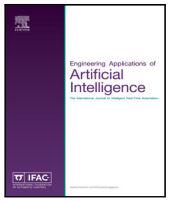
Engineering Applications of Artificial Intelligence 128 (2024) 107486

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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



Vision-enhanced Peg-in-Hole for automotive body parts using semantic image segmentation and object detection

Monica Sileo [a], Nicola Capece [b], Monica Gruosso [b], Michelangelo Nigro [c], Domenico D. Bloisi [d], Francesco Pierri [a,*], Ugo Erra [b]

[a] School of Engineering, University of Basilicata, 85100 Potenza, Italy
[b] Department of Mathematics, Computer Science, and Economics, University of Basilicata, 85100 Potenza, Italy
[c] Hyp-er_objects s.r.l., 85014 Laurenzana (PZ), Italy
[d] Department of International Humanities and Social Sciences, UNINT University, 00147 Rome, Italy

ARTICLE INFO

Keywords:
Peg-in-Hole
Robot vision
Deep learning

ABSTRACT

Artificial Intelligence (AI) is an enabling technology in the context of Industry 4.0. In particular, the automotive sector is among those who can benefit most of the use of AI in conjunction with advanced vision techniques. The scope of this work is to integrate deep learning algorithms in an industrial scenario involving a robotic Peg-in-Hole task. More in detail, we focus on a scenario where a human operator manually positions a carbon fiber automotive part in the workspace of a 7 Degrees of Freedom (DOF) manipulator. To cope with the uncertainty on the relative position between the robot and the workpiece, we adopt a three stage strategy. The first stage concerns the Three-Dimensional (3D) reconstruction of the workpiece using a registration algorithm based on the Iterative Closest Point (ICP) paradigm. Such a procedure is integrated with a semantic image segmentation neural network, which is in charge of removing the background of the scene to improve the registration. The adoption of such network allows to reduce the registration time of about 28.8%. In the second stage, the reconstructed surface is compared with a Computer Aided Design (CAD) model of the workpiece to locate the holes and their axes. In this stage, the adoption of a Convolutional Neural Network (CNN) allows to improve the holes' position estimation of about 57.3%. The third stage concerns the insertion of the peg by implementing a search phase to handle the remaining estimation errors. Also in this case, the use of the CNN reduces the search phase duration of about 71.3%. Quantitative experiments, including a comparison with a previous approach without both the segmentation network and the CNN, have been conducted in a realistic scenario. The results show the effectiveness of the proposed approach and how the integration of AI techniques improves the success rate from 84.5% to 99.0%.

1. Introduction

Industry 4.0 concerns the integration of new technologies, including Internet of Things (IoT), cloud computing, and Artificial Intelligence (AI) into manufacturing facilities. Therefore, Industry 4.0 smart factories require advanced sensors, integrated software, and robotics components that collect and analyze data to improve productivity and quality of the products (Banan et al., 2020; Fan et al., 2020; Afan et al., 2021). In particular, in the case of small-lot and customizable production, with high flexibility and reconfigurability needs, robots can play a crucial role to improve repeatability, overall quality of the operation, and ergonomy of the process by reducing the operator fatigue. However, to endow robotic systems with advanced capabilities of executing complex tasks in dynamic environments, where robots and humans share the same workspace, suitable sensors and learning methodologies need to be investigated.

This work is part of an industrial research project whose aim is to introduce vision and AI solutions to solve real industrial problems. Therefore, we focus on the design and development of an AI-based solution for autonomous Peg-in-Hole task in the context of the supercar automotive industry. Supercar market naturally represents a very small segment of the entire car production market because cars in this segment are created using technological craftsmanship. A Peg-in-Hole assembly task is considered, where five holes are located on the surface of a carbon fiber workpiece, representing a portion of a supercar's safety cell (see Fig. 1). The pegs are steel bolts with a peg-hole clearance below 1 mm. At the present time, the task is manually

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

https://doi.org/10.1016/j.engappai.2023.107486
Received 7 November 2022; Received in revised form 14 July 2023; Accepted 9 November 2023
Available online 21 November 2023

Fig. 1. Alfa Romeo C4 is a supercar with carbon fiber visible parts inside the car cockpit.

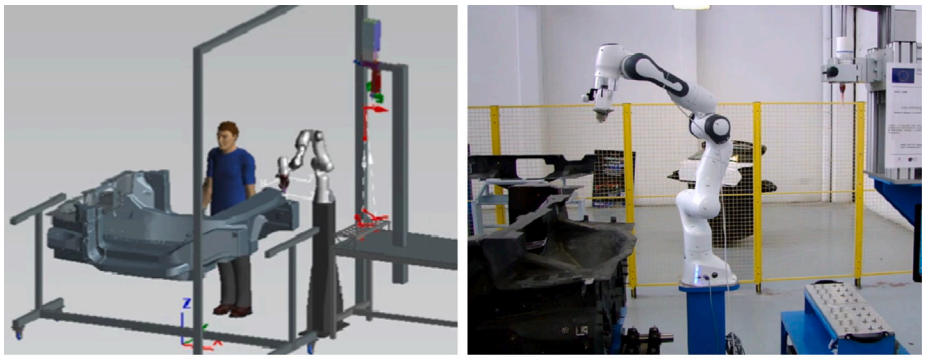


Fig. 2. Structure of the workcell. (a) The carbon fiber workpiece is manually positioned near the robot using a cart. (b) The robot does not know the exact position of the workpiece.

performed by a human operator, which manipulates the safety cell and inserts the steel bolts.

Such a problem has been recently handled by Nigro et al. (2023) where a reconstruction of the workpiece surface is obtained through an Iterative Closest Point (ICP) algorithm, by using a number of point clouds provided by a low-cost, off-the-shelf depth sensor. Then, the reconstructed surface is matched with a point cloud extracted from the Computer Aided Design (CAD) model of the workpiece, to have a localization of the holes and their axes. Due to the small peg-hole clearance, the accuracy of the estimate is not enough for a successful peg insertion, thus a search phase is necessary, requiring the peg's tip to slide on the surface following a trajectory described by Lissajous functions. Finally, the peg insertion is performed by imposing a compliant behavior to the robot at the peg tip level via an admittance control (Villani and De Schutter, 2008).

This work integrates deep learning based computer vision methods to improve the approach in Nigro et al. (2023). In detail, to make the ICP surface reconstruction algorithm more effective and robust, a Deep Neural Network (DNN) is adopted to filter the acquired point clouds by deleting the points of the background.

We preferred to use a DNN-based approach because traditional soft computing segmentation methods, including thresholding, region-based segmentation, and edge segmentation can be very sensitive to variations in the lighting conditions (Kaur and Kaur, 2014). Region-based segmentation is very useful when it is possible to define similarity rules, but the computational burden, in terms of time and memory, is high. These disadvantages are overcame by using a segmentation method based on deep learning, because the trained model can be robust with respect to changes in the lighting conditions and images with different quality. Moreover, it can handle the presence of complex background.

In order to reduce the hole pose estimation error, a Convolutional Neural Network (CNN) is used to detect the hole and better localize its position in the robot base frame. The use of a deep learning-based approach in place of hand crafted feature-based circle detection methods is justified by the fact that the latter are prone to errors due to illumination changes and, in general, they are sensitive to noise.

An extensive experimental campaign confirms the effectiveness of the proposed approach and highlights the improvements with respect to Nigro et al. (2023) due to the use of both the segmentation DNN and the CNN.

The main difficulties and challenges can be summarized as follows:

- The carbon fiber is a challenging material for computer vision techniques, since it is characterized by reflective surfaces.
- The workpiece is manually positioned by a human operator in the robotic workcell (see Fig. 2a). This creates uncertainties, far larger than the peg-hole clearance, on the relative pose of the workpiece with respect to the robot (see Fig. 2b).
- Since the scope is to design a system suitable for industrial contexts, low-cost vision sensors have been used in the experiments. Therefore, the acquired point clouds are not highly accurate.

The contribution of this paper is four-fold.

1. We integrate visual processing techniques, i.e., ICP, and deep learning methods, i.e., DeepLabv3+, to cope with pose uncertainties.
2. We employ an object detection module to better locate the destination target for completing the Peg-in-Hole task.
3. The laboratory setup used for the experiments accurately reproduces a real industrial setup where some modules of the proposed strategy have been implemented (see Fig. 2b).
4. A modular architecture is proposed, where the single modules are decoupled and each subsystem can be easily replaced provided that the same format of inputs and outputs is guaranteed.

The remainder of the paper is organized as follows. Section 2 presents an overview of existing methods for Peg-in-Hole tasks and semantic image segmentation. Section 3 describes in detail our strategy. Section 4 illustrates the segmentation method, Section 5 details the surface reconstruction process, Section 6 presents the object detection module, and Section 7 gives more details on the peg insertion process. Quantitative experiments are shown in Section 8. Finally, conclusions are drawn in Section 9.

Table 1
Comparison table between different approaches in industrial applications.

Method	Segmentation	Detection	Depth	Search
Alonso et al. (2020)	Yes	No	Yes	N.A.
Nigro et al. (2020)	No	Yes	Yes	No
Yin et al. (2021)	Yes	No	Yes	N.A.
Nigro et al. (2023)	No	No	Yes	Yes
Yasutomi et al. (2021)	No	Yes	No	No
Our method	Yes	Yes	Yes	Yes

2. Related work

In this section, we discuss first the Peg-in-Hole related methods and then some recent image segmentation methods. Moreover, a comprehensive table (Table 1) reports a comparison between the strategy proposed in this paper and some other recent approaches discussed in the following.

2.1. Peg-in-Hole methods

Autonomous and semi-autonomous Peg-in-Hole tasks require high positioning accuracy and high sensing capability to detect the hole. The common strategy to cope with the Peg-in-Hole problem includes two steps:

1. The search, in which the hole is localized.
2. The insertion, where the peg is aligned to the hole's axis and inserted.

Two main approaches to the search phase can be pursued, based on the exploration of the hole neighborhood or on visual feedback.

Exploration of the hole neighborhood. The methods in this category are based on moving the peg along an assigned path to cover the hole neighborhood. To use such a technique, force–torque sensors are needed. For example, in Kim et al. (2012), the outline of the hole is detected via a shape recognition algorithm based on force–torque sensor data collected during the contact with the object surface. Since the presence of force–torque sensors increases the overall cost and complexity of the system, other approaches exploit joint position sensors for estimating the contact forces, e.g., Park et al. (2013, 2017). Different search trajectories have been proposed, such as concentric circles, spiral paths (Jiang et al., 2022; Kang et al., 2022) and Lissajous curves (Nottensteiner et al., 2020). However, the search methods based on the exploration of the hole neighborhood are often time-consuming and require an accurate initial estimate of the hole position.

Visual feedback. To overcome the above cited limits, visual methods have been developed, as in Chang et al. (2011) that deals with a micro-peg-in-hole task combining different computer vision techniques: a Dynamic Position-Based Servo through Image Calibration (DPBS-IC) is used to control the gripper carrier stage, a Regional-Scanning with Edge-Fitting (RSEF) algorithm is utilized to track the needle tip, the peg and mating hole to achieve the alignment, and a Shadow-Aided Positioning (SAP) algorithm is employed for the final operation of micro-peg-in-hole assembly.

Hybrid solutions. Recently, computer vision methods are combined with both exploration and learning methods. In Triyonoputro et al. (2019) a neural network maps the distance from the peg center to the hole in the image coordinate system. In Lee et al. (2019) a deep learning method based on self supervised multi-modal representation of sensory output is proposed, while in De Magistris et al. (2018) a multi-layer perceptron network is trained on a data set including object position and interaction forces for a polyhedral pegs in contact with the holes. In Yasutomi et al. (2021) a deep neural network, trained via reinforcement learning, is used to localize holes with variable shape and surface finish in concrete wall. A combination of a learning approach for object localization and a Three-Dimensional (3D) surface reconstruction is proposed in Nigro et al. (2020) to accurately localize the holes on a workpiece, characterized by a non-flat steel surface: a CNN is in charge of detecting the holes while the 3D surface reconstruction is obtained with 3D-Digital Image Correlation (3D-DIC) (Sutton et al., 2009).

2.2. Image segmentation methods

Image segmentation aims at grouping sets of pixels based on predefined object classes (Guo et al., 2020). Instance segmentation concerns finding image regions that share similar features without knowing their content, while semantic segmentation classifies pixels taking care of understanding the region content (Gruosso et al., 2021b).

In recent years, a plethora of semantic segmentation approaches (Mo et al., 2022; Hao et al., 2020; Ulku and Akagündüz, 2022; Ren et al., 2023), including automatic feature extraction-based algorithms such as DNN (Goodfellow et al., 2016), have been developed. They are used in several different contexts, including medical image analysis (Siłka et al., 2023; Chen et al., 2023; Alalwan et al., 2021), virtual and augmented reality (Gruosso et al., 2021a, 2022), autonomous driving (Feng et al., 2021), cleaning of the point cloud in industrial applications (Yin et al., 2021; Xie et al., 2020), and robotic applications (Alonso et al., 2020).

One of the first approaches based on DNNs for pixel-wise semantic segmentation is SegNet (Badrinarayanan et al., 2017). It is a deep encoder–decoder U-shaped network (Ronneberger et al., 2015) trained end-to-end on a supervised task involving the decoder as an integral part of the network in test time. The encoder component corresponds to the first 13 VGG16 (Simonyan and Zisserman, 2015) network convolutional layers, and each of them has a corresponding decoder layer. It follows that the decoder component also consists of 13 layers. Although SegNet usually performs very well in outdoor scenes, it does not achieve the same accuracy in indoor scenes, due to the increased cluttering. Furthermore, when compared with other approaches, e.g., Noh et al. (2015) and Chen et al. (2017a), SegNet requires more hardware resources and computational time to be trained and inferred.

Recently, semi-supervised approaches for image-to-image translation problems, in particular Generative Adversarial Networks (GAN) based methods, are also used for semantic segmentation. The aim of GAN-based approaches is to augment the training data reducing the wasteful work of manual labeling. As an example, Souly et al. (2017) propose a semi-supervised GAN-based framework in which a generator provides extra training samples and the discriminator assigns a label of possible classes or marks it as fake.

An important approach for semantic segmentation is DeepLab (Chen et al., 2017a), which can tackle reduced feature resolution, objects at multiple scales, and low localization accuracy caused by invariance. In particular, reduced feature resolution is faced by removing downsampling operation from the last few DNN max pooling layers, thus obtaining feature maps with a high sampling rate that leads to inserting holes among non-zero filter taps (Holschneider et al., 1990). This convolution with the upsampling filter, called atrous convolution, allows to the recovery of full resolution feature maps with a simple bi-linear interpolation. In this way, it is possible to enlarge the field of view of the filter, which is beneficial for the number of parameters and the computational effort. To address the multiple scales objects, DeepLab uses a scheme, called Atrous Spatial Pyramid Pooling (ASPP),

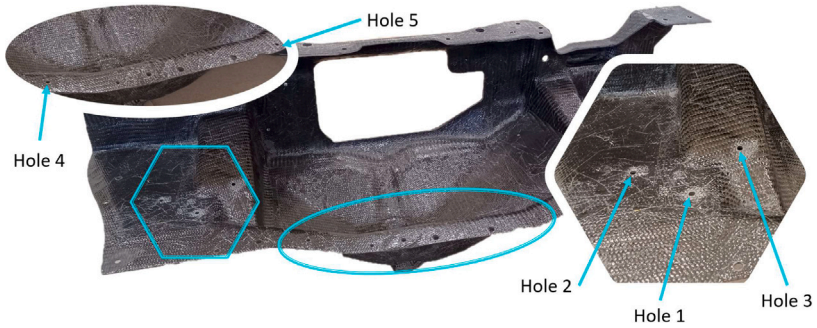


Fig. 3. Carbon fiber workpiece and detail of the holes.

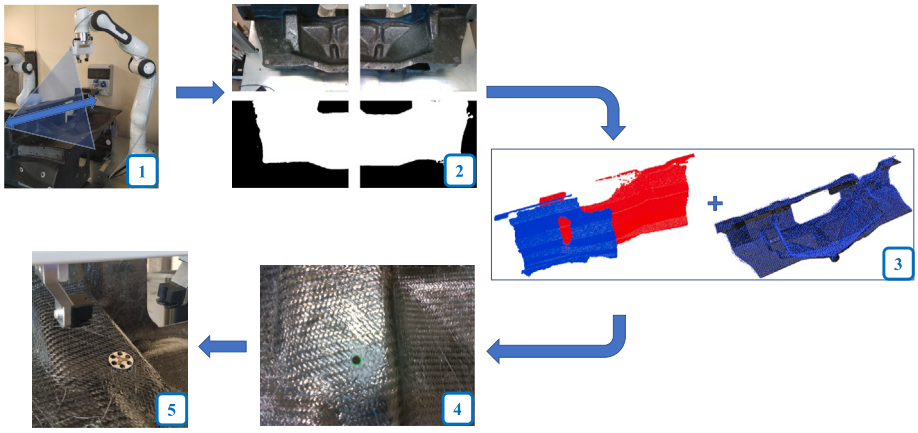


Fig. 4. Functional steps of the proposed strategy: (1) workpiece surface scanning; (2) the segmentation neural network detects the workpiece and deletes its background; (3) surface reconstruction and alignment of the reconstructed surface with the point cloud extracted from the CAD to have the initial guess estimation of the holes' position; (4) hole detection via the CNN; (5) search and insertion phase.

for resampling a feature layer at multiple rates prior to convolution. The last challenge is related to the fact that an object-centric classifier requires invariance with respect to spatial transformation. DeepLab addresses this problem using a fully-connected Conditional Random Field (CRF) (Krähenbühl and Koltun, 2011).

With its third version, called DeepLabv3+ (Chen et al., 2018), the DeepLab architecture reached the DNN state-of-the-art for semantic segmentation. It achieved impressive results on many benchmark datasets and in various research fields (Harkat et al., 2020; Wang and Liu, 2021; Wu et al., 2021; Kong et al., 2021; Gruosso et al., 2021a, 2022) surpassing among others, the previously mentioned approaches.

3. Proposed strategy

The proposed strategy is designed for handling complex 3D workpieces in the presence of small production volumes. Let assume that the workpiece is manually positioned in the robot workspace, in such a way that position uncertainties are far larger than the task tolerance. More in detail, a carbon fiber workpiece, i.e., a portion of a supercar's safety cell, has been considered. The task requires the insertion of steel bolts in 5 holes (see Fig. 3), with a peg-hole clearance below 1 mm.

Our strategy includes five functional steps, shown in Fig. 4:

1. The robot, equipped with a depth camera in an eye-in-hand configuration, spans the workspace to acquire, in N different positions, N RGB images and, through the depth sensor, N point clouds of the environment.
2. The acquired images represent the input for a segmentation neural network, which generates a binary mask that separates the workpiece from the background. The binary mask is used to filter out points belonging to the background in the acquired point clouds.
3. The N filtered point clouds, representing different parts of the workpiece surface, are aligned via an ICP registration algorithm (Chen and Medioni, 1992) to reconstruct the whole workpiece surface. The reconstructed point cloud is aligned to a reference point cloud, extracted from the CAD model of the workpiece, to have an initial guess of the holes' position on the surface and their normal unit vectors.
4. The accuracy of the hole position estimates, in the presence of small clearance, does not guarantee the peg insertion. Thus, the initial guess is used only for the initial positioning of the robot. Then, a CNN is adopted to detect the presence of the hole and identify its actual position in the robot base frame.
5. Once the hole positions and its normal unit vectors have been estimated through the CNN, the robot approaches the hole. A search phase is designed, where the peg's 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 following insertion phase, the robot is commanded to be compliant at the peg tip level by means of an admittance control (Villani and De Schutter, 2008).

It is worth noticing that the proposed architecture is modular, as can be seen in Fig. 5, and each module is independent of the others and built to be easily replaced. For example, further improvements in the neural network field could be exploited to replace the image segmentation and/or the object detection modules.

A flowchart representation, highlighting the whole process, is given in Fig. 6, while each single step is discussed in detail in the following sections.

Fig. 5. Modular architecture.

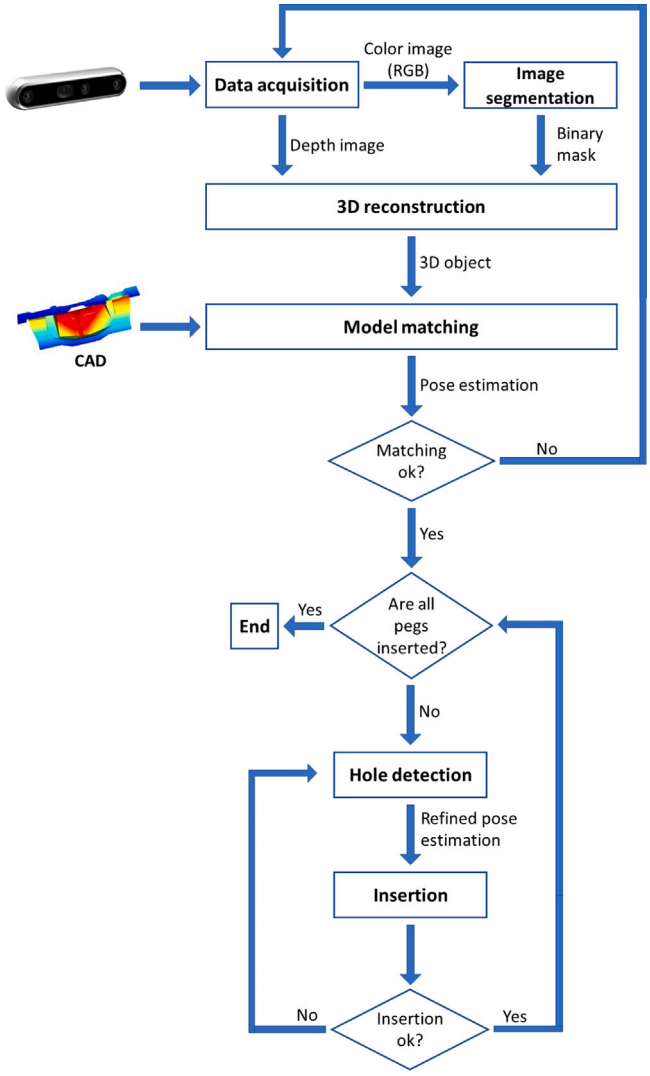


Fig. 6. Flowchart representation of the whole process.

4. Workpiece segmentation

We designed a DNN for the carbon fiber workpiece segmentation (see Fig. 7) based on DeepLabv3+ architecture. DeepLabv3+ consists of two main components. The first one is the encoder block, which extracts semantic information and low-level features from the RGB input image, gradually reducing the feature maps size. The second component is the decoder block, which is used to retrieve spatial and detailed object boundary information. The encoder includes a backbone network, followed by an ASPP module (Chen et al., 2017a) and a 1×1 convolutional layer. The ASPP module captures multi-scale context information and consists of three atrous convolutions (Papandreou et al., 2015), a 1×1 convolution, and an image pooling layer in parallel with each other. Atrous (or dilated) convolutions extend standard convolutions introducing a atrous (or dilation) rate parameter to enlarge the field of view of the convolutional filters without increasing the computational cost and the network parameters (Chen et al., 2014). We set the atrous rate of the atrous convolutions in the ASPP module to 6, 12, and 18, respectively. In light of the promising outcomes achieved in our previous works (Gruosso et al., 2021c, 2022), where fast computation time and accurate results were obtained for image segmentation tasks using partial or incomplete object images, we decided to select the Xception as backbone network with 65 layers, proposed by Chen et al. (2018) to cater specifically to the segmentation task.

The decoder block is built using convolutional and bilinear upsampling operations. In particular, the features extracted by the backbone network are given as input to a 1×1 convolution and then concatenated with the upsampled encoder output. Finally, a 3×3 convolution and a further bilinear upsampling are applied and a binary segmentation mask is obtained. Although there are alternative architectures available for semantic segmentation (Hao et al., 2020), we selected DeepLabv3+ with Xception-65 based on its proven effectiveness and availability of pre-trained weights. By leveraging the Xception-65 backbone, our network can effectively capture both local and global contextual information, leading to improved segmentation accuracy. In terms of the impact on results, the adoption of the DeepLabv3+ architecture provides several advantages. Firstly, it allows for precise boundary delineation and accurate segmentation of the carbon fiber workpiece, which is essential for our specific application. Additionally, the use of the Xception-65 backbone enhances the network's ability to capture fine-grained details and handle complex image features. This ultimately contributes to better segmentation performance and overall results.

4.1. Dataset description

To train our DNN, we collected a large and diverse semi-synthetic dataset that combines real foreground images of the carbon fiber workpiece in various positions with different background scenes. The use of synthetic and semi-synthetic data has become increasingly common due to the need for large training datasets with accurate labels (Nikolenko, 2021). Synthetic data, generated using simulation software and computer graphics techniques, provide perfect labels quickly and with minimal effort. However, to enhance realism, a preprocessing step is typically required.

In our dataset creation process, reported in Algorithm 1, we first acquired the foreground information by capturing videos using a green-screen setup. This setup involved an opaque green drape and two lights to minimize shadows. The videos were recorded at 60 Frames Per Second (FPS) using a standard RGB camera. Adobe After Effects was then used to remove the green background and extract the foreground images. To obtain accurate ground truth (labels) binary masks, alpha channel masks were saved for each frame and binarized using Otsu's global image thresholding method (Otsu, 1979).

The collection of background scenes and image compositing constituted the second step in our dataset creation process. Indoor videos were recorded at 30 FPS using RGB cameras with different resolutions.

After recording the indoor videos, we utilized an automatic procedure to composite the foreground images with the background scenes.

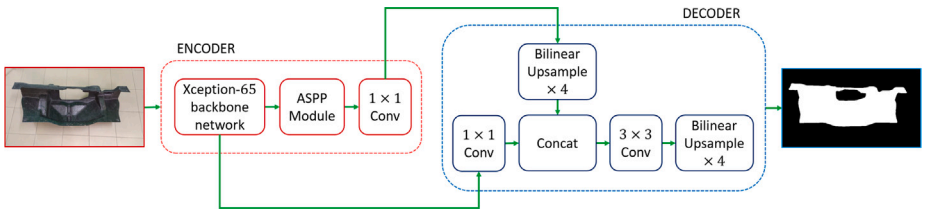


Fig. 7. Our network model based on the DeepLabv3+ encoder–decoder architecture (Chen et al., 2018). We chose the Xception-65 model as the backbone network, which allows extracting low-level features that are passed to the decoder. The input is an RGB image showing the carbon fiber workpiece, while the output is the segmentation binary mask.

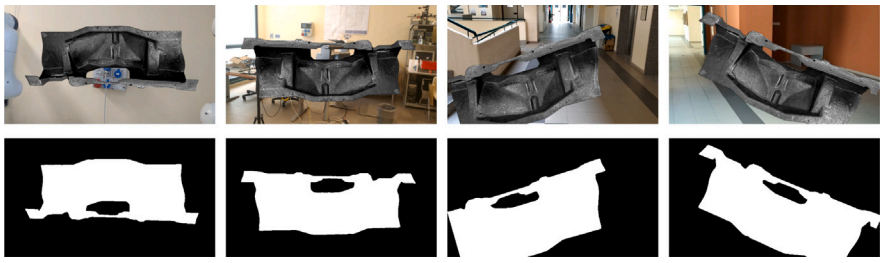


Fig. 8. Some input images (first row) and labels (second row) from our semi-synthetic dataset.

Algorithm 1 Create Dataset

Require: V_f, V_b, $D_{(c,b)}$ ▷ foreground and background videos, empty dataset
$\alpha \leftarrow \text{saveMask}(AAE(V_f))$ ▷ Save α channel masks using Adobe After Effect
$b \leftarrow \text{OtsuMethod}(\alpha)$ ▷ Otsu's thresholding binarization
for $s \leftarrow 1$ to $\text{minSeconds}(V_b, V_g)$ **do**
  $B_g, F_g, \alpha_s, b_s \leftarrow \text{TwoFourAlpha}(V_b, V_f, \alpha, s, b)$ ▷ frames, α, and b for second s
  $F_g, \alpha_s, b_s \leftarrow \text{randomTransforms}(F_g, \alpha_s, b_s)$ ▷ Apply random transforms
  for $i \leftarrow 1$ to 8 **do** ▷ Apply blending equation, bilateral filter, and resizing
    $j \leftarrow \lfloor (i+1)/2 \rfloor$
    $k \leftarrow (i-1)\%2 + 1$
    $C_i \leftarrow \alpha_s(j) \times F_g(j) + (1 - \alpha_s(j)) \times B_g(k)$
    $D.add(\text{resize}(\text{blFilter}(C_i), 360, 640), b_s(j))$ ▷ add a composite image and its corresponding binarized mask to the dataset
return $\text{dataAugmentation}(D)$ ▷ Perform data augmentation

This procedure involved selecting four foreground images per second and two background frames per second, with a sampling step of 15 to ensure a diverse set of frames. Each selected foreground image was composited with all background scenes using the alpha channel blending equation:

$$C = \alpha \times F + (1 - \alpha) \times B, \tag{1}$$

where C represents the composite image, F is the foreground image, B is the background, and α is the alpha channel.

To increase the variability of the dataset and improve network generalization, several random transformations were applied to each foreground image before compositing. These transformations included rotation within the range of $[-30, 30]$ degrees, horizontal flipping, and vertical flipping. The corresponding labels and alpha channels were transformed accordingly. Additionally, a bilateral filter was randomly applied to the composite images to reduce noise and enhance the preservation of foreground object edges (Capece et al., 2019; Gruosso et al., 2021b).

To speed up network training, all images and labels were resized to 360 × 640 pixels, and data augmentation was used by randomly left/right mirroring training data on the fly during training. Some examples of training images and the corresponding labels are shown in Fig. 8.

Although there are alternative options available for dataset selection (Xia et al., 2019; Garcia-Garcia et al., 2018), we adopting this approach based on several motivations:

1. The use of a semi-synthetic dataset allows us to combine real and computer-generated information, resulting in a dataset that better approximates real-world scenarios. This enables our DNN to learn from a more diverse range of data and generalize well to unseen real-world images.
2. By incorporating real foreground images of the carbon fiber workpiece captured under controlled conditions, we ensure that the dataset reflects the characteristics and variations present in the actual objects. This helps our DNN to effectively learn and capture the specific features relevant to the segmentation task.
3. The integration of different background images further enhances the dataset's diversity, ensuring that our DNN can handle various environmental contexts and backgrounds commonly encountered in practice.
4. The utilization of a large dataset size provides ample training samples for the DNN, enabling it to learn robust and discriminative features. This contributes to better generalization and improved segmentation performance.

The effect of using this large and varied semi-synthetic dataset on the results is significant. By training on a dataset that closely resembles the target domain, our DNN can better adapt to real-world images, leading to improved segmentation accuracy. The inclusion of diverse backgrounds also helps the model handle challenging scenarios and enhances its ability to segment the carbon fiber workpiece accurately.

4.2. Training details

Our DNN was trained using the above described segmentation dataset. Similar to Chen et al. (2017b) and Gruosso et al. (2022), we used the Stochastic Gradient Descent with Momentum (SGDM) optimization algorithm with polynomial learning rate policy, which proved to be more effective and with faster convergence than other learning rate update policies (Liu et al., 2015; Chen et al., 2017a).

SGDM is a widely used and effective optimization algorithm for training deep neural networks. It combines the benefits of Stochastic Gradient Descent (SGD) with an additional momentum term. This momentum term helps accelerate convergence by accumulating the past gradients and dampening the oscillations in the parameter updates.

By incorporating momentum, SGDM allows the optimization process to navigate the loss landscape more efficiently, potentially leading to faster convergence and improved generalization. One of the advantages of SGDM over alternative algorithms, such as SGD or Adam (Ruder, 2016; Kingma and Ba, 2014), is its ability to handle noisy or sparse gradients more effectively. This is particularly beneficial when working with large-scale datasets or complex architectures, where gradient noise and sparsity can pose challenges. SGDM's momentum term helps smooth out the noisy gradients and enables the optimizer to escape shallow local minima, leading to improved convergence and better generalization performance. SGDM's ability to accelerate convergence and handle noisy gradients leads to faster training and better overall performance of our DNN.

In our approach, the value of the learning rate is modified according to the following formula:

$$\alpha_t = \alpha_0 \times \left(1 - \frac{t}{T}\right)^p, \tag{2}$$

where α_t is the learning rate at the current iteration step t, α_0 is the base learning rate set to 0.0001 for our training phase, T is the total number of iterations set to 50K, and p is the power value set to 0.9.

Although this equation is a popular choice, there exist alternative strategies for adjusting the learning rate during training. Some feasible alternatives include fixed learning rates, step decay, exponential decay, and adaptive learning rates algorithms such as Adam (Kingma and Ba, 2014) or Adagrad (Duchi et al., 2011). The advantages of the adopted equation lie in its simplicity and flexibility. By gradually decreasing the learning rate over time, it allows for finer adjustments during later stages of training when approaching convergence. This can help to avoid overshooting and enable the model to settle into a more optimal solution. Additionally, the parameterization of the equation through the base learning rate, the current iteration, and the total number of iterations provide control over the decay rate. The effect of this equation on the results can vary depending on the specific dataset and model architecture. In general, a decaying learning rate can enhance convergence and prevent oscillations, leading to more stable and accurate results. However, the specific choice of decay function and its hyperparameters can impact training dynamics and the final performance.

We set the momentum γ of the SGDM algorithm to 0.9, the batch size to 4, and used cross-entropy as a loss function since it is the most used and efficient in the case of semantic segmentation (Jadon, 2020; Gruosso et al., 2021b, 2022).

Since training a DNN from scratch requires a copious amount of data and resources in terms of memory, computation, and time, starting from a pre-trained models on a large dataset is usually recommended. Therefore, we used weights pre-trained on the ImageNet (Russakovsky et al., 2015) and MS-COCO (Lin et al., 2014) datasets.[1] ImageNet is a huge and generic dataset employed for classifying and detecting 1000 different object categories, while MS-COCO is smaller and used for classification, detection, and segmentation of 80 classes. For this reason, a segmentation network pre-trained on both datasets may benefit more from the learned features than using only a general ImageNet pre-training (Chen et al., 2018). The DNN training was performed on a desktop computer equipped with an Intel Core i7-3rd generation CPU, 16 GB RAM, and an Nvidia Titan Xp GPU with 12 GB of memory.

5. Surface reconstruction

To reconstruct the workpiece surface, the robot moves the camera and scans its workspace acquiring N different point clouds. The acquired point clouds are aligned using an ICP registration algorithm (Choi et al., 2015).

[1] The pre-trained weights are publicly available on the DeepLab project page: https://github.com/tensorflow/models/tree/master/research/deeplab.

The basic idea of the registration algorithms is to find a set of rigid transformations, $\boldsymbol{T}_i$, that allows to align, in a global coordinate frame $\mathcal{F}_g$, the N acquired point clouds $\mathcal{P}_i$ ($i = 1, \dots, N$). In particular, in our case, the global coordinate frame is chosen coincident with the reference frame in which the first point cloud $\mathcal{P}_1$ is acquired. To achieve this, rigid transformations aligning each couple of consecutive point clouds are computed.

For the ith point cloud, the transformation matrix $\boldsymbol{T}^i_{i+1}$ that transforms the generic point $\boldsymbol{p}^{i+1}_j \in \mathcal{P}_{i+1}$ from its local reference frame to $\mathcal{F}_i$, is initialized to $\boldsymbol{T}^i_{i+1}(0)$, e.g., equal to the identity matrix, and then iteratively refined via the ICP algorithm. In this work, the point-to-plane ICP algorithm is adopted since it converges faster than the point-to-point ICP algorithm (Rusinkiewicz and Levoy, 2001).

To the aim, for the point cloud $\mathcal{P}_{i+1}$, m control points are selected with the corresponding points in $\mathcal{P}_i$, in such a way to define the correspondence set $\mathcal{K} = \{(\boldsymbol{p}^i_1, \boldsymbol{p}^{i+1}_1), \dots, (\boldsymbol{p}^i_m, \boldsymbol{p}^{i+1}_m)\}$. Then at the kth iteration, the transformation matrix $\boldsymbol{T}^i_{i+1}(k)$ is obtained by minimizing the cost function C

$$C(\boldsymbol{T}^i_{i+1}(k)) = \sum_{\boldsymbol{p}^i_j, \boldsymbol{p}^{i+1}_j \in \mathcal{K}} \left((\tilde{\boldsymbol{p}}^i_j - \boldsymbol{T}^i_{i+1}(k)\, \tilde{\boldsymbol{p}}^{i+1}_j)\, \tilde{\boldsymbol{n}}^i_{i+1} \right)^2, \tag{3}$$

where the symbol $\tilde{}$ denotes the homogeneous representation of the coordinate vectors (Siciliano et al., 2009), and $\boldsymbol{n}^i_{i+1}$ is the unit vector normal to the surface represented by the point cloud $\mathcal{P}_{i+1}$ expressed in the reference frame of $\mathcal{P}_i$. The dimension of the correspondence set is usually a trade-off between the computation time and the accuracy: the more points are selected the more the estimation of $\boldsymbol{T}^i_{i+1}$ is accurate and larger computation time is required.

By applying the above described algorithm to all the acquired point clouds, a reconstruction of the surface $\mathcal{Q}_r$, i.e., a point cloud representing the whole workpiece surface in the coordinate frame $\mathcal{F}_g$, is determined.

6. Hole detection

Once the reconstructed point cloud $\mathcal{Q}_r$ is obtained, a first estimate of the hole positions and tilts is computed by comparing $\mathcal{Q}_r$ with a known one, $\mathcal{Q}$, extracted from the CAD model. Again, an ICP algorithm is adopted to align the two point clouds: firstly a Random Sample Consensus (RANSAC) algorithm is used to determine the corresponding points of the two point clouds, then a procedure similar to the one described in Section 5 is carried out.

The hole positions, $\boldsymbol{p}_{h_i}$, and their axes, i.e., the normal unit vector to the surface, $\boldsymbol{n}_{h_i}$, are assumed known on the point cloud $\mathcal{Q}$, thus, thanks to the alignment procedure it is possible to estimate them in the reconstructed point cloud $\mathcal{Q}_r$ and localize them in the coordinate frame $\mathcal{F}_g$. Finally, by adopting a camera calibration process (Tsai et al., 1989) the camera-end effector transformation is determined and the hole positions can be expressed in the robot base coordinate frame.

Such an initial estimate is likely to be affected by errors, due to the reconstruction and calibration processes. Therefore, before the insertion, a CNN is exploited to detect with better accuracy the holes on the workpiece surface. The detection is performed by using the model built in Nigro et al. (2020) through the YOLOv3 supervised object detection architecture (Redmon and Farhadi, 2018).

The detector was created by considering 108 images (as training set) of size 480 × 640 of a non-flat steel surface. Some images of the dataset are shown in Fig. 9. The details about the training and the CNN performance can be found in Nigro et al. (2020).

As it can be seen in Fig. 9, the workpiece used to train the network was different from the one used in this work. In particular, the one in Fig. 9 has a white surface with stochastic black speckle pattern, in which the holes appear as black ellipses, while the one considered for the experiments is a dark carbon fiber workpiece, and the surface is characterized by high reflectivity. Despite this, the detection performance on the carbon fiber object are satisfactory. The hole detector runs with a processing time of about 1.05 s per image on CPU.

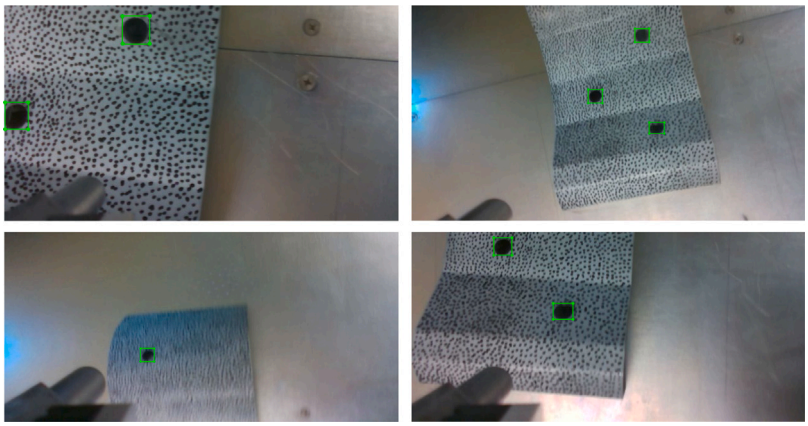


Fig. 9. Examples of images used for training the hole detector.

7. Search and insertion

Once the estimates of the hole positions, $\hat{\boldsymbol{p}}_{h_i}$, and its normal unit vector, $\hat{\boldsymbol{n}}_{h_i}$, are determined more accurately with the hole detector, three phases are executed to perform the Peg-in-Hole task:

1. *Approach phase*: where the robot moves the peg close to the hole and aligns the peg axis to $\hat{\boldsymbol{n}}_{h_i}$.
2. *Search phase*: where an exploration of the neighborhood of the estimated hole position, $\hat{\boldsymbol{p}}_{h_i}$, is exploited to compensate for any estimation errors.
3. *Insertion phase*: where the peg is inserted in the hole.

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

The *alignment task* is aimed at aligning the $\boldsymbol{z}_e$ axis of the end-effector reference frame to the hole axis $\hat{\boldsymbol{n}}_{h_i}$. The Jacobian matrix relative to this task is

$$\boldsymbol{J}_a = 2(\hat{\boldsymbol{n}}_{h_i} - \boldsymbol{z}_e)^{\mathrm{T}} \boldsymbol{S}(\boldsymbol{z}_e) \boldsymbol{J}_O(\boldsymbol{q}), \tag{4}$$

where $\boldsymbol{S}(\cdot)$ is the skew symmetric matrix operator performing the cross product and $\boldsymbol{J}_O(\boldsymbol{q})$ is the orientation part of the geometric Jacobian matrix of the robot (Siciliano et al., 2009).

The *position tracking* task is in charge of moving the peg tip in the neighborhood of the hole. The task function is the position of the peg, $\boldsymbol{p}_e$, and the task Jacobian is the positional part of the geometric Jacobian matrix of the robot, $\boldsymbol{J}_P(\boldsymbol{q})$.

The joint reference velocities are computed as follows (Antonelli, 2009)

$$\dot{\boldsymbol{q}}_r = \boldsymbol{J}_a^{\dagger}(-k((\boldsymbol{z}_e - \hat{\boldsymbol{n}}_{h_i})^{\mathrm{T}}(\boldsymbol{z}_e - \hat{\boldsymbol{n}}_{h_i}))) + \boldsymbol{J}_P^{\dagger}(\dot{\boldsymbol{p}}_{e,r} + \boldsymbol{K}(\boldsymbol{p}_{e,r} - \boldsymbol{p}_e)), \tag{5}$$

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 $k \in \mathbb{R}$ and $\boldsymbol{K} \in \mathbb{R}^{3\times3}$ are, respectively, a positive definite scalar and matrix gain.

To compensate the estimation errors, a search phase is exploited. It starts when a contact between the surface and the peg tip is experienced. Since most of the collaborative robots are not equipped with a wrist-mounted force–torque sensor, the contact wrench on the peg tip can be computed by recurring to an observer based on the generalized momentum, which exploits the measures of the joint torques (De Luca and Mattone, 2005).

Once the contact has been detected, the peg tip is moved along an exploring path, described by a Lissajous function

$$\begin{aligned} x &= a_x sin(\omega_x(t - t_c)), \\ y &= a_y sin(\omega_y(t - t_c)), \end{aligned} \tag{6}$$

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.

When the hole is detected and the peg tip is inserted of about 1 mm, the search phase ends. During the search and insertion phases, since there is physical interaction between the robot and workpiece, to keep bounded the interaction wrench, the manipulator has been commanded to be compliant by implementing an admittance control scheme (Nigro et al., 2023).

8. Experimental results

The experimental setup for validating the proposed strategy consists of a collaborative robot Franka Emika Panda and an Intel Realsense D435 RGB-D camera. The camera has been preliminarily calibrated by using 16 images of a 2D checkerboard flat pattern via the Tsai method implemented in the VISP library (Marchand et al., 2005). The vision system runs on a workstation equipped with the Ubuntu 18.04 LTS operating system, with a real-time kernel, running on an Intel Xeon 3.7 GHz CPU with 32 GB RAM; the `librealsense2` library is used for acquiring the camera data. The `Open3D` library (Zhou et al., 2018) is used both for the point cloud registration and for the overlapping of the reconstructed surface with the point cloud extracted by the CAD model. In the first case the `multiway registration` algorithm is exploited, while in the second one the `global registration` algorithm is used. Table 2 summarizes the main hardware and software components adopted for the experiments.

To have statistically significant results, 51 insertion tests have been carried out by randomly positioning the workpiece in the robot workspace. The effect of the segmentation network and the CNN have been evaluated by considering an ablation test. In particular, for each insertion, we carried out two experiments. In the first one, the surface reconstruction is performed, according to the method proposed in Nigro et al. (2023), without using the DeepLabv3+ network, i.e., the ICP algorithm is fed by the point cloud directly acquired by the camera. Moreover, in these experiments the hole localization does not exploit the CNN while it is performed only by comparing the reconstructed surface with the CAD model. In the second set of experiments, the full strategy described in Sections 3–7 is pursued.

In each test, the robot initially scans the workpiece moving along a planned path, designed to include the nominal position of the workpiece in the camera field of view with a certain tolerance to take into account the positioning errors. $N = 8$ different point clouds are acquired via the depth sensor to reconstruct the workpiece surface. The value of N is chosen to have a wide overlap between two consecutive point clouds. To test the robustness of the proposed methods, a set of

Table 2
Hardware and development tools.

Robot	Franka Emika Panda
Camera	Intel Realsense D435
Robot workstation CPU	Intel Xeon 3.7 GHz CPU with 32 GB RAM
Robot workstation operating system	Ubuntu 18.04 LTS real-time
DNN workstation CPU	Intel Core i7-3rd generation CPU with 16 GB RAM
DNN workstation GPU	Nvidia Titan Xp GPU with 12 GB memory
Robot library	libfranka 0.7.1 C++
Camera library	librealsense2 2.37.0 C++
Registration library	Open3D 0.12.0 Python
Python version	3.6.9

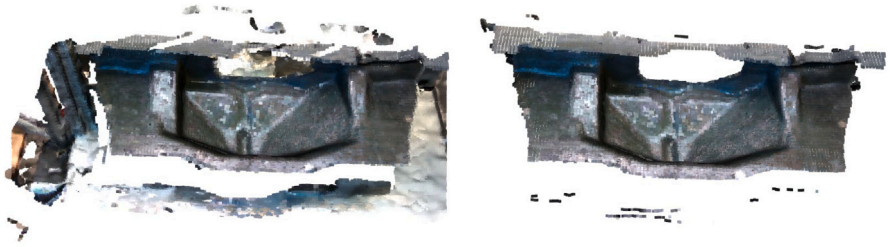


Fig. 10. Workpiece surface reconstruction without (a) and with (b) the application of the segmentation network.

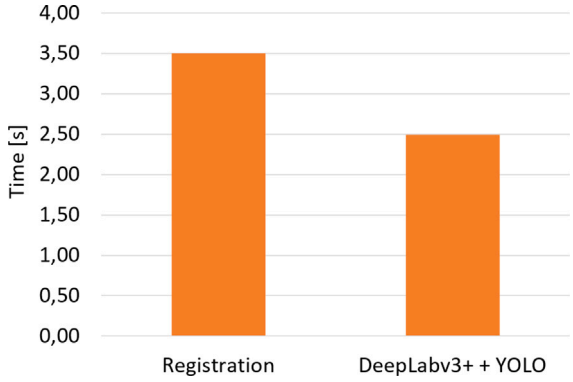


Fig. 11. Registration time by using the multiway algorithm without (left) and with (right) the application of the segmentation network.

experiments has been intentionally carried out with a high initial positioning error. The Vicon tracking system[2] has been used to accurately measure the workpiece position.

Regarding the registration, both the tested strategies allow to obtain the surface reconstruction in all tests. However, as can be noted in Fig. 10, the use of DeepLabv3+ allows to remove the background from the acquired point clouds and to have a better reconstruction of the surface. Moreover, the use of the network reduces the time required by the multiway registration algorithm of about 28.8%, as shown in Fig. 11.

When the norm of the initial positioning error is greater than 8 cm, the overlapping with the CAD model fails in the absence of the segmentation network. This is mainly due to the presence of high number of points belonging to the background in the point cloud. On the contrary, when DeepLabv3+ is adopted, the overlapping has been successfully executed in all the tests, also when the norm of the positioning error is 26 cm. Larger errors have not been tested due to the setup geometry. Fig. 12 shows a case in which the workpiece is positioned at the border of the camera field of view: the surface is correctly reconstructed even if not completely, while the overlap with the CAD is successful only when DeepLabv3+ is adopted.

Tests with large orientation errors have been conducted, but do not show significant discrepancies between the two methods.

To test the network generalization capabilities, experiments involving different carbon fiber workpieces have been conducted. Despite the surface of the used workpieces is highly reflective and they have not been included in the training dataset, the segmentation mask correctly matches the object, as shown in Fig. 13.

Regarding the performance of the task execution, two indices have been considered: (1) the error between the estimated and actual hole's position and (2) the duration of the search phase before the insertion. Fig. 14 shows the estimation obtained by simply comparing the reconstructed surface with the point cloud extracted by the CAD model and that obtained by using the CNN approach.

The adoption of the CNN allows to strongly reduce the error as demonstrated by the results shown in Fig. 15, where the mean errors for the 51 tests are compared with and without the CNN. The error is computed as the distance between the estimated position and the actual one computed on the hole's plane. The adoption of the CNN for estimating the hole position allows to halve the mean error on the five holes (from 5.3 mm to 2.3 mm).

Once the hole position has been estimated, a search phase, in which the peg explores the neighborhood of the estimated position by following a path on the surface planned via Lissajous functions, is necessary since the clearance between the hole and peg is very small (below 1 mm).

The Lissajous magnitude and frequency parameters have been set to reach a trade-off between the coverage surface and the amplitude of the search area. Regarding the control parameters, they have been set in such a way to make the robot rigid along the peg axis and compliant along the other directions. The parameters adopted during the tests are reported in Table 3.

The duration of the search phase is strictly related to the estimation error, thus it is not surprising that the adoption of the CNN allows to

Table 3
Search and insertion parameters.

Parameter	$\boldsymbol{K}$	k	a_x	a_y	ω_x	ω_y
Value	$150 \cdot \boldsymbol{I}_3$	5	0.0175	0.0245	2.5	3.5

[2] https://www.vicon.com/.

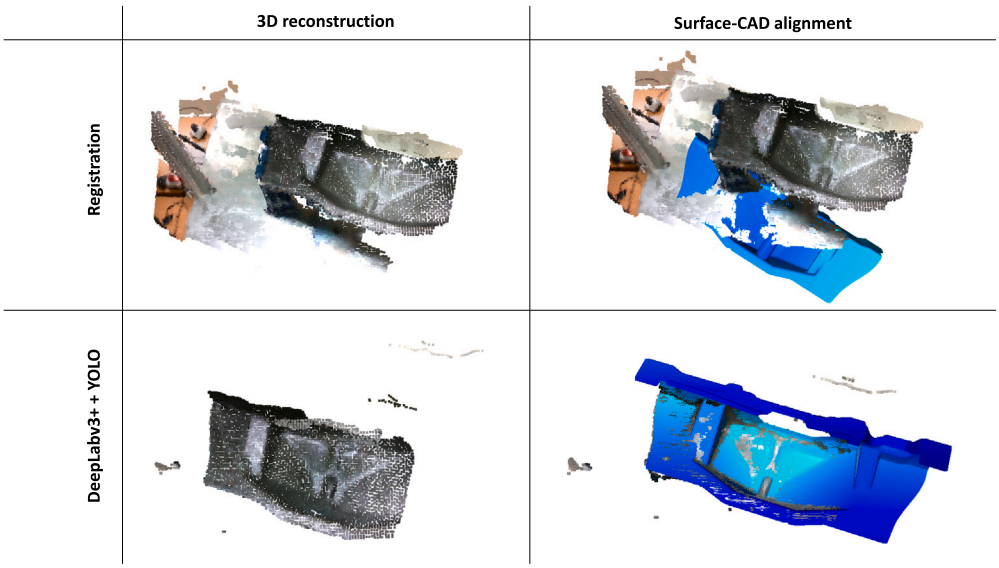


Fig. 12. Test on surface overlap with (bottom) and without (top) DeepLabv3+.

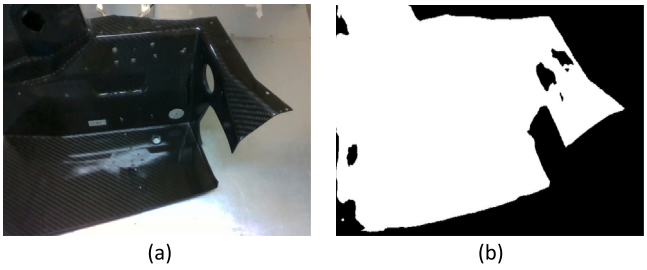


Fig. 13. Reflective carbon fiber workpiece used to test the segmentation network: despite the high reflectivity, the segmentation mask correctly matches the object.

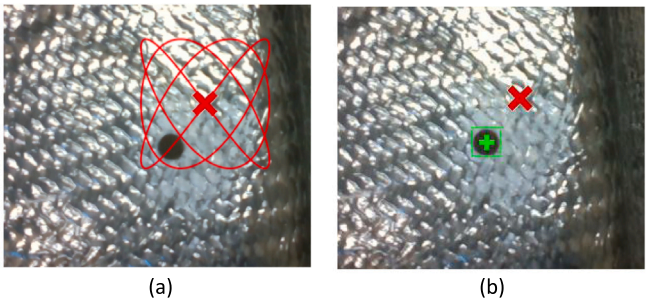


Fig. 14. Hole's position estimation without (a) and with (b) the use of the CNN. On the right, the green square is the bounding box of the CNN and the green cross is its center. The red cross represents the initial guess estimation.

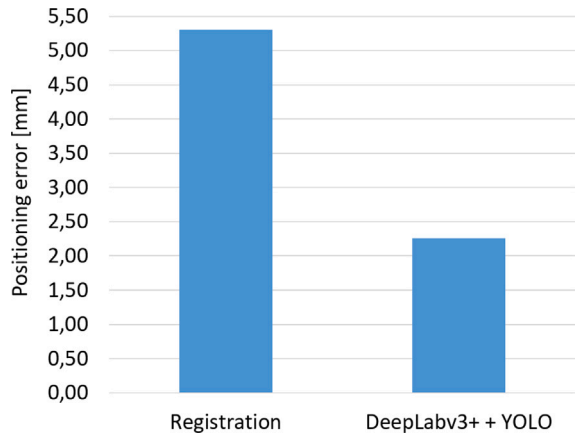


Fig. 15. Mean hole's estimation error without (left) and with (right) the use of the CNN.

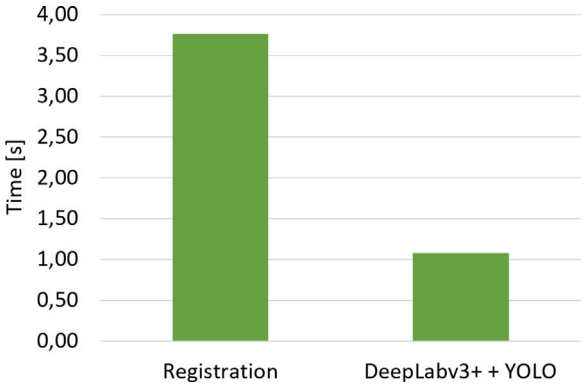


Fig. 16. Search phase duration without (left) and with (right) the use of the CNN.

reduce the search time of about 71.3%. In Fig. 16 the mean duration of the search time is shown in both cases.

Finally, Fig. 17 reports the success rate computed on the 51 tests. More in detail, it can be viewed that, the segmentation network as well as the adoption of the CNN, allows to obtain a success rate of 99.0%, i.e., only in one attempt of 103 the insertion failed. In the absence of the neural networks, i.e., by using only registration methods, the success rate decreases to 84.5%.

9. Conclusions

In this paper, we have presented a method for accomplishing a Peg-in-Hole task on a carbon fiber workpiece using a robot manipulator in a real-world industrial scenario. The workpiece is mounted on a cart and manually positioned near the robot by a human operator, thus creating uncertainty on its relative position with respect to the robot.

To accomplish the task, a three step process is proposed. The first step exploits a 3D reconstruction of the workpiece via an ICP

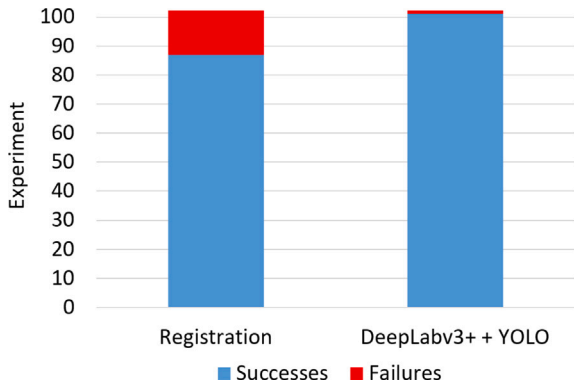


Fig. 17. Success rate both in the absence (left) and in the presence (right) of the neural networks.

registration algorithm. More in detail, a semantic image segmentation neural network has been adopted to remove the background of the scene leading to an improvement of the computational time for the registration. The second step includes the estimation of the hole position on the workpiece surface. The main contribution, in this stage, is the adoption of a CNN to reduce the estimation errors. In fact, hand crafted feature-based circle detection methods are prone to illumination changes and, in general, sensitive to noise. The proposed CNN-based approach demonstrated in our experiments to be robust to variation in the light conditions of the operational environment. Finally, in the third step, the peg is inserted through a fine search process in which the peg tip slides on the surface. We have experimented our approach in two different setting: our laboratory and a real-world factory, obtaining similar good detection results.

Despite the used techniques are not novel, integration of advanced vision methods and AI applications in an industrial context is an extremely relevant pillar on which is based the Industry 4.0 paradigm. The presented approach has been validated in our laboratory, but the setup accurately reproduces a real industrial setup. Quantitative experimental results, including an ablation test, demonstrate the effectiveness of the proposed approach and how the integration of AI techniques improves the success rate.

As a future work, we intend to extend our approach to manipulator mounted on a mobile base. Moreover, an additional module could be inserted in the proposed architecture to evaluate the quality level of the workpiece at hand by detecting possible manufacturing errors.

10. Available materials

We have shared our trained model and source code through the following link http://graphics.unibas.it:8080/share.cgi?ssid=0wSaJkQ. Furthermore, we shared the training and test dataset through this link http://graphics.unibas.it:8080/share.cgi?ssid=0G9ea07. A video demo of our work is available at https://youtu.be/AXJlpBvFuoU.

CRediT authorship contribution statement

Monica Sileo: Conceptualization, Collected the data, Performed the analysis, Software development, Experimental Validation, Writing – original draft, Writing – review & editing. **Nicola Capece:** Conceptualization, Performed the analysis, Writing – review & editing, Supervision. **Monica Gruosso:** Conceptualization, Collected the data, Performed the analysis, Software development, Experimental Validation, Writing – original draft. **Michelangelo Nigro:** Conceptualization, Software development, Experimental Validation, Writing – original draft. **Domenico D. Bloisi:** Conceptualization, Performed the analysis, Writing – original draft, Writing – review & editing, Supervision, Funding acquisition. **Francesco Pierri:** Conceptualization, Performed the analysis, Writing – original draft, Writing – review & editing, Supervision, Funding acquisition. **Ugo Erra:** Conceptualization, Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The link with the data and part of the code is reported in the paper.

Acknowledgments

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

Appendix A. Abbreviations

The following abbreviations are used in this manuscript:

3D	Three-Dimensional
3D-DIC	3D-Digital Image Correlation
AI	Artificial Intelligence
ASPP	Atrous Spatial Pyramid Pooling
CAD	Computer Aided Design
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CRF	Conditional Random Field
DNN	Deep Neural Network
DOF	Degrees of Freedom
DPBS-IC	Dynamic Position-Based Servo through Image Calibration
FPS	Frames Per Second
GAN	Generative Adversarial Networks
GPU	Graphics Processing Unit
ICP	Iterative Closest Point
IoT	Internet of Things
RAM	Random Access Memory
RANSAC	Random Sample Consensus
RGB	Red, Green, and Blue
RSEF	Regional-Scanning with Edge-Fitting
SAP	Shadow-Aided Positioning
SGD	Stochastic Gradient Descent
SGDM	Stochastic Gradient Descent with Momentum

Appendix B. List of symbols

The following symbols are used in this manuscript:

C	Composite image
α	Alpha channel
F	Foreground image
B	Background image
α_t	Learning rate at iteration step t
α_0	Base learning rate
T	Total number of iterations in training process
γ	Momentum of the SGDM algorithm
$\mathcal{F}_g$	Global reference frame
N	Number of acquired point clouds
$\mathcal{P}_i$	ith acquired point cloud

$\boldsymbol{T}_{i+1}^{i}$	Rigid transformation between the reference frame of $\mathcal{P}_{i+1}$ and the reference frame of $\mathcal{P}_i$
$\boldsymbol{p}_j^i$	jth generic point expressed in the reference frame of $\mathcal{P}_i$
$\mathcal{K}$	Set of correspondence points
$\mathcal{C}$	Cost function for ICP algorithm
$\boldsymbol{n}_{i+1}^{i}$	Unit vector normal to the surface represented by the point cloud $\mathcal{P}_{i+1}$ expressed in the reference frame of $\mathcal{P}_i$
$\mathcal{Q}_r$	Workpiece reconstructed point cloud
$\mathcal{Q}$	Workpiece known point cloud
$\boldsymbol{p}_{h_i}$	Position of the ith hole
$\boldsymbol{n}_{h_i}$	Normal unit vector of the ith hole
$\hat{\boldsymbol{p}}_{h_i}$	Estimates position of the ith hole
$\hat{\boldsymbol{n}}_{h_i}$	Estimates normal unit vector of the ith hole
$\boldsymbol{z}_e$	**z** axis of the end-effector reference frame
$\boldsymbol{q}$	Vector of the joint positions
$\dot{\boldsymbol{q}}_r$	Vector of the joint reference velocities
$\boldsymbol{J}_P(\boldsymbol{q})$	Positional part of the geometric Jacobian matrix of the robot
$\boldsymbol{J}_O(\boldsymbol{q})$	Orientation part of the geometric Jacobian matrix of the robot
$\boldsymbol{J}_a$	Jacobian matrix relative to the alignment task
$(\cdot)^{\dagger}$	Right pseudo-inverse of a matrix
$\boldsymbol{S}(\cdot)$	Skew symmetric matrix operator
$\boldsymbol{p}_e$	Position of the peg
$\boldsymbol{p}_{e,r}$	Reference position of the peg
$\dot{\boldsymbol{p}}_{e,r}$	Reference linear velocity of the peg
k	Positive definite scalar gain
$\boldsymbol{K}$	Positive definite matrix gain
x, y	Value of the Lissajous function for the **x** and **y** axes
a_x, a_y	Sine wave amplitudes of the Lissajous function
ω_x, ω_y	Sine wave frequencies of the Lissajous function
t_c	Time instant when the contact with the surface starts

References

Afan, H.A., Osman, A.I.A., Essam, Y., Ahmed, A.N., Huang, Y.F., Kisi, O., Sherif, M., Sefelnasr, A., wing Chau, K., El-Shafie, A., 2021. Modeling the fluctuations of groundwater level by employing ensemble deep learning techniques. Eng. Appl. Comput. Fluid Mech. 15 (1), 1420–1439. http://dx.doi.org/10.1080/19942060.2021.1974093, arXiv:https://doi.org/10.1080/19942060.2021.1974093.

Alalwan, N., Abozeid, A., ElHabshy, A.A., Alzahrani, A., 2021. Efficient 3D deep learning model for medical image semantic segmentation. Alex. Eng, J. 60 (1), 1231–1239. http://dx.doi.org/10.1016/j.aej.2020.10.046, URL https://www.sciencedirect.com/science/article/pii/S1110016820305639.

Alonso, I., Riazuelo, L., Murillo, A.C., 2020. MiniNet: An efficient semantic segmentation ConvNet for real-time robotic applications. IEEE Trans. Robot. 36 (4), 1340–1347. http://dx.doi.org/10.1109/TRO.2020.2974099.

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

Badrinarayanan, V., Kendall, A., Cipolla, R., 2017. SegNet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 39 (12), 2481–2495. http://dx.doi.org/10.1109/TPAMI.2016.2644615.

Banan, A., Nasiri, A., Taheri-Garavand, A., 2020. Deep learning-based appearance features extraction for automated carp species identification. Aquac. Eng. 89, 102053. http://dx.doi.org/10.1016/j.aquaeng.2020.102053, URL https://www.sciencedirect.com/science/article/pii/S0144860919302195.

Capece, N., Banterle, F., Cignoni, P., Ganovelli, F., Scopigno, R., Erra, U., 2019. Deepflash: Turning a flash selfie into a studio portrait. Signal Process., Image Commun. 77, 28–39.

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

Chen, J., Guo, Z., Xu, X., Zhang, L.-b., Teng, Y., Chen, Y., Woźniak, M., Wang, W., 2023. A robust deep learning framework based on spectrograms for heart sound classification. IEEE/ACM Trans. Comput. Biol. Bioinform. 1–12. http://dx.doi.org/10.1109/TCBB.2023.3247433.

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

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L., 2014. Semantic image segmentation with deep convolutional nets and fully connected crfs. arXiv preprint arXiv:1412.7062.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L., 2017a. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE Trans. Pattern Anal. Mach. Intell. 40 (4), 834–848.

Chen, L.-C., Papandreou, G., Schroff, F., Adam, H., 2017b. Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587.

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H., 2018. Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 801–818.

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

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

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

Duchi, J., Hazan, E., Singer, Y., 2011. Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res. 12 (null), 2121–2159.

Fan, Y., Xu, K., Wu, H., Zheng, Y., Tao, B., 2020. Spatiotemporal modeling for nonlinear distributed thermal processes based on KL decomposition, MLP and LSTM network. IEEE Access 8, 25111–25121. http://dx.doi.org/10.1109/ACCESS.2020.2970836.

Feng, D., Haase-Schütz, C., Rosenbaum, L., Hertlein, H., Gläser, C., Timm, F., Wiesbeck, W., Dietmayer, K., 2021. Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. IEEE Trans. Intell. Transp. Syst. 22 (3), 1341–1360. http://dx.doi.org/10.1109/TITS.2020.2972974.

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., Martinez-Gonzalez, P., Garcia-Rodriguez, J., 2018. A survey on deep learning techniques for image and video semantic segmentation. Appl. Soft Comput. 70, 41–65. http://dx.doi.org/10.1016/j.asoc.2018.05.018, URL https://www.sciencedirect.com/science/article/pii/S1568494618302813.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press, http://www.deeplearningbook.org.

Gruosso, M., Capece, N., Erra, U., 2021a. Exploring Upper Limb Segmentation with Deep Learning for Augmented Virtuality. In: Frosini, P., Giorgi, D., Melzi, S., Rodolà, E. (Eds.), Smart Tools and Apps for Graphics - Eurographics Italian Chapter Conference. The Eurographics Association, http://dx.doi.org/10.2312/stag.20211483.

Gruosso, M., Capece, N., Erra, U., 2021b. Human segmentation in surveillance video with deep learning. Multimedia Tools Appl. 80 (1), 1175–1199.

Gruosso, M., Capece, N., Erra, U., 2021c. Solid and effective upper limb segmentation in egocentric vision. In: The 26th International Conference on 3D Web Technology. In: Web3D '21, Association for Computing Machinery, New York, NY, USA, http://dx.doi.org/10.1145/3485444.3495179.

Gruosso, M., Capece, N., Erra, U., 2022. Egocentric upper limb segmentation in unconstrained real-life scenarios. Virtual Real. 1–13.

Guo, D., Pei, Y., Zheng, K., Yu, H., Lu, Y., Wang, S., 2020. Degraded image semantic segmentation with dense-gram networks. IEEE Trans. Image Process. 29, 782–795. http://dx.doi.org/10.1109/TIP.2019.2936111.

Hao, S., Zhou, Y., Guo, Y., 2020. A brief survey on semantic segmentation with deep learning. Neurocomputing 406, 302–321. http://dx.doi.org/10.1016/j.neucom.2019.11.118, URL https://www.sciencedirect.com/science/article/pii/S0925231220305476.

Harkat, H., Nascimento, J., Bernardino, A., 2020. Fire segmentation using a DeepLabv3+ architecture. In: Image and Signal Processing for Remote Sensing XXVI, Vol. 11533. International Society for Optics and Photonics, p. 115330M.

Holschneider, M., Kronland-Martinet, R., Morlet, J., Tchamitchian, P., 1990. A real-time algorithm for signal analysis with the help of the wavelet transform. In: Wavelets. Springer, pp. 286–297.

Jadon, S., 2020. A survey of loss functions for semantic segmentation. In: 2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB). IEEE, pp. 1–7.

Jiang, Y., Huang, Z., Yang, B., Yang, W., 2022. A review of robotic assembly strategies for the full operation procedure: planning, execution and evaluation. Robot. Comput.-Integr. Manuf. 78.

Kang, H., Zang, Y., Wang, X., Chen, Y., 2022. Uncertainty-driven spiral trajectory for robotic peg-in-hole assembly. IEEE Robot. Autom. Lett. 7 (3), 6661–6668.

Kaur, D., Kaur, Y., 2014. Various image segmentation techniques: a review. Int. J. Comput. Sci. Mob. Comput. 3 (5), 809–814.

Kim, Y.-L., Kim, B.-S., Song, J.-B., 2012. 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, pp. 1074–1079.

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kong, Y., Liu, Y., Yan, B., Leung, H., Peng, X., 2021. A novel Deeplabv3+ network for SAR imagery semantic segmentation based on the potential energy loss function of gibbs distribution. Remote Sens. 13 (3), 454.

Krähenbühl, P., Koltun, V., 2011. Efficient inference in fully connected CRFs with Gaussian edge potentials. In: Shawe-Taylor, J., Zemel, R., Bartlett, P., Pereira, F., Weinberger, K. (Eds.), Advances in Neural Information Processing Systems, Vol. 24. Curran Associates, Inc., URL https://proceedings.neurips.cc/paper/2011/file/beda24c1e1b46055dff2c39c98fd6fc1-Paper.pdf.

Lee, M.A., Zhu, Y., Srinivasan, K., Shah, P., Savarese, S., Fei-Fei, L., Garg, A., Bohg, J., 2019. 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, pp. 8943–8950.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft coco: Common objects in context. In: European Conference on Computer Vision. Springer, pp. 740–755.

Liu, W., Rabinovich, A., Berg, A.C., 2015. Parsenet: Looking wider to see better. arXiv preprint arXiv:1506.04579.

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

Mo, Y., Wu, Y., Yang, X., Liu, F., Liao, Y., 2022. Review the state-of-the-art technologies of semantic segmentation based on deep learning. Neurocomputing 493, 626–646. http://dx.doi.org/10.1016/j.neucom.2022.01.005, URL https://www.sciencedirect.com/science/article/pii/S0925231222000054.

Nigro, M., Sileo, M., Pierri, F., Bloisi, D., Caccavale, F., 2023. Assembly task execution using visual 3D surface reconstruction: An integrated approach to parts mating. Robot. Comput.-Integr. Manuf. 81, 102519.

Nigro, M., Sileo, M., Pierri, F., Genovese, K., Bloisi, D.D., Caccavale, F., 2020. 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, pp. 4235–4240.

Nikolenko, S.I., 2021. Synthetic Data for Deep Learning, Vol. 174. Springer.

Noh, H., Hong, S., Han, B., 2015. Learning deconvolution network for semantic segmentation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1520–1528.

Nottensteiner, K., Stulp, F., Albu-Schäffer, A., 2020. Robust, locally guided peg-in-hole using impedance-controlled robots. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE, pp. 5771–5777.

Otsu, N., 1979. A threshold selection method from gray-level histograms. IEEE Trans. Syst. Man Cybern. 9 (1), 62–66.

Papandreou, G., Kokkinos, I., Savalle, P.-A., 2015. Modeling local and global deformations in deep learning: Epitomic convolution, multiple instance learning, and sliding window detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 390–399.

Park, H., Bae, J.-H., Park, J.-H., Baeg, M.-H., Park, J., 2013. Intuitive peg-in-hole assembly strategy with a compliant manipulator. In: IEEE ISR 2013. IEEE, pp. 1–5.

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

Redmon, J., Farhadi, A., 2018. YOLOv3: An incremental improvement. arXiv.

Ren, W., Tang, Y., Sun, Q., Zhao, C., Han, Q.-L., 2023. Visual semantic segmentation based on few/zero-shot learning: An overview. IEEE/CAA J. Autom. Sin. 1–21. http://dx.doi.org/10.1109/JAS.2023.123207.

Ronneberger, O., Fischer, P., Brox, T., 2015. U-Net: Convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. Springer International Publishing, Cham, pp. 234–241.

Ruder, S., 2016. An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747.

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

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al., 2015. Imagenet large scale visual recognition challenge. Int. J. Comput. Vis. 115 (3), 211–252.

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

Siłka, W., Wieczorek, M., Siłka, J., Woźniak, M., 2023. Malaria detection using advanced deep learning architecture. Sensors 23 (3), http://dx.doi.org/10.3390/s23031501, URL https://www.mdpi.com/1424-8220/23/3/1501.

Simonyan, K., Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations.

Souly, N., Spampinato, C., Shah, M., 2017. Semi supervised semantic segmentation using generative adversarial network. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 5688–5696.

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

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

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

Ulku, I., Akagündüz, E., 2022. A survey on deep learning-based architectures for semantic segmentation on 2D images. Appl. Artif. Intell. 36 (1), 2032924. http://dx.doi.org/10.1080/08839514.2022.2032924, arXiv:https://doi.org/10.1080/08839514.2022.2032924.

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

Wang, J., Liu, X., 2021. Medical image recognition and segmentation of pathological slices of gastric cancer based on Deeplab v3+ neural network. Comput. Methods Programs Biomed. 106210.

Wu, W., Gan, J., Zhou, J., Wang, J., 2021. A lightweight and effective semantic segmentation network for ethnic clothing images based on DeepLab. In: 2021 9th International Conference on Communications and Broadband Networking. pp. 34–40.

Xia, W., Cheng, Z., Yang, Y., Xue, J.-H., 2019. Cooperative semantic segmentation and image restoration in adverse environmental conditions. arXiv preprint arXiv:1911.00679.

Xie, Y., Tian, J., Zhu, X.X., 2020. Linking points with labels in 3D: A review of point cloud semantic segmentation. IEEE Geosci. Remote Sens. Mag. 8 (4), 38–59. http://dx.doi.org/10.1109/MGRS.2019.2937630.

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

Yin, C., Wang, B., Gan, V.J., Wang, M., Cheng, J.C., 2021. Automated semantic segmentation of industrial point clouds using ResPointNet++. Autom. Constr. 130, 103874. http://dx.doi.org/10.1016/j.autcon.2021.103874, URL https://www.sciencedirect.com/science/article/pii/S0926580521003253.

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