3D Monocular Robotic Ball Catching

Vincenzo Lippiello, Fabio Ruggiero and Bruno Sicliano

PRISMA Lab, Dipartimento di Ingegneria Elettrica e Tecnologie dell'Informazione, Università degli Studi di Napoli Federico II, via Claudio 21, 80125, Napoli, Italy.

Abstract

A new method to catch a thrown ball with a robot endowed with an eye-in-hand monocular visual system integrated into a gripper is proposed. As soon as the thrown ball is recognized by the visual system, the camera carried by the robot end-effector is forced to follow a baseline in the space so as to acquire an initial data-set of visual measurements from several points of view, providing a first estimate of the catching point through a linear estimation algorithm. Hereafter, additional measurements are acquired to constantly refine the previous estimate by exploiting a nonlinear estimation algorithm. During the robot trajectory, the translational components of the camera are controlled in such a way as to follow the planned path to intercept the ball, while the rotational components are forced to keep the ball into the field of view. Experimental results performed on a common industrial robotic system prove the effectiveness of the presented solution.

Keywords: Ball catching, monocular camera, visual servoing, trajectory estimating

1. Introduction

Advanced robotic systems, which are required to perform quick reactions in response to visually perceived movements in a partially structured environment, are no doubt a good benchmark for testing of new control algorithms and new estimating/predicting processes. The challenging scenario of the robotic ball catching problem has been extensively considered in the literature for experimental testing of the above capabilities.

Email address: {vincenzo.lippiello, fabio.ruggiero, bruno.siciliano}@unina.it (Vincenzo Lippiello, Fabio Ruggiero and Bruno Sicliano)

The state of the art presented in Section 2 shows that most of the existing systems use either a stereo visual configuration to solve the 3D ball catching problem or a single camera for the 2D sub-problem. In the former case, the 3D tracking of the ball takes advantage from having the possibility to exploit the epipolar constraint in the two available images; in the latter case, only 2D information can be directly retrieved from the single available image. Nevertheless, in both situations, in order to obtain a successful catch, high frame rate and accurate cameras are required to have a fast and precise estimate of the ball trajectory. By employing just one camera it is possible to reduce the total cost of the set-up. Moreover, the computational cost to elaborate at high frame rate one image is lower than elaborating at the same frequency two images. Hence, when high frame rate is a strict constraint, using only one camera also saves computational resources. However, some improvements in the controller and in the prediction algorithm should be introduced to solve the problem in 3D.

In this paper, the robotic 3D ball catching problem is solved by using a monocular visual system. A standard industrial robot manipulator is equipped with a CCD camera mounted directly on the manipulator end-effector. The proposed control law is composed of a continuous refinement of the ball interception point through a nonlinear estimation algorithm, whose initial starting condition is provided by a fast linear estimating process. An initial camera motion is thus commanded along a suitable baseline so as to collect a sufficient initial number of visual data from different points of view and provide such initial estimate. Experimental results demonstrate the effectiveness of the presented solution. The present work extends what already presented by the authors in [1]. With respect to the past work, in this paper the analysis of the employed controller is more detailed and the stability proof is provided. Moreover, the estimating process has been improved by introducing a recursive outliers rejection algorithm that improves the measurement dataset employed for the estimation process. Finally, the policy employed for the interception point selection has been detailed considering the physical limits of the robot, while the performance of the proposed trajectory estimator is shown through a comparison with the ground-truth provided by an OptiTrack motion-capture system.

The outline of the paper is as follows: after the presentation of related work, an overview of the proposed algorithm is provided and the image processing is briefly revised. Then, the partitioned visual-servoing controller adopted during the catching motion is presented along with the stability

proof. Further, the linear and nonlinear estimation algorithms are formalized. Finally, the adopted set-up, the achieved experimental results and the comparison with the available ground-truth are shown and critically discussed.

2. Related work

Several research works address problems related to the catch of thrown objects and to the estimate of their trajectories. The approaches described in this section can be categorized as follows: stereo visual systems that deal with ball-catching tasks; monocular visual systems performing catching operations in a plane; monocular visual systems that estimate the object trajectory and perform catching operations in the 3D space; estimators dealing with Chapman hypothesis; systems that take into account the forces exchanged between the gripper and the thrown object; dynamic non-prehensile catching tasks; neural networks; virtual reality applications.

2.1. Stereo visual systems

Vision systems employing two (or more) cameras benefit by using the triangulation method to reconstruct the 3-D position of the ball [2, 3, 4]. However this requires an accurate calibration procedure and a sophisticated elaboration hardware.

A stereo vision system combined with an observer with a variable strength filter and an error estimator are employed in [5] to track and catch a thrown ball. An initial motion algorithm is chosen to maximize the response time so as to begin the motion of the arm as soon as the first visual data is taken.

A stereo visual system, an extended Kalman filter and a prediction algorithm are employed in [6] to build a robotic ball catcher. Without using specialized hardware, only using off-the-shelf components, the authors employ a stereo visual system to track a fast flying object and to catch it with a net mounted upon a robotic arm. In order to detect the ball, the difference between the actual image and some reference images is computed using a threshold method. Lately, a mobile humanoid and a gradient method to detect circles in the images are employed in [7].

An inexpensive and uncalibrated camera is exploited in [8] to track a rolling ball before it falls from a table. A robotic arm is programmed to catch it through an attractor-based dynamics that autonomously generates temporally discrete movements and sequences for the robot end-effector.

Only 2D visual information given by a stereo vision system is employed in [9] to achieve position control of a 3D robotic arm and catch a thrown object. The control is applied to achieve simultaneously 2D tasks defined directly on the image planes of the cameras: the 3D task is considered as accomplished if all the 2D tasks are simultaneously fulfilled. However, such working condition cannot guarantee the catch, since there is no estimate of the 3D ball motion. Moreover, no prior knowledge about configuration and dimension of the robotic arm is needed as well as no information about camera parameters is requested (i.e., the robot Jacobian and camera calibration are estimated on-line).

A planar robotic arm, a stereo visual system and a DSP equipment are utilized in [10] to predict the right falling place of moving balls through a Lagrangian interpolation formula.

2.2. Monocular visual systems: catching in a plane

Monocular visual systems have easier calibration procedures, but more effort has to be put in the 3D reconstruction of the scene.

A camera and a prediction-based control system are employed in [11] to catch a mouse moving on a plane. A single camera is employed to localize the moving part, but since that is free to change the velocity and the acceleration of its motion in the whole plane, then a continuous re-planning of the path of the manipulator is required. The catch is always performed along a predetermined catching line, and for this reason the target object should cross such a line in a finite amount of time.

An experiment consisting in the catch of a ball moving on a table with a robot manipulator is carried out in [12]. Two distinct visual servoing architectures are implemented: position-based and image-based visual servoing. The catch is always performed along a straight line on the table and thus the precise catching point is determined with the intersection between such a line and the predicted trajectory of the ball, which is observed by a camera mounted on the robot end-effector.

2.3. Monocular visual systems: catching in 3D

The 3D position and velocity of a thrown projectile are estimated in [13] through the analysis of a sequence of images taken by a single camera. A least-squares algorithm is employed to determine the state of such projectiles from their apparent trajectories and considering a model of the motion without air drag.

A recursive least-squares algorithm is even used in [14] to estimate the trajectory of a ball with an eye-to-hand visual system. Catching is performed through a combination of image-based and position-based visual servoing.

This case best fits the work here presented. With respect to the cited works, a monocular eye-in-hand configuration is considered in this paper, hence the camera is mounted directly on the end-effector of the robot. In this way, the control is in charge of maintaining the ball inside the camera field of view, which instead is not possible in a eye-to-hand configuration. However, the visual system has to cope with the change of resolution of the ball in the image during the throw and with possible blur motions. Moreover, with respect to the previous works, a more realistic model of the ball motion is here employed, including also the air drag factor, along with a more sophisticated nonlinear estimation process.

2.4. Estimators dealing with Chapman hypothesis

The Chapman strategy to catch a ball is introduced in [15], where it is stated that a fielder should run at a proper speed to maintain a constant increasing rate of the tangent of the ball elevation angle.

Reinforcement learning models are exploited in [16] to better understand the perceptual features that guide a fielder to learn how to catch a flying ball. For this reason, the authors implemented a system which learns both how to keep constant the increasing rate of the tangent of the elevation angle and how to use the velocity of the ball perpendicular to the fielder to decide whether to run forward or backward.

Linear Optimal Trajectory(LOT) is introduced in [17] to catch a ball with an autonomous mobile robot. By using LOT, the fielder problem, already presented in Chapman paper, is converted into the spatial problem of keeping the relation features in the 2D image plane. However, this approach requests information about the ball initial optical location relative to the robot and it is not applicable when the robot is located in the same vertical plane of the ball trajectory. Hence, the same authors proposed in [18] the GAG (*Gaining Angle of Gaze*) strategy, still based on Chapman hypothesis. GAG strategy requires only the information about the elevation angle of gaze captured as a 2D information, and it has also been extended to be used upon a robot manipulator equipped with a camera, mounted in eye-in-hand configuration, which moves in a plane [19].

In [20], the authors state that in order to track the ball in the manner indicated by Chapman, the correct catching position and time should be

known before starting the tracking task so as to determine the tracking speed. Therefore, they presented a class of analytic solutions that track properly the ball, and in which class Chapman results represent one solution among the others.

2.5. Systems taking into account the force exchange

A falling ball and a falling cylinder are caught through a high-speed multi-fingered hand in [21, 22] by using a stereo visual system. The proposed strategy gives to the fingers an impact force changing the movement of the object in the desired direction, in order to catch it in an optimal and stable way.

The impedance of the hand is regulated for virtual ball catching tasks in [23]. The authors underline the importance in regulating such impedance since, for instance, a human might miss the ball when (s)he makes her/his arm stiffen beyond necessity, because of the large contact force exerted on the hand from the ball or, in the same way, (s)he might miss the ball making her/his arm too compliant, thus without generating enough force to absorb the ball motion.

2.6. Dynamic non-prehensile catching

Lately, non-prehensile ball catching [24] is considered as a particular case of dynamic manipulation. The ball tracking is performed using a stereo vision configuration, while the prediction of the ball trajectory is carried out through a least-squares algorithm. The robot is guided to dynamically catch the ball and, after the catch, a balancing controller is activated to keep the ball on a plate mounted on the robot end-effector. The position of the ball on the plate is estimated by measuring the forces and the torques at the end-effector.

2.7. Neural networks and virtual reality applications

A suitable neural network is exploited in [25] to predict the path of the ball and catch it through the humanoid SAIKA, equipped with a stereo visual system.

Virtual reality applications are seen as additional test-bed applications for ball catching tasks [26, 27, 28]. In particular, in the last two cited papers, minimum jerk models for human kinematics are presented as a tool to predict user inputs in teleoperation with significant dynamics. A virtual reality simulation of a teleoperated ball-catching scenario is used in order to test the predictive power of the model.

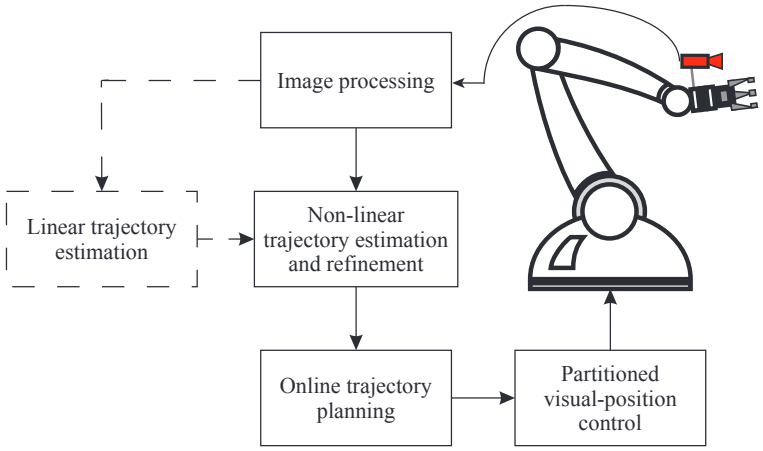


Figure 1: Block diagram of the proposed monocular robotic ball catching system. The dashed lines mean that the related block about Linear Trajectory Estimation is executed only once during the robot starting motion.

3. Algorithm overview

The overview of the proposed 3D monocular ball catching algorithm is shown with the block diagram of Figure 1.

As soon as the thrown ball is detected by the visual system, the camera mounted in an eye-in-hand configuration is forced to follow a suitable baseline in the 3D space. The ball is always kept in the field of view of the camera through a partitioned visual-servoing control. This starting motion is performed to both ensure a well-conditioned estimating problem and collect/process visual information to get a first prediction of the ball trajectory through a rough linear estimate. Such prediction is employed as a starting point for a more precise nonlinear refinement process of the trajectory.

When a new interception pose estimate is available, the on-line motion planner smoothly switches its target to the new one, always keeping the ball in the camera field of view. Hence, the new visual measurements are continuously processed by the nonlinear optimization to on-line update the estimate of the ball trajectory, and thus the prediction of the interception pose.

Finally, when the continuous refinement does no longer improve the prediction of the trajectory significantly, the final catching pose can be computed taking into account the ball and the robot dynamics, so as to accommodate the ball into the robotic gripper.

Figure 2: Picture illustrating how the employed tracker can detect the ball thanks to the desired selected color, defined by the HSL color space. In this case the selected ball is the red one. On the left the original image, on the right the elaborated one after clustering and blob analysis.

4. Image processing

The image processing stage has already been introduced in [1]. Hence, only the key aspects are revised here. Other thrown objects detection methods can be found in [29].

A calibrated [30] CCD camera is mounted on the robot end-effector and the acquired images are continuously processed to obtain the ball centroid in the image and then compute the position of the ball in the normalized image plane. The HSL (*Hue, Lightness, Saturation*) color space is used to cluster the image during its process. The adopted clustering consists in a binarization step and some post-elaboration processes employed to reduce the effects of the image noise. All the blobs in the binarized image are collected and filtered to eliminate, for example, the blobs with a very small area and the background. Finally, the centroid of the selected blob can be considered as an approximation of the ball center. In Figure 2 an example of how the employed tracker works is depicted.

In order to seek the ball for the first time, the whole image is processed with the above described modality until the ball is detected. Afterwards, a dynamic windowing technique is employed to reduce the computational effort of the image processing. The frame rate is speeded-up to more than 100 Hz thanks to the Region of Interest (RoI) acquisition modality available on the current USB cameras.

5. Partitioned visual servoing

The proposed visual control law belongs to the category named *Resolved-Velocity Image-Based Visual Servoing* [31], for which it is assumed that the

manipulator dynamics is taken into account directly by the low-level robot controller. The considered robotic arm structure is an open kinematic chain robot manipulator. The reference frames considered in the ball catching scenario are the robot base frame Σ_b, fixed with respect to the ground, the end-effector frame Σ_e, and the camera frame $\Sigma_c = O_c$–$x_c y_c z_c$ (see Figure 3). Being the last two reference systems fixed with each other, without loss of generality, only the camera frame Σ_c is considered in the remainder of the paper, since it could be assumed that it is coincident with Σ_e. The camera optical axis is then aligned with the approaching axis of Σ_e.

In order to successfully accomplish the ball catching task, the ball should never leave the field of view of the camera. Large parts of the scene can be observed with small movements of the camera orientation, and thus with small movements of the robot joints. For this reason, the rotational components of the robot end-effector –the fastest ones– are devoted to track the ball, while the translational components of the robot end-effector should be planned to intercept the ball. Hence, the partitioned approach presented in [32] has been employed in this work.

The position of the ball center with respect to Σ_c can be denoted as

$$\boldsymbol{p}_o^c = \begin{bmatrix} x^c \\ y^c \\ z^c \end{bmatrix} = z^c \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = z^c \tilde{\boldsymbol{s}},$$

in which $\boldsymbol{s} = \begin{bmatrix} X & Y \end{bmatrix}^{\mathrm{T}}$ is the (2×1) normalized image coordinates vector of the centroid of the ball, provided by image processing as described in Section 4, while $\tilde{\boldsymbol{s}} = \begin{bmatrix} \boldsymbol{s}^{\mathrm{T}} & 1 \end{bmatrix}^{\mathrm{T}}$ is the related (3×1) homogeneous vector of $\boldsymbol{s}$.

The expression relating the (6×1) absolute velocity vector of the camera $\boldsymbol{v}_c^c = \begin{bmatrix} \dot{\boldsymbol{p}}_c^{c\mathrm{T}} & \boldsymbol{\omega}_c^{c\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$, the (6×1) absolute velocity vector of the thrown ball $\boldsymbol{v}_o^c = \begin{bmatrix} \dot{\boldsymbol{p}}_o^{c\mathrm{T}} & \boldsymbol{\omega}_o^{c\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$, both expressed with respect to Σ_c, and the (2×1) velocity vector of the image feature $\dot{\boldsymbol{s}}$ in the image plane, is the following linear equation [31]

$$\dot{\boldsymbol{s}} = \boldsymbol{L}_s \boldsymbol{v}_c^c + \boldsymbol{L}_s \boldsymbol{\Gamma}(-\boldsymbol{p}_o^c) \boldsymbol{v}_o^c, \tag{1}$$

in which $\boldsymbol{L}_s$ is (2×6) is the interaction matrix of a point image feature defined as follows [31]

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

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

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

where $\boldsymbol{I}_n$ denotes the $(n \times n)$ identity matrix and $\boldsymbol{S}(\cdot)$ the skew-symmetric matrix. Let $\boldsymbol{L}_{sp}$ and $\boldsymbol{L}_{so}$ be the (2×3) sub-matrices corresponding to the first and last three columns of (2). Equation (1) can be then rewritten as

$$\dot{\boldsymbol{s}} = \boldsymbol{L}_{sp} \left(\dot{\boldsymbol{p}}_c^c - \dot{\boldsymbol{p}}_o^c + \boldsymbol{S}(-\boldsymbol{p}_o^c)\boldsymbol{\omega}_o^c \right) + \boldsymbol{L}_{so} \left(\boldsymbol{\omega}_c^c - \boldsymbol{\omega}_o^c \right).$$

The translational components $\dot{\boldsymbol{p}}_c^c$ of the robot motion are devoted to move the robot to intercept the ball with the gripper mounted on the robot end-effector. In order to ensure a smooth re-planned trajectory when a new interception target point is available, the continuity between the current motion state, in terms of position, velocity and acceleration, and the new initial one has to be imposed. In order to on-line compute the desired trajectory for the camera frame, a fifth-order polynomial vector in the 3D Cartesian space is taken into account

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

with $\boldsymbol{p}_{c,d}$ the (3×1) desired absolute position of the camera with respect to Σ_b, and $\boldsymbol{a}_h$ with $h = 0, \ldots, 5$ are (3×1) coefficient vectors. The desired velocity of the camera frame is then equal to

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

and the translational acceleration of the camera is equal to

$$\ddot{\boldsymbol{p}}_{c,d} = 20\boldsymbol{a}_5 t^3 + 12\boldsymbol{a}_4 t^2 + 6\boldsymbol{a}_3 t + 2\boldsymbol{a}_2. \tag{5}$$

The parameters $\boldsymbol{a}_h$ in (3)–(5) are updated on-line as follows. Let t_i, t_f, $\boldsymbol{p}_{c,d,i}$, $\boldsymbol{p}_{c,d,f}$, $\dot{\boldsymbol{p}}_{c,d,i}$, $\dot{\boldsymbol{p}}_{c,d,f}$, $\ddot{\boldsymbol{p}}_{c,d,i}$ and $\ddot{\boldsymbol{p}}_{c,d,f}$ be the initial and final planned time, the initial and final position, and the translational velocity and acceleration, respectively. By denoting 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 taking into account (3)–(5), the following linear quadratic system of 18 equations into 18 unknowns is retrieved

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

where $\boldsymbol{i}_3 = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$, $\boldsymbol{t}_i = t_i \boldsymbol{i}_3$, $\boldsymbol{t}_f = t_f \boldsymbol{i}_3$, and $\boldsymbol{0}_3 = 0\boldsymbol{i}_3$. It is worth noticing that the initial conditions are given by the current state of the motion of the robot, while the final conditions depend on the current available estimate (more details will be provided in the next sections).

Hence, on one hand, the translational components of the velocity input for the camera frame Σ_c can be generated as follows

$$\dot{\boldsymbol{p}}_c^c = \boldsymbol{R}_c^{\mathrm{T}} \left(\dot{\boldsymbol{p}}_{c,d} + \boldsymbol{K}_p \boldsymbol{e}_p \right), \tag{7}$$

with $\boldsymbol{R}_c$ the rotational matrix of the camera frame Σ_c with respect to the base frame Σ_b, $\boldsymbol{K}_p > 0$ a diagonal constant (3×3) gain matrix, and $\boldsymbol{e}_p$ the (3×1) error vector between the desired trajectory (3) and the one provided by the robot direct kinematics at time t.
On the other hand, the rotational components of the velocity input for the camera frame Σ_c can be generated in the image space as follows:

$$\boldsymbol{\omega}_c^c = \boldsymbol{L}_{so}^{\dagger} \left[\boldsymbol{K}_{so,e_{b2}}(\boldsymbol{e}_s) \boldsymbol{\tau}_{e_{b1}}(\boldsymbol{e}_s) - \hat{\boldsymbol{L}}_{sp} \left(\dot{\boldsymbol{p}}_c^c - \dot{\hat{\boldsymbol{p}}}_o^c + \boldsymbol{S}(-\hat{\boldsymbol{p}}_o^c) \hat{\boldsymbol{\omega}}_o^c \right) \right] + \hat{\boldsymbol{\omega}}_o^c, \tag{8}$$

with $\dagger$ denoting the pseudo-inverse of a matrix, $\hat{\boldsymbol{p}}_o^c$ the estimate of the unknown position of the ball in Σ_c, and $\dot{\boldsymbol{p}}_c^c$ evaluated in (7). The terms $\dot{\hat{\boldsymbol{p}}}_o^c$ and $\hat{\boldsymbol{\omega}}_o^c$ are the estimates of the unknown absolute translational and angular velocities of the ball with respect to Σ_c. Besides, the error term $\boldsymbol{e}_s = -\boldsymbol{s}$ is the image error vector becoming null when the camera is pointed towards the centroid of the ball, while the term $\boldsymbol{\tau}_{e_b}(\boldsymbol{e}_s)$ is a threshold function defined as

$$\boldsymbol{\tau}_{e_{b1}}(\boldsymbol{e}_s) = \begin{cases} \boldsymbol{0} & \text{if } \|\boldsymbol{e}_s\| \leq e_{b1} \\ \left(1 - \dfrac{e_{b1}}{\|\boldsymbol{e}_s\|} \right) \boldsymbol{e}_s & \text{if } \|\boldsymbol{e}_s\| > e_{b1}, \end{cases} \tag{9}$$

with $\boldsymbol{K}_{so,e_{b2}}(\boldsymbol{e}_s)$ a (2×2) gain matrix defined as

$$\boldsymbol{K}_{so,e_{b2}}(\boldsymbol{e}_s) = \begin{cases} k_o \boldsymbol{I}_2 & \text{if } \|\boldsymbol{e}_s\| \leq e_{b2} \\ \beta_o \left(\dfrac{\|\boldsymbol{e}_s\|}{e_{b2}} - 1 \right) \boldsymbol{I}_2 & \\ k_o e & \text{if } \|\boldsymbol{e}_s\| > e_{b2}, \end{cases} \tag{10}$$

where $k_o > 0$ is a gain factor, $e_{b2} > e_{b1} > 0$ are proper thresholds, and $\beta_o > 0$ is a restraint factor tuning the increasing rate of $\boldsymbol{K}_{so}$. In order to avoid the loss of the ball view, this last gain term suddenly increases when the centroid of the ball approaches the limits of the image plane.

Notice that $\hat{\boldsymbol{L}}_{sp}$ in (8) is an estimated term since it depends on $\hat{\boldsymbol{p}}_o^c$. The details about how to estimate $\hat{\boldsymbol{p}}_o^c$, $\dot{\hat{\boldsymbol{p}}}_o^c$ and $\hat{\boldsymbol{\omega}}_o^c$ are given in Appendix.

For what concerns the stability proof of the system, the following theorems hold.

Theorem 1. *Provided that $\boldsymbol{K}_p$ is a positive definite matrix, the control law* (7) *ensures an asymptotic convergence to zero of the position error $\boldsymbol{e}_p$.*

Proof. The time derivative of the position error can be computed as

$$\dot{\boldsymbol{e}}_p = \frac{\mathrm{d}}{\mathrm{dt}}(\boldsymbol{p}_{c,d} - \boldsymbol{p}_c(t)) = \dot{\boldsymbol{p}}_{c,d} - \dot{\boldsymbol{p}}_c,$$

where $\boldsymbol{p}_c(t)$ is the (3×1) vector denoting the current position of the camera with respect to Σ_b at time t, computed by solving the direct kinematics of the robot, while $\dot{\boldsymbol{p}}_c$ is the related translational velocity of the camera with respect to Σ_b. Pre-multiplying by $\boldsymbol{R}_c$ both sides of (7), folding the result into the previous equation yields

$$\dot{\boldsymbol{e}}_p + \boldsymbol{K}_p \boldsymbol{e}_p = \boldsymbol{0}.$$

Since $\boldsymbol{K}_P$ is a positive definite matrix, usually a diagonal matrix, the previous system is asymptotically stable and the error $\boldsymbol{e}_p$ tends to zero along the trajectory with a convergence rate depending on the eigenvalues of $\boldsymbol{K}_p$. ■

Theorem 2. *The system* (1) *is asymptotically stable under the control laws* (7)–(8)*, in presence of a perfect estimate of the unknown terms. Otherwise, only stability can be ensured.*

Proof. The following analysis is performed using the direct Lyapunov theorem. Consider the following candidate Lyapunov function

$$V(\boldsymbol{e}_s) = \boldsymbol{e}_s^{\mathrm{T}} \boldsymbol{K}_s \boldsymbol{e}_s, \tag{11}$$

where $\boldsymbol{K}_s$ is a (2×2) positive definite diagonal matrix. By noticing that $\dot{\boldsymbol{e}}_s = -\dot{\boldsymbol{s}}$, computing the time derivative of (11) and taking into account (1) yields

$$\dot{V} = -\alpha_1 - \alpha_2 - \alpha_3,$$

where

$$\alpha_1 = \boldsymbol{e}_s^{\mathrm{T}} \boldsymbol{K}_s \left(\boldsymbol{L}_{sp} - \hat{\boldsymbol{L}}_{sp} \right) \dot{\boldsymbol{p}}_c^c \tag{12a}$$

$$\alpha_2 = \boldsymbol{e}_s^{\mathrm{T}} \boldsymbol{K}_s \boldsymbol{K}_{so,e_{b2}}(\boldsymbol{e}_s) \boldsymbol{\tau}_{eb_1}(\boldsymbol{e}_s) \tag{12b}$$

$$\alpha_3 = \boldsymbol{e}_s^{\mathrm{T}} \boldsymbol{K}_s \left(\boldsymbol{L}_s \boldsymbol{\Gamma}(-\boldsymbol{p}_o^c) \boldsymbol{\upsilon}_o^c - \hat{\boldsymbol{L}}_s \boldsymbol{\Gamma}(-\hat{\boldsymbol{p}}_o^c) \hat{\boldsymbol{\upsilon}}_o^c \right). \tag{12c}$$

If each term in (12) is strictly positive, then $\dot{V} < 0$. However, no term in (12) is a quadratic form, hence only qualitative considerations can be achieved.

If $\hat{\boldsymbol{L}}_{sp} = \boldsymbol{L}_{sp}$, the term α_1 in (12a) vanishes, but there is no guarantee that such condition can happen. Nevertheless, the α_1 term is bounded since the condition for updating $\hat{\boldsymbol{L}}_{sp}$ through the estimate of $\hat{\boldsymbol{p}}_o^c$ seems to be the optimal one during the experiments [33].

By considering (9)-(10), the term α_2 in (12b) can be bounded as follows

$$0 \leq \alpha_2 \leq \boldsymbol{e}_s^{\mathrm{T}} \boldsymbol{K}_s \begin{pmatrix} k_o e^{\beta_o \left(\frac{\|\boldsymbol{e}_s\|}{e_{b2}} - 1 \right)} \boldsymbol{I}_2 \end{pmatrix} \boldsymbol{e}_s. \tag{13}$$

By choosing $\boldsymbol{K}_s = k_o e^{\beta_o \left(\frac{\|\boldsymbol{e}_s\|}{e_{b2}} - 1 \right)} \boldsymbol{I}_2$, the last term in (13) is positive definite. Hence, α_2 is always positive and limited.

The α_3 term in (12c) vanishes in case of perfect estimate. Otherwise, nothing can be said about the sign of α_3. However, (12b) is a quadratic form in $\boldsymbol{e}_s$, while (12c), and also (12a), are linear functions of the error in the image plane. Hence, on one hand, for an error of small norm, the linear terms prevail over the quadratic terms, but the norm of $\boldsymbol{K}_s$ can be increased, i.e., an higher value of k_o, to reduce the error as much as possible. On the other hand, for larger errors, the quadratic terms prevail over the linear ones. In conclusion, the error $\boldsymbol{e}_s$ in (12c) is bounded.

Finally, in case of a perfect estimate, the terms α_1 and α_3 vanish, while α_2 is positive and limited. Then, the chosen control laws lead to an asymptotically stable system. In case of an imperfect compensation, instead, the error in the image plane is anyway bounded. For a ball catching task, this stability condition can be considered sufficient, because the visual control goal is mainly in keeping the ball in the field of view of the camera. ■

Finally, the input to the robot controller, i.e., the joint velocity vector, can be computed as [31]

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{\dagger}(\boldsymbol{q})\boldsymbol{T}_c\boldsymbol{v}_c^c + \boldsymbol{N}_J\boldsymbol{K}_r\dot{\boldsymbol{q}}_r, \tag{14}$$

with $\boldsymbol{q}$ the vector of joint positions, $\boldsymbol{J}(\boldsymbol{q})$ the Jacobian matrix of the robot, $\boldsymbol{T}_c$ the (6×6) matrix relating $\boldsymbol{v}_c^c$ to the velocity of the robot end-effector with respect to the base frame, $\boldsymbol{N}_J$ the projector matrix into the null space of the robot Jacobian, $\boldsymbol{K}_r$ a gain diagonal matrix, and $\dot{\boldsymbol{q}}_r$ a set of joint velocities employed in a possible redundancy management to optimize some other subtasks. For instance, $\dot{\boldsymbol{q}}_r$ might be employed to avoid joint limits and kinematic singularities.

6. Trajectory estimation

6.1. Initial baseline

Since just one camera is employed, a classic static triangulation method cannot be adopted in the proposed scenario. The proposed process to estimate the trajectory of the thrown ball consists in the interpolation of 2D visual measurements along the time.

In order to yield a well-conditioned problem, the visual data collection has to be acquired moving the camera along a significant path. Therefore, as soon as the ball is recognized for the first time after the throw by the pitcher, the camera is forced to move along a straight line in the 3D Cartesian space with high velocity, i.e., the initial baseline. The orientation of the camera is controlled with the above introduced control law (8) to keep the ball in the field of view.

6.2. Linear initialization

The procedure for the linear initialization is explained in [34], hence only a brief description is here addressed.

During the initial baseline, a sequence of image measurements are collected. Let t_k be the k-th visual sample time, $\tilde{\boldsymbol{s}}_k$ the corresponding acquired image feature vector, $\boldsymbol{p} = \begin{bmatrix} x & y & z \end{bmatrix}^{\mathrm{T}}$ the points belonging to the camera *optical ray* and passing through the current origin of the camera $\boldsymbol{c}_k = \begin{bmatrix} c_{x,k} & c_{y,k} & c_{z,k} \end{bmatrix}^{\mathrm{T}}$. The k-th feature vector $\boldsymbol{r}_k = \begin{bmatrix} r_{x,k} & r_{y,k} & r_{z,k} \end{bmatrix}^{\mathrm{T}} =$

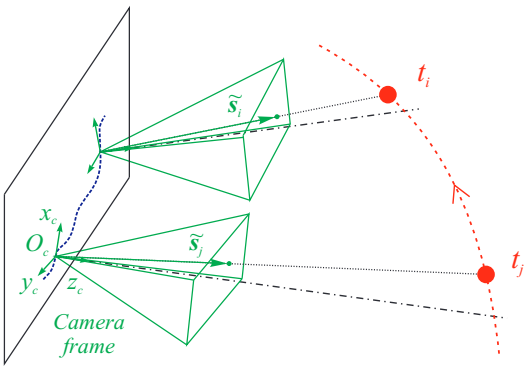


Figure 3: The camera reference frame Σ_c is shown in two different sample times, t_j and t_i. The ball trajectory is shown with a red dotted line, while in blue is represented the corresponding camera trajectory.

$\boldsymbol{c}_k + \boldsymbol{R}_{c,k}\tilde{\boldsymbol{s}}$ can be defined by the following equations representing a straight line in the 3D space (see Figure 3)

$$\begin{cases} (r_{y,k} - c_{y,k})x + (c_{x,k} - r_{x,k})y + r_{x,k}c_{y,k} - r_{y,k}c_{x,k} = 0 \\ (r_{z,k} - c_{z,k})x + (c_{x,k} - r_{x,k})z + r_{x,k}c_{z,k} - r_{z,k}c_{x,k} = 0, \end{cases} \tag{15}$$

where $\boldsymbol{R}_{c,k}$ is the rotation matrix of Σ_c with respect to Σ_b at time t_k. Both $\boldsymbol{c}_k$ and $\boldsymbol{R}_{c,k}$ are provided by the robot direct kinematics.

During this initialization, the effect of the air drag to the motion of the ball is neglected. Hence, the ballistic motion can be modeled as a parabolic function of time t as follows

$$\boldsymbol{p} = \boldsymbol{p}_0 + \boldsymbol{v}_0 t + \frac{1}{2}\boldsymbol{g}t^2, \tag{16}$$

with $\boldsymbol{g}$ the gravity acceleration, $\boldsymbol{p}_0$ and $\boldsymbol{v}_0$ the initial position and velocity of the ball ($t = 0$), respectively, corresponding to the time of the first ball detection. Notice that, without loss of generality, the gravity acceleration is aligned to the axis y of the chosen Σ_b, i.e., $\boldsymbol{g} = \begin{bmatrix} 0 & g & 0 \end{bmatrix}^{\mathrm{T}}$ and $g = 9.81\mathrm{m/s}^2$.

At each t_k, the optical rays intersect the ball trajectory. Folding (16) into (15) yields a system of 2 equations into 6 unknowns $\boldsymbol{p}_0$ and $\boldsymbol{v}_0$, that fully describes the trajectory of the ball. Stacking into rows the n_l measurements yields a system of n_l equations into 6 unknowns that can be solved through a least-squares solution. Additional considerations about this stage are given in [1, 34].

When the process produces the estimate, the first estimated interception point is then computed. In details, by starting from the actual state of the robot (time, position, velocity and acceleration of the camera frame), the system in (6) can be solved to obtain the new set of $\boldsymbol{a}_h$ parameters. These allow the robot to reach the predicted interception position, whose computational details are provided in Section 6.4, at the estimated catching time. The rotational part of the camera, instead, is kept free to track the ball in order to acquire more visual measurements during the movement.

The estimate provided by this linear algorithm is employed as a starting point for the next stage, that is a nonlinear algorithm for the estimate of the ballistic motion. Such an estimate is continuously refined by using new available observations of the ball and a more accurate trajectory model.

6.3. Nonlinear estimation refinement

New visual measurements are collected during the time required by the previous linear estimating process to give the result. Afterwards, both the new visual measurements and the old ones are employed in a nonlinear estimating process that starts from the result obtained by the previous linear method.

In details, let $\boldsymbol{s}_k$ be the centroid of the ball acquired at a time t_k, the cost function to be minimized is

$$\min_{\boldsymbol{p}_0, \boldsymbol{v}_0} \sum_{k=1}^{n} \left\| \frac{1}{\tilde{z}_k^c} \begin{bmatrix} \tilde{x}_k^c \\ \tilde{y}_k^c \end{bmatrix} - \boldsymbol{s}_k \right\|, \tag{17}$$

with n the current number of available visual measurements and $\tilde{\boldsymbol{p}}_k^c$ the estimated position of the ball with respect to Σ_c

$$\tilde{\boldsymbol{p}}_k^c = \begin{bmatrix} \tilde{x}_k^c \\ \tilde{y}_k^c \\ \tilde{z}_k^c \end{bmatrix} = \boldsymbol{R}_{c,k}^{\mathrm{T}} \left(\tilde{\boldsymbol{p}}_k - \boldsymbol{c}_k \right). \tag{18}$$

The estimated position of the ball $\tilde{\boldsymbol{p}}_k(\boldsymbol{p}_0, \boldsymbol{v}_0, t_k)$ is obtained numerically by integrating the following ballistic model [6]

$$\dot{\boldsymbol{v}}(t) = \boldsymbol{g} - \frac{c_w \pi d_b^2 \rho_a}{2 m_b} \parallel \boldsymbol{v}(t) \parallel \boldsymbol{v}(t), \tag{19}$$

with c_w a coefficient depending on the thrown object, d_b the diameter of the ball, ρ_a the density of the air and m_b the mass of the ball. Hence, the previous

model includes the air drag, and its numeric integration is performed in the time interval $[0, t_k]$, with the initial conditions $\boldsymbol{p}_0$ and $\boldsymbol{v}_0$.

Minimizing (17) means that the initial conditions of the ballistic model are tuned to generate an estimated trajectory of the ball that minimizes the distance between the predicted projection of the ball onto the image plane and the corresponding measured observations of the ball along the time.

In practice, the minimization of the cost function (17) is performed using the *Levembert-Marquardt* algorithm. The result is the updated values of $\boldsymbol{p}_0$ and $\boldsymbol{v}_0$.

With respect to [1], a statistical procedure to deal with the presence of image noise is proposed. During the trip, the ball can be subject to different illumination/shadow conditions that could generate different levels of noise for the samples into the measurement dataset. In details, once the minimization process ends, the mean error, the standard deviation and the contribution of each visual measurement $\boldsymbol{s}_k$ to the error residual are evaluated. The visual measurements that contribute to the error residual outperforming with a certain factor the standard deviation are temporarily excluded for the next estimation process.

New measures are acquired during the time in which such nonlinear estimating process computes the updated values of $\boldsymbol{p}_0$ and $\boldsymbol{v}_0$, and then the updated interception point. This new data set (without the previously considered outliers) is employed during the next nonlinear refinement that will adopt the previous optimal solution as initial condition. Again, once the minimization process ends, all the visual measurements (even the ones previously considered as outliers) that contribute to the error residual in a way that is not statistically coherent are excluded for the next optimization. This arrangement is able to improve significantly the accuracy of the estimation process when noisily visual measurements are available.

The $\boldsymbol{a}_h$ trajectory parameters are updated by solving (6) as soon as the new interception point is available. The new trajectories (3) and (4) start from the current motion state of the camera and end with the estimated position at the update final time of interception.

Such refinement process stops when the current estimate catching time is approaching with respect to the grasping time required by the available gripper. Afterwards, the final catching trajectory is planned to accommodate the motion of the ball into the available robotic hand, as explained in the following subsection.

6.4. Catching point selection

The current catching point $\boldsymbol{p}_\times$ is evaluated along the current estimated trajectory as the point that is reachable with a minimum effort of the actual joint torques. The time of the catch $t_\times$ and the velocity of the ball $\boldsymbol{v}_\times$ at the position $\boldsymbol{p}_\times$ can be as well evaluated from both the current predicted trajectory and the kinematic/dynamic robot models.

In detail, the current estimated path $\mathcal{T}$ falling into the working space of the robot manipulator is sampled with a fixed step in several candidate catching points $P_i \in \mathcal{T}$. For each of these points, the robot inverse kinematics is computed so as to retrieve the joint position of the manipulator. The inverse kinematics is computed with a closed-loop inverse kinematics (CLIK) algorithm [31].[1] A set of quality indices can be considered for the selection of the best catching point. In this paper, a joint limits and manipulability measurements are suitably combined with a convex linear combination (see [31] for more details about the adopted quality indices). The candidate catching point maximizing the above defined convex linear combination of quality indices is chosen as the current catching point $\boldsymbol{p}_\times$.

Therefore, the parameters $\boldsymbol{a}_h$ are tuned to lead the gripper from the current state of the robot to the point $\boldsymbol{p}_\times$ at the time $t_\times$ with the same velocity of the ball $\boldsymbol{v}_\times$. Notice that the planned trajectory could be not feasible with respect to the robot capabilities. If the maximum required acceleration for the end-effector, i.e. the camera, is greater than a fixed limit chosen in a conservative way accordingly with robot capabilities, then the catching time is properly scaled. Denoting with α_{max} the norm of the maximum acceleration that the end-effector can reach and with $\ddot{\overline{\boldsymbol{p}}}_{c,d}$ the maximum planned acceleration that can be retrieved from (5), if $\|\ddot{\overline{\boldsymbol{p}}}_{c,d}\| > \alpha_{max}$, then the catching time is scaled as $\bar{t}_\times = t_\times \sqrt{\|\ddot{\overline{\boldsymbol{p}}}_{c,d}\|/\alpha_{max}}$.

The *catching path* is then generated when the estimation process stops. In details, the orientation of the camera is controlled to have a direction of the optical axis, i.e., of the gripper, equal to the tangent to the estimated trajectory of the ball at $\boldsymbol{p}_\times$. Once that the catching point is reached at $t_\times$

[1] Notice that the inverse kinematics can be evaluated iteratively between two consecutive candidate catching points, by using the joint configuration of the previous catching point to initialize the algorithm for the new Cartesian configuration. In this way, being consecutive candidates close to each other, few iterations are required to converge to the solution of the joint configuration corresponding to the current candidate.

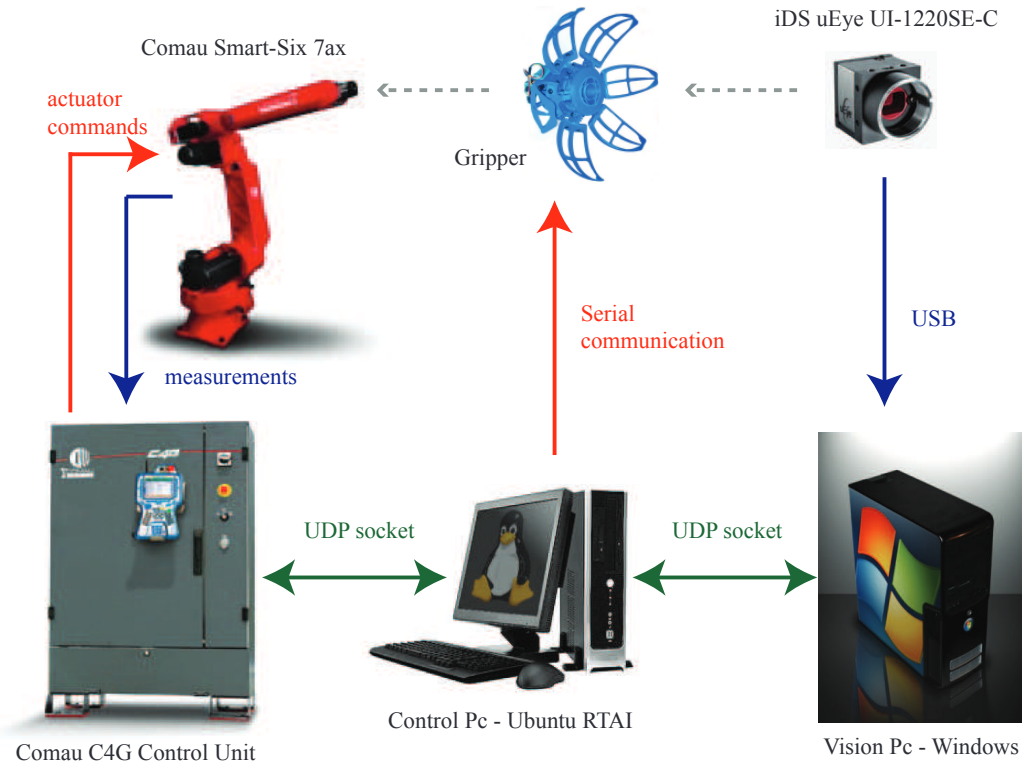


Figure 4: Architecture of the ball catching system.

with the same (or reduced) velocity of the ball, the gripper is closed and is moved along the predicted path of the ball, while its velocity is decreased to zero in a fixed time/space. In such a way it is possible to dissipate the impact energy in a proper time interval.

7. Experiments

7.1. Experimental set-up

The employed experimental set-up is depicted in Figure 4. A USB iDS UEYE UI-1220SE-C camera is mounted in an eye-in-hand configuration directly in the center of the palm of the gripper. Such a gripper is made up of two 12V brushed DC motors, with a metal gearbox and an integrated quadrature encoder. Through a rack-and-pinion mechanism the motion of these motors allows the closure of the gripper fingers. The hand is mounted on a COMAU Smart-Six robot manipulator standing on a sliding track. The

compensation of the dynamic model of the robot is in charge of the COMAU C4G control unit.

An external PC with a RTAI real-time kernel UBUNTU OS generates the references for the robot each 2ms. This PC communicates with a second PC with Windows OS that is in charge of the visual elaboration process. In order to improve the stability of the elaboration time and synchronize the visual measurements with the motion of the robot, a high-priority multi-thread programming has been employed.

A ground-truth for the estimation algorithm is achieved from an OptiTrack motion-capture system composed of ten S250e cameras that are employed to track the ball during its motion.

7.2. Technical details

An image size of (375×500) pixels together with a dynamic RoI window of (150×150) pixels are employed to increase the acquisition frame rate up to 140 fps.

The thrown ball has a diameter of 8.5 cm and a weight of about 32 g. Six reflecting markers are attached on the ball in order to make the OptiTrack able to observe it. These markers do not affect the detection of the ball in the scene since they are very small. In fact, the corresponding image blobs are small with respect to the ball area and are easily eliminated during image processing. The coefficients of the air drag factor have been tuned to $c_w = 0.45$ and $\rho_a = 1.293$ kg/m^3.

The gain matrix in (7) has been set to $\boldsymbol{K}_p = 500\boldsymbol{I}_3$, while the gains in (8) have been tuned to $k_o = 200$, $e_{b1} = 10$ and $e_{b2} = 100$.

The coefficients of the air drag factor have been firstly retrieved from classic fluid dynamics theory and then have been refined during the experiments. The control gains have been tuned in an experimental way as well.

The maximum joint velocities of the employed robot from axis 1 to 6 are namely: 140, 160, 170, 450, 375 and 550 degrees per second. The sliding track maximum velocity is 1.5 m/s. Through experimental validation it has been verified that, in a conservative way, the feasible maximum Cartesian acceleration norm for the end-effector is about $\alpha_{max} = 40$ m/s^2, for trajectories lying inside the catching volume. The intrinsic redundancy of the chosen robotic platform has been exploited in (14) to avoid joint limits, kinematic singularities and to reduce the movements of the sliding track since its dynamics is considerably slower than those of the other joints.

Considering some environmental constraints which are present in the lab and the dexterous working space of the robot, the catching volume is $1.5 \times 1.2 \times 0.5$ m. (w $\times$ h $\times$ d), seen with respect to the pitcher. The initial baseline has a planned length of 50 cm that should be performed by the camera in 500 ms. However, the first estimate of the trajectory starts when about $n_l = 45$ samples have been collected, i.e., after about 320 ms, hence typically the first catching trajectory starts before the end of the baseline path. Latency periods and delays between the robot control PC, the C4G control unit and the visual elaboration PC are present and they are estimated so as to synchronize at the best the direct kinematic measurements with the visual data.

7.3. Results

Several experiments have been performed with several pitchers and varying light conditions. Over a set of about 300 throws, the catching rate is about 87.5%, with an interception rate of 98%[2]. In Figure 5 it is possible to observe the complete ball trajectory for a given throw, with the overlay of the robot motion.

The OptiTrack system has been employed to give a ground-truth about the estimate of the ballistic trajectory. Some examples are depicted in Figure 6, where the green tube is the space occupied by the ball during its flight towards the robot and it is measured by the OptiTrack system. In details, the motion capture system provides the 3D position of the six markers at a frequency of 250 Hz. By knowing both the geometrical features of the ball and the position of these markers it is possible to reconstruct the real ballistic trajectory of the ball. The blue line is the final estimated trajectory of the centroid of the ball. It is possible to observe that the blue line is always inside the green tube: this gives a geometric quality measure about the performance of the proposed estimating process. The red line shows the path followed by the camera/gripper mounted on the robot. It is then possible to recognize both the initial baseline and the catching path, in which the gripper exactly follows the predicted trajectory to decrease the velocity of the ball.

Moreover, further to the geometrical path, another quality index to measure the performance of the estimating process is a comparison along the time

[2]Notice that some thrown ball are suitably intercepted but not firmly caught.

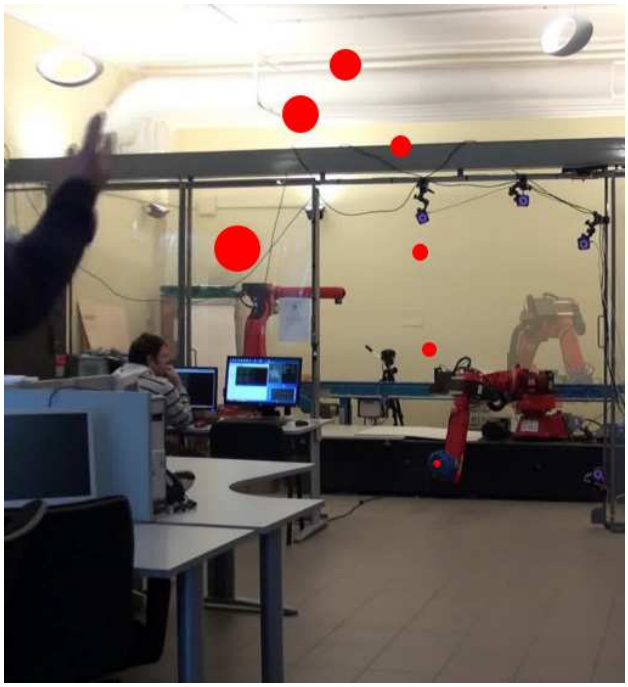
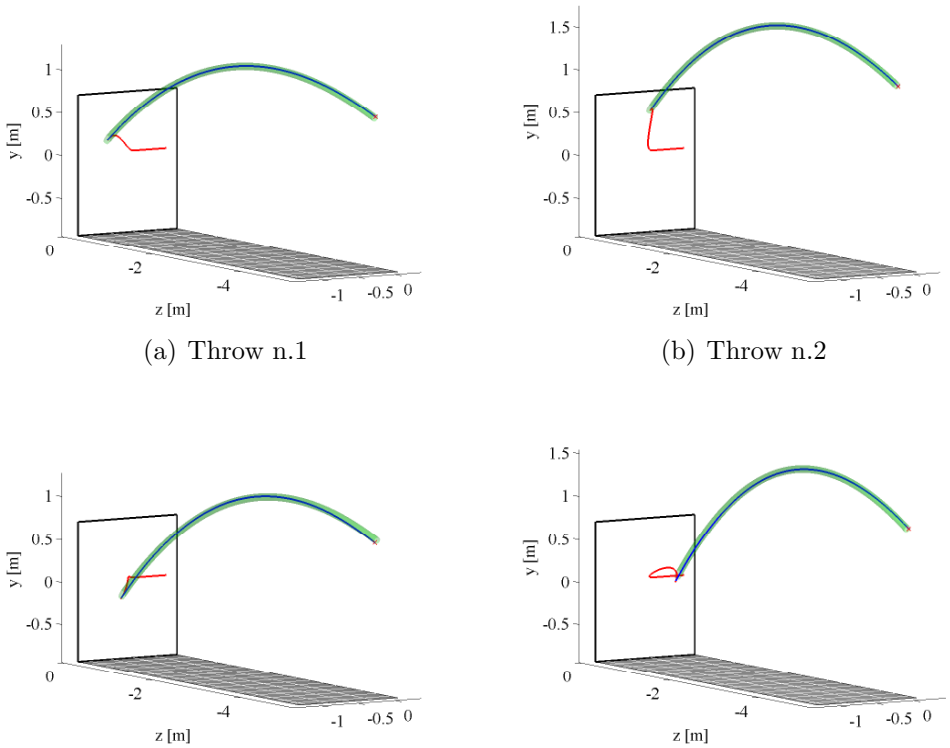


Figure 5: Overlay of the ball trajectory and robot motion.

between the ground-truth and the final predicted trajectory. With reference to the throw depicted in Figure 6(a), the time histories of both the ground-truth and the predicted trajectories are shown in Figure 7. Again, the time histories of the predicted trajectory fits inside the ground-truth provided by the OptiTrack system.

Again with reference to the throw depicted in Figure 6(a), all the predicted interception points $\boldsymbol{p}_{\times}$, projected in both the $(x-y)$ and $(z-y)$ planes of Σ_b, are represented with a cross point in Figure 8. The color bar identifies the ordered sequence of such predicted interception points, while biggest brow cross represent the final position $\boldsymbol{p}_{\times}$ in which the estimate has been considered as stable. The dashed lines represent the planned path for the hand, which is achieved using (3) starting from the current motion state and leading to the current estimated interception position, while the continuous line is the real path followed by the gripper, which starts with the baseline (green piece of the path). It is worth noticing that the first estimated point, the green one, is given by the linear estimating process. The big orange circle is the full representation of the ball in the final position measured by

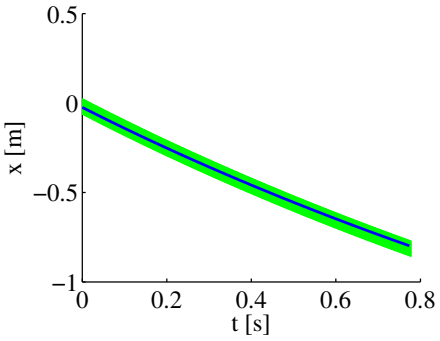


(a) Throw n.1 (b) Throw n.2

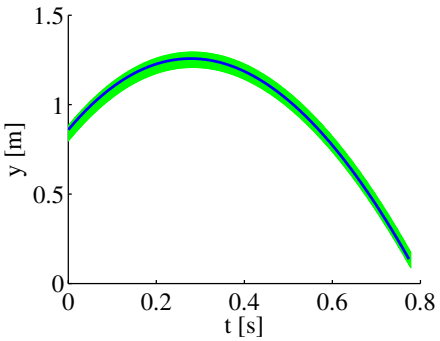
(c) Throw n.3 (d) Throw n.4

Figure 6: 3D plots of four different throws. The blue line is the final estimated trajectory of the ball centroid. The green cylinder is the space occupied by the ball during the flight that has been measured by the OptiTrack system. The red path is the motion of the camera/gripper.

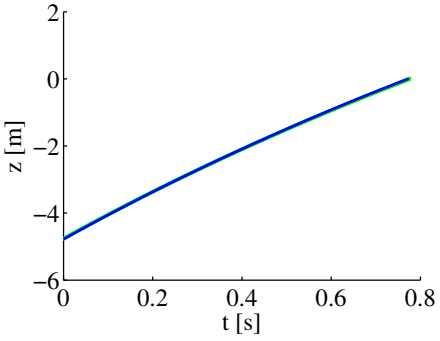
the OptiTrack system and projected in the above mentioned planes of Σ_b. The big blue circle in the background is instead the space occupied by the gripper base in the final estimated position and projected in the the above mentioned planes of Σ_b. The information provided by Figure 8 is twofold. First, it is possible to recognize the tolerance between the real position of the ball and the space occupied by the gripper at the interception point; then, another way to measure the quality of the estimate is the evaluation of how far the final estimated $\boldsymbol{p}_\times$ is from the center of the measured position of the ball.



(a)

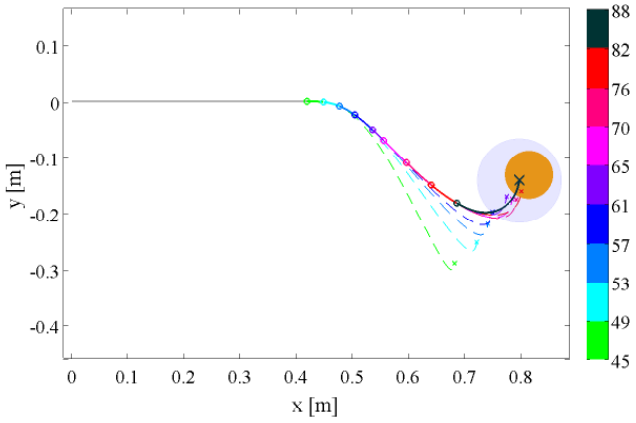


(b)

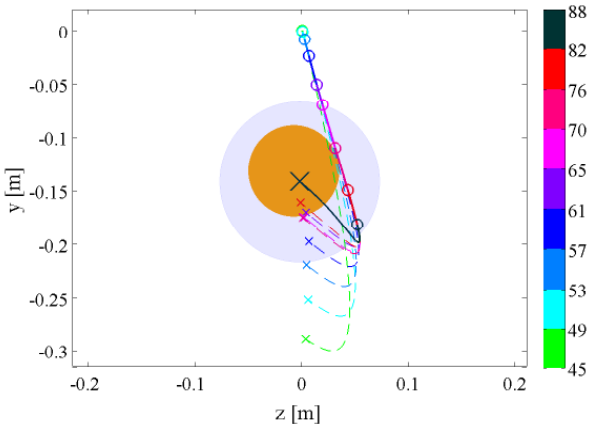


(c)

Figure 7: Time histories of the ground-truth (in green) and the final predicted trajectory (in blue) of the throw depicted in Figure 6(a).



(a) $x - y$ plane.



(b) $z - y$ plane.

Figure 8: Sequence of the interception points (cross points) projected into the $(x - y)$-plane and $(z - y)$-plane. The dashed lines represent the planned path, starting from the current hand position (circle points) and ending into the current estimated interception points. The continuous lines represent the real path followed by the gripper, starting with the initial baseline (green) and leading to the final catching point (biggest cross). The big orange circle represents the final position of the ball measured by the OptiTrack system. The big blue circle in the background is the estimated final position of the gripper catching base. The color bar on the right identifies the refinements of the interception points, while the related labels represent the number of visual measurements employed by the estimating process of each refinement.

8. Conclusion and future work

A new solution to cope with the problem of catching a thrown ball with only a single camera mounted on the robot end-effector has been described. To the best of the authors' knowledge, with respect to the current state of the art, this is the first successful attempt to catch a flying object with a singular moving camera. The estimate of the ball trajectory uses an iterative nonlinear optimization algorithm, which employs only 2D visual measurements together with a complete ballistic ball motion model. Moreover, a linear estimation algorithm based on a first initial collection of ball observations and on a simplified ball motion equation is employed to initialize the nonlinear optimization algorithm, resulting in a significant speed-up of the proposed method. In order to prove the results given by the proposed estimator, a comparison with the measurements given by an OptiTrack motion-capture system has been provided. The effectiveness of the proposed approach has been demonstrated both in theory and with experimental results on a common industrial robotic set-up.

Future work will be focused on the realization of other object recognition methods, for instance based on the shape. Other methods relying, for instance, on particle filters and similar can be investigated in such a framework. The stability proof can be enhanced by considering the camera calibration errors as well. Moreover, a deep comparison between catching a thrown ball by using either a moving or a static single camera could be developed.

Appendix

The necessary quantities to be estimated in (8) are the linear position $\hat{\boldsymbol{p}}_o^c$, linear velocity $\dot{\hat{\boldsymbol{p}}}_o^c$, and angular velocity $\hat{\boldsymbol{\omega}}_o^c$ of the center of the ball with respect to Σ_c. Starting from the current estimate of the position $\boldsymbol{p}_0$ and velocity $\dot{\boldsymbol{p}}_0$ of the ball, the ballistic model (19) is numerically integrated in the time interval $[0, t_i]$. In this way, $\hat{\boldsymbol{p}}_o^c$ can be obtained at a certain time t. With the same numerical integration, it is also possible to obtain the estimate of the linear velocity $\dot{\hat{\boldsymbol{p}}}_o^c$. Finally, the angular velocity can be retrieved as

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

It is worth noting that the first estimates of $\boldsymbol{p}_0$ and $\dot{\boldsymbol{p}}_0$ are obtained after n_l measurements. Before that n_l measurements are collected, the initial values of $\boldsymbol{p}_0$ and $\dot{\boldsymbol{p}}_0$ should be anyhow provided to compute the above mentioned

qualities. Hence, a statistical calibration has been preliminary realized to retrieve a rough initial estimation of $\boldsymbol{p}_0$ and $\dot{\boldsymbol{p}}_0$.

References

[1] V. Lippiello, F. Ruggiero, 3D monocular robotic ball catching with an iterative trajectory estimation refinement, in: IEEE International Conference on Robotics and Automation, St. Paul, MN, 2012, pp. 3950–3955.

[2] C. Borst, M. Fischer, S. Haidacher, H. Liu, G. Hirzinger, DLR hand II: Experiments and experiences with an anthropomorphic hand, in: IEEE International Conference on Robotics and Automation, Taipei, 2003, pp. 702–707.

[3] V. Lippiello, F. Ruggiero, B. Siciliano, L. Villani, Visual grasp planning for unknown objects using a multifingered robotic hand, IEEE/ASME Transactions on Mechatronics 18 (3) (2013) 1050–1059.

[4] Q. He, C. Hu, W. Liu, N. Wei, M. Meng, L. Liu, C. Wang, Simple 3-D point reconstruction methods with accuracy prediction for multiocular system, IEEE/ASME Transactions on Mechatronics 18 (1) (2013) 366–375.

[5] B. Hove, J. Slotine, Experiments in robotic catching, in: American Control Conference, Boston, MA, 1991, pp. 380–386.

[6] U. Frese, B. Bauml, S. Haidacher, G. Schreiber, I. Schaefer, M. Hahnle, G. Hirzinger, Off-the-shelf vision for a robotic ball catcher, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, Maui, HI, 2001, pp. 1623–1629.

[7] O. Birbach, U. Frese, B. Bauml, Realtime perception for catching a flying ball with a mobile humanoid, in: IEEE International Conference on Robotics and Automation, Shanghai, PRC, 2011, pp. 5955–5962.

[8] C. Santos, M. Ferreira, Ball catching by a Puma arm: A nonlinear dynamical systems approach, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, PRC, 2006, pp. 916–921.

[9] K. Deguchi, H. Sakurai, S. Ushida, A goal oriented just-in-time visual servoing for ball catching robot arm, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, Nice, F, 2008, pp. 3034–3039.

[10] C. Lin, Y. Chiu, The DSP based catcher robot system with stereo vision, in: IEEE/ASME International Conference on Advanced Intelligent Mechatronics, Xi'an, PRC, 2008, pp. 897–903.

[11] G. Buttazzo, B. Allotta, F. Fanizza, Mousebuster: A robot for real-time catching, IEEE Control Systems Magazine 14 (1) (1994) 49–56.

[12] D. Fernandes, P. Lima, A testbed for robotic visual servoing and catching of moving objects, in: IEEE International Conference on Electronics, Circuits and Systems, Lisboa, P, 1998, pp. 475–478.

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

[14] R. Herrejon, S. Kagami, K. Hashimoto, 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, ROC, 2009, pp. 526–531.

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

[16] S. Das, R. Das, Using reinforcement learning to catch a baseball, in: IEEE International Conference on Neural Networks, Orlando, FL, 1994, pp. 2808–2812.

[17] R. Mori, F. Miyazaki, Examination of human ball catching strategy though autonomous mobile robot, in: IEEE International Conference on Robotics and Automation, Washington, DC, 2002, pp. 4236–4241.

[18] R. Mori, F. Miyazaki, GAG (gaining angle of gaze) strategy for ball tracking and catching task, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, Lausanne, CH, 2002, pp. 281–286.

[19] R. Mori, K. Hashimoto, F. Miyazaki, Tracking and catching of 3d flying target based on GAG strategy, in: IEEE International Conference on Robotics and Automation, New Orleans, LA, 2004, pp. 5189–5193.

[20] F. Takagi, H. Sakahara, T. Tabata, H. Yamagishi, T. Suzuki, Navigation control for tracking and catching a moving target, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, MO, 2009, pp. 866–871.

[21] Y. Imai, A. Namiki, K. Hashimoto, M. Ishikawa, 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, LA, 2004, pp. 1849–1854.

[22] A. Namiki, M. Ishikawa, The analysis of high-speed catching with a multifingered robot hand, in: IEEE International Conference on Robotics and Automation, Barcelona, E, 2005, pp. 2655–2660.

[23] Y. Tanaka, T. Tsuji, M. Kaneko, Task readiness impedance in human arm movements for virtual ball-catching task, in: Conference of the IEEE Industrial Electronic Society, Roanoke, VA, 2003, pp. 478–483.

[24] G. Batz, A. Yaqub, H. Wu, K. Kuhnlenz, D. Wollherr, M. Buss, Dynamic manipulation: Nonprehensile ball catching, in: 18th Mediterranean Conference on Control and Automation, Marrakech, MA, 2010, pp. 365–370.

[25] K. Nishiwaki, A. Konno, K. Nagashima, M. Inaba, H. Inoue, The humanoid Saika that catches a thrown ball, in: IEEE International Workshop on Robot and Human Communication, Sendai, J, 1997, pp. 94–99.

[26] M. Kitazawa, J. Wu, Y. Sakai, A new method of 3-d movie based on 2-d photo images for the virtual playing catch system, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, Takamatsu, J, 2000, pp. 157–162.

[27] M. Bratt, C. Smith, H. Christensen, Minimum jerk based prediction of user actions for a ball catching task, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, 2007, pp. 2710–2716.

[28] C. Smith, M. Bratt, H. Christensen, Teleoperation for a ball-catching task with significant dynamics, Neural Networks 21 (1) (2008) 604–620.

[29] E. Ribnick, S. Atev, N. Papanikolopoulos, O. Masoud, R. Voyles, Detection of thrown objects in indoor and outdoor scenes, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, 2007, pp. 979–984.

[30] J. Bouguet, Camera calibration toolbox for Matlab, Tech. rep., California Institute of Technology (2010).
URL `http://www.vision.caltech.edu/bouguetj/`

[31] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, Robotics: Modelling, Planning and Control, Springer, London, UK, 2008.

[32] K. Deguchi, Optimal motion control for image-based visual servoing by decoupling translation and rotation, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, Victoria, CDN, 1998, pp. 705–711.

[33] F. Chaumette, Potential Problems of Stability and Convergence in Image-based and Position-based Visual Servoing, Springer, London, UK, 1998.

[34] V. Lippiello, F. Ruggiero, Monocular eye-in-hand robotic ball catching with parabolic motion estimation, in: 10th International IFAC Symposium on Robot Control, Dubrovnik, HR, 2012, pp. 229–234.