Archives of Control Sciences
Volume 12(XLVIII), 2002
No. 1–2, *pages 71–88*

3-D objects motion estimation based on Kalman Filter and BSP Tree Models for Robot Stereo Vision

VINCENZO LIPPIELLO, BRUNO SICILIANO and LUIGI VILLANI

The problem of the real time estimation of the position and orientation of moving objects for position-based visual servoing control of robotic systems is considered in this paper. A computationally efficient algorithm is proposed based on Kalman filtering of the visual measurements of the position of suitable feature points selected on the target objects. The efficiency of the algorithm is improved by adopting a pre-selection technique of the feature points, based on Binary Space Partitioning (BSP) tree geometric models of the target objects, which takes advantage of the Kalman filter prediction capability. Computer simulations are presented to test the performance of the estimation algorithm in the presence of noise, different types of lens geometric distortion, quantization and calibration errors.

Key words: vision, robot manipulators, visual servoing, visual tracking, Kalman filter, Binary Space Partitioning tree

1. Introduction

The effectiveness and autonomy of a robotic system operating in unstructured environments can be significantly enhanced if a vision system based on one or more cameras is adopted to achieve direct measurements of the state of the environment and of the task in progress. Visual measurements can be directly used to perform closed-loop position control of the robot end-effector, usually denoted visual servoing control [1].

Visual servoing techniques can be grouped into two categories: Those performing position-based control and those performing image-based control. The control techniques belonging to the first category make use of a tracking error defined in the cartesian space [2]; those belonging to the second category make use of a tracking error defined in the image space and, differently from the position-based control, do not require calibration of the vision system [3].

One fundamental issue of position-based control is the real time estimation of the pose, i.e. the position and orientation trajectory, of known target objects. Typically, a

The authors are with PRISMA Lab. Dipartimento di Informatica e Sistemistica, Università degli Studi di Napoli Federico II, Via Claudio 21, 80125 Napoli, Italy. e-mails: vincenzo.lippiello,siciliano,lvillani@unina.it

This work was supported by Ministero della Ricerca Scientifica e Tecnologica and Agenzia Spaziale Italiana.

visual system based on one or more cameras is used to measure the position of suitable feature points selected on the target objects.

Visual measurements are usually affected by significant noise and disturbances due to temporal and spatial sampling and quantization of the image signal, lens distortion, etc., which may produce large errors. If a sequence of images is available, the extended Kalman filter can be effectively adopted to improve estimation accuracy [2, 4]. In fact, Kalman filtering offers many advantages over other estimation methods, e.g., temporal filtering, recursive implementation, possibility of realizing a proper statistical combination of redundant measurements, ability to change the measurement set during the operation. Also, the prediction capability of the filter allows setting up a dynamic windowing technique of the image plane which may sensibly reduce processing time.

It should be pointed out, however, that the algorithms based on Kalman filter usually require high computation time which increases with the number of feature points. The adoption of suitable algorithms for the selection of an optimal subset of feature points at each sample time may be useful to reduce the computation burden but may be inadequate in case of target objects with a large number of feature points [5, 6].

In this paper, the extended Kalman filter is adopted for the real time estimation of the pose of moving objects using a sequence of images captured by a stereo camera system. The systematic procedure presented in [7] is used for the case of n video cameras fixed in the workspace. Thanks to a suitable definition of the output equations of the Kalman filter, the algorithm provides the estimate directly in the base frame, without requiring any additional frame transformation.

The estimation algorithm is improved in this work by adopting an efficient method of pre-selection of the feature points based on Binary Space Partitioning (BSP) trees for representing the geometry of the target objects [8]. In particular, the prediction of the objects pose provided by the Kalman filter is used to reconstruct a 2D model of the visible parts of the objects on the image plane of each camera. This allows to identify all the feature points visible at the next sample time, which can be input to a standard algorithm for optimal points selection and/or to a dynamic windowing algorithm. The proposed pre-selection algorithm can be used also in the case of objects and obstacles with interposing parts. Differently from other algorithms (see [6] and references therein), this method allows to recognize 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.

The effectiveness of the proposed approach is tested in a simulation case study. In order to reproduce a real situation, the effects of spatial quantization and amplitude discretization as well as calibration error and imperfect lens distortion compensation are rigorously taken into account in the model adopted for simulation.

2. Modelling

Consider a system of n video cameras fixed with respect to a base coordinate frame O–xyz. The geometry of the system can be characterized using the classical pinhole

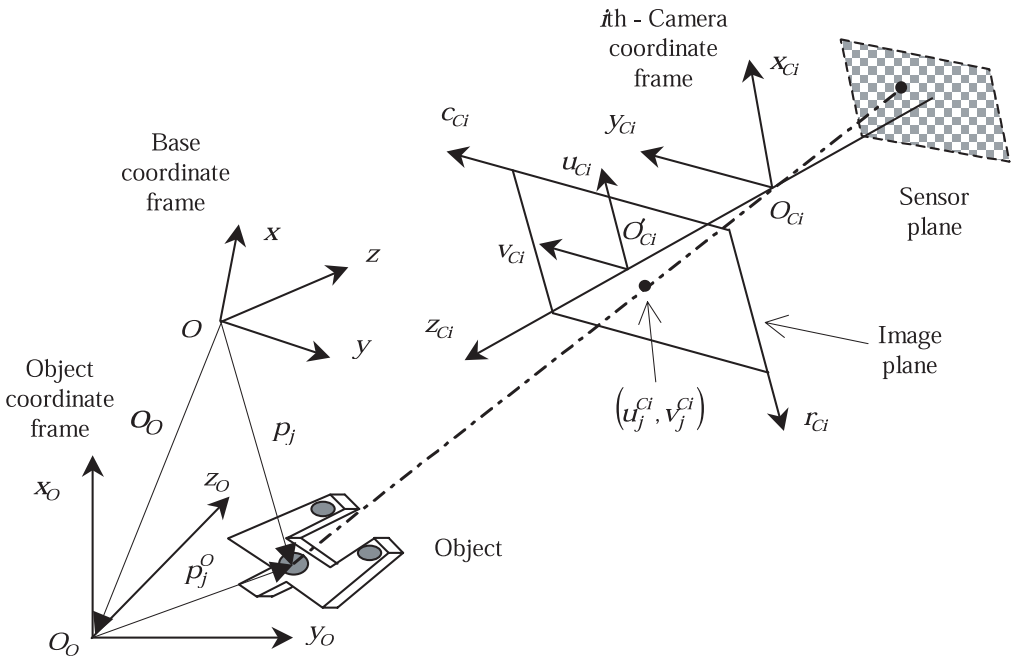

Figure 1. Reference frames for the i-th camera and the object using the pinhole model.

model, described in Fig. 1. A frame O_{ci}–$x_{ci}y_{ci}z_{ci}$ attached to the i-th camera (camera frame), with the z_{ci}-axis aligned to the optical axis and the origin in the optical center, is considered for each camera. In the following, a superscript will be used to denote the reference frame of a variable, when different from the base frame.

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 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 $\boldsymbol{p}^{ci} = [x^{ci} \quad y^{ci} \quad z^{ci}]^{\mathrm{T}}$ in the camera frame is projected onto the point of the image plane whose coordinates can be computed with the equation

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

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

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

being $[r_0^{ci} \quad c_0^{ci}]^{\mathrm{T}}$ the coordinates of the point O_{ci} whereas s_u^{ci} and s_v^{ci} are the row and column scaling factor, respectively.

Without loss of generality, the case of a single moving target object is considered. The position and orientation of a frame attached to the object O_o–$x_o y_o z_o$ with respect to the base frame can be expressed in terms of the coordinate vector of the origin $\boldsymbol{o}_o = [x_o \quad y_o \quad z_o]^{\mathrm{T}}$ and of the rotation matrix $\boldsymbol{R}_o(\boldsymbol{\phi}_o)$, where $\boldsymbol{\phi}_o = [\varphi_o \quad \vartheta_o \quad \psi_o]^{\mathrm{T}}$ is the vector of the Roll, Pitch and Yaw angles. The components of the vectors $\boldsymbol{o}_o$ and $\boldsymbol{\phi}_o$ are the six unknown quantities to be estimated.

Consider m feature points of the object. The coordinate vector $\boldsymbol{p}_j$ of the feature point P_j ($j = 1, \ldots, m$) can be expressed in the base frame as

$$\boldsymbol{p}_j = \boldsymbol{o}_o + \boldsymbol{R}_o(\boldsymbol{\phi}_o)\boldsymbol{p}_j^o, \tag{3}$$

where $\boldsymbol{p}_j^o$ is the coordinate vector of P_j expressed in the object frame. The constant vector $\boldsymbol{p}_j^o$ is assumed to be known, and can be computed from a CAD model of the object or via a suitable calibration procedure. Analogously, the coordinate vector of P_j can be expressed in the i-th camera frame ($i = 1, \ldots, n$) as

$$\boldsymbol{p}_j^{ci} = \boldsymbol{R}_{ci}^{\mathrm{T}}(\boldsymbol{p}_j - \boldsymbol{o}_{ci}), \tag{4}$$

where $\boldsymbol{o}_{ci}$ and $\boldsymbol{R}_{ci}$ are, respectively, the position vector and the rotation matrix of the i-th camera frame referred to the base frame. The quantities $\boldsymbol{o}_{ci}$ and $\boldsymbol{R}_{ci}$ are constant because the cameras are assumed to be fixed to the workspace, and can be computed through a suitable calibration procedure [9].

By folding equation (3) into (4), the following mn equations are obtained

$$\boldsymbol{p}_j^{ci} = \boldsymbol{R}_{ci}^{\mathrm{T}}(\boldsymbol{o}_o - \boldsymbol{o}_{ci} + \boldsymbol{R}_o(\boldsymbol{\phi}_o)\boldsymbol{p}_j^o), \tag{5}$$

that can be replaced into the perspective transformation (1) and into equation (2). Therefore, a system of $2mn$ nonlinear equations is achieved, which depend on the measurements of the m feature points in the image plane of the n cameras, whereas the six components of the vectors $\boldsymbol{o}_o$ and $\boldsymbol{\phi}_o$, expressed in the base frame, are the unknown variables. To solve these equation 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. Extended 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

$$\boldsymbol{w} = [x_o \quad \dot{x}_o \quad y_o \quad \dot{y}_o \quad z_o \quad \dot{z}_o \quad \varphi_o \quad \dot{\varphi}_o \quad \vartheta_o \quad \dot{\vartheta}_o \quad \psi_o \quad \dot{\psi}_o]^{\mathrm{T}}. \tag{6}$$

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 $\boldsymbol{\gamma}$ described by zero mean Gaussian noise with covariance given by the (12×12) matrix $\boldsymbol{Q}$. The discrete time dynamic model can be written as

$$\boldsymbol{w}_k = \boldsymbol{A}\boldsymbol{w}_{k-1} + \boldsymbol{\gamma}_k \tag{7}$$

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

$$\boldsymbol{A} = \operatorname{diag}\left\{ \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}, \cdots, \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \right\}.$$

The output of the Kalman filter is the vector of the *normalized* coordinates of the m feature points in the image plane of the n cameras

$$\boldsymbol{\zeta}_k = \left[\frac{u_1^{c1}}{f_e^{c1}} \;\; \frac{v_1^{c1}}{f_e^{c1}} \;\; \cdots \;\; \frac{u_1^{cn}}{f_e^{cn}} \;\; \frac{v_1^{cn}}{f_e^{cn}} \;\; \cdots \;\; \frac{u_m^{c1}}{f_e^{c1}} \;\; \frac{v_m^{c1}}{f_e^{c1}} \;\; \cdots \;\; \frac{u_m^{cn}}{f_e^{cn}} \;\; \frac{v_m^{cn}}{f_e^{cn}} \right]_k^{\mathrm{T}}. \tag{8}$$

By taking into account the equation (1), the corresponding output model can be written in the form

$$\boldsymbol{\zeta}_k = \boldsymbol{g}(\boldsymbol{w}_k) + \boldsymbol{\nu}_k \tag{9}$$

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

$$\boldsymbol{g}(\boldsymbol{w}_k) = \left[\frac{x_1^{c1}}{z_1^{c1}} \;\; \frac{y_1^{c1}}{z_1^{c1}} \;\; \cdots \;\; \frac{x_1^{cn}}{z_1^{cn}} \;\; \frac{y_1^{cn}}{z_1^{cn}} \;\; \cdots \;\; \frac{x_m^{c1}}{z_m^{c1}} \;\; \frac{y_m^{c1}}{z_m^{c1}} \;\; \cdots \;\; \frac{x_m^{cn}}{z_m^{cn}} \;\; \frac{y_m^{cn}}{z_m^{cn}} \right]_k^{\mathrm{T}}. \tag{10}$$

The coordinates of the feature points $\boldsymbol{p}_j^{ci}$ in equation (10) are computed from the state vector $\boldsymbol{w}_k$ via equation (5). Matrix $\boldsymbol{R}$ 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, which is an approximation of the true Kalman filter algorithm.

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

$$\hat{\boldsymbol{w}}_{k,k-1} = \boldsymbol{A}\hat{\boldsymbol{w}}_{k-1,k-1} \tag{11}$$

$$\boldsymbol{P}_{k,k-1} = \boldsymbol{A}\boldsymbol{P}_{k-1,k-1}\boldsymbol{A}^{\mathrm{T}} + \boldsymbol{Q}_{k-1}, \tag{12}$$

where $\boldsymbol{P}_{k,k-1}$ 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

$$\hat{\boldsymbol{w}}_{k,k} = \hat{\boldsymbol{w}}_{k,k-1} + \boldsymbol{K}_k(\boldsymbol{\zeta}_k - \boldsymbol{g}(\hat{\boldsymbol{w}}_{k,k-1})) \tag{13}$$

$$\boldsymbol{P}_{k,k} = \boldsymbol{P}_{k,k-1} - \boldsymbol{K}_k \boldsymbol{C}_k \boldsymbol{P}_{k,k-1}, \tag{14}$$

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

$$\boldsymbol{K}_k = \boldsymbol{P}_{k,k-1} \boldsymbol{C}_k^{\mathrm{T}} (\boldsymbol{R}_k + \boldsymbol{C}_k \boldsymbol{P}_{k,k-1} \boldsymbol{C}_k^{\mathrm{T}})^{-1}, \tag{15}$$

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

$$\boldsymbol{C}_k = \left. \frac{\partial \boldsymbol{g}(\boldsymbol{w})}{\partial \boldsymbol{w}} \right|_{\boldsymbol{w}=\hat{\boldsymbol{w}}_{k,k-1}}. \tag{16}$$

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

Notice that, in the case of multiple target objects, a Kalman filter for each object has to be considered.

4. Feature points selection

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 feature points between four and six, if properly chosen, may represent a good trade-off [4]. Automatic selection algorithms have been developed to find the optimal feature points [6]. 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 paper is based on BSP trees.

4.1. BSP tree geometric modelling

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 a set of objects as reported in [8].

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 as a set of planar polygons.

Each polygon 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* is chosen, characterized by a vector normal to the plane; the node is defined as the set containing the partition plane and all the polygons on it.

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

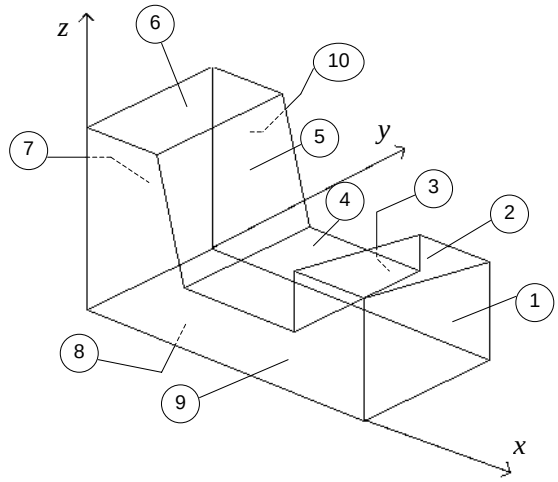

Figure 2. Object and corresponding polygons

If a polygon happens to span 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.

The choice of the partition planes depends on how the tree will be used. For the purpose of removing the hidden surfaces, it is appropriate to choose the partition planes from the initial set of polygons.

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. A partition plane is represented by the vector $\boldsymbol{\pi} = [a \quad b \quad c \quad d]^T$ of the coefficients of the equation of the plane with respect to a base reference frame

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

where $\boldsymbol{n} = [a \quad b \quad c]^T$ is the unit vector normal to the plane. The root of the tree contains the polygon number 10, which is on the first partition plane; 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 span the partition plane, were already added to previous nodes of the tree), hence they have been split into two pieces each (see polygons number 5f, 5b, 7f, 7b in Fig. 4).

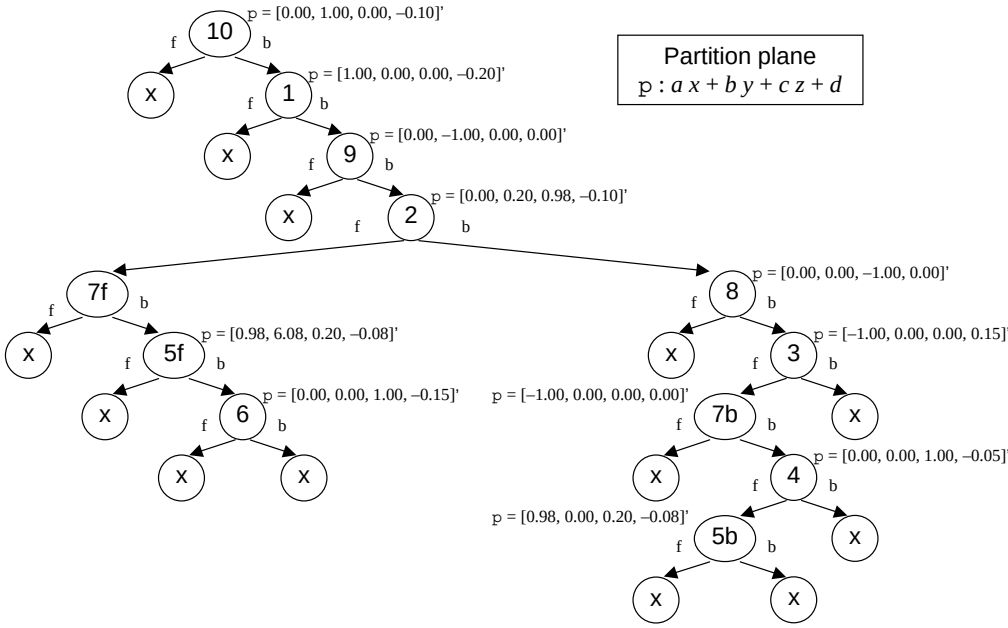

Figure 3. BSP tree of the object.

4.2. Pre-selection algorithm

Once that a BSP tree representation of an object is available, it is possible to select the feature points of the object that can be 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, 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 visible feature points is updated by adding all the feature points of the polygons of the current node and eliminating all the feature points of the set that are hidden by the polygons of the current node.

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

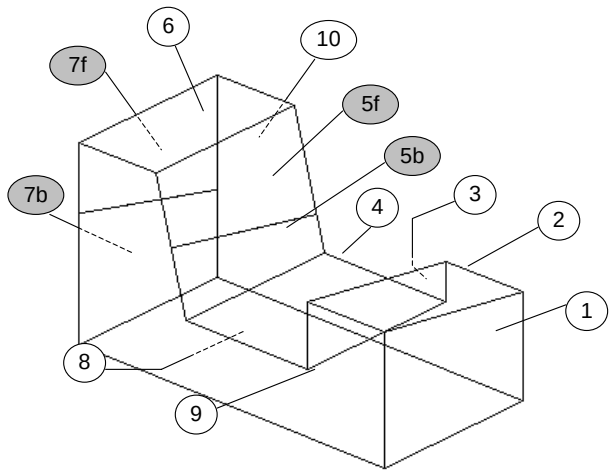

Figure 4. Case of partition plane splitting polygons number 5 and number 7

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

The technique described above can be suitably exploited to set up a real time pre-selection algorithm of the feature points to be localized on the cameras image planes, using the prediction of the estimated pose of the target objects provided by the Kalman filters.

5. Estimation procedure

In order to set up a procedure for the estimation of the pose of one or more target objects, two different situations must be considered: The case of objects whose parts cannot be interposed, and the case of interposing objects (e.g., a gripper grasping an object).

In Fig. 5 a functional chart of the estimation algorithm is represented, that can be used in the case of a single target object or multiple target objects whose parts cannot be interposed. It is assumed that a BSP tree representation of each object is built off-line from the CAD model. A Kalman filter is used for each object to estimate the corresponding pose with respect to the base frame at the next sample time. The feature points selection and windows placing operation, for each camera, can be detailed as follows:

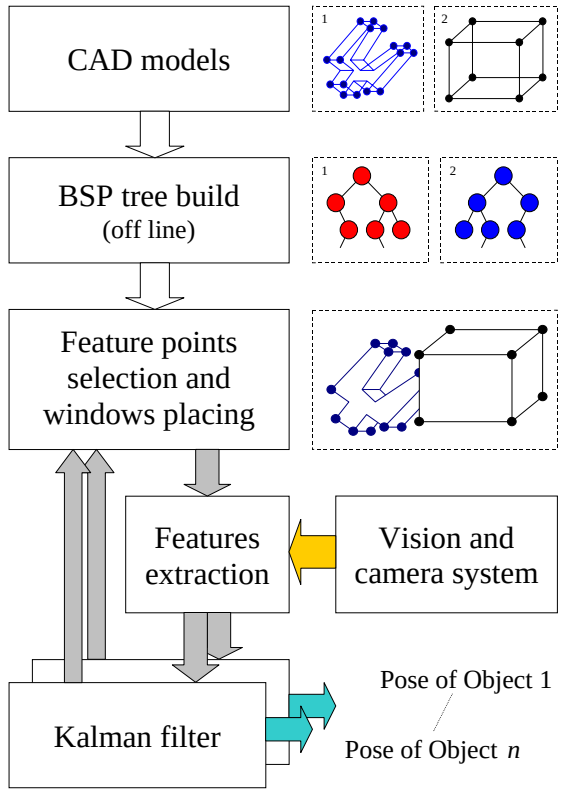

Figure 5. Functional chart of the estimation procedure for the case of non interposing objects.

- **Step 1:** According to the estimated distance of the objects from the camera, the BSP tree representations of the objects are put in an ordered sequence, from the farthest to the nearest.
- **Step 2:** The visit algorithm described in the previous Section is applied to each BSP tree of the sequence to find the set of all the feature points that are visible from the camera. For each object, a current set of visible points is updated, by adding all the visible feature points of the current object and eliminating all the feature points of the previous objects of the sequence hidden by the current object.
- **Step 3:** The resulting set of visible points is input to a standard algorithm for the selection of the optimal feature points.
- **Step 4:** The location of the optimal feature points in the image plane at the next sample time is computed on the basis of the objects pose estimation provided by the Kalman filter.

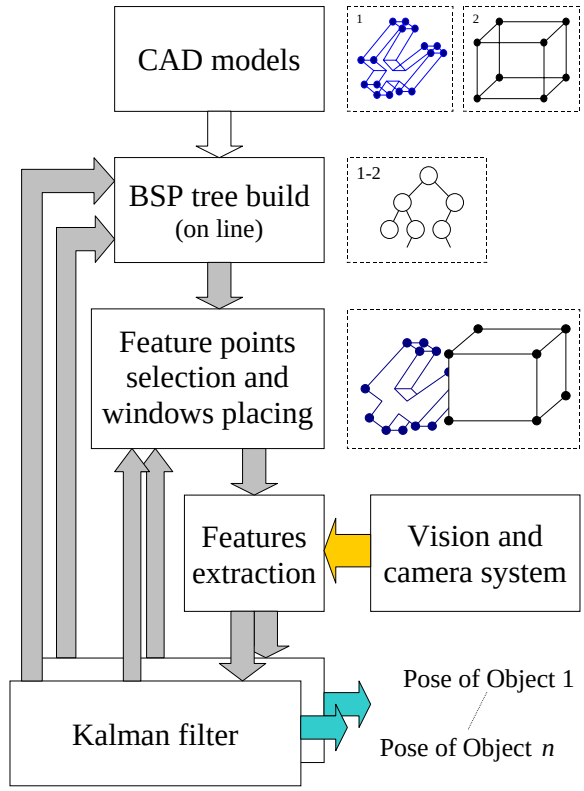

Figure 6. Functional chart of the estimation procedure for the case of interposing objects.

- **Step 5:** A dynamic windowing algorithm is executed to select the parts of the image plane to be input to the feature extraction algorithm.

In the case of multiple target objects whose parts can be interposed, this procedure may fail because the objects cannot be correctly ordered with respect to the distance from the camera. This problem can be overcome at expense of the computation time by adopting a different solution represented in the functional chart of Fig. 6.

As before, a Kalman filter is used for each object to estimate the corresponding pose with respect to the base frame at the next sample time. Differently form the previous case, a unique BSP tree representation of all the objects is built on-line, using the CAD models of the objects and the estimation provided by the Kalman filters. Hence, for each camera, the visit algorithm of the tree is executed once to find the set of all the visible points. Then, the Steps 3, 4, and 5 are executed.

Notice that the procedures described above can be extended to the case of objects moving among obstacles of known geometry; if the obstacles are moving with respect

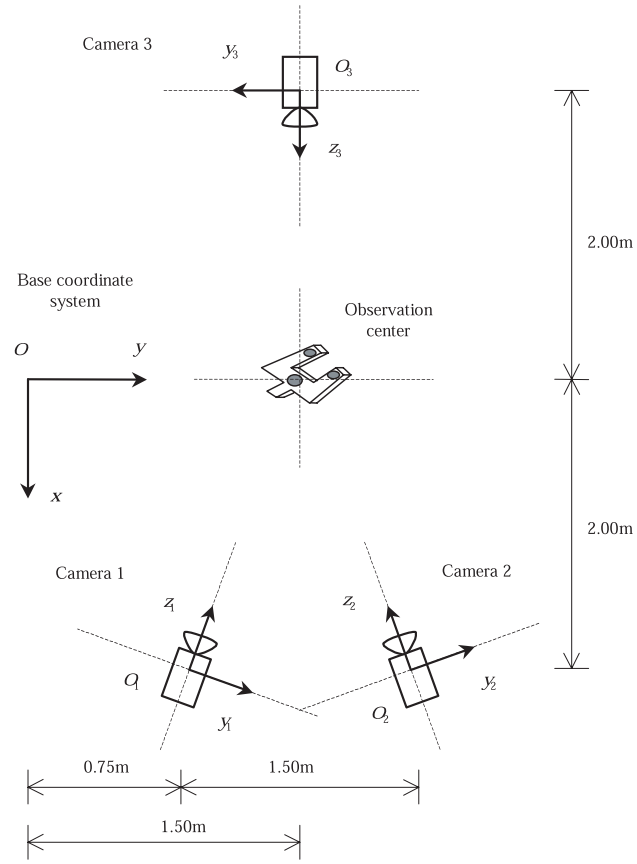

Figure 7. Position and orientation of the three cameras with respect to the base frame.

to the base frame, the corresponding motion variables can be estimated using Kalman filters.

6. Simulation

The effectiveness of the estimation algorithm has been tested in simulation case studies for a stereo vision system composed by three cameras with a 50 Hz sampling rate. The cameras are fixed with the workspace and their position and orientation with respect to the base frame is presented in Fig. 7. The optical axes of the three cameras are on a plane parallel to the xy-plane of the base frame at a distance $z = 1.5$ m. The effective focal length of the three cameras is 16 mm, and the pixel dimensions are 8.3×8.3 μm. To reproduce the geometric distortion effects of real cameras, a polyno-

mial model including radial distortion, decentering distortion and thin prism distortion has been adopted [9, 10]. The numerical values of the distortion parameters have been chosen so as to achieve errors up to 1.4% of the sensor half dimension. The same model has been considered in the Kalman filter algorithm for distortion compensation, and the corresponding parameters have been chosen so as to achieve errors ranging from 1% up to 56%, reproducing a typical situation of partial distortion compensation. Moreover, errors up to 3.8% have been introduced in the parameters of the geometric model of the stereo vision system in order to reflect a typical situation of imperfect system calibration.

A target object characterized by 16 feature points, corresponding to the object corners, is considered (see Fig. 2). The selection procedure is used to find four optimal feature points at each sample time.

The object moves according to a trajectory of sinusoidal type of 8 s duration and amplitude 200 mm, 150 mm, 100 mm, respectively, for the x, y and z component of the position, and 30 deg, 25 deg, 45 deg, respectively, for the Roll, Pitch and Yaw angles.

White independent Gaussian noise is added to the true projections on the cameras image plane of the feature points. In order to simulate spatial sampling and quantization errors, the variance of the noise has been chosen as $d^2/12$ where $d = 8.3$ μm is the pixel dimension.

The output measurement covariance matrix has been set to $\boldsymbol{R} = 0.029\boldsymbol{I}_{24}$, where $\boldsymbol{I}_n$ is the $(n \times n)$ identity matrix. The numerical value has been computed on the basis of the statistics of the calibration errors imposed on the camera calibration parameters. The disturbance noise covariance has been selected so as to give a rough representation of the errors due to the simplification introduced on the model (constant velocity), by considering only velocity disturbance, i.e.

$$\boldsymbol{Q} = \mathrm{diag}\{0, 10, 0, 10, 0, 10, 0, .2, 0, .2, 0, .2\}$$

where the units are (mm/s)^2 for translational velocities and (rad/s)^2 for rotational velocities.

The corresponding time histories of the components of the position and orientation estimation error are reported in Fig. 8.

It can be observed that the estimation error is in the range ± 2 mm for the x component whereas is in the range of ± 1 mm for the y and z components.

Analogously, the estimation error is in the range of $\pm 0.5^{\mathrm{o}}$ for the Yaw angle, whereas is in the range $\pm 1^{\mathrm{o}}$ for the Roll and Pitch angles. The reason of this asymmetric behaviour is due to the asymmetric geometry chosen for the cameras locations. It is worth noticing that a residual error remains also when the object stops, especially on the orientation components, due to the calibration error, the distortion and the measurement noise.

In Fig. 9 a sequence of 12 different projections of the target object on the image plane of camera 1 is reported, as reconstructed from the positions of the feature points selected by the proposed algorithm. It can be seen that the pre-selection algorithm allows to correctly predict the positions of the feature points (object corners) that are visible from the camera at the next sample time.

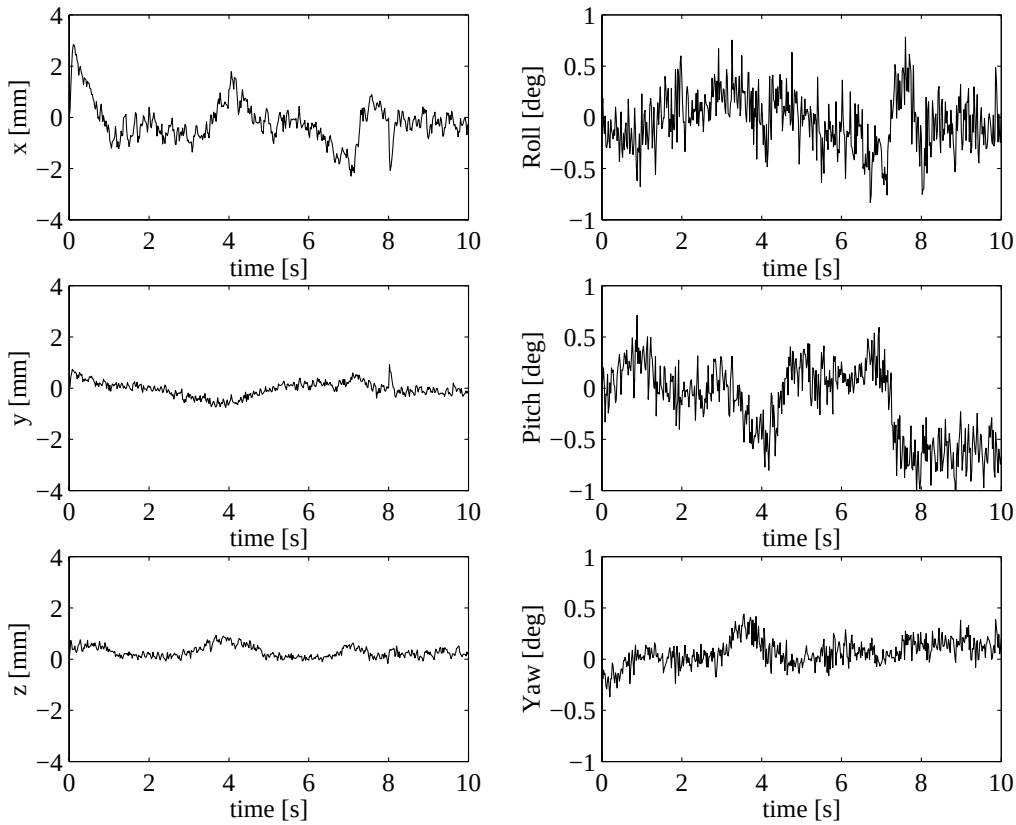

Figure 8. Position error (left) and orientation error (right).

In order to test the effectiveness of the proposed approach in the case of multiple target objects, a second object moving according to a sinusoidal trajectory has been added to the previous one. The time histories of the position and orientation errors are not reported for brevity. In Fig. 10 the projections of the two target objects on the image plane of camera 1 are represented, as reconstructed using the Kalman filter estimates and the pre-selection algorithm. It can be seen that the visible feature points are correctly identified also when the objects are superimposed.

7. Conclusion

The estimation of the pose (position and orientation)of moving objects from visual measurements was considered in this paper. The extended Kalman filter was used to recursively compute an estimate of the motion variables from the measurements of the

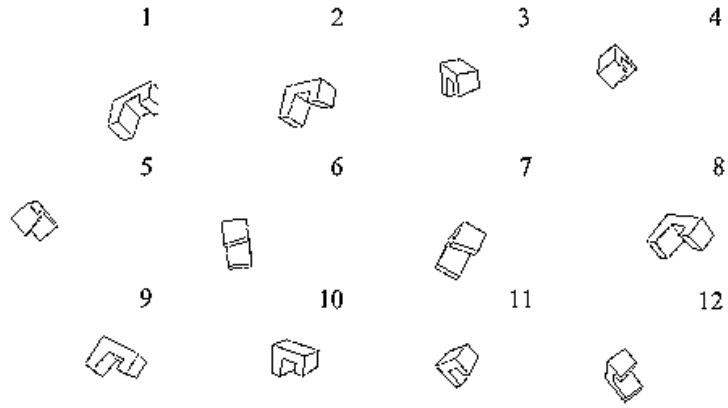

Figure 9. Sequence of projections of the target object on the image plane of camera 1

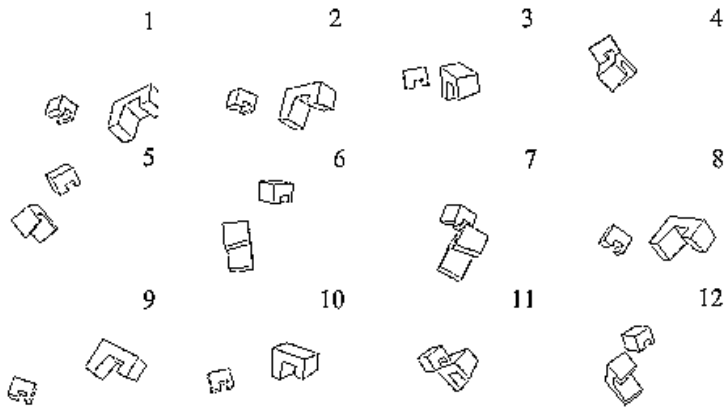

Figure 10. Sequence of projections of two target objects on the image plane of camera 1

position of suitable feature points of the objects. The efficiency of the algorithm was improved by adopting a technique of pre-selection of the visible feature points at each sample time based on a BSP tree representation of the objects geometry. The proposed algorithm can be used also to the case of target objects and obstacles with interposing parts. Computer simulations have shown the effectiveness of the algorithm both in the case of a single object and in the case of superimposing objects.

References

[1] S. HUTCHINSON, G.D. HAGER and P.I. CORKE: A tutorial on visual servo control. *IEEE Trans. on Robotics and Automation*, **12** (1996), 651-670.

[2] J.W. WILSON: Relative end-effector control using cartesian position based visual servoing. *IEEE Trans. on Robotics and Automation*, **12** (1996), 684-696.

[3] B. ESPIAU, F. CHAUMETTE and P. RIVES: A new approach to visual servoing in robotics. *IEEE Trans. on Robotics and Automation*, **8** (1992), 313-326.

[4] J. WANG and J.W. WILSON: 3D relative position and orientation estimation using Kalman filter for robot control. *IEEE Int. Conf. on Robotics and Automation*, (1992), 2638-2645.

[5] 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*, **7** (1991), 31-47.

[6] F. JANABI-SHARIFI and W.J. WILSON: Automatic selection of image features for visual servoing. *IEEE Trans. on Robotics and Automation*, **13** (1997), 890-903.

[7] V. LIPPIELLO, B. SICILIANO and L. VILLANI: Position and orientation estimation based on Kalman filtering of stereo images. *IEEE Int. Conf. on Control Applications*, Mexico City, MEX, (2001), 702-707.

[8] M. PATERSON and F. YAO: Efficient binary space partitions for hidden-surface removal and solid modeling. *Discrete and Computational Geometry*, **5** (1990), 485-503.

[9] J. WENG, P. COHEN and M. HERNIOU: Camera calibration with distortion models and accuracy evaluation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **14** (1992), 965-980.

[10] V. LIPPIELLO: Architetture, algoritmi di calibrazione e tecniche di stima dello stato per un sistema asservito in visione. *Laurea thesis*, (in Italian), (2000).

[11] L. SCIAVICCO and B. SICILIANO: Modelling and Control of Robot Manipulators. 2nd Ed., Springer-Verlag, London, 2000.

Appendix

The computation of the $(2mn \times 12)$ Jacobian matrix $\boldsymbol{C}_k$ in (16) gives

$$\boldsymbol{C}_k = \left[\frac{\partial \boldsymbol{g}}{\partial x_o} \; \mathbf{0} \; \frac{\partial \boldsymbol{g}}{\partial y_o} \; \mathbf{0} \; \frac{\partial \boldsymbol{g}}{\partial z_o} \; \mathbf{0} \; \frac{\partial \boldsymbol{g}}{\partial \varphi_o} \; \mathbf{0} \; \frac{\partial \boldsymbol{g}}{\partial \vartheta_o} \; \mathbf{0} \; \frac{\partial \boldsymbol{g}}{\partial \psi_o} \; \mathbf{0} \right]_k \tag{17}$$

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

Taking into account the expression of $\boldsymbol{g}$ in (10), the non-null elements of the Jacobian matrix (17) have the form:

$$\frac{\partial}{\partial \alpha}\left(\frac{x_j^{ci}}{z_j^{ci}}\right) = \left(\frac{\partial x_j^{ci}}{\partial \alpha} z_j^{ci} - x_j^{ci}\frac{\partial z_j^{ci}}{\partial \alpha}\right)(z_j^{ci})^{-2} \tag{18}$$

$$\frac{\partial}{\partial \alpha}\left(\frac{y_j^{ci}}{z_j^{ci}}\right) = \left(\frac{\partial y_j^{ci}}{\partial \alpha} z_j^{ci} - y_j^{ci}\frac{\partial z_j^{ci}}{\partial \alpha}\right)(z_j^{ci})^{-2} \tag{19}$$

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

The partial derivatives on the right-hand side of (18) and (19) can be computed as follows.

In view of (5), the partial derivatives with respect to the components of vector $\boldsymbol{o}_o = [x_o \quad y_o \quad z_o]^{\mathrm{T}}$ are the elements of the Jacobian matrix

$$\frac{\partial \boldsymbol{p}_j^{ci}}{\partial \boldsymbol{o}_o} = \boldsymbol{R}_{ci}^T.$$

In order to express in compact form the partial derivatives with respect to the components of the vector $\boldsymbol{\phi}_o = [\varphi_o \quad \vartheta_o \quad \psi_o]^{\mathrm{T}}$, it is useful considering the following equalities [11]

$$\mathrm{d}\boldsymbol{R}_o(\boldsymbol{\phi}_o) = \boldsymbol{S}(\mathrm{d}\boldsymbol{\omega}_o)\boldsymbol{R}_o(\boldsymbol{\phi}_o) = \boldsymbol{R}_o(\boldsymbol{\phi}_o)\boldsymbol{S}(\boldsymbol{R}_o^T(\boldsymbol{\phi}_o)\mathrm{d}\boldsymbol{\omega}_o) \tag{20}$$

$$d\boldsymbol{\omega}_o = \boldsymbol{T}_o(\boldsymbol{\phi}_o)\mathrm{d}\boldsymbol{\phi}_o \tag{21}$$

where $\boldsymbol{S}(\cdot)$ 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 $\boldsymbol{R}_o$ and $\boldsymbol{T}_o$, in the case of Roll, Pitch and Yaw angles, have the form

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

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

with $c_\alpha = \cos\alpha$ and $s_\alpha = \sin(\alpha)$. By virtue of (20), (21), and the properties of the skew-symmetric matrix operator, the following chain of equalities holds

$$\mathrm{d}(\boldsymbol{R}_o(\boldsymbol{\phi}_o)\boldsymbol{p}_j^o) = \mathrm{d}(\boldsymbol{R}_o(\boldsymbol{\phi}_o))\boldsymbol{p}_j^o = \boldsymbol{R}_o(\boldsymbol{\phi}_o)\boldsymbol{S}(\boldsymbol{R}_o^T(\boldsymbol{\phi}_o)\boldsymbol{T}_o(\boldsymbol{\phi}_o)\mathrm{d}\boldsymbol{\phi}_o)\boldsymbol{p}_j^o$$

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

hence

$$
\frac{\partial \boldsymbol{R}_o(\boldsymbol{\phi}_o)}{\partial \boldsymbol{\phi}_o}\boldsymbol{p}_j^o = \boldsymbol{S}^T(\boldsymbol{R}_o(\boldsymbol{\phi}_o)\boldsymbol{p}_j^o)\boldsymbol{T}_o(\boldsymbol{\phi}_o). \tag{22}
$$

At this point, by virtue of (5) and (22), the following equality holds

$$
\frac{\partial \boldsymbol{p}_j^{ci}}{\partial \boldsymbol{\phi}_o} = \boldsymbol{R}_{ci}^T\frac{\partial \boldsymbol{R}_o(\boldsymbol{\phi}_o)}{\partial \boldsymbol{\phi}_o}\boldsymbol{p}_j^o = \boldsymbol{R}_{ci}^T\boldsymbol{S}^T(\boldsymbol{R}_o(\boldsymbol{\phi}_o)\boldsymbol{p}_j^o)\boldsymbol{T}_o(\boldsymbol{\phi}_o).
$$