Non-Central Catadioptric Cameras Visual Servoing for Mobile Robots using a Radial Camera Model

Omar Tahri and Helder Araujo

Abstract— **Catadioptric cameras combine conventional cameras and mirrors to create omnidirectional sensors providing 360^o panoramic views of a scene. Modeling such cameras has been subject of significant research interest in the computer vision community leading to a deeper understanding of the image properties and also to different models for different types of configurations. Visual servoing applications using catadioptric cameras have essentially been using central cameras and the corresponding unified projection model. So far only in very few cases more general models have been used. In this paper we address the problem of visual servoing using the so-called the radial model. The radial model can be applied to many camera configurations and in particular to non-central catadioptric systems with mirror shapes that are symmetric around the optical axis. In this case we show that the radial model can be used with a non-central catadioptric camera to allow effective image-based visual servoing (IBVS) of a mobile robot.**

Using this model, which is valid for a large set of catadioptric cameras, new visual features are proposed to control the degrees of freedom of a mobile robot moving on a plane. Several simulation results are provided to validate the effectiveness of such features.

I. INTRODUCTION

In order to overcome the problem of keeping the features in the camera field of view (FOV), several methods have been developed namely: based on path planning [15], zoom adjustment [4], switching control [6]. More simple and different approaches consist on using omnidirectional vision sensors to increase the FOV using mirrors. Wide-angle cameras include catadioptric systems that combine mirrors and conventional cameras to create omnidirectional images providing 360^o panoramic views of a scene, or dioptric fish-eye lenses [3], [8]. Lately they have been subject of an increasing interest from robotics researchers [10], [13], [7], [20], [17].

In practice, it is advantageous that omnidirectional imaging systems have a single viewpoint [3], [18]. In those systems there exists a single center of projection, so that every pixel in the image measures the irradiance of the light passing through the same viewpoint. Central imaging systems can be modeled using two consecutive projections: spherical and then perspective. This geometric formulation, called the *unified model*, was proposed by Geyer and Daniilidis in [9] and include the classical perspective projection model. Many works in visual servoing are based on this model. In [20], an IBVS using invariants to rotational motion to control the translational degrees of freedom (DOFs) is proposed. In [10], the authors consider the problem of controlling a $6DOFs$ holonomic robot and a non-holonomic mobile robot from the projection of 3-D straight lines in the image plane on central catadioptric systems. Mariottini *et al* in [13] propose an IBVS based on the auto-epipolar condition, which occurs when two catadioptric views (current and desired) undergo a pure translation. The proposed approach is applicable to the autonomous navigation of a mobile holonomic robot.

Unfortunately, only few configurations lead to a single-viewpoint catadioptric system [3]. Obtaining the forward projection model for the general case of a non-central catadioptric camera is still an open problem in computer vision. This is considered a hard problem and iterative solutions are usually employed to determine the reflection point on the mirror. Recently, a forward projection model has been proposed for the case of non-central catadioptric cameras consisting on a perspective camera and a rotationally symmetric conic reflector [1]. In the latter work, the optical path from a given 3D point to the given viewpoint is obtained by solving a $6th$ degree polynomial equation for general conic mirrors. For a spherical mirror, the forward projection equation reduces to a $4th$ degree polynomial, resulting in a closed form solution. In [2], an analytical forward projection equation for the projection of a 3D point reflected by a quadric mirror into the image plane of a perspective camera, with no restrictions on the camera placement is derived. They show that the equation is a $8th$ degree polynomial in a single unknown. In absence of an analytical and simple forward model, the determination of some elements like the interaction matrix required for image-based servoing becomes difficult.

For scene reconstruction or control purposes, a complete knowledge of of the projection model is not always required. In [22], a technique to linearly estimate the radial distortion of a wide-angle lens given three views of a real-world plane has been proposed using the radial projection model. Based on [22], linear methods for the estimation of multi-view geometry of 1D radial cameras have been studied in [17] and [21]. In this paper, it will be shown that the simple radial projection model can be sufficient for mobile robot control using a large family of catadioptric cameras. More precisely, the contributions of this paper are:

- An image-based visual servoing method for mobile robots moving on a plane, valid for a large set of

Omar Tahri and Helder Araujo are with Institute for Systems and Robotics, Polo II 3030-290 Coimbra, Portugal, omartahri@isr.uc.pt, helder@isr.uc.pt. The authors would like to thank the support of project Morfeu–PTDC/EEA-CRO/108348/2008 funded by the Portuguese Science Foundation (FCT) and H. Araujo would like to thank the support of project FCT/PTDC/EIA-EIA/122454/2010, funded also by the Portuguese Science Foundation (FCT) by means of national funds (PIDDAC) and co-funded by the European Fund for Regional Development (FEDER) through COMPETE Operational Programme Competitive Factors (POFC).

catadioptric cameras (including radially symmetric non-central cameras) is proposed;
- Using the radial model, new visual features with decoupling properties are derived;
- An efficient image-based visual servoing approach based on the desired value of the interaction matrix is proposed.

In the next section, the mathematical background of this paper is detailed. In Section 3, new visual features are proposed. Finally, in Section 4, simulation results are presented.

II. Background

A. Radial camera model

In this paper, we consider an axial symmetric catadioptric system as shown in Figure 1. The system is made up of a pinhole camera and a rotationally symmetric mirror. The camera is positioned in such a way as to have its optical axis aligned with the mirror axis. Using the radial projection model, $3D$ point $\mathbf{P} = (X, Y, Z)$ is reflected first on a point on the mirror $\mathbf{P_r} = [X_r\, Y_r, Z_r]$ before being projected into point $\mathbf{x_m} = (x_m, y_m, 1) = \frac{\mathbf{P_r}}{Z_r}$ in the image plane, expressed in metric coordinates. Point $\mathbf{x_m}$ is projected into the catadioptric image at $\mathbf{x_d} = (x_d, y_d, 1) = \mathbf{K}\mathbf{x_m}$ expressed in pixel ($\mathbf{K}$ is the matrix of the camera intrinsic parameters, with the x and y focal lengths, principle point coordinates and zero skew). From the laws of reflection, we have: (a) vector n, the center of the projection $\mathbf{c}$, the $3D$ point $\mathbf{P}$ and $\mathbf{P_r}$ belong to the same plane π as shown in Fig. 1, (b) the angle between the incident ray and $\mathbf{n}$ is equal to the angle between the reflected ray and $\mathbf{n}$. In [21] and [17], the intersection of the planes π defined by the image points from multiple views has been used to recover linearly the structure of the scene.

The mirror is rotationally symmetric, and therefore the optical axis also belongs to π. Further, for symmetry reasons, the center of the image distortion ($\mathbf{c_{rad}}$) and the principal point coincide. In this paper, the coordinates of the image points are normalized so that they belong to the unit circle $\mathbf{x_n} = \frac{\mathbf{x_m}}{\|\mathbf{x_m}\|}$. The normalized coordinates of the pixels will be used in the derivation of the new features and image servoing algorithm. The computation of $\mathbf{x_n}$ from the image points expressed in pixel only requires the knowledge of the principal point coordinates (which coincides with the distortion center) and the ratio of the focal length parameters. The division by $\| \mathbf{x_m} \|$ implies that only the ratio between the two focal lengths needs to be known. The distortion center can be approximated by the center of the mirror border (assumed to be a circle or an ellipse) [14].

Let $\mathbf{x_u} = (x_u, y_u, 1) = \frac{\mathbf{P}}{Z}$ be the point coordinates in metric units of the projection of $\mathbf{P}$ using pinhole model as shown in Fig. 1. Since the center of the pin-hole camera and $\mathbf{P}$ belong to plane π, the point $\mathbf{x_u}$ also belongs also to the intersection of this plane with the image plane. Therefore, $\mathbf{c_{rad}}$, $\mathbf{x_u}$ and $\mathbf{x_m}$ belong to the same line. We have then $\mathbf{x_n} = \frac{\mathbf{x_u}}{\|\mathbf{x_u}\|}$, which leads to:

$$\mathbf{x_n} = \begin{bmatrix} x_n \\ y_n \end{bmatrix} = \frac{1}{\sqrt{X^2+Y^2}} \begin{bmatrix} X \\ Y \end{bmatrix} \tag{1}$$

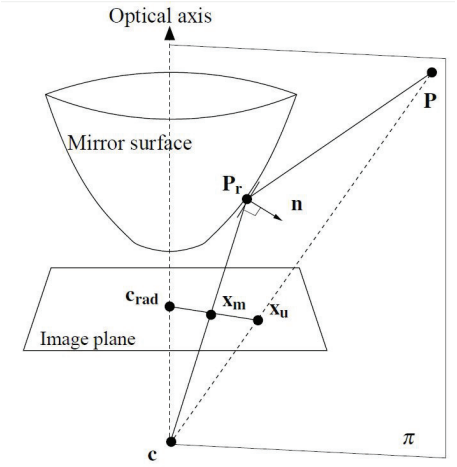

Fig. 1. Axial catadioptric system

Note that $\mathbf{x_n}$ is not defined only if the 3D point belongs to the camera optical axis (which does not happen in the case of the catadioptric camera). In the following $\mathbf{x_n}$ will be used to define new visual features to control the motion of a mobile robot moving on a plane.

B. Visual servoing

Recall that the time variation $\dot{\mathbf{s}}$ of the visual features $\mathbf{s}$ can be expressed linearly with respect to the relative camera-object kinematics screw:

$$\dot{\mathbf{s}} = \mathbf{L_s}\tau, \tag{2}$$

where $\mathbf{L_s}$ is the interaction matrix related to $\mathbf{s}$. In visual servoing, the control scheme is usually designed to reach an exponential decoupled decrease of differences on the visual features to their desired value $\mathbf{s}^*$. If we consider an eye-in-hand system observing a static object, the corresponding control law is:

$$\tau_c = -\lambda \widehat{\mathbf{L_s}}^{+}(\mathbf{s} - \mathbf{s}^*), \tag{3}$$

where $\widehat{\mathbf{L_s}}$ is a model or an approximation of $\mathbf{L_s}$, $\widehat{\mathbf{L_s}}^{+}$ the pseudo-inverse of $\widehat{\mathbf{L_s}}$, λ a positive gain tuning the time to convergence, and $\tau_c = (\mathbf{v}_c, \omega_c)$ the camera velocity sent to the low-level robot controller. In practice the matrix $\widehat{\mathbf{L_s}}$ could be chosen as the current value of the interaction matrix $\mathbf{L_s}$. This choice (except in the case of a singularity) ensures an exponential decrease of the error on features in image. Unfortunately, further to the problem of local minima and singularities, computing the exact current value of the interaction matrix requires the knowledge of the depth information. Determining this information can be time consuming, but also subject to instabilities. In order to avoid these problems, using the desired depth information values in $\mathbf{L_s}$ can be an alternative (i.e $\mathbf{L_s}(\mathbf{Z}^*)$). Unfortunately, in that case, the exponential decrease of the features errors in the image will no longer be ensured. Furthermore, if the value of $\widehat{\mathbf{L_s}}$ changes, its pseudo-inverse should also be computed at each iteration, which also can be time consuming if the size of the features vector increases significantly.

To avoid computing the depth information and inverting $\widehat{\mathbf{L_s}}$ at each time that the velocities have to be computed, a straightforward choice is to use the constant matrix $\mathbf{L_{s^*}}$ computed for the desired pose. Using $\mathbf{L_{s^*}}$ also permits to avoid the problem of the singularities (except if the desired position corresponds to a singular value of $\mathbf{L_{s^*}}$). Despite the mentioned advantages above, in practice using $\mathbf{L_{s^*}}$ ensures a local and limited domain of convergence around the desired position as compared to the case where $\mathbf{L_s}$ is used. Further, the behavior of the feature errors in the image as well as in 3D space is neither always predictable nor always satisfactory. Combining $\mathbf{L_{s^*}}$ and $\mathbf{L_s}$ in a single control law has been studied in [11] and [12] to improve the stability and 3D behavior. Unfortunately, once again, and as far as $\mathbf{L_s}$ is involved in the control law, the depth information has to be determined and $\widehat{\mathbf{L_s}}$ to be inverted.

Actually, the limited domain of convergence and the unpredictable behavior obtained using $\mathbf{L_{s^*}}$ results, in large part, from the problem of the tensor frame change. Indeed, $\mathbf{L_{s^*}}$ expresses the variations of features as a function of the camera velocities expressed in the desired frame. Therefore, if the current and the desired frames have different orientations, the tensor change of frame has to be taken into account since the velocities are to be applied in the current camera frame. This problem has been highlighted in [19] for instance. More precisely, instead of using the average $\widehat{\mathbf{L_s}} = \frac{\mathbf{L_{s^*}}+\mathbf{L_s}}{2}$, as proposed in [11], [19] proposed to use $\widehat{\mathbf{L_s}} = \frac{\mathbf{L_{s^*}}+\mathbf{L_s}\mathbf{T}^{-1}}{2}$ after integrating the spatial motion transform $\mathbf{T}$. In this paper we exploit the same idea to only use the desired value of the interaction matrix in the control law. More precisely, the velocity computed using $\widehat{\mathbf{L_s}} = \mathbf{L_{s^*}}$ in the control law (3) has to be multiplied by a spatial transformation $\mathbf{T}$. A method to effectively approximate the tensor change of frame in the case of a mobile robot (to avoid reconstructing depth data and inverting $\widehat{\mathbf{L_s}}$ at each iteration of the control loop) will be described next.

III. Visual features selection and control law

In the next paragraph, new visual features are proposed and their corresponding interaction matrices derived. A control law using the desired values of the interaction matrices is also proposed and derived.

A. Visual features

1) Features to control camera translational velocities: In order to control the translational motion of the camera, we use the inner product between two points $\mathbf{x_{n}}_i$ and $\mathbf{x_{n}}_j$ in the image:

$$c_{ij} = \mathbf{x_n}_i^\top \mathbf{x_n}_j \tag{4}$$

Taking the derivative of (4), one obtains:

$$\dot{c}_{ij} = \mathbf{x_{nj}}^\top \dot{\mathbf{x}}_{\mathbf{ni}} + \mathbf{x_{ni}}^\top \dot{\mathbf{x}}_{\mathbf{nj}} \tag{5}$$

The interaction matrix corresponding to $\mathbf{x_n}$ can be obtained by taking the derivative of (1):

$$\mathbf{L_{x_n}} = \begin{bmatrix} \mathbf{L_{x_n v}} & \mathbf{L_{x_n}}\omega \end{bmatrix} \tag{6}$$

with:

$$\mathbf{L_{x_n v}} = \begin{bmatrix} -\frac{(1-x_n^2)}{\sqrt{X^2+Y^2}} & \frac{x_n y_n}{\sqrt{X^2+Y^2}} & 0 \\ \frac{x_n y_n}{\sqrt{X^2+Y^2}} & -\frac{(1-y_n^2)}{\sqrt{X^2+Y^2}} & 0 \end{bmatrix} \tag{7}$$

and

$$\mathbf{L_{x_n}}\omega = \begin{bmatrix} -x_n y_n z_n & -(1-x_n^2) z_n & y_n \\ (1-y_n^2) z_n & x_n y_n z_n & -x_n \end{bmatrix} \tag{8}$$

where $d = \sqrt{X^2+Y^2}$ and $z_n = Z/\sqrt{X^2+Y^2}$. By combining (6) and (5), the interaction matrix $\mathbf{L}_{c_{ij}} = [\mathbf{L}_{c_{ij}\mathbf{v}} \ \mathbf{L}_{c_{ij}\omega}]$ corresponding to c_{ij} can be then obtained by:

$$\mathbf{L}_{c_{ij}\mathbf{v}} = \begin{bmatrix} (\frac{-1}{d_j} + \frac{c_{ij}}{d_i})\mathbf{x_n}_i^\top + (\frac{-1}{d_i} + \frac{c_{ij}}{d_j})\mathbf{x_n}_j^\top & 0 \end{bmatrix} \tag{9}$$

and

$$\mathbf{L}_{c_{ij}\omega} = \begin{bmatrix} y_{nj} z_{ij} + y_{ni} z_{ji} & -x_{nj} z_{ij} - x_{ni} z_{ji} & 0 \end{bmatrix} \tag{10}$$

where $z_{ij} = z_{ni} - c_{ij} z_{nj}$ and $z_{ji} = z_{nj} - c_{ij} z_{ni}$. From (9) and (10), it can be seen that c_{ij} is invariant to the motion around the optical axis. We assume that the camera is mounted on the mobile robot so that the translational motion takes place on the plane defined by the vectors $\mathbf{x}$ and $\mathbf{y}$ of the camera frame. Therefore, only the first two entries of the matrix $L_{c_{ij}\mathbf{v}}$ are useful for the control of the translational motion with respect to the x-axis and the y-axis. In the next paragraph, we explain how to select an adequate feature to control the remaining DOF, namely the rotation around the optical axis.

2) Features to control camera rotation: A natural feature in the image that can be computed from points $\mathbf{x_n}$ to control the rotation of the robot on a plane is:

$$\alpha = atan2(y_n, x_n) \tag{11}$$

The time variation of α can then be obtained by:

$$\dot{\alpha} = \frac{x_n \dot{y}_n - y_n \dot{x}_n}{x_n^2 + y_n^2} = x_n \dot{y}_n - y_n \dot{x}_n \tag{12}$$

By combining (12) and (6), the interaction matrix corresponding to α can be obtained by:

$$\mathbf{L}_\alpha = \begin{bmatrix} \frac{y_n}{d} & \frac{-x_n}{d} & 0 & x_n z_n & y_n z_n & -1 \end{bmatrix} \tag{13}$$

From (13), we can notice the direct link between α and the rotation around the z-axis. For the sake of robustness, all projected points $\mathbf{x_n}$ have to be used. A simple way to do it is by stacking all the angles α_i in a feature vector. A better choice can be combining all the points in a single and unique feature to control the rotation around the z-axis. A straightforward and simple way to use all the rotation angles could be using their average $\alpha_a = \frac{1}{N}\sum_{i=1}^{N} atan2(y_{ni}, x_{ni})$. Such feature is directly related to ω_z. However, the arithmetic average of rotations does not correspond to the real average of rotations, especially when the difference between the rotations considered is large. For instance, for a rotation angle close to π, and due to the effect of noise or due to translational motion, the computed rotation angles can have opposite signs. Therefore, the rotation angle corresponding to their arithmetic mean would have a value close to 0 instead of π or $-\pi$ generating some discontinuities in the estimation

of α_a. In this paper, we propose to define a rotation angle α_m for a point computed as a linear combination of the point projections on the circle. Let $\mathbf{p_1}$ be the point defined by:

$$\mathbf{p_1} = \sum_{i=1}^{N} a_i \mathbf{x_{ni}} \tag{14}$$

From $\mathbf{p_1}$, we define a new point $\mathbf{v_1}$ belonging to the unit circle by:

$$\mathbf{v_1} = \frac{\mathbf{p_1}}{\| \mathbf{p_1} \|} \tag{15}$$

By taking the derivative of (15), the interaction matrix corresponding to $\mathbf{v_1}$ can be obtained by:

$$\mathbf{L_{v_1}} = \frac{\mathbf{I_2} - \mathbf{v_1}\,\mathbf{v_1}^\top}{\| \mathbf{p_1} \|} \sum_{i=1}^{N} a_i \mathbf{L_{x_{ni}}} \tag{16}$$

Let α_m be the angle defined by:

$$\alpha_m = atan2(v_{1y}, v_{1x}) \tag{17}$$

By taking the derivative of (17), it can be obtained:

$$\dot{\alpha}_m = v_{1x}\dot{v}_{1y} - v_{1y}\dot{v}_{1x} \tag{18}$$

By combining (18) with (16), $L_{\alpha_m \omega_z} = -1$ is obtained. As a result one can conclude that α_m varies linearly with respect to the velocity ω_z.

a) Computation of the parameters a_i : In order to define point $\mathbf{v_1}$ on the unit circle, we need to determine the parameters a_i. More precisely, we have to define a virtual point $\mathbf{p}_1^*$ and next represent it as a linear combination of the desired projected points on the circle $\mathbf{x}_{ni}^*$. For the sake of simplicity, $\mathbf{p}_1^*$ is chosen to be unitary ($\| \mathbf{p}_1^* \| = 1$ then $\mathbf{v_1^*} = \frac{\mathbf{p_1^*}}{\|\mathbf{p_1^*}\|} = \mathbf{p}_1^*$). Let $\mathbf{p}_2^*$ be also a unit vector perpendicular to $\mathbf{p}_1^*$. As a result $\mathbf{p}_1^*$ and $\mathbf{p}_2^*$ form a direct orthogonal frame basis $\mathbf{V}^* = [\mathbf{p}_1^*; \mathbf{p}_2^*]$. It is possible to represent any given frame basis $\mathbf{V}^*$ as a linear combination of the coordinates of a set of points. For instance, $\mathbf{V}^*$ could be set as the desired frame of the camera. In any given frame basis $\mathbf{V}^*$, each projected point onto the circle can be expressed as:

$$\mathbf{x}_{ni}^* = b_{1i}\mathbf{v}_{n1}^* + b_{2i}\mathbf{v}_{n2}^* \tag{19}$$

Let $\mathbf{B}$ be the $2 \times N$ matrix that defines the coordinates of all the projected points on the new frame basis. We have:

$$\mathbf{x}_{nt}^* = \mathbf{V}^*\,\mathbf{B} \tag{20}$$

where $\mathbf{x}_{nt}^* = [\mathbf{x}_{n1}^*\ \mathbf{x}_{n2}^* \ldots \mathbf{x}_{nN}^*]$, and $\mathbf{B} = V^{*\top}\mathbf{x}_{nt}^*$. From (20), V^* can be represented as a linear combination of $\mathbf{x}_{nt}^*$ by:

$$V^* = \mathbf{x}_{nt}^*\,\mathbf{B}^+ \tag{21}$$

$\mathbf{B}^+$ is a $N \times 2$ matrix corresponding to the pseudo-inverse of $\mathbf{B}$. Therefore, the a_i and can be chosen as the first columns of $\mathbf{B}^+$.

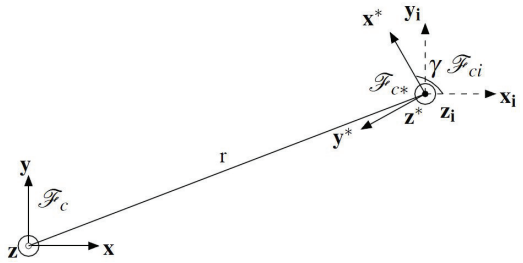

Fig. 2. Camera frame positions

B. Control law

Let $\mathbf{s_c}$ be the feature vector obtained by stacking the features c_{ij} and $\mathbf{s_c^*}$ their desired values. Let $\mathbf{L_{s_c}}$ be the interaction matrix obtained by stacking the two first entries v_x and v_y of the interaction matrix corresponding to each feature c_{ij}. Only the two first entries are taken into account because we are only concerned with a planar motion and c_{ij} is invariant to the rotation around z-axis. Let us consider that the goal is to move the desired camera position towards the initial one. Therefore, the velocities that have to be applied to the camera desired position using its corresponding interaction matrix are obtained from:

$$\begin{cases} \begin{bmatrix} v_x^* \\ v_y^* \end{bmatrix} = -\lambda \mathbf{L}_{\mathbf{s_{c*}}}^{+}(\mathbf{s_c^*} - \mathbf{s_c}) \\ \omega_z^* = \lambda(\alpha_m^* - \alpha_m) - L_{\alpha_m v_{x*}}\, v_x^* - L_{\alpha_m v_{y*}}\, v_y^* \end{cases} \tag{22}$$

where $\mathbf{L}_{\alpha_m \mathbf{v_{x*}}}$ and $\mathbf{L}_{\alpha_m \mathbf{v_{y*}}}$ represent the variation of α_m with respect to the velocities v_x and v_y respectively. Let us consider the three frames shown in Figure 2. Let $\mathscr{F}_c$ and $\mathscr{F}_{c*}$ represent respectively the current and the desired camera frames and $\mathscr{F}_{ci}$ an intermediate frame that has the same position of the center as $\mathscr{F}_{c*}$ but the orientation of $\mathscr{F}_c$. As it can be seen from Figure 2, the translational velocity to be applied to the frame $\mathscr{F}_c$ to move it towards its desired position is equal to the negative of the velocities that move $\mathscr{F}_{ci}$ towards $\mathscr{F}_c$. Therefore, to control the translational motion of the current camera position, it is more adequate to use the interaction matrix corresponding to $\mathbf{s_c}$ computed for the position corresponding to $\mathscr{F}_{ci}$:

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = -\lambda \mathbf{L}_{\mathbf{s_{ci}}}^{+}(\mathbf{s_c} - \mathbf{s_{ci}}) \tag{23}$$

In the case of the projection onto the sphere, it was shown in [20] that two interaction matrices $\mathbf{L}_{\mathbf{i_n}}^{\mathbf{2}}$ and $\mathbf{L}_{\mathbf{i_n}}^{\mathbf{1}}$ related to an invariant to the $3D$ rotation i_n and computed respectively for two camera poses 1 and 2 separated by a rotational motion are related by equation:

$$\mathbf{L}_{\mathbf{i_n}}^{\mathbf{2}} = \mathbf{L}_{\mathbf{i_n}}^{\mathbf{1}}\ {}^{\mathbf{1}}\mathbf{R_2} \tag{24}$$

where ${}^{\mathbf{1}}\mathbf{R_2}$ is the rotation matrix. Similarly, it can be shown for feature $\mathbf{s_{ci}}$ that if only a rotation is considered between $\mathscr{F}_{ci}$ and $\mathscr{F}_{c*}$, $\mathbf{L_{s_{ci}}}$ can be obtained from $\mathbf{L_{s_{c*}}}$ by:

$$\mathbf{L_{s_{ci}}} = \mathbf{L_{s_c}}\ {}^{\mathbf{c*}}\mathbf{R_i} \tag{25}$$

where ${}^{\mathbf{c*}}\mathbf{R_i}$ is the 2-dimensional rotation matrix corresponding to the rotation angle γ between $\mathscr{F}_{c*}$ and $\mathscr{F}_{ci}$. Furthermore, since c_{ij} is an invariant to the rotation around the

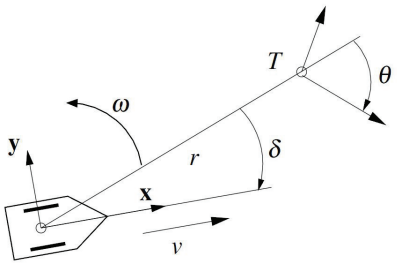

Fig. 3. Egocentric polar coordinate system with respect to the observer

z-axis, we have then $\mathbf{s_{ci}} = \mathbf{s_{c*}}$. By combining this result and (25) in (23), we obtain:

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = -{}^{\mathbf{i}}\mathbf{R}_{\mathbf{c}*}\lambda \mathbf{L}^{+}_{\mathbf{s_{c*}}}(\mathbf{s_c} - \mathbf{s_{c*}}) \tag{26}$$

By combining (26) and (22), we finally obtain:

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = -{}^{\mathbf{i}}\mathbf{R}_{*} \begin{bmatrix} v_{x*} \\ v_{y*} \end{bmatrix} \tag{27}$$

On the other hand, since and the z-axis has the same orientation in the current and the desired camera poses, we choose $\omega_z = -\omega_{z*}$. In the next section, we explain how to approximate effectively ${}^{\mathbf{i}}\mathbf{R}_{\mathbf{c}*}$.

IV. Simulation results

As non-holonomic vehicle, a differential drive mobile robot is considered. The coordinate system shown in Fig. 3 and the control law proposed in [16] are used to transform the camera velocities into linear and steering velocities to be applied to the mobile robot. More precisely, the steering velocity is defined by:

$$\omega = \frac{v_l}{r}\left[k_2(\delta - arctan(-k_1\,\theta) + (1 + \frac{k_1}{1+(k_1\theta)^2})sin(\delta)\right] \tag{28}$$

where k_1 and k_2 are two positive constants, v_l is the linear velocity, r is the distance between the current position of the robot and the target position, θ is the orientation of the target T with respect to the line of sight defined between the current and desired position of the robot (T); δ is the orientation of the vehicle heading with respect to the line of sight. To apply control law (28), it is necessary to represent the parameters v_l, r, θ and δ as a function of the cartesian camera velocities obtained by IBVS. Let $v_x(\lambda = 1)$, $v_y(\lambda = 1)$ and $\omega_z(\lambda = 1)$ be the camera velocities obtained using the scalar gain $\lambda = 1$ in (22). First, the linear velocity can be defined as $v_l = \sqrt{v_x^2 + v_y^2}$. The linear velocity becomes null when the translational motion is null (because of the invariance of the feature $\mathbf{s_c}$). The angle δ can also be estimated as the direction of the velocity to be applied to the current camera pose from $\delta = atan2(v_y, v_x)$ (since the camera is rigidly attached to the robot). The distance from the initial to the desired camera pose can be approximated by $r = \sqrt{v_x^2(\lambda = 1) + v_y^2(\lambda = 1)}$ after removing the time unit. This is equivalent to setting $\frac{v_l}{r} = \lambda$ in (28). Finally, angle θ can be defined as the rotation angle between the initial and desired camera pose. More precisely, we choose $\theta = \omega_z(\lambda = 1) - \delta$. Note also that θ as defined in Figure 3 is equal to γ as defined in Figure 2. The rotation matrix ${}^{\mathbf{i}}\mathbf{R}_*$ is also estimated using $\gamma = \omega_z(\lambda = 1)$ as a rotation angle.

In the first set of simulation results, we compare the application of the IBVS described in this paper with the application of the exact 3D parameters θ, r and δ in the control law (28). More precisely, four examples of robot parking are considered: all cases start from the same initial pose and have to converge towards four different desired poses obtained after shifting the initial pose by the translational motion defined by $[4\ 4]$ *meters* but with different orientations corresponding respectively to the angles 0, $\frac{\pi}{2}$, π and $\frac{3\pi}{2}$. The images of the following set of 8 points defined in the 3D environment were used to computed the velocities for IBVS:

$$\mathbf{X_o} = \begin{bmatrix} 15.6 & 15.6 & 7.98 & 5.38 & 9 & 6.62 & 0 & 0 \\ 7.62 & 8.29 & 15.6 & 15.6 & 0 & 0 & 8.15 & 10.32 \\ -0 & 0.86 & 2.52 & 0.32 & 2.07 & 2.14 & 1.30 & 1.79 \end{bmatrix} \tag{29}$$

In these simulations, the desired depths $r^* = \sqrt{X^{*2} + Y^{*2}}$, required to compute the interaction matrix, are assumed to be known (they could, for example, be estimated using the multiple-view scene reconstruction proposed in [2]). The simulations have been performed using the ISFfMR Integrated Simulation Framework for Mobile Robots [5]. The constants $k_1 = 4$ and $k_2 = 12$ were used in the control law (28) to control the robot using IBVS and the real 3D parameters. Figure 4 shows the trajectories performed by the robot using IBVS and using the real 3D data. From this Figure, it can be seen that the trajectories are similar and that IBVS has a performance similar (and as satisfactory) as using the 3D real data. Video 1 (in attachment to this paper) shows the behavior of the robot along the performed trajectories. The video confirms the similarities between the two trajectories and the good convergence towards the desired robot pose. Only the velocities of the robot along these trajectories differ: the convergence using IBVS is slightly slower than using the 3D real data for this experiment. This due to the fact that the amplitude of the translation estimated using IBVS is smaller than the real one.

In the second set of simulations, a wrong scale for the position of the points is used ($\widehat{r^*} = 1.3\,r^*$ is used as depth instead of the real values). The same cases of robot parking considered in the first set of simulations are considered here. Figure 5 compares the trajectories performed by the robot using the real value of εr^* and $\widehat{r^*}$ to computed the desired value of the interaction matrices. From these plots, it can be seen that the errors on the scene scale have negligible influence on the trajectories performed, which means that the curvature defined by the ratio between the steering velocity and the linear velocity $\frac{\omega}{v_l}$ is not very sensitive to the scene scale. Videos 2 and 3 compare the behavior of the robot motions along the performed trajectories and the point motions in the image using the correct and an erroneous scene scale. The two videos show that the robot converges in all cases cases, but with different velocities: the convergence using $\widehat{r^*} = 1.3\,r^*$ to computed the interaction matrices is faster than when using the real depth r^*. This is due to the fact that the scene scale used is bigger than the real one,

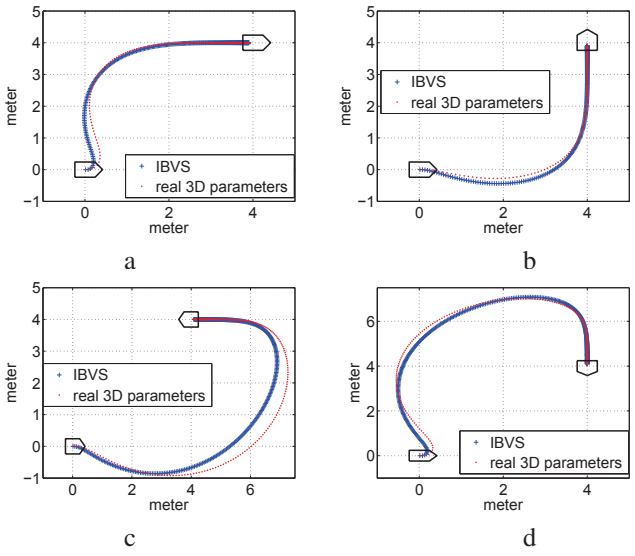

Fig. 4. Trajectory performed by the robot using the real 3D parameters and the 3D parameters estimated from an IBVS in the case of four different rotation angles: a) $0 degrees$, b) $90 degrees$, c) $180 degrees$, d) $270 degrees$

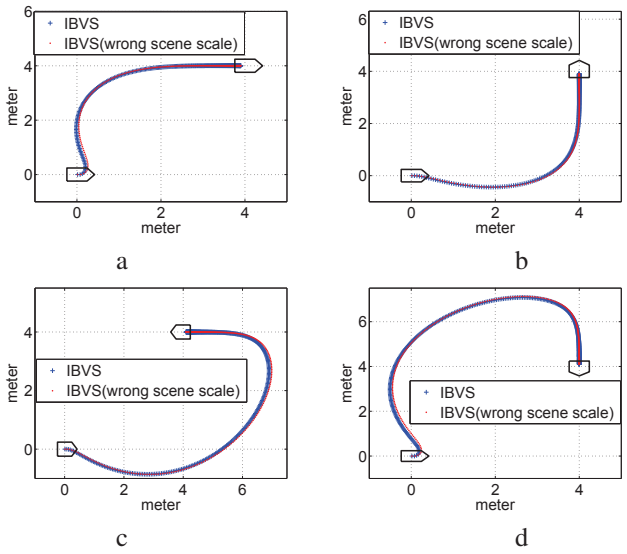

Fig. 5. Trajectory performed by the robot using the real 3D parameters and the 3D parameters estimated from an IBVS in the case of four different rotation angles: a) $0 degrees$, b) $90 degrees$, c) $180 degrees$, d) $270 degrees$

and therefore the amplitude of the translational motions to be performed are amplified.

V. Conclusion

In this paper, an IBVS using a radial camera model has been derived and proposed. This model allows the use of IBVS with non-central catadioptric systems (with axial symmetry). In addition new visual features derived from this projection model were also proposed. Furthermore, the IBVS proposed only uses the desired value of the interaction matrix to compute the velocities, which allows to avoid estimating depth during servoing as well as to avoid inverting the interaction matrix. Several simulation results are provided to show the effectiveness of our approach. Future works will include to extend the application of the radial camera model to control a 6 $DOFs$ robot.

References

[1] A. Agrawal, Y. Taguchi, and S. Ramalingam. Analytical forward projection for axial non-central dioptric and catadioptric cameras. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *ECCV 2010*, volume 6313/2010 of *Lecture Notes in Computer Science*, pages 129–143, 2010.

[2] A. Agrawal, Y. Taguchi, and S. Ramalingam. Beyond alhazen's problem: Analytical projection model for non-central catadioptric cameras with quadric mirrors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2993–3000, 2011.

[3] S. Baker and S. Nayar. A theory of catadioptric image formation. *Int. Journal of Computer Vision*, 35(2):175–196, November 1999.

[4] S. Benhimane and E. Malis. Vision-based control with respect to planar and non-planar objects using a zooming camera. In *The 11th International Conference on Advanced Robotics Coimbra, Portugal*, pages 863–866, Coimbra, Portugal, June 30 - July 3 2003.

[5] J. C. G. Cadavid. Isffmr integrated simulation framework for mobile robots. https://sites.google.com/site/isffmr/.

[6] G. Chesi, K. Hashimoto, D. Prattichizzo, and A. Vicino. Keeping features in the field of view in eye-in-hand visual servoing: a switching approach. *IEEE Transactions on Robotics*, 20(5):908–914, Oct. 2004.

[7] P. Corke, D. Strelow, and S. Singh. Omnidirectional visual odometry for a planetary rover. In *In IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 4, pages 4007–4012, Sendai, Japan, 28 Sept.–2 Oct. 2004.

[8] J. Courbon, Y. Mezouar, L. Eck, and M. Martinet. A generic fisheye camera model for robotic applications. In *IROS*, pages 1683–1688, 2007.

[9] C. Geyer and K. Daniilidis. Mirrors in motion: Epipolar geometry and motion estimation. *Int. Journal on Computer Vision*, 45(3):766–773, 2003.

[10] H. Hadj-Abdelkader, Y. Mezouar, P. Martinet, and F. Chaumette. Catadioptric visual servoing from 3d straight lines. *IEEE Trans. on Robotics*, 24(3):652–665, June 2008.

[11] E. Malis. Improving vision-based control using efficient second-order minimization techniques. In *IEEE Int. Conf. on Robotics and Automation*, volume 2, pages 1843– 1848, New Orleans, Louisiana, April 2004.

[12] M. Marey and F. Chaumette. Analysis of classical and new visual servoing control laws. In *IEEE Int. Conf. on Robotics and Automation, ICRA'08*, pages 3244–3249, Pasadena, California, May 2008.

[13] G. L. Mariottini and D. Prattichizzo. Image-based visual servoing with central catadioptric camera. *International Journal of Robotics Research*, 27:41–57, 2008.

[14] C. Mei and P. Rives. Single view point omnidirectional camera calibration from planar grids. In *IEEE Int. Conf. on Robotics and Automation*, pages 3945–3950, April 2007.

[15] Y. Mezouar and F. Chaumette. Path planning for robust image-based control. *IEEE Trans. on Robotics and Automation*, 18(4):534–549, August 2002.

[16] J. J. Park and B. Kuipers. A smooth control law for graceful motion of differential wheeled mobile robots in 2d environment. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4896–4901, Shanghai, China, May 9-13 2011.

[17] C. Sagues, A. Murillo, J. Guerrero, T. Goedeme, T. Tuytelaars, and L. Van Gool. Localization with omnidirectional images using the radial trifocal tensor. In *IEEE Int. Conf. on Robotics and Automation*, pages 551 – 556, Orlando, FL, 15-19 May 2006.

[18] T. Svoboda and T. Pajdla. Epipolar geometry for central catadioptric cameras. *Int. Journal on Computer Vision*, 49(1):23–37, August 2002.

[19] O. Tahri and Y. Mezouar. On visual servoing based on efficient second order minimization. *Robotics and Autonomous Systems*, 58(5):712–719, May 2010.

[20] O. Tahri, Y. Mezouar, F. Chaumette, and P. Corke. Decoupled image-based visual servoing for cameras obeying the unified projection model. *IEEE Trans. on Robotics*, 26(4):684 – 697, August 2010.

[21] S. Thirthala and M. Pollefeys. Multi-view geometry of 1d radial cameras and its application to omnidirectional camera calibration. In *Tenth IEEE International Conference on Computer Vision*, volume 2, pages 1539–1546, Beijing, China, 17-21 Oct. 2005.

[22] S. Thirthala and M. Pollefeys. The radial trifocal tensor: a tool for calibrating the radial distortion of wide-angle cameras. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 321 – 328, San Diego, CA, USA, 20-25 June 2005.