Monocular Eye-in-Hand Robotic Ball Catching with Parabolic Motion Estimation

Vincenzo Lippiello and Fabio Ruggiero

PRISMA Lab, Dipartimento di Informatica e Sistemistica, Università degli Studi di Napoli Federico II, via Claudio 21, 80125, Naples, Italy (e-mail: vincenzo.lippiello@unina.it, fabio.ruggiero@unina.it).

Abstract: A monocular robotic ball catching system is presented in this paper. A visual servoing control is used in order to keep a thrown ball in the field of view of an eye-in-hand camera and generate a robot motion toward the ball interception point. Simultaneously, the 3D ball trajectory is estimated with a least-squares method, which employs only information provided by the available camera. Experiments demonstrate the feasibility of the proposed approach.

Keywords: Robot vision. Visual motion. Path planning. Motion estimation. Image processing.

1. INTRODUCTION

In the robotic research community, catching a thrown object with a robotic system represents a challenging task which requires a combination of several capabilities: smart sensing, object tracking, trajectory estimation, online motion planning, fine motion coordination.

Frese et al. (2001) exploit a stereo vision system with a large baseline, an extended Kalman filter (EKF) and a ball trajectory predictor in order to build a robotic ball catcher. Imai et al. (2004) and Namiki and Ishikawa (2005) employ a high-speed multi-fingered hand and a high-speed stereo vision system to catch a falling ball and a falling cylinder. On the other hand, Deguchi et al. (2008) employ 2D data given by a visual stereo system: the 3D task is considered achieved when all the 2D tasks defined for all the images are simultaneously fulfilled.

Several papers make use of Chapman's strategy – the fielder should run at a proper speed to maintain a constant increasing rate of the ball's elevation angle tangent– to catch a ball (Chapman (1968)). Das and Das (1994) use reinforcement learning models in order to better understand both how to keep constant the increasing rate of the elevation-angle tangent and how to use the velocity of the ball perpendicular to the fielder so as to decide whether to run forward or backward. Mori and Miyazaki (2002a) consider an autonomous mobile robot for a ball catching task using a visual feedback control method based on a *Linear Optimal Trajectory* strategy. Mori and Miyazaki (2002b) propose an alternative strategy still based on Chapman's hypothesis, called *Gaining Angle of Gaze*, which requires only the information about the elevation angle of gaze captured as a 2D information.

Most approaches use either a stereo visual system to solve the 3D catching problem or a single camera for the 2D case. This scenario is reasonable because 3D tracking of the ball takes benefits from triangulation methods while, in the case of a single camera, only 2D information is directly available. However, a high frame rate and optics with a good accuracy are required to achieve an accurate and fast trajectory prediction, i.e. a successful catch. By using only one camera the cost of the equipment can be reduced. Moreover, the calibration procedure for one camera is easier than in the stereo case. Ribnick et al. (2009) estimate the 3D state of a thrown object by using a least-squares solution starting from a sequence of images given by a single camera. Further, Herrejon et al. (2009) employ a combination of image-based and position-based visual servoing with an eye-to-hand camera to catch a ball whose trajectory is estimated through a RLS algorithm.

A new monocular eye-in-hand robotic 3D ball catching is proposed in this paper, where a robot manipulator equipped with a standard CCD camera is driven by a visual servoing control in order to track and catch the ball. Unlike the works performed by Ribnick et al. (2009) and Herrejon et al. (2009), here the camera is mounted in an eye-in-hand configuration, which is a common configuration in robotics due to its high flexibility for visual servoing tasks. (Siciliano et al., 2008) explain pros and cons of using either an eye-to-hand or eye-in-hand visual configuration. The adopted control algorithm is composed of two phases – vision-based ball interception and estimation-based catching–, which will be described in the following sections. Experimental results performed using common available hardware demonstrate the effectiveness of the proposed solution.

2. ALGORITHM OVERVIEW

The proposed ball catching algorithm can be divided into two main phases, namely *vision-based ball interception* and *estimation-based catching*. First, when the ball is thrown by the pitcher, the robot is guided to follow the movements of the ball through the visual information provided by the camera mounted in eye-in-hand configuration. A visual servoing control keeps the centroid of the ball in the camera field of view and, at the same time, generates a robot

motion in a direction feasible with the ball interception. Visual measures are then elaborated in order to continuously update the prediction of the ball trajectory, here approximated as a parabola. When a stable estimation is available, the control switches from a visual servoing mode into a partitioned visual control mode. This last makes use of visual measures to keep the ball in the field of view, as well as to keep the orientation of the hand suitable for the catching, and at the same time to move the robot toward the predicted catching position. Moreover, by employing the visual data flow, a continuous refinement of the ball trajectory estimation, and hence of the catching position, is performed. This last is computed, without loss of generality, as the interception point between the estimated ball trajectory and a goal-plane, which is feasible with the robot kinematics, in the 3D space. Finally, according to the estimated catching time, the robotic hand closes its fingers in order to catch the ball with a timing depending on its own dynamics.

3. VISION-BASED BALL INTERCEPTION

3.1 Image processing

The image flow provided by an eye-in-hand calibrated camera is elaborated in order to measure the ball centroid. A dynamic windowing technique has been employed to reduce the portion of the image that has to be elaborated, so as to reduce the computational complexity of the image elaboration process and to facilitate the ball matching.

An equalized color-based clustering, which makes use of the *Hue, Lightness, and Saturation Color Space* (HSL), has been designed in order to mitigate the problems due to the environmental lightness uniformity. In details, a binarization process is performed through an equalized test, which makes use of a histogram of the H-channel centered around the desired ball color, together with a min/max S-channel threshold. After some post-elaboration processes employed to reduce the image noise, all the blobs present in the image are collected and filtered so as to eliminate, for example, all the blobs with a not significant area. If more than one blob overcomes this filtering process, a neighborhood selection method with respect to the predicted ball position and dimension is adopted. Finally, the centroid of the selected blob is evaluated as an approximation of the center of the ball.

This algorithm can also be executed to find other relevant objects in the environment. For example, a color tuned on the hand of the pitcher is also employed, which allows recognizing up to a fully hand occlusion whether the ball is held in the hand or it has been thrown, avoiding in such a way unnecessary initial movements of the robot.

3.2 Vision-based ball interception control

Without loss of generality, the camera frame is considered coincident with the hand (end-effector) frame, with the optical axis aligned with the hand approaching axis. Being the camera fixed with respect to the hand in an eye-in-hand camera configuration, a fixed transformation should be considered in the cases where these reference frames are not coincident. Therefore, in the remainder of the paper, only the camera frame will be addressed.

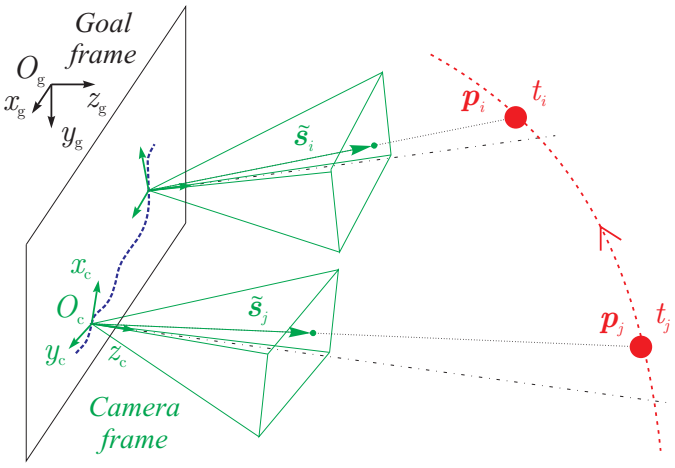

Fig. 1. Goal frame $O_g - x_g y_g z_g$ and camera frame $O_c - x_c y_c z_c$ shown in different sample times.

The proposed visual control law belongs to the category named *Resolved-Velocity Image-Based Visual Servoing* (Siciliano et al. (2008)), for which it is assumed that the manipulator dynamics is directly taken into account by the low-level robot controller.

In order to keep the ball in the field of view of the camera, the rotational components of the robot motion are mainly reserved to the ball tracking task, since with small movements of the camera orientation large parts of the scene can be observed. First, the normalized image coordinates of the ball centroid $\boldsymbol{s} = [\, X \;\; Y \,]^{\mathrm{T}}$, corresponding to a ball position $\boldsymbol{p}_o^c$ with respect to the camera frame $\Sigma_c : O_c - x_c y_c z_c$ (see Fig. 1), are considered

$$\boldsymbol{p}_o^c = [\, x^c \;\; y^c \;\; z^c \,]^{\mathrm{T}} = z^c [\, X \;\; Y \;\; 1 \,]^{\mathrm{T}} = z^c \tilde{\boldsymbol{s}}, \tag{1}$$

where $\tilde{\boldsymbol{s}} = \begin{bmatrix} \boldsymbol{s}^{\mathrm{T}} & 1 \end{bmatrix}^{\mathrm{T}}$. The absolute velocity of the camera $\boldsymbol{v}_c^c = \begin{bmatrix} \dot{\boldsymbol{p}}_c^{c\mathrm{T}} & \boldsymbol{\omega}_c^{c\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$ and the absolute velocity of the thrown object $\boldsymbol{v}_o^c = \begin{bmatrix} \dot{\boldsymbol{p}}_o^{c\mathrm{T}} & \boldsymbol{\omega}_o^{c\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$, both expressed with respect to Σ_c, and the velocity of the observed image feature $\dot{\boldsymbol{s}}$ in the image plane are related by the following linear equation (Siciliano et al. (2008))

$$\dot{\boldsymbol{s}} = \boldsymbol{L}_s \boldsymbol{v}_c^c + \boldsymbol{L}_s \boldsymbol{\Gamma}(-\boldsymbol{p}_o^c) \boldsymbol{v}_o^c, \tag{2}$$

where $\boldsymbol{L}_s$ is the well known (2×6) interaction matrix, for a point image feature, defined as

$$\boldsymbol{L}_s = \begin{bmatrix} -1/z^c & 0 & X/z^c & XY & -1-X^2 & Y \\ 0 & -1/z^c & Y/z^c & 1+Y^2 & -XY & -X \end{bmatrix}, \tag{3}$$

while $\boldsymbol{\Gamma}(\cdot)$ is the following (6×6) matrix

$$\boldsymbol{\Gamma}(\cdot) = \begin{bmatrix} -\boldsymbol{I}_3 & \boldsymbol{S}(\cdot) \\ \boldsymbol{0} & -\boldsymbol{I}_3 \end{bmatrix},$$

in which $\boldsymbol{I}_n$ denotes the $(n \times n)$ identity matrix and $\boldsymbol{S}(\cdot)$ the skew-symmetric matrix.

Hence, by denoting with $\boldsymbol{L}_{sp}$ and $\boldsymbol{L}_{so}$ the (2×3) submatrices corresponding to the first and last three columns of (3), the linear and angular velocity inputs for the camera frame are initially generated by using only visual measurements with the following control laws:

$$\dot{\boldsymbol{p}}_c^c = \hat{\boldsymbol{L}}_{sp}^{\dagger} \boldsymbol{K}_{sp} \boldsymbol{e}_s + \hat{\dot{\boldsymbol{p}}}_o^c - \boldsymbol{S}(-\hat{\boldsymbol{p}}_o^c) \hat{\boldsymbol{\omega}}_o^c \tag{4a}$$

$$\boldsymbol{\omega}_c^c = \boldsymbol{L}_{so}^{\dagger} \boldsymbol{K}_{so}(\boldsymbol{e}_s) \boldsymbol{\tau}_{eb_1}(\boldsymbol{e}_s) + \hat{\boldsymbol{\omega}}_o^c \tag{4b}$$

where the symbol † denotes the pseudo-inverse of a matrix, $\hat{\boldsymbol{p}}_o^c$ is the estimation of the unknown distance between the ball and the object expressed in Σ_c, while $\hat{\dot{\boldsymbol{p}}}_o^c$ and $\hat{\boldsymbol{\omega}}_o^c$ are the estimations of both the unknown absolute linear and angular velocities of the ball with respect to Σ_c. Moreover, only $\boldsymbol{L}_{sp}$ needs to be estimated since it depends on the third component of the unknown $\boldsymbol{p}_o^c$, while $\boldsymbol{L}_{so}$ depends on the available visual measurements. The stability analysis of the control laws (4) is described in the Appendix A, while the details about the estimation of the employed unknown parameters are presented in the Appendix B.

Equations (4) are composed of two terms: (4a) generates the translational camera velocity moving the hand so as to zero the error in the image plane, while (4b) generates a rotational camera velocity keeping the ball in the field of view. In (4a), $\boldsymbol{K}_{sp}$ is a diagonal constant (2×2) gain matrix, $\boldsymbol{e}_s = -\boldsymbol{s}$ is the image error vector that becomes zero when the camera is pointed toward the centroid of the ball. In (4b), instead, $\boldsymbol{\tau}_{eb_1}(\boldsymbol{e}_s)$ is a threshold function defined as follows

$$\boldsymbol{\tau}_{eb_1}(\boldsymbol{e}_s) = \begin{cases} \boldsymbol{0} & \text{if } \|\boldsymbol{e}_s\| \leq e_{b1} \\ \left(1 - \dfrac{e_{b1}}{\|\boldsymbol{e}_s\|}\right) \boldsymbol{e}_s & \text{if } \|\boldsymbol{e}_s\| > e_{b1}, \end{cases} \tag{5}$$

and $\boldsymbol{K}_{so}(\boldsymbol{e}_s)$ is a (2×2) gain matrix defined as follows

$$\boldsymbol{K}_{so}(\boldsymbol{e}_s) = \begin{cases} k_o \boldsymbol{I}_2 & \text{if } \|\boldsymbol{e}_s\| \leq e_{b2} \\ k_o e^{\beta_o \left(\frac{\|\boldsymbol{e}_s\|}{e_{b2}} - 1\right)} \boldsymbol{I}_2 & \text{if } e_{b2} < \|\boldsymbol{e}_s\| \leq e_{b3} \\ k_o e^{\beta_o \left(\frac{e_{b3}}{e_{b2}} - 1\right)} \boldsymbol{I}_2 & \text{if } \|\boldsymbol{e}_s\| > e_{b3} \end{cases} \tag{6}$$

where $e_{b3} > e_{b2} > e_{b1} > 0$, $k_o > 0$ is a gain factor and $\beta_o > 0$ is a barrier factor that tunes the increasing rate of $\boldsymbol{K}_{so}$ in accordance to the image plane limits. In this way, when the ball centroid approaches the image plane limits, the gain $\boldsymbol{K}_{so}$ rapidly increases in order to avoid the lost of the ball view. On the other hand, the threshold e_{b1} applied on the rotational error in (5) avoids that the ball centroid can be brought to zero only by rotational movements of the robot, leaving this task to the position control.

If the robot dynamics are sufficiently fast, then an interception trajectory should be followed. Notice that, as it will be explained in the next section, it is important that the robot moves the camera in such a way as to create a significant baseline so as to increase the robustness for what concerns the trajectory estimation process.

The joints velocity input to the robot control unit can be finally evaluated as (Siciliano et al. (2008))

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{\dagger} \boldsymbol{T}_c \boldsymbol{v}_c^c + \boldsymbol{N}_J \boldsymbol{K}_r \dot{\boldsymbol{q}}_r, \tag{7}$$

where $\boldsymbol{J}$ is the robot Jacobian matrix, which provides the relationship between the joints and the camera velocity, $\boldsymbol{T}_c$ is the (6×6) matrix which relates the velocity of the camera with respect to the camera frame to the velocity of the robot end-effector with respect to the base frame, $\boldsymbol{N}_J$ denotes a projector matrix onto the null space of $\boldsymbol{J}$, $\boldsymbol{K}_r$ is a gain diagonal matrix, and $\dot{\boldsymbol{q}}_r$ is a set of joint velocities exploited in the redundancy management in order to optimize some sub-tasks. In detail, $\dot{\boldsymbol{q}}_r$ is generated in a way as to avoid joint limits and kinematic singularities.

4. ESTIMATION-BASED CATCHING

4.1 Ball motion estimation

Initially, under the action of the previous visual control laws (4), a sequence of image measurements is collected. With reference to Fig. 1, let $\Sigma_g = O_g - x_g y_g z_g$ be a fixed goal frame in the space. By denoting with t_i the i-th visual sample time, $\tilde{\boldsymbol{s}}_i$ the corresponding acquired image feature vector and with the apex g the quantities expressed in such goal frame, the points $\boldsymbol{p}^g = [x^g \ y^g \ z^g]^{\mathrm{T}}$ which belong to the *optical ray* passing through the current origin of the camera $\boldsymbol{c}_i^g = \left[c_{x,i}^g \ c_{y,i}^g \ c_{z,i}^g\right]^{\mathrm{T}}$ and the i-th feature vector $\boldsymbol{r}_i^g = \left[r_{x,i}^g \ r_{y,i}^g \ r_{z,i}^g\right]^{\mathrm{T}} = \boldsymbol{c}_i^g + \boldsymbol{R}_{c,i}^g \tilde{\boldsymbol{s}}$, can be defined with the following straight line equations

$$\begin{cases} (r_{y,i}^g - c_{y,i}^g)x^g + (c_{x,i}^g - r_{x,i}^g)y^g + r_{x,i}^g c_{y,i}^g - r_{y,i}^g c_{x,i}^g = 0 \\ (r_{z,i}^g - c_{z,i}^g)x^g + (c_{x,i}^g - r_{x,i}^g)z^g + r_{x,i}^g c_{z,i}^g - r_{z,i}^g c_{x,i}^g = 0, \end{cases} \tag{8}$$

with $\boldsymbol{R}_{c,i}^g$ representing the rotation matrix of the camera frame at time t_i with respect to Σ_g. Notice that both $\boldsymbol{c}_i^g$ and $\boldsymbol{R}_{c,i}^g$ are provided by the robot direct kinematics.

By supposing negligible the effects of the air drag, e.g. by adopting no soft-balls and no fast initial velocities (Frese et al. (2001)), the ball trajectory can be described as a parabolic function of the time t with the following equation

$$\boldsymbol{p}^g = \boldsymbol{p}_0^g + \dot{\boldsymbol{p}}_0^g t + \frac{1}{2}\boldsymbol{g}^g t^2, \tag{9}$$

where $\boldsymbol{g}^g$ is the gravity acceleration, and $\boldsymbol{p}_0^g$ and $\dot{\boldsymbol{p}}_0^g$ are the initial ball position and velocity (for $t = 0$), respectively, all expressed with respect to Σ_g. Without loss of generality, the gravity acceleration is considered aligned to the axis y_g of Σ_g, i.e. $\boldsymbol{g}^g = [0 \ g \ 0]^{\mathrm{T}}$, with $g = 9.81\mathrm{m/s}^2$.

The optical rays measured at each visual sample time t_i intersect the ball trajectory at the same time. Hence, by replacing (9) into (8), the following system of 2 equations in the 6 unknowns $\boldsymbol{p}_0^g$ and $\dot{\boldsymbol{p}}_0^g$, which fully identify the ball trajectory, can be achieved

$$\boldsymbol{A}_i \begin{bmatrix} \boldsymbol{p}_0^g \\ \dot{\boldsymbol{p}}_0^g \end{bmatrix} = \boldsymbol{b}_i, \tag{10}$$

where

$$\boldsymbol{A}_i = \begin{bmatrix} r_{y,i}^g - c_{y,i}^g & c_{x,i}^g - r_{x,i}^g & 0 & \cdots \\ r_{z,i}^g - c_{y,i}^g & 0 & c_{x,i}^g - r_{x,i}^g & \cdots \\ (r_{y,i}^g - c_{y,i}^g)t_i & (c_{x,i}^g - r_{x,i}^g)t_i & 0 & \\ (r_{z,i}^g - c_{y,i}^g)t_i & 0 & (c_{x,i}^g - r_{x,i}^g)t_i & \end{bmatrix}$$

$$\boldsymbol{b}_i = \begin{bmatrix} r_{y,i}^g c_{x,i} - r_{x,i}^g c_{y,i} - \dfrac{1}{2}(c_{x,i} - r_{x,i}^g) g t_i^2 \\ r_{z,i}^g c_{x,i} - r_{x,i}^g c_{z,i} \end{bmatrix}.$$

By stacking into rows the n measurements $\boldsymbol{A}_i$ and $\boldsymbol{b}_i$ collected during the time, a least-squares solution is considered for the system of n equations and 6 unknowns

$$\boldsymbol{A} \begin{bmatrix} \boldsymbol{p}_0^g \\ \dot{\boldsymbol{p}}_0^g \end{bmatrix} = \boldsymbol{b}. \tag{11}$$

A weighted pseudo-inverse of the matrix $\boldsymbol{A}$ is employed in order to increase the relevance of the last measurements with respect to the first ones, since they are less affected by the negligence of the air drag (the velocity of the ball is smaller at the end of the trajectory) and they

are characterized by a higher image resolution (the ball is closer to the camera).

The assumptions about the knowledge of the ball trajectory model and of the gravity vector implicitly solve the scaling factor problem, due to the use of 2D visual data, especially for the y component. Moreover, the acquisition of measurements from several camera positions also contribute to the well conditioning of the last square problem.

4.2 Partitioned visual approach

The trajectory estimation is continuously improved by adding new available measures to (11) . When the sequence of solutions ($\hat{\boldsymbol{p}}_0^g$, $\hat{\dot{\boldsymbol{p}}}_0^g$) is stable enough, the intersection between the ball trajectory and the goal plane, as well as the catching time, are evaluated. The adoption of a goal plane is only one of the possible choices. Since the whole ball trajectory is available, other catching point/strategies in the 3D robot workspace can be considered, e.g. the catching point which requires the minimum effort in terms of joints velocity (Batz et al. (2010)).

Once that a catching point is available, the control law smoothly switches into a partitioned visual control (Deguchi (1998)). The position of the robot is led toward the estimated catching position, while the robot orientation is controlled, as before, in order to keep the ball in the field of view and to rightly orientate the hand.

In detail, a fifth-order polynomial is considered in order to compute the desired trajectory position for the camera in the 3D Cartesian space

$$\boldsymbol{p}_{c,d}^g(t) = \boldsymbol{a}_5 t^5 + \boldsymbol{a}_4 t^4 + \boldsymbol{a}_3 t^3 + \boldsymbol{a}_2 t^2 + \boldsymbol{a}_1 t + \boldsymbol{a}_0, \quad (12)$$

where $\boldsymbol{p}_{c,d}$ is the (3×1) desired catching position of the camera with respect to Σ_g, and $\boldsymbol{a}_h$ with $h = 0, \ldots, 5$ are the (3×1) coefficient vectors. Hence, the translational components of the absolute desired camera velocity and acceleration, both expressed with respect to Σ_g, are

$$\dot{\boldsymbol{p}}_{c,d}^g = 5\boldsymbol{a}_5 t^4 + 4\boldsymbol{a}_4 t^3 + 3\boldsymbol{a}_3 t^2 + 2\boldsymbol{a}_2 t + \boldsymbol{a}_1 \quad (13a)$$

$$\ddot{\boldsymbol{p}}_{c,d}^g = 20\boldsymbol{a}_5 t^3 + 12\boldsymbol{a}_4 t^2 + 6\boldsymbol{a}_3 t + 2\boldsymbol{a}_2. \quad (13b)$$

Starting from the current robot state (position, velocity and acceleration), the parameters $\boldsymbol{a}_h$ are computed to reach the estimated catching position, with null velocity and acceleration, at the estimated catching time. Hence, by denoting with t_i, t_f, $\boldsymbol{p}_{c,d,i}^g$, $\boldsymbol{p}_{c,d,f}^g$, $\dot{\boldsymbol{p}}_{c,d,i}^g$, $\dot{\boldsymbol{p}}_{c,d,f}^g$, $\ddot{\boldsymbol{p}}_{c,d,i}^g$ and $\ddot{\boldsymbol{p}}_{c,d,f}^g$ the initial and final planned time, position, linear velocity and acceleration, respectively, with $\bar{\boldsymbol{a}}_h = \begin{bmatrix}\boldsymbol{a}_5^{\mathrm{T}} & \boldsymbol{a}_4^{\mathrm{T}} & \boldsymbol{a}_3^{\mathrm{T}} & \boldsymbol{a}_2^{\mathrm{T}} & \boldsymbol{a}_1^{\mathrm{T}} & \boldsymbol{a}_0^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$, and by taking into account (12) and (13), the following linear system is obtained

$$\begin{bmatrix} \boldsymbol{t}_i^5 & \boldsymbol{t}_i^4 & \boldsymbol{t}_i^3 & \boldsymbol{t}_i^2 & \boldsymbol{t}_i & \boldsymbol{i}_3 \\ \boldsymbol{t}_f^5 & \boldsymbol{t}_f^4 & \boldsymbol{t}_f^3 & \boldsymbol{t}_f^2 & \boldsymbol{t}_f & \boldsymbol{i}_3 \\ 5\boldsymbol{t}_i^4 & 4\boldsymbol{t}_i^3 & 3\boldsymbol{t}_i^2 & 2\boldsymbol{t}_i & \boldsymbol{i}_3 & \boldsymbol{0_3} \\ 5\boldsymbol{t}_f^4 & 4\boldsymbol{t}_f^3 & 3\boldsymbol{t}_f^2 & 2\boldsymbol{t}_f & \boldsymbol{i}_3 & \boldsymbol{0_3} \\ 20\boldsymbol{t}_i^3 & 12\boldsymbol{t}_i^2 & 6\boldsymbol{t}_i & 2\boldsymbol{i}_3 & \boldsymbol{0_3} & \boldsymbol{0_3} \\ 20\boldsymbol{t}_f^3 & 12\boldsymbol{t}_f^2 & 6\boldsymbol{t}_f & 2\boldsymbol{i}_3 & \boldsymbol{0_3} & \boldsymbol{0_3} \end{bmatrix} \bar{\boldsymbol{a}}_h = \begin{bmatrix} \boldsymbol{p}_{c,d,i}^g \\ \boldsymbol{p}_{c,d,f}^g \\ \dot{\boldsymbol{p}}_{c,d,i}^g \\ \dot{\boldsymbol{p}}_{c,d,f}^g \\ \ddot{\boldsymbol{p}}_{c,d,i}^g \\ \ddot{\boldsymbol{p}}_{c,d,f}^g \end{bmatrix},$$

where $\boldsymbol{i}_3 = [1\ 1\ 1]$, $\boldsymbol{t}_i = t_i \boldsymbol{i}_3$, $\boldsymbol{t}_f = t_f \boldsymbol{i}_3$ and $\boldsymbol{0_3} = \boldsymbol{0}\boldsymbol{i}_3$. By solving the previous system for $\bar{\boldsymbol{a}}_h$ is thus possible to obtain the desired trajectories (12), (13a) and (13b). The

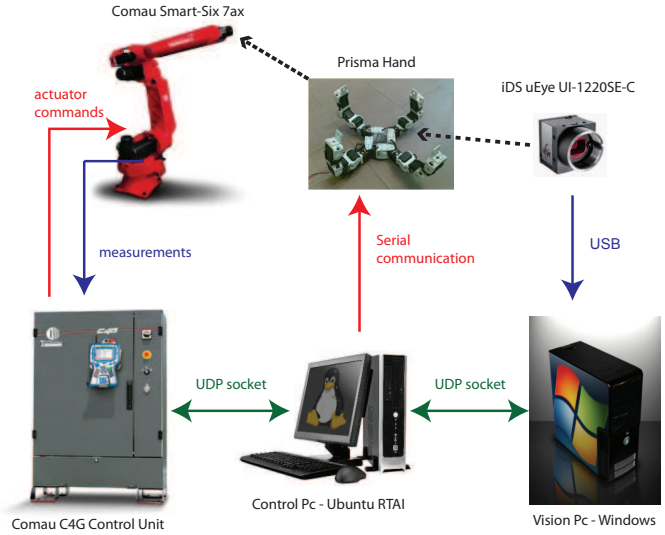

Fig. 2. Architecture of the ball catching system.

translational velocity input for the camera frame can thus be generated as

$$\dot{\boldsymbol{p}}_c^c = \boldsymbol{R}_g^c (\dot{\boldsymbol{p}}_{c,d}^g + \boldsymbol{K}_p \boldsymbol{e}_p), \quad (14)$$

where $\boldsymbol{K}_p$ is a diagonal constant (3×3) gain matrix and $\boldsymbol{e}_p$ is the (3×1) error vector between the desired planned trajectory (12) and the current one, that is provided by the robot direct kinematics and expressed with respect to Σ_g. On the other hand, the control input $\boldsymbol{\omega}_c^c$ in (4b) is modified as follows

$$\begin{aligned} \boldsymbol{\omega}_c^c = \boldsymbol{L}_{so}^{\dagger} \big[& \boldsymbol{K}_{so,e_{b2}}(\boldsymbol{e}_s) \boldsymbol{\tau}_{e_{b1}}(\boldsymbol{e}_s) \\ & - \hat{\boldsymbol{L}}_{sp} \left(\dot{\boldsymbol{p}}_c^c - \hat{\dot{\boldsymbol{p}}}_o^c + \boldsymbol{S}(-\hat{\boldsymbol{p}}_o^c) \hat{\boldsymbol{\omega}}_o^c \right) \big] + \hat{\boldsymbol{\omega}}_o^c, \end{aligned} \quad (15)$$

where $\dot{\boldsymbol{p}}_c^c$ is evaluated in (14). The stability analysis of the control laws (14) and (15) is similar to the one described in the Appendix A. It is here omitted for brevity, but maybe obtained from the authors by request.

When a new estimation is available, starting from the current state of the camera frame, the parameters $\boldsymbol{a}_h$ can be tuned again so as to reach the new estimated catching position in the new estimated catching time.

5. EXPERIMENTS

5.1 Experimental setup

Figure 2 shows the experimental set-up implementing the proposed control algorithm. A Comau Smart-Six robot manipulator mounted on a sliding track and equipped with a 4-fingered hand using 16 Dynamixel AX-12 servomotors has been employed. The Comau C4G control unit compensates the dynamic model of the robot, while an external PC with Ubuntu OS patched with RTAI-real time kernel generates the position/orientation references at 2 ms. The control PC communicates with a second Windows OS PC that is responsible of the visual elaboration. An industrial USB iDS UEYE UI-1220SE-C camera has been mounted, behind a transparent plexiglass, directly in the center of the palm of the hand. A multi-thread programming together with the synchronization signal provided by the camera have been employed to improve the stability of the elaboration time and synchronize the visual measurements with the robot motion.

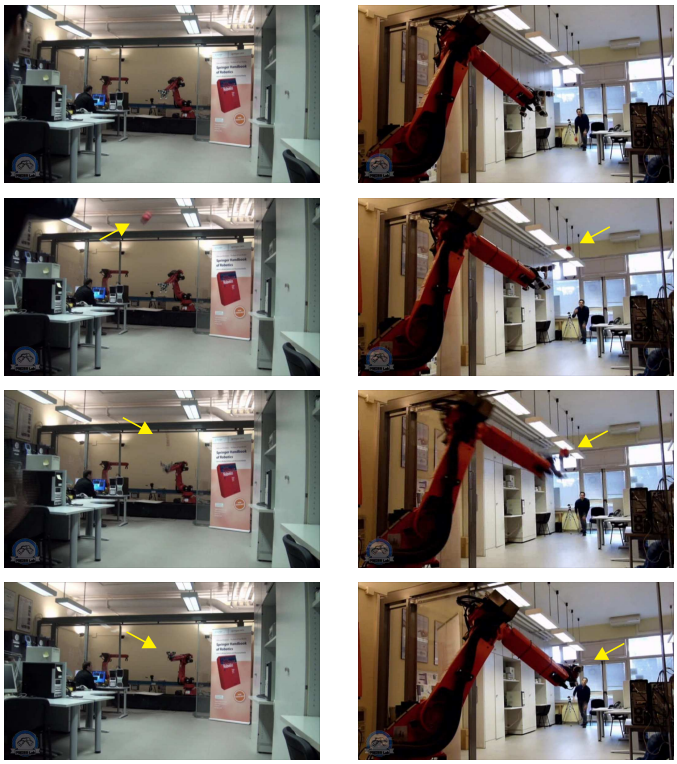

Fig. 3. Snapshots of a catch from different points of view.

In order to increase the acquisition frame rate, which affects the overall performances of the algorithm, an image size of (375×500) pixels and a dynamic RoI windowing with a dimension of (150×150) pixels have been employed: the achieved visual sampling frequency is 112 fps. A ball with a radius of 6 cm and a weight of 150 g has been used.

The control gains of the ball tracking algorithm have been tuned to $\boldsymbol{K}_{sp} = 50\boldsymbol{I}_2$ and $k_o = 6.77$, with $e_{b1} = 10$ pixels, $e_{b2} = 100$ pixels and $e_{b3} = 300$ pixels and $\beta_o = 1$. The matrix gain $\boldsymbol{K}_p$ in (14) has been set to $50\boldsymbol{I}_3$, while the redundancy management in (7) has been employed in order to avoid joint limits and kinematic singularities and to minimize the sliding track motions (the slowest one).

The trajectory estimation starts when a number of at least $n = 50$ samples have been collected and the robot end-effector has moved for at least 10 cm with respect to the initial position. Then, for each new visual measurement, new estimations $\hat{\boldsymbol{p}}_0^g$ and $\hat{\dot{\boldsymbol{p}}}_0^g$ are computed.

5.2 Results

Several experiments have been carried out with different light conditions and pitchers. In detail, two pitchers threw 50 shots in total, while the initial distance between the camera and the pitcher belongs to a range of $6 - 6.5$ m. and each throw takes about $0.7 - 0.9$ s. to intercept the goal plane. The percentage of ball interception evaluated over this shots set is 80%, while the percentage of catching is 50%. This difference is due to the poor performances of the available hand, which has slow dynamics. This last is partially compensated by starting in advance the closing movement on the basis of the estimated catching time, but some limitations remain. Other sources of inaccuracy are related to the negligence of the air drag factor and to

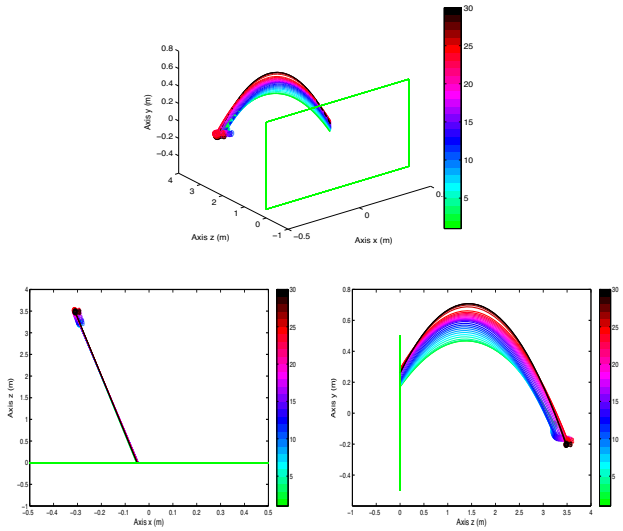

Fig. 4. Estimated trajectories with respect to Σ_g, for each new visual measurement (color gradient bar) over then the initial set of $n = 50$. *Top*: 3D space; *bottom-left*: $x_g z_g$-plane; *bottom-right*: $y_g z_g$-plane.

noisy visual measurements due to the quick change of the illumination conditions along the shot path. In Fig. 3 some snapshots of a catch are shown.

In Fig. 4 the estimated ball trajectories for the shot of Fig. 3 are shown. The trajectory in the $x_g z_g$-plane is already stable from the first estimation, while the trajectories of the ball projected in the other two planes require an adjustment time due to the neglected influence of the air drag. But, since the palm of the adopted hand is a square with a side-length of 11 cm., it is reasonable to consider a measure stable when the values coming out from the estimation process are inside the half of such dimension. This is achieved, for example, in the last 20 estimations of the trajectories in Fig. 4.

6. CONCLUSION

A new solution to cope with the problem of catching a thrown ball with only a single camera mounted on the robot end-effector in eye-in-hand configuration has been described. The effectiveness of the proposed algorithm has been demonstrated with experimental results.

REFERENCES

Batz, G., Yaqub, A., Wu, H., Kuhnlenz, K., Wollherr, D., and Buss, M. (2010). Dynamic manipulation: Nonprehensile ball catching. In *18th Mediterranean Conference on Control and Automation*. Marrakech.

Chapman, S. (1968). Catching a baseball. *American Journal of Physics*, 36(10), 868–870.

Chaumette, F. (1998). *Potential problems of stability and convergence in image-based and position-based visual servoing*. Springer, London.

Das, S. and Das, R. (1994). Using reinforcement learning to catch a baseball. In *IEEE International Conference on Neural Networks*. Orlando.

Deguchi, K. (1998). Optimal motion control for image-based visual servoing by decoupling translation and

rotation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Victoria.

Deguchi, K., Sakurai, H., and Ushida, S. (2008). A goal oriented just-in-time visual servoing for ball catching robot arm. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Nice.

Frese, U., Bauml, B., Haidacher, S., Schreiber, G., Schaefer, I., Hahnle, M., and Hirzinger, G. (2001). Off-the-shelf vision for a robotic ball catcher. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Maui.

Herrejon, R., Kagami, S., and Hashimoto, K. (2009). Composite visual servoing for catching a 3-D flying object using RLS trajectory estimation from a monocular image sequence. In *IEEE International Symposium on Computational Intelligence in Robotics and Automation*. Daejeon.

Imai, Y., Namiki, A., Hashimoto, K., and Ishikawa, M. (2004). Dynamic active catching using a high-speed multifingered hand and a high-speed vision system. In *IEEE International Conference on Robotics and Automation*. New Orleans.

Mori, R. and Miyazaki, F. (2002a). Examination of human ball catching strategy though autonomous mobile robot. In *IEEE International Conference on Robotics and Automation*. Washington.

Mori, R. and Miyazaki, F. (2002b). GAG (gaining angle of gaze) strategy for ball tracking and catching task. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Lausanne.

Namiki, A. and Ishikawa, M. (2005). The analysis of high-speed catching with a multifingered robot hand. In *IEEE International Conference on Robotics and Automation*. Barcelona.

Ribnick, E., Atev, S., and Papanikolopoulos, N. (2009). Estimating 3D positions and velocities of projectiles from monocular views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5), 938–944.

Siciliano, B., Sciavicco, L., Villani, L., and Oriolo, G. (2008). *Robotics. Modelling, Planning and Control*. Springer, London.

Appendix A. STABILITY ANALYSIS

An analysis about the convergence of the error $\boldsymbol{e}_s = -\boldsymbol{s}$ employed in (4) can be shown using the direct Lyapunov method based on the following candidate function

$$V(\boldsymbol{e}_s) = \boldsymbol{e}_s^{\mathrm{T}} \boldsymbol{K}_s \boldsymbol{e}_s > 0, \tag{A.1}$$

in which $\boldsymbol{K}_s$ is a (2×2) positive definite diagonal matrix.

By computing the time derivative of (A.1) along the trajectories of the system (2), and by using the control laws (4a) and (4b), it yields

$$\dot{V} = -\alpha_1 - \alpha_2 - \alpha_3, \tag{A.2}$$

where

$$\alpha_1 = \boldsymbol{e}_s^{\mathrm{T}} \boldsymbol{K}_s \boldsymbol{L}_{sp} \hat{\boldsymbol{L}}_{sp}^{\dagger} \boldsymbol{K}_{sp} \boldsymbol{e}_s \tag{A.3a}$$

$$\alpha_2 = \boldsymbol{e}_s^{\mathrm{T}} \boldsymbol{K}_s \boldsymbol{K}_{so}(\boldsymbol{e}_s) \boldsymbol{\tau}_{eb_1}(\boldsymbol{e}_s) \tag{A.3b}$$

$$\alpha_3 = \boldsymbol{e}_s^{\mathrm{T}} \boldsymbol{K}_s \left(\boldsymbol{L}_s \boldsymbol{\Gamma}(-\boldsymbol{p}_o^c) \boldsymbol{v}_o^c - \hat{\boldsymbol{L}}_s \boldsymbol{\Gamma}(-\hat{\boldsymbol{p}}_o^c) \hat{\boldsymbol{v}}_o^c \right). \tag{A.3c}$$

Hence, if each term in (A.3) is strictly positive then $\dot{V} < 0$.

By considering $\boldsymbol{K}_s = \boldsymbol{K}_{sp}$, α_1 in (A.3a) becomes a positive definite quadratic form if and only if $\boldsymbol{L}_{sp} = \hat{\boldsymbol{L}}_{sp}$. As stated by (Chaumette, 1998), the condition in which $\hat{\boldsymbol{L}}_{sp}$ is updated through the estimation of $\hat{z}^c$, i.e. the estimation of $\hat{\boldsymbol{p}}_o^c$, seems to be the optimal one in order to satisfy this condition during the experiments. Even if this condition is not strictly satisfied, it has been demonstrated that this term is anyway bounded.

By taking into account (5) and (6), it is possible to write the following bounds for α_2 in (A.3b)

$$0 \leq \alpha_2 \leq \boldsymbol{e}_s^{\mathrm{T}} \boldsymbol{K}_s \left(k_o e^{\beta_0 \left(\frac{e_{b3}}{e_{b2}} - 1 \right)} \right) \boldsymbol{I}_2 \boldsymbol{e}_s. \tag{A.4}$$

By choosing $\boldsymbol{K}_s = \boldsymbol{K}_{sp} = \left(k_o e^{\beta_0 \left(\frac{e_{b3}}{e_{b2}} - 1 \right)} \right) \boldsymbol{I}_2$, the last term of (A.4) becomes a positive definite quadratic form and this means that α_2 is always positive and limited.

In case of perfect estimation α_3 in (A.3c) vanishes, otherwise other cases do not cancel this term and then nothing can be said about its sign. However, both (A.3a) and (A.3b) are quadratic form of the error $\boldsymbol{e}_s$, while (A.3c) is a linear function of the error. Therefore, for an error of small norm, the linear term prevails over the quadratic terms, hence the norm of $\boldsymbol{K}_s$ should be increased in order to reduce the error as much as possible. Otherwise, for larger errors, the quadratic terms prevail over the linear terms. Anyway, the error $\boldsymbol{e}_s$ is norm-bounded.

In conclusion, in case of perfect compensation, by excluding interaction matrix singularities, the chosen control laws lead to an asymptotic stable system; while, in other cases, the error in the image plane is bounded. For the considered ball-catching application this stability condition can be considered sufficient, because the visual control aim is mainly in keeping the ball in the camera field of view.

Appendix B. ESTIMATED PARAMETERS

Starting form the estimated initial ball position $\hat{\boldsymbol{p}}_0^g$ and velocity $\hat{\dot{\boldsymbol{p}}}_0^g$, in order to obtain $\hat{\boldsymbol{p}}_o^c$ at a certain time t_i, the parabolic motion model of the ball (9) is evaluated

$$\hat{\boldsymbol{p}}_o^c = \boldsymbol{R}_g^{c,i} \left(\hat{\boldsymbol{p}}_0^g + \hat{\dot{\boldsymbol{p}}}_0^g t_i + \frac{1}{2} \boldsymbol{g}^g t_i^2 \right),$$

while the term $\hat{z}^c$ required for the evaluation of $\hat{\boldsymbol{L}}_{sp}$ in (4) is the third component of such vector.

In order to complete the terms required in (4), the two (3×1) vectors $\hat{\dot{\boldsymbol{p}}}_o^c$ and $\hat{\boldsymbol{\omega}}_o^c$ should be computed. The first one can be retrieved by considering the time derivative of (9) and evaluating it at the required time t_i

$$\hat{\dot{\boldsymbol{p}}}_o^c = \boldsymbol{R}_g^{c,i} \left(\hat{\dot{\boldsymbol{p}}}_0^g + \boldsymbol{g} t_i \right).$$

The last term $\hat{\boldsymbol{\omega}}_o^c$ can be obtained as

$$\hat{\boldsymbol{\omega}}_o^c = (1/\|\hat{\boldsymbol{p}}_o^c\|^2)(\hat{\boldsymbol{p}}_o^c \times \hat{\dot{\boldsymbol{p}}}_o^c).$$

Since before of n measurements is not possible to have any estimation of $\boldsymbol{p}_0^g$ and $\dot{\boldsymbol{p}}_0^g$, an initial rough estimation should be provided in order to compute the above quantities. For such a reason, a statistical calibration has been preliminary realized, and the results have been in turn employed in the experiments presented in Section 5.