# Applying vision-guided graph neural networks for adaptive task planning in dynamic human robot collaborative scenarios

MA, Ruidong, LIU, Yanan, GRAF, Erich W and OYEKAN, John

Advanced Robotics

ISSN: (Print) (Online) Journal homepage: www.tandfonline.com/journals/tadr20

# Applying vision-guided graph neural networks for adaptive task planning in dynamic human robot collaborative scenarios

Ruidong Ma, Yanan Liu, Erich W. Graf & John Oyekan

**To cite this article:** Ruidong Ma, Yanan Liu, Erich W. Graf & John Oyekan (30 Sep 2024): Applying vision-guided graph neural networks for adaptive task planning in dynamic human robot collaborative scenarios, Advanced Robotics, DOI: 10.1080/01691864.2024.2407115

**To link to this article:** https://doi.org/10.1080/01691864.2024.2407115



Published online: 30 Sep 2024.

ADVANCED ROBOTICS
https://doi.org/10.1080/01691864.2024.2407115

FULL PAPER

OPEN ACCESS Check for updates

# Applying vision-guided graph neural networks for adaptive task planning in dynamic human robot collaborative scenarios

Ruidong Ma[a], Yanan Liu[b], Erich W. Graf[c] and John Oyekan[a,d]

[a]Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, UK; [b]School of Microelectronics, Shanghai University, Shanghai, People's Republic of China; [c]Department of Psychology, University of Southampton, Southampton, UK; [d]Department of Computer Science, University of York, York, UK

**ABSTRACT**

The Assemble-To-Order (ATO) strategy is increasingly becoming prevalent in the manufacturing sector due to the high demand for high-volume personalised and customised goods. The use of Human-Robot Collaborative (HRC) Systems are increasingly being investigated in order to make use of the dexterous strength of human hands while at the same time make use of the ability of robots to carry massive loads. However, current HRC systems struggle to adapt dynamically to varying human actions and cluttered workspaces. In this paper, we propose a novel neural network framework that integrates both Graph Neural Network (GNN) and Long Short-Term Memory (LSTM) for adaptive response during HRC scenarios. Our framework enables a robot to interpret human actions and generate detailed action plans while dealing with objects in a cluttered workspace thereby addressing the challenges of dynamic human-robot collaboration. Experimental results demonstrate improvements in assembly efficiency and flexibility, making our approach the first integration of iterative grasping and flexible HRC within a unified neural network architecture.

**ARTICLE HISTORY**
Received 24 March 2024
Revised 12 July 2024
Accepted 30 August 2024

**KEYWORDS**
Human-Robot-Collaboration; Graph Neural Network; robot learning from demonstration; Assemble-To-Order

## 1. Introduction

Market fragmentation, a concept where a marketplace is divided into many smaller markets with each containing customers with distinct preferences or requirements, is fast becoming a trend in various sectors. This is driven by consumer preferences for more bespoke and unique goods. As a result, large manufacturers are having to adapt their production lines to cater for increasing fragmented and smaller markets in order to stay competitive [1].

Towards this, Assemble-To-Order (ATO) manufacturing strategies, particularly the use of Human-Robot Collaborative systems, are increasingly being investigated in order to facilitate the manufacture of personalised and varied products in large quantities from standardised discrete components. This is a different problem set from the application of bin picking technologies in which Robots mostly follow a set of programmed instructions [2,3]. In HRC systems, the holy grail is that humans and robot collaborate together on a task and make use of their inherent strengths such as human dexterity with deformable objects or robotic strength for lifting heavy objects to complete a task.

However, traditional HRC systems are often constrained by rigid, pre-programmed workflows, face challenges in dealing with cluttered workspaces as well as in adapting to the dynamic nature of human actions and varying task requirements. Advances in Learning from Demonstration (LfD) has shown promise in creating more adaptable and intuitive HRC systems. LfD allows robots to learn from human actions and hence adjust to changing conditions [4].

Despite this progress, two main research gaps remain: **(1).** the need for a dynamic and adaptive architecture to interpret a wide range of previously unseen human action sequences in ATO scenarios and **(2).** the need for computational models that can generate and execute action plans directly from 2D images in cluttered environments while minimising the reliance on pre-defined rules and enhancing real-time responsiveness. In addressing these gaps, we propose a general framework that integrates a vision-based Graph Neural Network (GNN) with Long-Short-Term-Memory (LSTM) towards enabling adaptive robot planning in dynamic ATO scenarios. We show the adaptability of our framework in two practical ATO use cases where HRC is used.

**CONTACT** John Oyekan john.oyekan@york.ac.uk, oyekanjohn@gmail.com

All authors equally contributed.

In the first use case, we consider a scenario where the robot observes a human's assembly actions and then produces responsive robot plans accordingly to assist the human. Since the final product can vary according to customer demands, human actions would vary from product to product. For example, the human could assemble different types of components in the same assembly hole, or the same component in different assembly holes (See Figure 3). As a result, it is crucial to recognise and infer human actions towards the intended final product.

In the second use case, we make use of the same GNN+LSTM framework to enable a robot find and grasp the required object from a cluttered environment. Since various objects have unique geometric shapes, they might overlap each other and thereby impede direct grasping by the robot through simple object detection and visual servoing. To address this issue, we encode the spatial object scene into graph-based observations and use this as the input of our framework. We then iteratively produce robot grasping plans to handle objects as required. We refer to the second use case as object handling. Through these two use cases, we demonstrate our GNN+LSTM framework's ability to produce efficient, human-understandable robot plans and as result, improve flexibility and responsiveness in various manufacturing use cases that apply HRC systems.

The rest of this paper is organised as follows: Section 2 reviews the current advances in dealing with human plan and object-grasping variations in HRC scenarios. Section 3 introduces our proposed framework, detailing the integration of HRC and vision-based grasping techniques. Sections 4 and 5 present our experimental design and results, demonstrating the efficacy of our approach in a real-world ATO scenario. Finally, Section 6 concludes the paper with a discussion of the implications of our findings and directions for future research.

## 2. Literature review

Traditionally, HRC methodologies, typically involve offline programming, necessitating that human collaborators adhere to a predetermined workflow [5]. This approach becomes problematic when there are deviations in the workflow, either due to human variability or changes in the task itself. In order to achieve effective HRC during manual assembly tasks, it is important that the robot detects and understands the variations (both in human actions and objects) in the workspace within which a HRC-based ATO process is being conducted. For example, during the assembly of valves for different types of product models as described in Section 3.1 (See Figure 3), various types of valves are positioned in different holes of a bracket for different products. The robot must recognise the various types of valves used as well as their relation to others in order to infer the type of product that is being assembled and then subsequently offer the appropriate assistance. If a human worker were to alter the positioning of the valve due to a change in product type, the robot needs to infer this and dynamically update its understanding of the workspace layout.

In this work, we investigate the application of GNNs in understanding human intended goals and their requirements in Human-Robot-Collaboration (HRC) scenarios. We also show that our approach is capable of producing feasible robot plans as natural language which is more understandable and user-friendly to human co-workers. In the following subsections, we discuss the current research in dynamic human action understanding and planning in HRC as well as object manipulation by robots.

Graph Neural Networks (GNNs) were originally designed to tackle graph-structured data. They represent information as a set of nodes and edges that connect nodes. In the field of robotic task planning, GNNs have shown great promise in dealing with long-horizontal tasks. This is due to their ability to represent each task as a node and the dependencies between tasks as edges. As a result, GNN's inherent structure enables task sequences and their dependencies to be captured [6]. Furthermore, recent work has shown GNN's ability to propose symbolic task plans as well as sequential object manipulation plans [7–10].

### 2.1. Vision-based human intention understanding and planning

Understanding human actions and intentions is a primary task in order to ensure successful HRC. Research has focussed on interpreting human assembly scenes, including human poses and the surrounding environment, from spatial images using CNNs [11–13]. The analysis of human movement trajectories is also critical for robots to recognise assembly intentions. Previous works have utilised Hidden Markov Models (HMM) and Multilayer Perceptrons (MLP) with skeleton joints [14], alongside Recurrent Neural Networks (RNN) [15] and semi-flexible Neural Networks [16]. Moreover, a combination of temporal motion features with spatial assembly context, as suggested by Moutinho et al. [17], could also distinguish different assembly actions. Most studies often employ CNNs for object detection and RNNs to identify assembly actions from either depth image sequences [18] or skeleton data [19].

In the context of ATO environments, where diverse products are created, different components may be placed in the same position based on customer demands, as

illustrated in Figure 5. Here, while the assembly motions (moving to a position) remain consistent, the corresponding objects might differ. In this case, the robot is tasked with identifying the human's intended goal based on the sequences of human actions observed. In this regard, probabilistic methods are frequently employed to infer the likely goals intended by humans. For example, Bayesian Inference has been applied to deduce human navigational goals [20] and assembly plans by incorporating prior knowledge about human poses and object interactions [15]. Additionally, the variable-length Markov Model (VMM) has been utilised to analyse classified action sequences, thereby aiding in generating optimal plan predictions [12]. These methods, however, often depend on prior expert knowledge.

Additionally, in order to equip robots to assist human coworkers with planned actions, the concept of hierarchy within demonstrated task structures has been investigated. These task structures can be established using a predefined AND/OR graph while taking into account all possible plan combinations, as discussed in [19,21,22]. An extension of this is the Hierarchical Task Network (HTN) methodology which has been utilised in defining task structures and model transition probabilities [23,24]. By applying Hidden Markov Models (HMM), the identification of hidden states that link various subtasks has been investigated in [25]. In [5], inverse reinforcement learning (IRL) was employed to determine the most desirable actions that yield optimal rewards for custom-made products. However, all these methodologies often necessitate design based on domain-specific knowledge.

### 2.2. Vision-based iterative grasping generation for sequential object handling

In a cluttered environment such as in Figure 7(b), simple object detection and visual servoing approach is difficult to apply and inefficient in grasping an object that might be required by a human during an assembly task. This is because the objects can overlap with each other and their unique geometric shapes can impede the direct grasping of the robot.

In order to address this problem, one research direction is to model the geometric property of the target grasp object. Previous works focussed on predicting the grasping quality and grasping pose for the robot's end-effector based on the pixels from the object's depth image [26,27], point cloud data [28] or event-based camera [29,30] propose a hierarchical framework that learns goal-driven grasps based on partial point cloud observations. Furthermore, some other research focussed on applying haptic information rather than visual features. For example, Abi-Farraj et al. [31] described a novel tactile shared control method to assist human operators in sorting and segregating multiple targets in cluttered and unknown environments. Garcia-Garcia et al. [32] utilised GNN for processing tactile data while [33] combined language models with GNN for adaptive deformable object manipulations. The aforementioned methods often require rich information about the target object, which can be problematic when the target object is overlapped by the surrounding objects too closely. In these cases, the direct grasping pose estimation may not be effective.

Another possible solution is to reason about the objects' relationship with each other through the use of visual features. This would allow the robot to plan the grasping of the intended object through the sequential manipulation of other objects in the workspace. Current research focuses on classifying the visual relationship between each object pair in a spatial image. Abi-Farraj et al. [31] describes a novel tactile shared control method to assist human operators in sorting and segregating multiple targets in cluttered and unknown environments. Lu et al. [34] combines the detected spatial, semantic and visual information from object pairs and produces the object interaction status through CNN and tree-based gradient boosting models (GBMS). In [35], spatial and temporal object graphs were constructed through detected object features and detected human-object interactions. By leveraging semantic and visual information through CLIP [36,37] investigated the rearrangement of unseen but similar objects with seen goal images. Huang et al. [38] studied the objects' relationship over the robot manipulation via partial point cloud and GNN while [39] constructed a knowledge graph representation using Markov Logic Networks to obtain the probability distribution of an object's grasp availability. GNN has also been applied to predict the relationship nodes through the encoded graph observation in [40,41]. However, their work still needs to examine every possible relation between each pair.

## 3. Methodology

This section explores the capability of GNN in dealing with dynamic changes in HRC. The idea is to encode the detected observations into structured graph representations, by aggregating neighbour information through GNN. Subsequently, robot actions can be generated in natural language text form by decoding the graph-based feature representations using LSTM (Long Short-Term Memory).

Figure 1 shows our proposed overall framework for graph-based robot planning and learning during varying HRC scenarios. In the first use case as mentioned above,

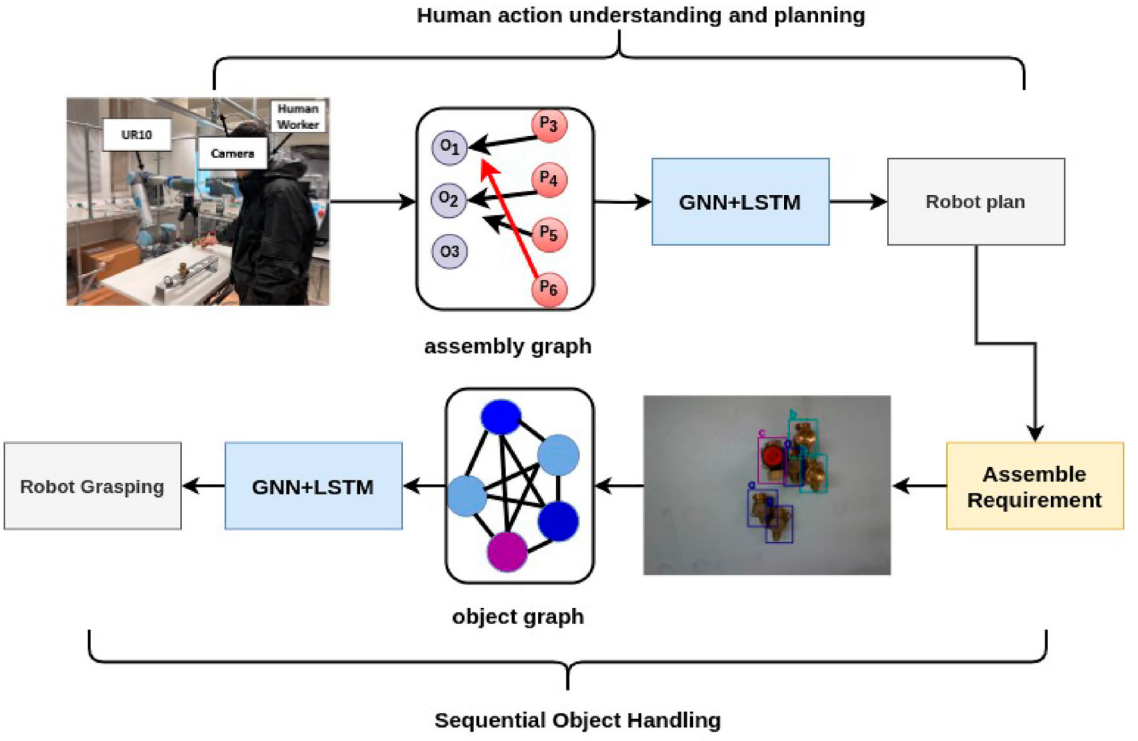



**Figure 1.** Proposed unified framework of GNN+LSTM for proposed use cases.

we study an HRC scenario where the robot needs to infer and understand what the human is doing (Section 3.1) as they work towards achieving an end goal. In this case, the human will be using various assembly strategies to achieve various end-goal configurations. In order to support the human in their task, our framework decodes the human's actions and then produces a detailed robot action plan that involves grasping an object and carrying to the appropriate location in the workspace. We refer to this use case as human intention understanding and planning.

In the second use case, we consider the situation where the required object needs to be grasped from a cluttered workspace (Section 3.2). This can be considered as a sequential grasping problem in which the required object is heavily occluded by other objects in the workspace. As a result, direct object grasping would not work. In this use case, we apply the same GNN+LSTM framework to process the spatial image scenes and produce a sequence of robot grasping plans that manipulate other objects by moving them in order to make the required object graspable. We refer to this use case as sequential object handling.

### 3.1. Human intention understanding and planning

Assume an ATO scenario for one end-product with $z$ goal configurations $G = \{g_z\}_{z=1}^{z}$ according to customer demands. It is assumed that there are $n$ types of components or objects $O = \{o_n\}_{n=1}^{n}$ and that each of them has different amounts. Moreover, assume there will be $m$ possible assemble positions $P = \{p_m\}_{m=1}^{m}$ for $O$ towards achieving final goal configurations (Please see our previous work [42] for more detailed information for this section).

The framework, as illustrated in Figure 2, is designed to establish a HRC system that is capable of dynamically detecting human actions. The objective of this framework is to accurately identify a human's intended goal, denoted as $g$. Based on this recognition, the framework generates a detailed plan for the robot. This plan specifies the position, represented as $p$, where the next workpiece, $o$, should be assembled. It is important to note that the position $p$ may vary for each workpiece $o$ in the workspace.

MediaPipe [43] is utilised for initial hand tracking. It provides accurate and efficient hand detection, which is crucial for understanding human actions in real-time. As the human fingers may not be fully detected during assembly, estimating hand pose can be difficult. Furthermore, object detection can also be challenging when the hand overlaps the object [18]. Thus, instead of hand pose estimation and object detection, the spatial features are termed as the interaction status between the hand and objects through the hand region cropped by MediaPipe as shown in Figure 3. A CNN-based VGG16 model

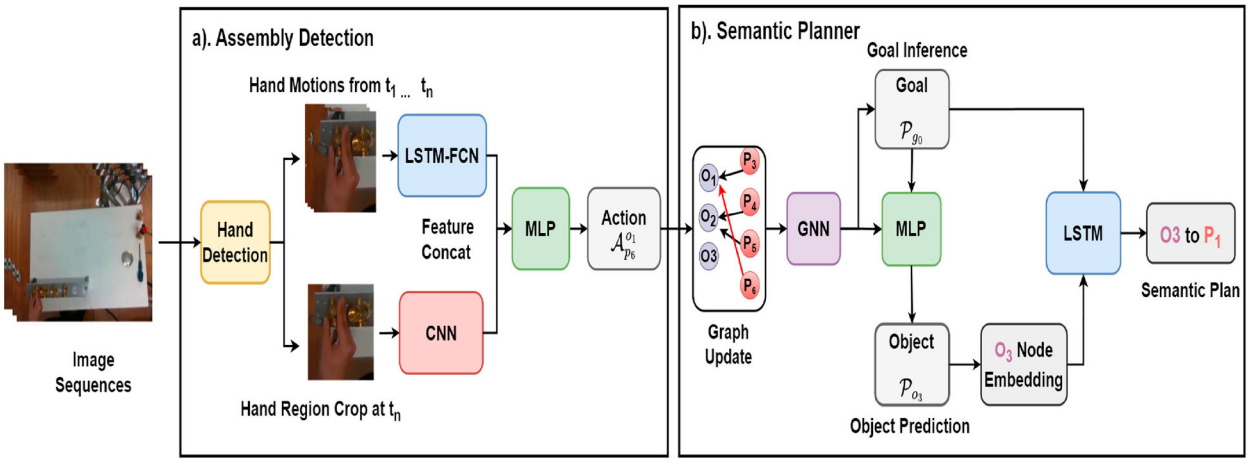



**Figure 2.** Pipeline of the dynamic human action understanding: (a). The Assembly Detector utilises MediaPipe for right hand detection in images and classifies actions by merging CNN-extracted hand-object features with LSTM-FCN motion features. (b). The Semantic Planner updates the assembly graph with classified actions, infers goals, and identifies the next assembly object using GNN, translating this into robot action instructions via LSTM.

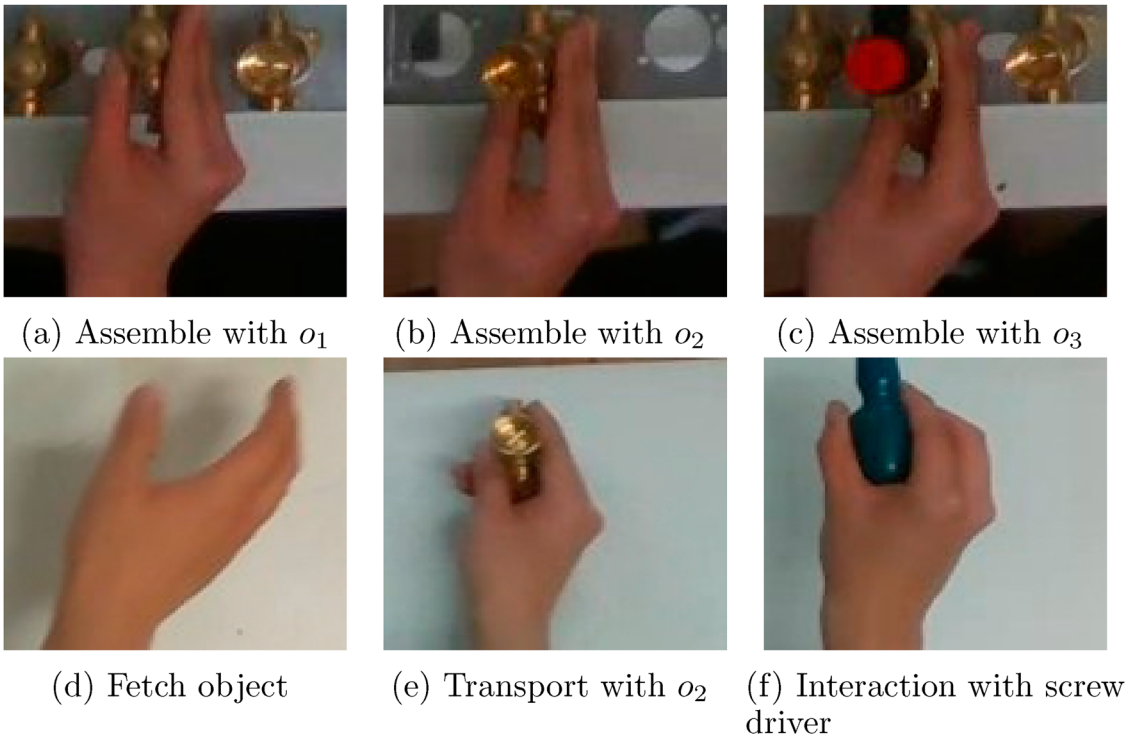



**Figure 3.** Examples of spatial assembly actions. (a) Assemble with $o_1$. (b) Assemble with $o_2$. (c) Assemble with $o_3$. (d) Fetch object. (e) Transport with $o_2$ and (f) Interaction with screw driver.

[44] extracts spatial features from hand-centric images, $f_{spatial} = CNN(I)$, where $I$ is the last frame of hand crops.

Moreover, we utilise an LSTM Fully Convolutional Neural Network (LSTM-FCN) [45] which extracts temporal features from hand positions. It provides robustness to noise with minimal pre-processing by combining LSTM's temporal dependencies $h_t$ and FCN's time-invariant features $f_{FCN}$ as $f_{temporal} = concat(h_t, f_{FCN})$.

Temporal and spatial features are thus merged using a Multilayer Perceptron (MLP) to classify assembly actions as $\mathcal{A}_{p_m}^{o_n}$, which stands for $o_n$ has been assembled to $p_m$. There are also intermediate actions like fetching $\mathcal{A}_{trans}$ and screwing $\mathcal{A}_{screw}$ as shown in Figure 3(d–f). These classifications inform semantic planning and guide robot actions, based on the likelihood of each assembly action. The framework is trained end-to-end with cross entropy

loss to optimise action detection as shown in $\mathcal{L}_a = \text{argmin}_\theta\,[-\sum_{i=1}^{i}(\hat{\mathcal{A}}_i \log \mathcal{A}_i + (1-\hat{\mathcal{A}}_i)\log(1-\mathcal{A}_i))]$, where the $\hat{\mathcal{A}}_i$ stands for the ground truth hand-object interaction status.

### 3.1.1. Graph-based semantic planning

The semantic planner learns **which** object should be handled to **where** based on inferred **varying** goal configurations. The overall pipeline of this proposed graph-based semantic planner are described as in Figure 2(b). We use our proposed GNN+LSTM framework to process and generate adaptive robot plan from the information obtained from the assembly detector as described above.

Assembly scenes are represented by a graph $\mathcal{G}$ with nodes $V$ for objects $v_{o_n}$ and positions $v_{p_m}$, totalling $n+m$ nodes. Nodes feature are two-dimensional categorical data indicating object types and assembly positions. Detected assembly actions $\mathcal{A}_{p_1}^{o_1}, \ldots \mathcal{A}_{p_m}^{o_n}$ are used to generate an adjacency matrix $E$ with directed edges $\mathrm{e}_{p_m}^{o_n}$ linking objects to their assembly positions based on classified actions, effectively mapping the relationships between object and position nodes:

$$\mathrm{e}_{p_m}^{o_n} = \begin{cases} 1 & \mathcal{A}_{p_m}^{o_n} = 1 \\ 0 & otherwise \end{cases} \tag{1}$$

This study employs a GraphSAGE layer [46] to process the assembly graph, which averages feature embeddings from neighbouring nodes for each object node using $\Gamma_{\mathcal{N}(i)}^{k} = \frac{1}{N}\sum_{j\in\mathcal{N}(i)}(\Gamma^{k-1}j)$, where $k$ denotes the layer number. The resulting embedding is then combined with the target node's previous layer embedding and processed through a weighted matrix $W_k$ as Equation (2). This method incorporates trainable parameters $\theta$ of $\theta_{gnn_{k-1}}$ and $\theta_{gnn_k}$ across Sage layers with the ReLu activation function for embedding transformations.

$$\Gamma_i^k = \sigma\left(W_k \cdot \left(f_{\theta_{gnn_{k-1}}}(\Gamma_i^{k-1}) + f_{\theta_{gnn_k}}(\Gamma_{\mathcal{N}(i)}^k)\right)\right) \tag{2}$$

A readout layer is used to aggregate nodes embedding $\Gamma_i^k$ into graph embedding as $\Phi_k = \frac{1}{\mathcal{N}(v)}\sum_{i\in\mathcal{N}(v)}\Gamma_i^k$. A final output layer will accept $\Phi_k$ and produce $z$ dimensional outputs $\mathcal{P}_g^{pred} = \{\mathcal{P}_{g_1}^{pred}, \ldots, \mathcal{P}_{g_z}^{pred}\}$ which describe the probability of the inferred human indented goal configuration $g_z$. It further predicts the next object should be assembled as $\mathcal{P}_o^{pred} = \{\mathcal{P}_{o_1}^{pred}, \ldots, \mathcal{P}_{o_n}^{pred}\}$ through another MLP with the inputs containing graph embedding $\Phi_k$ and inferred goal $\mathcal{P}_g^{pred}$ as $\mathcal{P}_o^{pred} = MLP(\Phi_k, \mathcal{P}_g^{pred})$. To train this model, it is considered as a classification problem where the demonstrated ground truth $\mathcal{P}_g^{tgt}$ and $\mathcal{P}_o^{tgt}$. $\mathcal{P}_g^{pred}$ and $\mathcal{P}_o^{pred}$ are jointly learnt via cross-entropy loss:

$$\mathcal{L}_g = \text{argmin}_\theta \times \left[-\sum_{n=1}^{n}[\mathcal{P}_{o_n}^{tgt}]\log(\mathcal{P}_{o_n}^{pred}) - \sum_{z=1}^{z}[\mathcal{P}_{g_z}^{tgt}]\log(\mathcal{P}_{g_z}^{pred})\right] \tag{3}$$

In ATO scenarios, a simple multi-class classification is inefficient in producing a plan such as 'handling objects to multiple positions'. This is because it will always produce a deterministic result (i.e. the label with the highest probability). An advantage of applying a graph in this work is that: for each object node, it only aggregates the assembled position information that is relevant to itself at different HRC stages under different goal configurations. Therefore, this work aims to produce 'semantic plans' that interpret the objects' graphical observations through a simple LSTM with the inferred goal.

During training, the LSTM model commences with the input features including the node embedding $\Gamma_o^k$ and the label $\mathcal{P}_g^{pred}$ denoted as $f_{lstm} = \{\Gamma_o^k, \mathcal{P}_g^{pred}\}$. Concurrently, the ground truth captions (i.e. semantic plan) are processed through an embedding layer, converting discrete word indices into continuous vectors **embeds** $= W_{\text{embedding}}$[captions], where $W_{\text{embedding}}$ represents the embedding matrix. These embedding, denoted as **embeds**, are then concatenated with the input features to form the complete input for the LSTM layer in **inputs** $= concat(f_{lstm}, \textbf{embeds})$. This is known as the 'teacher forcing' strategy in order to improve the training and enhance model stability. Then, the LSTM updates its hidden state $h_t$ and cell state $c_t$ using $(h_t, c_t) = \text{LSTM}(\textbf{inputs}, h_{t-1}, c_{t-1})$ at each time step $t$. Finally, the output is passed through a fully connected layer to predict the next word with the highest probability.

During the testing stage, the procedure adopts a slightly different approach by commencing with the $f_{lstm}$ as the initial input to the LSTM. At each time step, the LSTM's output is passed through the fully connected layer, which predicts the most probable word. This word's embedding then serves as the input for the next time step, creating an iterative loop until a termination condition is met. Therefore, it can be simply expressed as $txt = LSTM(\Gamma_o^k, \mathcal{P}_g^{pred})$. The generated $txt$ contains the information regarding the object type and its unfinished assembly positions, for example, '$o_1$, $p_3$, $p_4$' refers to $o_1$ should be assembled to $p_3$ and $p_4$ afterwards. The object type is necessary for the further robot control selections for different shape of the object. Moreover, the planner can recognise the situation when all the positions of $o_1$ have been completed and generate text as '$o_1$, Finished'.

The graph $\mathcal{G}$ is updated using the most recent action label $\mathcal{A}_t$, which varies from $\mathcal{A}_{p_1}^{o_1}$ to $\mathcal{A}_{p_m}^{o_n}$. The framework then provides Semantic Guidance to each object type $o$.

**Algorithm 1:** Proposed HRC system

Initialize assembly graph $\mathcal{G}$
Trained action detection system $\mathcal{M}$, semantic planner $\mathcal{C}$ including a GNN encoder *GNN* and a decoder *LSTM*
**while** *Assembly not finished* **do**
  Track hand motion via MediaPipe
  **if** *hand detected* **then**
    Hand trajectories segment $s$ with $t$ frame length and hand-centric image crop $I$ at the last step
    Predict assembly actions $\mathcal{A}_t = \mathcal{M}(I, s)$
    **if** $\mathcal{A}_t \in \{\mathcal{A}_{p_1}^{o_1}, ...\mathcal{A}_{p_m}^{o_n}\}$ **and** *hand motion finished* **then**
      Update Graph Edge $e_{p_m}^{o_n}$ according to assembly actions $\mathcal{A}_{p_m}^{o_n}$
      Update objects status in pending area
      Infer final goal $g_z = GNN(\mathcal{G})$
      Generate **Semantic Guidance** for each object $o_n$
      $[o_n, p_1, ..., p_m] = LSTM(\Gamma_{o_n}, g_z)$
    **else if** $\mathcal{A}_t = \mathcal{A}_{screw}$ **then**
      Infer final goal according to current graph $\mathcal{G}$
      as $g_z = GNN(\mathcal{G})$
      Predict next object $o_{next} = MLP(\Phi, g_z)$
      Produce **Semantic Control** command $[o_{next}, p_1, ..., p_m] = LSTM(\Gamma_{o_{next}}, g_z)$
      Robot execution $PickandPlace(o_{next}, p_1)$
      Update Graph

In HRC scenarios, upon detecting $\mathcal{A}_{screw}$, the robot executes a predefined 'PickandPlace()' function based on Semantic Control as produced by our planner, allowing the human to concentrate on screwing or assembling tasks. During these phases, the detector $\mathcal{M}$ is disabled, and the graph is updated to reflect the progress, This will continue until each object $o$ is designated as 'Finished'.

### 3.2. Sequential object handling

Given an image scene containing various objects, the goal is to enable the robot to identify the graspability of the object required by the produced semantic control from the previous section. Considering the cluttered environment as shown in Figure 4, the graspability of an object is determined by both its surrounding objects and its own geometrical properties, while respecting the capability of the robot. For example, considering a robot can only grasp a required object from the top, if such an object is overlapped by a taller object, it is considered as not graspable at the current stage. As a result, the focus of this research is to devise an iterative approach for removing obstructions until the targeted object becomes accessible for grasping.

Figure 4 describes the proposed framework in which we use a Faster R-CNN [47] with Resnet101 as the backbone feature extractor for object detection. In addition to the extracted objects' features and its feature values $\mathbf{f}_i$, the label $l_i$ and the 2D centre position of the bounding box $\mathbf{d}_i$ within the 2D image are also extracted. However, relying solely on the spatial information of objects within an image, such as the bounding box coordinates, does not adequately convey information regarding the graspability of objects in the given scenario.

Instead, we encode the detected objects into a graph observation. Suppose graph observation $\mathcal{G}$ containing $n$ nodes as $V = \{v_0, v_1, v_2, \ldots, v_n\}$. Each node contains the extracted features $\mathbf{f}_i$, object label $l_i$ and the binary goal feature indicating the demand from human $g_i$, $v_i = \{\mathbf{f}_i, l_i, g_i\}$. The graph uses fully connected directed edges to emphasise spatial relationships between objects by employing normalised inverse weighted edges $E = e_{ij}$ as shown in Equation (4). These relationships are defined by the 2D Euclidean distance $\|\mathbf{d}_i - \mathbf{d}_j\|$ between the centres of objects' bounding boxes. This captures the spatial proximity of each object to its neighbours more effectively within the graph structure than previous work which focussed on temporal human action changes.

$$e_{ij} = \frac{\frac{1}{\|\mathbf{d}_i - \mathbf{d}_j\|}}{\sum_{j=1}^{n} \frac{1}{\|\mathbf{d}_i - \mathbf{d}_j\|}} \tag{4}$$

In order to process such a graph observation, the Weisfeiler-Lehman (WL) inspired Graph Neural Network operator (WL-GNN) is adopted [48]. It iteratively updates node labels through multiple rounds of neighbour aggregation. The WL algorithm takes into account the multi-hop neighbourhood information of nodes, enabling it to capture the intricate structures within a graph:

$$\Gamma_i^{k+1} = \sigma\left(W_1^k \cdot \Gamma_i^k + W_2^k \cdot \sum_{i \in \mathcal{N}(i)} \cdot e_{ij} \cdot \Gamma_j^k\right) \tag{5}$$

where $W_1^{(k)}$ and $W_2^{(k)}$ are the weighted matrix to handle the information from the current node and its neighbours. This ensures that during the information aggregation process, the information from the current node

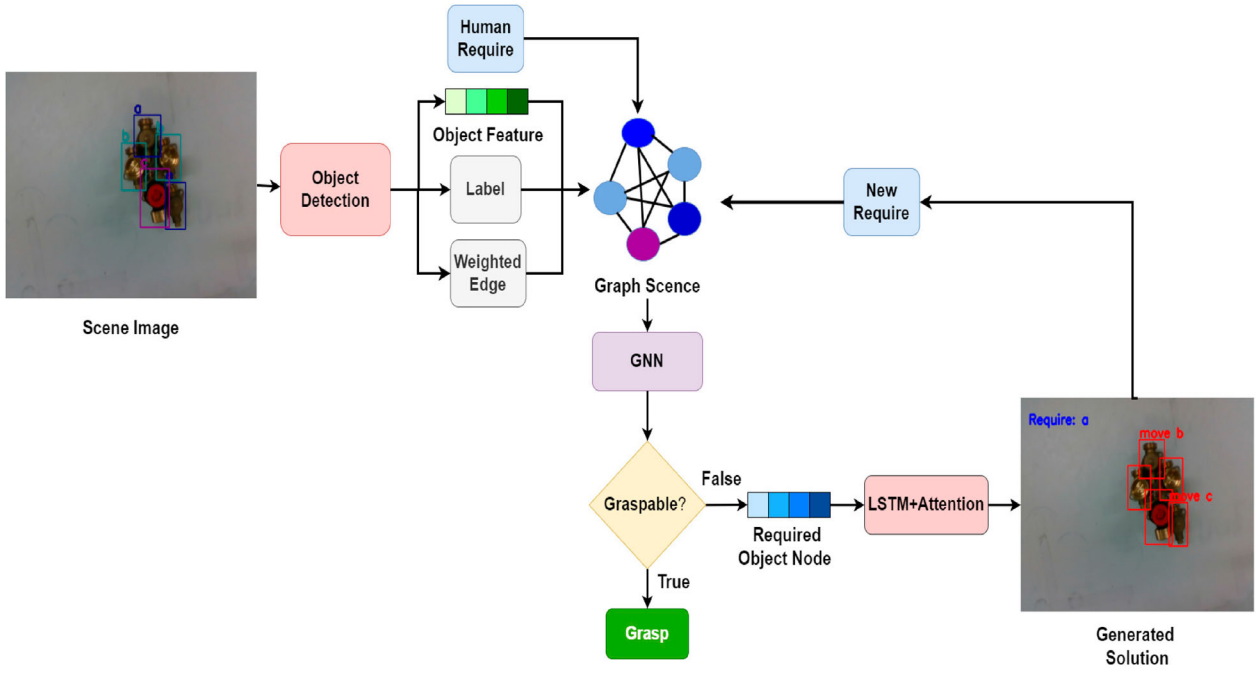



**Figure 4.** The framework starts with object detection in scene images, using networks like Faster-RCNN to extract object features for a graph. It then assesses object graspability with a GNN. If an object is ungraspable, an LSTM with Attention suggests adjustments (e.g. moving other objects) to achieve graspability, iterating this process until the object can be grasped.

and its neighbours is kept distinct. Furthermore, during message passing, the weighted matrix $e_{ij}$ is used to modulate the propagation of information between nodes. The neighbour node with a higher weight edge has a greater influence on the target node. This is unlike previous work in which node representations are based on their immediate neighbours. WL-GNN is more capable of exploring richer and more in-depth graph structural information.

Afterwards, the graspability is considered as a node classification problem with binary output as $P^{gra}_{pred}$. The cross-entropy loss is used to optimise the model:

$$loss = -[P^{gra}_{goal} \log(P^{gra}_{pred}) + (1 - P^{gra}_{goal}) \log(1 - P^{gra}_{pred})] \tag{6}$$

For objects identified as non-graspable, the trained nodes not only retain information about themselves but also about their weighted neighbours. Additionally, during the learning process within the graph, it appears that the graph captures the demonstrated task structure. Consequently, the object node embedding is decoded into its corresponding solution as contextual information through Long-Short-Term-Memory with Attention Mechanism (LSTM_Att) [49].

The input feature consists of the trained node embedding $\Gamma^k_i$, object label $l$ and the graspablity $P^{gra}_{pred}$ as $f' = \{\Gamma^k_i, l, P^{gra}_{pred}\}$. The original intent behind the Attention mechanism was to assist models in determining which part of the encoded sequence to focus on during decoding. In this task, this implies identifying which parts of the input features should be emphasised when generating a particular word. Given the current LSTM hidden state $h_t$ and the input feature, the attention mechanism first computes an attention score as $\text{score}(h_t, f') = h_t^T W_k f'_s$, where $W_k$ is the trainable weights and $f'_s$ is a segment of the input feature. Accordingly, the attention weight can be expressed as $\alpha_{ts} = \text{softmax}(\text{score}(h_t, f'_s))$. This provides a context-based weight to different portions of the input feature as $context_t = \sum_s \alpha_{ts} h'_s$. Therefore, $context_t$ is the new input feature to the LSTM at time step t. A similar training process is applied as described in the previous section.

During testing, the output can be expressed as Equation (7) where the initial word embedding **embeds** will always be the 'start' token.

$$txt_i = LSTM\_Att(\Gamma^k_i, l, P^{gra}_{pred}, \textbf{embeds}) \tag{7}$$

Alg. 2 outlines our iterative visual grasping framework initiated by human requirements. In complex scenarios where the target object is obstructed by other objects, the framework, leveraging extracted visual features, iteratively devises solutions to remove these obstacles by treating them as new inputs for the GNN and LSTM with Attention. This process seeks to identify the most graspable object in a scene while aiming to ultimately enable the robot to grasp the human-specified target

**Algorithm 2:** Proposed Iterative Grasping Framework

Initialize the object detector $O$
Trained GNN model $G$
*LSTM_Att* decoder
Input: human required object $o_{goal}$
**while** *not* $P^{gra}_{goal}$ **do**
  Obtain the raw 2D image $I$
  Extract visual information $f, l, d = O(I)$
  Construct the graph scene $\mathcal{G}$ according to $o_{goal}, f, l, d$
  Predict the graspability of the required object $P^{gra}_{goal} = G(\Gamma^{k}_{o_goal})$
  **if** *not* $P^{gra}_{goal}$ **then**
    Find the new require object $o_{re}, ..., = LSTM\_Att(\Gamma^{k}_{o_goal}, l, P^{gra}_{goal})$
    **while** *not* $P^{gra}_{re}$ **do**
      Construct the graph scene $\mathcal{G}$ according to $o_{re}, f, l, d$
      Predict the graspability of the required object $P^{gra}_{re} = G(\Gamma^{k}_{o_re})$
      **if** *not* $P^{gra}_{re}$ **then**
        Find the new require object $o_{re}, ..., = LSTM\_Att(\Gamma^{k}_{o_goal}, l, P^{gra}_{re})$
      **else**
        Robot Grasp
        break

object $o_{goal}$. The process may involve multiple steps of solution generation to achieve this end.

## 4. Experimental setup

In this section, we demonstrate the effectiveness of our proposed framework through the use of a real-industrial ATO scenario. This ATO scenario involves assembling valve brackets, which vary depending on the product model. The use case studies three object types ($O = o1, o2, o3$) as shown in Figure 3(a–c), needing three $o_1$, two $o_2$ and one $o_3$. These objects are used to construct three different product/goal configurations $G = \{g_0, g_1, g_2\}$ as shown in Figure 5. We now describe the experimental setups for our proposed HRC system in two use cases of: **(1)** Human intention understanding and planning as well as **(2)** Sequential object handling.

In the human intention understanding and planning use case, our framework's task is to understand which object $O$ has been assembled by human and then construct robot plans to predict the next object $O$ to be grasped to the assembled hole under different goals $G$ as shown in Figure 6(b).

In the second sequential object handling use case, once the 'next object' $O$ has been decided, our framework aims to enable the robot grasp the required object from a cluttered environment. As this object can be occluded, our framework's task is to understand the spatial relationships between the objects and make plans to grasp them as shown in Figure 7.

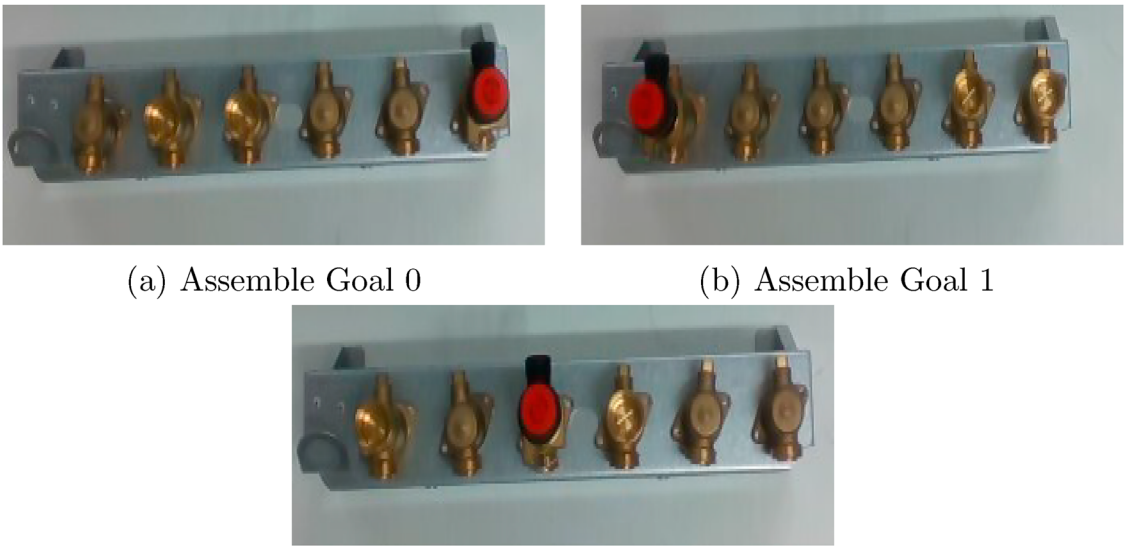
In both use cases, we used the same Universal 10 robot arm with two-finger gripper robitq2F-85. We also used the same object types $O$ and quantities for both use cases. However, in order to aid better understanding of the results, we rename the objects as $O = \{a, b, c\}$ in the second use case.

### 4.1. Human intention understanding and planning

These experiments are structured to validate the efficiency of the Hand-centric Action Detector in recognising human assembly actions using only 2D video demonstrations. Additionally, the experiments highlight the proficiency of the Semantic Planner in managing diverse goal configurations. This includes its adaptability to different human action sequences and task-planning strategies.

We use a RealSense camera (D435i) to monitor the assembly workspace within which a worker assembles objects onto a steel bracket as shown Figure 6(b). A user interface (Figure 6(b)) shows detected hand frames, assembly actions, goals and positions produced by the **Semantic Guidance** in our framework. A UR10 manipulator is controlled through our **Semantic Control** as shown in Figure 6(a). The 'Goal' output stands for the inferred human indented goal configuration. The 'Obj1, Obj2, Obj3' stand for the object type. The positioning outputs after 'Obj' are denoted by the numbers '1, 2, 3, 4, 5' indicating the assembly positions on the bracket from right to left.

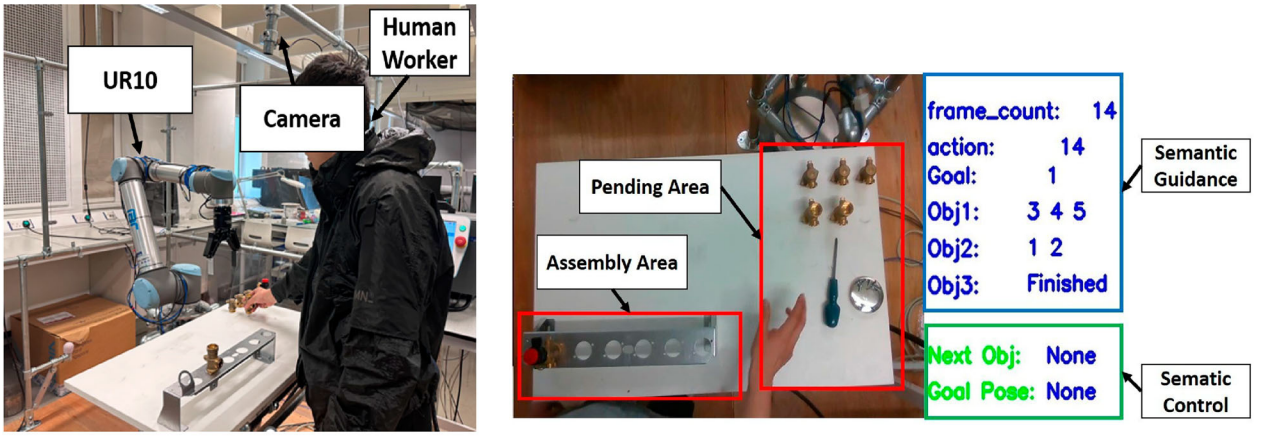
Demonstration video are segmented at 45-frame intervals at 15 FPS, with the assembly involving specific actions for each object type and intermediate screwing actions. This yields a total of 18 actions for training the action detector. For the semantic planner, the assembly graph is automatically generated using the data from the trained action detector. This graph is composed of three object nodes, denoted as $v_o$ and six position nodes as $v_p$. The expert demonstrate consistent task structures, which trains the robot to always select the first unassembled object from the right (i.e. $P^{pred}_{o}$), in alignment with the



(a) Assemble Goal 0 (b) Assemble Goal 1

(c) Assemble Goal 2

**Figure 5.** Three types of customised product/goal configurations. In this use case, the valves are assembled onto the steel bracket. (a) Assemble Goal 0. (b) Assemble Goal 1 and (c) Assemble Goal 2.




(a) Experimental setup (b) A designed user interface.

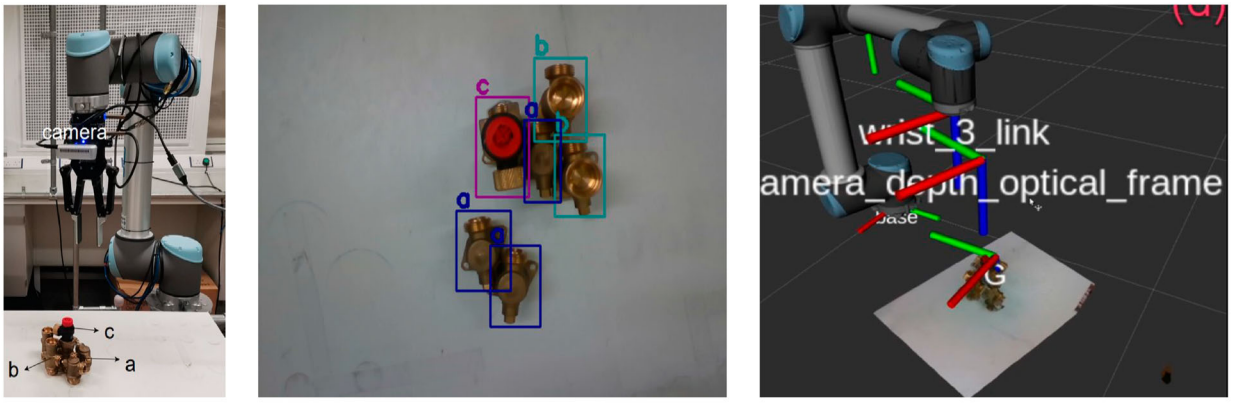
**Figure 6.** The experimental setup and a designed user interface that provides an indication of the robot actions in real time. (a) Experimental setup and (b) A designed user interface.

inferred goal $P_g^{pred}$. Subsequently, the embeddings of the trained object nodes are annotated, which is essential for training the LSTM.

### 4.2. Sequential object handling

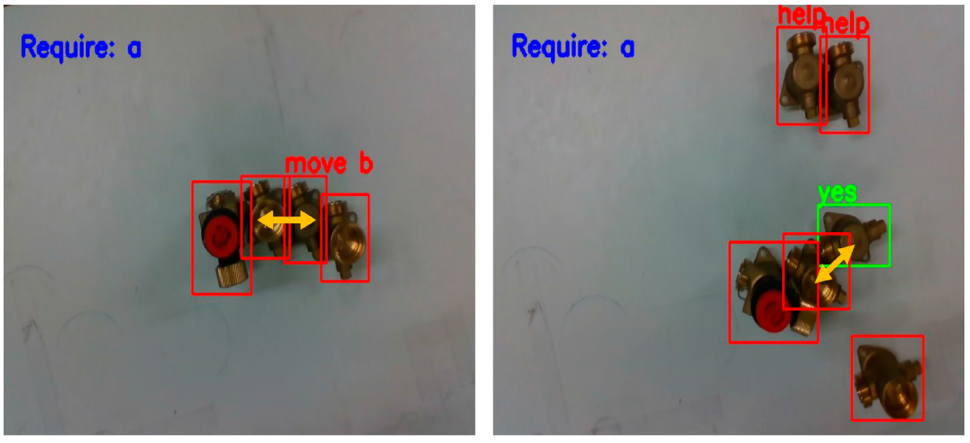
In the second use case of sequential object handling, the camera was mounted on the end-effector of the robot arm so that top-to-bottom object grasps could be achieved (Figure 7(a,c)). The scene includes three component types ('a, b, c') with maximum quantities of 3, 2, and 1 respectively, as depicted in Figure 7(b). Our framework focuses on geometric and spatial characteristics of objects while assessing object graspability. This is achieved by considering the geometric shapes, heights and spatial relationships among the objects. This knowledge is then used to prioritise objects' graspability accordingly. For example, in Figure 7(b), object *c* is deemed most graspable as it is the tallest among the object.

The different object orientations and the inefficiency of applying a deterministic distance threshold highlight the need for capturing and analysing visual information, as shown in Figure 8. For this purpose, we used a Faster-RCNN architecture with Resnet101 and feature pyramid networks (Faster-RCNN-fpn) for labelling and extraction of object visual features [50]. During the labelling process for the object detection, we draw bounding boxes




(a) Physical setup (b) Robot top view of the cluttered scene (c) Robot side view of the cluttered scene

**Figure 7.** The industrial components handling setup. (a) Physical setup. (b) Robot top view of the cluttered scene and (c) Robot side view of the cluttered scene.




(a) (b)

**Figure 8.** The double-headed arrows show the distance between the objects. The distance between part a and part b (0.08) in Figure 8(b) is smaller than the distance (0.10) in Figure 8(a). In the above, the green box stands for graspable objects while the red boxes stand for ungraspable objects.

for the objects while allowing some parts of the occluded object to be involved according to [41]. Therefore, each object node in the GNN contained the object itself and some part of the occluding objects. Due to this approach, the weighted edges of the GNN also aggregated the relevant features of the occluded objects as discussed in Section 3.2. We annotated each object node's graspability as the ground truth. The GNN could therefore be trained to directly classify the node graspability without a specific design. This demonstrated the feasibility and adaptability of the GNN in various occlusion situations.

For the objects classified as ungraspable, their trained node embeddings are annotated with potential solutions such as 'Move b and/or c, or help' for training the LSTM.

Additionally, our proposed framework was validated using the open-source Visual Manipulation Relationship (VMR) V2 Dataset [41]. This dataset contains an object's stacking environment as shown in Figure 9. Similar to our first use case, the main task was to produce manipulation plans for the intended goal object.

## 5. Experimental results

All the models that were trained and tested in the following experiments were written in Pytorch and PyG with a GPU acceleration. The robot was controlled using the Robotics Operation System (ROS) and MoveIt software.

### 5.1. Human intention understanding and planning

In order to ensure that our framework can efficiently detect human assembly actions and therefore update the semantic planner accurately, we first compare the assembly detector with three baseline methods including:

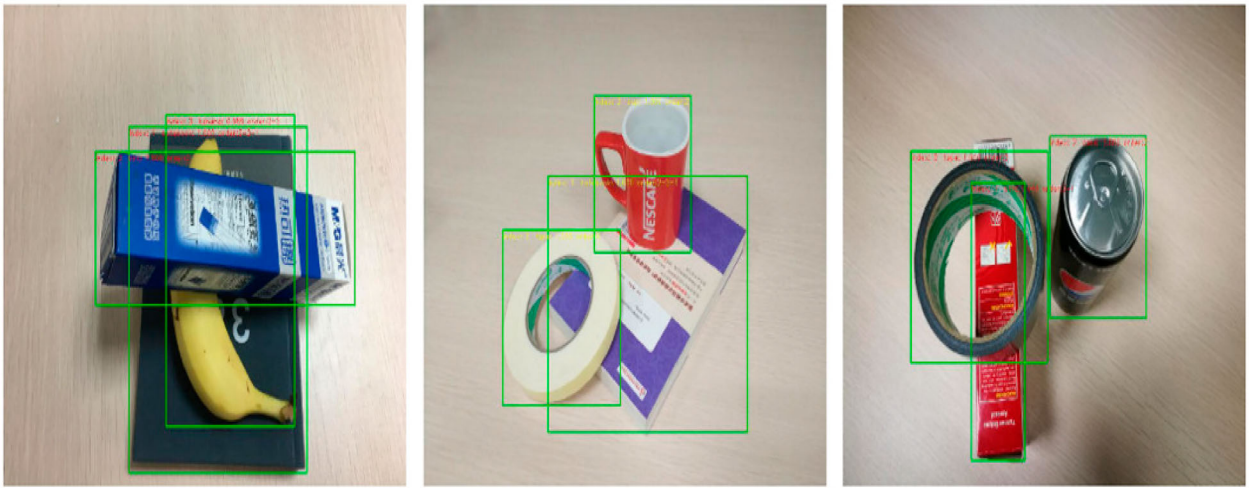


**Figure 9.** Examples from VMR dataset for stacking environment [51].

(1) **CNN+LSTM**: The LSTM-FCN is replaced by a simple LSTM to process the hand motions;
(2) **Hand-centric CNN**: The assembly actions are classified with only spatial hand crops via the VGG16 backbone;
(3) **Long-Term Recurrent Convolutional Networks (CNN-LSTM)** [52]: which first processes the image sequences through VGG16 followed by LSTM for processing the extracted feature sequences.

Figure 10 presents the training and testing outcomes with a training-testing ratio of 0.8:0.2. In these comparisons, our proposed method demonstrates better performance. The integration of LSTM-FCN enhances the detector's ability to handle noisy data. In contrast, relying solely on hand spatial information, as seen in the **Hand-centric CNN** experiment, proves less effective in action classification. Similarly, the **CNN-LSTM** approach also falls short in delivering accurate results. This is due to the presence of extraneous features in the scene. These findings validate that the amalgamation of hand-centric temporal motion and spatial features significantly augments the accuracy in recognising flexible assembly actions.

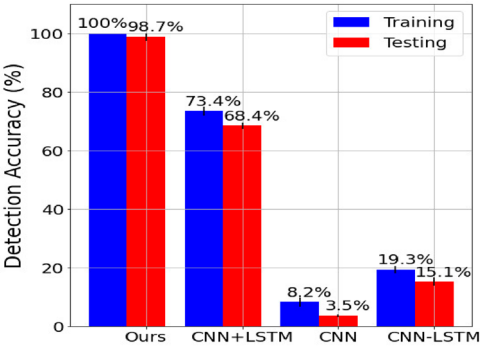



**Figure 10.** Comparison of the proposed action detector with other approaches with average accuracy over five experiments.

### *5.1.1. Semantic planner*

The previous studies on task planning in HRC, for example, Bayesian Inference [15], Hierarchical task networks (HTN) [23] and AND/OR graph [19], can produce plans as to which object or robot action should be performed. However, these methods are not suitable in the ATO use case. The reasons are:

(1) The previous algorithms such as Bayesian Inference, often produce one plan conditioned on prior knowledge. On the other hand, the proposed method is adaptable to situations in which actions vary for the robot (i.e. picking one object to multiple positions).
(2) There are approaches with symbolic representations, for instance, HTN and AND/OR graph, that can offer feasible plans under one final goal, while our proposed approach can recognise and work for various goals.
(3) More importantly, this work is dedicated to releasing the burden of designing the task rules manually in the ATO problem. Through the simulation experiment, it is reported that the graph-based approach is generalisable to unseen human action sequences, which means the planner is capable of producing new plans.

In this assembly framework, the semantic planner can be involved at any stage, leading to $(m-1)C_m$ different possible human action sequences for assembling a single final product. For $z$ distinct goal configurations, this results in a total of $(m-1)C_m \times z$ potential scenarios. In this study, the model is trained using a subset

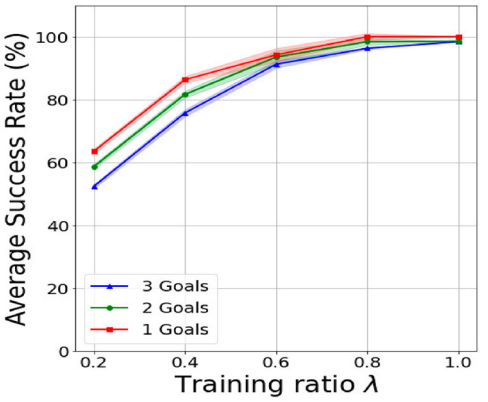



**Figure 11.** The average performance of the Semantic Planner over the training ratio.

of data randomly selected from all demonstrations, with the proportion determined by the training ratio $\lambda$. The experiments include the assembly of one, two, or three customised products. In order to evaluate the model, the Success Rate (SR), which is the ratio of successful trials to the total number of possible scenarios, is employed. A trial is deemed successful if the predicted $\mathcal{P}_g^{pred}$, $\mathcal{P}_o^{pred}$, and the semantic description *txt* for each object *o* are all accurate.

Figure 11 demonstrates the GNN-based model's generalisability to unseen human action sequences. For instance, the model achieves an average SR of 96.3% when trained with 143 demonstrations out of a total of 186 scenarios involving 3 different final goal configurations. The primary instances of failure occur in situations where the same object is assembled to the same position under different goal configurations, such as assembling $o_1$ to position $p_2$ for both $g_0$ and $g_1$.

### 5.1.2. Overall performance

The whole system's performance was assessed in real-world experiments as shown in Table 1. Our system detected human actions every 45 frames with the camera running over 15 FPS similar to the training phase. The average action detection time was 95*ms* and it cost an average of 2.1 *ms* to plan for each object with GPU acceleration.

In our experiments, the human worker could produce random action sequences. Due to the limitation in the current action detector, fast motions were not well detected. Such errors will cause the decreasing accuracy of the semantic planner. For each different assembly goal, we carried out 20 experiments over five times.

Figure 12 (see video at: https://www.youtube.com/watch?v=IBFLMZrSFL0) shows that our proposed system can adaptively guide and assist the human assembly.

**Table 1.** The average detection accuracy of Action Detector (AD), the success rate of the **Semantic Guidance** (SG) for human workers and the **Semantic Control** (SC) for the robot produced by the semantic planner.

| Number of goals | AD (%) | SG(%) | SC(%) |
|---|---|---|---|
| 1 | 98.3 ±1.2 | 97.3 ±1.2 | 95.3 ±1.2 |
| 2 | 97.7 ±2.1 | 94.3 ±1.1 | 93.2 ±1.3 |
| 3 | 95.5 ±1.7 | 93.8±1.6 | 91.8 ±1.6 |

Moreover, as stated before, in different goal configurations, a part can occupy the same assembly position. This may lead to confusion for the semantic planner. However, with the further actions of human workers, this error can be eliminated as shown in Figure 13. This demonstrates that our proposed system can dynamically correct the wrong predictions by actively updating the assembly graph.

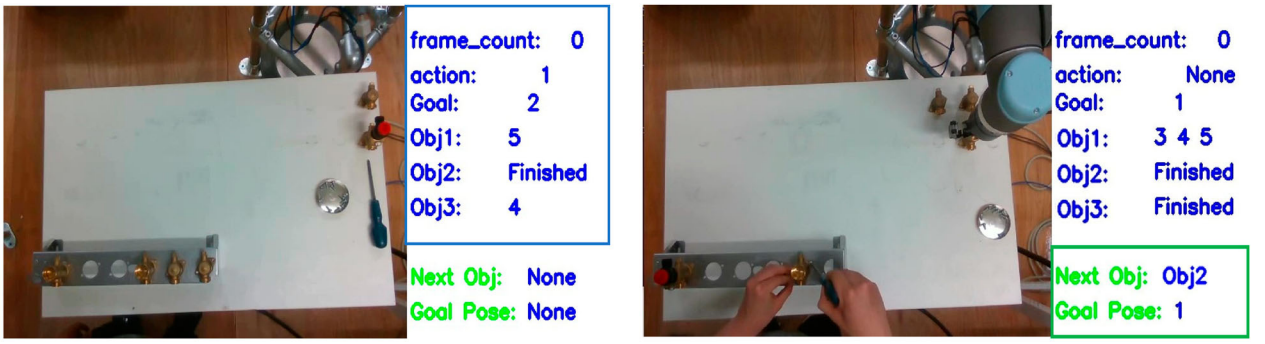
## 5.2. Sequential object handling

The following set of experiments are designed to demonstrate the generalisability of the proposed visual manipulation framework. Considering that human requirements can vary from one product type to another, for each training image, there will be randomly selected object goals or requirements. This means that not all the object types shown in the image scene will be annotated with manipulation plans. Furthermore, the trained framework will face various unseen scenarios including unseen goals or totally unseen images during testing.

### 5.2.1. Industrial parts handling in cluttered environment

This experiment makes use of industrial parts in a cluttered environment. In this experiment, there are 427 training samples with randomly selected goal objects and 50 testing images with various types and numbers of objects. We first conduct ablation studies to understand the effectiveness of each components that include:

(1) **Graph Neural Networks with Attention and without Weighted Edges (GNN_Att w/o WE)**: The WL-GNN processes the nodes' features without the proposed weighted edges (WE) (i.e. all weights equal to 1).
(2) **GNN_LSTM**: The LSTM_Att is replaced by a simple LSTM with the same parameter settings.
(3) **GNN_DF_Att**: In order to demonstrate the importance of processing visual features when orientation varies, we used a Distance Filter (DF) to filter the irrelevant object pairs based on a pre-defined Euclidean distance threshold (0.16 in this case).




(a) Semantic Guidance for unassembled objects (b) Semantic Control to assist human

**Figure 12.** Real-time HRC with our proposed system. (a) Semantic Guidance for unassembled objects and (b) Semantic Control to assist human.

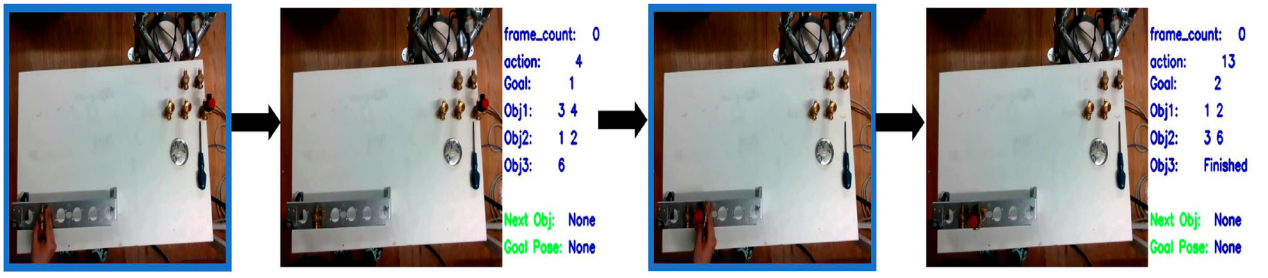



**Figure 13.** This figure demonstrates that our proposed system can dynamically construct the graph based on detected human actions. The blue squared pictures are the last detected frame.

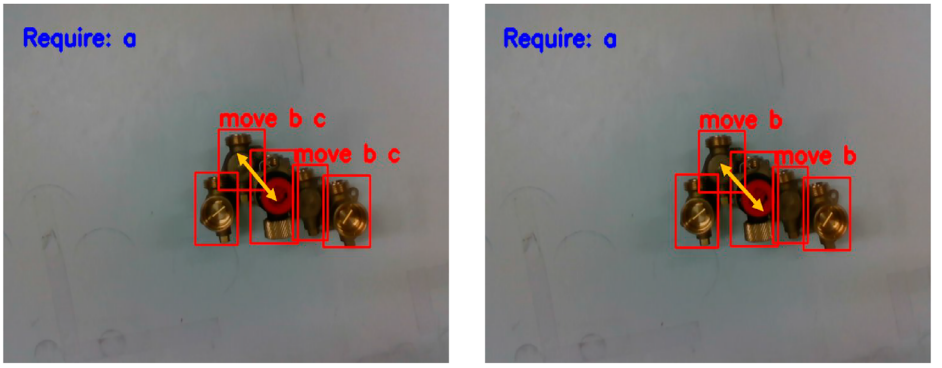
Furthermore, we compare our approach with **GVMRN_RF** [40], where they use Graph Convolutional Neural Network (GCN) to perform the relationship reasoning between each object pairs. Unlike our proposed framework, their work extracts union box features, which cover two overlapped objects' bounding boxes, as node features. As a result, the relationship reasoning can be termed as a multi-label classification problem with labels such as 'under', 'above', 'help' and 'no relation'. In order to filter out the irrelevant objects, they further proposed a Relation Filter (RF) based on the intersection area and a pre-defined distance threshold.

Moreover, in order to ensure fairness and consistency in the comparison criteria, only the relationship nodes relevant to the required objects have been trained. Since their approach is not able to directly produce a grasping solution, we only assess the label prediction accuracy during testing. Also, in order to further compare with **GVMRN_RF**, the WL-GNN in our proposed framework is replaced by GCN. We refer to this as **GCN_Att**.

In this work, the object-based accuracy (OA) is first assessed as if the predicted graspability and generated solutions for ungraspable objects are correct based on one single image. There are three scenarios being assessed as randomly requiring 1,2 or 3 types of objects. Considering the geometric shapes, the grasping solutions can vary. For example, the requirement grasp object 'a' is the most difficult case whereby the accuracy is around 0.81. This is because it is the smallest part and therefore the generated solution could include both 'b' and 'c' objects as shown in Figure 14a.

There is also a special case when two of the same type of objects get too close, the generated solution will ask for 'help' from the human co-worker as the robot is not capable of grasping it by removing other objects. Figure 15(a) illustrates successful test case where the most graspable object based on human requirements is found.

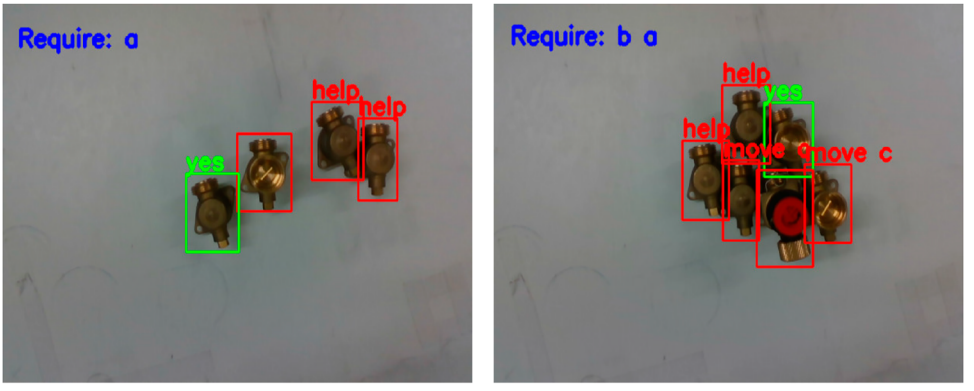
Table 2 describes the OA in a single image for different approaches. It indicates the weighted edges can improve the aggregation process of the GNN and therefore provide more effective features for decoding in the study of **GNN_Att w/o WE**. Furthermore, the LSTM_Att shows a slight improvement when compared to simple LSTM. For the results in **GNN_DF_Att**, it is found that a pre-defined distance threshold can not fully describe the spatial relationship between each object. Figures 14(b) and 15(b) describe a failure testing case with DF, where the distance between object 'a' and object 'c' is greater than the pre-defined DF. This will cause incomplete solution generation.




(a) Grasping solution generated by the proposed **GNN_Att** (Ours)

(b) Grasping solution generated by the **GNN_DF_Att**

**Figure 14.** Our proposed framework's performance on one single image. (a) Require object a and (b) Require object a,b.




(a) Require object a

(b) Require object a,b

**Figure 15.** Comparison between our proposed framework and **GNN_DF_Att**. (a) Grasping solution generated by the proposed **GNN_Att (Ours)** and (b) Grasping solution generated by the **GNN_DF_Att**.

**Table 2.** OA comparison between different approaches in single image.

| | Train_Acc | One type | Two type | Three type |
|---|---|---|---|---|
| GNN_Att (**Ours**) | 1 | 0.91 ±0.018 | 0.891 ±0.021 | 0.882±0.016 |
| GNN_Att w/o WE | 0.97 | 0.634 ±0.013 | 0.586 ±0.004 | 0.545 ±0.003 |
| GNN_LSTM | 1 | 0.89 ±0.007 | 0.872±0.017 | 0.864±0.014 |
| GNN_DF_Att | 1 | 0.881 ±0.015 | 0.843 ±0.008 | 0.829 ±0.005 |
| GVMRN_RF | – | 0.393±0.023 | 0.467±0.019 | 0.544±0.018 |
| GCN_Att | 0.738 | 0.421±0.025 | 0.621 ±0.022 | 0.692 ±0.022 |

Note: Meanwhile, the performance based on different numbers of human requirements has been shown. **GNN_Att** stands for our proposed algorithm. **Train_Acc** stands for the graspability classification in training sets.

In the comparison of **GVMRN_RF** and **GCN_Att**, we found that their architecture has worse performance in terms of generalisability. This is because the relationship node feature was totally new during testing. As a result, it could not efficiently predict labels. Also, our proposed framework only modifies parts of the node features (i.e. goal information) when a new requirement is set. Furthermore, it was found that GCN could not extract informative spatial features. The reason for this is that GCN performs only average aggregation over the node embeddings and its neighbours.

**Table 3.** The accuracy for iterative visual grasping despite the object detection error.

| | 3 objects | 4 objects | 5 objects | 6 objetcs |
|---|---|---|---|---|
| Single image | 0.962 | 0.901 | 0.864 | 0.806 |
| robot grasping | 0.91 | 0.874 | 0.782 | 0.684 |

To further assess our proposed framework's ability to iteratively generate grasping solutions, two more experiments were conducted as shown in Table 3.

As mentioned above, for one initial image, if none of the required objects is graspable, the proposed system

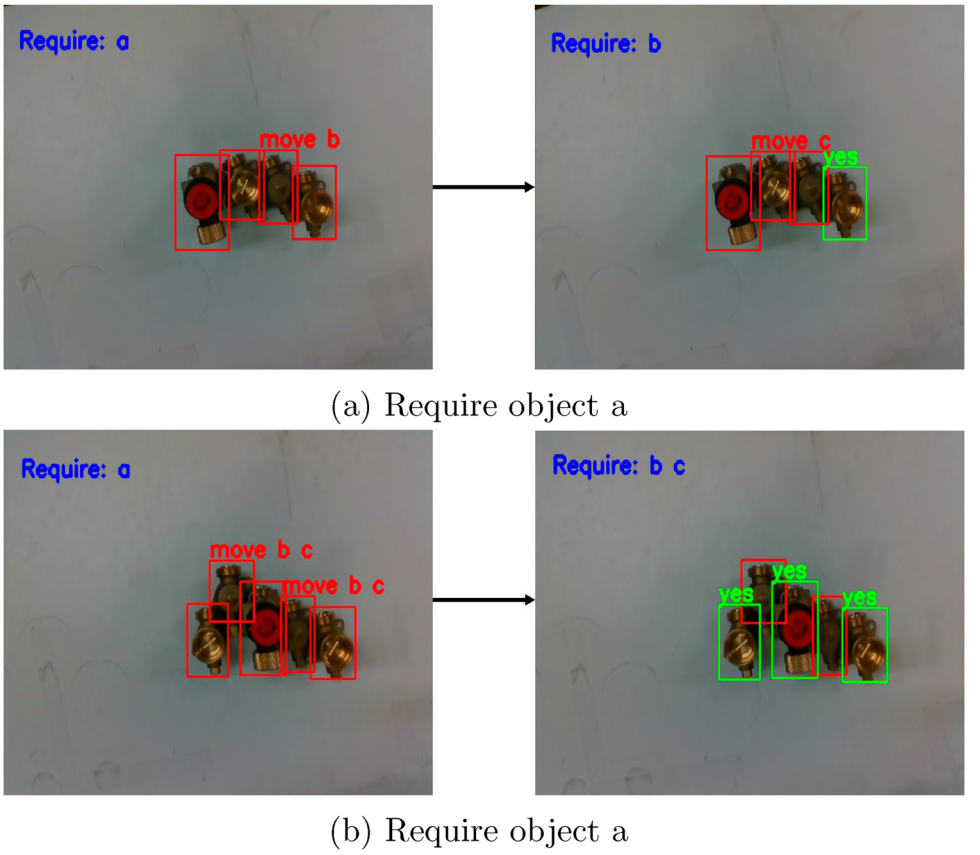



**Figure 16.** Iterative grasping solution generation from the initial image. (a) Require object a and (b) Require object a.

aims to iteratively propose the most graspable object as the solution. This is illustrated in Figure 16(a,b). As a result, our first experiments were to manually assess if the grasping solutions are corrected for the goal objects from one initial image. There were 10 testing cases for different numbers of objects. As shown in Table 3, the accuracy drops down with an increase in the number of objects.

Secondly, the proposed framework was integrated with a real robot arm for real-world grasping tasks. There were 10 testing cases for each of the objects used. As shown in Figure 17, our proposed system has the capability of dealing with varying number of objects, different object types as well as different task lengths depending on the initial scene or starting point. Furthermore, it also has the ability to generate new solutions for new unseen images, which can occur after robot grasping. However, it has been noticed that the gripper may accidentally collide with the surrounding objects and may lead to a decrease in performance. This reveals one drawback of our proposed system: it is not capable of recognising visual features with too-large orientation variations in the object and these variations can lead to infinite possible scenarios.

#### *5.2.2. Daily life objects in stacking environment*

In this experiment, our approach is further validated with the Visual Manipulation Relationship Dataset (VMR) [51] in comparison with **GVMRN_RF** [40] and an additional CNN-based approach **VMRN [51]** in which they classify the relationship between every possible object pairs through CNN.

Image-based accuracy (IA) is assessed in this experiment. However, unlike our proposed framework, **GVMRN_RF** and **VMRN** only classify the relationship between different parts. In regards to robot manipulation, they need to traverse the neighbour objects related to the target objects. As a result, in order to compare IA, with our proposed framework, we assessed if the graspable object is classified correctly and if solutions for the ungraspable object are correct.

Table 4 describes the IA performance in the testing dataset. The testing dataset contains 31 types of different daily life objects and the maximum number of objects in one image is 5. As the table shows, the proposed framework performs worse than the baseline algorithms.

In order to investigate the reason behind this issue, we carried out further studies. We found that the main failure case happens as the generated solutions produce

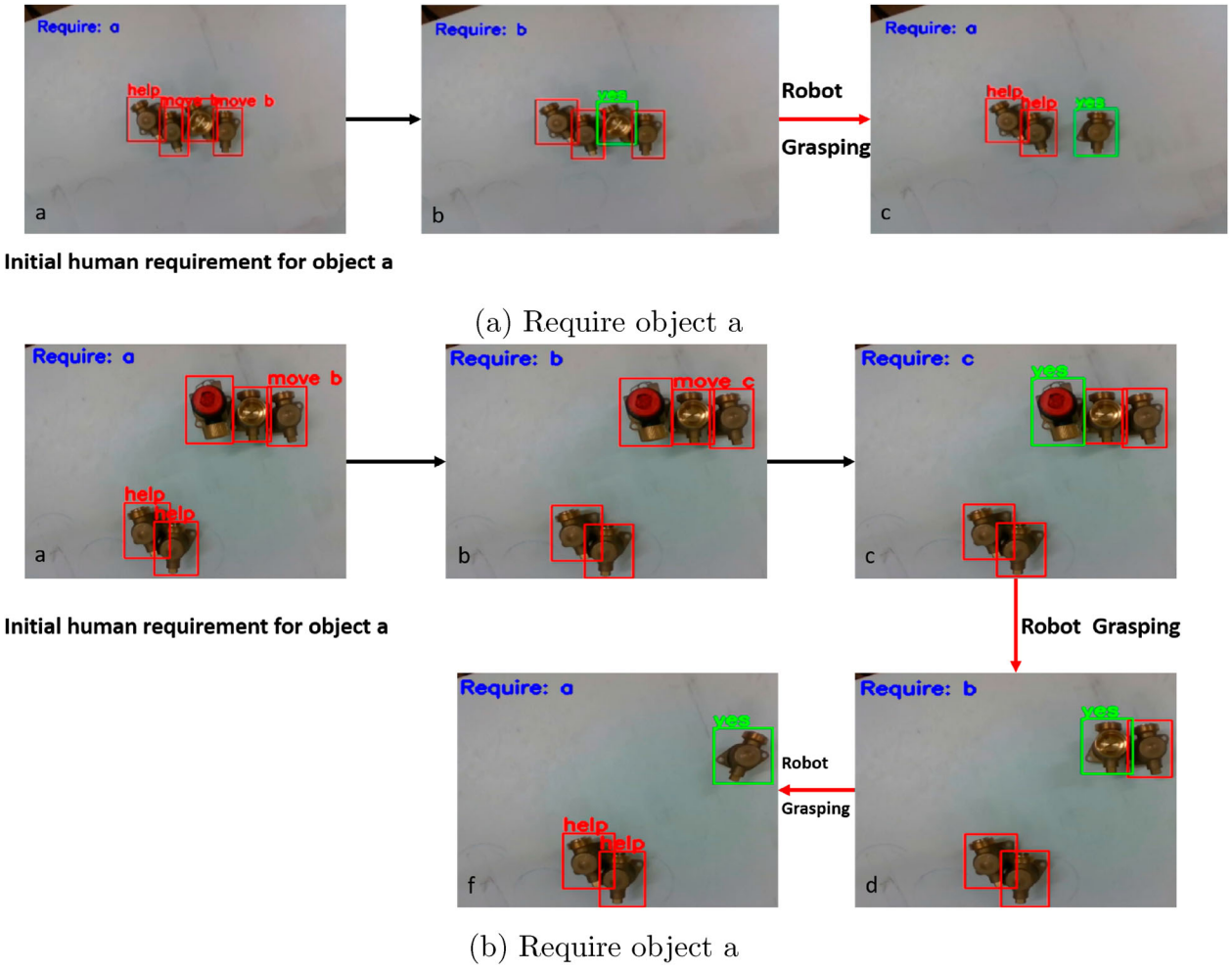



**Figure 17.** Physical robot experiment scenes for dealing variations including object numbers, object types, and task lengths. As the figure shows, the framework allows the robot to remove the graspable obstacle objects until it identifies that the objects required by the human are graspable. The experimental results can be seen at https://youtu.be/1_vid7AGsw0?si = Fa44yIFvUUTDDaxK. (a) Require object a and (b) Require object a.

**Table 4.** Image-based accuracy for different methods in VMR dataset.

| | VMRN | GVMRN_RF | GNN_Att |
|---|---|---|---|
| IA | 0.658 | 0.688 | 0.62 |

the wrong name or label for the objects. Figure 18 illustrates the IA performance on both GNN classification and solution generated from LSTM_Att over the increasing number of types of objects. It was found that GNN performance does not decrease by a large amount while the LSTM_Att's performance drops. Therefore, it was suspected that the main reason for poor performance is that LSTM_Att can not decode the features efficiently as the variety of objects grows. Moreover, considering that less object orientation can happen in this dataset, the RF method in **GVMRN_RF** is more effective in aggregating relevant object information.

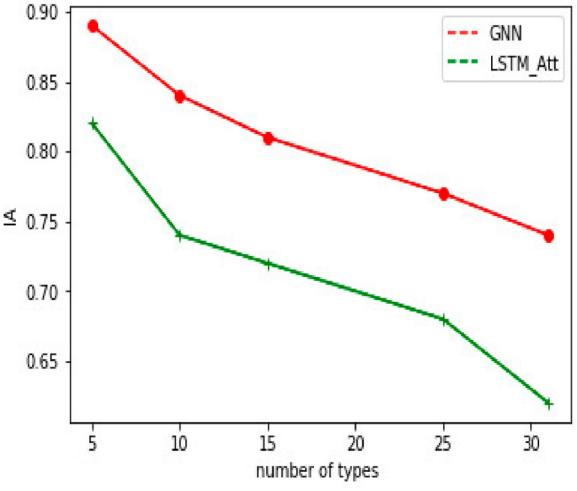



**Figure 18.** The average performance of GNN and LSTM_Att over the increasing number of types of objects.

## 6. Conclusion

Our work advances the field of Human-Robot Collaboration (HRC) by introducing a new vision-based Graph Neural Network (GNN) integrated with Long Short-Term Memory (LSTM) architecture for dynamic robot task planning during Human-Robot-Collaboration scenarios. The proposed GNN+LSTM framework merges temporal and spatial data to classify human actions and facilitate intuitive robot-human interactions while reducing training labelling efforts. By converting assembly actions into a graph format, our framework can deduce human goals and plan collaborative strategies. Our framework not only supports the planning of assembly processes but also aids human operators thereby potentially reducing fatigue during extensive tasks.

Furthermore, our framework showcases adaptability to varying human actions, as evidenced by its successful application in two distinct use cases. The first use case involves vision-informed human intention understanding and planning in which our framework accurately interprets human assembly strategies under different end-goal configurations and produces detailed as well as responsive robot action plans. This capability is critical for ensuring seamless robot-human interactions in dynamic assembly environments. The second use case involved sequential object handling in cluttered environments in which our architecture managed the complexities of object manipulation in cluttered spaces. By encoding the spatial scene into a graph and iteratively generating grasping plans, the robot was able to handle required objects even when they were occluded or surrounded by other items. Our work innovates by leveraging both visual and symbolic data to determine an object's graspability as well as prioritising actions based on spatial relationships. The proposed iterative grasping solution, validated through real-world experiments, demonstrates the system's robustness and practical applicability.

Despite these advancements, challenges remain, particularly with rapid human movements and complex object orientations due to the reliance on 2D imaging. Future work will explore the use of high-speed cameras and enhanced decoding models to address these limitations and improve performance across diverse settings.Overall, this research marks a step forward in integrating GNN and LSTM for adaptive task planning in HRC scenarios thereby offering a versatile and efficient solution for modern manufacturing challenges.

## Disclosure statement

No potential conflict of interest was reported by the author(s).

## Funding

We would like to acknowledge the support of the Engineering Physical Sciences Research Council (EPSRC) for funding this research [grant number EP/W014688/2] as well as the contribution in kind of Bosch Thermotechnology Ltd, UK and National Natural Science Foundation of China [grant number 62202285].

## Notes on contributors

***Ruidong Ma*** received his B.Eng. degree in electrical and electronics engineering from the University of Liverpool and the M.Sc. degree in human and biological robotics from Imperial College London. He obtained his Ph.D. in robotics from the Department of Automatic Control and Systems Engineering at the University of Sheffield. His research interest includes machine learning in robotics and human-robot-collaboration.

***Yanan Liu*** obtained a Ph.D. in Robotics and Autonomous Systems from the University of Bristol at UK. He is a Lecturer at the School of Microelectronics, Shanghai University at Shanghai, China and a visiting researcher at the Department of Computer Engineering, University of York, UK. He researches in the areas of in-sensor computing, unconventional computing, embedded vision, robotic perception, machine learning and applications.

***Erich Graf*** received his BSc (Psychology) from the University of Washington and PhD (Vision Science) from the University of California-Berkeley. He is currently an Associate Professor at the University of Southampton. His research interests are in visual perception in humans and robots.

***John Oyekan*** is a Chartered Engineer (CEng) and Senior Lecturer in Human-Centred AI for Autonomous Manufacturing in the Department of Computer Science. He was previously a Lecturer in Digital Manufacturing at the University of Sheffield. He received a Ph.D degree in Computer Science and Electronic Engineering from the University of Essex as well as a MSc Robotics and Embedded Systems from same. Prior to the University of Sheffield, he was an Engineer at the Manufacturing Technology Centre in Coventry were he developed software architectures and algorithms for Autonomous Systems.

## References

[1] Webster W, Caspi L. What is market fragmentation? 2023. [cited 2024 July 9].
[2] Park J, Jun MB, Yun H. Development of robotic bin picking platform with cluttered objects using human guidance and convolutional neural network (CNN). J Manuf Syst. 2022;63:539–549. doi: 10.1016/j.jmsy.2022.05.011
[3] Li X, Cao R, Feng Y, et al. A sim-to-real object recognition and localization framework for industrial robotic bin picking. IEEE Robot Autom Lett. 2022;7(2):3961–3968. doi: 10.1109/LRA.2022.3149026
[4] Turner CJ, Ma R, Chen J, et al. Human in the loop: industry 4.0 technologies and scenarios for worker mediation of automated manufacturing. IEEE Access. 2021;9:103950–103966. doi: 10.1109/ACCESS.2021.3099311

[5] Wang W, Li R, Chen Y, et al. Facilitating human-robot collaborative tasks by teaching-learning-collaboration from human demonstrations. IEEE Trans Autom Sci Eng. 2019;16(2):640–653. doi: 10.1109/TASE.8856
[6] Scarselli F, Gori M, Tsoi AC, et al. The graph neural network model. IEEE Trans Neural Netw. 2009 Jan;20:61–80. doi: 10.1109/TNN.2008.2005605
[7] Huang DA, Nair S, Xu D, et al. Neural task graphs: generalizing to unseen tasks from a single video demonstration. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition; 2019. p. 8557–8566.
[8] Silver T, Chitnis R, Curtis A, et al. Planning with learned object importance in large problem instances using graph neural networks. In: 35th AAAI Conference on Artificial Intelligence, AAAI 2021; 2021. p. 11962–11971.
[9] Lin Y, Wang AS, Undersander E, et al. Efficient and interpretable robot manipulation with graph neural networks. IEEE Robot Autom Lett. 2022;7(2):2740–2747. doi: 10.1109/LRA.2022.3143518
[10] Ma R, Chen J, Oyekan J. A learning from demonstration framework for adaptive task and motion planning in varying package-to-order scenarios. Robot Comput Integr Manuf. 2023;82:102539. doi:10.1016/j.rcim.2023.102539.
[11] Chen Y, Wang W, Krovi V, et al. Enabling robot to assist human in collaborative assembly using convolutional neural networks. In: IEEE International Conference on Intelligent Robots and Systems; 2020. p. 11167–11172.
[12] Zhang J, Wang P, Gao RX. Hybrid machine learning for human action recognition and prediction in assembly. Robot Comput Integr Manuf. 2021 May;72:102184. doi:10.1016/j.rcim.2021.102184.
[13] Lu Y, Liu Y. Egocentric hand-object interaction detection. In: 2022 IEEE Smartworld, Ubiquitous Intelligence & Computing, Scalable Computing & Communications, Digital Twin, Privacy Computing, Metaverse, Autonomous & Trusted Vehicles (SmartWorld/UIC/ScalCom/DigitalTwin/PriComp/Meta), IEEE; 2022. p. 25–32.
[14] Wu D, Shao L. Leveraging hierarchical parametric networks for skeletal joints based action segmentation and recognition. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition; 2014. p. 724–731.
[15] Cheng Y, Sun L, Liu C, et al. Towards efficient human-robot collaboration with robust plan recognition and trajectory prediction. IEEE Robot Autom Lett. 2020;5(2):2602–2609. doi: 10.1109/LSP.2016.
[16] Cheng Y, Zhao W, Liu C, et al. Human motion prediction using semi-adaptable neural networks. Proc Am Control Conf. 2019 Jul;2019:4884–4890.
[17] Moutinho D, Rocha LF, Costa CM, et al. Deep learning based human action recognition to leverage context awareness in collaborative assembly. Robot Comput Integr Manuf. 2023 Oct;80(2022):102449.
[18] Rückert P, Papenberg B, Tracht K. Classification of assembly operations using machine learning algorithms based on visual sensor data. Procedia CIRP. 2020;97:110–116. doi: 10.1016/j.procir.2020.05.211
[19] Zhang R, Lv J, Li J, et al. A graph-based reinforcement learning-enabled approach for adaptive human-robot collaborative assembly operations. J Manuf Syst. 2022 Apr;63:491–503. doi: 10.1016/j.jmsy.2022.05.006
[20] Bruckschen L, Bungert K, Dengler N, et al. Predicting human navigation goals based on Bayesian inference and activity regions. Rob Auton Syst. 2020;134:103664. doi:10.1016/j.robot.2020.103664.
[21] Darvish K, Simetti E, Mastrogiovanni F, et al. A hierarchical architecture for human–robot cooperation processes. IEEE Trans Robot. 2021;37(2):567–586. doi: 10.1109/TRO.2020.3033715
[22] Zhang R, Li X, Zheng Y, et al. Cognition-driven robot decision making method in human-robot collaboration environment. IEEE Int Conf Autom Sci Eng. 2022 Aug;2022:54–59.
[23] Hayes B, Scassellati B. Autonomously constructing hierarchical task networks for planning and human-robot collaboration. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), May 2016, p. 5469–5476.
[24] Cheng Y, Sun L, Tomizuka M. Human-aware robot task planning based on a hierarchical task model. IEEE Robot Autom Lett. 2021;6(2):1136–1143. doi: 10.1109/LSP.2016.
[25] Grigore EC, Roncone A, Mangin O, et al. Preference-based assistance prediction for human-robot collaboration tasks. In: IEEE International Conference on Intelligent Robots and Systems; 2018. p. 4441–4448.
[26] Morrison D, Corke P, Leitner J. Closing the loop for robotic grasping: a real-time, generative grasp synthesis approach. In: Robotics: science and systems; 2018.
[27] Morrison D, Corke P, Leitner J. Learning robust, real-time, reactive robotic grasping. Int J Robot Res. 2019 Jun;39:183–201. doi: 10.1177/0278364919859066
[28] Zhang Z, Zhou C, Koike Y, et al. Single RGB image 6D object grasping system using pixel-wise voting network. Micromachines. 2022;13(2):1–13.
[29] Huang X, Halwani M, Muthusamy R, et al. Real-time grasping strategies using event camera. J Intell Manuf. 2022;33(2):593–615. doi: 10.1007/s10845-021-01887-9
[30] Wang L, Meng X, Xiang Y, et al. Hierarchical policies for cluttered-scene grasping with latent plans. IEEE Robot Autom Lett. 2022;7(2):2883–2890. doi: 10.1109/LRA.2022.3143198
[31] Abi-Farraj F, Pacchierotti C, Arenz O, et al. A haptic shared-control architecture for guided multi-target robotic grasping. IEEE Trans Haptics. 2020;13(2):270–285. doi: 10.1109/TOH.4543165
[32] Garcia-Garcia A, Zapata-Impata BS, Orts-Escolano S, et al. TactileGCN: a graph convolutional network for predicting grasp stability with tactile sensors. Proc Int Jt Conf Neural Netw. 2019 Jul;2019:1–8.
[33] Mo K, Deng Y, Xia C, et al. Learning language-conditioned deformable object manipulation with graph dynamics; 2023.
[34] Lu Y, Chang C, Rai H, et al. Learning effective visual relationship detector on 1 GPU; 2019.
[35] Qi S, Wang W, Jia B, et al. Learning human-object interactions by graph parsing neural networks. Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics). LNCS; 2018, Vol. 11213. p. 407–423.
[36] Radford A, Kim JW, Hallacy C, et al. Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning, PMLR; 2021. p. 8748–8763.

[37] Goodwin W, Vaze S, Havoutis I, et al. Semantically grounded object matching for robust robotic scene rearrangement. In: Proceedings – IEEE International Conference on Robotics and Automation; 2022. p. 11138–11144.
[38] Huang Y, Conkey A, Hermans T. Planning for multi-object manipulation with graph neural network relational classifiers. In: 2023 IEEE International Conference on Robotics and Automation (ICRA), ICRA; 2022. p. 1822–1829.
[39] Ardon P, Pairet E, Petrick RP, et al. Learning grasp affordance reasoning through semantic relations. IEEE Robot Autom Lett. 2019;4(4):4571–4578. doi: 10.1109/LSP.2016.
[40] Zuo G, Tong J, Liu H, et al. Graph-based visual manipulation relationship reasoning network for robotic grasping. Front Neurorobot. 2021 Aug;15:1–12. doi: 10.3389/fnbot.2021.719731
[41] Zuo G, Tong J, Liu H, et al. Graph-based visual manipulation relationship reasoning in object-stacking scenes. In: 2021 International Joint Conference on Neural Networks (IJCNN); 2021. p. 1–8.
[42] Ma R, Chen J, Oyekan J. Graph-based semantic planning for adaptive human-robot-collaboration in assemble-to-order scenarios. In: 2023 32nd IEEE International Conference on Robot and Human Interactive Communication (RO-MAN); 2023. p. 2197–2203.
[43] Lugaresi C, Tang J, Nash H, et al. MediaPipe: a framework for building perception pipelines; 2019.
[44] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. In: 3rd International Conference on Learning Representations, ICLR 2015; 2015. p. 1–14.
[45] Karim F, Majumdar S, Darabi H, et al. LSTM fully convolutional networks for time series classification. IEEE Access. 2017;6:1662–1669. doi: 10.1109/ACCESS.2017.2779939
[46] Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems, Curran Associates, Inc.; 2017, Vol. 30.
[47] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition. Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit. 2016 Dec;2016:770–778.
[48] Morris C, Ritzert M, Fey M, et al. Weisfeiler and leman go neural: higher-order graph neural networks. In: 33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019; 2019. p. 4602–4609.
[49] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. Adv Neural Inf Process Syst. 2017 Dec;2017:5999–6009.
[50] Wu Y, Kirillov A, Massa F, et al. Detectron2. Available from: https://github.com/facebookresearch/detectron2, 2019.
[51] Zhang H, Lan X, Zhou X, et al. Visual manipulation relationship recognition in object-stacking scenes. Pattern Recognit Lett. 2020;140:34–42. doi: 10.1016/j.patrec.2020.09.014
[52] Donahue J, Hendricks LA, Rohrbach M, et al. Long-term recurrent convolutional networks for visual recognition and description. IEEE Trans Pattern Anal Mach Intell. 2017;39(4):677–691. doi: 10.1109/TPAMI.2016.2599174