2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Uncalibrated Visual Compass from Omnidirectional Line Images with Application to Attitude MAV Estimation

Stefano Scheggi[1], Fabio Morbidi[2], Domenico Prattichizzo[3]

Abstract— **This paper presents a new algorithm based on previous results of the authors, for the estimation of the yaw angle of an omnidirectional camera/robot undergoing a 6-DoF rigid motion. Our *real-time* algorithm is *uncalibrated*, *robust* to noisy data, and it only relies on the projection of *3-D parallel lines* as image features. Numerical and real-world experiments conducted with an eye-in-hand robot manipulator, which we used to simulate the 3-D motion of a *Micro unmanned Aerial Vehicle* (MAV), show the accuracy and reliability of our estimation algorithm.**

I. INTRODUCTION

In recent years we have witnessed a growing number of applications involving *Unmanned Aerial Vehicles* (UAVs) [1]–[5]: these applications range from infrastructure inspection, mapping of unknown terrains, espionage, object transportation, to entertainment (film shooting and light shows [6], [7]). Several factors have contributed to this success and to the recent large diffusion of UAVs: decreasing cost (cf. AR.Drone's Parrot quadricopter), enhanced sensing and autonomy, as well as the ability to carry heavier payloads. Three categories of *Micro-UAVs* (or MAVs, for short) are currently under study or development in the research community: fixed-wing aircraft, avian-style flapping wing aircraft and rotor craft [4]. Two configurations of rotor craft have recently gained wide acceptance: co-axial rotor craft, which are equipped with two counter-rotating co-axial rotors and with a stabilizer bar, and multi-rotor aircraft (e.g., the popular quadrotors with four propellers).

The *localization* of MAVs represents a challenging research issue: in fact, while Vicon systems or overhead camera networks can be employed to precisely localize an aerial vehicle in limited workspaces, they are unusable outdoors. When GPS information is not available or is too inaccurate for the application at hand, the only possibility is then to rely on onboard sensors for vehicle's position and attitude estimation. Because of their small size, limited weight, affordability and low-power consumption, *vision sensors* are particularly suited for this task.

Several works have appeared in the recent robotics literature, dealing with MAV localization using only *visual information*. The problem of autonomously landing a MAV on a known platform, was addressed in [8]. In [9], an aerial vehicle is localized using a downward-looking monocular camera. A visual SLAM algorithm tracks the pose of the camera while simultaneously building an incremental map of the surrounding region. Based on this pose estimation, a LQG/LTR-based controller stabilizes the vehicle at a desired setpoint and allows the execution of some simple maneuvers, like take-off, landing and hovering.

In [10], a simple technique is described for estimating the roll and pitch angle of a UAV, based on the detection of the horizon line in a pinhole image. In [11], instead, the attitude (roll and pitch angles) of a UAV is estimated by leveraging the geometric properties of the on-board catadioptric sensor. Since the horizon line used in [11] becomes an inadequate feature in a urban environment, the same authors, in [12], have proposed an omnidirectional vision system based on straight lines, that is able to compute the roll and pitch angles. The method consists in finding bundles of horizontal and vertical parallel lines in order to obtain an absolute reference for the attitude computation. Finally, recently in [13] a novel rotation-estimation approach based on the extraction of vanishing points in omnidirectional images of urban environments, has been presented. However, a common strong assumption in [10]–[13] is that the vision sensor is *fully calibrated*.

As it is known, *panoramic cameras* have a wider field of view than standard pinhole sensors, but to the best of our knowledge, relatively few works in the literature have explored their potential in aerial robotics (see, e.g., [13], [14] and the references therein). This paper builds upon our previous work [15], and presents a *robust* (thanks to our RANSAC-based implementation) and *uncalibrated* visual compass (VC) algorithm for estimating the yaw angle of a camera/robot, which *only* leverages omnidirectional line images, widely available in man-made environments. While the method described in [15] is uniquely valid for planar motions of the camera/robot, in this paper we extend its applicability to sensors undergoing a 6-DoF motion, thus making it applicable to the MAV attitude estimation problem. Numerical as well as real-world experiments conducted with a paracatadioptric camera mounted on the end-effector of a robot manipulator (that we used to generate accurate 3-D trajectories, thus simulating the motion of a MAV), illustrate the theory and show the accuracy and robustness of our VC algorithm as well as its *real-time* capabilities. It is worth underlying here that although MEMS gyroscopes may represent a valid alternative to panoramic cameras for yaw-angle estimation, in terms of weight, size and price, they are known to be sensitive to temperature/calibration and to suffer from bias errors.

The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement n° 288917 of the project "DALi - Devices for Assisted Living".

[1]S. Scheggi and D. Prattichizzo are with the Department of Information Engineering and Mathematics, University of Siena, 53100 Siena, Italy, scheggi,prattichizzo@dii.unisi.it

[2]F. Morbidi is with the NeCS team, INRIA Grenoble Rhône-Alpes, France, fabio.morbidi@inria.fr

[3]D. Prattichizzo is also with the Department of Advanced Robotics, Istituto Italiano di Tecnologia, via Morego, 30, 16163 Genova, Italy.

The rest of this paper is organized as follows. Sect. II briefly reviews the basics on paracatadioptric projection of 3-D lines. Sect. III and Sect. IV describe our VC estimation algorithm and its main properties. The results of simulation and real-world experiments are discussed in Sect. V. Finally, in Sect. VI, conclusions are drawn and possible avenues of future research are highlighted.

II. Basics on paracatadioptric projection of 3-D lines

Fig. 1 illustrates the imaging model of a paracatadioptric camera with mirror focus at O: a generic 3-D scene point $\mathbf{X} \in I\!R^3$ (expressed in the mirror frame $\{M\}$), is projected onto the parabolic mirror surface at $\mathbf{x} \in I\!R^3$ through O. Then, an orthographic projection maps $\mathbf{x}$ at $\mathbf{u}$ (pixels), onto the image plane $\mathcal{I}$. The transformation from $\mathbf{X}$ to $\mathbf{u}$ is analytically described by a nonlinear function $\eta : I\!R^3 \to I\!R^2$ that depends on both the camera intrinsic calibration parameters and the mirror geometry [16].

Let us now consider the case in which a generic 3-D line $\mathcal{L}$ is observed by the paracatadioptric camera. We will refer to the *interpretation plane* as the plane with normal vector $\mathbf{n} = [n_x,\ n_y,\ n_z]^T$ (in $\{M\}$) that passes through $\mathcal{L}$ and O.

Proposition 1 (Paracatadioptric line image [17]): Consider the setup in Fig. 1, where a line $\mathcal{L}$ is observed by a paracatadioptric camera at O. If $n_z \neq 0$, then $\mathcal{L}$ projects onto the image plane $\mathcal{I}$ at a *circle* $\mathcal{C}$ with center $\mathbf{c} \triangleq [c_x,\ c_y]^T$ (pixels) and radius r (pixels) given by,

$$\mathbf{c} = \mathbf{u}_0 - 2\,af\left[\frac{n_x}{n_z},\ \frac{n_y}{n_z}\right]^T, \qquad r = \frac{2\,a\,f}{n_z},$$

where a is the focal parameter of the parabolic mirror (i.e. the distance between the focus and the vertex of the paraboloid), $\mathbf{u}_0 \triangleq [u_0,\ v_0]^T$ the optical center (in pixels), and f (pixels) the focal length of the camera. ■

In Prop. 1, we have assumed that the line $\mathcal{L}$ is in a generic 3-D configuration. In the special case of a line *orthogonal* to the image plane $\mathcal{I}$ (see Fig. 2), the projected circle $\mathcal{C}$ reduces to an image line $\boldsymbol{\ell}$ through $\mathbf{u}_0$, as stated

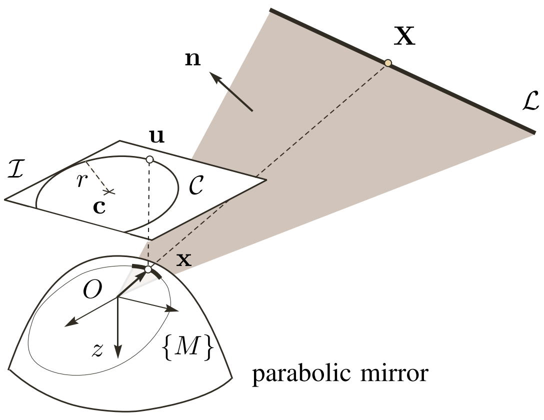


Fig. 1. *Projection of a 3-D line $\mathcal{L}$*: the interpretation plane passes through the focus O of the parabolic mirror and the line $\mathcal{L}$, and intersects the mirror at a curve that is orthographically projected at a circle $\mathcal{C}$ onto the image plane $\mathcal{I}$ (with center $\mathbf{c}$ and radius r).

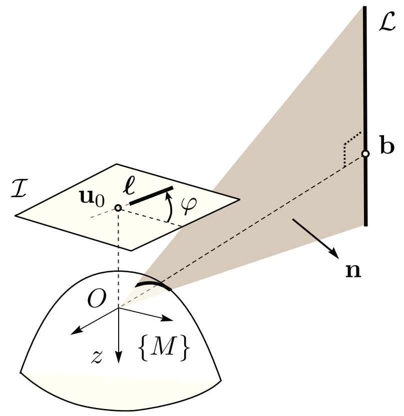


Fig. 2. *Projection of a 3-D vertical line $\mathcal{L}$*: the interpretation plane intersects the mirror at a curve that is orthographically projected onto the image plane at a line $\boldsymbol{\ell}$ passing through the optical center $\mathbf{u}_0$ and with a slope φ with respect to the image horizontal axis.

in [18]. We henceforth refer to this category of lines $\mathcal{L}$, as *vertical lines*.

III. Invariant parallel lines

The following definition is crucial for the subsequent developments.

Definition 1 (Invariant property): A set of non-vertical parallel lines is *invariant* to a particular camera rotation and/or translation, if the line joining the centers of the circles obtained as the projection of such lines on the camera image plane, does not change its slope.

Remark 1: It has been shown in [15], that non-vertical parallel lines are invariant to camera *translations*. In what follows, we will simply refer to such lines as *parallel lines*. ⋄

In order to estimate the camera z-rotation angle when the sensor undergoes a full 6-DoF motion, we need to find sets of parallel lines which are *invariant* to rotations about the x- and y- axes. The next proposition identifies set of lines which can be used to estimate the z-rotation angle between two views, without the knowledge of the camera calibration parameters. We will use $\mathbf{R}_{x,\alpha} \in I\!R^{3\times 3}$ to denote the basic rotation matrix about the x-axis of an angle α.

Proposition 2 (Invariant parallel lines): Consider a set of parallel lines and suppose that a rigid transformation $(\mathbf{R}, \mathbf{t}) \in \mathrm{SE}(3)$ with $\mathbf{R} = \mathbf{R}_{z,\theta}\,\mathbf{R}_{y,\beta}\,\mathbf{R}_{x,\alpha}$ and $\mathbf{t} \triangleq [t_x,\ t_y,\ t_z]^T$ is applied to them:

- If the parallel lines have direction $\mathbf{d}_x = [1,\ 0,\ 0]^T$, then the set is *invariant* to $\mathbf{R}_{y,\beta}\,\mathbf{R}_{x,\alpha}$.
- If the parallel lines have direction $\mathbf{d}_y = [0,\ 1,\ 0]^T$, then the set is *invariant* to $\mathbf{R}_{x,\alpha}$.

Proof: From Remark 1, we know that *parallel lines* are invariant to camera translations: hence, in the rest of the proof we will only focus on the rotational motion.

Let us suppose to have a set of parallel lines with direction $\mathbf{d}_x = [1,\ 0,\ 0]^T$ and a generic point $\mathbf{P} = [x,\ y,\ z]^T$. The interpretation plane passing through $\mathbf{P}$ with direction $\mathbf{d}_x$ has normal vector $\mathbf{n} = \mathbf{d}_x \times \mathbf{P} = [0,\ -z,\ y]^T$. By applying $\mathbf{R}_{x,\alpha}$ to $\mathbf{n}$, we obtain,

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}\begin{bmatrix} 0 \\ -z \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ -z\cos\alpha - y\sin\alpha \\ -z\sin\alpha + y\cos\alpha \end{bmatrix}. \quad (1)$$

From Prop. 1, it follows that such line projects onto a circle with center's coordinates,

$$\mathbf{c} = \mathbf{u}_0 - 2\,af\left[0,\ \frac{-z\cos\alpha - y\sin\alpha}{-z\sin\alpha + y\cos\alpha}\right]^T.$$

As a consequence, given a set of parallel lines with direction $\mathbf{d}_x$, all lines project onto circles having $c_x = u_0$. In this case the line joining these centers is always a vertical line in the camera image plane, having slope $\varphi = \infty$.

By applying $\mathbf{R}_{y,\beta}\,\mathbf{R}_{x,\alpha}$ to $\mathbf{n}$, from (1) we obtain,

$$\begin{bmatrix}\cos\beta & 0 & \sin\beta\\ 0 & 1 & 0\\ -\sin\beta & 0 & \cos\beta\end{bmatrix}\begin{bmatrix}0\\ -z\cos\alpha - y\sin\alpha\\ -z\sin\alpha + y\cos\alpha\end{bmatrix} = \begin{bmatrix}(-z\sin\alpha + y\cos\alpha)\sin\beta\\ -z\cos\alpha - y\sin\alpha\\ (-z\sin\alpha + y\cos\alpha)\cos\beta\end{bmatrix}. \quad (2)$$

Such line projects onto a circle with center's coordinates,

$$\mathbf{c} = \mathbf{u}_0 - 2\,af\left[\tan\beta,\ \frac{-z\cos\alpha - y\sin\alpha}{(-z\sin\alpha + y\cos\alpha)\cos\beta}\right]^T. \quad (3)$$

Since c_x depends on the camera internal parameters and on the rotation angle β, it is constant for all lines having direction $\mathbf{d}_x$. Then, parallel lines with direction $\mathbf{d}_x$ project onto circles all having $c_x = u_0 - 2\,af\tan\beta$. Also in this case, the line joining these centers is always a vertical line. In conclusion, sets of parallel lines having direction $\mathbf{d}_x = [1,\ 0,\ 0]^T$ are invariant to $\mathbf{R}_{y,\beta}\,\mathbf{R}_{x,\alpha}$ because the line joining the circles' centers does not change its slope when rotations about the x-, y-axes are applied.

Let us now repeat the previous considerations with $\mathbf{d}_y = [0,\ 1,\ 0]^T$. The interpretation plane passing through $\mathbf{P}$ with direction $\mathbf{d}_y$ has normal vector $\mathbf{n} = \mathbf{d}_y \times \mathbf{P} = [z,\ 0,\ -x]^T$. By applying $\mathbf{R}_{x,\alpha}$ to $\mathbf{n}$, we obtain,

$$\begin{bmatrix}1 & 0 & 0\\ 0 & \cos\alpha & -\sin\alpha\\ 0 & \sin\alpha & \cos\alpha\end{bmatrix}\begin{bmatrix}z\\ 0\\ -x\end{bmatrix} = \begin{bmatrix}z\\ x\sin\alpha\\ -x\cos\alpha\end{bmatrix}. \quad (4)$$

Such line projects onto a circle with center's coordinates,

$$\mathbf{c} = \mathbf{u}_0 - 2\,af\left[\frac{z}{-x\cos\alpha},\ -\tan\alpha\right]^T.$$

As a consequence, parallel lines with direction $\mathbf{d}_y$ project onto circles all having $c_y = v_0 + 2\,af\tan\alpha$, which is constant for all lines. In this case the line joining these centers is always an horizontal line, being the slope $\varphi = 0$.

By applying $\mathbf{R}_{y,\beta}\,\mathbf{R}_{x,\alpha}$ to $\mathbf{n}$, from (4) we end up with,

$$\begin{bmatrix}\cos\beta & 0 & \sin\beta\\ 0 & 1 & 0\\ -\sin\beta & 0 & \cos\beta\end{bmatrix}\begin{bmatrix}z\\ x\sin\alpha\\ -x\cos\alpha\end{bmatrix} = \begin{bmatrix}z\cos\beta - x\sin\beta\cos\alpha\\ x\sin\alpha\\ -z\sin\beta + x\cos\beta\cos\alpha\end{bmatrix}.$$

Such line projects onto a circle with center's coordinates,

$$\mathbf{c} = \mathbf{u}_0 - 2\,af\begin{bmatrix}\dfrac{z\cos\beta - x\sin\beta\cos\alpha}{-z\sin\beta + x\cos\beta\cos\alpha}\\[2ex] \dfrac{x\sin\alpha}{-z\sin\beta + x\cos\beta\cos\alpha}\end{bmatrix}.$$

Since the centers' coordinates depend on the line displacement, these lines are not *invariant* to $\mathbf{R}_{y,\beta}\,\mathbf{R}_{x,\alpha}$. ■

Prop. 2 states that we can use *parallel lines* with direction $\mathbf{d}_x$ in the initial camera frame, to recover the yaw angle θ when the camera undergoes a full 6-DoF motion.

From Prop. 1, we know that *parallel lines* project onto the image plane at circles. In the next proposition, we show how a rotation about the $z-$axis influences the slope φ of the line joining the circles' centers obtained as projection of parallel lines with directions $\mathbf{d}_x$ and $\mathbf{d}_y$, respectively.

Proposition 3:

- If the rigid trasformation $(\mathbf{R}, \mathbf{t}) \in$ SE(3) with $\mathbf{R} = \mathbf{R}_{z,\theta}\,\mathbf{R}_{y,\beta}\,\mathbf{R}_{x,\alpha}$ and $\mathbf{t} \triangleq [t_x,\ t_y,\ t_z]^T$, is applied to a set of parallel lines having direction $\mathbf{d}_x$, then the slope of the line joining the circles' centers is $\varphi = -\cot\theta$.
- If the rigid trasformation $(\mathbf{R}, \mathbf{t}) \in$ SE(3) with $\mathbf{R} = \mathbf{R}_{z,\theta}\,\mathbf{R}_{x,\alpha}$ and $\mathbf{t} \triangleq [t_x,\ t_y,\ t_z]^T$, is applied to a set of parallel lines having direction $\mathbf{d}_y$, then the slope of the line joining the circles' centers is $\varphi = \tan\theta$.

Proof: Let us first consider parallel lines with direction $\mathbf{d}_x = [1,\ 0,\ 0]^T$. By applying $\mathbf{R}_{z,\theta}$ to (2) we obtain that such lines project onto circles having center coordinates,

$$\mathbf{c} = \mathbf{u}_0 - 2\,af\begin{bmatrix}\dfrac{(-zs\alpha + yc\alpha)\,c\theta s\beta + (zc\alpha + ys\alpha)\,s\theta}{(-zs\alpha + yc\alpha)\,c\beta}\\[2ex] \dfrac{(-zs\alpha + yc\alpha)\,s\theta s\beta - (zc\alpha + ys\alpha)\,c\theta}{(-zs\alpha + yc\alpha)\,c\beta}\end{bmatrix},$$

where $c(\cdot)$, $s(\cdot)$ stand for $\cos(\cdot)$ and $\sin(\cdot)$, respectively. Let us consider two parallel lines $\mathcal{L}_i$ and $\mathcal{L}_j$, having the same direction $\mathbf{d}_x$. From [18], $\mathcal{L}_i$ and $\mathcal{L}_j$ project onto circles whose centers are collinear. Let the centers of the image circles be $\mathbf{c}^i \triangleq [c_x^i,\ c_y^i]^T$, $\mathbf{c}^j \triangleq [c_x^j,\ c_y^j]^T$, and let us compute the slope φ of the line joining these centers as,

$$\varphi = \frac{c_y^i - c_y^j}{c_x^i - c_x^j}. \quad (5)$$

Since,

$$c_y^i - c_y^j = 2\,af\,\frac{(-y^i z^j + z^i y^j)\,c\theta}{(-z^i s\alpha + y^i c\alpha)(-z^j s\alpha + y^j c\alpha)\,c\beta},$$

$$c_x^i - c_x^j = 2\,af\,\frac{-(-y^i z^j + z^i y^j)\,s\theta}{(-z^i s\alpha + y^i c\alpha)(-z^j s\alpha + y^j c\alpha)\,c\beta},$$

then $\varphi = -\cot\theta$. As a consequence, θ is the angle between such a line and the y-axis of the camera image plane.

Let us now consider the direction $\mathbf{d}_y = [0,\ 1,\ 0]^T$. By applying $\mathbf{R}_{z,\theta}$ to (4), we obtain that,

$$\mathbf{c} = \mathbf{u}_0 - 2\,af\left[\frac{zc\theta - xs\theta\,s\alpha}{-x\,c\alpha},\quad \frac{zs\theta + xc\theta\,s\alpha}{-x\,c\alpha}\right]^T. \quad (6)$$

By plugging (6) in (5), we get $\varphi = \tan\theta$. ■

As we will see in the next section, Prop. 2 and Prop. 3 are at the core of our visual compass algorithm.

IV. Visual compass algorithm

In this section we present a VC algorithm which allows us to estimate the camera yaw angle θ, when it undergoes a full 6-DoF motion. The algorithm relies on Prop. 2, Prop. 3 and

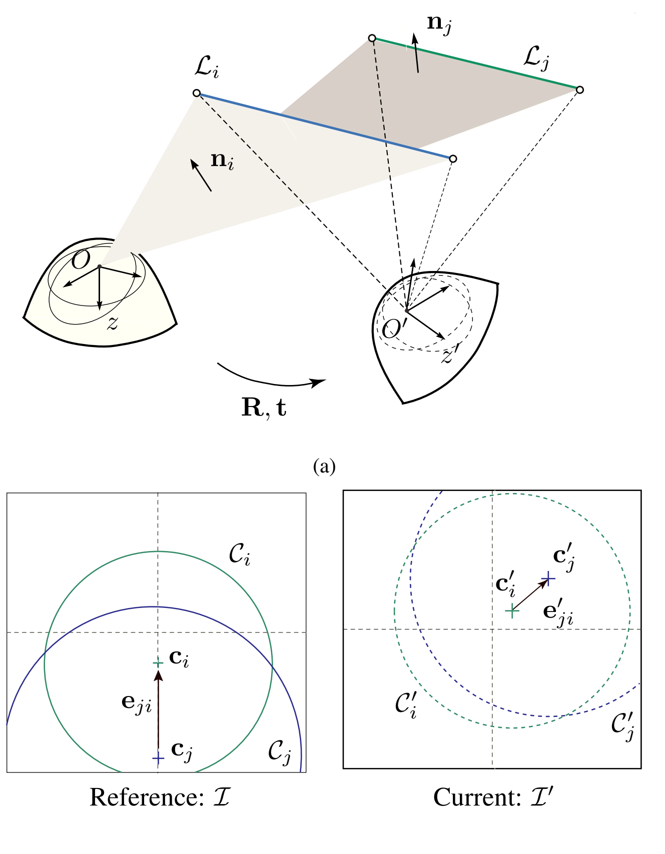


Fig. 3. *Estimation of the rotation about the z-axis*: (a) Two paracatadioptric cameras are displaced of $(\mathbf{R}, \mathbf{t})$ and observe two generic 3-D parallel lines $\mathcal{L}_i$ and $\mathcal{L}_j$; (b)-(c) The two lines project onto each image plane $\mathcal{I}$ and $\mathcal{I}'$, at two circle pairs $(\mathcal{C}_i, \mathcal{C}_j)$ and $(\mathcal{C}'_i, \mathcal{C}'_j)$, respectively. From the centers of the circles we obtain the unit-norm vectors $\mathbf{e}'_{ji}$ and $\mathbf{e}_{ji}$ which are rotated of an angle $\theta \in (-\pi/2, \pi/2]$.

on the extension of the *disparity-circles constraint* in [15], reported in next theorem.

Theorem 1 (Extended disparity-circles constraint): Consider the two-views setup shown in Fig. 3(a) in which the *current view* at O is rotated about the x, y, z-axes of an angle α, β, θ, respectively, with $\alpha, \beta, \theta \in (-\pi/2, \pi/2]$, and translated of $\mathbf{t} \in \mathbb{R}^3$ (with respect to the *reference view* at O'). Let the rotation between the two cameras be $\mathbf{R} = \mathbf{R}_{x,\alpha}\,\mathbf{R}_{y,\beta}\,\mathbf{R}_{z,\theta}$, and let us assume that two 3-D parallel lines $\mathcal{L}_i$ and $\mathcal{L}_j$ having direction $\mathbf{d}_x = [1,\ 0,\ 0]^T$ in O project onto the image planes $\mathcal{I}$ and $\mathcal{I}'$, at two circle pairs $(\mathcal{C}_i, \mathcal{C}_j)$ and $(\mathcal{C}'_i, \mathcal{C}'_j)$, respectively, with centers $(\mathbf{c}_i, \mathbf{c}_j)$ and $(\mathbf{c}'_i, \mathbf{c}'_j)$ (see Figs. 3(b)-(c)). Then, the following constraint holds true:

$$\mathbf{e}_{ji} = \mathbf{R}^{2\mathrm{D}}_{z,\theta}\,\mathbf{e}'_{ji}, \tag{7}$$

where

$$\mathbf{e}_{ji} \triangleq \frac{\mathbf{c}_j - \mathbf{c}_i}{\|\mathbf{c}_j - \mathbf{c}_i\|},\ \ \mathbf{e}'_{ji} \triangleq \frac{\mathbf{c}'_j - \mathbf{c}'_i}{\|\mathbf{c}'_j - \mathbf{c}'_i\|},\ \ \mathbf{R}^{2\mathrm{D}}_{z,\theta} \triangleq \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}.$$

Proof: Since $\mathcal{L}_i$, $\mathcal{L}_j$ are *invariant* to translation (recall Remark 1), we can focus our attention on the unknown rotation $\mathbf{R}$ between the two views. Being $\mathbf{d}_x = [1,\ 0,\ 0]^T$ in O, from Prop. 3 it follows that the slope of vector $\mathbf{e}_{ji}$ is $\varphi = \infty$. Vector $\mathbf{d}_x$ can be expressed in O' as, $\mathbf{d}'_x = \mathbf{R}^{-1}\,\mathbf{d}_x = (\mathbf{R}_{x,\alpha}\,\mathbf{R}_{y,\beta}\,\mathbf{R}_{z,\theta})^T\,\mathbf{d}_x = \mathbf{R}_{z,-\theta}\,\mathbf{R}_{y,-\beta}\,\mathbf{R}_{x,-\alpha}\,\mathbf{d}_x$. From Prop. 2 it follows that sets of parallel lines with direction $\mathbf{d}_x$ are *invariant* to rotations about the x-, y-axes. As a consequence, owing to Prop. 3 it follows that the slope φ' of vector $\mathbf{e}'_{ji}$ depends only on the matrix $\mathbf{R}_{z,\theta}$, i.e. $\varphi' = -\cot(-\theta) = \cot\theta$. Finally, since $\varphi = \infty$ and $\varphi' = \cot\theta$, we obtain (7). ■

Algorithm 1 Implementation of the VC algorithm

```
Phase 1 [Feature Detection]:
1: Automatically detect sets of parallel lines as described
   in [15]. Such lines project at circles onto the camera
   image plane.
Phase 2 [Initialization]:
1: Select one set of parallel lines and ensure that the line
   joining the circles' centers $\mathbf{e}_{ji}$ in the image plane at
   the initial time instant, has slope $\varphi = \infty$ and passes
   through $u_0$ (cf. Remark 2).
Phase 3 [Estimation]:
1: while the camera/robot moves do
2:    Use the ViSP software [19] to track circles from the
      previous to the current frame.
3:    Use the tracked and the initial (Phase 2) circles'
      centers in Th. 1, and determine $\theta$ using a RANSAC-
      based approach (see [15]).
4:    if some of the 3-D lines are no more visible then
5:       Go to Phase 1.
6:       Add only the 3-D lines that are parallel to the initial
         set selected in Phase 2.
7:    end if
8: end while
```

Algorithm 1, summarizes the different phases of our VC algorithm for the estimation of the z-rotation angle during the camera motion. At each time frame the algorithm estimates θ between the current and the initial view.

Remark 2: The proposed algorithm needs a bundle of parallel lines with direction $\mathbf{d}_x = [1,\ 0,\ 0]^T$ in the camera frame at the *initial time*. Such a requirement can be satisfied by ensuring that the line joining the circles' centers in the initial image has slope $\varphi = \infty$ and passes through u_0 (recall Eq. (3) and see Fig. 7). Although this step might seem critical in practice, in the next section we provide experimental evidence that the proposed algorithm is robust against image noise and initial camera alignment errors. ◇

V. Experimental validation

In order to test the effectiveness of the proposed VC algorithm, we conducted extensive numerical simulations (see Sect. V-A) and experimental tests (see Sect. V-B).

A. Simulations

The simulation results reported in this section have been obtained using the *Epipolar Geometry Toolbox* (EGT) for MATLAB. For the sake of generality, we implemented the unified panoramic-camera imaging model by Geyer and Daniilidis [16] which describes any central panoramic camera projection as a projection between a sphere and a plane. In order to assess the accuracy and robustness of our algorithm to noisy data, we added Gaussian image noise with increasing standard deviation $\sigma \in \{0, 0.5, \ldots, 2\}$ pixels. The two views are rotated of $\mathbf{R} = \mathbf{R}_{x,\pi/4}\,\mathbf{R}_{y,\pi/6}\,\mathbf{R}_{z,\pi/3}$ and translated of $\mathbf{t} = [1,\ 2,\ 1]^T$ meters. A set of 4 parallel lines was considered in our test, and we did not assume to know the correspondences (cf. [15, Cor. 1]).

Fig. 4 shows the mean and the standard deviation of the rotation-estimation error $|\widehat{\theta} - \theta|$ (deg.) obtained from the VC algorithm. These values have been computed by averaging the rotation errors over 100 realizations. From Fig. 4, we observe that the proposed algorithm is robust against image noise and exhibits a maximum estimation error of about 1.4 deg. for $\sigma = 2$ pixels. We also tested the robustness of the proposed algorithm against an initial misalignment of the camera with respect to the set of parallel lines (recall Remark 2). To this end, we introduced independent random rotations with zero mean and standard deviation σ_u, about the camera's x-, y- and z-axes (the standard deviation of the image noise is $\sigma = 1.5$ pixels). Fig. 5 shows the mean and the standard deviation of the rotation-estimation error $|\widehat{\theta} - \theta|$ over 100 realizations: despite the initial misalignment, the mean of the error is always smaller than 4 deg.

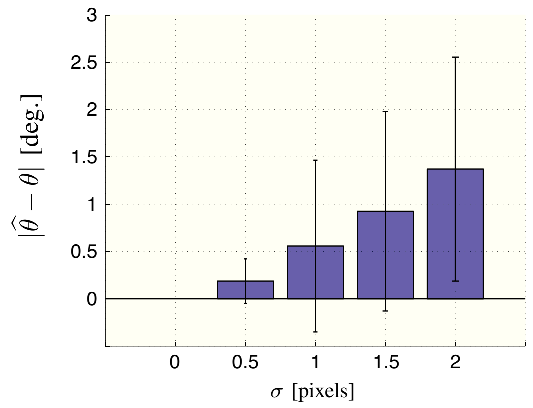


Fig. 4. *Simulation results with image noise*: mean and standard deviation of the rotation-estimation error $|\widehat{\theta} - \theta|$. The reported values are obtained by averaging the rotation errors over 100 realizations.

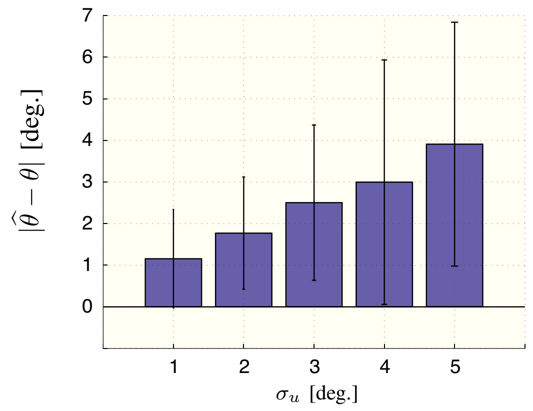
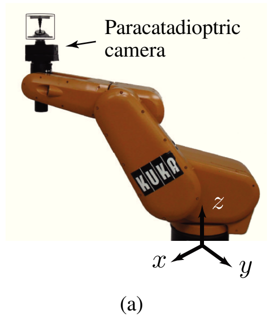


Fig. 5. *Simulation results with image noise and initial camera misalignment*: mean and standard deviation of the rotation-estimation error $|\widehat{\theta} - \theta|$ when normally-distributed random rotations (with zero mean and standard deviation σ_u) about the x-, y-, z-axes are introduced, and an image noise with standard deviation of 1.5 pixels is used. The reported values are obtained by averaging the rotation errors over 100 realizations.



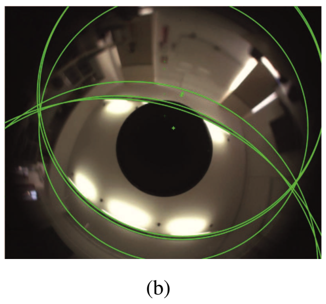


Fig. 6. (a) The paracatadioptric camera is mounted on the end-effector of a KUKA manipulator; (b) Camera view with detected circles (green).

B. Experiments

In the experiments described in this section, we used a Remote Reality NetVision360 paracatadioptric mirror screwed on a Lumenera LU071C pinhole camera. The camera was mounted on the end-effector of a 6-DoF KUKA KR 3 manipulator (see Fig. 6)[1]. We decided to use this robot because of its highly-accurate measurements of the camera pose (in the order of millimeters/tenths of degree), that we adopted as our ground truth. As aforementioned, the manipulator was used to simulate the 3-D motion of a MAV. Space constraints prevent us to provide here a detailed description of the computer-vision algorithms that we used for the automatic detection of the image circles and their tracking as the camera moves. For more information, the reader is referred to [15].

Figs. 7(a)-(c) show the motion of the camera with respect to the initial pose, that we considered in our two experimental tests. In particular, Fig. 7(a) and Fig. 7(b) report the time evolution of the camera position and orientation, and Fig. 7(c) its 3-D trajectory. A maximum number of 6 3-D lines for the VC algorithm, was considered in our two tests. The algorithm run at an average frame rate of about 10fps.

In the first test (see video 1), the VC algorithm was accurately initialized (i.e., there was not an appreciable misalignment between the orientation of the camera and the selected set of parallel lines at the initial time). As shown in Fig. 7(d), the maximum estimation error of the angle θ is about 1.3 deg. in this case.

In the second test (see video 2), we violated on purpose the initial camera alignment by adding a spurious rotation of 4 deg. about the x-, y-, z-axes (as shown in Fig. 7(f), this misalignment can be easily noticed in the image). In spite of this uncertainty, the maximum estimation error $|\widehat{\theta} - \theta|$ is lower than 4.6 deg. (see Fig. 7(e)).

VI. Conclusions and Future Work

In this paper we have presented a new algorithm for estimating the yaw angle of an omnidirectional camera/robot moving in a 3-D environment. The proposed VC algorithm has a number of attractive features: it is *uncalibrated*, *robust* to noisy data, capable of *real-time* operation and it only relies on the projection of *3-D parallel lines* on the image

[1]The videos of the real-time experiments are available at: http://sirslab.dii.unisi.it/research/vision/6dof-visual-compass/

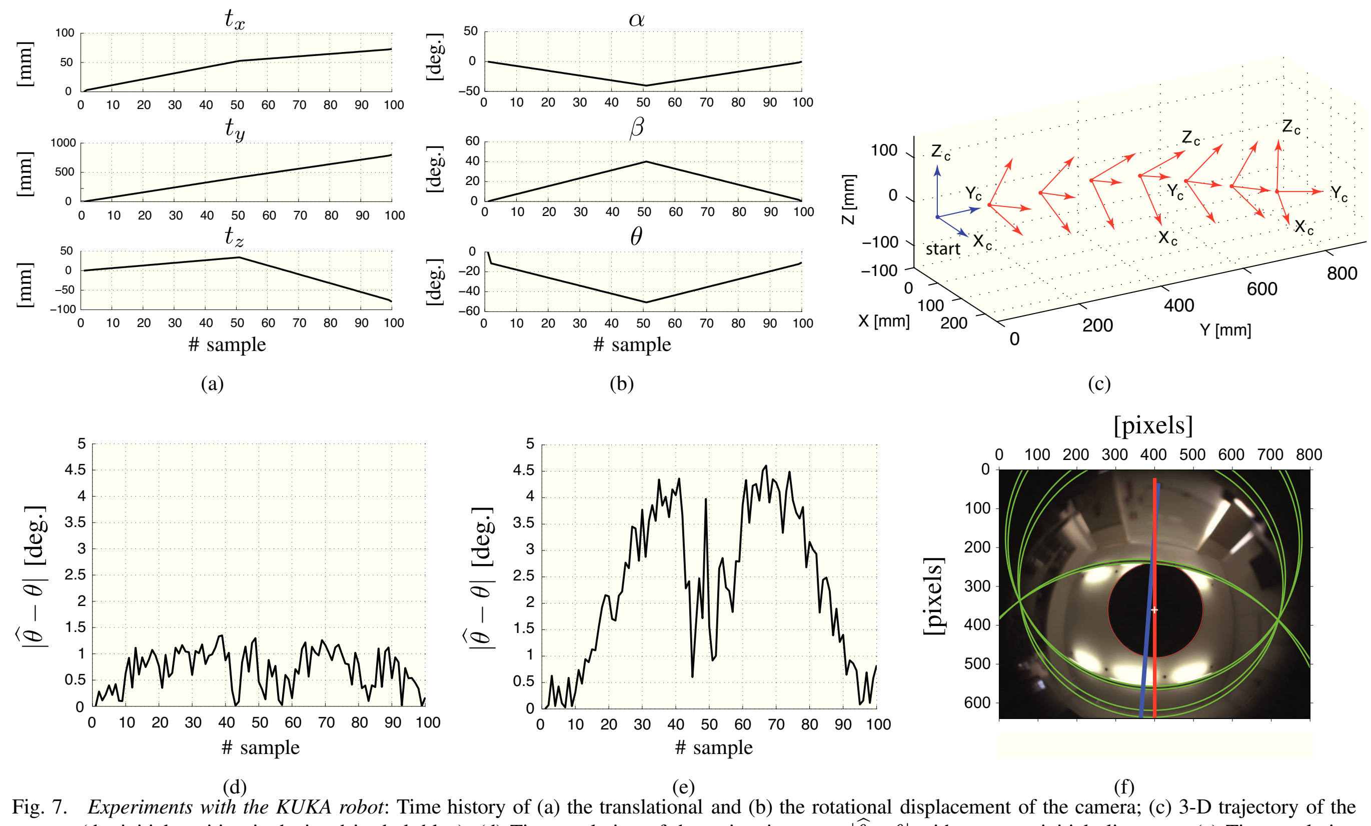


Fig. 7. *Experiments with the KUKA robot*: Time history of (a) the translational and (b) the rotational displacement of the camera; (c) 3-D trajectory of the camera (the initial position is depicted in dark blue); (d) Time evolution of the estimation error $|\widehat{\theta} - \theta|$ with accurate initial alignment; (e) Time evolution of the estimation error $|\widehat{\theta} - \theta|$ when a spurious rotation of 4 deg. about the x-, y-, z-axes was introduced in the initial camera pose: (f) the misalignment can be easily noticed in the image (the blue line joining the circles' centers and the red line should coincide).

plane, which are widely available in urban environments. Simulations as well as real-world experiments have demonstrated the effectiveness of our algorithm, and its potential for attitude estimation in aerial robotics.

Although our results are promising, an effort still needs to be done in order to implement our VC algorithm onboard a real MAV. In particular, it will be necessary to tailor our algorithm to the stringent memory and computational requirements of commercial MAVs, and to make it robust against propellers-induced vibrations. Finally, work is in progress to relax the starting alignment condition, in order to enlarge the set of possible initial camera orientations with respect to the environment.

References

[1] R. Beard, D. Kingston, M. Quigley, D. Snyder, R. Christiansen, W. Johnson, T. McLain, and M. Goodrich. Autonomous vehicle technologies for small fixed wing UAVs. *AIAA J. Aerospace Comp. Inf. Comm.*, 2(1):92–108, 2005.

[2] F. Morbidi and G.L. Mariottini. On Active Target Tracking and Cooperative Localization for Multiple Aerial Vehicles. In *Proc. IEEE/RSJ Int. Conf. Intel. Robots Syst*, pages 2229–2234, 2011.

[3] M. Schwager, B.J. Julian, M. Angermann, and D. Rus. Eyes in the sky: Decentralized control for the deployment of robotic camera networks. *Proc. of the IEEE*, 99(9):1541–1561, 2011.

[4] V. Kumar and N. Michael. Opportunities and challenges with autonomous micro aerial vehicles. *Int. J. Robot. Res.*, 31(11):1279–1291, 2012.

[5] L. Doitsidis, S. Weiss, A. Renzaglia, M.W. Achtelik, E. Kosmatopoulos, R. Siegwart, and D. Scaramuzza. Optimal surveillance coverage for teams of micro aerial vehicles in GPS-denied environments using onboard vision. *Auton. Robot.*, 33:173–188, 2012.

[6] "Klangwolke". http://spectrum.ieee.org/automaton/robotics/robotics-hardware/fifty-quadrotors-put-on-glowing-sky-show-in-austria.

[7] "MIT Flyfire". http://senseable.mit.edu/flyfire/.

[8] S. Saripalli, J.F. Montgomery, and G.S. Sukhatme. Visually guided landing of an unmanned aerial vehicle. *IEEE Trans. Robot. Autom.*, 19(3):371–380, 2003.

[9] M. Blosch, S. Weiss, D. Scaramuzza, and R. Siegwart. Vision based mav navigation in unknown and unstructured environments. In *Proc. IEEE Int. Conf. Robot. Automat*, pages 21–28, 2010.

[10] S. Thurrowgood, D. Soccol, R. Moore, D. Bland, and M.V. Srinivasan. A vision based system for attitude estimation of UAVs. In *Proc. IEEE/RSJ Int. Conf. Intel. Robots Syst*, pages 5725–5730, 2009.

[11] C. Demonceaux, P. Vasseur, and C. Pégard. Omnidirectional vision on UAV for attitude computation. In *Proc. IEEE Int. Conf. Robot. Automat*, pages 2842–2847, 2006.

[12] C. Demonceaux, P. Vasseur, and C. Pégard. UAV attitude computation by omnidirectional vision in urban environment. In *Proc. IEEE Int. Conf. Robot. Automat*, pages 2017–2022, 2007.

[13] J.-C. Bazin, C. Demonceaux, P. Vasseur, and I. Kweon. Rotation estimation and vanishing point extraction by omnidirectional vision in urban environment. *Int. J. Robot. Res.*, 31(1):63–81, 2012.

[14] I.F. Mondragón, P. Campoy, C. Martinez, and M. Olivares. Omnidirectional vision applied to unmanned aerial vehicles (uavs) attitude and heading estimation. *Robot. Autonom. Syst.*, 58(6):809–819, 2010.

[15] G.L. Mariottini, S. Scheggi, F. Morbidi, and D. Prattichizzo. An Accurate and Robust Visual-Compass Algorithm for Robot-mounted Omnidirectional Cameras. *Robot. Autonom. Syst.*, 60(9):1179–1190, 2012.

[16] C. Geyer and K. Daniilidis. A unifying theory for central panoramic systems. In *6th European Conf. Comput. Vis.*, pages 445–462, 2000.

[17] J.P. Barreto and H. Araujo. Geometric Properties of Central Catadioptric Line Images and their Application in Calibration. *IEEE Trans. Pattern Anal.*, 27(8):1327–1333, 2005.

[18] C. Geyer and K. Daniilidis. Catadioptric Camera Calibration. In *7th Int. Conf. Comput. Vis.*, volume 1, pages 398–404, 1999.

[19] E. Marchand, F. Spindler, and F. Chaumette. ViSP for visual servoing: a generic software platform with a wide class of robot control skills. *IEEE Rob. Autom. Mag.*, 12(4):40–52, 2005.