Robotics and Computer–Integrated Manufacturing 81 (2023) 102519

Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim

Full length article

Assembly task execution using visual 3D surface reconstruction: An integrated approach to parts mating

M. Nigro [a], M. Sileo [a], F. Pierri [a,*], D.D. Bloisi [b], F. Caccavale [a]

[a] School of Engineering, University of Basilicata, Viale dell'Ateneo Lucano, 10, Potenza, 85100, Italy

[b] Department of Mathematics, Computer Science, and Economics, University of Basilicata, Viale dell'Ateneo Lucano, 10, Potenza, 85100, Italy

ARTICLE INFO

Keywords:
Robot manipulators
Assembly tasks
Parts mating

ABSTRACT

This paper is focused on assembly tasks executed by an industrial robotic manipulator in the presence of uncertainties. The goal is to achieve higher levels of autonomy and flexibility of robotic systems in the execution of such tasks. In particular, as a well-established paradigm of assembly tasks, a Peg-in-Hole task has been considered, where the pose of the target object with respect to the robot is known with uncertainties far larger than the task tolerance, e.g., due to manual positioning of the object in the workcell. The proposed approach is based on the reconstruction of the object surface by means of a number of point clouds provided by a depth sensor. The reconstruction is then compared with a known CAD model of the surface, in order to localize the position and tilt of the holes. Finally, the peg insertion is performed in two steps: a search phase, in which the peg tip gently slides on the surface following a trajectory described by Lissajous functions, and a mechanical coupling phase, in which a compliant behavior is imposed to the peg. Experiments on a collaborative manipulator confirm that the proposed approach allows to achieve a better degree of autonomy and flexibility for a class of robotic tasks in partially structured environments.

1. Introduction

Current trends in manufacturing require the adoption of more autonomous and effective robots, capable of quick adaption to changes of the production mix, while ensuring proper levels of safety and reliability. Such a challenge can be tackled by developing advanced approaches to task planning, control, and execution.

In this paper, the focus is on the execution of complex tasks in partially structured environments, by exploiting the availability of affordable and reliable sensing devices. In particular, a class of assembly tasks is considered, involving insertion and/or bonding of mechanical parts, where the goal is to achieve autonomous task execution by resorting to visual sensing. As test case, highly representative of the considered class of assembly task, a Peg-in-Hole operation is considered.

1.1. State of the art

Although Peg-in-Hole tasks have been extensively considered over the last decades, they are still challenging, due to required accuracy in the determination of the workpiece pose and the robot positioning. The commonly adopted approach includes two phases:

1. The search phase, aimed at localizing the hole and aligning the peg to its axis.
2. The insertion phase, where the peg is coupled with the assigned hole.

Successful insertion can be attained via pure positional control only in the case of perfect relative localization of the hole with respect to the peg. However, this can be rarely guaranteed in real application scenarios, due to imperfect knowledge of the hole position and robot positioning errors. This means that pure position control can be adopted only in the presence of generous clearances, while active or passive interaction control approaches need to be adopted to ensure task completion. Among interaction control schemes, the impedance control paradigm [1] has been extensively utilized for Peg-in-Hole, e.g., in [2–4], for a single-arm robots, or in [5,6] for multi-arm systems.

Regarding the search phase, the existing approaches can be roughly classified into those based on visual sensor feedback and those based on exploration of the hole neighborhood. Among the first category, in [7] a high-speed camera is adopted to align the peg to the hole, while, more recently, [8] and [9] propose the use of visual coaxial systems. Visual approaches can lead to satisfactory performance, as e.g., in [10], where a micro-peg-in-hole tasks is considered, and in [11], where an automated optical system is used to detect and classify aesthetical defects after the casting process. However, visual methods can

* Corresponding author.
E-mail address: francesco.pierri@unibas.it (F. Pierri).

https://doi.org/10.1016/j.rcim.2022.102519
Received 6 August 2022; Received in revised form 16 November 2022; Accepted 19 December 2022
Available online 30 December 2022
0736-5845/

suffer from illumination conditions, texture or reflection of the objects. Moreover, visual servoing methods can be affected by calibration errors and occlusion problems, especially when an eye-in-hand configuration is adopted. On the other hand, exploration of the hole neighborhood often requires the use of force–torque sensors, as in [3], where a shape recognition algorithm is adopted to extract the outline of the peg from the force–torque sensor data collected during the contact with the object surface.

A common approach is to map the interaction moments onto the position and tilt of the hole, in order to guide the peg pose during the insertion, see, e.g., [12–14]. However, the presence of force–torque sensors mounted at the end effector of the robot manipulator increases the overall cost and complexity of the system. Thus, estimation of the contact forces from joint position sensors has been adopted [2, 15]. A sensorless method is proposed in [16], where a virtual three-dimensional triple pegs-in-holes model is used for the calculation of contact forces and torques in order to perform a tool exchange task.

Search methods that are proven to be effective consist in tilting and covering the hole neighborhood by moving the peg along an assigned path. In [17], dealing with the problem of inserting a charging plug in its socket, the search path is a Lissajous curve, while the wrist-mounted force–torque sensor is replaced by joint torque sensors often mounted on collaborative robots. However, blind search methods are often time-consuming and require an accurate initial estimate of the hole position. In order to overcome these drawbacks, visual and force–torque data are combined, as in [18], where visual servoing is used to move the peg close to the hole and fine alignment is achieved via spiral search. In [18] a neural network maps the distance from the peg center to the hole in the image coordinate system. Deep learning has been adopted in [19], based on self supervised multi-modal representation of sensory output, and in [20], where a multi-layer perceptron network is trained on a dataset including object position and interaction forces for a polyhedral pegs in contact with the holes. In [21] a method for the visual recognition of complex parts using machine learning is adopted for manipulation tasks with a small process tolerance. In [22] a deep neural network, trained via reinforcement learning, is used to find holes with variable shape and surface finish in a concrete wall. The proposed method consists in moving the peg toward the wall and trying to insert it. If the hole is not found, the peg is detached from the surface and moved to the next position provided by the neural network. In addition to force and moment, displacement is also used as input of the neural network. Instead of a supervised learning technique, the reinforcement learning was used for training in order to avoid the time-consuming dataset labeling process. A combination of a learning approach for object localization and a 3D surface reconstruction has been proposed in [23] to accurately localize the holes on a workpiece, characterized by non-flat steel surface. More in detail, a Convolutional Neural Network is in charge of detecting the holes while the 3D surface reconstruction is obtained with 3D-Digital Image Correlation (3D-DIC) [24]. The 3D-DIC leads to a very accurate surface reconstruction, but it is very time-consuming and its performance strongly depends on the quality of the pattern on the surface and the illumination of the environment. Due to the availability of cheap depth sensing cameras, in the last years, different 3D registration algorithms have been proposed with the goal of detecting the pose of an object in a known coordinate frame [25]. In particular, the Iterative Closest Point (ICP) algorithm [26], which requires to iteratively minimize a cost function depending on an estimate of point correspondences between two point clouds, is largely adopted. Examples of environmental reconstruction can be found in [27], where the geometric registration is combined with a global optimization that makes the process robust to erroneous geometric alignments, and in [28] where the ICP algorithm is adopted for handling a SLAM problem, while in [29,30] the ICP algorithm and its variants are used for object reconstruction.

1.2. Contributions

In this paper, a novel integrated strategy for the execution of a Peg-in-Hole assembly task by means of a collaborative robotic arm is proposed, where the holes are placed on the surface of the target object, a carbon fiber portion of a car safety cell, and the pegs are steel bolts. Due to the manual positioning of the target object in the robotic workcell, large uncertainties on the relative pose of the holes with respect to the robot base are present. More in detail, a reconstruction of the object surface is performed by means of an ICP algorithm, by using a number of point clouds provided by a low-cost off-the-shelf depth sensor; such a reconstruction is compared with the CAD model of the workpiece, in order to localize the hole position and tilt. The accuracy of the hole position estimates depends on the reconstruction error and, in the presence of small peg-hole clearances, it does not guarantee successful peg insertion. Thus, a search phase is introduced, that requires the peg tip to slide on the surface following a trajectory described by Lissajous functions. During the search and the insertion phases, a compliant behavior of the robot at the peg tip level is enforced via admittance control. Extensive experiments confirm that the proposed approach allows to fulfill the assigned task with different conditions of light and in the presence of random position errors. Moreover, the results show the effectiveness of the solution in terms of the required time for surface scanning, reconstruction, and pre-processing of visual data.

The main contributions of the paper can be thus summarized as follows:

- To the authors' best knowledge, sophisticated visual processing techniques, such as ICP, are for the first time applied to solve a class of typical manufacturing tasks, i.e., assembly and parts mating tasks, by adopting low-cost sensors.
- A complete and effective solution to the execution of the above mentioned tasks is developed, where control and visual techniques are fully integrated to achieve a step towards autonomous execution of a class of manufacturing tasks.

This work is part of an industrial research project aimed at integrating the human operators with collaborative automation systems. The considered case study is the insertion of steel bolts in a carbon fiber workpiece, representing a portion of a car safety cell. Currently, the work cycle is performed manually by a human operator, who inserts the bolts by using suitably machined covers/fixtures to determine the correct position of the holes in which the different types of bolts are to be inserted. The expected advantages of an automated system with respect to current manual work cycle are summarized below:

- Improve repeatability and overall quality of the operation.
- Avoid the use of covers/fixtures, subject to wear and deformations.
- Improve ergonomy of the process and reduce operator fatigue, by letting the human to become a supervisor of the process, called to intervene directly only when the automated system encounters problems (e.g., failed insertions).

2. Proposed strategy

In an assembly line, in the presence of small production volumes and complex 3D parts, often the positioning of the target object (on which the holes are located) is manually operated. Thus, it is possible to assume that the workpiece is located within the robot workspace, but uncertainties on the holes position and tilt can be far larger than the task tolerance. Therefore, the following strategy, depicted in Fig. 1, is proposed:

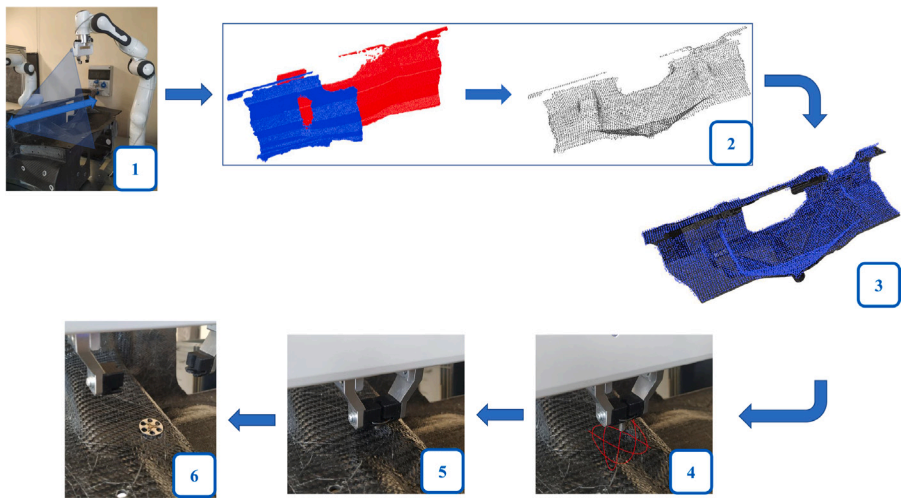

Fig. 1. Proposed strategy: (1) workpiece surface scanning; (2) surface reconstruction; (3) alignment of the reconstructed surface with a known one in order to estimate the holes position; (4) search phase; (5) insertion phase; (6) release of the peg.

- The robot moves an eye-in-hand depth camera along a planned path spanning its workspace, in order to scan the surface of the target object. Along the trajectory, in a predefined set of positions, the depth measures are acquired and a point cloud of the surface is stored.
- The whole surface is reconstructed by aligning the acquired point clouds via an Iterative Closest Point (ICP) registration algorithm [26].
- The reconstructed surface is aligned to a known CAD model of the workpiece, in such a way to detect the hole positions and their normal unit vectors on the reconstructed surface.
- Since the accuracy of the hole position estimates, in the presence of small peg-hole clearances, might not guarantee successful peg insertion, a search phase is performed, where the peg tip explores the neighborhood of the hole by sliding on the surface along a trajectory described by Lissajous functions. During the search phase and the subsequent insertion phase, the robot is commanded to be compliant at the peg tip level by means of an admittance control [31].

For each execution cycle, including the insertion of N pegs, the specific process sequence is detailed in the sequence diagram shown in Fig. 2.

In the following sections, the single steps of the above depicted strategy are discussed in detail.

3. Surface reconstruction and holes localization

In order to build a 3D model of the surface, a depth camera mounted at the end effector of the robot scans the workpiece surface. It is assumed that the robot control unit roughly knows the geometry of the workpiece (e.g., from a CAD model) and its pose with respect to the robot base frame. On the basis of the available information on the surface geometry and pose, as well as on the camera features (range, field of view), n measurement points, each corresponding to an end effector pose, are determined such that the whole surface is spanned. The choice of n is determined in such a way to ensure an overlapping of each two consecutive views: the more wide is the camera field of view, the less are the measurement points needed. Then, the robot visits the n measurement poses and, in each pose, a point cloud is built on the basis of the acquired depth measures.

3.1. Surface reconstruction

In order to reconstruct the workpiece surface, the acquired point clouds are aligned via an Iterative Closest Point (ICP) registration algorithm [26], by using the approach in [27]. In this work, the point-to-plane ICP algorithm has been chosen due to its faster convergence speed with respect to the point-to-point ICP algorithm [32].

More in detail, two point clouds $\mathcal{S}$ and $\mathcal{Q}$, obtained by the same surface in two different views, are in registration if, for any pair of corresponding points $(\boldsymbol{s}_i, \boldsymbol{q}_j)$ representing the same point on the surface, there exists a unique homogeneous transformation matrix $\boldsymbol{T} \in \Re^{4\times4}$ such that

$$\forall \boldsymbol{s}_i \in \mathcal{S}, \exists \boldsymbol{q}_j \in \mathcal{Q} \ | \ \|\boldsymbol{T}\tilde{\boldsymbol{s}}_i - \tilde{\boldsymbol{q}}_j\| = \boldsymbol{0}, \tag{1}$$

where the symbol $\tilde{}$ denotes the homogeneous representation of the coordinate vectors [33].

For reconstructing the surface, a set of homogeneous transformation matrices, $\boldsymbol{T}_i$ $(i = 1, \dots, n)$, are to be determined that allow to align, in a global coordinate frame, the n acquired point clouds, each composed by N_i points, $\mathcal{P}_i = \left\{\boldsymbol{\pi}^i_{i,l}\right\}_{l=1,\dots,N_i}$, expressed in the camera coordinate frame, $\mathcal{F}^i_c$, corresponding to the ith measurement point. In the following, for notation compactness, the superscript i will be dropped. The chosen global coordinate frame is the camera coordinate frame, $\mathcal{F}^1_c$, in which the first point cloud $\mathcal{P}_1$ is acquired. Thus, the transformation matrix $\boldsymbol{T}_1$ is the identity matrix $\boldsymbol{I}_{4\times4}$, while each $\boldsymbol{T}_i$ $(i = 2, \dots, n)$ is in charge of aligning the point cloud $\mathcal{P}_i$ to $\mathcal{P}_1$.

In order to compute the matrices $\boldsymbol{T}_i$ $(i = 1, \dots, n)$, the procedure described in [27] can be adopted. In particular, for each $\mathcal{P}_i$, the transformation matrix $\boldsymbol{T}^i_j$ $(j = i+1, \dots, n)$ that aligns $\mathcal{P}_j$ to $\mathcal{P}_i$ is computed. To this aim, first, the point clouds are down-sampled by using a volume elements (voxel) grid filter [34], and then the ICP algorithm is adopted, where the following cost function has to be minimized

$$C(\boldsymbol{T}^i_j) = \sum_{\boldsymbol{\pi}_{j,l} \in \mathcal{P}_j,\, \boldsymbol{\pi}_{i,l} \in \mathcal{P}_i} \|\boldsymbol{T}^i_j \tilde{\boldsymbol{\pi}}_{j,l} - \tilde{\boldsymbol{\pi}}_{i,l}\|^2. \tag{2}$$

Since $\boldsymbol{T}^i_j$ has 12 unknown components, by considering at least 4 pair of corresponding points it is possible to find the matrix $\boldsymbol{T}^i_j$ that minimizes the cost function (2) by resorting to a least square estimation.

In the *point-to-plane* ICP, the surface, represented by the point cloud $\mathcal{P}_i$, is locally approximated with its tangent plane and the cost function (2) becomes

$$C(\boldsymbol{T}^i_j) = \sum_{\boldsymbol{\pi}_{j,l} \in \mathcal{P}_j,\, \boldsymbol{\pi}_{i,l} \in \mathcal{P}_i} \left((\boldsymbol{T}^i_j \tilde{\boldsymbol{\pi}}_{j,l} - \tilde{\boldsymbol{\pi}}_{i,l})^{\mathrm{T}}\, \tilde{\boldsymbol{n}}^i_{j,l} \right)^2, \tag{3}$$

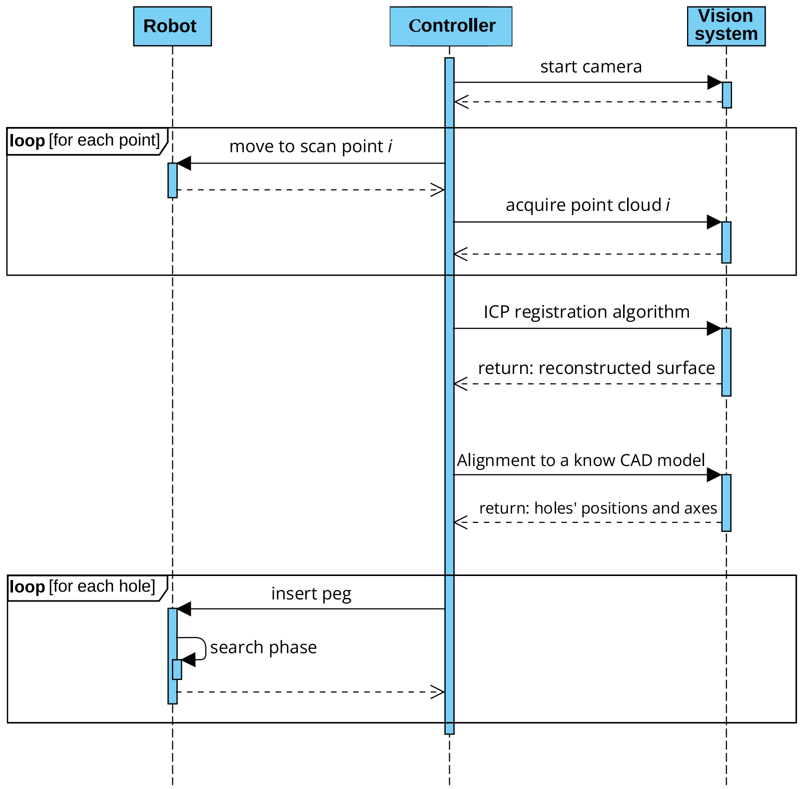

Fig. 2. Sequence diagram of the execution process.

where $\tilde{\boldsymbol{n}}^i_{j,l} = \boldsymbol{T}^i_j \tilde{\boldsymbol{n}}_{j,l}$ is the homogeneous representation of the unit vector normal to the surface represented by the point cloud $\mathcal{P}_j$ in the point $\boldsymbol{\pi}_{j,l}$, expressed in the reference frame of $\mathcal{P}_i$.

The ICP algorithm, for estimating the matrix $\boldsymbol{T}^i_j$, consists of the following steps:

1. A set of $m \geq 4$ control points, with corresponding normals $\boldsymbol{n}_{j,u}$ $(u = 1, \ldots, m)$, is selected in the point cloud $\mathcal{P}_j$. Such points can be chosen on a regular grid.
2. $\boldsymbol{T}^i_j$ is initialized to the identity matrix, i.e., $\boldsymbol{T}^i_j(0) = \boldsymbol{I}_{4\times4}$.

At kth iteration, the following steps are executed:

3. For each control point $\boldsymbol{\pi}_{j,u}$ the corresponding point in $\mathcal{P}_i$ is determined as follows:
   - the homogeneous transformation matrix $\boldsymbol{T}^i_j(k-1)$ is applied to the control point $\boldsymbol{\pi}_{j,u}$ and its normal, $\boldsymbol{n}_{j,u}$, to obtain $\tilde{\boldsymbol{\pi}}^i_{j,u}(k-1) = \boldsymbol{T}^i_j(k-1)\tilde{\boldsymbol{\pi}}_{j,u}$ and $\tilde{\boldsymbol{n}}^i_{j,u}(k-1) = \boldsymbol{T}^i_j(k-1)\tilde{\boldsymbol{n}}_{j,u}$.
   - the intersection $\boldsymbol{\pi}_{i,u}$ of the surface defined by point cloud $\mathcal{P}_i$ with the line defined by $\boldsymbol{\pi}^i_{j,u}(k-1)$ and $\boldsymbol{n}^i_{j,u}(k-1)$ is computed.
4. The transformation matrix, $\bar{\boldsymbol{T}}$, that minimizes the cost function

$$C_m(\bar{\boldsymbol{T}}, \boldsymbol{T}^i_j(k-1)) = \sum_{u=1}^{m} \left((\bar{\boldsymbol{T}}\tilde{\boldsymbol{\pi}}^i_{j,u}(k-1) - \tilde{\boldsymbol{\pi}}_{i,u})^{\mathrm{T}}\, \tilde{\boldsymbol{n}}^i_{j,u}(k-1) \right)^2, \quad (4)$$

   is computed.
5. The transformation matrix is then updated as $\boldsymbol{T}^i_j(k) = \bar{\boldsymbol{T}}\boldsymbol{T}^i_j(k-1)$.

The iterative procedure is stopped when the convergence error is below a tolerance threshold $\bar{e}$, i.e.

$$e = \frac{C_m(\boldsymbol{I}_{4\times4}, \boldsymbol{T}^i_j(k)) - C_m(\boldsymbol{I}_{4\times4}, \boldsymbol{T}^i_j(k-1))}{m^*} \leq \bar{e}, \quad (5)$$

where $m^* \leq m$ is the number of control points for which a correspondence is found.

In [27], it has been shown that most of the identified transformation matrices $\boldsymbol{T}^i_j$ are false positives that lead to misalignment, thus a further global optimization must be carried out. To the aim, a new cost function is defined

$$\mathcal{E}(\boldsymbol{T}_1, \ldots, \boldsymbol{T}_n) = \sum_{\substack{i,j=1 \\ i\neq j}}^{n} \sum_{u=1}^{m} \left(\boldsymbol{T}_i \tilde{\boldsymbol{\pi}}_{i,u} - \boldsymbol{T}_j \tilde{\boldsymbol{\pi}}_{j,u} \right), \quad (6)$$

where $\boldsymbol{T}_i = \boldsymbol{T}_1 \boldsymbol{T}^1_2 \cdots \boldsymbol{T}^{i-1}_i$.

Since $\boldsymbol{T}^i_j$ with $j \neq i+1$ are mostly incorrect, in [27] it is suggested to make the procedure more robust by adding a *line process* [35] to (6). The new cost function is reported in [27] and is omitted here for brevity.

Once the n point clouds are aligned, it is possible to merge them in a single point cloud, i.e.

$$\mathcal{P}_r = \bigcup_{i=1}^{n} \bigcup_{j=1}^{N_i} \boldsymbol{T}_i \tilde{\boldsymbol{\pi}}_{i,j}. \quad (7)$$

Then, the point cloud is down-sampled by using a volume elements (voxel) grid filter [34]. The result is a point cloud, $\mathcal{P}_r$, representing a reconstruction of the surface of the whole workpiece in the coordinate frame $\mathcal{F}^1_c$. An example of reconstructed 3D surface is shown in Fig. 3.

3.2. Holes localization

The next step is the comparison of the obtained point cloud with a known one $\mathcal{P}$, obtained, e.g., via a CAD model. To this aim, for each point of $\mathcal{P}_r$ a vector of local features, called Fast Point Feature Histograms (FPFH), are computed [36]. Then a RANSAC (RANdom SAmple Consensus) algorithm [37] is adopted to find the corresponding points of the two point clouds. At each iteration, μ points are randomly extracted from $\mathcal{P}_r$; then, the corresponding points of $\mathcal{P}$ are determined by looking for the nearest points on the basis of the extracted features. False matchings, i.e., connections between two points that do not

Fig. 3. Example of reconstructed 3D surface.

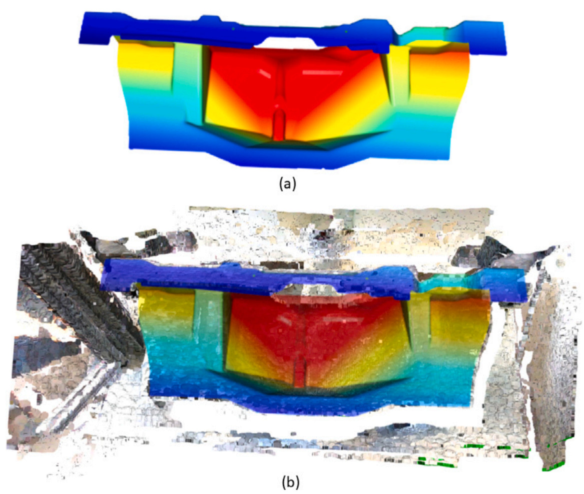

Fig. 4. Holes localization. (a) The point cloud extracted from the CAD model. (b) Alignment of the reconstructed surface and the CAD model.

actually correspond in the point clouds, can be deleted by resorting to pruning algorithms [38]. The transformation matrix between $\mathcal{P}_r$ and $\mathcal{P}$ obtained via the RANSAC algorithm could be not very accurate, thus it is adopted as initial guess of an ICP algorithm that is in charge of refining the alignment. An example of surface-CAD alignment is shown in Fig. 4.

The position of each hole center $\boldsymbol{p}_{h_i}$ and its corresponding axis, i.e. the unit vector normal to the surface, $\boldsymbol{n}_{h_i}$, are known on the point cloud $\mathcal{P}$. Thus, thanks to the above described alignment procedure (see Algorithm 1), it is possible to estimate them in the reconstructed point cloud $\mathcal{P}_r$ and localize them in the coordinate frame $\mathcal{F}_c^1$. Finally, it is possible to determine the camera-end effector transformation via a camera calibration process [39] and, then, localize the holes in the robot base coordinate frame.

4. Task execution

Once the estimates of the hole positions, $\hat{\boldsymbol{p}}_{h_i}$, and the corresponding axes, $\hat{\boldsymbol{n}}_{h_i}$, are determined, the Peg-in-Hole task is performed in three phases:

1. *Approach phase*: in this phase the robot moves the peg close to the hole and aligns the peg axis to the estimated hole axis, $\hat{\boldsymbol{n}}_{h_i}$.
2. *Search phase*: in this phase the neighborhoods of the estimated hole position, $\hat{\boldsymbol{p}}_{h_i}$, are explored in such a way to compensate errors in the position estimate.
3. *Insertion phase*: in this phase the peg is inserted in the hole.

The peg is moved by a robot manipulator with ν joints driven by a position controller. Since in the last two phases physical interaction between the robot and workpiece arises, in order to keep bounded the

Algorithm 1: Holes localization

Input : $\mathcal{P}_r$, $\mathcal{P}$, $\boldsymbol{p}_{h_i}$, $\boldsymbol{n}_{h_i}$
Output: Estimated position of the holes center $\hat{\boldsymbol{p}}_{h_i}$ and their axis $\hat{\boldsymbol{n}}_{h_i}$

for each $\pi^1_{r,i} \in \mathcal{P}_r$ **do**
  Extract FPFH vector
end for
while $i < max\_iterations$ **do**
  Randomly extract μ points from $\mathcal{P}_r$ and find the μ corresponding points in $\mathcal{P}$ via RANSAC algorithm, as the nearest point on the bases of the FPFH features
  Compute the transformation matrix $\boldsymbol{T}_R^i$ between $\mathcal{P}_r$ and $\mathcal{P}$
  if $\mathtt{EvaluationError}(\boldsymbol{T}_R^i) < \mathtt{EvaluationError}(\boldsymbol{T}_0)$ **then**
    $\boldsymbol{T}_0 \leftarrow \boldsymbol{T}_R^i$
  end if
  $i++$
end while
$\boldsymbol{T} \leftarrow \mathtt{ICP\_algorithm}(\mathcal{P}, \mathcal{P}_r, \boldsymbol{T}_0)$
for $i \leftarrow 0$ **to** $holes\_number - 1$ **do**
  $\hat{\boldsymbol{p}}_{h_i} \leftarrow \boldsymbol{T}\tilde{\boldsymbol{p}}_{h_i}$
  $\hat{\boldsymbol{n}}_{h_i} \leftarrow \boldsymbol{T}\tilde{\boldsymbol{n}}_{h_i}$
end for
return $\hat{\boldsymbol{p}}_{h_i}$, $\hat{\boldsymbol{n}}_{h_i}$

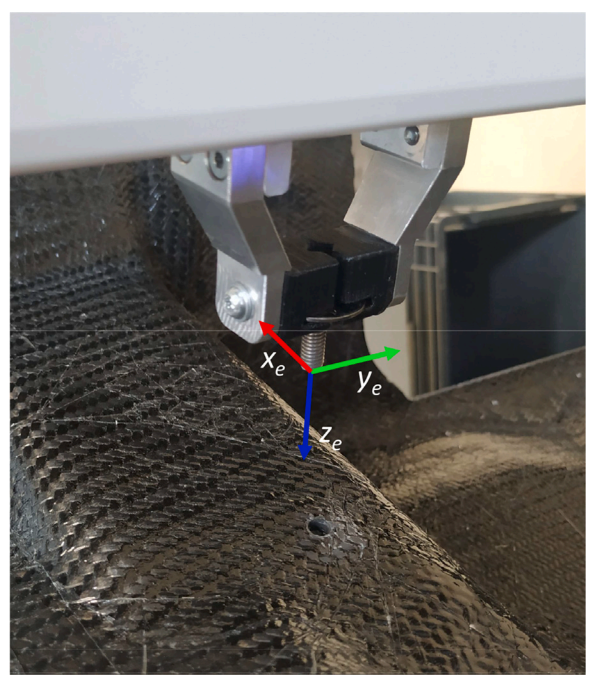

Fig. 5. Reference frame attached to the peg tip.

interaction wrench, the manipulator has been controlled in a compliant mode by implementing an admittance control scheme.

Consider a coordinate frame attached to the peg tip, $\mathcal{F}_e(O_e, \boldsymbol{x}_e, \boldsymbol{y}_e, \boldsymbol{z}_e)$, such that the $\boldsymbol{z}_e$ axis is aligned to the peg axis (see Fig. 5), the position of O_e with respect to the robot base frame is expressed by the (3×1) vector $\boldsymbol{p}_e$, while the orientation of $\mathcal{F}_e$ can be expressed either by the (3×3) orientation matrix, $\boldsymbol{R}_e$, or by a (3×1) vector of Euler angles, $\boldsymbol{\phi}_e$ (e.g., roll-pitch-yaw angles) extracted from $\boldsymbol{R}_e$ [33]. Let $\boldsymbol{x}_e = \left[\boldsymbol{p}_e^{\mathrm{T}}\ \boldsymbol{\phi}_e^{\mathrm{T}}\right]^{\mathrm{T}}$ the (6×1) operational space vector.

Let $\boldsymbol{q}$ ($\dot{\boldsymbol{q}}$) be the $(\nu \times 1)$ vector of joint positions (velocities). The direct kinematics of the manipulator can be expressed either via the functions $\boldsymbol{p}_e = \boldsymbol{p}(\boldsymbol{q})$ and $\boldsymbol{R}_e = \boldsymbol{R}(\boldsymbol{q})$, or by the function $\boldsymbol{x}_e = \boldsymbol{k}(\boldsymbol{q})$.

The $(6 \times \nu)$ matrix $\boldsymbol{J}(\boldsymbol{q}) = \left[\boldsymbol{J}_P{}^{\mathrm{T}}(\boldsymbol{q})\ \boldsymbol{J}_O{}^{\mathrm{T}}(\boldsymbol{q})\right]^{\mathrm{T}}$ is the robot Jacobian, which relates the joint velocities to the linear and angular velocity of the peg, where $\boldsymbol{J}_P(\boldsymbol{q})$ is the $(3\times\nu)$ positional part and $\boldsymbol{J}_O(\boldsymbol{q})$ is the $(3\times\nu)$ orientation part of $\boldsymbol{J}(\boldsymbol{q})$.

4.1. Admittance control

It is assumed that the robot manipulator is not equipped with a wrist-mounted force–torque sensor, but only with torque sensors at the joints, as usual in collaborative robots. The wrench acting on the end effector can be estimated via an observer based on the generalized momentum

$$\boldsymbol{\upsilon} = \boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}}\,, \tag{8}$$

where $\boldsymbol{M}(\boldsymbol{q})$ is the $(\nu \times \nu)$ inertia matrix of the robot [33].

By recalling the dynamic model of a manipulator [33], an estimate of the torques induced at the joints by the contact wrench exerted on the peg tip, $\hat{\boldsymbol{\tau}}_e$, can be computed as follows [40]

$$\hat{\boldsymbol{\tau}}_e = \boldsymbol{K}_o\left[(\boldsymbol{\upsilon}(t) - \boldsymbol{\upsilon}(t_0)) - \int_{t_0}^{t}(\boldsymbol{C}^{\mathrm{T}}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{F}\dot{\boldsymbol{q}} - \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{\tau} + \hat{\boldsymbol{\tau}}_e)d\varsigma\right], \tag{9}$$

where t and t_0 are the current and initial time instant, respectively, $\boldsymbol{K}_o \in \Re^{\nu}$ is a positive definite gain matrix, $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$ is the $(\nu\times\nu)$ matrix of the centripetal and Coriolis terms, $\boldsymbol{g}(\boldsymbol{q})$ is the $(\nu\times 1)$ vector of gravity terms, $\boldsymbol{F}$ is the $(\nu \times \nu)$ matrix of viscous friction terms, $\boldsymbol{\tau} \in \Re^{\nu}$ is the vector of joint torques. By considering the robot dynamics and the expression of the derivative of the inertia matrix, the following dynamics for the estimation is obtained

$$\dot{\hat{\boldsymbol{\tau}}}_e + \boldsymbol{K}_o\hat{\boldsymbol{\tau}}_e = \boldsymbol{K}_o\boldsymbol{\tau}_e. \tag{10}$$

Eq. (10) is a first-order low-pass dynamic system, therefore, $\hat{\boldsymbol{\tau}}_e \to \boldsymbol{\tau}_e$ when $t \to \infty$ for any positive definite gain matrix $\boldsymbol{K}_o$. Since the torques $\boldsymbol{\tau}_e$ can be expressed as

$$\boldsymbol{\tau}_e = \boldsymbol{J}^{\mathrm{T}}\boldsymbol{h}_e, \tag{11}$$

where $\boldsymbol{h}_e \in \Re^6$ is the contact wrench acting on the end-effector, an estimate of the external wrench is [41]

$$\hat{\boldsymbol{h}}_e = \boldsymbol{J}^{\dagger\mathrm{T}}(\boldsymbol{q})\hat{\boldsymbol{\tau}}_e. \tag{12}$$

where $\boldsymbol{J}^{\dagger}$ is a right pseudo-inverse of $\boldsymbol{J}$.

In order to confer proper compliance to the peg tip, an impedance behavior is imposed via the admittance control scheme proposed in [42]. More in detail, the peg tip reference pose $\boldsymbol{x}_{e,r} = [\boldsymbol{p}_{e,r}{}^{\mathrm{T}}\ \boldsymbol{\phi}_{e,r}{}^{\mathrm{T}}]^{\mathrm{T}}$ is computed as

$$\boldsymbol{M}_p\ddot{\boldsymbol{x}}^e_{dr} + \boldsymbol{D}_p\dot{\boldsymbol{x}}^e_{dr} + \boldsymbol{K}_p\boldsymbol{x}^e_{dr} = \boldsymbol{T}_A{}^{\mathrm{T}}(\boldsymbol{\phi}^e_{dr})\hat{\boldsymbol{h}}^e, \tag{13}$$

where $\boldsymbol{M}_p$, $\boldsymbol{D}_p$ and $\boldsymbol{K}_p$ are, respectively, the virtual inertia, damping and stiffness matrices imposed to the peg. The vector $\boldsymbol{x}^e_{dr}$ is the relative displacement between the desired and the reference pose, expressed in the frame $\mathcal{F}_e$

$$\boldsymbol{x}^e_{dr} = \begin{bmatrix}\boldsymbol{p}^e_{dr}\\ \boldsymbol{\phi}^e_{e,dr}\end{bmatrix} = \begin{bmatrix}\boldsymbol{R}_e{}^{\mathrm{T}}(\boldsymbol{p}_{e,d} - \boldsymbol{p}_{e,r})\\ \boldsymbol{\phi}^e_{dr}\end{bmatrix}, \tag{14}$$

where $\boldsymbol{p}_{e,d}$ and $\boldsymbol{p}_{e,r}$ are the desired and reference position, respectively, while $\boldsymbol{\phi}^e_{dr}$ is the vector of Euler angles expressing the relative orientation between the desired frame (whose orientation is expressed by the rotation matrix $\boldsymbol{R}_{e,d}$) and the reference frame (whose orientation is expressed by the rotation matrix $\boldsymbol{R}_{r,d}$), i.e., is the vector of Euler angles extracted from the matrix $\boldsymbol{R}_{e,d}{}^{\mathrm{T}}\boldsymbol{R}_{e,r}$. The matrix $\boldsymbol{T}_A(\cdot)$ in (13) is given by

$$\boldsymbol{T}_A(\cdot) = \begin{bmatrix}\boldsymbol{I}_{3\times 3} & \boldsymbol{O}_{3\times 3}\\ \boldsymbol{O}_{3\times 3} & \boldsymbol{T}(\cdot)\end{bmatrix},$$

where $\boldsymbol{T}(\cdot)$ is the matrix that maps the time derivative of the Euler angles to the angular velocity [33], $\boldsymbol{I}_{3\times 3}$ and $\boldsymbol{O}_{3\times 3}$ are the (3×3) identity and null matrices, respectively. In order to overcome the drawback of representation singularities, as illustrated in [42], the variables involved in the admittance filter have been written with respect to $\mathcal{F}_e$.

The reference trajectory, $\boldsymbol{x}_{e,r}$, output by the admittance filter (13) is fed to the inverse kinematics algorithms (17), during the approach phase, and (20), during the search and insertion phases.

4.2. Approach phase

The robot motion to approach the hole is commanded via a closed-loop inverse kinematics algorithm with two tasks: the first one is aimed at aligning the peg to the hole axis, while the second task is aimed at moving the peg close to the surface.

The *alignment task* is characterized by the task function [23]

$$\sigma_a = (\boldsymbol{z}_e - \hat{\boldsymbol{n}}_{h_i})^{\mathrm{T}}(\boldsymbol{z}_e - \hat{\boldsymbol{n}}_{h_i}), \tag{15}$$

where $\boldsymbol{z}_e$ is the unit vector of the end effector reference frame aligned to the peg tip (see Fig. 5). The corresponding Jacobian matrix is

$$\boldsymbol{J}_a = 2(\hat{\boldsymbol{n}}_{h_i} - \boldsymbol{z}_e)^{\mathrm{T}}\boldsymbol{S}(\boldsymbol{z}_e)\boldsymbol{J}_O(\boldsymbol{q}), \tag{16}$$

where $\boldsymbol{S}(\cdot)$ is the skew symmetric matrix operator performing the cross product [33].

The *position tracking* task is aimed at moving the peg tip from its current position to a target point that is a point belonging to the axis $\hat{\boldsymbol{n}}_{h_i}$ at a given distance from the surface. The task function is the position of the peg, $\boldsymbol{p}_e$, and the task Jacobian is $\boldsymbol{J}_P(\boldsymbol{q})$.

The joint references are computed as

$$\dot{\boldsymbol{q}}_r = \boldsymbol{J}_a^{\dagger}(-k\sigma_a) + \boldsymbol{J}_P^{\dagger}(\dot{\boldsymbol{p}}_{e,r} + \boldsymbol{K}(\boldsymbol{p}_{e,r} - \boldsymbol{p}_e)), \tag{17}$$

where $(\cdot)^{\dagger}$ denotes the right pseudo-inverse of a matrix, $\boldsymbol{p}_{e,r}$ ($\dot{\boldsymbol{p}}_{e,r}$) is the reference position (linear velocity) of the peg, while $k \in \Re$ and $\boldsymbol{K} \in \Re^{3\times 3}$ are, respectively, a positive definite scalar and matrix gain. Since

$$\boldsymbol{J}_P\boldsymbol{J}_a^{\dagger} = \boldsymbol{0}_3, \quad \boldsymbol{J}_a\boldsymbol{J}_P^{\dagger} = \boldsymbol{0}_3{}^{\mathrm{T}}, \tag{18}$$

where $\boldsymbol{0}_3$ is the (3×1) null vector, the two tasks are not conflicting and thus they do not need to be arranged in a priority order [43].

4.3. Search phase

The accuracy of the hole position estimates depends on the reconstruction error and could not guarantee the peg insertion if it is larger than the hole clearance. In order to overcome this problem, a search phase is introduced. The search phase starts when a contact along the direction normal to the reconstructed surface is detected. Such a contact is detected when the projection of the estimated contact force (12) on $\hat{\boldsymbol{n}}_{h_i}$ exceeds a certain threshold.

The search is performed by sliding the peg tip on the workpiece surface, by following a path in the $\boldsymbol{x}_e$–$\boldsymbol{y}_e$ plane of $\mathcal{F}_e$ described by Lissajous functions

$$\begin{aligned} x^e &= a_x \sin(\omega_x(t - t_c)),\\ y^e &= a_y \sin(\omega_y(t - t_c)), \end{aligned} \tag{19}$$

where a_x, a_y, ω_x and ω_y are the sine wave amplitudes and frequencies, respectively, and t_c is the time instant when the peg tip comes in contact with the surface. The superscript e denotes that the trajectory is expressed in the frame $\mathcal{F}_e$.

In order to reduce mechanical stresses both on the workpiece and on the robot's joints, and to allow the sliding of the peg, a compliant behavior is enforced to the robot at the peg tip level. To this aim, the admittance control described in Section 4.1, has been adopted with the gain matrices reported in Table 1.

Table 1
Gain matrices adopted in the task execution.

Gain Matrix	Value
$\boldsymbol{M}_p$	diag$[15 \quad 15 \quad 15 \quad 0.5 \quad 0.5 \quad 0.5]$
$\boldsymbol{D}_p$	diag$[30 \quad 30 \quad 30 \quad 1 \quad 1 \quad 1]$
$\boldsymbol{K}_p$	diag$[45 \quad 45 \quad 150 \quad 1.5 \quad 1.5 \quad 1.5]$
$\boldsymbol{K}$	diag$[150 \quad 150 \quad 150]$
$\boldsymbol{\Lambda}$	diag$[150 \quad 150 \quad 150 \quad 30 \quad 30 \quad 30]$
k	5

In order to compute the joint references for the robot from the operational-space references output by the admittance filter, an inverse kinematics algorithm is used, i.e.,

$$\dot{\boldsymbol{q}}_r = \left[\boldsymbol{T}_A(\boldsymbol{\phi}^e_{d r})\boldsymbol{J}\right]^{\dagger}(\dot{\boldsymbol{x}}_r + \boldsymbol{\Lambda}(\boldsymbol{x}_{e,r} - \boldsymbol{x}_e)), \tag{20}$$

where $\boldsymbol{\Lambda} \in \Re^{6\times 6}$ is a positive definite gain matrix.

4.4. Insertion phase

Since, during the search phase, the planned trajectory has been chosen some millimeters under the hole surface, in order to allow a coupling force along the z_e axis, when the peg tip detects the hole, the peg is immediately inserted of about 1 mm and the search phase is stopped. From this peg's position a new trajectory along the estimated hole's axis is planned, in order to perform the insertion phase. The robot manipulator, compliantly, push the peg into the hole for other 7 mm and then releases it. During the insertion the admittance control scheme in Section 4.1 has been used with the gain matrices reported in Table 1.

It is worth noticing that the compliance conferred by the admittance scheme is in charge of compensating small orientation errors between the peg and the normal of the hole. Such errors can cause resistant torques that, through (13), provide a new orientation for the peg tip.

The joint references for the robot are computed from the operational-space references output by the admittance filter via the inverse kinematics algorithm (20).

5. Experimental case study

The proposed strategy has been experimentally validated using a collaborative robot Franka Emika Panda, equipped with an Intel Realsense D435 RGB-D camera.

The camera system has been calibrated by using 16 images of a 2D checkerboard flat pattern according to the eye-in-hand camera calibration method proposed in [39], implemented in the VISP library [44]. A carbon fiber 3D workpiece has been used, representing a portion of a supercar's safety cell, while steel bolts, named *Big Heads*, have to be inserted in 3 holes (see Fig. 6). A very small clearance, below 1 mm, is present between the hole and the Big Heads; moreover, the task is made more challenging by the fact that the peg is threaded while the holes are very rough.

In order to have statistically significant results, 35 tests have been carried out in different light conditions and by randomly positioning the object in the robot workspace, so as to mimic the actual work cycle, where the workpiece is manually positioned and errors up to about 10 cm–15 cm and 15 deg–20 deg can be experienced with respect to the planned nominal pose.

5.1. Software implementation

The Franka Emika Panda is controlled by using the Franka Control Interface (FCI) with the `libfranka` C++ open source library, that enables the direct robot control with an external workstation PC connected via Ethernet. The workstation is equipped with the Ubuntu 18.04 LTS operating system running on an Intel Xeon 3.7 GHz CPU with 32 GB RAM. The vision system is executed on the above described workstation and the `librealsense2` library has been used in order to acquire the camera data. The surface reconstruction has been performed, according to the procedure described in Section 3.1, by using the `multiway registration` algorithm provided by the `Open3D` library [38]. The reconstructed surface is overlapped and compared with the point cloud extracted by the CAD model, according to the procedure described in Section 3.2, by using the `global registration` algorithm provided by the `Open3D` library [38].

5.2. Experimental results

In the first phase, the robot scans the 3D surface moving along a fixed path, planned by means of a linear interpolation between the first and last camera poses. Such poses, in terms of position and orientation of the camera, have been computed in such a way to have the nominal position of the workpiece in the camera field of view with a certain tolerance to take into account the positioning errors. During the scanning phase, for each test, $n = 8$ different point clouds have been acquired via the depth sensor to reconstruct the workpiece surface. The number of point clouds is chosen in order to have a wide overlap between two consecutive ones. An example of reconstructed surface is shown in Fig. 7.

In order to evaluate the quality of the reconstruction, the *re-projection* error between the 3D reconstructed surface and the CAD model of the workpiece has been computed. More in detail, the acquired points have been projected onto the CAD surface using a 3D software and then the distance between the acquired and projected points has been computed. In Fig. 8, the root mean square error for a single test is reported, the average error is around 5 mm and it is quite homogeneous along the whole surface. The mean square error computed by considering all the 35 tests is about 6 mm. It is worth noticing that different reconstruction errors have been experienced due to both the different positioning and the different light conditions of the experiments.

Regarding the performance of the task execution, the following indices have been considered: the duration of the search phase and the error between the estimated and actual hole position. In order to compute the duration of the search phase, the initial time is the instant, t_c, when the first contact between the peg tip and the surface arises, while the search ends when the peg tip is inserted of about 1 mm in the hole.

In Fig. 9 the duration of the search phase evaluated for each hole in the 35 tests is reported. The average time is around 5 s, the maximum duration has been experienced for the hole 3, since, in order to validate the procedure also on a tilted surface, it was drilled in post-processing without using a CNC drilling machines. In test 27, for hole 3, the search phase has not been executed, since the bolt has been directly inserted in the hole.

Some snapshots of the search phase are reported in Fig. 10: Fig. 10(a) shows the bolt approaching the surface with a certain error, Fig. 10(b) shows the bolt sliding on the surface, Fig. 10(c) shows the phase in which the hole is detected and the insertion is performed.

In Fig. 11 the holes position errors computed in the 35 tests are reported. More in detail, the bottom side of the grey box is the minimum error obtained for the 3 holes in each test, while the upper side is the average one. The upper side of the orange box represents the maximum error obtained for the 3 holes in each test. For each hole, the position error is evaluated by considering the distance, in the plane of the hole neighborhood, between the reconstructed and the actual hole position. The average error is about 4 mm, the minimum error has been registered for test 27, equal to zero. The worst performance is experienced at test 29, and depends on an imperfect reconstruction that leads to an error of 16.5 mm for the third hole.

Table 2 reports the numerical data registered in the experiments, the first three columns report the search phase duration and the last three contain the position error measured for each hole. On the total

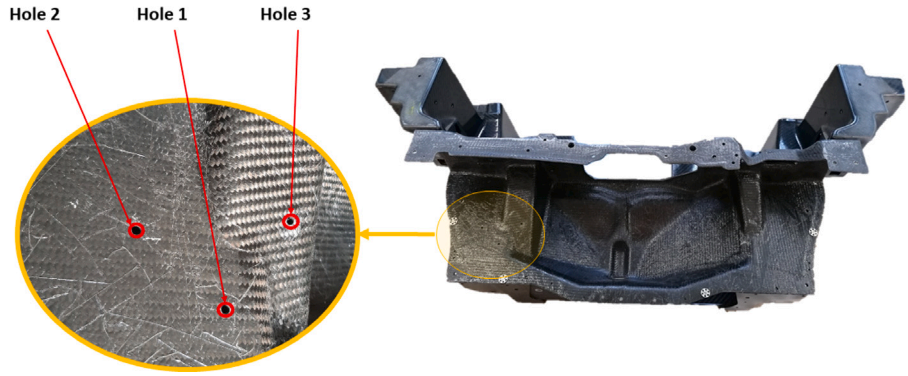

Fig. 6. Workpiece and holes.

Fig. 7. Workpiece surface reconstruction.

Table 2
Test results. The empty cell represents the failed insertions.

Test	Search duration [s]			Position error [mm]		
	Hole 1	Hole 2	Hole3	Hole 1	Hole 2	Hole3
1	4.772	5.028	4.85	3.3526	3.4395	4.6157
2	5.016	5.307	5.056	4.2496	3.4747	2.6349
3	4.841	5.086	4.861	4.3832	0.37241	2.5932
4	4.97	–	4.905	3.6456	–	1.7596
5	4.755	4.973	–	5.3298	5.5974	–
6	4.642	4.863	–	4.2966	4.5197	–
7	4.746	4.946	5.103	3.5001	4.411	4.6674
8	4.927	5.15	6.171	5.377	4.8082	5.1504
9	5.044	5.309	5.038	3.8065	5.2065	1.2463
10	4.797	5.023	5.903	3.1594	2.4921	3.4863
11	4.973	5.227	4.955	2.5786	3.3899	2.599
12	5.082	5.29	5.06	3.1414	4.0013	1.3339
13	5.082	5.404	5.141	2.3173	2.6899	1.3026
14	4.93	5.154	5.796	1.8998	2.058	1.6834
15	4.655	4.906	–	4.1183	5.5196	–
16	4.653	4.937	–	5.2602	5.7614	–
17	4.77	5.03	5.066	4.7524	4.7199	4.2072
18	4.776	5.051	–	6.0915	6.2506	–
19	4.843	5.139	4.885	3.4184	3.1839	3.9258
20	4.805	5.04	4.982	4.8406	4.9056	5.8433
21	4.936	5.208	4.942	3.6834	3.624	2.8966
22	4.682	4.962	5.94	1.7234	2.9898	3.284
23	4.939	5.227	5.545	4.398	4.7071	1.8894
24	5.067	5.401	5.096	6.0223	6.0605	4.0043
25	5.125	5.415	5.098	3.7016	2.0056	3.617
26	5.037	5.308	5.048	3.4188	3.623	1.892
27	5.294	5.558	0	4.608	5.1097	0
28	4.979	5.246	4.967	3.8716	3.8885	2.5253
29	4.981	5.22	5.285	3.9818	4.2679	16.544
30	5.031	5.233	5.372	3.4597	4.0809	1.499
31	5.201	5.469	5.248	5.5154	5.953	1.5262
32	5.268	5.503	5.3	5.377	5.3839	3.1954
33	5.403	5.705	5.469	5.6531	5.8107	2.7984
34	5.472	5.833	4.671	6.0433	6.4256	5.17683
35	5.163	5.372	5.19	5.3041	5.399	3.2412

105 insertions only 6 failed, obtaining an overall insertion success rate of 94.2%. The video of a test can be found at https://youtu.be/nc-Ku3eiZqc.

A set of additional experiments has been carried out to assess the performance of the approach in the presence of non-negligible surface irregularities (e.g., protrusions), by adding artificial protrusions in the neighborhood of the second hole. The results, which are not detailed here for the sake of brevity, have shown that the task is correctly executed if the protrusion is not characterized by too sharp edges, thanks to compliance enforced by the admittance control, which allows the peg tip to adapt to the surface profile smoothly.

The whole cycle from scanning to releasing the first peg has an average duration, computed over the 35 tests, of about 2 min and 17 s. More in detail, about 2 min are needed for the scanning phase, executed once at the process start, including the storage of the point clouds and the reconstruction computation. This duration is mainly due to the need to starting the communications with the camera and save the point clouds, that include a total of 921600 points. The average of the time needed for point cloud registrations, surface reconstruction and comparison with the point cloud extracted from the CAD model is about 14 s. In the remaining 17 s the peg is moved in contact with the surface and the search phase (which has an average duration of 5 s) is performed.

5.3. Discussion

The last performance index, i.e. the position error between the estimated and actual hole's position, takes into account the overall procedure error. More in detail, the results are affected by different uncertainties and errors, i.e.:

- error between the actual workpiece and the reconstructed one;
- overlapping error between the reconstructed point cloud and the CAD model;
- workpiece fabrication error;
- robot positioning error;
- camera-robot calibration error.

The overlapping between the reconstructed point cloud and the CAD model, is performed via numerical optimization, and thus is affected by a certain error, that is a trade off between the acceptable computation burden and the accuracy. Moreover, the experiments have been conducted in different light conditions due to the presence of natural light in the laboratory, but the obtained reconstruction errors are comparable, with differences on the root mean squared error of the order of few millimeters.

The workpiece has been fabricated with a geometrical tolerance of about 1.5 mm; thus, such a tolerance influences the discrepancy between the real workpiece and its CAD model. It is worth noticing that the tolerance is larger than the peg-hole clearance. Moreover, also the camera-robot homogeneous transformation matrix between the peg's

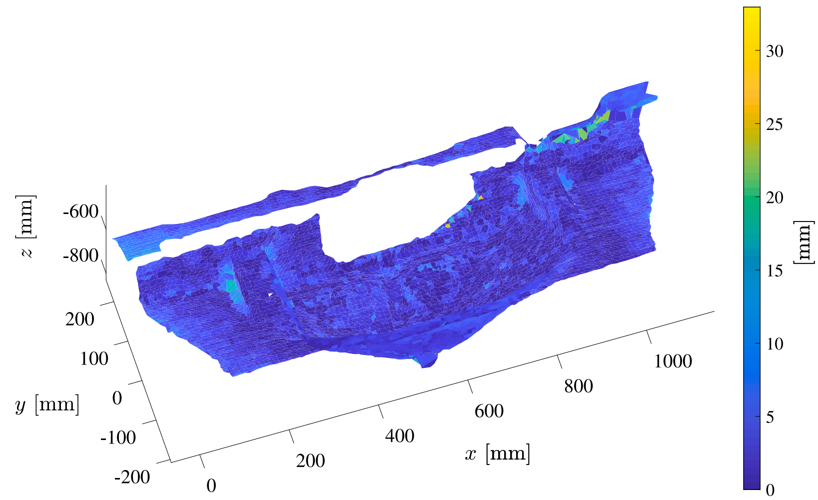

Fig. 8. Root mean square error of the reconstructed point cloud with respect to the CAD model.

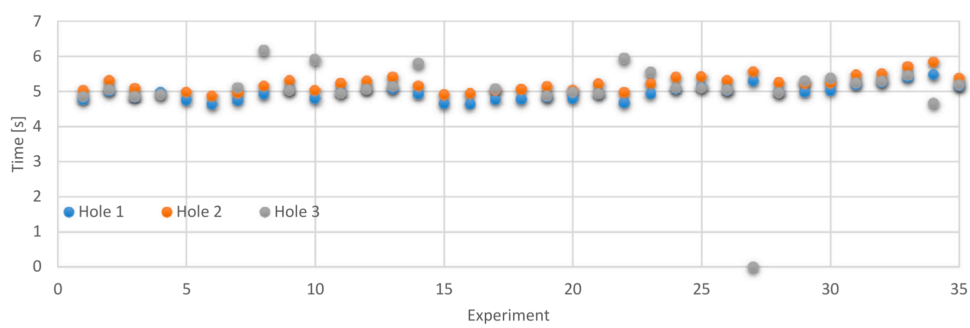

Fig. 9. Search phase duration.

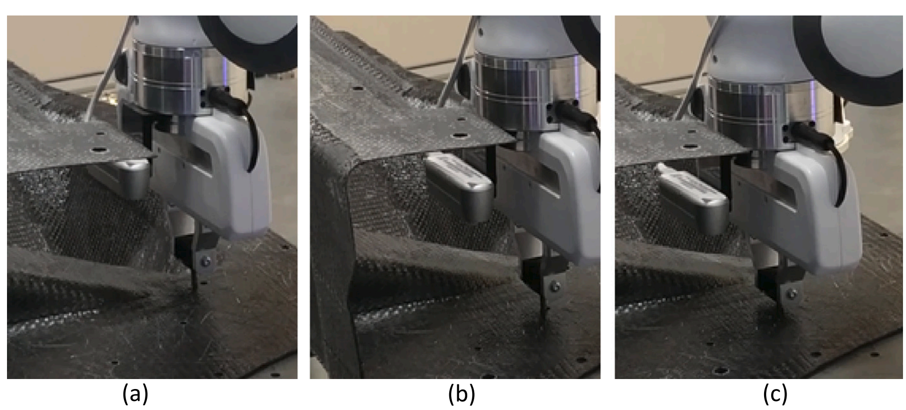

Fig. 10. Snapshots of the search phase.

tip coordinate frame and the camera frame has been computed via an optimization procedure [39] and, thus, is affected by uncertainty.

In this paper, a complete procedure to carry out an entire assembly sequence of peg-in-hole type for a wide class of workpieces is built, while most of the existing approaches:

1. either adopt completely different strategies (so as to make very hard a comparison)
2. or consider the holes laying in a small portion of the workpiece's surface, e.g., the ones in the camera field of view, as a difference with our method that works also for large objects requiring a wide motion of the robot to span the whole surface.

For the above reasons, a complete quantitative performance comparison with other approaches present in the literature can be hardly carried out. However, some considerations about accuracy can be drawn with respect to specific published approaches. In particular, in [22] a use-case scenario, where the clearance between the peg and hole is below 1 mm, is presented. A visual module for approaching the hole is adopted and, then, a deep neural network with reinforcement learning is used for finding the hole. The authors experienced a success rate of 96.1% for holes not considered in the training with a mean tempting time of 12.4 s. The success rate obtained with our approach (94.2%) is comparable, while the search time is largely smaller (5 s). Our approach require additional time (about 132 s) for completing the surface reconstruction, the comparison with the CAD, and the robot

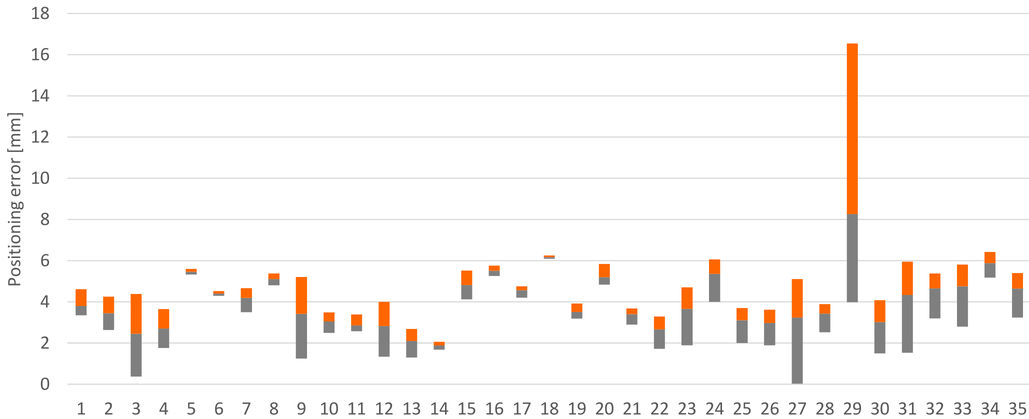

Fig. 11. Error between the reconstructed and the actual holes' position. Grey box (bottom side): minimum error. Grey box (upper side): average error. Orange box (upper side): maximum error.

Table 3
Comparison table between different peg-in-hole experiments setup. Legend: N.H. (Number of Holes) W.D. (Workpiece Dimension) H.S.G. (Holes Surface Geometry)

Method	Features		
	N.H.	W.D.	H.S.G.
Chang et al. [10]	Single	Small	2D
Park et al. [15]	Single	Small	2D
Jokesch et al. [17]	Single	Wide	2D
Triyonoputro et al. [18]	Single	Medium	2D
Yasutomi et al. [22]	Single	Small	2D
Our method	Multiple	Wide	3D

approach to the holes, but, it is worth noticing that, such phases are required only once for each workpiece. In [22], also a blind search method was tested, consisting of moving the peg using a spiral trajectory. This method has achieved a success rate of 100%, but the execution took up a mean of 102.5 s, depending on the starting point. In such a case, our method allows to reduce the total timing in the presence of more holes in the same workpiece.

Finally, a qualitative comparison with similar methods (see Table 3) have been conducted taking into account the number of holes used for the experiments, the dimension of the workpiece, and the holes' surface geometry. In particular, the hole surface geometry, i.e., the morphology of the workpiece region where the holes are located, is considered "2D" when there are only holes with parallel axes on the same plane and "3D" when the holes are characterized by axes with different orientations. The proposed method is more general than others and it can be used to perform the peg insertion in multiple holes located on large workpieces with complex shapes.

6. Conclusions and future directions

In this work, a class of assembly tasks have been considered, involving insertion and/or bonding of mechanical parts, where the goal is to complete the task execution by resorting to visual sensing. In particular, a peg-in-hole task has been accomplished by a fully autonomous manipulator, equipped with a depth camera, in an industrial scenario.

Experimental results confirm the effectiveness of the approach even with wide object positioning errors (10 cm–15 cm and 15 deg–20 deg) with respect to the nominal pose, assigned in the task definition. Moreover, the search phase strengthen the procedure by adding robustness to reconstruction errors, as can be seen in the case of test 29, when even if an error of 16.5 mm on the third hole positioning has been experienced, the insertion has been successfully completed.

As future work, a neural network for the image segmentation could be added to the pipeline in order to better discriminate the object from the background and improve the overlapping with the CAD model. Moreover, a high-resolution camera could be adopted in lieu of the Intel RealSense D435, that allows more accurate point clouds and a better reconstruction.

CRediT authorship contribution statement

M. Nigro: Conceptualization, Methodology, Software, Investigation, Validation, Formal analysis, Writing – original draft. **M. Sileo:** Conceptualization, Methodology, Software, Investigation, Validation, Formal analysis, Writing – original draft. **F. Pierri:** Conceptualization, Methodology, Formal analysis, Writing – review & editing, Supervision. **D.D. Bloisi:** Conceptualization, Methodology, Software, Writing – review & editing, Supervision. **F. Caccavale:** Conceptualization, Methodology, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.rcim.2022.102519.

Data availability

Data will be made available on request.

Acknowledgment

This research has been supported by the project ICOSAF (Integrated Collaborative Systems for Smart Factory - ARS01_00861), funded by MUR under PON R&I 2014–2020.

References

[1] N. Hogan, Impedance control: An approach to manipulation: Parts I-III, J. Dyn. Syst. Meas. Control 107 (1) (1985) 1–24.
[2] H. Park, J.-H. Bae, J.-H. Park, M.-H. Baeg, J. Park, Intuitive peg-in-hole assembly strategy with a compliant manipulator, in: IEEE ISR 2013, IEEE, 2013, pp. 1–5.
[3] Y.-L. Kim, B.-S. Kim, J.-B. Song, Hole detection algorithm for square peg-in-hole using force-based shape recognition, in: 2012 IEEE International Conference on Automation Science and Engineering, CASE, IEEE, 2012, pp. 1074–1079.
[4] H.-C. Song, Y.-L. Kim, J.-B. Song, Guidance algorithm for complex-shape peg-in-hole strategy based on geometrical information and force control, Adv. Robot. 30 (8) (2016) 552–563.
[5] F. Caccavale, C. Natale, B. Siciliano, L. Villani, Control of two industrial robots for parts mating, in: Proceedings of the 1998 IEEE International Conference on Control Applications (Cat. No. 98CH36104), Vol. 1, IEEE, 1998, pp. 562–566.

[6] H. Park, P.K. Kim, J.-H. Bae, J.-H. Park, M.-H. Baeg, J. Park, Dual arm peg-in-hole assembly with a programmed compliant system, in: 2014 11th International Conference on Ubiquitous Robots and Ambient Intelligence, URAI, IEEE, 2014, pp. 431–433.

[7] S. Huang, K. Murakami, Y. Yamakawa, T. Senoo, M. Ishikawa, Fast peg-and-hole alignment using visual compliance, in: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2013, pp. 286–292.

[8] P. Nagarajan, S.S. Perumaal, B. Yogameena, Vision based pose estimation of multiple peg-in-hole for robotic assembly, in: International Conference on Computer Vision, Graphics, and Image Processing, Springer, 2016, pp. 50–62.

[9] Z. Yang, W. Liu, H. Li, Z. Li, A coaxial vision assembly algorithm for un-centripetal holes on large-scale stereo workpiece using multiple-DOF robot, in: 2018 IEEE International Conference on Imaging Systems and Techniques, IST, IEEE, 2018, pp. 1–6.

[10] R.-J. Chang, C.Y. Lin, P.S. Lin, Visual-based automation of peg-in-hole microassembly process, J. Manuf. Sci. Eng. 133 (4) (2011).

[11] A. Papavasileiou, P. Aivaliotis, S. Aivaliotis, S. Makris, An optical system for identifying and classifying defects of metal parts, Int. J. Comput. Integr. Manuf. 35 (3) (2022) 326–340.

[12] H. Bruyninckx, S. Dutre, J. De Schutter, Peg-on-hole: A model based solution to peg and hole alignment, in: Proceedings of 1995 IEEE International Conference on Robotics and Automation, Vol. 2, IEEE, 1995, pp. 1919–1924.

[13] W.S. Newman, Y. Zhao, Y.-H. Pao, Interpretation of force and moment signals for compliant peg-in-hole assembly, in: Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164), Vol. 1, IEEE, 2001, pp. 571–576.

[14] K. Sharma, V. Shirwalkar, P.K. Pal, Intelligent and environment-independent peg-in-hole search strategies, in: 2013 International Conference on Control, Automation, Robotics and Embedded Systems, CARE, IEEE, 2013, pp. 1–6.

[15] H. Park, J. Park, D.-H. Lee, J.-H. Park, M.-H. Baeg, J.-H. Bae, Compliance-based robotic peg-in-hole assembly strategy without force feedback, IEEE Trans. Ind. Electron. 64 (8) (2017) 6299–6309.

[16] P. Aivaliotis, G. Michalos, S. Makris, Cooperating robots for fixtureless assembly: Modelling and simulation of tool exchange process, Int. J. Comput. Integr. Manuf. 31 (12) (2018) 1235–1246.

[17] M. Jokesch, J. Suchỳ, A. Winkler, A. Fross, U. Thomas, Generic algorithm for peg-in-hole assembly tasks for pin alignments with impedance controlled robots, in: Robot 2015: Second Iberian Robotics Conference, Springer, 2016, pp. 105–117.

[18] J.C. Triyonoputro, W. Wan, K. Harada, Quickly inserting pegs into uncertain holes using multi-view images and deep network trained on synthetic data, 2019, arXiv preprint arXiv:1902.09157.

[19] M.A. Lee, Y. Zhu, K. Srinivasan, P. Shah, S. Savarese, L. Fei -Fei, A. Garg, J. Bohg, Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks, in: 2019 International Conference on Robotics and Automation, ICRA, IEEE, 2019, pp. 8943–8950.

[20] G. De Magistris, A. Munawar, T.-H. Pham, T. Inoue, P. Vinayavekhin, R. Tachibana, Experimental force-torque dataset for robot learning of multi-shape insertion, 2018, arXiv preprint arXiv:1807.06749.

[21] P. Aivaliotis, A. Zampetis, G. Michalos, S. Makris, A machine learning approach for visual recognition of complex parts in robotic manipulation, Procedia Manuf. 11 (2017) 423–430.

[22] A.Y. Yasutomi, H. Mori, T. Ogata, A peg-in-hole task strategy for holes in concrete, in: 2021 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2021, pp. 2205–2211.

[23] M. Nigro, M. Sileo, F. Pierri, K. Genovese, D.D. Bloisi, F. Caccavale, Peg-in-hole using 3D workpiece reconstruction and CNN-based hole detection, in: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2020, pp. 4235–4240.

[24] M.A. Sutton, J.J. Orteu, H. Schreier, Image Correlation for Shape, Motion and Deformation Measurements: Basic Concepts, Theory and Applications, Springer Science & Business Media, 2009.

[25] B. Bellekens, V. Spruyt, R. Berkvens, R. Penne, M. Weyn, A benchmark survey of rigid 3D point cloud registration algorithms, Int. J. Adv. Intell. Syst. 8 (2015) 118–127.

[26] Y. Chen, G. Medioni, Object modelling by registration of multiple range images, Image Vis. Comput. 10 (3) (1992) 145–155.

[27] S. Choi, Q.-Y. Zhou, V. Koltun, Robust reconstruction of indoor scenes, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5556–5565.

[28] P. Henry, M. Krainin, E. Herbst, X. Ren, D. Fox, RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments, in: Experimental Robotics, Springer, 2014, pp. 477–491.

[29] L. Valgma, M. Daneshmand, G. Anbarjafari, Iterative closest point based 3d object reconstruction using rgb-d acquisition devices, in: 2016 24th Signal Processing and Communication Application Conference, SIU, IEEE, 2016, pp. 457–460.

[30] M. Zollhöfer, P. Stotko, A. Görlitz, C. Theobalt, M. Nießner, R. Klein, A. Kolb, State of the art on 3D reconstruction with RGB-d cameras, in: Computer Graphics Forum, Vol. 37, Wiley Online Library, 2018, pp. 625–652.

[31] L. Villani, J. De Schutter, Force control, in: Springer-Verlag, B. Siciliano, O. Khatib (Eds.), Handbook of Robotics, 2008.

[32] S. Rusinkiewicz, M. Levoy, Efficient variants of the ICP algorithm, in: Proceedings Third International Conference on 3-D Digital Imaging and Modeling, IEEE, 2001, pp. 145–152.

[33] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, Robotics – Modelling, Planning and Control, Springer, London, UK, 2009.

[34] A. Kaufman, D. Cohen, R. Yagel, Volume graphics, Computer 26 (7) (1993) 51–64.

[35] M.J. Black, A. Rangarajan, On the unification of line processes, outlier rejection, and robust statistics with applications in early vision, Int. J. Comput. Vis. 19 (1) (1996) 57–91.

[36] R.B. Rusu, N. Blodow, M. Beetz, Fast point feature histograms (FPFH) for 3D registration, in: 2009 IEEE International Conference on Robotics and Automation, IEEE, 2009, pp. 3212–3217.

[37] H. Li, J. Qin, X. Xiang, L. Pan, W. Ma, N.N. Xiong, An efficient image matching algorithm based on adaptive threshold and RANSAC, IEEE Access 6 (2018) 66963–66971.

[38] Q.-Y. Zhou, J. Park, V. Koltun, Open3D: A modern library for 3D data processing, 2018, arXiv:1801.09847.

[39] R.Y. Tsai, R.K. Lenz, et al., A new technique for fully autonomous and efficient 3D robotics hand/eye calibration, IEEE Trans. Robot. Autom. 5 (3) (1989) 345–358.

[40] A. De Luca, R. Mattone, Sensorless robot collision detection and hybrid force/motion control, in: 2005 IEEE Int.Conf. on Robotics and Automation, 2005, pp. 999–1004.

[41] F. Ficuciello, A. Romano, L. Villani, B. Siciliano, Cartesian impedance control of redundant manipulators for human-robot co-manipulation, in: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2014, pp. 2120–2125.

[42] F. Caccavale, C. Natale, B. Siciliano, L. Villani, Six-dof impedance control based on angle/axis representations, IEEE Trans. Robot. Autom. 15 (2) (1999) 289–300.

[43] G. Antonelli, Stability analysis for prioritized closed-loop inverse kinematic algorithms for redundant robotic system, IEEE Trans. Robot. 25 (2009) 985–994.

[44] É. Marchand, F. Spindler, F. Chaumette, Visp for visual servoing: A generic software platform with a wide class of robot control skills, IEEE Robot. Autom. Mag. 12 (4) (2005) 40–52.