Visual Navigation With Obstacle Avoidance

Andrea Cherubini, F. Chaumette

Visual Navigation With Obstacle Avoidance

Andrea Cherubini and François Chaumette

***Abstract*— We present and validate a framework for visual navigation with obstacle avoidance. The approach was originally designed in [1], but major improvements and real outdoor experiments are added here. Visual navigation consists of following a path, represented as an ordered set of key images, that have been acquired in a preliminary teaching phase. While following such path, the robot is able to avoid new obstacles which were not present during teaching, and which are sensed by a range scanner. We guarantee that collision avoidance and navigation are achieved simultaneously by actuating the camera pan angle, in the presence of obstacles, to maintain scene visibility as the robot circumnavigates the obstacle. The circumnavigation verse and the collision risk are estimated using a potential vector field derived from an occupancy grid. The framework can also deal with unavoidable obstacles, which make the robot decelerate and eventually stop.**

***Index Terms*— Visual Navigation, Visual Servoing, Collision Avoidance.**

I. INTRODUCTION

A great amount of robotics research focuses on vehicle guidance, with the goal of automatically reproducing the tasks usually performed by humans [2], [3]. Among others, an important task is obstacle avoidance, i.e., computing a control such that the trajectory generated is collision-free, and drives the robot to the goal [4]. A common obstacle avoidance technique is the potential field method [5], which is often associated to a global path planner. Instead of using a global model, we propose a framework for obstacle avoidance with simultaneous execution of a *visual servoing* task [6]. Visual servoing is a well known method that uses vision directly in the control loop, and that has been applied on mobile robots in [7 – 9]. In [7] and [8], the epipoles are exploited to drive a nonholonomic robot to a desired configuration. Trajectory tracking is tackled in [9] by merging differential flatness and predictive control.

We focus on appearance-based navigation, which in contrast with model-based navigation, operates in the sensor space, without any geometrical knowledge of the environment. In the framework that we developed in the past [10, 11], the path is represented by a database of ordered key images[1]. Navigation is divided into subtasks, each consisting of driving the robot towards the next key image in the database. To our knowledge, this scheme, which is popular in robotics [12 – 15], has never been extended to take into account obstacles. Navigation schemes with obstacle avoidance have been presented in [16 – 19]. However, all these approaches require a 3D model of the environment (e.g., of walls and doors). An exception is [20], which exploits redundancy to avoid obstacles during a visual positioning task. Here, we focus on this problem: a wheeled robot, equipped with an actuated pinhole camera and with a range scanner, must follow a visual path represented by key images, without colliding with the ground obstacles. To maintain scene visibility while avoiding the obstacles, we have decided to use an actuated camera. Although an alternative design would consist of using an omnidirectional camera, we have prefered a more conventional and widespread platform setup. The camera detects the features for navigating, while the scanner senses the obstacles in front of the robot. As in [20], we guarantee that obstacle avoidance has no effect on the visual task. In contrast with that work, however, our controller is compact, while in [20] three controllers are needed and the transitions between them are quite complex.

A. Cherubini and F. Chaumette are with INRIA Rennes - Bretagne Atlantique and IRISA, Campus de Beaulieu, Rennes, France.
E-mail: {Andrea.Cherubini, Francois.Chaumette}@inria.fr

[1]See: www.irisa.fr/lagadic/demo/demo-cycab-vis-navigation/vis-navigation.

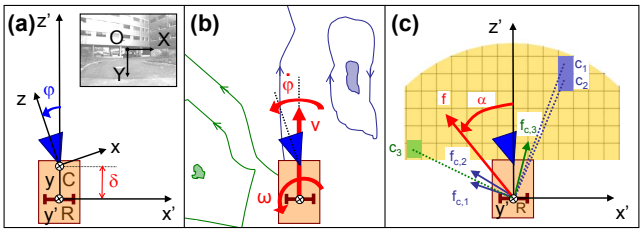

Fig. 1. Top view of the robot (orange), equipped with an actuated camera (blue). (a) Reference frames. (b) Obstacle-induced vortex fields, and control variables (v, ω, $\dot{\varphi}$). (c) Occupancy grid and potential field f construction.

Let us summarize the main contributions of our work. For the first time, obstacle avoidance and visual navigation are merged at the control level (without the need for planning), and validated in real outdoor experiments. Besides, our approach is merely appearance-based, and no model of the environment is necessary. Moreover, the desired states are globally asymptotically stable for our closed-loop system in the safe context and when the obstacles are unavoidable, guaranteeing that the robot never collides. Our framework is inspired from the work in [1]. However, many modifications have been applied. Firstly, the current framework does not require redundancy, as in [1]. Besides, the linear velocity, which was constant in [1], is varied here, to improve visual tracking and eventually stop the robot in the presence of unavoidable obstacles. Thirdly, the design of the activation function has been modified, to reduce undesired accelerations, which jeopardize convergence. Finally, the present article reports real experiments, carried out on our outdoor mobile robot.

II. MODELING AND CONTROL

A. General Definitions

The reader is referred to Fig. 1 for the definitions below. We define the robot frame $\mathcal{F}_\mathcal{R}$ (R, X', Y', Z') (R is the robot center of rotation), image frame $\mathcal{F}_\mathcal{I}(O, x, y)$ (O is the image center), and camera frame $\mathcal{F}_\mathcal{C}(C, X, Y, Z)$ (C is the optical center). The robot control inputs are:

$$\mathbf{u} = (v,\ \omega,\ \dot{\varphi})$$

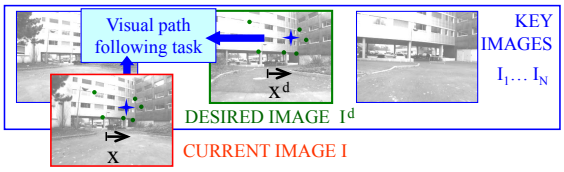

Fig. 2. The current and desired images contain some common visual features, that are used for navigation.

These are, respectively, the linear and angular velocities of the vehicle, and the camera pan angular velocity. We use the normalized perspective camera model:

$$x = \frac{X}{Z}, \qquad y = \frac{Y}{Z}$$

We assume that the camera pan angle is bounded: $|\varphi| \leq \frac{\pi}{2}$, that C belongs to the pan axis, and that the path can be tracked with continuous $v(t) > 0$. This ensures safety, since only obstacles in front of the robot can be detected by our scanner. The distance between R and C is denoted by $\delta \geq 0$.

The path that the robot must follow is represented as a database of ordered key images, such that each neighboring pair contains some common static visual features (points). First, the vehicle is manually driven along a *taught* path, with the camera pointing forward ($\varphi = 0$), and all the images are saved. Afterwards, a subset (database) of N key images $I_1, \ldots, I_N$ representing the path (Fig. 2) is selected. Then, during autonomous navigation, the current image, noted I, is compared with the next key image in the database, $I^d \in \{I_1, \ldots, I_N\}$, and a pose estimation between I and the key images is used to check when the robot passes the pose where I^d was acquired. For key image selection, as well as visual point detection and tracking, we use the algorithm presented in [12], which has proved successful in the presence of environmental changes (e.g., luminosity variation) and non-static features. The output of this algorithm, which is used by our navigation controller, is the set of points which are visible in both I and I^d. Then, navigation consists of driving the robot forward, while I is driven towards the next key image in the database. The task of maximizing similarity between I and I^d can be achieved with only one feature: the abscissa x of the centroid of the points matched on I and I^d [10]. When I^d has been passed, a topological transition is made: the next image in the set becomes the desired one, and so on, until I_N is reached.

Along with the visual path following problem, which we have tackled in [10] and [11], here we consider obstacles which are on the path, but not in the database, and sensed by the range scanner. To model these obstacles, we use an occupancy grid, yellow in Fig. 1(c). In Sect. II-C, we will explain how it is utilized in our controller.

In summary, the desired specifications, in this work, are:

1) orienting the camera to drive the feature centroid abscissa x to its value at the next key image x^d,
2) making the vehicle progress forward along the path,
3) avoiding collision with the obstacles, while remaining near the 3D taught path.

These specifications will be detailed just below.

B. Task Specifications and Control Design

Let us recall the Jacobian paradigm which relates the control inputs and the desired task. We name $\mathbf{s} \in I\!R^m$ the task vector, and $\mathbf{u} \in I\!R^m$ the inputs. These are related by:

$$\dot{\mathbf{s}} = \mathbf{J}\mathbf{u}$$

where $\mathbf{J}$ is the *task jacobian* of size $m \times m$. In this work, $m = 3$. The required task evolution can be written:

$$\dot{\mathbf{s}}^* = \dot{\mathbf{s}}^d - \Lambda\left(\mathbf{s} - \mathbf{s}^d\right)$$

with $\mathbf{s}^d$ and $\dot{\mathbf{s}}^d$ indicating the desired values of the task, and of its first derivative, and $\Lambda = \text{diag}\,(\lambda_1 \ldots \lambda_m)$ a positive definite diagonal gain matrix.

Since we assume that the visual features are static, the first specification on camera orientation can be expressed by:

$$\dot{x}^* = -\lambda_x\left(x - x^d\right) \tag{1}$$

This guarantees that the abscissa of the centroid of the points converges exponentially to its value at the next key image x^d, with null velocity there ($\dot{x}^d = 0$). The dynamics of this task can be related to the robot control inputs by:

$$\dot{x} = \mathbf{J}_x\mathbf{u} = \begin{bmatrix} j_v & j_\omega & j_{\dot{\varphi}} \end{bmatrix}\mathbf{u} \tag{2}$$

The components of $\mathbf{J}_x$ will be given in Sect. II-C.

The two other specifications (vehicle progression with collision avoidance) are related to the danger represented by the obstacles. To assess such danger, we discern two contexts (*safe* and *unsafe*), and we design the following *obstacle activation function* to smoothen the transition in between:

$$H : \mathcal{C} \mapsto [0, 1]$$

The value of H indicates the danger in a given context $\mathcal{C}$, and it will be defined in Sect. II-C.

- In the *safe context* ($H = 0$), since no obstacles are present, it is not necessary to deform the taught path. We use ω for the visual task (1). Thus, as it was during teaching, the camera pan should point forward (i.e, to $\varphi = 0$, $\dot{\varphi} = 0$). Moreover, the linear velocity v must be reduced in the presence of sharp turns, to ease the visual tracking of quickly moving features; we specify this using a function v_s, which will be given in Sect. II-C. In summary, the specifications in the safe context are:

$$\begin{cases} \dot{x} = -\lambda_x\left(x - x^d\right) \\ v = v_s \\ \dot{\varphi} = -\lambda_\varphi\varphi \end{cases} \tag{3}$$

Thus, the current and desired task dynamics are:

$$\dot{\mathbf{s}}_s = \begin{bmatrix} \dot{x} \\ v \\ \dot{\varphi} \end{bmatrix} \qquad \dot{\mathbf{s}}_s^* = \begin{bmatrix} -\lambda_x\left(x - x^d\right) \\ v_s \\ -\lambda_\varphi\varphi \end{bmatrix}$$

With (2) we can derive the Jacobian relating $\dot{\mathbf{s}}_s$ and $\mathbf{u}$:

$$\dot{\mathbf{s}}_s = \mathbf{J}_s\mathbf{u} = \begin{bmatrix} j_v & j_\omega & j_{\dot{\varphi}} \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}\mathbf{u} \tag{4}$$

Matrix $\mathbf{J}_s$ is invertible if $j_\omega \neq 0$, a condition that we can easily guarantee, as will be shown in Sect. II-C.

- In the *unsafe context* ($H > 0$), obstacles are present. If they are too near to be circumnavigated ($H = 1$), the vehicle should stop ($v = 0$). Instead, if they can be circumnavigated ($0 < H < 1$), the robot should avoid collision by orienting its heading to a desired value α

(i.e., by setting $\omega = \lambda_\alpha \alpha$). The value of α is related to the obstacles position (see Fig. 1(c)), and will be defined in Sect. II-C. Since this heading variation drives the robot away from the 3D taught path, the camera pan angle must be actuated to maintain visibility of the features, i.e., to guarantee (1). The linear velocity must be reduced for safety reasons; we specify this using a function v_u, which will be given in Sect. II-C. In summary, the specifications in the unsafe context are:

$$\begin{cases} \dot{x} = -\lambda_x \left(x - x^d\right) \\ v = v_u \\ \omega = \lambda_\alpha \alpha \end{cases} \tag{5}$$

Thus, the current and desired task dynamics are:

$$\dot{\mathbf{s}}_u = \begin{bmatrix} \dot{x} \\ v \\ \omega \end{bmatrix} \qquad \dot{\mathbf{s}}_u^* = \begin{bmatrix} -\lambda_x \left(x - x^d\right) \\ v_u \\ \lambda_\alpha \alpha \end{bmatrix} \tag{6}$$

With (2) we can derive the Jacobian relating $\dot{\mathbf{s}}_u$ and $\mathbf{u}$:

$$\dot{\mathbf{s}}_u = \mathbf{J}_u \mathbf{u} = \begin{bmatrix} j_v & j_\omega & j_{\dot{\varphi}} \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \mathbf{u} \tag{7}$$

Matrix $\mathbf{J}_u$ is invertible if $j_\varphi \neq 0$, a condition that we will prove to be always true in Sect. II-C.

To fulfill the desired tasks in the safe and unsafe contexts, we propose the following control law:

$$\mathbf{u} = H \mathbf{J}_u^{-1} \dot{\mathbf{s}}_u^* + (1 - H)\, \mathbf{J}_s^{-1} \dot{\mathbf{s}}_s^* \tag{8}$$

Replacing (8) in (4) and (7) it is trivial to see that this controller leads to convergence to the desired tasks:

$$\begin{array}{ll} \dot{\mathbf{s}}_s = \dot{\mathbf{s}}_s^* & \text{if } H = 0 \\ \dot{\mathbf{s}}_u = \dot{\mathbf{s}}_u^* & \text{if } H = 1 \end{array}$$

and that, in these cases, the desired states are globally asymptotically stable for the closed loop system. Note also that, since the first rows of $\mathbf{J}_u^{-1}$ and $\mathbf{J}_s^{-1}$ are both equal to $[0\ 1\ 0]$, the linear velocity derived using (8) is:

$$v = H v_u + (1 - H)\, v_s \tag{9}$$

This equation will be used in Sect. II-C to derive v_u and v_s.

C. System Characteristics

In this section, we will define the variables introduced above. We will show how to derive the centroid abscissa Jacobian $\mathbf{J}_x$, the linear velocity in the safe and unsafe context (v_s and v_u), and the obstacle characteristics (heading for avoidance α, and activation function H).

1) Jacobian of the Centroid Abscissa: We will hereby derive the components of $\mathbf{J}_x$ introduced in (2). Let us define: $\mathbf{v} = (v_c, \omega_c)$ the camera velocity, expressed in $\mathcal{F}_\mathcal{C}$. Since we have assumed that the features are static, the dynamics of x can be related to $\mathbf{v}$ by:

$$\dot{x} = \mathbf{L}_x \mathbf{v}$$

where $\mathbf{L}_x$ is the interaction matrix of x [6]. In the case of a point of depth Z, it is given by [6]:

$$\mathbf{L}_x = \begin{bmatrix} -\frac{1}{Z} & 0 & \frac{x}{Z} & xy & -1 - x^2 & y \end{bmatrix} \tag{10}$$

We consider the centroid as a point, which is known to be a sufficiently accurate approximation. We thus use (10) instead of the exact but more complex form given in [21]. To avoid depth estimation, which can be unreliable and time-consuming, we set the depth to a fixed value. Although this approximation requires Z to be tuned by the user, depending on the workspace characteristics, it has proved successful for most visual servoing applications [6], including nonholonomic navigation [11]. In that work, we have also shown that the uniform depth approximation proved successful even in the presence of depth tuning errors.

For our robot model, the camera velocity $\mathbf{v}$ can be expressed in function of $\mathbf{u}$ by using the homogeneous transformation:

$$\mathbf{v} = {}^C\mathbf{T}_R \mathbf{u}$$

with:

$${}^C\mathbf{T}_R = \begin{bmatrix} \sin\varphi & -\delta\cos\varphi & 0 \\ 0 & 0 & 0 \\ \cos\varphi & \delta\sin\varphi & 0 \\ 0 & 0 & 0 \\ 0 & -1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$$

Then, multiplying $\mathbf{L}_x$ by ${}^C\mathbf{T}_R$, it is trivial to obtain:

$$\begin{aligned} j_v &= \tfrac{-\sin\varphi + x\cos\varphi}{Z} \\ j_\omega &= \tfrac{\delta(\cos\varphi + x\sin\varphi)}{Z} + 1 + x^2 \\ j_{\dot{\varphi}} &= 1 + x^2. \end{aligned} \tag{11}$$

From (11) it is clear that $j_{\dot{\varphi}} \geq 1\ \forall x \in I\!R$; hence, $\mathbf{J}_u$ is never singular (see (7)). On the other hand, it is possible to ensure that $j_\omega \neq 0$, so that $\mathbf{J}_s$ is also invertible (see (4)), by setting $Z > \frac{\delta}{2}$. Indeed, condition $j_\omega \neq 0$ is equivalent to:

$$\frac{\delta\,(\cos\varphi + x\sin\varphi)}{Z} + 1 + x^2 \neq 0 \tag{12}$$

Since $|\varphi| \leq \frac{\pi}{2}$: $\cos\varphi + x\sin\varphi \geq -x$, $\forall x \in I\!R$. Hence, a sufficient condition for (12) is:

$$x^2 - \frac{\delta}{Z} x + 1 > 0$$

which occurs $\forall x \in I\!R$ when $\frac{\delta}{Z} < 2$. In practice, $\frac{\delta}{Z} < 2$ is always guaranteed, since on most robots $\delta < 1$ m, which is much less than the scene depth in outdoor environments. Moreover, Z is tuned by the user, and in [11], we have shown that overestimating it with respect to its real value is more effective than underestimating it, for navigation performance.

2) Linear Velocity In The Safe and Unsafe Context: Let us now define the velocities v_s and v_u, used in (9).

As we mentioned, when the features motion in the image is fast, the visual tracker is less effective, and the linear velocity should be reduced. To assess the features motion, we compare their position in the current image I and in the next I^d and second next I^{sd} key images[2]. Since the robot moves on a plane, we relate v_s to the image abscissa error between the centroids of the points matched in I and I^d:

$$e^d = x - x^d$$

and to the abscissa error between the centroids of the points matched in I and I^{sd}:

$$e^{sd} = x - x^{sd}$$

[2]When the next image is the final one, we use $I^{sd} = I^d$, since the second next key image is undefined.

Then, defining the weighted average between these errors as:

$$e = \frac{2e^d + e^{sd}}{3}$$

we design the linear velocity in the safe context as:

$$v_s = \frac{V}{2}\left[1 + \tanh\left(\pi - \frac{|e|}{\gamma}\right)\right] \tag{13}$$

Function (13) has an upper bound $V > 0$, and smoothly decreases to 0, as $|e|$ grows. Its inflection point is determined by empirically tuned parameter $\gamma > 0$.

Instead, in the presence of obstacles, the robot should reduce its linear velocity to v_u, and stop for unitary H. Imposing $v = 0$ when $H = 1$ in (9) obviously yields:

$$v_u = 0$$

as our design choice for the velocity in the unsafe context.

3) Modeling The Obstacles: For obstacle modeling, we use an occupancy grid linked to $\mathcal{F}_{\mathcal{R}}$, with cell sides parallel to the X' and Z' axes (see Fig. 1(c)). Its forward and lateral extensions are smaller than the scanner radial range, to ignore obstacles that are too far to jeopardize the robot. Given an arbitrary integer K, the grid is built from the latest $2K+1$ scans. For each cell centered at $c = (X', Z')$, we define the $2K+1$ *occupancies* r at the j-th oldest iteration as:

$$r_j(c) = \{0, 1\}, \qquad j = 0, \ldots, 2K+1$$

We set $r_j = 1$ if an obstacle has been sensed in c at the j-th iteration prior to the current one, and 0 otherwise. Then, we associate to each cell a coefficient $\mu(c)$, obtained by linear combination of the occupancies, weighted with a normalized Gaussian filter that smoothens the cell effect over time:

$$\mu(c) = \sum_{j=0}^{2K+1} \frac{e^{-(j-K)^2/K}}{\sqrt{K\pi}}\ r_j(c)$$

The filter maximum weight is set at the K-th latest scan, to avoid control input overshoot at a new obstacle detection (an issue that existed in [1]). If the robot velocity is negligible with respect to the scanner acquisition frequency, and K is reasonably small, the effect of motion on the occupancies can be ignored, and this model is consistent with the current robot position. We will show that the above assumptions are appropriate in our experimental setup.

Obstacle avoidance is derived by using vortex potential fields [22]. For each cell c, we define the potential:

$$U_c = \frac{\mu(c)}{\|c\|}$$

where $\mu(c)$ has been defined above, and $\|c\|$ is the distance from R to the cell[3]. In practice, for two cells with equal $\mu(c) \neq 0$, the nearest one will yield the highest potential. We define the vortex field for each cell as the rotor of U_c:

$$f_c = \begin{bmatrix} f_{c,X'} \\ f_{c,Z'} \end{bmatrix} = \begin{bmatrix} \pm\dfrac{\partial U_c}{\partial Z'} \\ \mp\dfrac{\partial U_c}{\partial X'} \end{bmatrix} = \mu(c) \begin{bmatrix} \mp\dfrac{Z'}{\|c\|^3} \\ \pm\dfrac{X'}{\|c\|^3} \end{bmatrix}$$

[3]Designing the grid without the cell at $R = (0, 0)$ (where obstacles are not detectable by the range scanner), guarantees that U_c is non-singular.

The signs of $f_{c,X'}$ and $f_{c,Z'}$ depend on the cell abscissa: X' positive (strictly negative) will induce a clockwise (counter-clockwise) vortex, so that the field always points forward. The fields $f_{c,i}$ generated by all the n cells c_i are then superimposed to obtain the total field:

$$f = \sum_{i=1}^{n} f_{c,i}$$

The orientation $\alpha \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$ of this field is (see Fig. 1):

$$\alpha = \begin{cases} 0 & \text{if } f = 0 \\ -\text{ATAN2}\ (f_{X'}, f_{Z'}) & \text{otherwise} \end{cases}$$

As in [22], this is the desired heading for circumnavigation in unsafe situations (see Sect. II-B). However, α alone is not indicative of the danger. Thus, we also use the field norm:

$$|f| = \sqrt{f_{X'}^2 + f_{Z'}^2}$$

which is a good metric for evaluating the obstacle distance, and reducing v accordingly, when the obstacles are near. Hence, to assess the context danger, we consider both α and $|f|$, and using two empirically tuned thresholds ρ and ϱ such that $0 < \rho < \varrho$, we design the activation function as:

$$H = \begin{cases} \kappa\,|\alpha| & \text{if } |f| <= \rho \\ 1 & \text{if } |f| >= \varrho \\ \frac{1+\kappa|\alpha|}{2} + \frac{1-\kappa|\alpha|}{2}\tanh\left(\frac{1}{\varrho-|f|} + \frac{1}{\rho-|f|}\right) & \text{otherwise} \end{cases}$$

Note that $H = 0$ if no obstacle is detected (since both $|f|$ and α are null in that case), and it is bounded by 1. For small $|f|$, H is determined only by α: the obstacles are 'far enough' to be circumnavigated, and parameter $\kappa \in \left]0, \frac{2}{\pi}\right]$ weighs the danger provoked by the field orientation[4]. On the other hand, for large $|f| \geq \varrho$, the obstacles are 'too near': $H = 1$ and the robot must stop. A hyperbolic function guarantees that H is $\mathcal{C}^\infty$ for all $(X', Z') \in I\!R^{*2}$. Parameters ρ and ϱ are tuned to determine the three intervals. For example, small values of ϱ make the approach more conservative, since the robot will stop even in the presence of distant obstacles.

D. Control Analysis

In this section, we will instantiate and comment our controller for visual navigation with obstacle avoidance. Plugging all the variables defined above in (8) we obtain:

$$\begin{cases} v = (1-H)\, v_s \\ \omega = (1-H)\, \frac{\lambda_x\left(x^d - x\right) - j_v v_s + \lambda_\varphi j_\varphi \varphi}{j_\omega} + H\lambda_\alpha \alpha \\ \dot{\varphi} = H\, \frac{\lambda_x\left(x^d - x\right) - \lambda_\alpha j_\omega \alpha}{j_\varphi} - (1-H)\, \lambda_\varphi \varphi \end{cases} \tag{14}$$

This control law has the following interesting properties.

1) In the *safe* context ($H = 0$), (14) becomes:

$$\begin{cases} v = v_s \\ \omega = \frac{\lambda_x\left(x^d - x\right) - j_v v_s + \lambda_\varphi j_\varphi \varphi}{j_\omega} \\ \dot{\varphi} = -\lambda_\varphi \varphi \end{cases} \tag{15}$$

As in [10] and [11], where obstacles were not considered, the visual task (1) is realized by ω, which also compensates the centroid displacements due to v and

[4]To guarantee $H < 1$, parameter κ must be chosen smaller than $\frac{2}{\pi}$.

Fig. 3. Top to bottom: experiments 1 to 3. Left: reconstruction of the replayed path with (red) and without (yellow) obstacles. Right: snapshots (above) and grids (below) in the presence of obstacles. The red vectors show the orientation of f.

to $\dot{\varphi}$ through the image jacobian components (11). The two other specifications in (3) are achieved by v and $\dot{\varphi}$: v is regulated to improve tracking according to (13), while the camera is driven forward, to $\varphi = 0$.

2) In the *unsafe* context when $H \approx 1$, a good approximation of (14) is:

$$\begin{cases} v = (1-H)\, v_s \\ \omega = \lambda_\alpha \alpha \\ \dot{\varphi} = \frac{\lambda_x \left(x^d - x\right) - \lambda_\alpha j_\omega \alpha}{j_{\dot{\varphi}}} \end{cases} \tag{16}$$

In this case, the visual task (1) is executed by $\dot{\varphi}$, which also compensates the robot rotation, to keep the features in view. The two other specifications are ensured by v and ω: the linear velocity is reduced (and even zeroed to $v = v_u = 0$, for $H = 1$), while the angular velocity aligns the robot with f.

3) Control law (14) can also be derived using a redundancy-based framework, as has been done in [1]. In that work, we proved that for a similar problem, obstacle avoidance had no effect on the visual task, which could be achieved for any $H \in [0, 1]$. Note that also here, plugging the expressions of v, ω, and $\dot{\varphi}$ from (14) into the visual task equation:

$$\dot{x} = j_v v + j_\omega \omega + j_{\dot{\varphi}} \dot{\varphi}$$

yields (1). Thus, desired state x^d is globally asymptotically stable for the closed loop system, $\forall H \in [0, 1]$.

III. EXPERIMENTAL RESULTS

Here, we report the experiments obtained with controller (14) and shown in the video attached to this paper. All experiments have been carried out on a CyCab robot equipped with a 70° field of view, B&W camera mounted on a TRACLabs Biclops Pan/Tilt head (the tilt angle is null, to keep the optical axis parallel to the ground), and with a 4-layer, 110° scanning angle, laser SICK LD-MRS. The offset between R and C is $\delta = 0.7$ m, and we set $Z = 15$ m that meets condition $Z > \frac{\delta}{2}$. The grid is limited to 1.5 m on each side, and to 5 m in front, and it is built by projecting the laser readings from the 4 layers on the ground. Since camera (10 Hz) and laser (40 Hz) processing are not synchronized, they run on two different threads, and control input $\mathbf{u}$ is sent to the robot when the visual information is available (10 Hz). For the linear velocity design, we use $V = 0.4$ ms^{-1} and $\gamma = 75$. The velocity is limited for safety reasons, since the framework runs on a real robot, operating on our campus, with image processing at 10 Hz. With $V = 0.4$ ms^{-1}, scanner acquisition at 40 Hz, and $K = 20$, neglecting the effect of motion on the occupancies is reasonable. In all experiments, we set: $\lambda_x = 0.5$, $\lambda_\varphi = \lambda_\alpha = 0.3$, $\kappa = 0.25$, $\rho = 3$ and $\varrho = 4$.

Controller (14) has been validated in the three experiments in Fig. 3. In each case, after manually driving CyCab to acquire the image database, we test two setups (safe and unsafe). First, no obstacle is present: since $H = 0$, (15) is used, with fixed forward-looking camera ($\varphi = 0$). Then, some obstacles (including pedestrians) are present, near and on the taught path, and the robot must deal with them; in addition, the obstacles may occlude the features. The replayed paths, estimated from odometry and snapshots, are outlined in Fig. 3 (left), in the safe (yellow) and unsafe (red) setup. Since the focus here is on obstacle avoidance, the taught path is not drawn. It is hard to find a general interpretation of the results, since these depend on many factors (e.g., position of obstacles and visual features). Nevertheless, in all the experiments, the robot follows the visual path without colliding, in spite of occlusions (although a minimal number of visual features is obviously required). Although some portions of the replayed 3D path are far from the taught one, these motions are indispensable to avoid collisions. In the following, we detail the experiments.

The path used in experiment 1 is 60 m long, and composed of $N = 34$ key images. Three obstacles are present, and the robot overtakes them all (the first and third on the left, the second on the right), to reach the last key image. For the unsafe setup, snapshots are shown in Fig. 3 (top), and v, ω, $\dot{\varphi}$ and φ are plotted in Fig. 4 (left). The iterations with $H > 0$ are highlighted in yellow. The smooth trend of v at the beginning and end of all three experiments is due to the acceleration saturation carried out at the CyCab low-level control. CyCab is initially near a wall parallel to the Z' axis, which is too far to be considered in the grid. This shows the utility of ignoring lateral data, which would have driven the robot away from the path. The first obstacle is detected at iteration 250: vector field f generates a positive rotation (green in the figure), compensated by the camera pan, and v is reduced. The obstacle is overtaken while maintaining feature visibility. Then, since the path is free, the pan angle (blue) is zeroed, until the second obstacle, which triggers negative ω, and positive $\dot{\varphi}$. Afterwards, H is activated by the third obstacle, which is overtaken on the left (iteration 1200), while decelerating. Again, H is canceled at iteration 1400, and the robot is driven by (15). The average velocity v is slightly reduced, from 0.39 ms^{-1} in the safe setup, to 0.35 in the presence of obstacles. The image error with respect to the database, averaged over the experiment, increases from 11 to 17 pixels. This slight increment in the unsafe setup is simply due to the visual occlusions provoked by the obstacles (as shown in the video).

Experiments 2 and 3 are carried out on a straight path of 60 m and $N = 30$ key images.

In experiment 2, a grey car, which was not present during teaching, is on the path during navigation, and a barrier blocks the road halfway through. Snapshots of the experiment are shown in the center of Fig. 3, and the velocities are plotted in Fig. 4 (center). The grey car is overtaken on the left at iterations 240−380, while the robot slows down (black curve). Then, as soon as the context is safe again, the robot

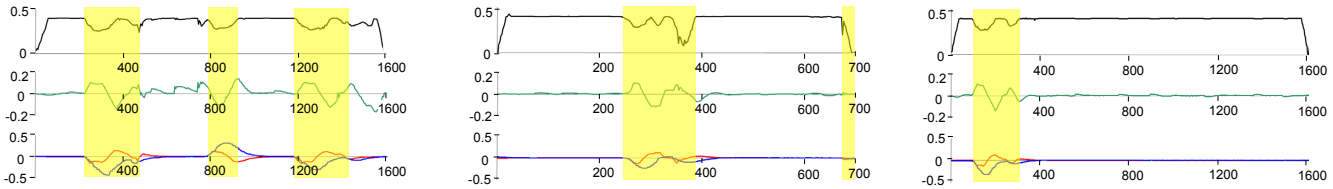

Fig. 4. Relevant variables in experiments 1 to 3 (left to right): v (black, in ms^{-1}), ω (green, in rad s^{-1}), φ (blue, in rad) and $\dot{\varphi}$ (red, in rad s^{-1}). The iterations with non-null H are highlighted in yellow.

accelerates and returns on the path, while the camera is reset forward. The barrier is more difficult to deal with. In fact, when detected, it is centered on the Z' axis and orthogonal to it (see the grids on Fig. 3); this induces $\alpha \approx 0$ and drives the robot towards the barrier. However, as the CyCab approaches the barrier, the norm of f increases, and eventually becomes greater than ϱ, to make $H = 1$ and stop the robot. Note that when there is no obstacle, $v \approx V$, because, in contrast with experiment 1, the taught path here is straight, leading to $v_s \approx V$, from (13). The value of v, averaged over the path diminishes, from 0.39 in the safe setup, to 0.34 in the unsafe one. This time, the image error, averaged up to the barrier, does not vary in the two setups (13 pixels in both cases).

In experiment 3, although the grey car and barrier are not present anymore, three persons are standing on the path. On Fig. 3 (bottom), we show some snapshots of the experiment, while the velocities are plotted in Fig. 4 (right). From iteration 100 to 320, the activation function is triggered by the detection of the pedestrians. This generates a counterclockwise angular velocity (the persons are overtaken on the left), and a slight deceleration. Afterwards, since the street is free again, the robot accelerates and returns on the path, which this time is completed up to the last key image, since the barrier has been removed. Just like in experiment 2, when there is no obstacle, $v \approx V$, since the path is straight. The average linear velocity ($v = 0.38$ ms^{-1}) is almost the same as in the safe setup (0.39), and so is the average image error (12 pixels).

IV. CONCLUSIONS

For the first time, a framework with simultaneous obstacle avoidance and outdoor visual navigation is presented. It merges techniques from potential fields and visual servoing, and guarantees path following, obstacle bypassing, and collision avoidance by deceleration. The method has been validated by outdoor experiments with real obstacles (parked cars and pedestrians). In the future, we plan to take into account moving obstacles, as well as visual occlusions provoked by the obstacles. Finally, it may be interesting to record scanner data during the teaching step too, in order to improve obstacle avoidance.

ACKNOWLEDGMENTS

This work has been supported by ANR (French National Agency) CityVIP project under grant ANR-07 TSFA-013-01.

REFERENCES

[1] A. Cherubini and F. Chaumette, "A redundancy-based approach for obstacle avoidance in mobile robot navigation", *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2010.

[2] M. Buehler, K. Lagnemma, S. Singh (Editors), "Special Issue on the 2007 DARPA Urban Challenge, Part I-III", in *Journal of Field Robotics*, vol. 25, no. 8–10, 2008, pp. 423–860.

[3] A. Broggi, L. Bombini, S. Cattani, P. Cerri and R. I. Fedriga, "Sensing requirements for a 13000 km intercontinental autonomous drive", IEEE Intelligent Vehicles Symposium, 2010.

[4] J. Minguez, F. Lamiraux and J. P. Laumond, "Motion planning and obstacle avoidance", in *Springer Handbook of Robotics*, B. Siciliano, O. Khatib (Eds.), Springer, 2008, pp. 827–852.

[5] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots", *IEEE Int. Conf. on Robotics and Automation*, 1985.

[6] F. Chaumette and S. Hutchinson, "Visual servo control tutorial, part I and II", *IEEE Robotics and Automation Magazine*, vol. 13, no. 4, and vol. 14, no. 1, 2007.

[7] G. L. Mariottini, G. Oriolo, D. Prattichizzo, "Image-Based Visual Servoing for Nonholonomic Mobile Robots Using Epipolar Geometry", *IEEE Trans. on Robotics*, vol. 23, no. 1, 2007, pp. 87–100.

[8] H. M. Becerra, G. López-Nicolás and C. Sagüés, "A Sliding-Mode-Control Law for Mobile Robots Based on Epipolar Visual Servoing From Three Views", *IEEE Trans. on Robotics*, vol. 27, no. 1, 2011, pp. 175–183.

[9] G. Allibert, E. Courtial and Y. Touré, "Real-time visual predictive controller for image-based trajectory tracking of a mobile robot", *Int. Federation of Automatic Control*, 2008.

[10] S. Šegvić, A. Remazeilles, A. Diosi and F. Chaumette, "A mapping and localization framework for scalable appearance-based navigation", *Computer Vision and Image Understanding*, vol. 113, no. 2, 2008, pp. 172–187.

[11] A. Cherubini, M. Colafrancesco, G. Oriolo, L. Freda and F. Chaumette, "Comparing appearance-based controllers for nonholonomic navigation from a visual memory", *ICRA Workshop on safe navigation in open and dynamic environments*, 2009.

[12] E. Royer, M. Lhuillier, M. Dhome and J.-M. Lavest, "Monocular vision for mobile robot localization and autonomous navigation", *Int. Journal of Computer Vision*, vol. 74, no. 3, 2007, pp. 237–260.

[13] A. M. Zhang and L. Kleeman, "Robust Appearance Based Visual Route Following for Navigation in Large-scale Outdoor Environments", *Int. Journal of Robotics Research*, vol. 28, no. 3, 2009, pp. 331-356.

[14] D. Fontanelli, A. Danesi, F. A. W. Belo, P. Salaris and A. Bicchi, "Visual Servoing in the Large", *The International Journal of Robotics Research*, vol. 28, no. 6, 2009, pp. 802-814.

[15] J. Courbon, Y. Mezouar and P. Martinet, "Autonomous Navigation of Vehicles from a Visual Memory Using a Generic Camera Model", *IEEE Trans. on Intelligent Transportation Systems*, vol. 10, no. 3, 2009, pp. 392–402.

[16] A. Ohya, A. Kosaka and A. Kak, "Vision-based navigation by a mobile robot with obstacle avoidance using a single-camera vision and ultrasonic sensing", *IEEE Trans. on Robotics and Automation*, vol. 14, no. 6, 1998, pp. 969–978.

[17] Z. Yan, X. Xiaodong, P. Xuejun and W. Wei, "Mobile robot indoor navigation using laser range finder and monocular vision", *IEEE Int. Conf. on Robotics, Intelligent Systems and Signal Processing*, 2003.

[18] F. Lamiraux and O. Lefebvre, "Sensor-based trajectory deformation: application to reactive navigation of nonholonomic robots", in *Visual Servoing via Advanced Numerical Methods*, G. Chesi, K. Hashimoto (Eds.), Springer LNCIS Series, 2010, pp. 315–334.

[19] L. Lapierre, R. Zapata and P. Lepinay, "Simultaneous path following and obstacle avoidance control of a unicycle-type robot", *IEEE Int. Conf. on Robotics and Automation*, 2007.

[20] D. Folio and V. Cadenat, "A redundancy-based scheme to perform safe vision-based tasks amidst obstacles", *IEEE Int. Conf. on Robotics and Biomimetics*, 2006.

[21] O. Tahri and F. Chaumette, "Point-based and region-based image moments for visual servoing of planar objects", *IEEE Trans. on Robotics*, vol. 21, no. 6, 2005, pp. 1116–1127.

[22] A. De Luca and G. Oriolo, "Local Incremental Planning for Nonholonomic Mobile Robots", *IEEE Int. Conf. on Robotics and Automation*, 1994.