

Real-Time Visual Tracking of 3D Objects

Fabrizio Caccavale¹, Vincenzo Lippiello², Bruno Siciliano², and Luigi Villani²

¹ Dipartimento di Ingegneria e Fisica dell'Ambiente
Università della Basilicata
Contrada Macchia Romana, 85100 Potenza, Italy
caccavale@unibas.it

² PRISMA Lab
Dipartimento di Informatica e Sistemistica
Università di Napoli Federico II
Via Claudio 21, 80125 Napoli, Italy
<[@unina.it">vincenzo.lippiello,bruno.siciliano,luigi.villani](mailto:vincenzo.lippiello,bruno.siciliano,luigi.villani)>@unina.it
<http://www.prisma.unina.it>

Abstract. The use of visual sensors may have high impact in applications where it is required to measure the pose (position and orientation) and the visual features of objects moving in unstructured environments. In robotics, the measurements provided by video cameras can be directly used to perform closed-loop control of the robot end-effector pose. In this chapter the problem of real-time estimation of the position and orientation of a moving object using a fixed stereo camera system is considered. An approach based on the use of the Extended Kalman Filter (EKF) combined with a 3D representation of the objects geometry based on Binary Space Partition (BSP) trees is illustrated. The performance of the proposed visual tracking algorithm is experimentally tested in the case of an object moving in the visible space of a fixed stereo camera system.

1 Introduction

In the last decade, research on visual sensing has received a new impulse because digital signal processing hardware with high computational capability is becoming available at low cost. In fact, visual sensors offer the possibility to extract a great variety of information from a scene in a noninvasive manner. This information can be used by automatic systems either at high level, e.g., for inspection, recognition and planning tasks, and at low-level, e.g., for autonomous guidance of vehicles, real-time control in scarcely structured environments.

In robotics, the measurements provided by video cameras can be directly used to perform closed-loop position/orientation control of the robot end effector, usually denoted as *visual servoing* control [11]. In this framework, two different approaches have been developed. The first approach is the position-based visual servoing, which defines the tracking error in the Euclidean space and requires the estimation of the position and orientation of a target object with respect to a reference frame [27]. The second approach is the image-based visual servoing, which defines a tracking error directly in the image space of the cameras, thus avoiding accurate calibration of the vision system [5,10]. Hybrid methods using position-based visual servoing to control

certain degrees of freedom and image-based visual servoing to control the remaining degrees of freedom can be adopted [7,19]. More recently, vision measurements have been used in combination with force measurements to develop control strategies aimed at improving the robot performance for the execution of tasks in scarcely structured environments [2].

In the case of position-based visual servo, computationally efficient techniques for *visual tracking* of the pose (position and orientation) of the target object must be adopted. One of the major problems to cope with is represented by the noise and disturbances affecting the visual measurements, due to temporal and spatial sampling and quantization of the image signal, lens distortion, etc., which may produce large pose estimation errors. The use of the Extended Kalman Filter (EKF) may improve the accuracy and speed of the visual tracking algorithm [15,25,21,17].

In fact, Kalman filtering offers many advantages over other pose estimation methods [1,9,28], e.g., implicit solution of photogrammetric equations with recursive implementation, temporal filtering, ability to change the measurement set during the operation. Moreover, the statistical properties of Kalman filter may be tuned to those of the image measurements noise of the particular vision system. Last but not least, the prediction capability of the filter allows setting up a dynamic windowing technique of the image plane which may sensibly reduce image processing time. Applications of Kalman filter in machine vision range from visual tracking of objects with many internal degrees of freedom [20], to automatic grasp planning [13] as well as pose and size parameters estimation of objects with partially known geometry [14].

A widely adopted strategy for object pose computation is based on the recognition of some geometric features of the object, such as edges and corners, from a camera image. In particular, the extraction of a suitable number of corners (feature points) allows computing the pose by using a simple point CAD model of the object [27,13]. In principle, the accuracy of the estimate increases with the number of the available feature points, at the expense of the computation time. However, when Kalman filter is adopted, it has been shown that the best achievable accuracy that can be obtained using all the available points is quite the same as that obtained using a number of five or six feature points, if properly chosen [25].

The choice of the optimal feature points can be performed by using suitable selection algorithms, whose complexity grows at factorial rate with the number of the available points [6,12]. Hence, to reduce the computational burden in the presence of a large number of feature points, it is crucial to perform a pre-selection, e.g., by eliminating all the points that, in a given object pose, are occluded with respect to the camera [24,8].

In this chapter, the EKF is adopted for real-time visual tracking of an object in the Euclidean space. In order to reduce computational time, a new pre-selection algorithm of the feature points is proposed, based on the selection of all the points that are visible to the camera at a given sample time. This algorithm exhibits a complexity which grows linearly, thanks to the use of Binary Space Partitioning (BSP) tree for object geometric representation [4]. In detail, the prediction of the object pose provided by the Kalman filter is used to drive a visit algorithm of the

BSP tree which allows identifying all the feature points that are visible at the next sample time. After the pre-selection, a dynamic windowing algorithm and an optimal point selection algorithm are adopted to find the windows of the image plane to be processed and input to the Kalman filter.

The proposed pre-selection algorithm can be used also in the case of objects and obstacles with interposing parts. Differently from other algorithms (see [12] and references therein), this method allows recognizing all the points of the surfaces of the objects which are hidden to the camera or occluded by some other objects or obstacles of known geometry [18].

The effectiveness of the proposed approach is tested in experimental case studies where the position and orientation of an object carried by a robot manipulator is tracked using both one fixed camera and a stereo system composed by two fixed cameras.

The chapter is organized as follows. In Section 2 the pin-hole model of the cameras is introduced and the photogrammetric equations are derived. The model used for object motion and the equations of the EKF are presented in Section 3. In Section 4 a BSP tree is derived from a CAD geometric model of an object. Section 5 is devoted to illustrate the pre-selection algorithm and an optimal point selection technique, based on dynamic windowing and quality indices. The whole estimation procedure is analyzed in Section 6. Finally, the experimental set up and the experimental tests are described in Section 7, while Section 8 presents some concluding remarks and open problems. Details on the derivation of EKF equations are reported in the Appendix.

2 Modelling

Consider system of n video cameras fixed with respect to a base coordinate frame $O-xyz$ and the pin-hole model of camera i (see Fig. 1). Let $O_{ci}-x_{ci}y_{ci}z_{ci}$ be a frame attached to the camera (camera frame), with the z_{ci} -axis aligned to the optical axis and the origin in the optical center. In the following, a superscript will be used to denote the reference frame of a variable, when different from the base frame.

For each camera, the sensor plane is parallel to the $x_{ci}y_{ci}$ -plane at a distance $-f_e^{ci}$ along the z_{ci} -axis, where f_e^{ci} is the effective focal length of the camera lens, which may be different from the nominal focal length f^{ci} . The image plane is parallel to the $x_{ci}y_{ci}$ -plane at a distance f_e^{ci} along the z_{ci} -axis. The intersection of the optical axis with the image plane defines the principal optic point O'_{ci} , which is the origin of the image frame $O'_{ci}-u_{ci}v_{ci}$ whose axes u_{ci} and v_{ci} are taken parallel to the axes x_{ci} and y_{ci} , respectively.

A point P with coordinates $\mathbf{p}^{ci} = [x^{ci} \ y^{ci} \ z^{ci}]^T$ in the i -th camera frame is projected onto the point of the image plane whose coordinates can be computed with the equation

$$\begin{bmatrix} u^{ci} \\ v^{ci} \end{bmatrix} = \frac{f_e^{ci}}{z^{ci}} \begin{bmatrix} x^{ci} \\ y^{ci} \end{bmatrix} \quad (1)$$

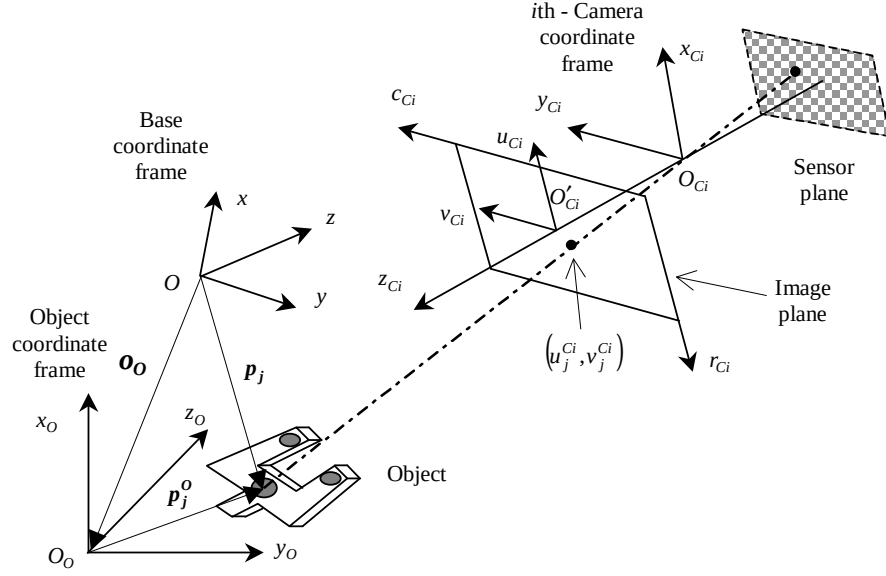


Fig. 1. Reference frames for the i -th camera and the object using the pin-hole model.

which is known as perspective transformation. A spatial sampling can be applied to the image plane by expressing the coordinates in terms of number of pixels as

$$\begin{bmatrix} r^{ci} \\ c^{ci} \end{bmatrix} = \begin{bmatrix} r_0^{ci} \\ c_0^{ci} \end{bmatrix} + \begin{bmatrix} s_u^{ci} & 0 \\ 0 & s_v^{ci} \end{bmatrix} \begin{bmatrix} u^{ci} \\ v^{ci} \end{bmatrix} \quad (2)$$

being $[r_0^{ci} \ c_0^{ci}]^T$ the coordinates of the point O'_{ci} whereas s_u^{ci} and s_v^{ci} are the row and column scaling factor, respectively, for the i -th camera.

Consider an object frame $O_o-x_o y_o z_o$ attached to target object. The position and orientation of the object frame with respect to the base frame can be expressed in terms of the coordinate vector of the origin $\mathbf{o}_o = [x_o \ y_o \ z_o]^T$ and of the rotation matrix $\mathbf{R}_o(\phi_o)$, where the components of $\phi_o = [\varphi_o \ \vartheta_o \ \psi_o]^T$ are the Roll, Pitch and Yaw angles. The components of the vectors $\mathbf{o}_o$ and ϕ_o are the six unknown quantities to be estimated.

Consider m feature points of the object. The coordinate vector $\mathbf{p}_j^{ci}$ of the feature point P_j ($i = 1, \dots, n, j = 1, \dots, m$) can be expressed in the i -th camera frame as

$$\mathbf{p}_j^{ci} = \mathbf{R}_{ci}^T(\mathbf{o}_o - \mathbf{o}_{ci} + \mathbf{R}_o(\phi_o)\mathbf{p}_j^o), \quad (3)$$

where $\mathbf{o}_{ci}$ and $\mathbf{R}_{ci}$ are, respectively, the position vector and the rotation matrix of the i -th camera frame referred to the base frame, $\mathbf{p}_j^o$ is the coordinate vector of P_j expressed in the object frame. Notice that $\mathbf{p}_j^o$ is a constant vector that is assumed to be known, since it can be computed from a CAD model of the object or via a suitable calibration procedure. Moreover, the quantities $\mathbf{o}_{ci}$ and $\mathbf{R}_{ci}$ are constant, because

each camera is assumed to be fixed to the workspace, and can be computed through a suitable calibration procedure [26].

By folding the $3m$ equations (3) into the perspective transformation (1) of the n cameras and into Eq. (2), a system of $2mn$ nonlinear equations is achieved. The equations depend on the measurements of the m feature points in the image plane of each cameras, while the six components of the vectors $\mathbf{o}_o$ and ϕ_o are the unknown variables. To solve these equations at least six independent equations are required.

The computation of the solution is nontrivial and for visual servoing applications it has to be repeated at a high sampling rate. The recursive Kalman filter provides a computationally tractable solution, which can also incorporate redundant measurement information.

3 Kalman Filtering

In order to estimate the pose of the object, a discrete time state space dynamic model has to be considered, describing the object motion. The state vector of the dynamic model is chosen as the (12×1) vector

$$\mathbf{w} = [x_o \ \dot{x}_o \ y_o \ \dot{y}_o \ z_o \ \dot{z}_o \ \varphi_o \ \dot{\varphi}_o \ \vartheta_o \ \dot{\vartheta}_o \ \psi_o \ \dot{\psi}_o]^T. \quad (4)$$

For simplicity, the object velocity is assumed to be constant over one sample period T . This approximation is reasonable in the hypothesis that T is sufficiently small. The corresponding dynamic modelling error can be considered as an input disturbance γ described by zero mean Gaussian noise with covariance given by the (12×12) matrix $\mathbf{Q}$. The discrete time dynamic model can be written as

$$\mathbf{w}_k = \mathbf{A}\mathbf{w}_{k-1} + \gamma_k \quad (5)$$

where $\mathbf{A}$ is a (12×12) block diagonal matrix of the form

$$\mathbf{A} = \text{diag} \left\{ \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}, \dots, \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \right\}.$$

The output of the Kalman filter, for each camera, is the vector of the *normalized* coordinates of the m feature points in the image plane of the camera

$$\boldsymbol{\zeta}_k = \left[\frac{u_1^{c1}}{f_e^{c1}} \frac{v_1^{c1}}{f_e^{c1}} \cdots \frac{u_1^{cn}}{f_e^{cn}} \frac{v_1^{cn}}{f_e^{cn}} \cdots \frac{u_m^{c1}}{f_e^{c1}} \frac{v_m^{c1}}{f_e^{c1}} \cdots \frac{u_m^{cn}}{f_e^{cn}} \frac{v_m^{cn}}{f_e^{cn}} \right]_k^T. \quad (6)$$

In view of (1), the corresponding output model can be written in the form

$$\boldsymbol{\zeta}_k = \mathbf{g}(\mathbf{w}_k) + \boldsymbol{\nu}_k \quad (7)$$

where $\boldsymbol{\nu}_k$ is the measurement noise, which is assumed to be zero mean Gaussian noise with covariance given by the $(2m \times 2m)$ matrix $\mathbf{R}$, and the function $\mathbf{g}(\mathbf{w}_k)$ is

$$\mathbf{g}(\mathbf{w}_k) = \left[\frac{x_1^{c1}}{z_1^{c1}} \frac{y_1^{c1}}{z_1^{c1}} \cdots \frac{x_1^{cn}}{z_1^{cn}} \frac{y_1^{cn}}{z_1^{cn}} \cdots \frac{x_m^{c1}}{z_m^{c1}} \frac{y_m^{c1}}{z_m^{c1}} \cdots \frac{x_m^{cn}}{z_m^{cn}} \frac{y_m^{cn}}{z_m^{cn}} \right]_k^T. \quad (8)$$

The coordinates of the feature points $\mathbf{p}_j^{ci}$ in (8) are computed from the state vector $\mathbf{w}_k$ via (3). Matrix $\mathbf{R}$ can be evaluated during the camera calibration procedure or by means of specific experiments.

Since the output model is nonlinear in the system state, it is required to linearize the output equations about the current state estimate at each sample time. This leads to the so-called Extended Kalman Filter (EKF).

The first step of the EKF algorithm provides an optimal estimate of the state at the next sample time according to the recursive equations

$$\hat{\mathbf{w}}_{k,k-1} = \mathbf{A}\hat{\mathbf{w}}_{k-1,k-1} \quad (9)$$

$$\mathbf{P}_{k,k-1} = \mathbf{A}\mathbf{P}_{k-1,k-1}\mathbf{A}^T + \mathbf{Q}_{k-1}, \quad (10)$$

where $\mathbf{P}_{k,k-1}$ is the (12×12) covariance matrix of the estimate state error. The second step improves the previous estimate by using the input measurements according to the equations

$$\hat{\mathbf{w}}_{k,k} = \hat{\mathbf{w}}_{k,k-1} + \mathbf{K}_k(\zeta_k - \mathbf{g}(\hat{\mathbf{w}}_{k,k-1})) \quad (11)$$

$$\mathbf{P}_{k,k} = \mathbf{P}_{k,k-1} - \mathbf{K}_k\mathbf{C}_k\mathbf{P}_{k,k-1}, \quad (12)$$

where $\mathbf{K}_k$ is the $(12 \times 2m)$ Kalman matrix gain

$$\mathbf{K}_k = \mathbf{P}_{k,k-1}\mathbf{C}_k^T(\mathbf{R}_k + \mathbf{C}_k\mathbf{P}_{k,k-1}\mathbf{C}_k^T)^{-1}, \quad (13)$$

being $\mathbf{C}_k$ the $(2m \times 12)$ Jacobian matrix of the output function

$$\mathbf{C}_k = \left. \frac{\partial \mathbf{g}(\mathbf{w})}{\partial \mathbf{w}} \right|_{\mathbf{w}=\hat{\mathbf{w}}_{k,k-1}}. \quad (14)$$

The analytic expression of $\mathbf{C}_k$ can be found in the Appendix.

4 BSP Tree Geometric Modelling

The accuracy of the estimate provided by the Kalman filter depends on the number of the available feature points. Inclusion of extra points will improve the estimation accuracy but will increase the computational cost. It has been shown that a number of five or six feature points, if properly chosen, may represent a good trade-off [25]. Selection algorithms have been developed to find the optimal feature points [12]. In order to increase the efficiency of the selection algorithms, it is advisable to perform a pre-selection of the points that are visible to the camera at a given sample time. The pre-selection technique proposed in this chapter is based on Binary Space Partitioning (BSP) trees.

A BSP tree is a data structure representing a recursive and hierarchical partition of a n -dimensional space into convex subspaces. It can be effectively adopted to represent the 3D CAD geometry of an object [22].

In order to build the tree, each object has to be modelled as a set of planar *polygons*; this means that the curved surfaces have to be approximated. Each polygon

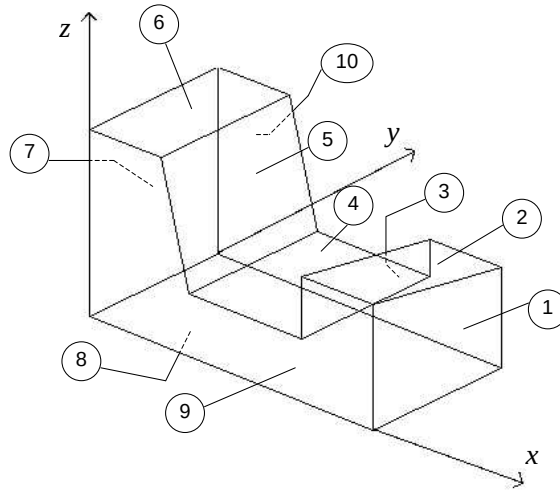


Fig. 2. Object and corresponding polygons.

is characterized by a set of *feature points* (the vertices of the polygon) and by the vector normal to the plane leaving from the object. For each node of the tree, a *partition plane*, characterized by its normal vector and a point, is chosen according to a specific criterion; the node is defined as the set containing the partition plane and all the polygons lying on it. The choice of the partition planes depends on how the tree will be used. For the purpose of removing the hidden surfaces, it is necessary to choose the partition planes in the set of the planes containing the polygons corresponding to the object surfaces.

The first node of the tree can be arbitrarily chosen. Different choices determine different trees. For the application considered here, the structure of the tree is not important because the visit algorithm must consider all the nodes.

Each node is the root of two subtrees: the *front* subtree corresponding to the subset of all the polygons lying entirely on the front side of the partition plane (i.e. the side corresponding to the half-space containing the normal vector), and the *back* subtree corresponding to the subset of all the polygons lying entirely on the back side of the partition plane.

The construction procedure can be applied recursively to the two subsets by choosing, for each node, a new partition plane among those corresponding to the polygons contained in that subtree.

If a polygon intersects the partition plane, it can be split into two or more pieces and the resulting parts are added to the corresponding subsets.

The construction ends when all the polygons and their parts are placed in a node of the tree.

As an example, consider the object represented in Fig. 2, which contains ten polygons. A possible BSP tree representation of the object is reported in Fig. 3, which

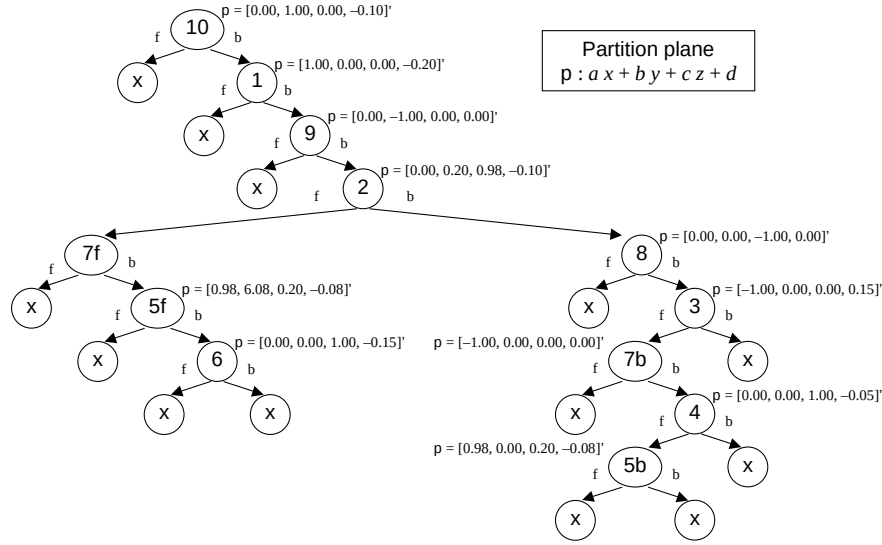


Fig. 3. BSP tree of the object.

has been obtained considering as root node the partition plane containing polygon number 10. A partition plane is represented by the vector $\pi = [a \ b \ c \ d]^T$ of the coefficients of the equation of the plane with respect to a base reference frame

$$ax + by + cz + d = 0,$$

where $n = [a \ b \ c]^T$ is the unit vector normal to the plane. The root of the tree contains the polygon number 10; the front subtree is empty while the back subtree contains all the remaining polygons. The partition plane of the back subtree contains the polygon number 1; the front subtree is empty while the back subtree contains the polygons from number 2 to number 9. The construction ends when all the polygons are added to the nodes of the tree. Remarkably, the partition plane containing the polygon number 2 cuts polygons number 5 and 7 (notice that polygons number 9 and 10, which also intersect the partition plane, were already added to previous nodes of the tree), and thus they have been split into two pieces each (see polygons number 5f, 5b, 7f, 7b in Fig. 4).

In most cases, however, it is possible to choose the partition planes so that splitting of polygons is avoided. In this way the construction process of the tree and the visit algorithm are faster. This solution has to be preferred when the BSP tree must be built on line [18].

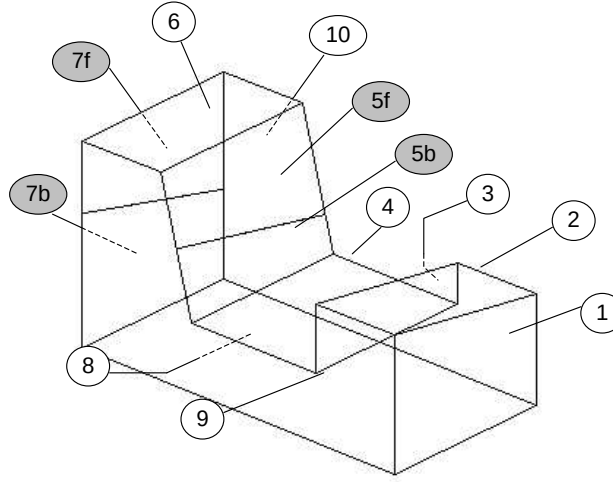


Fig. 4. Case of partition plane splitting polygons number 5 and number 7.

5 Features Selection

5.1 Pre-Selection Algorithm

Once a BSP tree representation of an object is available, it is possible to select the feature points of the object that are visible from a given camera position and orientation, by implementing a suitable visit algorithm of the tree. The algorithm can be applied recursively to all the nodes of the tree, starting from the root node as showed in Fig. 5, by updating a current set of visible feature points as follows.

For the current node, classify the camera position with respect to the current partition plane: **Front** side, **Back** side, **On** the plane. Hence:

- **Front:** Visit the back subtree; process the node; visit the front subtree.
- **Back:** Visit the front subtree; process the node; visit the back subtree.
- **On:** Visit the front subtree; visit the back subtree.

When the algorithm processes a node, the current set of projections of the visible feature points on the image plane is updated by adding all the projections of the feature points of the polygons of the current node and eliminating all the projections of the feature points that are hidden by the projections of the polygons of the current node.

If a polygon is hidden from the camera (i.e., the angle between the normal vector to the polygon and the camera z -axis is not in the interval $]-\pi/2, \pi/2[$ or the polygon is behind the camera), the corresponding feature points are not added to the set.

At the end of the visit, the current set will contain all the projections of the feature points visible from the camera, while all the hidden feature points will be discarded.

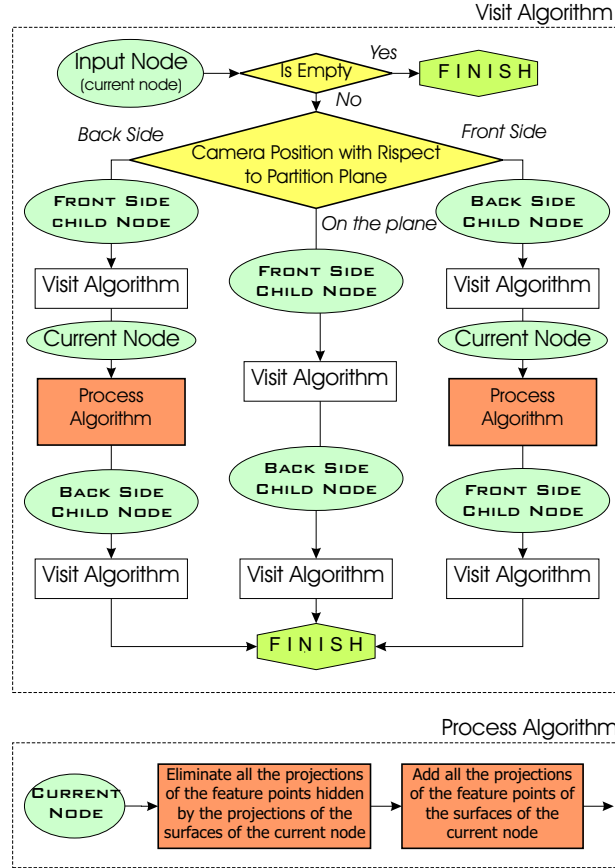


Fig. 5. Recursive visit algorithm of the BSP tree for the selection of visible feature points.

Notice that the visit algorithm updates the set by ordering the polygons with respect to the camera from the background to the foreground.

With reference to the BSP tree of Fig. 3, assuming that the camera is placed as the observer of the image in Fig. 4, the sequence of the processed nodes is: 10, 8, 7b, 4, 5b, 3, 2, 7f, 6, 5f, 9, 1, where the polygons number 10, 8, 7b, 3, 7f turn out to be hidden at the same time and will not be processed.

The technique described above can be suitably exploited to set up a real-time pre-selection algorithm of the feature points on the camera image plane, using the prediction of the estimated pose of the target object provided by the Kalman filter.

5.2 Selection Algorithm

The pre-selection technique recognizes all the feature points that are visible from a camera view point. However, this does not ensure that all the visible points are

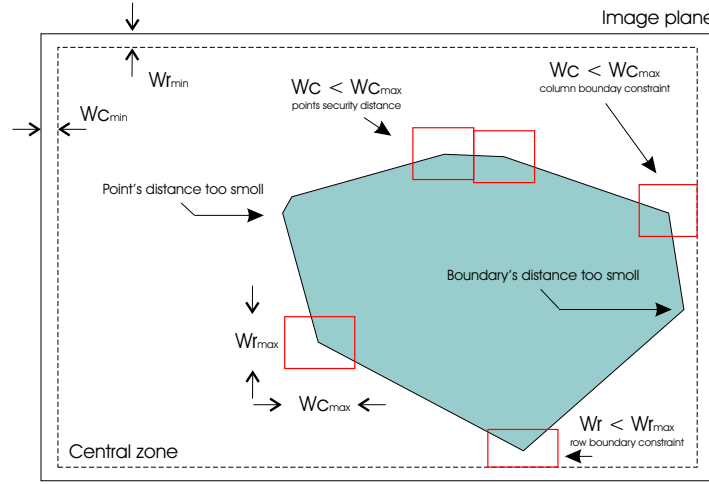


Fig. 6. Examples of significant situations during windowing test

“well” localizable, i.e., their positions can be effectively measured with a given accuracy. For instance, some points could be out of the field of view of the camera, or they could be too close to each other to guarantee absence of ambiguity in the localization. Moreover, the number of the well localizable feature points may be larger than the *optimal* number of points ensuring the best pose estimation accuracy.

In the following, a windowing test is adopted to select the projections of the feature points that can be well localized. Then, a selection algorithm is used to choose an optimal subset of points to be considered for feature extraction.

Windowing test The measurements of the coordinates of the projections of the feature points are obtained by considering suitable rectangular windows of the image plane to be grabbed and processed. Each window must contain one feature point. The windows are centered on the positions of the feature points on the image plane so as predicted by the Kalman filter. Their semi-dimensions are dynamically chosen in the interval $[W_{r_{min}}, W_{r_{max}}]$ for the base (the side parallel to the row’s direction) and in the interval $[W_{c_{min}}, W_{c_{max}}]$ for the height (the side parallel to the column’s direction). The minimum values are set so as to achieve a prescribed accuracy and robustness in the feature extraction, while the maximum values are set on the basis of the available memory and processing time.

A windowing test can be set up to select all the projections of the feature points that can be “well” localized.

First, all the points that are out of the field of view of the camera, or too close to the boundaries of the image plane, are discarded. This is achieved by eliminating all the points whose projections, so as predicted by the Kalman filter, are out of a central window of the image plane. The central window is obtained by reducing the

height (base) of the whole image plane of the quantity $Wr_{min}(Wc_{min})$ from each side, as shown in Fig. 6.

Then, all the feature points that are too close to each other are discarded. This happens when the estimated distance between the projections of two or more points is lower than $S_f \cdot Wr_{min}(S_f \cdot Wc_{min})$ along the row's (column's) direction; $S_f > 1$ is a suitable security factor.

All the remaining points are “well” localizable; the effective dimensions of the corresponding windows are dynamically adapted to the maximum allowable semi-dimension, so as to guarantee an assigned security distance from the other points and from the boundaries of the image plane (see Fig. 6).

Optimal feature points selection The number of feature points after pre-selection and windowing test is typically too high with respect to the minimum number sufficient to achieve the best Kalman filter precision. It has been demonstrated that an optimal set of five or six feature points guarantees about the same precision as that of the case when an higher number of feature points is considered [27,25].

The optimality of a set Γ of feature points is valued through the composition of suitably selected quality indexes into an optimal cost function. The quality indexes must be able to provide accuracy, robustness and to minimize the oscillations in the pose estimation variables. To achieve this goal it is necessary to ensure an optimal spatial distribution of the projections of the feature points on the image plan and to avoid chattering events between different optimal subsets of feature points chosen during the object motion. Moreover, in order to exploit the potentialities of a multi-camera system, it is important to achieve an optimal distribution of the feature points among the different cameras.

Without loss of generality, the case of two identical cameras is considered.

A first quality index is the measure of spatial distribution of the predicted projections on the image planes of a subset of q_i selected points for the i -th camera, $i = 1, 2$:

$$Q_{si} = \frac{1}{q_i} \sum_{k=1}^{q_i} \min_{\substack{j \in \{1, \dots, q_i\} \\ j \neq k}} \|p_j - p_k\|.$$

Notice that $q = q_1 + q_2$ is chosen between 6 and 8 to handle fault cases.

A second quality index is the measure of angular distribution of the predicted projections on the image planes of a subset of q_i selected points for the i -th camera, $i = 1, 2$:

$$Q_{ai} = 1 - \sum_{k=1}^{q_i} \left| \frac{\alpha_k}{2\pi} - \frac{1}{q_i} \right|$$

where α_k is the angle between the vector $p_{k+1} - p_{Ci}$ and the vector $p_k - p_{Ci}$, being p_{Ci} the central gravity point of the whole subset of feature points, and the q_i points of the subset are considered in a counter-clockwise ordered sequence with respect to p_{Ci} , with $p_{q_i+1} = p_1$.

In order to avoid chattering phenomena, the following quality index, which introduces hysteresis effects on the change of the optimal combination of points, is considered for the i -th camera, $i = 1, 2$:

$$Q_h = \begin{cases} 1 + \epsilon & \text{if } \Gamma = \Gamma_{opt} \\ 1 & \text{otherwise} \end{cases}$$

where ϵ is a positive constant and Γ_{opt} is the optimal set of feature points at the previous sample time.

In order to distribute the points among the two cameras, the following indexes are considered:

$$Q_e = 1 + \frac{2}{q} \left(\frac{2}{q} - 1 \right) \left| q_1 - \frac{q}{2} \right|$$

$$Q_d = \frac{q_1/d_1 + q_2/d_2}{q/\min\{d_1, d_2\}}$$

where q_i is the number of points assigned to the i -th camera, and d_i is the distance of the i -th camera from the object, $i = 1, 2$. The first index ensures an equal distribution of points among the cameras. The second index takes into account the distance of the cameras from the object, and thus allows managing different resolution zones of different cameras.

The proposed quality indexes represent only some of the possible choices, but guarantee satisfactory performance when used with the pre-selection method and the windowing test presented above, for the case of two fixed cameras. Other examples of quality indexes have been proposed [12], and some of them can be added to the indexes adopted here.

The cost function is chosen as

$$Q = Q_h \frac{Q_e Q_d}{q} (q_1 Q_{s1} Q_{a1} + q_2 Q_{s2} Q_{a2})$$

and must be evaluated for all the possible combinations of the visible points on q positions. In order to determine the optimal set at each sample time, the initial optimal combination of points is first evaluated off line. Then, only the combinations that modify at most one point for camera with respect to the current optimal combination are tested on line, thus achieving a considerable reduction of processing time.

It should be pointed out that, in some cases, the number of points resulting at the end of the pre-selection step may be too high to perform the optimal selection in a reasonable time. In such a cases, a computational cheaper solution, based on the optimal set at the previous time-step, can be adopted to find a sub-optimal set. For sufficiently small sampling time, the sub-optimal solution is very close or coincides with the optimal one.

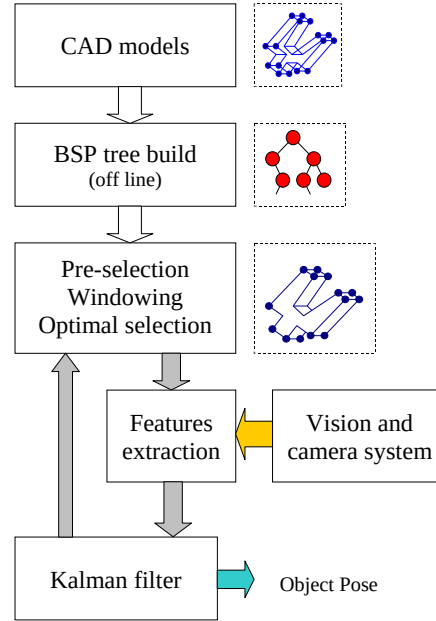


Fig. 7. Functional chart of the estimation procedure.

6 Estimation Procedure

A functional chart of the estimation procedure is reported in Fig. 7. It is assumed that a BSP tree representation of the object is built off-line from the CAD model. A Kalman filter is used to estimate the corresponding pose with respect to the base frame at the next sample time. The feature points selection and windows placing operation can be detailed as follows.

- **Step 1:** The visit algorithm described in the previous section is applied to the BSP tree of the object to find the set of all the feature points that are visible from the camera.
- **Step 2:** The resulting set of visible points is input to the algorithm for the selection of the optimal feature points.
- **Step 3:** The location of the optimal feature points in the image plane at the next sample time is computed on the basis of the object pose estimation provided by the Kalman filter.
- **Step 4:** A dynamic windowing algorithm is executed to select the parts of the image plane to be input to the feature extraction algorithm.

At this point, all the image windows of the optimal selected points are elaborated using a feature extraction algorithm. The computed coordinates of the points in the image plane are input to the Kalman filter which provides the estimate of the actual



Fig. 8. Robot COMAU SMART3-S and SONY 8500CE cameras.

object pose and the predicted pose at the next sample time used by the pre-selection algorithm.

Notice that the procedure described above can be extended to the case of multiple objects moving among obstacles of known geometry [18]; if the obstacles are moving with respect to the base frame, the corresponding motion variables can be estimated using Kalman filters.

7 Experiments

7.1 Experimental Set-Up

The experimental set-up is composed by a PC with Pentium IV 1.7GHz processor equipped with two MATROX Genesis boards, two SONY 8500CE B/W cameras, and a COMAU SMART3-S robot (see Fig 8). The MATROX boards are used as frame grabber and for a partial image processing (e.g., windows extraction from the image). The PC host is also used to realize the whole BSP structures management, the pre-selection algorithm, windows processing, the selection algorithm and the

Kalman filtering. Some steps of image processing have been parallelized on the MATROX boards and on the PC, so as to reduce computational time. The robot is used to move an object in the visual space of the camera; thus the object position and orientation with respect to the base frame of the robot can be computed from joint position measurements via the direct kinematic equation. In order to test the accuracy of the estimation provided by the Kalman filter, the cameras were calibrated with respect to the base frame of the robot using the calibration procedure presented in [26], where the robot is exploited to place a calibration pattern in some known pose of the visible space of the cameras. The cameras resolution is 576×763 pixels and the nominal focal length of the lenses is 16 mm, while the calibration parameters for the two cameras are shown in Table 1. Notice that the parameters resulting from the calibration procedure are slightly different for the two cameras, although their nominal values are equal.

Camera # 1
$r_0 = 187.96$
$c_0 = 318.20$
$f_u = -1955.84$
$f_v = 1953.41$
$\mathbf{o}_{c1} = [1.2244 \ -1.7437 \ 0.8540]^T \text{ m}$
$\mathbf{c}_1 = [-90.234^\circ \ -1.880^\circ \ 88.511^\circ]^T$
$\mathbf{d} = [0.018 \ -0.019 \ -0.024 \ 0.012 \ 0.194]^T$
Camera # 2
$r_0 = 263.76$
$c_0 = 369.64$
$f_u = -1966.35$
$f_v = 1958.10$
$\mathbf{o}_{c2} = [1.6149 \ -1.6565 \ 0.8623]^T \text{ m}$
$\mathbf{c}_2 = [-92.188^\circ \ -18.032^\circ \ -89.111^\circ]^T$
$\mathbf{d} = [0.009 \ 0.0048 \ -0.0027 \ -0.0096 \ 0.1393]^T$

Table 1. Calibration parameters resulting from the calibration procedure.

Vector ϕ_{ci} contains the Roll, Pitch and Yaw angles of the i -th camera frame with respect to the base frame corresponding to the matrix $\mathbf{R}_{ci}$, while the vector $\mathbf{d} = [g_1 \ g_2 \ g_3 \ g_4 \ d_1]^T$ contains the parameters used for compensating the distortion effects due to the imperfections of the lens profile and the alignment error of the optical system, as described in [26]. The estimated value of the residual mean triangulation error for the stereo camera system is 1.53 mm. The sampling time used for estimation is limited by the camera frame rate, which is about 26 fps. No particular illumination equipment has been used to test the robustness of the setup in the case of noisy visual measurements.

All the algorithms for BSP structure management, image processing and pose estimation have been implemented in ANSI C. The image features are the corners

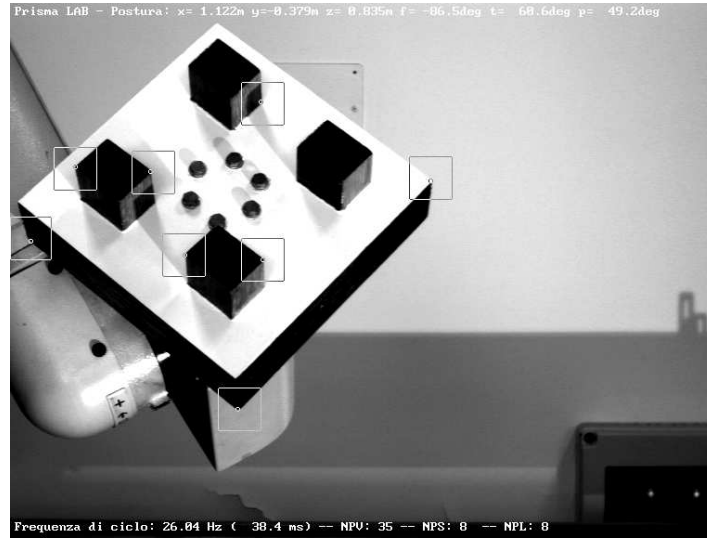


Fig. 9. Image seen by the camera with the windows selected for feature extraction. A point close to the center of each window marks the measured position of the corresponding feature point.

of the object, which can be extracted with high robustness in various environmental conditions. The feature extraction algorithm is based on Canny's method for edge detection [3] and on a simple custom implementation of a corner detector. In particular, to locate the position of a corner in a small window, all the straight segments are searched first, using an LSQ interpolator algorithm; then all the intersection points of these segments into the window are evaluated. The intersection points closer than a given threshold are considered as a unique average corner, due to the image noise. All the corners that are at a distance from the center of the window (which corresponds to the position of the corner so as predicted by the Kalman filter) greater than a maximum distance, are considered as fault measurements and are discarded. The maximum distance corresponds to the variance of the distance between the measured corner positions and those predicted by the Kalman filter.

The object used in the experiment is shown in Fig. 9, so as seen from the camera during the motion, as well as in Fig. 8, where the whole experimental setup is presented. The coordinates of the 40 vertices of the object, used as feature points, are reported in Table 2.

7.2 Experimental Results Using One Camera

Two different experiments have been realized for this case study. The first experiment reflects a favorable situation where the object moves in the visible space of the camera and most of the feature points that are visible at the initial time remain visible during

#	x^o	y^o	z^o	#	x^o	y^o	z^o
0	0.100	0.100	0.000	20	0.070	-0.039	0.092
1	0.100	-0.100	0.000	21	0.070	-0.070	0.092
2	-0.100	-0.100	0.000	22	0.029	-0.070	0.092
3	-0.100	0.100	0.000	23	0.029	-0.039	0.092
4	0.100	0.100	0.051	24	-0.029	-0.038	0.051
5	0.100	-0.100	0.051	25	-0.029	-0.069	0.051
6	-0.100	-0.100	0.051	26	-0.070	-0.070	0.051
7	-0.100	0.100	0.051	27	-0.070	-0.039	0.051
8	0.070	0.069	0.051	28	-0.029	-0.038	0.092
9	0.070	0.038	0.051	29	-0.029	-0.069	0.092
10	0.029	0.038	0.051	30	-0.070	-0.070	0.092
11	0.029	0.069	0.051	31	-0.070	-0.039	0.092
12	0.070	0.069	0.092	32	-0.028	0.069	0.051
13	0.070	0.038	0.092	33	-0.028	0.038	0.051
14	0.029	0.038	0.092	34	-0.069	0.039	0.051
15	0.029	0.069	0.092	35	-0.069	0.069	0.051
16	0.070	-0.039	0.051	36	-0.028	0.069	0.092
17	0.070	-0.070	0.051	37	-0.028	0.038	0.092
18	0.029	-0.070	0.051	38	-0.069	0.039	0.092
19	0.029	-0.039	0.051	39	-0.069	0.069	0.092

Table 2. Feature points coordinates with respect to the object frame, expressed in meters.

all the motion. The second experiment reflects an unfortunate situation where the set of the visible points is very variable, and a large part of the object goes out of the visible space of the camera during the motion.

The time history of the trajectory used for the first experiment is represented in Fig. 10. The maximum linear velocity is about 3 cm/s and the maximum angular velocity is about 3 deg/s.

The time history of the estimation errors is shown in Fig. 11. Noticeably, the accuracy of the system reaches the limit allowed by camera calibration, for all the components of the motion. As it was expected, the errors for some motion components are larger than others because only 2D information is available in a single camera system. In particular, the estimation accuracy is lower along z_c axis for the position, and about x_c and y_c axis for the orientation. Considering that in the experiment the z_c axis is almost aligned and opposed to the y axis of the base frame, the estimation errors are larger for the y component of the position, as well as the *roll* and *yaw* components of the orientation.

In Fig. 12 the output of the whole selection algorithm is reported. For each of the 40 feature points, two horizontal lines are considered: a point of the bottom line indicates that the feature point was classified as visible by the pre-selection algorithm at a particular sample time; a point of the top line indicates that the visible feature point was chosen by the selection algorithm. Notice that 8 feature points are selected at each sample time, in order to guarantee at least five or six measurements

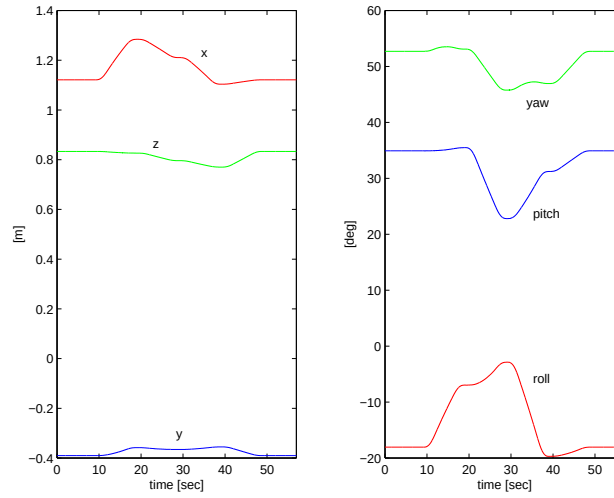


Fig. 10. Object trajectory with respect to the base frame used in the first experiment: position trajectory (*left*); orientation trajectory (*right*).

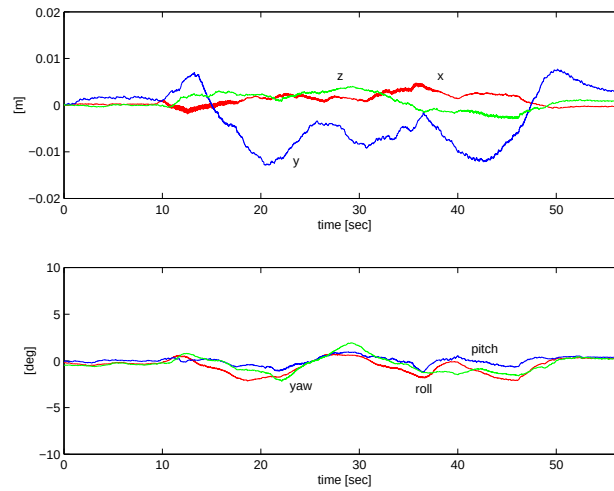


Fig. 11. Time history of the estimation errors in the first experiment: position errors (*top*); orientation errors (*bottom*).

in the case of fault of the extraction algorithm for some of the points. Also, some feature points are hidden during all the motion, while point number1 is only visible over some time intervals. Finally, no chattering phenomena are present.

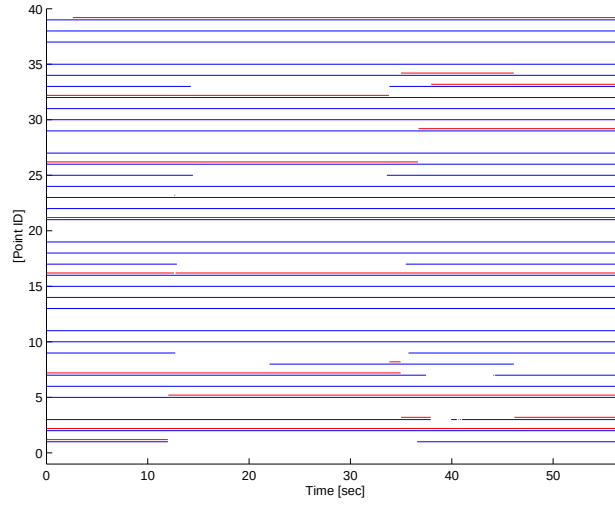


Fig. 12. Visible points and selected points in the first experiment. For each point, the bottom line indicates when it is visible, the top line indicates when it is selected for feature extraction.

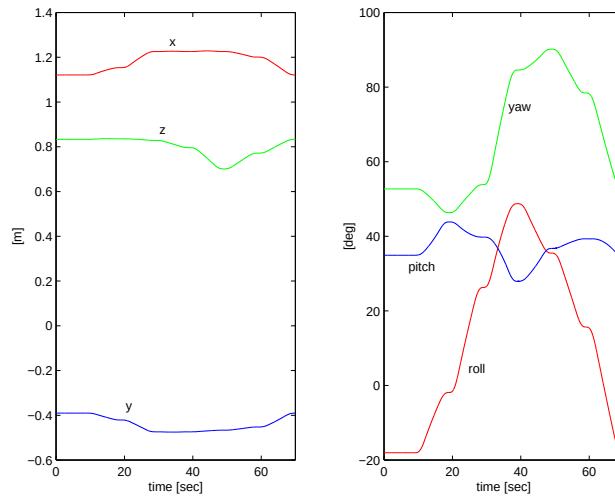


Fig. 13. Object trajectory with respect to the base frame used in the second experiment: position trajectory (*left*); orientation trajectory (*right*).

The time history of the trajectory used for the second experiment is represented in Fig. 13. The maximum linear velocity is about 2 cm/s and the maximum angular velocity is about 7 deg/s.

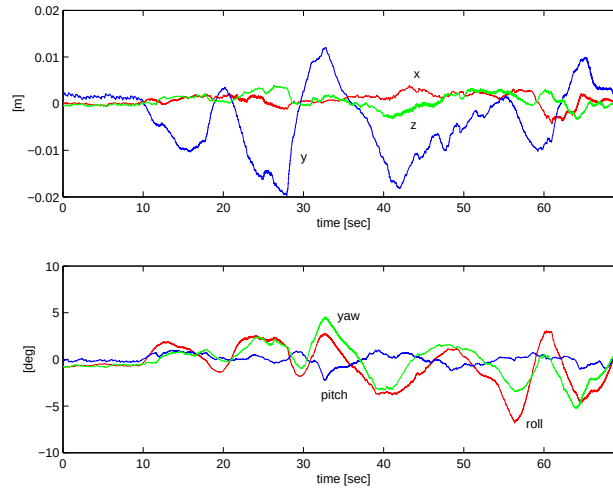


Fig. 14. Time history of the estimation errors in the second experiment: position errors (*top*); orientation errors (*bottom*).

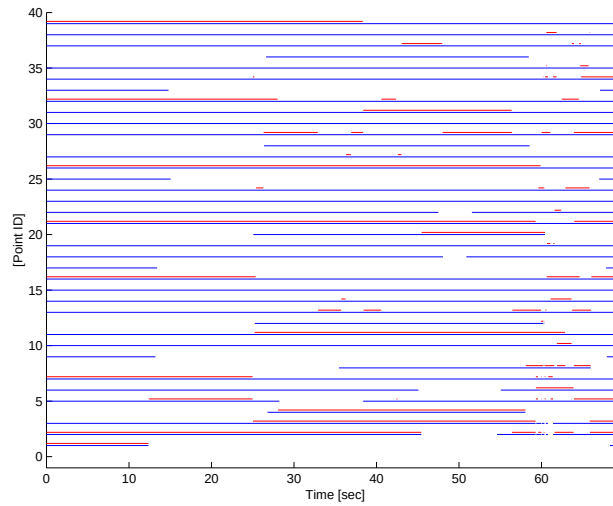


Fig. 15. Visible and selected points for the second experiment. For each point, the bottom line indicates when it is visible, the top line indicates when it is selected for feature extraction.

The time history of the estimation error is shown in Fig. 14. It can be observed that the error remains low but is greater than the estimation error of the previous experiment. This is due to the fact that from $t = 10$ s to $t = 60$ s the object moves so that it is partially out of the visible space of the camera; also, it rotates in such a way that a side remains almost parallel to the image plane. In this situation, just a few

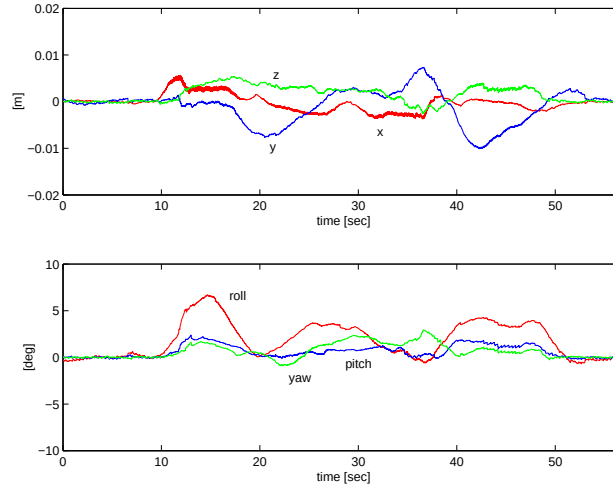


Fig. 16. Time history of the estimation errors in the case of two cameras: position errors (*top*); orientation errors (*bottom*).

feature points are visible; in addition, their projections on the image plane tend to be close or aligned so that the points that can be well localizable is further reduced and/or the spatial and angular distribution of the selected points is not optimal. This fact penalizes the estimation accuracy and explains how the magnitude of the estimation error components is one order greater than in the previous experiment, especially for the y component for the position error and the “roll” and “yaw” components of the orientation errors. The corresponding output of the pre-selection and selection algorithms are reported in Fig. 15. It should be pointed out that the pre-selection and selection algorithm are able to provide the optimal set of points independently from the operating condition, although slight chattering phenomena appear in some situation where the elements in set of localizable points is rapidly changing.

7.3 Experimental Results Using Two Cameras

The trajectory used for the experiment in the case of two cameras is the same represented in Fig. 10. The time history of the estimation errors is shown in Fig. 16. Noticeably, the accuracy of the system reaches the limit allowed by cameras calibration, for all the components of the motion, when the object does not move (about $5 \cdot 10^{-3}$ m for the position and about 1 deg for the orientation); during the motion the tracking errors grow but remain limited. As it was expected, the errors for the motion components are of the same order of magnitude, thanks to the use of a stereo camera system.

In Fig. 17 the output of the whole selection algorithm, for the two cameras, is reported. For each of the 40 feature points, two horizontal lines are considered: a point of the bottom line indicates that the feature point was classified as visible by the

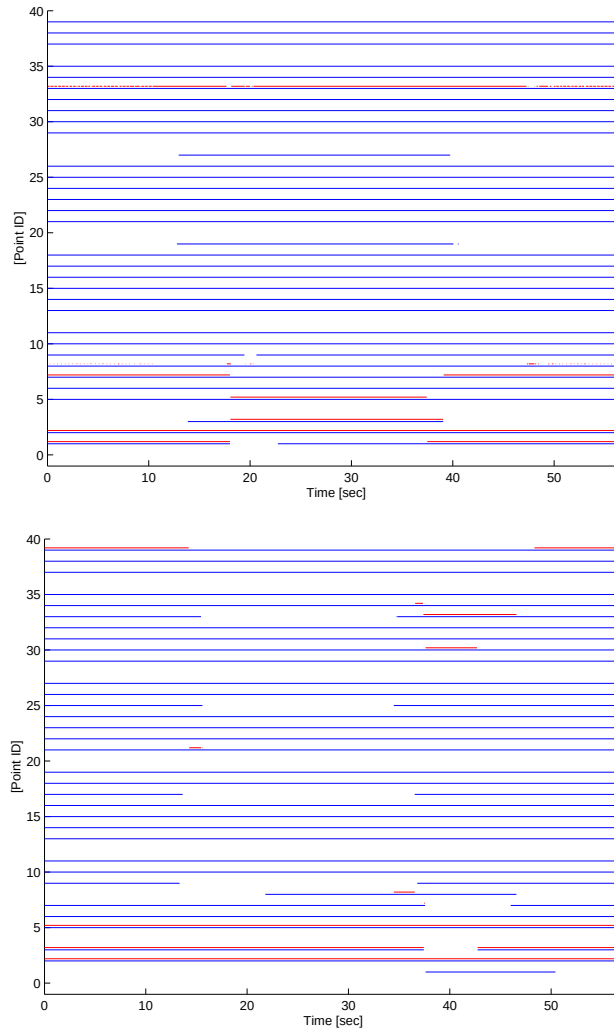


Fig. 17. Visible and selected points for Camera 1 (*top*) and Camera 2 (*bottom*), in the case of two cameras. For each point, the bottom line indicates when it is visible, the top line indicates when it is selected for feature extraction.

pre-selection algorithm at a particular sample time; a point of the top line indicates that the visible feature point was chosen by the selection algorithm. Notice that 8 feature points are selected at each sample time in order to guarantee at least five or six measurements in the case of fault of the extraction algorithm for some of the points. Remarkably, 4 feature points for camera are chosen at each sampling time, coherently with the almost symmetric disposition of the cameras with respect to the object.

8 Conclusion

The problem of real-time estimation of the pose (position and orientation) of a moving object from visual measurements has been considered in this chapter. A computationally efficient selection procedure has been presented, that allows evaluating the optimal set of feature points of the object to be used for image feature extraction and pose estimation. The procedure can be applied to polyhedral objects and is based on the representation of 3D objects by means of Binary Space Partitioning trees. The estimation technique fully exploits the noise rejection and the prediction capabilities of the EKF. Experimental results have been reported, which confirm the computational feasibility and the robustness of the presented visual tracking scheme for the case of two cameras.

The algorithm presented in this chapter may represent a good starting point to solve an important open issue for robotics applications: the visual tracking of objects in an unstructured and dynamic environment. A typical application may be the grasping of a moving object guided by a fixed visual system. In fact, for this scenario, the end effector may be considered as a second object of known pose. The proposed methodology may be used to develop a new strategy of automatic detection of the occlusions that happen during the grasp execution, which can be used to increase the task reliability. Similar problems may arise in cooperative robots applications.

Acknowledgement

This work has been co-funded by ASI.

References

1. K.S. Arun, T.S. Huang, and S.B. Blostein, "Least square fitting of two 3D point sets," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 9, pp. 698–700, 1997.
2. J. Baeten and J. De Schutter, "Hybrid vision/force control at corners in planar robotic-contour following," *IEEE/ASME Trans. on Mechatronics*, vol. 7, pp. 143–151, 2002.
3. J. Canny, "A computational approach to edge detection," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679–698, 1986.
4. T.W. Drummond and R. Cipolla, "Real-time tracking of complex structures with on-line camera calibration," *British Machine Vision Conf.*, pp. 574–583, 1999.
5. B. Espiau, F. Chaumette, and P. Rives, "A new approach to visual servoing in robotics," *IEEE Trans. on Robotics and Automation*, vol. 8, pp. 313–326, 1992.
6. J.T. Feddema, C.S.G. Lee, and O.R. Mitchell, "Weighted selection of image features for resolved rate visual feedback control," *IEEE Trans. on Robotics and Automation*, vol. 7, pp. 31–47, 1991.
7. A. Fox and S. Hutchinson, "Exploiting visual constraints in the synthesis of uncertainty-tolerant motion plans," *IEEE Trans. on Robotics and Automation*, vol. 11, pp. 56–71, 1995.
8. P. C. Ho and W. Wang, "Occlusion culling using minimum occluder set and opacity map," *Proc. of 1999 IEEE Int. Conf. on Information Visualization*, pp. 292–300, 1999.

9. B.K.P. Horn, K.M. Hilden, and S. Negahdaripour, "Closed-form solution of absolute orientation using orthonormal matrices," *J. of Optical Society of America*, vol. A-5, pp. 1127-1135, 1998.
10. R. Horaud, F. Dornaika, and B. Espiau, "Visually guided object grasping," *IEEE Trans. on Robotics and Automation*, vol. 14, pp. 525-532, 1998.
11. S. Hutchinson, G.D. Hager, and P.I. Corke, "A tutorial on visual servo control," *IEEE Trans. on Robotics and Automation*, vol. 12, pp. 651-670, 1996.
12. F. Janabi-Sharifi and W.J. Wilson, "Automatic selection of image features for visual servoing," *IEEE Trans. on Robotics and Automation*, vol. 13, pp. 890-903, 1997.
13. F. Janabi-Sharifi and W.J. Wilson, "Automatic grasp planning for visual-vervo controlled robotic manipulators," *IEEE Trans. on Systems, Man, and Cybernetics — Part B: Cybernetics*, vol. 28, pp. 693-711, 1998.
14. F. Keçeci and H.-H. Nagel, "Machine-vision-based estimation of pose and size parameters from a generic workpiece description," *Proc. of 2001 IEEE Int. Conf. on Robotics and Automation*, pp. 2159-2164, 2001.
15. S. Lee and Y. Kay, "An accurate estimation of 3-D position and orientation of a moving object for robot stereo vision: Kalman filter approach," *Proc. of 1990 IEEE Int. Conf. on Robotics and Automation*, pp. 414-419, 1990.
16. V. Lippiello, *Architetture, Algoritmi di Calibrazione e Tecniche di Stima dello Stato per un Sistema Asservito in Visione*, DIS, Univ. of Naples, Laurea Thesis, 2000.
17. V. Lippiello, B. Siciliano, and L. Villani, "Position and orientation estimation based on Kalman filtering of stereo images," *Proc. of 2001 IEEE Int. Conf. on Control Applications*, pp. 702-707, 2001.
18. V. Lippiello, B. Siciliano, and L. Villani, "Objects motion estimation via BSP tree modeling and Kalman filtering of stereo images," *Proc. of 2002 IEEE Int. Conf. on Robotics and Automation*, pp. 2968-2973, 2002.
19. E. Malis, F. Chaumette, and S. Boudet, "2 1/2 D visual servoing," *IEEE Trans. on Robotics and Automation*, vol. 15, pp. 234-246, 1999.
20. K. Nickels and S. Hutchinson, "Weighting observations: The use of kinematic models in object tracking," *Proc. of 1998 IEEE Int. Conf. on Robotics and Automation*, pp. 1677-1682, 1998.
21. J.N. Pan, Y.Q. Shi, and C.Q. Shu, "A Kalman filter in motion analysis from stereo image sequence," *Proc. of 1994 IEEE Int. Conf. on Image Processing*, pp. 63-67, 1994.
22. M. Paterson and F. Yao, "Efficient binary space partitions for hidden-surface removal and solid modeling," *Discrete and Computational Geometry*, vol. 5, pp. 485-503, 1990.
23. L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators*, 2nd Ed., Springer Verlag, 2000.
24. K. Tarabanis, R. Y. Tsai, and A. Kaul, "Computing occlusion-free viewpoints," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, pp. 279-292, 1996.
25. J. Wang and J.W. Wilson, "3D relative position and orientation estimation using Kalman filter for robot control," *Proc. of 1992 IEEE Int. Conf. on Robotics and Automation*, pp. 2638-2645, 1992.
26. J. Weng, P. Cohen, and M. Herniou, "Camera calibration with distortion models ad accuracy evaluation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 14, pp. 965-980, 1992.
27. J.W. Wilson, C.W. Hulls, and G. Bell, "Relative end-effector control using cartesian position based visual servoing," *IEEE Trans. on Robotics and Automation*, vol. 12, pp. 684-696, 1996.
28. J.S.-C. Yuan, "A general photogrammetric method for determining object position and orientation," *IEEE Trans. on Robotics and Automation*, vol. 5, pp. 129-142, 1999.

Appendix

The computation of the $(2mn \times 12)$ Jacobian matrix C_k in (14) gives

$$C_k = \begin{bmatrix} \frac{\partial \mathbf{g}}{\partial x_o} & \mathbf{0} & \frac{\partial \mathbf{g}}{\partial y_o} & \mathbf{0} & \frac{\partial \mathbf{g}}{\partial z_o} & \mathbf{0} & \frac{\partial \mathbf{g}}{\partial \varphi_o} & \mathbf{0} & \frac{\partial \mathbf{g}}{\partial \vartheta_o} & \mathbf{0} & \frac{\partial \mathbf{g}}{\partial \psi_o} & \mathbf{0} \end{bmatrix}_k \quad (15)$$

where $\mathbf{0}$ is a null $(2mn \times 1)$ vector corresponding to the partial derivatives of $\mathbf{g}$ with respect to the velocity variables, which are null because function $\mathbf{g}$ does not depend on the velocity.

Taking into account the expression of $\mathbf{g}$ in (8), the non-null elements of the Jacobian matrix (15) have the form:

$$\frac{\partial}{\partial \alpha} \begin{pmatrix} x_j^c \\ z_j^c \end{pmatrix} = \begin{pmatrix} \frac{\partial x_j^c}{\partial \alpha} z_j^c - x_j^c \frac{\partial z_j^c}{\partial \alpha} \end{pmatrix} (z_j^c)^{-2} \quad (16)$$

$$\frac{\partial}{\partial \alpha} \begin{pmatrix} y_j^c \\ z_j^c \end{pmatrix} = \begin{pmatrix} \frac{\partial y_j^c}{\partial \alpha} z_j^c - y_j^c \frac{\partial z_j^c}{\partial \alpha} \end{pmatrix} (z_j^c)^{-2} \quad (17)$$

where $\alpha = x_o, y_o, z_o, \varphi_o, \vartheta_o, \psi_o, i = 1, \dots, n, j = 1, \dots, m$.

The partial derivatives on the right-hand side of (16) and (17) can be computed as follows.

In view of (3), the partial derivatives with respect to the components of vector $\mathbf{o}_o = [x_o \ y_o \ z_o]^T$ are the elements of the Jacobian matrix

$$\frac{\partial \mathbf{p}_j^c}{\partial \mathbf{o}_o} = \mathbf{R}_c^T.$$

In order to express in compact form the partial derivatives with respect to the components of the vector $\phi_o = [\varphi_o \ \vartheta_o \ \psi_o]^T$, it is useful to consider the following equalities [23]

$$d\mathbf{R}_o(\phi_o) = \mathbf{S}(d\boldsymbol{\omega}_o)\mathbf{R}_o(\phi_o) = \mathbf{R}_o(\phi_o)\mathbf{S}(\mathbf{R}_o^T(\phi_o)d\boldsymbol{\omega}_o) \quad (18)$$

$$d\boldsymbol{\omega}_o = \mathbf{T}_o(\phi_o)d\phi_o \quad (19)$$

where $\mathbf{S}(\cdot)$ is the skew-symmetric matrix operator, $\boldsymbol{\omega}_o$ is the angular velocity of the object frame with respect to the base frame, and the matrices $\mathbf{R}_o$ and $\mathbf{T}_o$, in the case of Roll, Pitch, Yaw angles, have the form

$$\mathbf{R}_o(\phi_o) = \begin{bmatrix} c_{\varphi_o}c_{\vartheta_o} & c_{\varphi_o}s_{\vartheta_o}s_{\psi_o} - s_{\varphi_o}c_{\psi_o} & c_{\varphi_o}s_{\vartheta_o}c_{\psi_o} + s_{\varphi_o}s_{\psi_o} \\ s_{\varphi_o}c_{\vartheta_o} & s_{\varphi_o}s_{\vartheta_o}s_{\psi_o} + c_{\varphi_o}c_{\psi_o} & s_{\varphi_o}s_{\vartheta_o}c_{\psi_o} - c_{\varphi_o}s_{\psi_o} \\ -s_{\vartheta_o} & c_{\vartheta_o}s_{\psi_o} & c_{\vartheta_o}c_{\psi_o} \end{bmatrix}$$

$$\mathbf{T}_o(\phi_o) = \begin{bmatrix} 0 & -s_{\varphi_o} & c_{\varphi_o}c_{\vartheta_o} \\ 0 & c_{\varphi_o} & s_{\varphi_o}c_{\vartheta_o} \\ 1 & 0 & -s_{\vartheta_o} \end{bmatrix},$$

with $c_\alpha = \cos \alpha$ and $s_\alpha = \sin(\alpha)$. By virtue of (18), (19), and the properties of the skew-symmetric matrix operator, the following chain of equalities holds

$$\begin{aligned} d(\mathbf{R}_o(\phi_o)\mathbf{p}_j^o) &= d(\mathbf{R}_o(\phi_o))\mathbf{p}_j^o = \mathbf{R}_o(\phi_o)\mathbf{S}(\mathbf{R}_o^T(\phi_o)\mathbf{T}_o(\phi_o)d\phi_o)\mathbf{p}_j^o \\ &= \mathbf{R}_o(\phi_o)\mathbf{S}^T(\mathbf{p}_j^o)\mathbf{R}_o^T(\phi_o)\mathbf{T}_o(\phi_o)d\phi_o \\ &= \mathbf{S}^T(\mathbf{R}_o(\phi_o)\mathbf{p}_j^o)\mathbf{T}_o(\phi_o)d\phi_o, \end{aligned}$$

hence

$$\frac{\partial \mathbf{R}_o(\phi_o)}{\partial \phi_o} \mathbf{p}_j^o = \mathbf{S}^T(\mathbf{R}_o(\phi_o)\mathbf{p}_j^o)\mathbf{T}_o(\phi_o). \quad (20)$$

At this point, by virtue of (3) and (20), the following equality holds

$$\frac{\partial \mathbf{p}_j^c}{\partial \phi_o} = \mathbf{R}_c^T \frac{\partial \mathbf{R}_o(\phi_o)}{\partial \phi_o} \mathbf{p}_j^o = \mathbf{R}_c^T \mathbf{S}^T(\mathbf{R}_o(\phi_o)\mathbf{p}_j^o)\mathbf{T}_o(\phi_o).$$