are respectively the mass, damping and stiffness diagonal matrices. As a consequence of the force $\boldsymbol{f}$ applied on $\mathcal{F}_h$, and on the base of the values of the admittance matrices, (23) generates variations of pose $\Delta\boldsymbol{x}$, velocity $\Delta\dot{\boldsymbol{x}}$ and acceleration $\Delta\ddot{\boldsymbol{x}}$ of $\mathcal{F}_h$ with respect to $\mathcal{F}_s$. Thus, the solution of (23) leads to the vector $\Delta\boldsymbol{x}$ that can be used to compute the transformation matrix $\Delta\boldsymbol{T}$, and to build up the new desired pose for the robot hands:

$$\boldsymbol{T}_h^b = \boldsymbol{T}_h^{b*} \cdot \Delta\boldsymbol{T}. \tag{24}$$

In cases where the admittance controller is not necessary, we simply set $\Delta\boldsymbol{T} = \boldsymbol{I}$.

6.2 Pedal operation

Since there exists a linear relationship between the variation of the robot ankle and the variation of the gas pedal angle, to operate the gas pedal it is sufficient to move the ankle joint angle q_a. From (22), we compute the command for the robot ankle's angle as:

$$q_a = \frac{\zeta}{\zeta_{\max}}(q_{a,\max} - q_{a,\min}) + q_{a,\min}. \tag{25}$$

Here, $q_{a,\max}$ is the robot ankle configuration, at which the foot pushes the gas pedal, producing a significant car acceleration. Instead, at $q_a = q_{a,\min}$, the foot is in contact with the pedal, but not yet pushing it. These values depend both on the car type, and on the position of the foot with respect to the gas pedal. A calibration procedure is run before starting driving, to identify the proper values of $q_{a,\min}$ and $q_{a,\max}$. Finally, $\zeta_{\max}$ is set to avoid large accelerations, while saturating the control action.

6.3 Humanoid task-based control

As shown above, wheel and pedal operation are realized respectively in the operational space (by defining a desired hand pose $\boldsymbol{T}_h^b$) and in the articular space (via the desired ankle joint angle q_a). Both can be realized using our task-based quadratic programming (QP) controller, assessed in complex tasks such as ladder climbing (Vaillant et al., 2016). The joint angles and desired hand pose are formulated as errors that appear among the sum of weighted least-squares terms in the QP cost function. Other intrinsic robot constraints are formulated as linear expressions of the QP variables, and appear in the constraints. The QP controller is solved at each control step. The QP variable vector $\boldsymbol{x} = (\ddot{\boldsymbol{q}}^T, \boldsymbol{\lambda}^T)^T$, gathers the joint acceleration $\ddot{\boldsymbol{q}}$, and the linearized friction cones' base weights $\boldsymbol{\lambda}$, such that the contact forces $\boldsymbol{f}$ are equal to $\boldsymbol{K}_f\boldsymbol{\lambda}$ (with $\boldsymbol{K}_f$ the discretized friction cone matrix). The desired acceleration

$\ddot{\boldsymbol{q}}$ is integrated twice to feed the low level built-in PD control of HRP-2Kai. The driving task with the QP controller writes as follows:

$$\begin{aligned}
&\underset{\boldsymbol{x}}{\text{minimize}} \sum_{i=1}^{N} w_i \|\boldsymbol{E}_i(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})\|^2 + w_\lambda \|\boldsymbol{\lambda}\|^2 \\
&\text{subject to} \\
&\text{1) dynamic constraints} \\
&\text{2) sustained contact positions} \\
&\text{3) joint limits} \\
&\text{4) non-desired collision avoidance constraints} \\
&\text{5) self-collision avoidance constraints,}
\end{aligned} \tag{26}$$

where w_i and w_λ are task weights or gains, and $\boldsymbol{E}_i(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$ is the error in the task space. Details on the QP constraints (since they are common to most tasks) can be found in (Vaillant et al., 2016).

Here, we explicit the tasks used specifically during the driving (i.e. after the driving posture is reached). We use four ($N = 4$) set-point objective tasks; each task (i) is defined by its associated task-error $\boldsymbol{\epsilon}_i$ so that $\boldsymbol{E}_i = K_{p_i}\boldsymbol{\epsilon}_i + K_{v_i}\dot{\boldsymbol{\epsilon}}_i + \ddot{\boldsymbol{\epsilon}}_i$.

The driving wheel of the car has been modeled as another 'robot' having one joint (rotation). We then merged the model of the driving wheel to that of the humanoid and linked them, through a position and orientation constraint, so that the desired driving wheel steering angle α, as computed by (24), induces a motion on the robot (right arm) gripper. The task linking the humanoid robot to the driving wheel 'robot' is set as part of the QP constraints, along with all sustained contacts (e.g. buttock on the car seat, thighs, left foot).

The steering angle α (i.e. the posture of the driving wheel robot) is a set-point task ($\boldsymbol{E}_1$). The robot whole-body posture including the right ankle joint control (pedal) is also a set-point task ($\boldsymbol{E}_2$), which realizes the angle q_a provided by (25). Additional tasks were set to keep the gaze direction constant ($\boldsymbol{E}_3$), and to fix the left arm, to avoid collisions with the car cockpit during the driving operation ($\boldsymbol{E}_4$).

7 Experimental results

We tested our driving framework with the full-size humanoid robot HRP-2Kai built by Kawada Industries. For the experiments, we used the Polaris Ranger XP900, the same utility vehicle employed at the DRC. HRP-2Kai has 32 degrees of freedom, is 1.71 m tall and weighs 65 kg. It is equipped with an Asus Xtion Pro 3D sensor, mounted on its head and used in this work as a monocular camera. The Xtion camera provides images at 30 Hz with a resolution of 640×480 pixels. From camera calibration, it results $S_x \simeq S_y = 535$ pixels. In the presented experiments, $x_c^w = -0.4$ m, $y_c^w = 1$ m and $z_c^w = 1.5$ m were manually measured. However, it would be possible to estimate the robot camera position, with respect to the

car frame, by localization of the humanoid (Oriolo et al., 2015), or by using the geometric information of the car (that can be known, e.g., in the form of a CAD model, as shown in Fig. 2). HRP-2Kai is also equipped with an IMU (of rate 500 Hz) located in the chest. Accelerometer data have been merged with the optical flow to estimate the car linear velocity, as explained in Sect. 4.2. Furthermore, a built-in filter processes the IMU data to provide an accurate measurement of the robot chest orientation. This is kinematically propagated up to the Xtion sensor to get γ, the tilt angle of the camera with respect to the ground.

The task-based control is realized through the QP framework (see Sect. 6.3) which allows to easily set different tasks that can be achieved concurently by the robot. The following table gives the weights of the 4 set-point tasks described in Sect. 6.3. Note that $K_{v_i} = 2 \times \sqrt{K_{p_i}}$.

Table 2: QP weights and set-point gains.

	$\boldsymbol{E}_1$	$\boldsymbol{E}_2$	$\boldsymbol{E}_3$	$\boldsymbol{E}_4$
w	100	5	1000	1000
K_p	5	1 (ankle = 100)	10	10

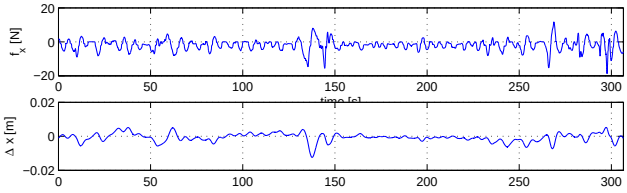
As for the gains in Section 5, we set $k_{v,p} = 10^{-8}$, $k_{v,d} = 3 \cdot 10^{-9}$ and $k_{v,i} = 2 \cdot 10^{-9}$ to track the car desired velocity v^*, whereas in the steering wheel controller we choose the gain $k_p = 3$, and we set the parameter $k_\alpha = -5$. While the controller gains have been chosen as a tradeoff between reactivity and control effort, the parameter k_α was roughly estimated. Given the considered scenario, an exact knowledge of this parameter is generally not possible, since it depends on the car characteristics. It is however possible to show that, at the kinematic level, this kind of parameter uncertainty will induce a non-persistent perturbation on the nominal closed loop dynamics.

Proving the boundedness of the perturbation term induced by parameter uncertainties would allow to conclude about the local asymptotic stability of the perturbed system. In general, this would imply a bound on the parameter uncertainty, to be satisfied to preserve local stability. While this analysis is beyond the scope of this paper, we note also that in practice it is not possible to limit the parameter uncertainty, that depends on the car and the environment characteristics. Therefore, we rely on the experimental verification of the vision-based controller robustness, delegating to the autonomous-assisted-teleoperated framework the task of taking the autonomous mode controller within its region of local asymptotic stability. In other words, when the system is too far from the equilibrium condition, and convergence of the vision-based controller could be compromised, due to model uncertainties and unexpected perturbations, the user can always resort to the other driving modes.

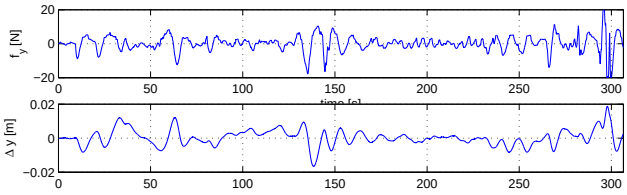
In the KF used for the car velocity estimation, the process and the measurement noise covariances matrices are set to $\text{diag}(1\text{e}^{-4}, 1\text{e}^{-4})$ and $\text{diag}(1\text{e}^{2}, 1\text{e}^{2})$, respectively. Since the forward axis of the robot frame is aligned with the forward axis of the vehicle frame, to get a_{IMU} we didn't apply the transformation (16), but we simply collected the acceleration along the forward axis of the robot frame, as given by the accelerometers. The sampling time of the KF was set to $\Delta T = 0.002$ s (being 500 Hz the frequency of the IMU measurements, the



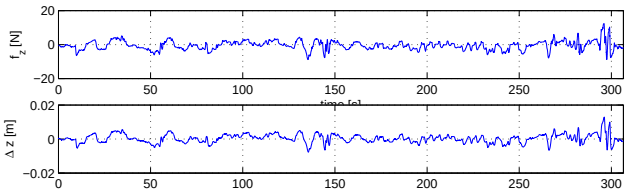
(a) Human-like grasp.



(b) Admittance along x-axis



(c) Admittance along y-axis



(d) Admittance along z-axis

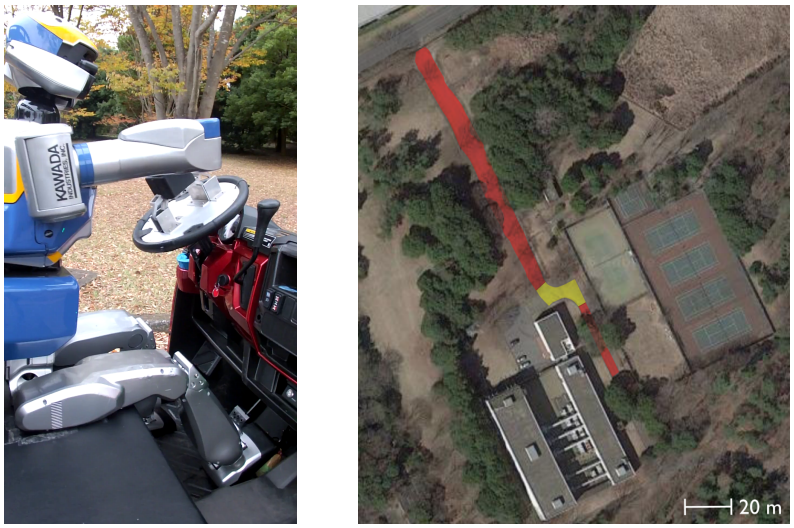
Figure 8: Left: setup of an experiment that requires admittance control on the steering hand. Right: output of the admittance controller in the hand frame during the same experiment.

filter runs at the same rate).

The cut-off frequencies of the low-pass filters applied to the visual features and the car velocity estimate were set to 8 and 2.5 Hz, respectively.

At the beginning of each campaign of experiments, we arrange the robot in the correct driving posture in the car as shown in Fig. 9a. This posture (except for the driving leg and arm) is assumed constant during driving: all control parameters are kept constant. At initialization, we also correct eventual bad orientations of the camera with respect to the ground plane, by applying a rotation to the acquired image, and by regulating the pitch and yaw angles of the robot neck, so as to align the focal axis with the forward axis of the car reference frame. The right foot is positioned on the gas pedal, and the calibration procedure described in Sect. 6.2 is used to obtain $q_{a,\max}$ and $q_{a,\min}$.

To ease full and stable grasping of the steering wheel, we designed a handle, fixed to the wheel (visible in Fig. 9a), allowing the alignment of the wrist axis with that of the steer. With reference to Fig. 3, this corresponds to configuring the hand grasp with $r = 0$ and, to comply with the shape of the steering wheel, $\beta = 0.57$ rad. Due to the robot kinematic constraints, such as joint limits and auto-collisions avoidance, imposed by our driving configuration, the range of the steering angle α is restricted from approximately -2 rad to 3 rad. These limits cause bounds on the maximum curvature realizable by the car. Nevertheless, all of the



(a) Driving posture (b) Top view of the experimental area

Figure 9: The posture taken by HRP-2Kai during the experiments (a) and the experimental area at the AIST campus (b).

followed paths were compatible with this constraint. For more challenging maneuvers, grasp reconfiguration should be integrated in the framework.

With this grasping setup, we achieved a good alignment between the robot hand and the steering wheel. Hence, during driving, the robot did not violate the geometrical constraints imposed by the steering wheel mechanism. In this case, the use of the admittance control for safe manipulation is not necessary. However, we showed in (Paolillo et al., 2014), that the admittance control can be easily plugged in our framework, whenever needed. In fact, in that work, an HRP-4, from Kawada Industries, turns the steering wheel with a more 'human-like' grasp ($r = 0.2$ m and $\beta = 1.05$ rad, see Fig. 8a). Due to the characteristics of both the grasp and the HRP-4 hand, admittance control is necessary. For sake of completeness, we report, in Fig. 8b-8d, plots of the admittance behavior relative to that experiment. In particular, to have good tracking of the steering angle α, while complying with the steering wheel geometric constraint, we designed a fast (stiff) behavior along the z-axis of the hand frame, $\mathcal{F}_h$, and a slow (compliant) along the x and y-axes. To this end, we set the admittance parameters: $m_x = m_y = 2000$ kg, $m_z = 10$ kg, $b_x = b_y = 1600$ kg/s, $b_z = 240$ kg/s, and $k_x = k_y = 20$ kg/s^2, $k_z = 1000$ kg/s^2. Furthermore, we set the desired forces $f_x^* = f_z^* = 0$ N, while along the y-axis of the hand frame $f_y^* = 5$ N, to improve the grasping stability. Note that the evolution of the displacements along the x and y-axes (plots in Fig. 8b-8c), are the results of a dynamic behavior that filters the high frequency of the input forces, while along the z-axis the response of the system is more reactive.

In the rest of this section, we present the HRP-2Kai outdoor driving experiments. In particular, we present the results of the experiments performed at the authorized portion of the AIST campus in Tsukuba, Japan. A top view of this experimental field is shown in Fig. 9b. The areas highlighted in red and yellow correspond to the paths driven using

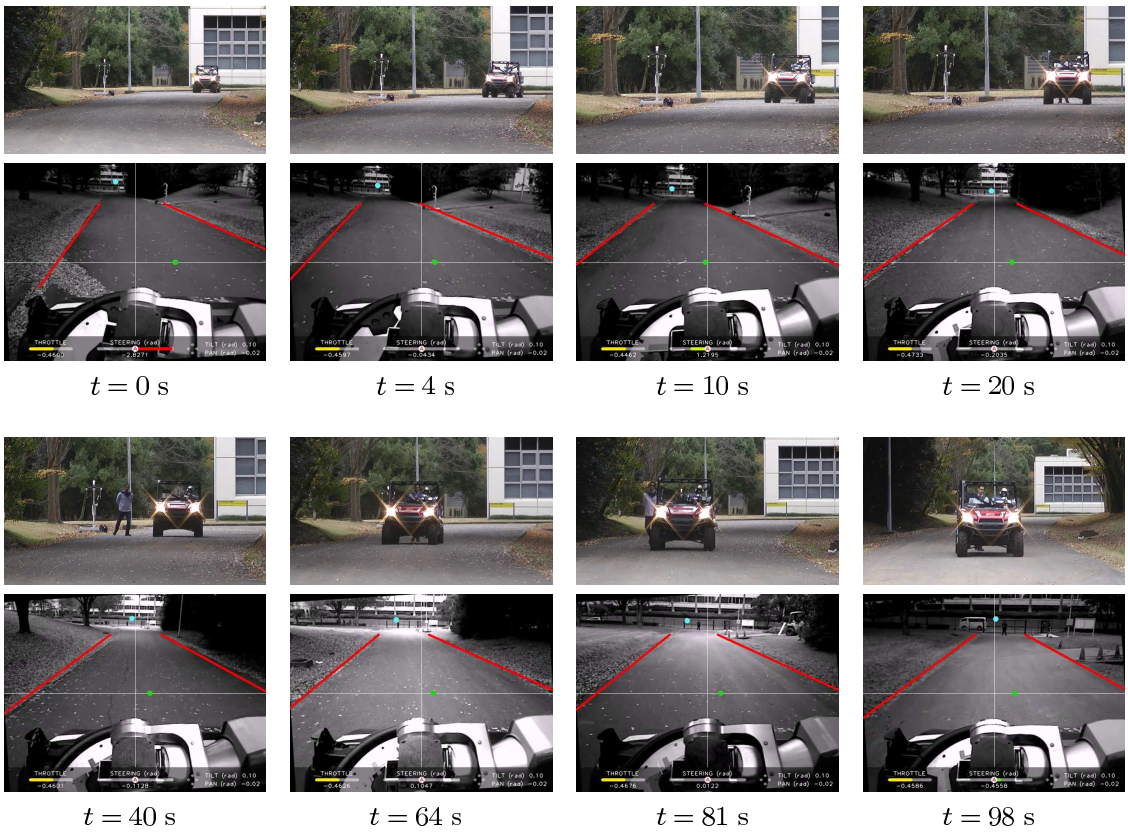


Figure 10: First experiment: autonomous car driving.

the autonomous and teleoperated mode, respectively, as further described below. Furthermore, we present an experiment performed at the DRC final, showing the effectiveness of the assisted driving mode. For a quantitative evaluation of the approach, we present the plots of the variables of interest. The same experiments are shown in the video available at `https://youtu.be/SYHI2JmJ-lk`, that also allows a qualitative evaluation of the online image processing. Quantitatively, we successfully carried out 14 experiments over of 15 repetitions, executed at different times, between 10:30 a.m. and 4 p.m., proving image processing robustness in different light conditions.

7.1 First experiment: autonomous car driving

In the first experiment, we tested the autonomous mode, i.e., the effectiveness of our framework to make a humanoid robot drive a car autonomously. For this experiment, we choose $v^* = 1.2$ m/s, while the foot calibration procedure gave $q_{a,\max} = -0.44$ rad and $q_{a,\min} = -0.5$ rad.

Figure 10 shows eight snapshots taken from the video of the experiment. The car starts with an initial lateral offset, that is corrected after a few meters. The snapshots (as well as the video) of the experiment show that the car correctly travels at the center of a curved path, for about 100 m. Furthermore, one can observe that the differences in the light conditions

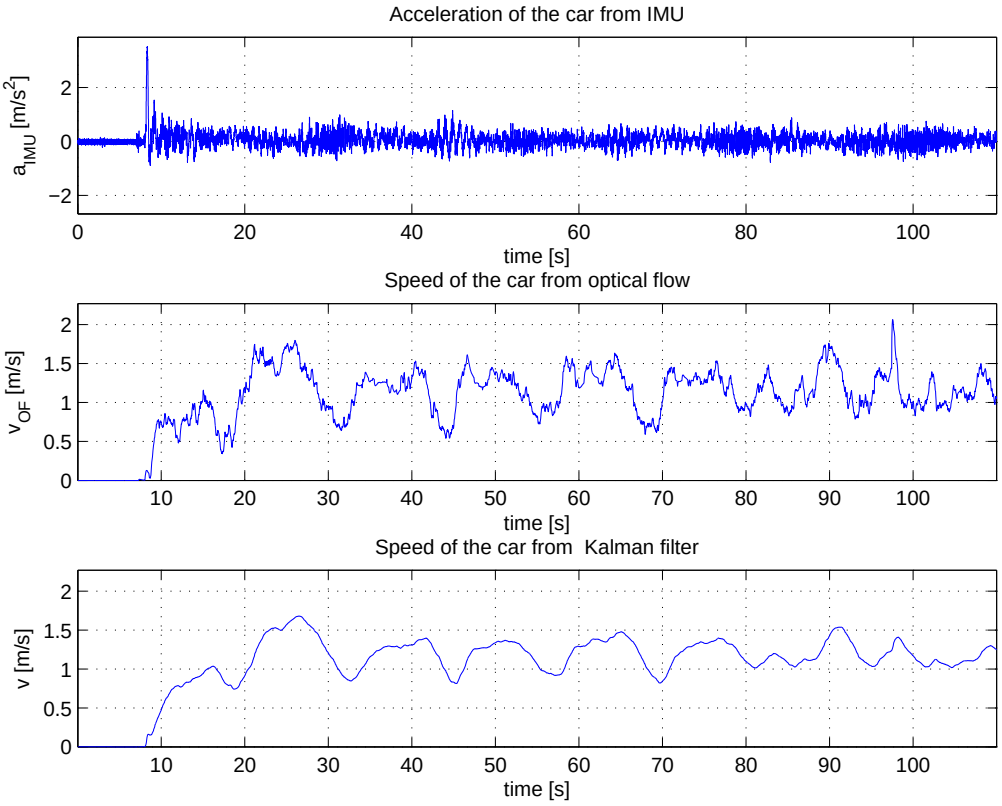
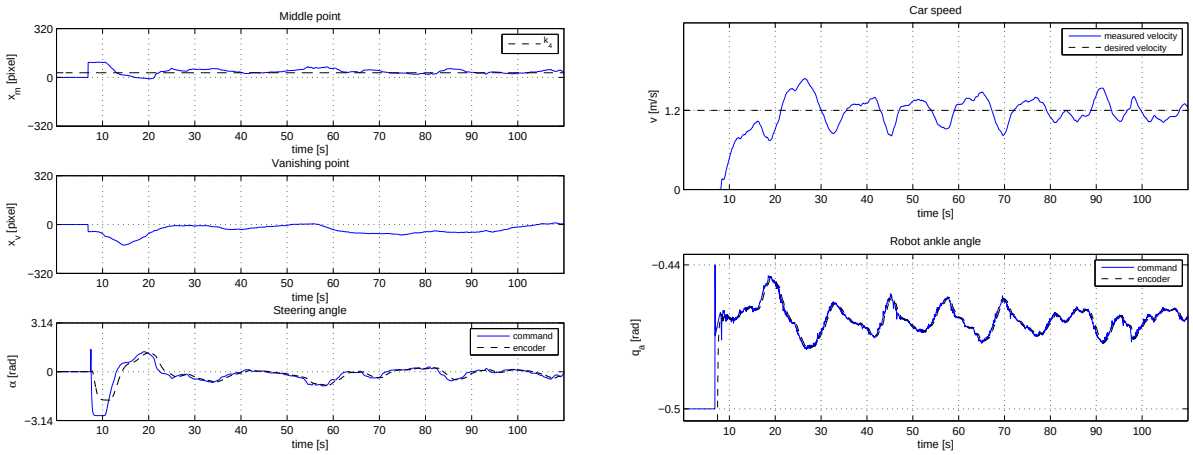


Figure 11: First experiment: autonomous car driving. Acceleration a_{IMU} measured with the robot IMU (top), linear velocity v_{OF} measured with the optical flow (center), and car speed v estimated by the KF (bottom).

(due to the tree shadows) and in the color of the road, do not jeopardize the correct detection of the borders and, consequently, the driving performance.

Figure 11 shows the plots related to the estimation of the car speed, as described in Sect. 4.2. On the top, we plot a_{IMU}, the acceleration along the forward axis of the car, as reconstructed from the robot accelerometers. The center plot shows the car speed measured with the optical flow-based method (v_{OF}), whereas the bottom plot gives the trace of the car speed v obtained by fusing a_{IMU} and v_{OF}. Note that the KF reduces the noise of the v_{OF} signal, a very important feature for keeping the derivative action in the velocity control law (22).

As well known, reconstruction from vision (e.g., the "structure from motion" problem) suffers from a scale problem, in the translation vector estimate (Ma et al., 2003). This issue, due to the loss of information in mapping 2D to 3D data, is also present in optical flow velocity estimation methods. Here, this can lead to a scaled estimate of the car velocity. For this reason, we decided to include another sensor information in the estimation process: the acceleration provided by the IMU. Note, however, that in the current state of the work, the velocity estimate accuracy has been only evaluated qualitatively. In fact, that high accuracy is only important in the transient phases (initial error recovery and curve negotiation). Instead, it can be easily shown that the perturbation induced by velocity estimate inaccuracy



(a) (b)

Figure 12: First experiment: autonomous car driving. (a) Middle point abscissa x_m (top), vanishing point abscissa x_v (center), and steering angle α (bottom). (b) Car speed v (top), already shown in fig 10, and ankle joint angle q_a (bottom).

on the features dynamics vanishes at the regulation point corresponding to the desired driving task, and that by limiting the uncertainty on the velocity value, it is possible to preserve local stability. In fact, the driving performance showed that the estimation was accurate enough, for the considered scenario. In different conditions, finer tuning of the velocity estimator may be necessary.

Plots related to the steering wheel control are shown in Fig. 12a. The steering control is activated about 8 s after the start of the experiment and, after a transient time of a few seconds, it leads the car to the road center. Thus, the middle and vanishing points (the top and center plots, respectively) correctly converge to the desired values, i.e., x_m goes to $k_4 = 30$ pixels (since $\gamma = 0.2145$ rad –see expression of k_4 in Sect. 5.1), and x_v to 0. The bottom plot shows the trend of the desired steering command α, as computed from the visual features, and from the estimated car speed according to (21). The same signal, reconstructed from the encoders (black dashed line) shows that the steering command is smoothed by the task-based quadratic programming control, avoiding undesirable fast signal variations.

Fig. 12b presents the plots of the estimated vs desired car speed (top) and the ankle angle command sent to the robot to operate the gas pedal and drive the car at the desired velocity (bottom).

Also in this case, after the initial transient, the car speed converges to the nominal desired values (no ground truth was available). The oscillations observable at steady state are due to the fact that the resolution of the ankle joint is coarser than that of the gas pedal. Note, in fact, that even if the robot ankle moves in a small range, the car speed changes significantly. The noise on the ankle command, as well as the initial peak, are due to the derivative term of the gas pedal control (22). However, the signal is smoothed by the task-based quadratic programming control (see the dashed black line, i.e., the signal reconstructed by encoder

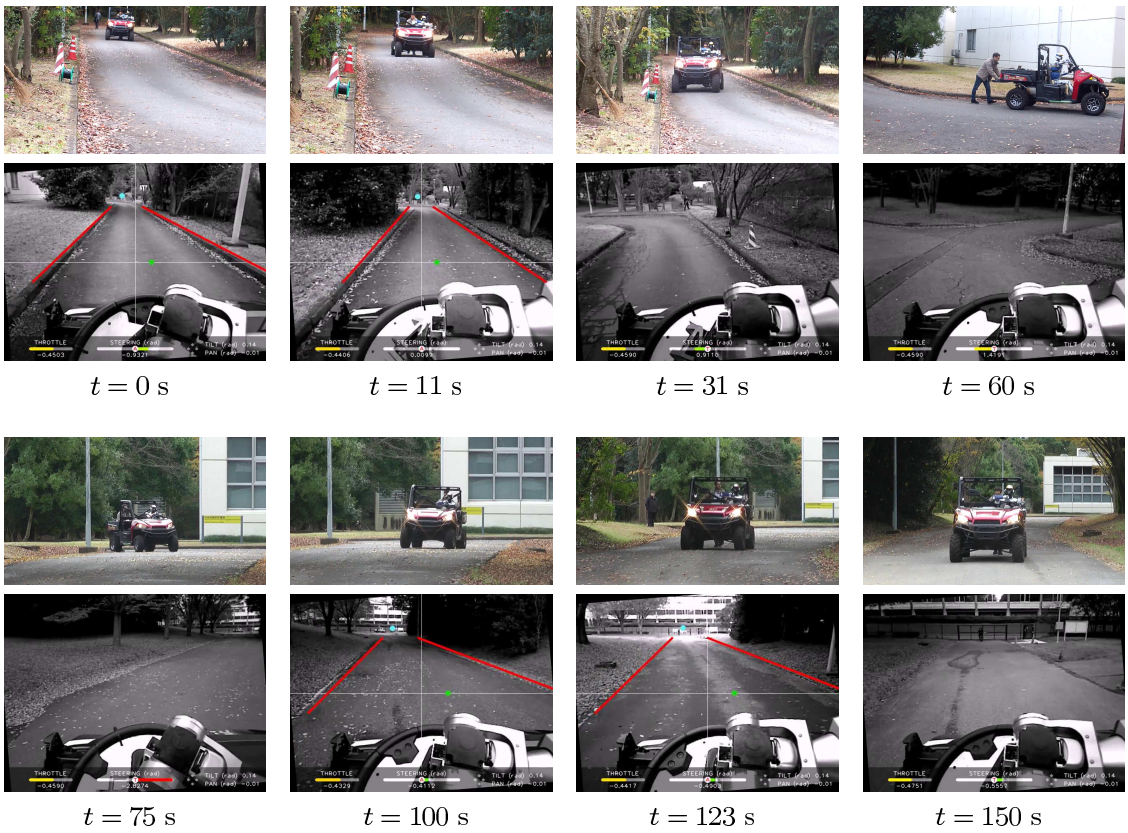


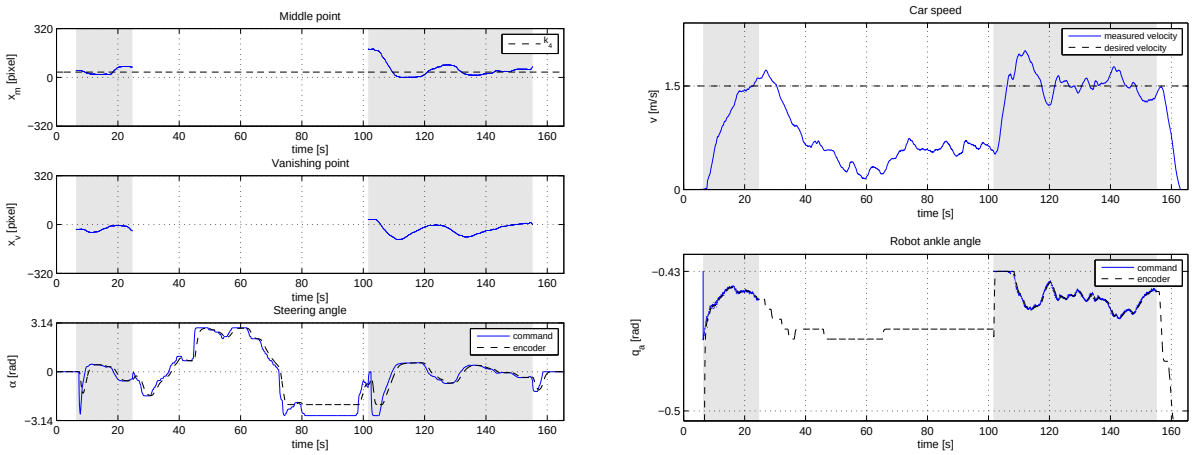
Figure 13: Second experiment: switching between teleoperated and autonomous modes.

readings), preventing jerky motion of the robot foot.

In the same campaign of experiments, we performed ten autonomous car driving experiments. In nine of them (including the one presented just above), the robot successfully drove the car for the entire path. One of the experiments failed due to a critical failure of the image processing. It was not possible to perform experiments on other tracks (with different road shapes and environmental conditions), because our application was rejected after complex administrative paperwork required to access other roads in the campus.

7.2 Second experiment: switching between teleoperated and autonomous modes

In some cases, the conditions ensuring the correct behaviour of the autonomous mode are risky. Thus, it is important to allow a user to supervise the driving operation, and control the car if required. As described in Sect. 2, our framework allows a human user to intervene at any time, during the driving operation, to select a particular driving strategy. The second experiment shows the switching between the autonomous and teleoperated modes. In particular, in some phases of the experiment, the human takes control of the robot, by selecting the teleoperated mode. In these phases, proper commands are sent to the robot, to drive the car along two very sharp curves, connecting two straight roads traveled in



(a) (b)

Figure 14: Second experiment: switching between teleoperated and autonomous modes. (a) Middle point abscissa x_m (top), vanishing point abscissa x_v (center), and steering angle α (bottom). (b) Car speed v (top) and ankle joint angle q_a (bottom).

autonomous mode. Snapshots of this second experiment are shown in Fig. 13.

For this experiment we set $v^* = 1.5$ m/s, while after the initial calibration of the gas pedal, $q_{a,\min} = -0.5$ rad and $q_{a,\max} = -0.43$ rad. Note that the difference in the admissible ankle range with respect to the previous experiment is due to a slightly different position of the robot foot on the gas pedal.

Figure 14a shows the signals of interest for the steering control. In particular, one can observe that when the control is enabled (shadowed areas of the plots) there is the same correct behavior of the system seen in the first experiment. When the user asks for the teleoperated mode (non-shadowed areas of the plots), the visual features are not considered, and the steering command is sent to the robot via keyboard or joystick by the user. Between 75 and 100 s, the user controlled the robot (in teleoperated mode) to make it steer on the right as much as possible. Because of the kinematic limits and of the grasping configuration, the robot saturated the steering angle at about -2 rad even if the user asked a wider steering. This is evident on the plot of the steering angle command of Fig. 14a (bottom): note the difference between the command (blue continuous curve), and the steering angle reconstructed from the encoders (black dashed curve).

Similarly, Fig. 14b shows the gas pedal control behavior when switching between the two modes. When the gas pedal control is enabled, the desired car speed is properly tracked by operating the robot ankle joint (shadowed areas of the top plot in Fig. 14b). On the other hand, when the control is disabled (non-shadowed areas of the plots), the ankle command (blue curve in Fig. 14b, bottom), as computed by (25), is not considered, and the robot ankle is teleoperated with the keyboard/joystick interface, as noticeable from the encoder plot (black dashed curve).

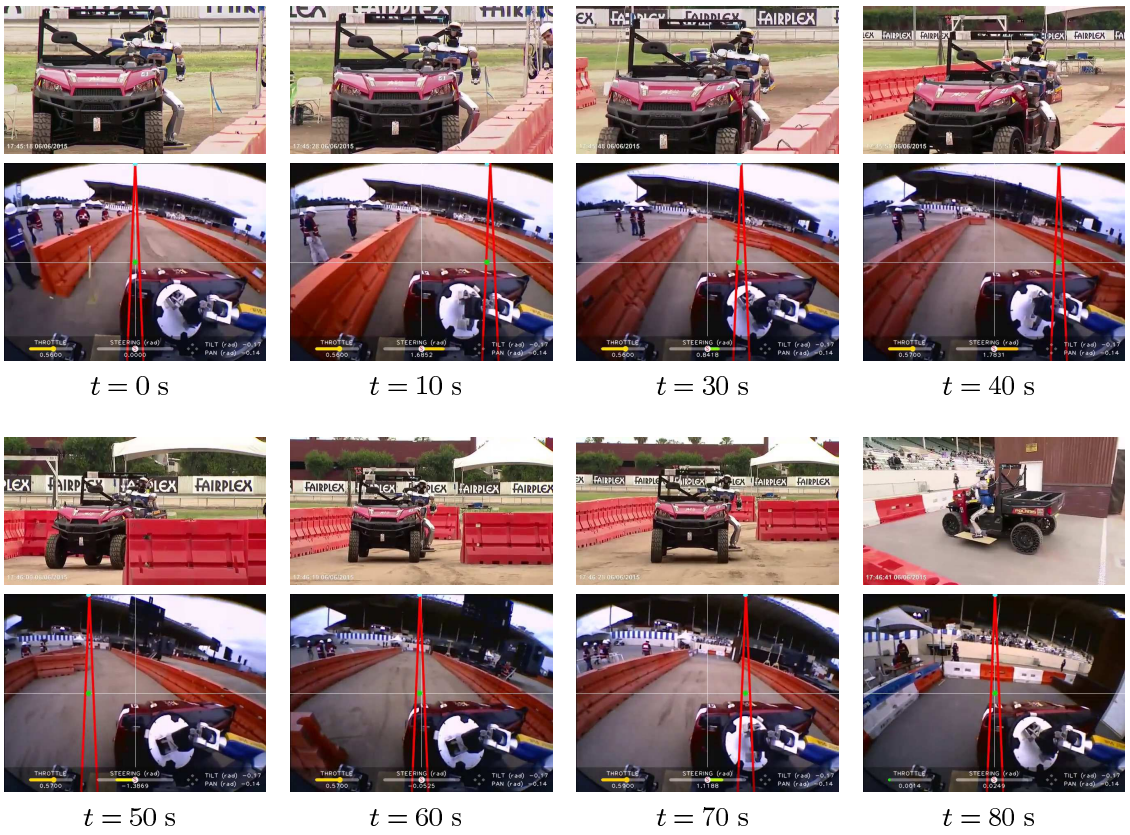


Figure 15: Third experiment: assisted driving mode at the DRC finals. Snapshots taken from the DRC official video.

At the switching between the two modes, the control keeps sending commands to the robot without any interruption, and the smoothness of the signals allows to have continuous robot operation. In summary, the robot could perform the entire experiment (along a path of 130 m ca., for more than 160 s) without the need to stop the car. This was achieved thanks to two main design choices. Firstly, from a perception viewpoint, monocular camera and IMU data are light to be processed, allowing a fast and reactive behavior. Secondly, the control framework at all the stages (from the higher level visual control to the low level kinematic control) guarantees smooth signals, even at the switching moments.

The same experiment presented just above was performed five other times, during the same day. Four experiments resulted successful, while two failed do to human errors during tele-operation.

7.3 Third experiment: assisted driving at the DRC finals

The third experiment shows the effectiveness of the assisted driving mode. This strategy was used to make the robot drive at the DRC finals, where the first of the eight tasks consisted in driving a utility vehicle along a straight path, with two sets of obstacles. We successfully completed the task, by using the assisted mode. Snapshots taken from the DRC

finals official video (DARPAtv, 2015) are shown in Fig. 15. The human user teleoperated HRP-2Kai remotely, by using the video stream from the robot camera as the only feedback from the challenge field. In the received images, the user selected, via mouse, the proper artificial road borders (red lines in the figure), to steer the car along the path. Note that these artificial road borders, manually set by the user, may not correspond to the real borders of the road. In fact, they just represent geometrical references - more intuitive for humans - to easily define the vanishing and middle points and steer the car by using (21). Concurrently, the robot ankle was teleoperated to achieve a desired car velocity. In other words, with reference to the block diagram of Fig. 1, the user provides the visual features to the steering control, and the gas pedal reference to the pedal operation block. Basically, s/he takes the place of the road detection and car velocity estimation/control blocks. The assisted mode could be seen as a sort of shared control between the robot and the a human supervisor, and allows the human to interfere with the robot operation if required. As stated in the previous section, at any time, during the execution of the driving experience, the user can instantly and smoothly switch to one of the other two driving modes. At the DRC, we used a wide angle camera, although the effectiveness of the assisted mode was also verified with a Xtion camera.

8 Conclusions

In this paper, we have proposed a reactive control architecture for car driving by a humanoid robot on unkown roads. The proposed approach consists in extracting road visual features, to determine a reference steering angle to keep the car at the center of a road. The gas pedal, operated by the robot foot, is controlled by estimating the car speed using visual and inertial data. Three different driving modes (autonomous, assisted, and teleoperated) extend the versatility of our framework. The experimental results carried out with the humanoid robot HRP-2Kai have shown the effectiveness of the proposed approach. The assisted mode was successfully used to complete the driving task at the DRC finals.

The driving task has addressed, as an explicative case-study of humanoids controlling human-tailored devices. In fact, besides the achievement of the driving experience, we believe that humanoids are the most sensible platforms for helping humans with everyday task, and the proposed work shows that complex real-world tasks can be actually performed in autonomous, assisted and teleoperated way. Obviously, the complexity of the task comes also with the complexity of the framework design, on both perception and control point-of-views. This led us to make some working assumptions that, in some cases, limited the range of application of our methods.

Further investigations shall deal with the task complexity, to advance the state-of-art of algorithms, and make humanoids capable of helping humans with dirty, dangerous and demanding jobs. Future work will be done, in order to make the autonomous mode work efficiently in the presence of sharp curves. To this end, and to overcome the problem of limited steering motions, we plan to include, in the framework, the planning of variable grasping configurations, to achieve more complex manoeuvres. We are also planning to go to driving

on uneven terrains, where the robot has also to sustain its attitude, w.r.t. sharp changes of the car orientation. Furthermore, the introduction of obstacle avoidance algorithms, based on optical flow, will improve the driving safety. Finally, we plan to add brake control and to perform the entire driving task, including car ingress and egress.

Acknowledgments

This work is supported by the EU FP7 strep project KOROIBOT `www.koroibot.eu`, and by the Japan Society for Promotion of Science (JSPS) Grant-in-Aid for Scientific Research (B) 25280096. This work was also in part supported by the CNRS PICS Project ViNCI. The authors deeply thank Dr. Eiichi Yoshida for taking in charge the administrative procedures in terms of AIST clearance and transportation logistics, without which the experiments could not be conducted; Dr Fumio Kanehiro for lending the car and promoting this research; Hervé Audren and Arnaud Tanguy for their kind support during the experiments.

References

Atkeson, C. G., Babu, B. P. W., Banerjee, N., Berenson, D., Bove, C. P., Cui, X. DeDonato, M., Du, R., Feng, S., Franklin, P., Gennert, M., Graff, J. P., He, P., Jaeger, A., Kim, J., Knoedler, K., Li, L., Liu, C., Long, X., Padir, T., Polido, F., Tighe, G. G., and Xinjilefu, X. (2015). NO FALLS, NO RESETS: Reliable humanoid behavior in the DARPA Robotics Challenge. In *2015 15th IEEE-RAS Int. Conf. on Humanoid Robots*, pages 623–630.

Barbosa, R., Silva, J., Junior, M. M., and Gallis, R. (2007). Velocity estimation of a mobile mapping vehicle using filtered monocular optical flow. In *International Symposium on Mobile Mapping Technology*, volume 5.

Bradski, G. (2000). The OpenCV library. *Dr. Dobb's Journal of Software Tools*, 25(11):120–126.

Broggi, A., Bombini, L., Cattani, S., Cerri, P., and Fedriga, R. I. (2010). Sensing requirements for a 13,000 km intercontinental autonomous drive. In *2010 IEEE Intelligent Vehicles Symposium (IV)*, pages 500–505.

Buehler, M., Lagnemma, K., and Singh, S. E. (2008). Special issue on the 2007 DARPA Urban Challenge, part I-III. *Journal of Field Robotics*, 25(8-10):423–860.

Chaumette, F. and Hutchinson, S. (2006). Visual servo control, Part I: Basic approaches. *IEEE Robot. Automat. Mag.*, 13(4):82–90.

Cherubini, A., Spindler, F., and Chaumette, F. (2014). Autonomous visual navigation and laser-based moving obstacle avoidance. *IEEE Trans. on Intelligent Transportation Systems*, 31(1–2):2101–2110.

Dahlkamp, H., Kaehler, A., Stavens, D., Thrun, S., and Bradski, G. R. (2006). Self-supervised monocular road detection in desert terrain. In *Robotics: Science and Systems*.

DARPA Robotics Challenge (2015). DRC Finals. http://www.theroboticschallenge.org/.

DARPAtv (2015). Team AIST-NEDO driving on the second day of the DRC finals. Retrieved from https://youtu.be/s6ZdC_ZJXK8.

De Luca, A. and Oriolo, G. (1995). *Kinematics and Dynamics of Multi-Body Systems, CISM Courses and Lectures*, volume 360, chapter Modelling and control of nonholonomic mechanical systems, pages 277–3429. Springer Verlag, Wien.

DRC-Teams (2015). What happened at the DARPA Robotics Challenge? www.cs.cmu.edu/ cga/drc/events.

Farnebäck, G. (2003). *Image Analysis: 13th Scandinavian Conference, SCIA 2003 Halmstad, Sweden*, volume 2749 of *Lecture Notes in Computer Science*, chapter Two-Frame Motion Estimation Based on Polynomial Expansion, pages 363–370. Springer Berlin Heidelberg.

Giachetti, A., Campani, M., and Torre, V. (1998). The use of optical flow for road navigation. *IEEE Trans. on Robotics and Automation*, 14(1):34–48.

Google (2015). Google self-driving car project. https://www.google.com/selfdrivingcar.

Grabe, V., Bulthoff, H., and Giordano, P. (2012). On-board velocity estimation and closed-loop control of a quadrotor UAV based on optical flow. In *IEEE Int. Conf. on Robotics and Automation*, pages 491–497.

Hasunuma, H., Nakashima, K., Kobayashi, M., Mifune, F., Yanagihara, Y., Ueno, T., Ohya, K., and Yokoi, K. (2003). A tele-operated humanoid robot drives a backhoe. In *IEEE Int. Conf. on Robotics and Automation*, pages 2998–3004.

Hentschel, M. and Wagner, B. (2010). Autonomous robot navigation based on open street map geodata. In *13th Int. IEEE Conf. on Intelligent Transportation Systems*, pages 1645–1650.

Hirata, N., Mizutani, N., Matsui, H., Yano, K., and Takahashi, T. (2015). Fuel consumption in a driving test cycle by robotic driver considering system dynamics. In *2015 IEEE Int. Conf. on Robotics and Automation*, pages 3374–3379.

Hogan, N. (1985). Impedance control - An approach to manipulation. I - Theory. II - Implementation. III - Applications. *Journal of Dynamic Systems and Measurement Control B*, 107:1–24.

Jeong, H., Oh, J., Kim, M., Joo, K., Kweon, I. S., and Oh, J.-H. (2015). Control strategies for a humanoid robot to drive and then egress a utility vehicle for remote approach. In *2015 15th IEEE-RAS Int. Conf. on Humanoid Robots*, pages 811–816.

Kim, S., Kim, M., Lee, J., Hwang, S., Chae, J., Park, B., Cho, H., Sim, J., Jung, J., Lee, H., Shin, S., Kim, M., Kwak, N., Lee, Y., Lee, S., Lee, M., Yi, S., Chang, K.-S. K., and Park, J. (2015). Approach of team SNU to the DARPA Robotics Challenge finals. In *2015 15th IEEE-RAS Int. Conf. on Humanoid Robots*, pages 777–784.

Kumagai, I., Terasawa, R., Noda, S., Ueda, R., Nozawa, S., Kakiuchi, Y., Okada, K., and Inaba, M. (2015). Achievement of recognition guided teleoperation driving system for humanoid robots with vehicle path estimation. In *2015 15th IEEE-RAS Int. Conf. on Humanoid Robots*, pages 670–675.

Laganière, R. (2011). *OpenCV 2 Computer Vision Application Programming Cookbook: Over 50 recipes to master this library of programming functions for real-time computer vision.* Packt Publishing Ltd.

Lim, K. H., Seng, K. P., Ngo, A. C. L., and Ang, L.-M. (2009). Real-time implementation of vision-based lane detection and tracking. In *Int. Conf. on Intelligent Human-Machine Systems and Cybernetics, 2009*, volume 2, pages 364–367.

Liu, W., Zhang, H., Duan, B., Yuan, H., and Zhao, H. (2008). Vision-based real-time lane marking detection and tracking. In *11th Int. IEEE Conf. on Intelligent Transportation Systems, 2008*, pages 49–54.

Ma, B., Lakshmanan, S., and Hero, A. O. (2000). Simultaneous detection of lane and pavement boundaries using model-based multisensor fusion. *IEEE Transactions on Intelligent Transportation Systems*, 1(3):135–147.

Ma, Y., Soatto, S., Kosecka, J., and Sastry, S. S. (2003). *An Invitation to 3-D Vision: From Images to Geometric Models.* SpringerVerlag.

McGill, S., Yi, S.-J., and Lee, D. D. (2015). Team THOR's adaptive autonomy for disaster response humanoids. In *2015 15th IEEE-RAS Int. Conf. on Humanoid Robots*, pages 453–460.

Meuter, M., Muller-Schneiders, S., Mika, A., Hold, S., Nunny, C., and Kummert, A. (2009). A novel approach to lane detection and tracking. In *12th Int. IEEE Conf. on Intelligent Transportation Systems, 2009*, pages 1–6.

Mohellebi, H., Kheddar, A., and Espié, S. (2009). Adaptive haptic feedback steering wheel for driving simulators. *IEEE Transactions on Vehicular Technology*, 58(4):1654–1666.

Newman, P., Sibley, G., Smith, M., Cummins, M., Harrison, A., Mei, C., Posner, I., Shade, R., Schroeter, D., Murphy, L., Churchill, W., Cole, D., and Reid, I. (2009). Navigating, recognizing and describing urban spaces with vision and lasers. *Int. J. of Robotics Research*, 28(11-12):1406–1433.

Nieto, M., Cortés, A., Otaegui, O., Arróspide, J., and Salgado, L. (2012). Real-time lane tracking using Rao-Blackwellized particle filter. *Journal of Real-Time Image Processing*, pages 1–13.

Nunes, U., Laugier, C., and Trivedi, M. M. (2009). Guest editorial introducing perception, planning, and navigation for intelligent vehicles. *IEEE Trans. on Intelligent Transportation Systems*, 10(3):375–379.

Oriolo, G., Paolillo, A., Rosa, L., and Vendittelli, M. (2015). Humanoid odometric localization integrating kinematic, inertial and visual information. *Autonomous Robots*, pages 1–13. DOI:10.1007/s10514-015-9498-0.

Paolillo, A., Cherubini, A., Keith, F., Kheddar, A., and Vendittelli, M. (2014). Toward autonomous car driving by a humanoid robot: A sensor-based framework. In *2014 14th IEEE-RAS Int. Conf. on Humanoid Robots*, pages 451–456.

Paolillo, A., Faragasso, A., Oriolo, G., and Vendittelli, M. (2016). Vision-based maze navigation for humanoid robots. *Autonomous Robots*, pages 1–17. DOI:10.1007/s10514-015-9533-1.

Pratt, G. and Manzo, J. (2013). The DARPA Robotics Challenges. *IEEE Robotics Automation Magazine*, 20(2):10–12.

Rasmussen, C., Sohn, K., Wang, Q., and Oh, P. (2014). Perception and control strategies for driving utility vehicles with a humanoid robot. In *2014 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 973–980.

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., Lau, K., Oakley, C., Palatucci, M., Pratt, V., Stang, P., Strohband, S., Dupont, C., Jendrossek, L.-E., Koelen, C., Markey, C., Rummel, C., van Niekerk, J., Jensen, E., Alessandrini, P., Bradski, G., Davies, B., Ettinger, S., Kaehler, A., Nefian, A., and Mahoney, P. (2006). Stanley: The robot that won the DARPA Grand Challenge. *Journal of Field Robotics*, 23(9):661–692.

Toibero, J., Soria, C., Roberti, F., Carelli, R., and Fiorini, P. (2009). Switching visual servoing approach for stable corridor navigation. In *IEEE Int. Conf. on Advanced Robotics*, pages 1–6.

Vacek, S., Schimmel, C., and Dillmann, R. (2007). Road-marking analysis for autonomous vehicle guidance. In *3rd European Conference on Mobile Robots*.

Vaillant, J., Kheddar, A., Audren, H., Keith, F., Brossette, S., Escande, A., Bouyarmane, K., Kaneko, K., Morisawa, M., Gergondet, P., Yoshida, E., Kajita, S., and Kanehiro, F. (2016). Multi-contact vertical ladder climbing with an HRP-2 humanoid. *Autonomous Robots*, pages 1–20.

Vassallo, R. F., Schneebeli, H. J., and Santos-Victor, J. (2000). Visual servoing and appearance for navigation. *Robotics and Autonomous Systems*, 31(1–2):87–97.

Yokoi, K., Nakashima, K., Kobayashi, M., Mihune, H., Hasunuma, H., Yanagihara, Y., Ueno, T., Gokyuu, T., and Endou, K. (2006). A tele-operated humanoid operator. *Int. J. of Robotics Research*, 5-6:593–602.

Zhang, Y., Lin, W., and Chin, Y.-K. (2008). Driving skill characterization: A feasibility study. In *IEEE Int. Conf. on Robotics and Automation, 2008*, pages 2571–2576.