2008 10th Intl. Conf. on Control, Automation, Robotics and Vision
Hanoi, Vietnam, 17–20 December 2008

An Image-based Visual Servoing Scheme for Following Paths with Nonholonomic Mobile Robots

Andrea Cherubini François Chaumette
INRIA / IRISA
Campus de Beaulieu
35042 Rennes Cedex, France
{firstname.lastname}@irisa.fr

Giuseppe Oriolo
Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Ariosto 25, 00185 Roma, Italy
oriolo@dis.uniroma1.it

***Abstract*—We present an image-based visual servoing controller enabling nonholonomic mobile robots with a fixed pinhole camera to reach and follow a continuous path on the ground. The controller utilizes only a small set of features extracted from the image plane, without using the complete geometric representation of the path. A Lyapunov-based stability analysis is carried out. The performance of the controller is validated and compared by simulations and experiments on a car-like robot equipped with a pinhole camera.**

***Index Terms*—Visual servoing, nonholonomic mobile robots.**

I. Introduction

In many recent works, mobile robot navigation is done by processing visual information [1]. One of the prominent methods in this area is visual servoing [2], which was originally developed for manipulator arms [3]. In some cases, the vision system developed for navigation relies on the geometry of the environment and on other metrical information, for driving the vision processes and performing self-localization. In this case, *position-based* visual servoing techniques can be used to control the robot. The feedback law is computed by reducing errors in estimated pose space. Alternative systems use no explicit representation of the environment. In this case, *image-based* visual servoing techniques can be used to control the robot: an error signal measured directly in the image is mapped to actuator commands. The image-based approach eliminates the necessity for image interpretation, and errors due to camera modeling and calibration.

Applying such control techniques for navigation with wheeled mobile robots, involves well known control problems related to the nonholonomic constraint. Firstly, the linearization of these systems is uncontrollable, and, secondly, there do not exist smooth state feedback laws, for stabilizing these systems to an equilibrium. However, advanced visual servoing techniques have been successfully used to control nonholonomic mobile robots in [4], [5], [6], [7], and more recently in [8]. Since a complete survey of the vision-based nonholonomic control literature is impossible, we focus uniquely on some relevant works in the field of path following (PF).

In the PF task, the controller must drive some suitable *path error function*, indicating the position of the robot with respect to the path to a desired value (usually, zero, as in [9], [10]). Many articles have focused on the design of visual PF controllers, especially in the field of autonomous vehicle guidance [11], [12]. Most of these works address the problem of zeroing the lateral displacement and orientation error of the vehicle at a *lookahead* distance. However, these studies require a complete geometric representation of the path. Another position-based approach is presented in [13], where only straight line following is considered. On the other hand, image-based techniques have been used in [14], [15], and [16]. In [14], differential flatness properties are used to generate effective path following strategies. However, a fixed ceiling camera (instead of an on-board camera) is used to control the mobile robot. In [15], the PF problem is formulated by controlling the shape of the curve in the image plane. The practical implementation is, however, rather sophisticated, implying an extended Kalman filter to dynamically estimate the path curve derivatives up to order three.

In summary, most of these approaches impose constraints on the path shape, curvature, and initial configuration. Besides, they rely on accurate online extraction of the path shape. The main contribution of this work is that the proposed control scheme requires only two visible path features, along with a coarse camera model, and that, under certain conditions, it guarantees convergence when the initial error is large. To our knowledge, this is also the first work, where a stability analysis of nonholonomic image-based path following is carried out. In fact, local asymptotic stability of an equilibrium state has been proved only for a circular path in [16].

The paper is organized as follows. In Sect. II, the PF problem is defined along with all the variables used in our method. In Sect. III, we illustrate our control design. In Sect. IV, a Lyapunov-based stability analysis, taking into account the robot kinematics, is carried out. The experimental setup, simulated and experimental results are presented respectively in Sect. V, VI, and VII.

II. Problem definition

In this work, we focus on the path following task for nonholonomic mobile robots equipped with a fixed pinhole camera. The workspace where the robot moves is planar: $\mathcal{W} = I\!R^2$. The *path* p to be followed is represented by a continuous curve in $\mathcal{W}$. A *following direction* is associated to the path (see Fig. 1(a)). We name r the point on the robot sagittal plane that should track the path. With reference to Fig. 1(a), let us define the reference frames: world frame

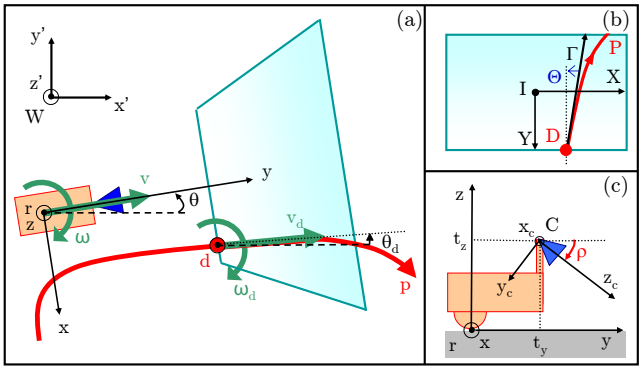

Fig. 1. Variables used in this work. The task for the robot (orange), equipped with a fixed pinhole camera (blue) is to follow the red path, p. The camera field of view and its projection on the ground are represented in cyan. (a) Top view: frames $\mathcal{F}_\mathcal{W}$ and $\mathcal{F}_\mathcal{R}$, applied (v, ω) and desired (v_d, ω_d) control. (b) Image plane view: frame $\mathcal{F}_\mathcal{I}$ and image variables. (c) Side view: frames $\mathcal{F}_\mathcal{C}$ and $\mathcal{F}_\mathcal{R}$, optical center position in $\mathcal{F}_\mathcal{R}$ and camera tilt offset ρ.

$\mathcal{F}_\mathcal{W}\,(W, x', y', z')$, and robot frame $\mathcal{F}_\mathcal{R}\,(r, x, y, z)$. We assume that in $\mathcal{F}_\mathcal{W}$, the path curve can be expressed by a twice differentiable function. The robot state coordinates (i.e., the robot *generalized coordinates*) are $q = [x'\ y'\ \theta]^T$, where $[x'\ y']^T$ represent the Cartesian position of r in $\mathcal{F}_\mathcal{W}$, and $\theta \in\ ]-\pi, +\pi]$ is the orientation of the robot frame y axis with respect to the world frame x' axis (positive counterclockwise).

We choose $u = [v\ \omega]^T$ as the pair of control variables for our system; these represent respectively the linear and angular velocities (positive counterclockwise) of the robot. Point r is chosen as the projection on the ground of the wheel center in the case of a unicycle robot, and of the rear axis center in the case of a car-like robot. In some cases, the robot kinematic constraints can impose a bound c_M on the instantaneous applicable curvature:

$$\left|\frac{\omega}{v}\right| < c_M \tag{1}$$

The maximum applicable curvature c_M can be interpreted as the inverse of the radius of the narrowest curve that the robot can track. In the case of a unicycle robot, there is no such bound. Instead, for a car-like robot, the curvature bound is imposed by the steering angle constraint.

We denote with W and H respectively the image width and height, in pixels. The camera optical axis has a constant tilt offset $\rho \in\ ]\text{ATAN}\frac{H}{2}, \frac{\pi}{2}[$ with respect to the y axis[1], and the optical center C is positioned in the robot frame at $[x\ y\ z]^T = [0\ t_y\ t_z]^T$(see Fig. 1). A pinhole camera model is considered. In the rest of the paper, we shall also utilize the camera frame $\mathcal{F}_\mathcal{C}(C, x_c, y_c, z_c)$, and image frame $\mathcal{F}_\mathcal{I}(I, X, Y)$ (I is the image plane center) (see Fig. 1).

In our PF scheme, we utilize only two path features extracted from the image plane: the position of a path point, and the path tangent orientation at that point (see Fig. 1(b)). Under the assumption that a portion of the path is always visible, we use the features of the *first* (considering the path direction) visible path point d, projected to $D = [X\ Y]^T$ on the image plane (see Figures 1(a) and 1(b)). We also note: P the projection of p on the image plane, Γ the oriented (according to the path direction) tangent of P at D, and $\Theta \in\ ]-\pi, \pi]$ the angular offset from Γ to the $-Y$ axis (positive counterclockwise). [2]. In this work, the PF task consists of driving D to the bottom pixel row of the image plane with $X = \Theta = 0$, as shown in Fig. 2(g). In order to achieve this task with general initial conditions (in particular, when D is not initially on the bottom pixel row), we designed a switching control scheme that will be described in the next section.

[1]The assumptions: $\rho \in\ ]\text{ATAN}\frac{H}{2}, \frac{\pi}{2}[$, and planar workspace, guarantee that the retroperspective projection of all image points is on the ground, since $\text{ATAN}\frac{H}{2}$ is the camera vertical semi-angle of view.

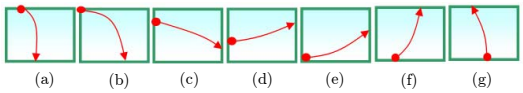

Fig. 2. Seven possible configurations of P in the image (D is represented by the red circle).

III. Control Design

Similarly to [9], [13], and [16], in our control scheme, we select an arbitrary constant forward linear velocity ($v = \text{const} = v_d > 0$), and we apply a nonlinear feedback on ω. The feedback law is based on the two visible features: D and Θ, and the task is to drive the two features to the configuration shown in Fig. 2(g) (i.e., $X = \Theta = 0$).

Depending on the position of D in the image, the PF controller will switch between two primitive controllers: a *row controller*, and a *column controller*. In both primitive controllers, the task is to drive the path features to a desired intermediate configuration, while D is constrained to a line in the image plane: a row of pixels ($Y = \text{const}$) in the first case, and a column of pixels ($X = \text{const}$) in the second case. The desired intermediate configurations are defined as $(\hat{X}, \hat{\Theta})$ for the row controller, and $(\hat{Y}, \hat{\Theta})$ for the column controller. The control scheme utilizes the two primitive controllers in general initial conditions, based on a switching mechanism. Consider the initial configuration shown in Fig. 2(a), with D on the top row of the image plane. Initially, the row controller must be used to drive D to the desired configuration, corresponding to a lateral pixel column of the image plane (e.g., the left column, as in Fig. 2(b)). Afterwards, the column controller will be used to drive D along the left column of the image to the desired configuration, corresponding to the bottom left corner (Fig. 2(c), 2(d) and 2(e)). Then, the row controller will be used to drive D along the bottom row of the image plane to the desired configuration $\hat{X} = \hat{\Theta} = 0$ (Fig. 2(g)). When this configuration is reached, and the robot is following the path, the components of the tracking control u_d must be compliant with the path curvature at d in $\mathcal{F}_\mathcal{R}$, noted c_d[3] : $\omega_d = c_d v_d$.

In the remainder of this section, we shall first present the relationships between the visual features and the robot velocity, and then the two primitive controllers (row and column) developed in this work.

[2]Γ and Θ are always defined, since we have assumed that the path curve can be expressed by a twice differentiable function in $\mathcal{F}_\mathcal{W}$ and this property is preserved in $\mathcal{F}_\mathcal{I}$.

[3]Note that c_d is always defined, since we have assumed that the path curve can be expressed by a twice differentiable function in $\mathcal{F}_\mathcal{W}$, and this property is preserved in $\mathcal{F}_\mathcal{R}$.

$$L_s = \begin{bmatrix} \frac{-\sin\rho - Y\cos\rho}{t_z} & 0 & \frac{X(\sin\rho + Y\cos\rho)}{t_z} & XY & -1-X^2 & Y \\ 0 & \frac{-\sin\rho - Y\cos\rho}{t_z} & \frac{Y(\sin\rho + Y\cos\rho)}{t_z} & 1+Y^2 & -XY & -X \\ \frac{\cos\rho\cos^2\Theta}{t_z} & \frac{\cos\rho\cos\Theta\sin\Theta}{t_z} & -\frac{\cos\rho\cos\Theta(Y\sin\Theta + X\cos\Theta)}{t_z} & -(Y\sin\Theta + X\cos\Theta)\cos\Theta & -(Y\sin\Theta + X\cos\Theta)\sin\Theta & -1 \end{bmatrix} \quad (2)$$

A. Relating visual features and robot velocity

Let us denote $u_c = [v_{c,x}\ v_{c,y}\ v_{c,z}\ \omega_{c,x}\ \omega_{c,y}\ \omega_{c,z}]^T$ the robot velocity expressed in $\mathcal{F}_\mathcal{C}$. The interaction matrix $L_s(X, Y, \Theta)$ relates the path image features with u_c:

$$\left[\dot{X}\ \dot{Y}\ \dot{\Theta}\right]^T = L_s(X, Y, \Theta)\, u_c$$

The expression of $L_s(X, Y, \Theta)$ for the normalized perspective camera model is shown at the top of this page. In the following, we will note L_X, L_Y and L_Θ the lines of L_s (respectively, top to bottom). The robot velocity in $\mathcal{F}_\mathcal{C}$ can be expressed in function of $u = [v\ \omega]^T$ by using the homogeneous transformation from $\mathcal{F}_\mathcal{R}$ to $\mathcal{F}_\mathcal{C}$:

$$u_c = {}^C T_R u$$

with:

$${}^C T_R = \begin{bmatrix} 0 & -t_y \\ -\sin\rho & 0 \\ \cos\rho & 0 \\ 0 & 0 \\ 0 & -\cos\rho \\ 0 & -\sin\rho \end{bmatrix}$$

In the following, we will note T_v and T_ω respectively the first and second columns of ${}^C T_R$.

B. Row controller

The task of the row controller is to drive (X, Θ) to a desired set point $(\hat{X}, \hat{\Theta})$ under constraint $Y = \text{const} = Y^*$ (i.e., D is constrained to a pixel row in the image plane). Since $\dot{Y} = 0$, the system state equations are:

$$\begin{bmatrix} \dot{X} \\ \dot{\Theta} \end{bmatrix} = A_r v + B_r \omega \quad (3)$$

where:

$$A_r = \begin{bmatrix} L_X \\ L_\Theta \end{bmatrix} T_v \qquad B_r = \begin{bmatrix} L_X \\ L_\Theta \end{bmatrix} T_\omega$$

When $B_r \neq 0$, we select as control law:

$$\omega = -B_r^+ \left(\begin{bmatrix} \lambda_X e_X \\ \lambda_\Theta e_\Theta \end{bmatrix} + A_r v_d \right) \quad (4)$$

where $e_X = X - \hat{X}$ and $e_\Theta = \Theta - \hat{\Theta}$ are the state errors defined in the image, and λ_X, λ_Θ are positive gains.

C. Column controller

The task of the column controller is to drive (Y, Θ) to a desired set point $(\hat{Y}, \hat{\Theta})$ under constraint $X = \text{const} = X^*$ (i.e., D is constrained to a pixel column in the image plane). Since $\dot{X} = 0$, the system state equations are:

$$\begin{bmatrix} \dot{Y} \\ \dot{\Theta} \end{bmatrix} = A_c v + B_c \omega \quad (5)$$

primitive controller	row	column
$\mathcal{X}_1$	X	Y
$\mathcal{X}_2$	Θ	Θ
$\mathcal{A}_1$	$L_X T_v$	$L_Y T_v$
$\mathcal{A}_2$	$L_\Theta T_v$	$L_\Theta T_v$
$\mathcal{B}_1$	$L_X T_\omega$	$L_Y T_\omega$
$\mathcal{B}_2$	$L_\Theta T_\omega$	$L_\Theta T_\omega$
$\mathcal{G}_1$	λ_X	λ_Y
$\mathcal{G}_2$	λ_Θ	λ_Θ
$\mathcal{E}_1$	$X - \hat{X}$	$Y - \hat{Y}$
$\mathcal{E}_2$	$\Theta - \hat{\Theta}$	$\Theta - \hat{\Theta}$

TABLE I
COMPONENTS OF: $\mathcal{X}$, $\mathcal{A}$, $\mathcal{B}$, $\mathcal{G}$, AND $\mathcal{E}$ FOR THE TWO CONTROLLERS

where:

$$A_c = \begin{bmatrix} L_Y \\ L_\Theta \end{bmatrix} T_v \qquad B_c = \begin{bmatrix} L_Y \\ L_\Theta \end{bmatrix} T_\omega$$

When $B_c \neq 0$, we select as control law:

$$\omega = -B_c^+ \left(\begin{bmatrix} \lambda_Y e_Y \\ \lambda_\Theta e_\Theta \end{bmatrix} + A_c v_d \right) \quad (6)$$

where $e_Y = Y - \hat{Y}$ and $e_\Theta = \Theta - \hat{\Theta}$ are the state errors defined in the image plane, and λ_Y, λ_Θ are given positive gains.

IV. STABILITY ANALYSIS

The stability analysis has been carried out for the proposed control scheme by using a Lyapunov-based approach. The camera model is assumed to be known (extending the stability analysis to the case of camera modeling error is out of the scope of this paper). Note that the two state equations (3) and (5) can be generally written:

$$\dot{\mathcal{X}} = \mathcal{A} v + \mathcal{B} \omega \quad (7)$$

and, similarly, when $\mathcal{B} \neq 0$, [4] the two control laws (4) and (6) can be generally expressed as:

$$\omega = -\mathcal{B}^+ (\mathcal{G}\mathcal{E} + \mathcal{A} v_d) \quad (8)$$

where $\mathcal{X} = [\mathcal{X}_1\ \mathcal{X}_2]^T$, $\mathcal{A} = [\mathcal{A}_1\ \mathcal{A}_2]^T$, $\mathcal{B} = [\mathcal{B}_1\ \mathcal{B}_2]^T$, and $\mathcal{E} = [\mathcal{E}_1\ \mathcal{E}_2]^T$ are two-dimensional column vectors, and:

$$\mathcal{G} = \begin{bmatrix} \mathcal{G}_1 & 0 \\ 0 & \mathcal{G}_2 \end{bmatrix}$$

The components of $\mathcal{X}$, $\mathcal{A}$, $\mathcal{B}$, $\mathcal{G}$ and $\mathcal{E}$ for the two controllers are recalled in Table I.

Hence, the following stability analysis is valid for both primitive controllers. Let us consider the quadratic Lyapunov function candidate:

$$\mathcal{V} = \frac{|\mathcal{E}|^2}{2}$$

[4] We don't manage singularity $\mathcal{B} = 0$, since it is extremely unlikely to occur in practice.

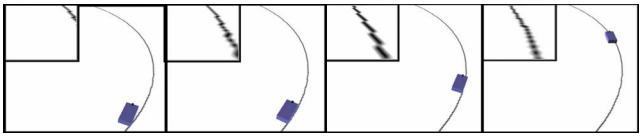

Fig. 3. Robot positions and corresponding processed images at consecutive time frames during PF simulation.

Taking the time derivative of this function along a solution of the closed-loop system gives:

$$\dot{\mathcal{V}} = \mathcal{E}^T \dot{\mathcal{X}}$$

Using (7) and (8) leads to:

$$\dot{\mathcal{V}} = \mathcal{E}^T \left(\mathcal{A} v_d - \mathcal{B}\mathcal{B}^+ \left(\mathcal{G}\mathcal{E} + \mathcal{A} v_d \right) \right)$$

If we set $\mathcal{G}_1 = \mathcal{G}_2 = \mathcal{G}^* > 0$, since $v_d > 0$, $\dot{\mathcal{V}}$ is negative semidefinite if and only if:

$$\frac{\mathcal{E}^T \left(\mathcal{A} - \mathcal{B}\mathcal{B}^+ \mathcal{A} \right)}{\mathcal{E}^T \mathcal{B}\mathcal{B}^+ \mathcal{E}} < \frac{\mathcal{G}^*}{v_d} \tag{9}$$

To verify the Lyapunov sufficient condition (9), the robot kinematic constraint on c_M must be analyzed, since it imposes a constraint on the maximum applicable gain $\mathcal{G}^*$. In fact, replacing (8) in (1), gives:

$$-c_M + \mathcal{B}^+ \mathcal{A} < -\frac{\mathcal{G}^*}{v_d} \mathcal{B}^T \mathcal{E} < c_M + \mathcal{B}^+ \mathcal{A}$$

From this equation we derive a sufficient condition for (9):

$$\left| \frac{\mathcal{E}^T \left(\mathcal{A} - \mathcal{B}\mathcal{B}^+ \mathcal{A} \right)}{\mathcal{E}^T \mathcal{B}} + \mathcal{B}^+ \mathcal{A} \right| < c_M \tag{10}$$

In (10) we have expressed a sufficient condition for asymptotic stability as a condition on the maximum applicable curvature c_M, hence on the robot kinematic model. Note that, for a unicycle robot, where the curvature is not bounded, asymptotic stability is guaranteed. Condition (10) will be verified numerically, depending on the robot parameters (which determine the values of $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{B}^+$) and on the desired states (which determine the values of $\mathcal{E}$), as will be shown in the next section for the experimental setup used in this work.

V. Experimental Setup

In the following sections, we report the simulated and real experimental results obtained by applying the proposed PF control scheme. All experiments have been carried out with a CyCab. CyCabs are 4 wheel drive, 4 wheel steered intelligent vehicles designed to carry two passengers. In our CyCab, all computations except the low-level control have been executed on a laptop with a 2 GHz Centrino processor. A 70° field of view, forward looking, B&W Marlin F-131B camera is mounted on the robot. The robot is used in car-like mode (i.e., only the front wheels are used for steering), and the camera is used in auto shutter mode, with image resolution 320×240 pixels. The maximum curvature constraint (1) must be considered. In particular, for a car-like robot, it is:

$$c_M = \frac{\tan \phi_M}{L}$$

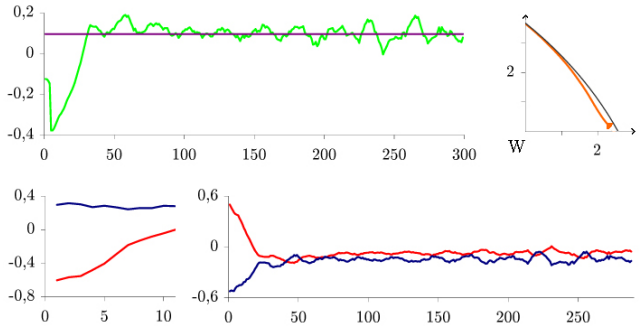

Fig. 4. Evolution of relevant variables during PF simulation. Top left: steering angles ϕ_d (purple), and ϕ (green) in rad. Top right: robot position (orange) and path (black) in $\mathcal{F}_\mathcal{W}$ during the first 150 iterations. Bottom: state errors $\mathcal{E}_1^*$ (red, non-dimensional) and $\mathcal{E}_2$ (blue, in rad) for the column controller (left) and row controller (right).

where ϕ_M is the robot maximum applicable steering angle, and L is the distance between the front and rear wheel axes. For CyCab, $\phi_M = 0.40$ rad and $L = 1.21$ m; thus, $c_M = 0.30$ m^{-1}. The system has been coarsely calibrated, to obtain: $\rho = 0.55$ rad, $t_y = 0.55$ m and $t_z = 1.65$ m.

The steering angle ϕ used to control CyCab is derived from angular speed ω (calculated using either (4), or (6)):

$$\phi = \text{ATAN} \frac{L\omega}{v_d}$$

We expect that, when the robot is on the path, the steering angle tracks the value:

$$\phi_d = \text{ATAN} L c_d \tag{11}$$

In all the experiments[5], we set $v_d = 0.2$ ms $^{-1}$.

VI. Simulations

Preliminary testing has been carried out in the Cyberbotics Webots simulator[6], where we have drawn a circular path of radius 12.5 m (i.e., $c_d = \text{const} = 0.08$ m^{-1}, and correspondingly, from (11), $\phi_d = 0.096$ rad). Our proposed control scheme has been simulated starting from various initial configurations.

Consider the initial configuration with the path intersecting the right pixel column in the image (see Fig. 3, left). A switching strategy combining the column and row controllers is used. Initially, the column controller (6) is used to drive D along the right pixel column of the image to the bottom right corner. We use $\mathcal{G}^* = 0.5$. Afterwards, the row controller (4) is used to drive D along the bottom row of the image plane to: $\hat{X} = \hat{\Theta} = 0$. An adaptive gain is used. This design choice provides fast convergence for large error values, while avoiding overshoot for small errors. We use: $\mathcal{G}^* = 3 \exp^{-10||\mathcal{E}||}$ with $||\mathcal{E}|| = \sqrt{e_X^2 + e_\Theta^2}$ the error norm. With these gain values, the simulated robot is able to successfully follow the path, without abrupt steering changes at the switching point between the two controllers. The robot positions and processed images at consecutive time frames are shown in Fig. 3. The evolution of relevant variables (steering angle, robot position, and state errors) is plotted in Fig. 4. Instead of state errors $\mathcal{E}_1 = e_X$

[5] Videos of the simulations and experiments are on the web site: www.irisa.fr/lagadic/demo/demo-cycab-path-following/cycab-path-following

[6] www.cyberbotics.com

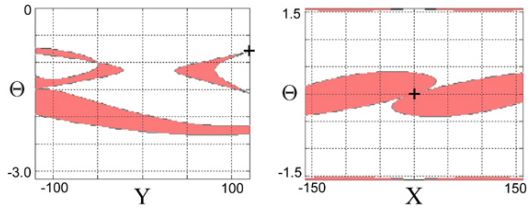

Fig. 5. Stability loci of the state variables (X, Y in pixels, Θ in rad) that verify the Lyapunov sufficient asymptotic stability condition (pink) for: right column controller (left), and bottom row controller (right). The desired states are indicated with the black cross.

and $\mathcal{E}_1 = e_Y$, we have respectively plotted the scaled values $\mathcal{E}_1^* = \frac{e_X}{W}$ for the row controller, and $\mathcal{E}_1^* = \frac{e_Y}{H}$ for the column controller. Note that the steering angle ϕ, as well as the state errors $\mathcal{E}_1^*$ and $\mathcal{E}_2$, slightly oscillate during the simulation. This occurs because, although 3D data can be exactly derived in Webots, the features utilized by the image-based scheme are biased, since we have decided to visually extract them in the simulator. Note also that at the end of the first phase (after the column controllers have been applied) the errors on the tangent orientation $\mathcal{E}_2$ have not reached 0. This is due to the switching condition, which is imposed only by the error on the point position $\mathcal{E}_1$. Nevertheless, when the row controller is applied, the tracking errors converge, and the mean value of the steering angle at steady state coincides, as expected, with ϕ_d. Since Webots provides the robot with an absolute localization system, we have plotted the robot position in the top right graph of Fig. 4. This graph confirms the effectiveness of our path following scheme.

VII. Experiments

After the Webots simulations, the proposed control scheme has been tested on the real CyCab in a series of outdoor experiments. The path used in the experiments is composed of two straight lines of length 6 m joined by a 60° arc of circle of radius 10 m (i.e., $\phi_d = \pm 0.12$ rad, with the sign of ϕ_d depending on the path direction to be followed by the robot). The path features are derived by tracking straight lines and arcs of parabola with the ViSP software [17]. The tracker must be initialized by clicking on five path points oriented in the desired path direction.

In order to verify the robustness of the controller, the experiments have been repeated by considering a random calibration error on the camera parameters. For the calibrated camera experiments, we have numerically verified the system sufficient stability condition (10) as the system state variables evolve. The state loci that verify condition (10) are represented in Fig. 5. In the proposed experiments, two instances of the primitive controllers are used:

- right column controller,
- bottom row controller.

Hence, in Fig. 5, the state loci are represented for each of these controllers: right column (left in the figure), and bottom row (right). The desired state values are also indicated in the figure for each controller. Note that in both cases, the desired state values belong to the loci where the asymptotic stability condition is verified. The loci of Fig. 5 will be used to verify the asymptotic stability condition during the experiments.

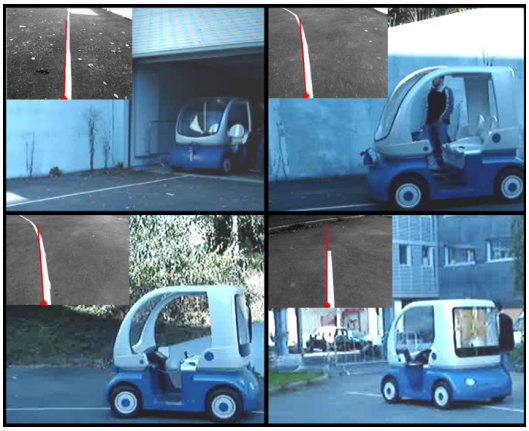

Fig. 6. First experiment: Cycab is initially positioned on the path.

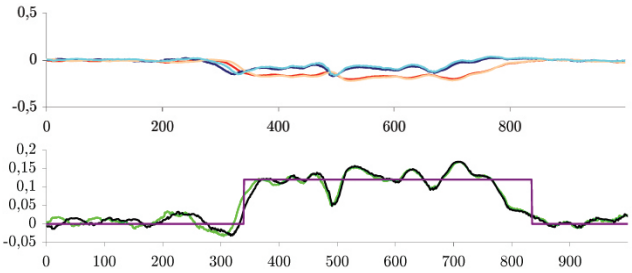

Fig. 7. Evolution of relevant variables during the first experiment. Top: errors $\mathcal{E}_1^*$ (non-dimensional), and $\mathcal{E}_2$ (in rad) are plotted (red and blue: correct camera calibration, pink and cyan: coarse calibration). Bottom: ϕ_d (purple) and ϕ (green: correct camera calibration, black: coarse calibration) in rad.

In a first experiment, Cycab is initially positioned on the path with the correct orientation and small initial error: D is on the bottom pixel row of the image plane (see Fig. 6, top left). The row controller (4) is used to drive the states to $\hat{X} = \hat{\Theta} = 0$, with: $\mathcal{G}^* = 0.18 \exp^{-30||\mathcal{E}||} + 0.02$. The robot positions and processed images at consecutive time frames while Cycab follows the path are shown in Fig. 6. The evolution of the relevant variables during the experiment is shown in Fig. 7. Instead of e_X, we have plotted the scaled value $\mathcal{E}_1^* = \frac{e_X}{W}$. The robot is able to successfully follow the path, and the tracking errors are low throughout the experiment. At the end of the experiment, both errors are below 0.03. Both errors increase when the robot reaches the discontinuity in the path curvature (frame 335). Correspondingly, ϕ increases in order to compensate for the error and enables CyCab to follow the curve. Using the right locus in Fig. 5, we verify that throughout the experiment, the state variables verify the asymptotic stability condition.

In the second experiment (shown in Fig. 8), CyCab is initially near the path, with D on the right pixel column of the image plane. A switching strategy, combining both controllers (6) and (4), is used. Initially (phase 1), the column controller (6) is used is used to drive D along the right pixel column of the image to the bottom right corner. We use: $\mathcal{G}^* = 0.98 \exp^{-3.6||\mathcal{E}||} + 0.05$. Then (phase 2), the row controller (4) is used with: $\mathcal{G}^* = 0.18 \exp^{-30||\mathcal{E}||} + 0.02$ (as in the first experiment), to drive D along the bottom row of the image plane to the desired states $\hat{X} = \hat{\Theta} = 0$. The state

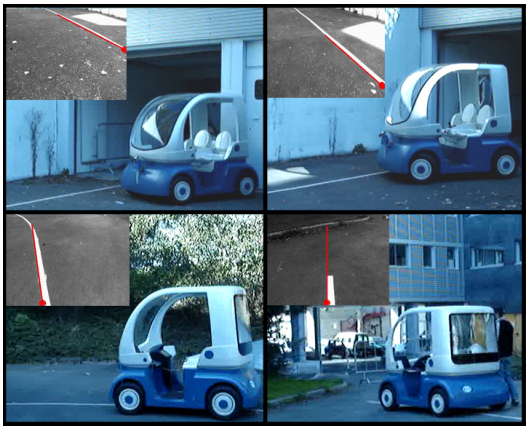

Fig. 8. Second experiment: D is initially on the right image column.

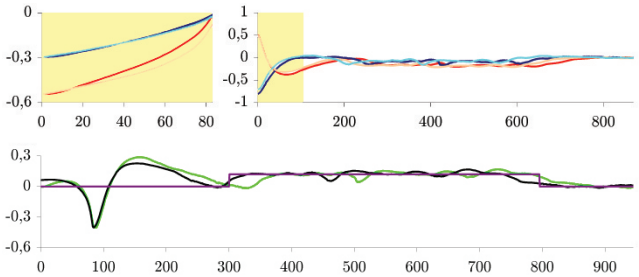

Fig. 9. Evolution of relevant variables during the second experiment. Top: errors during phases 1 (left) and 2 (right). $\mathcal{E}_1^*$ (non-dimensional) and $\mathcal{E}_2$ (in rad) are plotted (red and blue: correct camera calibration, pink and cyan: coarse calibration). The iteration steps with state variables not verifying the asymptotic stability condition are highlighted in yellow. Bottom: ϕ_d (purple) and ϕ (green: correct camera calibration, black: coarse calibration) in rad.

errors are plotted in the top of Fig. 9, for phases 1 (left) and 2 (right). Instead of e_X and e_Y, we have respectively plotted the scaled values $\mathcal{E}_1^* = \frac{e_X}{W}$ for phase 1, and $\mathcal{E}_1^* = \frac{e_Y}{H}$ for phase 2. The robot is able to successfully follow the path, and the tracking errors converge during both phases. Note also that there is no abrupt change of the steering angle at the switching iteration. At the end of the experiment both state errors are zeroed. The iteration steps with state variables not verifying the asymptotic stability condition (i.e., values of $\mathcal{X}$ outside the loci of Fig. 5) are highlighted in yellow in Fig. 9. The plots show that, throughout phase 1 and during the beginning of phase 2, condition (10) is not verified. Nevertheless, the system is able to converge.

The two image-based experiments have been repeated, by considering a random calibration error of either $+10\%$ or -10% on ρ, t_y, t_z, as well as on the camera focal length. The evolution of the relevant variables in the coarse calibration experiments is also shown in Fig. 7 and 9 (pink and cyan for the errors, black for ϕ), for comparison with the calibrated camera experiments. The robot is able to successfully follow the path in both cases. The convergence rate is slightly lower than in the calibrated camera experiments.

VIII. Conclusions

In this paper, we presented an image-based visual servoing control scheme enabling nonholonomic mobile robots with a fixed pinhole camera to reach and follow a continuous path on the ground. The controller utilizes only two path features extracted from the image plane, without using the complete geometric representation of the path. The features are: the position of a path point, and the path tangent orientation at that point. Although the disadvantage in using only two features is greater noise sensitivity, less computational resources are required. The main contributions of our work are that the approach can be used in general initial conditions, thanks to a switching strategy between two primitive controllers, and that a Lyapunov-based stability analysis has been carried out. The controller performance has been experimentally validated on a car-like robot equipped with a pinhole camera starting from two different initial conditions. In both experiments, a smooth control input ϕ is needed to achieve the task. The scheme robustness was also verified, by adding camera model errors in all the experiments. Future work might consist in testing the controller on a slow time-varying target.

References

[1] G. N. De Souza and A. C. Kak, "Vision for Mobile Robot Navigation: a Survey", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, 2002.

[2] 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.

[3] B. Espiau, F. Chaumette and P. Rives, "A new approach to visual servoing in robotics", *IEEE Trans. on Robotics and Automation*, vol. 8, no. 3, pp. 313–326, 1992.

[4] D. P. Tsakiris, P. Rives, and C. Samson, "Applying visual servoing techniques to control nonholonomic mobile robots", *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 1997.

[5] K. Hashimoto, and T. Noritsugu, "Visual servoing of nonholonomic cart", *IEEE Int. Conf. on Robotics and Automation*, 1999.

[6] H. H. Abdelkader, Y. Mezouar, N. Andreff, P. Martinet, "Omnidirectional Visual Servoing From Polar Lines", *IEEE Int. Conf. on Robotics and Automation*, 2006.

[7] M. Maya-Mendez, P. Morin, C. Samson, "Control of a Nonholonomic Mobile Robot Via Sensor-based Target Tracking and Pose Estimation", *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2006.

[8] 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, pp. 87–100, 2007.

[9] C. Canudas de Wit, B. Siciliano and G. Bastin Eds., *Theory of Robot Control*, Communication and Control Engineering, Springer-Verlag, 1996.

[10] F. Diaz del Rio, G. Jimenez, J. L. Sevillano, S. Vicente and A. Civit Balcells, "A generalization of path following for mobile robots", *IEEE Int. Conf. on Robotics and Automation*, 1999.

[11] M. Bertozzi, A. Broggi and A. Fascioli, "Vision-based intelligent vehicles: state of the art and perspectives", *Robotics and Automation Systems*, vol. 32, pp. 1–16, 2000.

[12] E. Royer, J. Bom, M. Dhome, B. Thuilot, M. Lhuillier and F. Marmoiton, "Outdoor autonomous navigation using monocular vision", *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2005.

[13] S. Skaff, G. Kantor, D. Maiwand and A. Rizzi, "Inertial navigation and visual line following for a dynamical hexapod robot", *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 2, pp. 1808–1813, 2003.

[14] R. Rao, V. Kumar and C. Taylor, "Visual servoing of a UGV from a UAV using differential flatness", *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 1, pp. 743–748, 2003.

[15] Y. Ma, J. Kosecka and S. Sastry, "Vision-guided navigation for a nonholonomic mobile robot", *IEEE Trans. on Robotics and Automation*, vol. 15, no. 3, pp. 521–536, 1999.

[16] J. B. Coulaud, G. Campion, G. Bastin and M. De Wan, "Stability analysis of a vision-based control design for an autonomous mobile robot", *IEEE Trans. on Robotics*, 2006.

[17] E. Marchand, F. Spindler, and F. Chaumette, "ViSP for visual servoing: a generic software platform with a wide class of robot control skills", *IEEE Robotics and Automation Magazine*, vol. 12, no. 4, 2005.