A Redundancy-Based Approach for Obstacle Avoidance in Mobile Robot Navigation

Andrea Cherubini and François Chaumette

***Abstract*— In this paper, we propose a framework for visual navigation with simultaneous obstacle avoidance. The obstacles are modeled by using a vortex potential field, derived from an occupancy grid. Kinematic redundancy guarantees that obstacle avoidance and navigation are achieved concurrently, and the whole scheme is merely sensor-based. The problem is solved both in an obstacle-free and in a dangerous context, and the control law is smoothened in the intermediate situations. In a series of simulations, we show that with our framework, a robot can replay a taught visual path while avoiding collisions, even in the presence of visual occlusions.**

***Index Terms*— Visual servoing, Navigation, Motion control.**

I. INTRODUCTION

Recently, a great amount of research in the robotics fields has focused on obstacle avoidance. Referring to [1], the obstacle avoidance task consists of computing, at each sample time, a motion control such that the trajectory generated is free of collisions, and makes the robot progress to the goal. One of the most common techniques for obstacle avoidance is the potential field method [2]. The gap between global path planning and real-time sensor-based control has been closed with the elastic band [3], a deformable collision-free path, whose initial shape is generated by a planner, and then deformed in real time, according to the sensed data. Instead of using a global model, which would infringe the *perception to action* paradigm [4], we propose a framework for obstacle avoidance with simultaneous execution of a sensor-based task. The task that we focus on is appearance-based navigation, which has been the target of our research in [5] and [6]. In the framework that we have developed in the past[1], the path is a topological graph, represented by a database of ordered key images. Navigation is divided into subtasks, each consisting of reaching the next key image in the database. Although this approach is quite popular in the robotics community [7], it has been rarely extended to take into account unexpected obstacles. An exception is presented in [8], which exploits redundancy [9].

Our work is also based on redundancy. A kinematically redundant robot possesses more DOFs than those required to execute its primary task. This provides increased dexterity that may be used for a secondary task (here, obstacle avoidance). The tasks can be executed by using a projecting operator [10], or a task priority strategy [11-12]. Sensor-based tasks have been tackled with redundancy in various works (e.g., on a robot arm in [13], and on a nonholonomic manipulator in [8]. Here, we focus on the following problem: a wheeled robot equipped with an actuated pinhole camera and with a range scanner must follow a visual path represented by a series of key images (primary task), while avoiding collisions with the ground obstacles (secondary task). The camera must detect the features necessary for navigation, while the range scanner senses the obstacles in front of the robot. In our work, the tasks are executed both in a dangerous context (where many obstacles are present), and in a safe context (where the near workspace is clear). An appropriate function is used to smoothen the transitions in between. Although similar to the one presented in [8], our framework guarantees that obstacle avoidance has no effect on visual navigation. Besides, we use a compact controller, and the transition between contexts is operated only at the secondary task level, whereas in [8], three controllers are necessary, and the transitions are more complicated.

A. Cherubini and F. Chaumette are with INRIA Rennes - Bretagne Atlantique, IRISA, Campus de Beaulieu 35042, Rennes, France. {Andrea.Cherubini, Francois.Chaumette}@irisa.fr

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

II. REDUNDANCY FRAMEWORK FOR 2 TASKS

We hereby utilize the relationships between the robot kinematic control inputs and two desired tasks, to recall the redundancy-based controller used in [12], and originally introduced in [11]. We name $\mathbf{s}_1 \in I\!R^k$ the primary task vector, and $\mathbf{u} \in I\!R^m$ the control inputs. Redundancy exists when $m > k$, and enables handling a secondary task, $\mathbf{s}_2 \in I\!R^{m-k}$. The tasks dynamics are related to the control inputs by the equations:

$$\dot{\mathbf{s}}_i = \mathbf{J}_i \mathbf{u} \qquad i = 1, 2 \tag{1}$$

where $\mathbf{J}_1$ and $\mathbf{J}_2$ are the *task jacobians*, respectively of size $k \times m$ and $(m-k) \times m$. We assume that both have full rank.

With the *Projected Gradient* method [10], the solution of (1), for tracking the primary task [4] is:

$$\mathbf{u} = \mathbf{J}_1^+ \dot{\mathbf{s}}_1^* + \mathbf{P}_1 \mathbf{h} \tag{2}$$

In the above equation:

- $\mathbf{J}_1^+ = \mathbf{J}_1^\top \left(\mathbf{J}_1 \mathbf{J}_1^\top\right)^{-1}$ is the $m \times k$ Moore-Penrose pseudoinverse of $\mathbf{J}_1$, i.e., a particular solution of: $\mathbf{J}_1 \mathbf{J}_1^+ = \mathbf{I}$;
- $\dot{\mathbf{s}}_1^* \in I\!R^k$ is the required primary task error evolution;
- $\mathbf{h} \in I\!R^m$ is an arbitrary control input;
- $\mathbf{P}_1 = \mathbf{I} - \mathbf{J}_1^+ \mathbf{J}_1$ projects $\mathbf{h}$ onto the null space of $\mathbf{J}_1$; hence, the second part of (2) has no effect on the primary task.

Vector $\mathbf{h}$ can be used to apply a command that will carry out at best the secondary task $\mathbf{s}_2$, without disturbing $\mathbf{s}_1$.

Introducing (2) in (1) (with $i = 2$), we obtain:

$$\dot{\mathbf{s}}_2 = \mathbf{J}_2 \mathbf{J}_1^+ \dot{\mathbf{s}}_1^* + \mathbf{J}_2 \mathbf{P}_1 \mathbf{h}$$

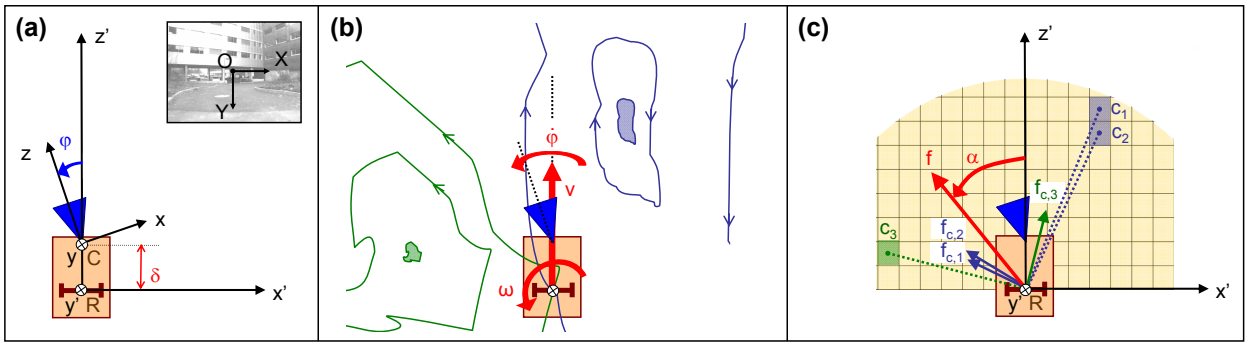

Fig. 1. Top view of the mobile robot (orange), equipped with actuated pinhole camera (blue). (a) Reference frames. (b) Outline of the obstacle-induced vortex fields, and control variables (v, ω, $\dot{\varphi}$). (c) Occupancy grid and potential field f construction.

The authors of [11] and [12] derive $\mathbf{h}$ from this equation in order to track the desired secondary task dynamics $\dot{\mathbf{s}}_2^*$:

$$\mathbf{h} = (\mathbf{J}_2\mathbf{P}_1)^+ \left(\dot{\mathbf{s}}_2^* - \mathbf{J}_2\mathbf{J}_1^+\dot{\mathbf{s}}_1^*\right) \tag{3}$$

Plugging into (2), we obtain:

$$\mathbf{u} = \mathbf{J}_1^+\dot{\mathbf{s}}_1^* + \mathbf{P}_1 \left(\mathbf{J}_2\mathbf{P}_1\right)^+ \left(\dot{\mathbf{s}}_2^* - \mathbf{J}_2\mathbf{J}_1^+\dot{\mathbf{s}}_1^*\right)$$

The term $\mathbf{J}_2\mathbf{P}_1$ gives the available range for $\mathbf{s}_2^*$ to be performed without affecting $\mathbf{s}_1$, while $\dot{\mathbf{s}}_2^* - \mathbf{J}_2\mathbf{J}_1^+\dot{\mathbf{s}}_1^*$ is the secondary task function, without the part $\mathbf{J}_2\mathbf{J}_1^+\dot{\mathbf{s}}_1^*$ accomplished by the first task. Since $\mathbf{P}_1$ is Hermitian and idempotent (it is a projection operator), the above equation becomes:

$$\mathbf{u} = \mathbf{J}_1^+\dot{\mathbf{s}}_1^* + \mathbf{h} \tag{4}$$

with $\mathbf{h}$ defined in (3).

The desired evolution of the task errors can be written:

$$\dot{\mathbf{s}}_i^* = \dot{\hat{\mathbf{s}}}_i - \lambda_i \left(\mathbf{s}_i - \hat{\mathbf{s}}_i\right) \qquad i = 1, 2 \tag{5}$$

with $\hat{\mathbf{s}}_i$ and $\dot{\hat{\mathbf{s}}}_i$ indicating the desired values of the i-th task, and of its first derivative, and $\lambda_i > 0$ the associated gain.

Controller (4) has the following properties.

- It guarantees convergence of the primary task. Replacing (4) in (1) yields:

  $$\dot{\mathbf{s}}_1 = \dot{\mathbf{s}}_1^*$$

  Considering (5), this is equivalent to the linear system:

  $$\dot{\mathbf{s}}_1 - \dot{\hat{\mathbf{s}}}_1 = -\lambda_1 \left(\mathbf{s}_1 - \hat{\mathbf{s}}_1\right)$$

  for which, as desired, $\left(\hat{\mathbf{s}}_1, \dot{\hat{\mathbf{s}}}_1\right)$ is an exponentially stable equilibrium, since $\lambda_1 > 0$.
- The secondary task has no effect on the primary task, since $\mathbf{h}$ is projected onto the null space of $\mathbf{J}_1$.
- The secondary task $\mathbf{s}_2$ is realized at best, under the constraint that it does not affect the primary task.

III. PROBLEM CHARACTERISTICS

For the following definitions, the reader is referred to Fig. 1. The robot control inputs are:

$$\mathbf{u} = [v\ \omega\ \dot{\varphi}]^\top$$

These are, respectively, the linear and angular velocities of the robot, and the camera pan angular velocity. We assume that the camera angle is bounded: $|\varphi| \leq \frac{\pi}{2}$, and that the path can be tracked with continuous $v(t) > 0$. This guarantees safety, since only obstacles in front of the robot can be detected by the range scanner. We also define the robot frame $\mathcal{F}_\mathcal{R}(R, x', y', z')$ (R is the center of rotation of ω), 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). We neglect the distance between C and the camera pan rotation axis. The distance between R and C is denoted by $\delta \geq 0$. We use the normalized perspective camera model:

$$X = \frac{x}{z} \qquad\qquad Y = \frac{y}{z}$$

The four specifications that we intend to fulfill are:

1) visual path following,
2) desired linear velocity $\hat{v}$,
3) security with respect to the obstacles,
4) maximal feature visibility.

In our problem, $m = 3$. Hence, we will fulfill a primary task of dimension $k = 2$ (visual path following at desired $\hat{v}$), and use the extra degree of freedom for a secondary task $\mathbf{s}_2 \in I\!R$ that realizes the two other specifications. Since only 1 DOF is available for the 2 specifications, we discern two contexts (*dangerous* and *safe*): in each context, one specification has priority over the other. To smoothen transitions, we design an *obstacle activation function*:

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

where the context $\mathcal{C}$ indicates the danger represented by detected obstacles. The definition of H will be given later.

- In the *dangerous context* ($H = 1$), the priority is security. The robot must avoid collisions, by orienting its heading to a desired value α, related to the obstacles position (see Fig. 1), and defined later in the paper. This is done by imposing:

  $$\omega = \lambda_2\alpha \qquad \forall\ (\mathbf{s}_1, \dot{\mathbf{s}}_1) \in I\!R^2 \times I\!R^2 \tag{6}$$

  This condition will be guaranteed by control input $\mathbf{h}_d$, defined below.
- In the *safe context* ($H = 0$), the priority is maximal visibility. We do not take directly into account occlusions, or feature position, and assume that the visible features are homogeneously distributed in the environment. Then, visibility can be maximized by pointing the camera forward: $\varphi = 0$. This can be obtained with:

  $$\dot{\varphi} = -\lambda_2\varphi \qquad \forall\ (\mathbf{s}_1, \dot{\mathbf{s}}_1) \in I\!R^2 \times I\!R^2 \tag{7}$$

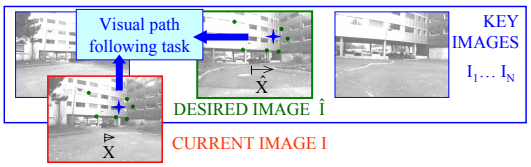

Fig. 2. The current and desired images contain some common visual features, which are used for navigation.

This condition will be guaranteed by control input $\mathbf{h}_s$, defined just below.

A smooth transition between the inputs associated to the two contexts can be obtained by setting, in (4):

$$\mathbf{h} = H\mathbf{h}_d + (1-H)\,\mathbf{h}_s$$

to obtain:

$$\begin{cases} \mathbf{u} = \mathbf{J}_1^+\dot{\mathbf{s}}_1^* + H\mathbf{h}_d + (1-H)\,\mathbf{h}_s \\ \mathbf{h}_d = (\mathbf{J}_d\mathbf{P}_1)^+\left(\dot{\mathbf{s}}_d^* - \mathbf{J}_d\mathbf{J}_1^+\dot{\mathbf{s}}_1^*\right) \\ \mathbf{h}_s = (\mathbf{J}_s\mathbf{P}_1)^+\left(\dot{\mathbf{s}}_s^* - \mathbf{J}_s\mathbf{J}_1^+\dot{\mathbf{s}}_1^*\right) \end{cases} \tag{8}$$

In the above equation:

- $\dot{\mathbf{s}}_{d,s}^*$ are the desired secondary tasks dynamics, in the dangerous and safe context; as in (5), they are:

$$\begin{cases} \dot{\mathbf{s}}_d^* = \dot{\hat{\mathbf{s}}}_d - \lambda_2\left(\mathbf{s}_d - \hat{\mathbf{s}}_d\right) \\ \dot{\mathbf{s}}_s^* = \dot{\hat{\mathbf{s}}}_s - \lambda_2\left(\mathbf{s}_s - \hat{\mathbf{s}}_s\right) \end{cases} \tag{9}$$

- $\mathbf{J}_{d,s}$ are the corresponding jacobian matrices.

In the next section, controller (8) will be instantiated for the problem of visual navigation with obstacle avoidance.

IV. VISUAL NAVIGATION AMONG OBSTACLES

A. Primary Task

Visual Path Following: The robot must follow a path defined by a set of N ordered key images $I_1, \ldots, I_N$ (Fig. 2). This task is divided into N subtasks, each consisting of driving the current image I to the next key image $\hat{I}$. We consider that $\hat{I}$ is reached, as soon as the image error e (i.e., the average distance in the image space between the points matched in I and $\hat{I}$) is below a threshold τ, and starts to rise. When $\hat{I}$ is reached, a topological transition is made: the next image in the set becomes the desired one, and so on, until I_N is reached. This task can be achieved by using only 1 visual feature: the abscissa X of the centroid of the points matched on I and $\hat{I}$ [5]. Hence, for the first component of $\mathbf{s}_1$ we use:

$$s_{1,1} = X \qquad \hat{s}_{1,1} = \hat{X} \qquad \dot{\hat{s}}_{1,1} = 0$$

which, replaced in (5), yields: $\dot{s}_{1,1}^* = \lambda_1\left(\hat{X} - X\right)$.

Assuming static features, the corresponding row of $\mathbf{J}_1$ is:

$$[j_v \; j_\omega \; j_{\dot{\varphi}}] \tag{10}$$

where a good approximation of the scalar components is (see [6]):

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

In (11), to avoid depth estimation, which can be unreliable, we set the depths to a fixed value z, as in [6].

Desired Linear Velocity: To track a desired linear velocity $\hat{v}(t)$, we define the second component of $\mathbf{s}_1$ as:

$$s_{1,2} = \int_0^t v(t)\,dt \qquad \hat{s}_{1,2} = \int_0^t \hat{v}(t)\,dt \qquad \dot{\hat{s}}_{1,2} = \hat{v}(t)$$

which, replaced in (5), yields:

$$\dot{s}_{1,2}^* = \hat{v} + \lambda_1\left[\int_0^t (\hat{v} - v)\,dt\right]$$

It is trivial to obtain the corresponding row of $\mathbf{J}_1$:

$$[1\;0\;0] \tag{12}$$

Jacobian: The primary task Jacobian is obtained by stacking (10) and (12):

$$\mathbf{J}_1 = \begin{bmatrix} j_v & j_\omega & j_{\dot{\varphi}} \\ 1 & 0 & 0 \end{bmatrix}$$

The Moore-Penrose pseudoinverse of $\mathbf{J}_1$ is:

$$\mathbf{J}_1^+ = \frac{1}{j_\omega^2 + j_{\dot{\varphi}}^2}\begin{bmatrix} 0 & j_\omega^2 + j_{\dot{\varphi}}^2 \\ j_\omega & -j_\omega j_v \\ j_{\dot{\varphi}} & -j_{\dot{\varphi}} j_v \end{bmatrix}$$

Since $j_{\dot{\varphi}} \neq 0$, $\forall X \in I\!R$ (see (11)), $\mathbf{J}_1$ and $\mathbf{J}_1^+$ are full rank. The projecting operator $\mathbf{P}_1$ can also be derived:

$$\mathbf{P}_1 = \frac{1}{j_\omega^2 + j_{\dot{\varphi}}^2}\begin{bmatrix} 0 & 0 & 0 \\ 0 & j_{\dot{\varphi}}^2 & -j_\omega j_{\dot{\varphi}} \\ 0 & -j_\omega j_{\dot{\varphi}} & j_\omega^2 \end{bmatrix}$$

B. Secondary Task

Modeling the Obstacles: For obstacle modeling, we use an occupancy grid [14], shown in Fig. 1 (c): it is rigidly linked to $\mathcal{F}_\mathcal{R}$, with cell sides parallel to the x' and z' axes. Its extension is limited by the sensor range, and we limit it laterally by ignoring obstacles too far on the sides, to jeopardize the robot. The grid is built from the latest K scans. For each cell centered at $c = [x'\; z']^\top$, we define the K *occupancies* r at the j-th oldest iteration as:

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

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, weighed with a factor that diminishes for the more remote data:

$$\mu(c) = \sum_{j=0}^{K} e^{-j/K} r_j(c)$$

Although local, the grid is effective for two reasons. First, if the sensor acquisition time is negligible with respect to the robot velocity, the grid is consistent with the robot position, and even moving obstacles, are unlikely to be missed. Second, the use of consecutive sensor readings provides smooth transitions in the model evolution.

Obstacle avoidance is derived by using vortex potential fields [15]. For each cell c, we define the potential:

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

where $\|c\|$ is the distance of the cell from R^2. The vortex field for each cell is simply the rotor of U_c:

$$f_c = \begin{bmatrix} f_{c,x'} \\ f_{c,z'} \end{bmatrix} = \begin{bmatrix} \pm\dfrac{\partial U}{\partial z'} \\ \mp\dfrac{\partial U}{\partial x'} \end{bmatrix} = \mu(c) \begin{bmatrix} \mp\dfrac{z'}{\|c\|^3} \\ \pm\dfrac{x'}{\|c\|^3} \end{bmatrix}$$

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

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

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

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

This variable will be used to define both the context $\mathcal{C}$, and the desired heading in dangerous situations.

Obstacle Activation Function: For interpolating between the safe and dangerous contexts, we use:

$$H = \begin{cases} 0 & \text{if } \alpha = 0 \\ 1 & \text{if } |\alpha| >= \beta \\ \frac{1}{2}\left(1+\tanh\left(\frac{1}{1-\alpha/\beta} - \frac{\beta}{\alpha}\right)\right) & \text{otherwise} \end{cases}$$

where $\beta > 0$ tunes the length of the transition interval $]0, \beta[$. Function H is $\mathcal{C}^\infty$ for all α, and monotonically increases with $|\alpha|$, from $H = 0$ when no obstacle is detected, to $H = 1$ when the obstacle vector field strongly deviates the robot from the forward direction. We evaluate the danger by using the field orientation, rather than its norm: near obstacles which do not cause strong deviations are thus considered less dangerous than farther - but deviating - obstacles.

Dangerous Context: In the dangerous context, the secondary task consists of aligning the robot with f. Hence, its heading must be shifted by α, with null angular speed at the equilibrium. Denoting with θ the current robot orientation, this can be done by using:

$$\mathbf{s}_d = \theta \qquad \hat{\mathbf{s}}_d = \theta + \alpha \qquad \dot{\hat{\mathbf{s}}}_d = 0$$

which, replaced in (9), yields:

$$\dot{\mathbf{s}}_d^* = \lambda_2 \alpha$$

Since deriving s_d yields ω, the corresponding secondary task Jacobian is:

$$\mathbf{J}_d = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$$

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

Safe Context: In the safe context, the camera pan should point forward. Hence, the pan angle should be set to 0, with null angular speed at the equilibrium. This task can be realized by using:

$$\mathbf{s}_s = \varphi \qquad \hat{\mathbf{s}}_s = 0 \qquad \dot{\hat{\mathbf{s}}}_s = 0$$

which, replaced in (9), yields:

$$\dot{\mathbf{s}}_s^* = -\lambda_2 \varphi$$

Since deriving s_s yields $\dot{\varphi}$, the corresponding secondary task Jacobian is:

$$\mathbf{J}_s = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$$

C. Complete Control Design

Let us now use the variables defined above, to instantiate our controller (8) for visual navigation:

$$\begin{cases} v = \hat{v} + \lambda_1 \left[\int_0^t (\hat{v} - v)\, dt\right] \\ \omega = (1-H)\, \frac{\lambda_1\left(\hat{X}-X\right) - j_v v + j_{\dot{\varphi}} \lambda_2 \varphi}{j_\omega} + H \lambda_2 \alpha \\ \dot{\varphi} = H\, \frac{\lambda_1\left(\hat{X}-X\right) - j_v v - j_\omega \lambda_2 \alpha}{j_{\dot{\varphi}}} - (1-H)\, \lambda_2 \varphi \end{cases} \tag{13}$$

This control law has the following nice properties.

1) In the *dangerous* context ($H = 1$), (13) becomes:

$$\begin{cases} v = \hat{v} + \lambda_1 \left[\int_0^t (\hat{v} - v)\, dt\right] \\ \omega = \lambda_2 \alpha \\ \dot{\varphi} = \frac{\lambda_1\left(\hat{X}-X\right) - j_v v - j_\omega \lambda_2 \alpha}{j_{\dot{\varphi}}} \end{cases}$$

The angular velocity aligns the robot with f. The camera velocity compensates such rotation, to keep the features in view. Note that, for large z, and small image error, $\dot{\varphi} \approx -\omega$ (since, from (11), $j_v \approx 0$ and $j_\omega \approx j_{\dot{\varphi}}$), which is an expected behavior.

2) In the *safe* context ($H = 0$), (13) becomes:

$$\begin{cases} v = \hat{v} + \lambda_1 \left[\int_0^t (\hat{v} - v)\, dt\right] \\ \omega = \frac{\lambda_1\left(\hat{X}-X\right) - j_v v + j_{\dot{\varphi}} \lambda_2 \varphi}{j_\omega} \\ \dot{\varphi} = -\lambda_2 \varphi \end{cases} \tag{14}$$

The camera is driven forward, to $\varphi = 0$. The image error is controlled only by ω, as in [5] and [6], where obstacles were not considered.

3) Controller (13) is well defined when the centroid is visible, if $z > \frac{\delta}{2}$, as we will show in Theorem 1. Moreover, in [6], we have shown that overestimating z with respect to its real value, is more effective than underestimating it, in terms of navigation performance.

Theorem 1: For (13) to be well defined when the centroid is visible, it is sufficient to set: $z > \frac{\delta}{2}$.

Proof: A sufficient condition for (13) to be well defined, is that both j_ω and $j_{\dot{\varphi}}$ are non-null. The first case ($j_\omega \neq 0$) is equivalent to:

$$\frac{\delta\,(\cos\varphi + X \sin\varphi)}{z} + 1 + X^2 \neq 0 \tag{15}$$

Since $|\varphi| \leq \frac{\pi}{2}$: $\cos\varphi + X \sin\varphi \geq -X$, $\forall X \in I\!R$. Hence, a sufficient condition for (15) is:

$$X^2 - \frac{\delta}{z} X + 1 > 0$$

which occurs $\forall X \in I\!R$ when $\frac{\delta}{z} < 2$. The second case ($j_{\dot{\varphi}} \neq 0$), is guaranteed $\forall X \in I\!R$. ■

V. EXPERIMENTS

For simulations, we made use of Webots[3], where a unicycle robot with a 320×240 pixels camera and a scanner of range 3 m have been designed. Both sensors operate at 30 Hz. The offset between R and C is $\delta = 0.7$ m. The visual features, represented by green spheres, are distributed randomly in the environment. We set $z = 5$ m (which meets the condition in Theorem 1) and we limit the lateral extent of the grid to 1.5 m on each side. We impose a constant linear velocity $\hat{v} = 2$ m s^{-1}. At this speed, since the scanner acquires at 30 Hz, the grid is consistent with the robot position, because 45 grids are built while the robot covers the sensor extension. At first, no obstacle is present in the environment; the robot is driven along a 60 m closed path (black in Fig. 3), and $N = 32$ key images are acquired. Then, 4 obstacles are located, near and on the taught path, and the robot must replay the visual path and avoid the obstacles. In addition to endangering the robot, the obstacles may occlude the features. Although the sensors are noise-free, and feature matching is ideal, these simulations allow validation of controller (13).

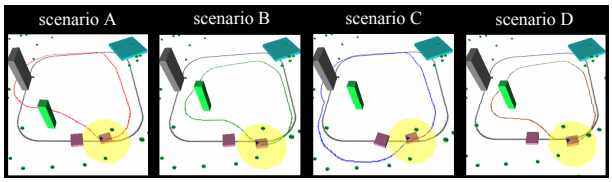

Fig. 3. Four obstacle scenarios designed for replaying the same taught path (black). Visual features are represented by the green points, the range scanner by a yellow disk, and the replayed paths are drawn in red (scenario A), green (B), blue (C), and brown (D).

By displacing the obstacles, we have designed the 4 scenarios in Fig. 3. In all scenarios, the robot is able to navigate without colliding, and this is done with the same parameters: $K = 5$, $\beta = 0.45$, $\lambda_1 = 3$ and $\lambda_2 = 1$. The threshold on image error e is $\tau = 10$ pixels. The metrics used to assess the experiments are: the image error with respect to the visual database, averaged over the whole experiment ($\bar{e}$, in pixels), and the distance, at the end of the experiment, from the final 3D key pose (Δ, in cm). The latter metric is less relevant, since navigation is defined in the image space.

The robot behavior is different in the four scenarios:

- Scenario A: the purple, gray, and cyan obstacles are located on the path, whereas the green one is near the path. The robot first senses the purple one, which is overtaken on the left, and then the green (also overtaken on the left). Then, the gray obstacle is overtaken on the right; finally, the robot converges back to the path using vision, and avoids the cyan obstacle, before returning to the initial position. The experiment is successful, with $\bar{e} = 9$, and $\Delta = 35$.
- Scenario B: it is identical to A, except that the green obstacle is farther from the path. This causes the robot to pass it on the right. Then, the gray and cyan obstacles are overtaken on the left. We obtain $\bar{e} = 10$, $\Delta = 39$.
- Scenario C: the difference with respect to B is that the purple obstacle is rotated clockwise. Thus, the robot overtakes both this and the gray obstacle on the right. We obtain $\bar{e} = 8$, and $\Delta = 31$. The experiment shows the utility of ignoring lateral data in the grid: considering the green obstacle, would have made the robot curve farther from the path.
- Scenario D: this scenario is designed to test the controller in the presence of potential field local minima. In fact, when detected, the purple obstacle is centered on the z' axis and orthogonal to it; this induces $\alpha = 0$ and drives the robot towards the obstacle; however, after only 2 iterations, the position of the visual features misaligns the robot and the obstacle, which is thus avoided. The rest of the experiment is very similar to B, with $\bar{e} = 10$, and $\Delta = 39$.

The experiments are shown in the video attached to this paper. It is difficult to find a general interpretation of the results, since these are dependent on many factors (e.g., the obstacles position and orientation with respect to the taught path, and to the visual features position). However, in all cases, the robot is able to follow the visual path, without colliding. Since this is not equivalent to following the 3D path, some portions of the replayed path are far from the taught one. However, these errors correspond to the obstacle locations, and are indispensable to avoid collisions. Path following has been achieved in spite of occlusions, although a minimal number of visual features is obviously required. In two cases (scenarios A and C), navigation was possible using only 3 points. Since obstacle avoidance is defined as secondary task, it is realized in the best way without affecting visual navigation. This implies that in particularly constrained environments, the only way of avoiding collisions is to set $\hat{v} = 0$ in order to stop the robot.

In Scenario D, although the purple obstacle induces a potential which 'attracts' the robot to collision, the non-symmetric visual features distribution enabled the robot to exit the minimum. We have repeated this experiment with 10 randomly generated visual feature distributions, and in all cases the robot exited the minimum. The rare situation where both obstacle and visual data are symmetric could be tackled with standard escape methods [16].

In Fig. 4, we show 4 stages of scenario A. First, the visual features are driving the robot towards the gray obstacle. When detected, this obstacle forces the robot away from the path, while occluding many features. The camera rotates clockwise to maintain visibility. Finally, the controller drives the robot back to the path, and the camera to the forward-looking position. Note that at these iterations, in spite of having been detected, the obstacle is not considered, because of its non influential pose.

Further details can be obtained by studying some relevant variables. In Fig. 5, we have plotted activation function H, during the 4 experiments. The curves show when the obstacles have intervened in (13) (i.e., when $H > 0$): just the purple, gray and cyan obstacles in C (blue curve), and all 4 obstacles in the other cases. The purple and gray obstacles, which are the most difficult to circumnavigate, generate a long-lasting $H \neq 0$, whereas the green one is

[3]www.cyberbotics.com

quickly avoided. In most cases, there is a spike to $H = 1$ when an obstacle is sensed, since we related H to α, which leaps from 0 (in the safe context) to a value greater than β, as soon as the obstacle is detected. An exception is D (brown), where, because of the potential field minimum, H increases gradually. Since 3 obstacles have been detected only for C, we have shown, in Fig. 6, ω, e, φ and $\dot{\varphi}$ during navigation in this scenario. Since we have set $v = \hat{v}$ throughout replaying, the primary task error on $s_{1,2}$ is null, and plotting v is unnecessary. In the graphs, we have highlighted in yellow the iterations, with strong $H \neq 0$. At iteration 15, the purple obstacle is detected: the obstacle vector field induces an abrupt rotation on the robot (green in Fig. 6), and on the camera. The strategy proves efficient since the consequent image error increase is slight (approximately 30 pixels, black in the figure) and is reduced as the robot overtakes the obstacle. Both velocities are almost zeroed, until iteration 280, when the gray obstacle triggers a positive robot rotation and negative pan velocity. Finally, H is activated by the detection of the cyan obstacle. In this case, the robot angular velocity is negative, whereas the camera velocity is positive. From iteration 700 onwards, the activation function is canceled. Correspondingly, the variables are driven by (14). Note also that the camera angle (blue) is reset to 0 in less than 100 iterations, and remains null until the end of the experiment.

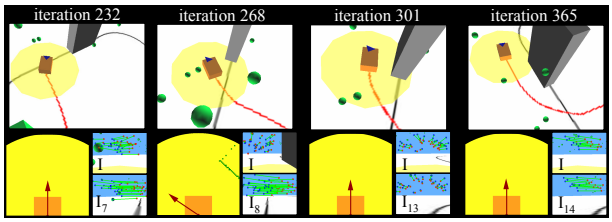

Fig. 4. The robot overtakes the gray obstacle in scenario A. For each iteration we show: the occupancy grid (bottom left), and current and next key image (bottom right).

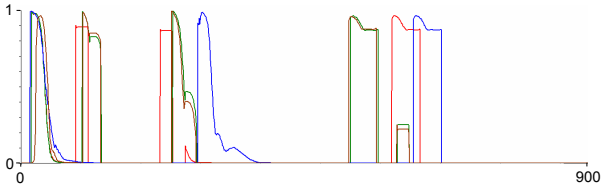

Fig. 5. Obstacle activation function H during simulations in the four scenarios A (red), B (green), C (blue) and D (brown).

VI. CONCLUSIONS

We have applied a redundancy-based approach to the problem of visual navigation with obstacle avoidance. Our approach is novel in appearance-based navigation: it merges techniques from potential fields, redundancy and visual servoing to tackle a sensor-based problem. Simulations are successful in various situations, and in future work, we plan to assess the controller on a real robot.

ACKNOWLEDGMENTS

This work was funded by the ANR CityVIP project. The authors thank G. Zecca and O. Kermorgant for their help.

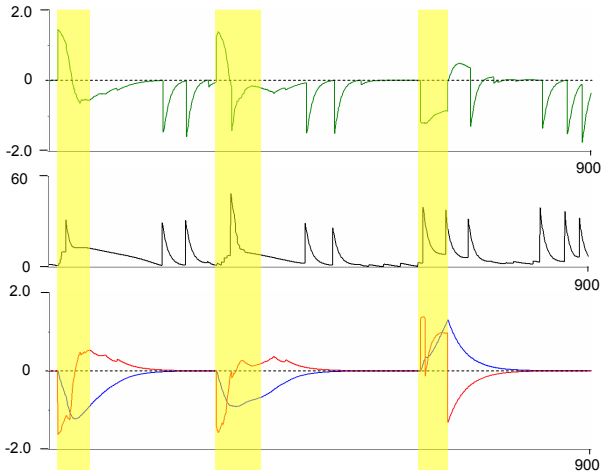

Fig. 6. Evolution of relevant variables during navigation in scenario C: ω (green, in rad s^{-1}), image error e (black, in pixels), φ (blue, in rad) and $\dot{\varphi}$ (red, in rad s^{-1}). The iterations with non-null H are highlighted in yellow.

REFERENCES

[1] 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.
[2] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots", *IEEE Int. Conf. on Robotics and Automation*, 1985.
[3] S. Quinlan and O. Khatib, "Elastic bands: connecting path planning and control", *IEEE Int. Conf. on Robotics and Automation*, 1993.
[4] L. Sciavicco and B. Siciliano, *Modeling and Control of Robot Manipulators*, Springer, 2000.
[5] A. Diosi, A. Remazeilles, S. Šegvić and F. Chaumette, "Outdoor Visual Path Following Experiments", *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2007.
[6] 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.
[7] F. Bonin-Font, A. Ortiz and G. Oliver, "Visual navigation for mobile robots: a survey", *Journal of Intelligent and Robotic Systems*, vol. 53, 2008, pp. 263–296.
[8] 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.
[9] S. Chiaverini, G. Oriolo and I. D. Walker, "Kinematically redundant manipulators", in *Springer Handbook of Robotics*, B. Siciliano, O. Khatib (Eds.), Springer, 2008, pp. 245–268.
[10] A. Liegeois, "Automatic supervisory control of configurations and behavior of multibody mechanisms", *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 7, no. 6, 1977, pp. 868-871.
[11] H. Hanafusa, T. Yoshikawa, Y. Nakamura, "Analysis and control of articulated robot arms with redundancy", *IFAC World Congress*, 1981.
[12] P. Baerlocher and R. Boulic, "An inverse kinematic architecture enforcing an arbitrary number of strict priority levels", *The Visual Computer*, vol. 6, no. 20, 2004, pp. 402-417.
[13] F. Chaumette and E. Marchand, "A redundancy-based iterative approach for avoiding joint limits: Application to visual servoing", *IEEE Trans. on Robotics and Automation*, vol. 17, no. 5, 2001, pp. 719-730.
[14] A. Elfes, "Using occupancy grids for mobile robot perception and navigation", *Computer*, vol. 22(6), 1989, pp. 46-57.
[15] A. De Luca and G. Oriolo, "Local Incremental Planning for Nonholonomic Mobile Robots", *IEEE Int. Conf. on Robotics and Automation*, 1994.
[16] J. C. Latombe, *Robot Motion Planning*, Kluwer Academic, 1991.