**Article**

**Status Recognition Using Pre-Trained YOLOv5 for Sustainable Human-Robot Collaboration (HRC) System in Mold Assembly**

Yee Yeng Liau and Kwangyeol Ryu *

Department of Industrial Engineering, Pusan National University, Busan 46241, Korea; yeeyeng85@pusan.ac.kr
* Correspondence: kyryu@pusan.ac.kr

**Abstract:** Molds are still assembled manually because of frequent demand changes and the requirement for comprehensive knowledge related to their high flexibility and adaptability in operation. We propose the application of human-robot collaboration (HRC) systems to improve manual mold assembly. In the existing HRC systems, humans control the execution of robot tasks, and this causes delays in the operation. Therefore, we propose a status recognition system to enable the early execution of robot tasks without human control during the HRC mold assembly operation. First, we decompose the mold assembly operation into task and sub-tasks, and define the actions representing the status of sub-tasks. Second, we develop status recognition based on parts, tools, and actions using a pre-trained YOLOv5 model, a one-stage object detection model. We compared four YOLOv5 models with and without a freezing backbone. The YOLOv5l model without a freezing backbone gave the optimal performance with a mean average precision (mAP) value of 84.8% and an inference time of 0.271 s. Given the success of the status recognition, we simulated the mold assembly operations in the HRC environment and reduced the assembly time by 7.84%. This study improves the sustainability of the mold assembly from the point of view of human safety, with reductions in human workload and assembly time.

**Keywords:** human-robot collaboration; transfer learning; status recognition; mold assembly

---

**1. Introduction**

The use of robots in manufacturing began in Industry 3.0 as industrial robots were introduced for automated mass production. However, there are challenges in expanding industrial robot systems' application in mass personalization. Industrial robot systems can work fast with a low error rate. Still, industrial robots are less flexible and require high-cost reconfiguration to cope with the frequent demand changes in mass personalization production. In contrast, manual systems can adapt to changes with lower investment costs, but human workers tend to become fatigued and have a higher error rate [1]. Therefore, the application of human-robot collaboration (HRC) systems in manufacturing has gained attention in Industry 4.0. HRC systems combine the cognitive ability of humans with the consistency and strength of robots to increase the flexibility and adaptability of an automated system [2]. Besides this, the key enabling technologies in Industry 4.0, such as artificial intelligence and augmented reality, are integrated into the HRC systems to support interaction and collaboration between humans and robots [3]. Although we are still at the stage of realizing Industry 4.0, the term Industry 5.0, which focuses on bringing humans back into the production line and collaboration between humans and machines, has been introduced [4]. Hence, HRC systems are foreseen to be an active research area in Industry 5.0 as well.

This paper focuses on the application of HRC systems in mold assembly. Most molds are still assembled manually, while full automation systems are implemented in various assembly systems, such as for automotive parts [5–8] and electronic parts [9,10]. A full automation assembly cell is insufficiently flexible to cope with the frequent changes in assembly.
low-volume mold assembly production and the wide variety of mold components that vary in weight and geometry. However, musculoskeletal disorders (MSD) caused by heavy part handling and repetitive motion during mold assembly have increased the need for robots in mold assembly [11]. Therefore, we propose the application of the HRC mold assembly cell to overcome the problems in mold assembly. The implementation of collaborative robots can reduce repetitive strain injuries and relieve heavy part lifting for human workers. At the same time, the use of collaborative robots can increase the work consistency and productivity of the assembly cell by integrating computer vision into HRC assembly cells.

This study aims to develop a vision-based status recognition system for a mold assembly operation to achieve a sustainable HRC mold assembly operation by reducing assembly time and improving human working conditions. An assembly operation comprises tasks performed to join or assemble various parts to create a functional model. Each task consists of a series of sub-tasks executed to assemble a specific part at a defined location using a defined tool. The main contributions of this paper are presented as follows. First, we classify the mold assembly tasks into sub-tasks and actions. In this study, the status of a task is represented by the actions involved. For example, the sequence of actions of the screw-tightening sub-task is as follows: picking up the screwdriver, locating the screwdriver on a screw, tightening the screw by rotating the screwdriver, then returning the screwdriver. Second, we develop a vision-based status recognition system that includes part, tool, and action recognition. We apply the transfer learning technique using a pre-trained YOLOv5 model (refer to Section 2.2.2) on the Common Objects in Context (COCO) dataset [12] to recognize mold components and tools. Then, we identify the status of the sub-task by recognizing defined actions. The expected outcome of this paper is a status recognition method to enable robots to assist humans in the HRC mold assembly operation. In most practical applications, human workers control the robot’s execution using a push button. However, with status recognition, they can induce the early execution of the robot’s task before completing the manual task. This minimizes the robot’s idle time and the completion time of the recognized task. Furthermore, we can reduce the production time and the energy consumption to achieve a sustainable HRC mold assembly operation.

The structure of this paper is as follows: Section 2 provides the literature review related to this research, including deep learning-based recognition and transfer learning techniques. Section 3 explains the proposed status recognition system for HRC mold assembly and the methodology. Section 4 describes the experiment and results. Finally, Section 5 concludes our paper and provides the potential future work for our research.

2. Literature Review

2.1. Deep Learning-Based Recognition in HRC Assembly

The Convolutional Neural Network (CNN) is the most common deep learning method used in computer vision tasks. Image classification, object localization, and object detection are the three main computer vision tasks. Image classification seeks to classify the image by assigning it to a specific label [13]. AlexNet [14], ResNet [15], VGGNet [16], Inception Net and GoogleLeNet [17] are the most common CNN architectures that researchers on image classification have implemented. Object localization takes an image with one or more objects as input and identifies the objects’ location with bounding boxes. The combination of image classification and object localization results in object detection or recognition, which identifies the types of classes of the located objects [13]. The most common deep learning-based object recognition models are R-CNN (Region-based Convolutional Neural Network) [18], Fast R-CNN [19], Faster R-CNN [20], Mask R-CNN [21], and You Only Look Once (YOLO) [22–25] models. R-CNN families are two-stage object detectors that extract the regions of interest (ROIs), then perform feature extraction and classify objects only within the ROIs. Hence, two-stage object detectors require longer detection times than one-stage object detectors. YOLO models are one-stage detectors that directly classify and regress the candidate boundary boxes without extracting ROIs. Our study detects the parts and tools required during an assembly operation so as to recognize a task and then
estimates the task’s progress based on the position of parts or tools. Therefore, we focus on object recognition that involves classification and detection tasks.

The training data used in the existing recognition systems were raw data, collected using wearable sensors, and image data, such as an image captured during operation, and images of spatial and frequency domains derived from sensors. Uzunovic et al. [26] introduced a conceptual task-based robot control system that received human activity recognition and robot capability inputs. They recognized the ten human activities in the car production environment based on the data from nineteen wearable sensors on both arms using machine learning models. However, the attachment of wearable sensors on the human worker caused discomfort during the practical assembly operation. Furthermore, deep learning algorithms in computer vision allow us to perform motion recognition better using assembly videos or images. Researchers have developed deep learning-based approaches using images to recognize common tasks based on different recognition algorithms: gestures or motion recognition and combinations of part and motion recognition in manufacturing assemblies [27,28]. Therefore, this study focuses on applying deep learning algorithms to assembly videos or image data for task recognition.

The research on action and phase recognition has been developed and applied widely for common human activities and surgical applications. Still, the related research on manufacturing assembly applications is worth exploring. Wen et al. [29] used a 3D CNN to recognize seven human tasks in visual controller assembly for the learning process of the robot. They separated eleven assembly videos, collected into seven labeled segments representing seven tasks, and performed data augmentation to increase the dataset for training. The accuracy of the task recognition was only 82% because of the small training dataset and the environmental changes during the assembly operation. Wang et al. [28] used two AlexNets for human motion recognition and part tool identification, respectively. They recognized grasping, holding, and assembling motions to identify human intention. For the part tool identification of a screwdriver, small and large parts were the only parts and tools included in this study. However, they only tested the proposed method on a simple assembly that involved a single tool and limited types of parts.

Chen et al. [27] implemented the YOLOv3 algorithm to detect tools for assembly action recognition and the convolutional pose machine (CPM) to estimate the poses and operating times of the repetitive assembly actions. They tested the algorithm on three assembly actions, which were filling, hammering, and nut screwing. They only estimated the operating times using the cycle of action curve, and not the progress or the remaining operating times. Action recognition based on this tool is inefficient in monitoring the assembly progress because different tasks may require the same tool. Chen et al. [30] extended the previous study [27] and proposed a 3D CNN model with batch normalization for assembly action recognition to reduce the environmental effect and improve recognition speed. Besides this, they employed fully convolutional networks (FCN) to perform depth image segmentation, in order to recognize different parts from assembled products for assembly sequence inspection. They recognized parts using computer-aided design (CAD) models instead of original parts, and compared the accuracy and training time for RGB, binary, gray, and depth images. The results show that the RGB image data gave the highest accuracy, but the training time was longer than for the gray images dataset.

The performance of the developed recognition models in the existing research for assembly applications is worse than expected due to the limited dataset and the environmental changes during the assembly operation. Therefore, this study uses a transfer learning technique to overcome these problems.

2.2. Transfer Learning and YOLO Algorithm

2.2.1. Transfer Learning

Transfer learning is a technique that uses the pre-trained model on other large datasets, such as ImageNet [31] or COCO [12], to train the model on custom data for a new but related problem. This technique helps speed up the development and training process
with a small dataset that limits the deep learning model’s capacity to be trained from scratch [32].

Deep learning models with transfer learning have been applied in various fields, such as computer vision and natural language processing. Computer vision includes recognizing objects, activities, and scenes that usually require numerous labeled image datasets. However, it is difficult to obtain large-scale labeled data in most practical applications. This problem can be solved using the transfer technique to transfer the knowledge from a source domain to a target domain. The integration of transfer learning with a pre-trained CNN, such as AlexNet, ResNet, or VGG, often solves vision-based recognition tasks [33]. We can implement transfer learning on a convolutional neural network using two approaches. First, we freeze the convolutional layers and use the pre-trained model as a feature extractor. The second approach is fine-tuning, whereby we freeze the initial layers and unfreeze deeper convolutional layers. The unfrozen convolutional layers are trained to update the weights. If we have limited new data, we can apply the first approach to prevent overfitting. On the other hand, we can use the second approach with larger datasets to train the deeper layers to detect task-specific features [34].

In this paper, we focus on applying transfer learning in object and action recognition in manufacturing assemblies. Židek et al. [35] applied transfer learning to detect assembly parts and product features. They tested two pre-trained models trained on the COCO dataset: Mobilenet V2 and Fast RCNN Inception V2, for screw and nut recognition. To apply transfer learning in assembly action recognition, Liu and Wang [36] implemented a transfer learning-based human poses recognition method in a collision-free HRC system to recognize operator’s poses with low computational expense. Besides this, Tao et al. [37] applied transfer learning to perform real-time operation recognition during desktop CNC carving machine assembly. They chose the pre-trained DenseNet model because it performed the best among the pre-trained VGG, ResNet, and DenseNet models. They used the pre-trained model trained on ImageNet to recognize ten sequential operations and achieved a 95% recognition accuracy. An assembly task contains parts to be assembled, the tool used, and the action. Therefore, Wang et al. [28] implemented two pre-trained AlexNet models trained on ImageNet to recognize three actions (grasping, holding, and assembling) and the part/tool (small, large parts, and screwdriver), respectively. They adapted AlexNet trained on ImageNet because ImageNet contains image categories of human actions and tools related to manufacturing.

The existing research shows that the application of transfer learning on a pre-trained model improved the accuracy, even with a small dataset. In this study, we aim to detect and localize the parts and tools used during the assembly operation. Therefore, we use the YOLO model instead of image classification CNN models.

2.2.2. YOLO Algorithm

An object detection task identifies objects present on an image and determines the location of the identified objects on the image. YOLO is an object detector that detects objects in images and localizes them directly into bounding box coordinates and class probabilities [38]. First, the image is divided into an S × S grid. The grid cell, which consists of the center of an object, is responsible for detecting the object. Each grid cell predicts bounding boxes, the confidence scores of boxes, and the class probabilities of the grid cell containing an object. The first developed YOLO network had 24 convolutional layers followed by two fully connected layers [22]. Redmon et al. [23] introduced a few improvements to YOLOv2. They added batch normalization on all the convolutional layers and used a high-resolution classification network to increase the mean average precision. Besides this, they used the k-means clustering method to cluster bounding boxes so that the grid cell could detect more than one object. YOLOv3 uses a feature extractor network known as Darknet-53 and improves the detection accuracy and speed [24]. However, YOLOv3 performed worse than the previous YOLO version on medium and large objects. Bochkovskiy et al. [25] proposed YOLOv4, consisting of CSPDarknet53 as a backbone network and spatial pyra-
mid pooling (SPP), with PANet as the neck part and YOLOv3 as the head part. YOLOv5 is the latest YOLO version that uses the CBL (Conv2D + Batch Normal + LeakyRELU) module as the basic convolution module and the BottleneckCSP module for feature extraction [39,40]. YOLOv5 includes different models, such as YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x, which differ by the width and depth of the BottleneckCSP module [41].

YOLOv5 is an object detection model trained on the COCO dataset, which contains 80 classes and more than 200,000 labeled images. This study applies transfer learning on the pre-trained YOLOv5 model for part, tool, and action recognition in the context of status recognition in the HRC mold assembly operation.

3. Status Recognition for HRC Mold Assembly Operation

Figure 1 illustrates the proposed conceptual framework of status recognition. An assembly operation consists of tasks for joining or assembling various parts to create a functional model. We define a task as a series of sub-tasks executed to assemble a specific part at the designated item using a defined tool. Thus, we decompose the assembly operation into tasks and sub-tasks and define actions that represent the status of a sub-task. Status recognition identifies an assembly task based on the recognized unique part, and recognizes the status of the task based on the actions that are decomposed from sub-tasks.

![Figure 1. Proposed status recognition model for HRC system.](image)

3.1. Decomposition of Mold Assembly Operation

This paper focuses on a two-plate mold assembly operation that consists of core and cavity sub-assemblies. Table 1 lists the sixteen mold assembly tasks and each corresponding part. Each task assembles a unique mold part and joining part, such as screws and pins. Since the unique part assembled in each task is not repeated in other tasks, we can recognize a task by recognizing the corresponding unique part.

In this paper, we define a task as assembling a component consisting of a series of sub-tasks to assemble and join components. We also categorize sub-tasks in mold assembly into nine categories [11]. Table 2 lists the tools used in each sub-task. We must perform a series of sub-tasks on the component and corresponding joining components, such as screws and pins, to complete a task. We further decompose these sub-tasks into a series of actions for status recognition purposes. Mold assembly requires two types of tools, which are the hammer and hex-keys. Some sub-tasks need to be executed with a tool or without any tool. For sub-tasks that require a tool, we must include the actions to handle the tool.
Table 1. List of tasks in a two-plate mold assembly.

| No. | Task                          | Part                          |
|-----|-------------------------------|-------------------------------|
| 1   | Prepare A side plate          | A side plate                  |
| 2   | Assemble sprue bushing        | Sprue bushing                 |
| 3   | Assemble top clamp plate      | Top clamp plate, screws       |
| 4   | Assemble location ring        | Location ring, screws         |
| 5   | Prepare outer B side plate    | Outer B side plate            |
| 6   | Assemble guide pin            | Inner B side, guide pin       |
| 7   | Assemble core                 | Core                          |
| 8   | Assemble ejection pin         | Ejection pin                  |
| 9   | Assemble B side plates        | Screws                        |
| 10  | Assemble ejection plate       | Ejection plate, pin           |
| 11  | Assemble return pin           | Return pin                    |
| 12  | Assemble ejection support plate | Ejection support plate, screws |
| 13  | Assemble space plate          | Space plate                   |
| 14  | Assemble bottom clamp plate   | Bottom clamp plate, screws    |
| 15  | Assemble core plate           | Core plate, screws            |
| 16  | Assemble core and cavity sub-assembly | Sub-assemblies     |

Table 2. Categories of sub-tasks for mold assembly and tool used.

| Code | Description of Sub-Tasks                          | Tool          |
|------|---------------------------------------------------|---------------|
| A    | Lift and position plate with rough tolerance      | No            |
| B    | Lift and position plate with fair tolerance       | No            |
| C    | Lift and position plate with tight tolerance      | No            |
| D    | Pick and locate component with fair tolerance     | No            |
| E    | Pick and locate component with tight tolerance    | No            |
| F    | Pick, locate and insert screw                     | No            |
| G    | Tighten screw                                     | Hex-key       |
| H    | Insert small component with force                 | Hammer        |
| I    | Insert plate with force                           | Hammer        |

The identification of actions plays an essential role in status recognition. Generally, a sub-task starts with a hand approaching the part or tool, and ends with an empty hand leaving the assembly area or returning the tool. Based on the common actions, we can summarize that a sub-task starts when the hand approaches the part and ends when the hand leaves the assembly area. The common actions in a sub-task can be listed as follows:

- Picking or grasping part/tool;
- Positioning part;
- Assembly using a tool, such as tightening a screw or inserting a pin;
- Leaving assembly area with an empty hand.

This study aims to develop a status recognition system for HRC mold assembly based on object and action recognition. The status recognition consists of two stages, as shown in Figure 2. In the first stage, we recognize a task by recognizing parts and tools. In the second stage, we recognize the status of a sub-task based on the executed action. Figure 2 shows an example of the stages of the proposed status recognition model. We decompose the task “Assemble location ring” into two sub-tasks: “Lift and place location ring” and “Insert and tighten the screw”. We recognize the part during the first stage as location ring, and the screws and hex-key as the tools, in order to identify the sub-tasks. Then, we recognize the status based on the actions defined for the sub-task. During the execution of “insert and tighten screws”, the defined action sequence is “insert screws”, “tighten screws” with hex-key, then leave the assembly area. Status recognition plays an essential role in enabling the robot to identify the status of the manual task and execute the subsequent task in any future study.
3.2. Implementation of YOLOv5 and Transfer Learning

In this paper, we use the YOLOv5 model to develop a status recognition model because YOLOv5 has been proven to perform better in detection speed compared to R-CNN families [42,43]. We aim to implement status recognition in real-time task re-assignment and task execution in a future study. Therefore, the fast detection speed of the YOLO model is an important characteristic that enables us to recognize objects and actions in real-time during the assembly operation. Since we do not have a large assembly parts and tool images dataset, we implement a pre-trained YOLOv5 model instead of building a model from scratch. In other words, we apply the transfer learning technique using a pre-trained YOLOv5 model to recognize assembly parts and tools based on small image datasets.

3.2.1. Data Collection and Processing

In this paper, we focus on a two-plate mold assembly operation, as shown in Figure 2. We need three image datasets to train the status recognition model: parts, tools, and hand actions. For the parts, we categorized the mold parts into seven types based on the geometric shape. Besides this, we collected images of tools, such as pins, screws, guide pins, sprue bushings, and location rings, from the internet. The mold assembly operation requires two types of tools, which are hammer and hex-key. Then, we captured images from a YouTube video for actions representing the status of the sub-task during mold assembly [44]. After we gathered the images, we increased the number of images for training by rotating those images 90, 180, and 270 degrees. After collecting the images, we used the LabelImg data annotation tool to label and create annotation files in the YOLO format [45]. Finally, we partitioned the dataset into training and testing sets containing 80% and 20% of the data, respectively. We then implemented $k$-fold cross-validation in the YOLOv5m model to evaluate the effects on model performance. We divided the datasets into five batches (i.e., $k = 5$), with 80% training datasets and 20% validation datasets for each fold.

3.2.2. Transfer Learning

We trained the models using the Windows 10 operating system and the Pytorch 1.7.0 framework with a single NVIDIA GeForce RTX2080Ti GPU. In this paper, we used the YOLOv5 pre-trained models trained on the COCO dataset. We trained the datasets using YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x models to compare the performances of the models. We downloaded the weights obtained from the pre-trained model as the initial weights for training purposes. In the YOLOv5 model, the backbone acts as the feature extractor, and the head locates the bounding box and classifies the objects in each box.
Therefore, we froze the backbone to use the YOLOv5 models as a feature extractor and trained the head using the collected training datasets.

For YOLOv5 training parameter setting, we set the image size to 640 \times 640 because the images collected from the internet have different sizes close to 640 \times 640 pixels. We trained the models by employing different batch sizes and numbers of epochs with early stopping conditions. We obtained the best precision and weight from trial-and-error experiments by setting the batch size as 8 and using 600 epochs, with a learning rate = 0.01.

4. Results and Discussion

4.1. Comparison and Results

In this section, we compare the performances of different pre-trained YOLOv5 models and two conditions of transfer learning. The four YOLOv5 models included in the comparison are YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. We compared the performances of models with freezing of the backbone (F = 10) and without freezing (F = 0) any layers. We evaluated the performance based on mean average precision (mAP). The mAP value is calculated by taking the mean average precision over all classes or the overall intersection over union (IoU) thresholds. We evaluated the performance of the recognition models based on the mAP value. The higher the value, the better the average detection accuracy [40]. Tables 3 and 4 compare the mAP values and inference times of four different YOLOv5 models with pre-trained weights, respectively.

Figure 3 shows the result of tool detection, which are hammer and hex-key. Some examples of part detection are shown in Figure 4. Figure 5 illustrates the status recognition of the “Assemble location ring” task (Figure 2), which consists of three actions: “Position plate”, “Insert screw”, and “Tighten screw.” Figure 5d shows the completion of the “Assemble location ring” task, where no action is detected. Table 5 compares the performances of the pre-trained YOLOv5 models with and without freezing of the backbone layers in order to detect the parts, tools, and statuses of the “Assemble location ring” task. We evaluated the performances of the YOLOv5 models by comparing the class probabilities of the statuses and parts/tools detected. The values in Tables 5 and 6 indicate the class probabilities. As regards YOLOv5s (F = 10) in Table 5, the value of 0.71 indicates that the YOLOv5s model without freezing detected “Position plate” status with a probability of 0.71. However, the YOLOv5s (F = 10) model could not detect the hex-key tool, indicated by “X”. We set the confidence threshold to 0.4, meaning any class probability lower than 0.4 is considered as “not detected”, which is indicated using an “X” in Tables 5 and 6. All the models in Table 5 were trained using image datasets that consist of all parts, tools, and action classes. Then, we ran inferences based on the weight trained (W_{all}). In addition, the image dataset for action classes was smaller than that for part and tool classes. Hence, we trained the YOLOv5m model without freezing again using a dataset consisting of action classes only, then performed inference using weights trained using all classes and actions classes only (W_{all}&W_{action}). As mentioned in Section 3.2.1, we trained YOLOv5m using the k-fold cross-validation method, where k = 5. Thus, we used the weights obtained from 5-fold cross-validation training (W_{k=5}) to evaluate the effect of cross-validation in the inference. Table 6 compares the recognition ability of the YOLOv5m model without freezing using different weights.

Table 3. Comparison of mAP score among different YOLOv5 models.

| Model     | Without Freeze (F = 0) | Freeze Backbone (F = 10) |
|-----------|------------------------|--------------------------|
| YOLOv5s   | 0.841                  | 0.846                    |
| YOLOv5m   | 0.849                  | 0.819                    |
| YOLOv5l   | 0.848                  | 0.837                    |
| YOLOv5x   | 0.856                  | 0.854                    |
**Table 4.** Comparison of average inference time (in seconds) among different YOLOv5 models.

| Model    | Without Freeze (F = 0) | Freeze Backbone (F = 10) |
|----------|------------------------|--------------------------|
| YOLOv5s  | 0.0148                 | 0.0162                   |
| YOLOv5m  | 0.0253                 | 0.0283                   |
| YOLOv5l  | 0.0271                 | 0.0280                   |
| YOLOv5x  | 0.0354                 | 0.0346                   |

**Figure 3.** Tool detection: hammer and hex-key.

**Figure 4.** Part detection: screws, guide pin, sprue bush, and location ring.

**Figure 5.** Status recognition of “Assemble location ring” using pre-trained YOLOv5l model without freezing. The colored boxes show the parts and status recognition results: (a) position plate (magenta); (b) insert screw/pin (teal); (c) tighten screw (orange) and hex-key (purple); (d) location ring after assembled (yellow).

**Table 5.** Performance comparison of detecting parts, tools, and statuses during “Assemble location ring” sub-task using different YOLOv5 models (F = 0: without freezing; F = 10: freeze backbone).

| Status       | Part/Tool     | YOLOv5s | YOLOv5m | YOLOv5l | YOLOv5x |
|--------------|---------------|---------|---------|---------|---------|
|              | F = 0 | F = 10 | F = 0 | F = 10 | F = 0 | F = 10 | F = 0 | F = 10 |
| Position plate | X     | 0.71   | 0.71   | X      | 0.65  | 0.67  | 0.68  | 0.55  |
| Insert screw  |       | 0.71   | 0.86   | 0.86   | 0.58  | 0.81  | 0.83  | 0.81  |
| Location ring |       | 0.84   | 0.84   | 0.90   | 0.56  | 0.84  | 0.89  | 0.77  | 0.78  |
| Tighten screw |       | X     | 0.40   | 0.47   | 0.60  | 0.84  | 0.63  | 0.68  | 0.65  |
| Hex-key      |       | X     | X     | X      | 0.46  | X     | 0.56  | X     |
| Location ring |       | X     | X     | X      | 0.74  | 0.81  | 0.73  | X     |
### Table 6. Performance comparison of detecting parts, tools, and statuses during “Assemble location ring” sub-task using different trained weights ($W_{all}$: using weights trained using all classes; $W_{action}$: weight trained using actions classes only; $W_{k=5}$: weight trained using 5-fold cross-validation).

| Status          | Part/tool     | YOLOv5m F = 0 | $W_{all}$ | $W_{all} & W_{action}$ | $W_{k=5}$ |
|-----------------|---------------|--------------|-----------|-------------------------|-----------|
|                 | Inference time (seconds) |              |           |                         |           |
| Position plate  | 0.71          | 0.71         | 0.79      |                         |           |
| Insert screw    | 0.86          | 0.80         | 0.90      |                         |           |
|                 | Location ring | 0.90         | 0.90      | 0.91                    |           |
| Tighten screw   | 0.47          | 0.47         | 0.7       |                         |           |
|                 | Hex-key       | 0.57         | 0.81      |                         |           |
| Location ring   | X             | X            | 0.67      |                         |           |

### 4.2. Discussion

From Table 3, we see that the pre-trained YOLOv5x without freezing model has the best average mAP score (85.6%) compared with other YOLOv5 models. All the models except YOLOv5s performed better without freezing layers than with freezing the backbone. As mentioned in Section 3.2.2, the backbone acts as the feature extractor, and the head locates bounding boxes and classifies the objects. This result shows that training the backbone enables the larger YOLOv5 models to extract the features of new datasets before locating the bounding boxes and classifying the objects.

We set the confidence threshold as 0.4 and used the best weight from training to run the inference for all models. The YOLOv5l models both with and without freezing can recognize all the parts, tools, and statuses (Table 5). The YOLOv5s model achieved the fastest average inference time, 0.0148 s, but it could not detect some statuses of the sub-tasks. The YOLOv5x without freezing model achieved the best mAP score (85.6%) and could detect all parts and statuses, but it had a longer inference time, 0.035 s. Compared to the YOLOv5x model, the YOLOv5l without freezing model achieved a lower mAP score (84.8%) but had the fastest inference time, 0.0271 s. The inference time of all models was less than 0.04 s. The quickest action execution time of a mold assembly operation based on the simulation was 5 s. We tested the YOLOv5l model’s ability to recognize the status of “Assemble return pin” and “Assemble ejection plate” tasks so as to show the model’s compatibility with other tasks. The two statuses of the “Assemble return pin” task are “Insert pin” and “Hammering”, as shown in Figure 6. Figure 7 shows the status recognition of “Assemble ejection plate”, which consists of “Position plate”, “Insert screw”, and “Tighten screw”. Therefore, we can conclude that the pre-trained YOLOv5l without freezing model performed the best, and can be implemented in practical HRC assembly operations for task and status recognition.
In this study, we mixed images from the internet and images from an assembly video to train the model. Our collected datasets had different backgrounds, and the number of images for each class was uneven. Thus, these limitations may be why the mAP score lower than 90% for all the models. Smaller YOLOv5 models, such as YOLOv5s and YOLOv5m, failed to detect some actions when using images from assembly videos because of the lower number of action images, different scenes, and environmental changes. The number of images available for parts/tools classes was more than that available for actions classes. We trained the YOLOv5m without freezing model separately using only the action images, different scenes, and environmental changes.

Figure 6. Status recognition of “Assemble return pin” using pre-trained YOLOv5l model without freezing. The colored boxes show the parts and status recognition results: (a) insert screw/pin (teal) and pin (green); (b) hammering (dark pink) and pin (green).

Figure 7. Status recognition of “Assemble ejection plate” using pre-trained YOLOv5l model without freezing. The colored boxes show the parts and status recognition results: (a) position plate (magenta); (b) insert screw/pin (teal); (c) tighten screw (orange); hex-key (purple).
basic models, especially when using 5-fold cross-validation. We aim to implement this study in real-time assembly operations that recognize tasks and statuses as fast as possible. Thus, we will collect images from an HRC mold assembly operation for model training to improve recognition accuracy in the future.

Our previous study developed a task allocation model for HRC mold assembly composed of one human and two robots with flexible collaboration mode [46]. Based on the result of task allocation in the previous study, the robot tasks followed by manual tasks were determined as “pick and place” and “screw tightening.”

In the simulation, we divided robot tasks into pick, move, and place parts. The pick and move motions did not interfere with the human at the assembly area. In other words, robots can start to pick and move a part for the next task after the human worker has moved a part of the current task, even when the human worker is assembling a part in the assembly area. Therefore, we can reduce the time a robot takes to pick and move parts based on the status recognized. Since the robot picks and moves parts simultaneously with manual assembling task, the time of a robot task is only the time required for a robot to place and position a part in the assembly area. In the simulation, the average time for a robot to place a part was five seconds. For “screw tightening,” the robot tightens the screw after the human worker inserts the screw. In the previous simulation, the human worker inserted all required screws (four screws to assemble the bottom clamp plate). The robot began screw-tightening after the human worker had inserted all screws. However, we have enabled robots to tighten the first screw after status recognition once the human’s hand leaves the first position. Based on the previous study (refer to Table 7), the human worker required twenty seconds to pick and insert four screws ($t_i$), while the robot required sixteen seconds to tighten the screws ($t_j$). This means that the time required for both actions ($t_i + t_j$) was thirty-six seconds. With the status recognition of inserting screws performed by humans, we designed the robot to tighten the first screw after the human worker had inserted the second screw for safety reasons. In other words, the robot starts tightening screws ten seconds earlier. Therefore, the execution time for both actions was reduced from thirty-six seconds to twenty-six seconds.

| Sub-Task by Human | $t_i$ (s) | Subsequent Sub-Task by Robot | $t_j$ (s) | $t_i + t_j$ (Before) | $t_i + t_j$ (After) |
|-------------------|----------|-------------------------------|----------|----------------------|---------------------|
| ST #7: Insert 4 screws | 20       | ST #8: Tighten screws         | 16       | 36                   | 26                  |
| ST #17: Insert guide pin with force | 40       | ST #18: Lift and place side plate | 18       | 58                   | 53                  |
| ST #20: Insert core with force | 30       | ST #21: Pick and locate pins | 10       | 40                   | 35                  |
| ST #24: Insert 4 screws | 20       | ST #25: Tighten screws         | 16       | 36                   | 26                  |
| ST #32: Insert 4 screws | 20       | ST #33: Tighten screws         | 16       | 36                   | 26                  |
| ST #34: Position plate | 16       | ST #35: Lift plate            | 15       | 31                   | 26                  |

Table 7. List of early execution tasks by robot and execution time reduction.

There were four “screw tightening” tasks performed by the robot after the human worker had inserted screws, and two “pick and place” tasks performed by the robot(s) after the human worker had assembled the part (see Table 7). The assembly time without early execution was 638 s (10 min 38 s). With early execution, we eliminated 50 s of the robot’s idle time. Hence, we achieved a 7.84% assembly time reduction based on the simulation. Since we calculated the result based on the simulation time, we are in the process of establishing an HRC mold assembly testbed to evaluate the practical applications of this study in the future.

This study shows that task and status recognition during mold assembly operation is achievable using transfer learning on a pre-trained YOLOv5 model, even with small image datasets. The criteria to measure the sustainability introduced in Sustainable Artificial Intelligence (AI) include the use of reusable data and the training of the algorithm [47].
The rule of thumb is 1000 images per class for developing object recognition using deep learning. Furthermore, training a YOLO model from scratch may require up to several days. However, this study reduced the number of images and the training time by using pre-trained YOLOv5 models. We spent one and half hours training the YOLOv5s without freezing model, and up to seven and half hours training the YOLOv5x without freezing model. With the developed recognition model, the robot involved in HRC mold assembly can identify the status of the current manual task in order to execute the subsequent task earlier without a signal from the human worker. Besides this, the robot can avoid collision with a human by detecting human hands in the assembly area. In the future, we will implement this recognition model in two robot HRC mold assembly testbeds to improve the human workload and efficiency of the mold assembly operation. Therefore, this study supports sustainability in terms of human safety, working conditions, and reductions in assembly time.

5. Conclusions

This study presents the development of task and status recognition for an HRC mold assembly operation. The proposed recognition model consists of task recognition stages utilizing part and tool detection and status recognition, which identify the status of a task based on the human action. Before developing the recognition model, we decomposed the assembly operation into tasks, sub-tasks, and action. The sub-tasks contain information on parts and tools used. Then, we decomposed the sub-tasks into a series of actions that defined the status of the task. Therefore, we collected images of parts and tools, and defined actions to train the recognition model. We used a pre-trained YOLOv5 model to develop the model due to the limited dataset available. We selected pre-trained YOLOv5l without freezing layers to implement the task and status recognition because it showed the best performance based on accuracy and inference time among all YOLOv5 models. Besides this, smaller YOLOv5 models cannot detect all the statuses and parts because of the uneven number of images in each class. We re-trained the YOLOv5m model with only images of action classes and with a 5-fold cross-validation method. Then, we combined the weights during the inference to investigate the detection ability. We found that the 5-fold cross-validation method improved the average mAP score and detection ability, but the inference time increased 2.4 times.

We are currently pursuing physical experiments using an HRC assembly cell testbed to further evaluate and verify the real-time practical implementations of this study. In this study, we focused on recognizing the manual task. However, it is necessary to recognize robot tasks so as to enable progress estimation and communication between humans and robots. Furthermore, we will expand the developed task recognition model to estimate the progress of the recognized task based on object tracking, and to estimate the completion time of the recognized task.

Author Contributions: Conceptualization, Y.Y.L.; methodology, Y.Y.L.; software, Y.Y.L.; validation, Y.Y.L. and K.R.; formal analysis, Y.Y.L.; investigation, Y.Y.L. and K.R.; resources, Y.Y.L.; data curation, Y.Y.L.; writing—original draft preparation, Y.Y.L.; writing—review and editing, K.R.; visualization, Y.Y.L.; supervision, K.R.; project administration, K.R.; funding acquisition, K.R. All authors have read and agreed to the published version of the manuscript.

Funding: Not applicable.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data available on request due to restrictions. The data presented in this study are available on request from the corresponding author. The data are not publicly available because the data are also parts of the authors’ ongoing research.
Acknowledgments: This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) (No. 2021R1A2C2009984).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Barosz, P.; Golda, G.; Kampa, A. Efficiency analysis of manufacturing line with industrial robots and human operators. Appl. Sci. 2020, 10, 2862. [CrossRef]

2. Khalid, A.; Kirisci, P.; Ghraiiri, Z.; Thoben, K.D.; Pannek, J. A methodology to develop collaborative robotic cyber physical systems for production environments. Logist. Res. 2016, 9, 1–15. [CrossRef]

3. Demir, K.A.; Döven, G.; Sezen, B. Industry 5.0 and human-robot co-working. Procedia Comput. Sci. 2019, 158, 688–695. [CrossRef]

4. Maddikunta, P.K.R.; Pham, Q.V.; Prabadevi, B.; Deepa, N.; Dev, K.; Gadekallu, T.R.; Ruby, R.; Liyanage, M. Industry 5.0: A survey on enabling technologies and potential applications. J. Ind. Inf. Integr. 2021, 100257. [CrossRef]

5. Krüger, J.; Lien, T.K.; Verl, A. Cooperation of human and machines in assembly lines. CIRP Ann. 2009, 58, 628–646. [CrossRef]

6. Cherubini, A.; Passama, R.; Crosnier, A.; Lasnier, A.; Fraisse, P. Collaborative manufacturing with physical human-robot interaction. Robot Comput. Integr. Manuf. 2016, 40, 1–3. [CrossRef]

7. Makris, S.; Karagiannis, P.; Koukas, S.; Matthaiakis, A.S. Augmented reality system for operator support in human-robot collaborative assembly. CIRP Ann. 2016, 65, 61–64. [CrossRef]

8. Müller, R.; Vette, M.; Mailahn, O. Process-oriented task assignment for assembly processes with human-robot interaction. Procedia CIRP 2016, 44, 210–215. [CrossRef]

9. Ranz, F.; Komenda, T.; Reisinger, G.; Hold, P.; Hummel, V.; Sihn, W. A morphology of human robot collaboration systems for industrial assembly. Procedia CIRP 2018, 72, 99–104. [CrossRef]

10. Kasalino, A.; Civitini, F.; Zanchetti, A.M.; Piroddi, L.; Rocco, P. Human-robot collaborative assembly: A use-case application. IFAC-PapersOnLine 2018, 51, 194–199. [CrossRef]

11. Liau, Y.Y.; Ryu, K. Task Allocation in human-robot collaboration (HRC) Based on task characteristics and agent capability for mold assembly. Procedia Manuf. 2020, 51, 179–186. [CrossRef]

12. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common objects in context. In Proceedings of the 13th European Conference on Computer Vision (ECCV 2014), Zurich, Switzerland, 6–12 September 2014; Springer: Cham, Switzerland, 2014.

13. Russakovskiy, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karparyth, A.; Khosla, A.; Bernstein, M.; et al. Imagenet large scale visual recognition challenge. Int. J. Comput. Vis. 2015, 115, 211–252. [CrossRef]

14. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. Adv. Neural Inf. Process. Syst. 2012, 25, 1097–1105. [CrossRef]

15. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.

16. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. In International Conference on Learning Representation (ICLR), San Diego, CA, USA, 7–9 May 2015.

17. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2015), Boston, MA, USA, 7–12 June 2015; pp. 1–9.

18. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR 2014), Columbus, GA, USA, 23–28 June 2014; pp. 580–587.

19. Girshick, R. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.

20. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. Adv. Neural Inf. Process. Syst. 2015, 28, 91–99. [CrossRef] [PubMed]

21. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.

22. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.

23. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017), Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.

24. Redmon, J.; Farhadi, A. YOLOv3: An incremental improvement. arXiv 2018, arXiv:1804.02767.

25. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object detection. arXiv 2020, arXiv:2004.10934.

26. Uzunovic, T.; Golubovic, E.; Tucakovic, Z.; Acikmese, Y.; Sabanovic, A. Task-based control and human activity recognition for human-robot collaboration. In Proceedings of the 44th Annual Conference of the IEEE Industrial Electronics Society (IECON 2018), Washington, DC, USA, 21–23 October 2018; pp. 5100–5115.

27. Chen, C.; Wang, T.; Li, D.; Hong, J. Repetitive assembly action recognition based on object detection and pose estimation. J. Manuf. Syst. 2020, 55, 325–333. [CrossRef]
28. Wang, P.; Liu, H.; Wang, L.; Gao, R.X. Deep learning-based human motion recognition for predictive context-aware human-robot collaboration. *CIRP Ann.* 2018, *67*, 17–20. [CrossRef]

29. Wen, X.; Chen, H.; Hong, Q. Human assembly task recognition in human-robot collaboration based on 3D CNN. In Proceedings of the 9th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER 2019), Suzhou, China, 29 July–2 August 2019; pp. 1230–1234.

30. Chen, C.; Zhang, C.; Wang, T.; Li, D.; Guo, Y.; Zhao, Z.; Hong, J. Monitoring of assembly process using deep learning technology. *Sensors* 2020, *20*, 4208. [CrossRef]

31. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Li, F. Imagenet: A large-scale hierarchical image database. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR 2009), Miami, FL, USA, 20–25 June 2009; pp. 248–255.

32. Niu, S.; Liu, Y.; Wang, J.; Song, H. A decade survey of transfer learning (2010–2020). *IEEE Trans. on Artif. Intell.* 2020, *1*, 151–166. [CrossRef]

33. Yang, Q.; Zhang, Y.; Dai, W.; Pan, S.J. *Transfer Learning*; Cambridge University Press: Cornwall, UK, 2020; pp. 14, 221–222.

34. Vasilev, I. *Advanced Deep Learning with Python*; Packt Publishing Ltd.: Birmingham, UK, 2019; pp. 90–91.

35. Židek, K.; Hosovsky, A.; Pite’, J.; Bednár, S. Recognition of assembly parts by convolutional neural networks. In *Advances in Manufacturing Engineering and Materials, Proceedings of the International Conference on Manufacturing Engineering and Materials (ICMEM 2018), Nový Smokovec, Slovakia, 18–22 June 2018*; Springer: Cham, Switzerland, 2018; pp. 281–289.

36. Liu, H.; Wang, L. Collision-free human-robot collaboration based on context awareness. *Robot Comput. Integr. Manuf.* 2021, *67*, 101997. [CrossRef]

37. Tao, W.; Al-Amin, M.; Chen, H.; Leu, M.C.; Yin, Z.; Qin, R. Real-time assembly operation recognition with fog computing and transfer learning for human-centered intelligent manufacturing. *Procedia Manuf.* 2020, *48*, 926–931. [CrossRef]

38. Jiao, L.; Zhang, F.; Liu, F.; Yang, S.; Li, L.; Feng, Z.; Qu, R. A survey of deep learning-based object detection. *IEEE Access* 2019, *7*, 128837–128868. [CrossRef]

39. GitHub. Yolov5. Available online: https://github.com/ultralytics/yolov5 (accessed on 30 June 2021).

40. Yao, J.; Qi, J.; Zhang, J.; Shao, H.; Yang, J.; Li, X. A Real-time detection algorithm for kiwifruit defects based on YOLOv5. *Electronics* 2021, *10*, 1711. [CrossRef]

41. Zhou, F.; Zhao, H.; Nie, Z. Safety helmet detection based on YOLOv5. In Proceedings of the International Conference on Power Electronics, Computer Applications (ICPECA 2021), Shenyang, China, 22–24 January 2021.

42. Kim, J.A.; Sung, J.Y.; Park, S.H. Comparison of Faster-RCNN, YOLO, and SSD for real-time vehicle type recognition. In Proceedings of the International Conference on Consumer Electronics-Asia (ICCE 2020–Asia), Busan, Korea, 1–3 November 2020.

43. Yang, G.; Feng, W.; Jin, J.; Lei, Q.; Li, X.; Gui, G.; Wang, W. Face mask recognition system with YOLOV5 based on image recognition. In Proceedings of the 6th International Conference on Computer and Communications (ICCC 2020), Chengdu, China, 11–14 December 2020; pp. 1398–1404.

44. Cheng, S. Plastic mold assembly. Available online: https://www.youtube.com/watch?v=laEWSU4oulw (accessed on 31 January 2021).

45. GitHub. LabelImg. Available online: https://github.com/tzutalin/labelImg.git (accessed on 28 February 2021).

46. Liu, Y.Y.; Ryu, K. Genetic algorithm-based task allocation in multiple modes of human-robot collaboration systems with two cobots. *Int. J. Adv. Manuf. Technol.* under review.

47. Van Wynsbergh, A. Sustainable AI: AI for sustainability and the sustainability of AI. *A.I. Ethics* 2021, *1*, 213–218. [CrossRef]