A Collaborative Visual Sensing System for Precise Quality Inspection at Manufacturing Lines

JIALE CHEN and DUC VAN LE, HP-NTU Digital Manufacturing Corporate Lab, Nanyang Technological University, Singapore
RUI TAN, School of Computer Science and Engineering, Nanyang Technological University, Singapore
DAREN HO, HP Inc., Singapore

Visual sensing has been widely adopted for quality inspection in production processes. This article presents the design and implementation of a smart collaborative camera system, called *BubCam*, for automated quality inspection of manufactured ink bags in Hewlett-Packard (HP) Inc.'s factories. Specifically, BubCam estimates the volume of air bubbles in an ink bag, which may affect the printing quality. The design of BubCam faces challenges due to the dynamic ambient light reflection, motion blur effect, and data labeling difficulty. As a starting point, we design a single-camera system that leverages various deep learning (DL)-based image segmentation and depth fusion techniques. New data labeling and training approaches are proposed to utilize prior knowledge of the production system for training the segmentation model with a small dataset. Then, we design a multi-camera system that additionally deploys multiple wireless cameras to achieve better accuracy due to multi-view sensing. To save power of the wireless cameras, we formulate a configuration adaptation problem and develop the single-agent and multi-agent deep reinforcement learning (DRL)-based solutions to adjust each wireless camera's operation mode and frame rate in response to the changes of presence of air bubbles and light reflection. The multi-agent DRL approach aims to reduce the retraining costs during the production line reconfiguration process by only retraining the DRL agents for the newly added cameras and the existing cameras with changed positions. Extensive evaluation on a lab testbed and real factory trial shows that BubCam outperforms six baseline solutions including the current manual inspection and existing bubble detection and camera configuration adaptation approaches. In particular, BubCam achieves 1.3x accuracy improvement and 300x latency reduction compared with the manual inspection approach.

CCS Concepts: • **Computer systems organization** → **Sensor networks**; • **Computing methodologies** → **Image segmentation;**

Additional Key Words and Phrases: Product quality inspection, visual sensing, deep learning, reinforcement learning

A preliminary version [5] of this work appeared in the ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS'23) held in San Antonio, Texas, USA, May 2023.
This study is supported under the RIE2020 Industry Alignment Fund–Industry Collaboration Projects (IAF-ICP) Funding Initiative, as well as cash and in-kind contribution from the industry partner, HP Inc., through the HP-NTU Digital Manufacturing Corporate Lab.
Authors' addresses: J. Chen and D. Van Le, HP-NTU Digital Manufacturing Corporate Lab, Nanyang Technological University, 65 Nanyang Drive, Singapore 637460; e-mails: {jiale.chen, vdle}@ntu.edu.sg; R. Tan, School of Computer Science and Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798; e-mail: tanrui@ntu.edu.sg; D. Ho, HP Inc., 1A Depot Cl, Singapore 109842; e-mail: kok-loon.ho@hp.com.

ACM 2378-962X/2024/10-ART45
https://doi.org/10.1145/3643136

ACM Reference Format:
Jiale Chen, Duc Van Le, Rui Tan, and Daren Ho. 2024. A Collaborative Visual Sensing System for Precise Quality Inspection at Manufacturing Lines. *ACM Trans. Cyber-Phys. Syst.* 8, 4, Article 45 (October 2024), 27 pages. https://doi.org/10.1145/3643136

1 INTRODUCTION

With the recent advancements of the **Internet of Things (IoT)** infrastructure and **deep learning (DL)** techniques in dealing with complex data patterns, visual inspection has been increasingly employed to enable automated quality control at manufacturing lines [3, 13, 17]. The purpose is to automatically detect any defects in the manufactured products with low latency before the final distribution to the customers [25]. The designs and implementations of such automated inspection systems often require a lot of effort in dealing with the strict requirements, practical constraints, and complex environmental conditions of the industrial processes. In this article, the target application is the quality inspection of the ink bag products manufactured in **Hewlett-Packard (HP)** Inc.'s ink factories for the large industry printers. It is the final quality control procedure that targets estimating the volume of air bubbles inside the ink bags. During the production process, the ink-filling machines may inject air bubbles into the ink bags. The ink bags with a high volume of air bubbles can significantly reduce the printing quality of the printers. Specifically, during the printing process, the ink may absorb the bubbles, which leads to unstable velocity of the ink droplets [8]. Thus, the quality inspection is needed to ensure that the ink bags with a large volume of air bubbles are not distributed to the customers.

The current protocol of HP's factories adopts a manual quality inspection procedure that begins with removing the outer plastic layer of the inspected bag and then squeezing the air bubbles to form an air cone in a corner of the bag. Finally, the size of the air cone is manually measured to determine the volume of the air bubbles. This manual measurement suffers from low accuracy and high latency. From the historical records, it can take up to 10 minutes to inspect an ink bag. Moreover, it is a costly and destructive test due to the removal of the outer layer of the ink bag.

To increase the inspection accuracy and throughput and reduce costs, in this article, we design and implement a smart camera system, called *BubCam*, which leverages a **fog computing (FC)**-assisted wall-powered depth camera and wireless cameras to enable the automated quality inspection of the ink bags. The primary goal of BubCam is to accurately estimate the volume of the air bubbles inside the manufactured ink bags without manual intervention. However, in our application, the size of the air bubbles is small, at the millimeter level. Our experiments in Section 8 show that the existing bubble detection approaches using the conventional **computer vision (CV)**-based [26] and the recent **convolutional neural network (CNN)**-based [11] object detection algorithms have inferior performance in estimating the volume of these small bubbles. To achieve high accuracy, BubCam employs a DL-based image segmentation model that extracts multiple regions of bubbles in the images for the bubble volume estimation.

Due to the dynamic environment condition and complex settings of the production lines, the design of BubCam also faces the following three additional challenges. First, in the production lines, the ambient light often reflects on the surface of the ink bags. As a result, the air bubbles may be blocked and invisible in the captured RGB images. Moreover, the location of the reflected areas may change over time due to the dynamic lighting condition in the factories. Second, the quality of the images can suffer from the motion blur effect caused by the movement of the ink bag on the conveyor belt. Third, labeling images of the manufactured ink bags is labor intensive and time-consuming. Specifically, determining the ground-truth air bubble areas in the captured images cannot be performed by normal persons and requires a collaboration with experienced technicians.

To address the first two challenges, we develop a late fusion scheme that combines the segmentation results of multiple consecutive RGB images to generate the final result of the bubble volume estimation. Fusion can help improve the estimation accuracy because different frames may have different quality and reflected areas under variations of the light reflection and motion blur conditions. To further obtain high accuracy under the dynamic lighting conditions, the depth sensing information is also fused with the RGB images when the reflection areas are detected in the RGB images. To solve the problem of labeling difficulty, we propose a knowledge-based labeling approach that utilizes the prior knowledge of the production to facilitate the labeling process.

Multiple views of a scene are generally robust against the light reflection variation problem. Thus, we further develop a multi-camera BubCam system that uses multiple wireless cameras to assist the main wall-powered camera. The wireless cameras are deployed to capture the inspected ink bag from different view angles. These images are then fused with the images captured by the main camera. Specifically, due to the low cost and easy installation of the wireless cameras, the multi-camera BubCam can be deployed in an *ad hoc* fashion to achieve better accuracy, especially under the dynamic light reflection condition.

However, the wireless cameras are often powered by batteries with finite capacities. Thus, it is desirable to adapt their configuration to minimize the camera's energy consumption while maintaining the performance requirement in response to the variations of the presence of air bubbles and the lighting condition. For instance, the wireless cameras should only be activated when the bubble areas are blocked by light reflection in the images of the main camera. Furthermore, the capturing frame rate of the wireless cameras should increase when the air bubbles appear in their field of view. Otherwise, the frame rate should be kept to a minimum to save power.

To achieve the goal, we propose a novel configuration adaptation approach to obtain desired accuracy with minimum energy consumption. Specifically, we formulate an adaptation problem as a **Markov decision process (MDP)** that aims to configure the activation mode and image frame rate of the wireless cameras. The objective of the formulated MDP is to satisfy the bubble volume estimation accuracy while maximizing the system's lifetime in response to variations of the presence of the air bubbles and light reflection. The preliminary version of this work [5] presented the design of a **single-agent deep reinforcement learning (SADRL)** approach to learn the optimal configuration policy for the wireless cameras. In the factory, the manufacturing line is often reconfigured four or five times per month to produce ink bags with different sizes and colors. The existing wireless cameras may need to be repositioned and/or additional cameras can be deployed to maintain sufficient coverage and inspection accuracy. The **deep reinforcement learning (DRL)** agent may need to be retrained to learn a new configuration policy for the repositioned and new wireless cameras. This article also presents the extended multi-camera BubCam design that employs a **multi-agent DRL (MADRL)** approach to reduce the retraining costs such as the required number of the training data samples and the retraining latency.

We perform extensive evaluation via testbed experiments in both the controlled lab and factory deployments. Specifically, we collect an image dataset to drive the design of the proposed DL-based image processing pipeline and DRL-based configuration adaptation approach. We compare BubCam with six baseline approaches, which include the factories' manual inspection, three bubble detection, and two configuration adaptation baseline approaches. The evaluation results show that BubCam achieves accuracy improvement of about 34% and latency reduction of up to 300x compared with the factories' manual inspection.

The contributions of this work can be summarized as follows:

— We design and implement BubCam, which leverages the DL-based image segmentation and fusion techniques to accurately estimate the volume of the air bubbles of the manufactured ink bags in HP's production lines. BubCam aims to address challenges caused by combined

impacts of the motion blur and strong reflection conditions in the industrial environment. Our design can be useful to the developments of other vision systems for the automated quality inspection of the relevant products in the industrial processes.
— We formulate the camera configuration adaptation problem and propose the SADRL- and MADRL-based approaches to learn the efficient configuration policy in the industrial settings.
— We implement and deploy real-world testbeds in both a controlled system and the factories' manufacturing lines to evaluate BubCam. The effectiveness of BubCam is compared with six baseline approaches.

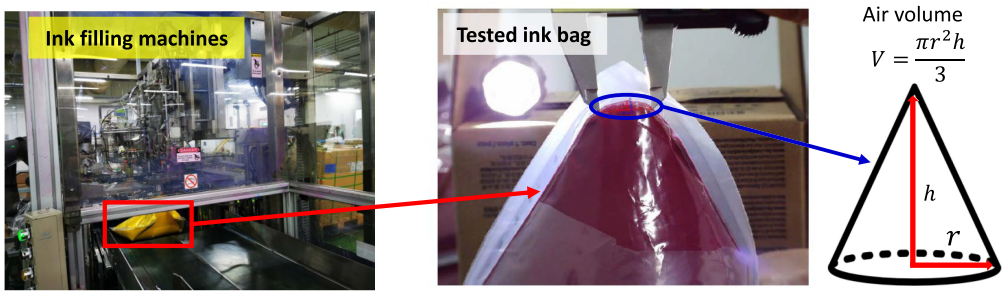
Article organization: Section 2 reviews related work. Section 3 describes the background and approach overview. Section 4 overviews the design of BubCam. Section 5 presents the detailed design of the single-camera BubCam system. Section 6 and Section 7 present the multi-camera BubCam system with the SADRL-based and MADRL-based configuration adaptation approaches, respectively. Section 8 presents the deployment and evaluation results. Section 9 discusses the limitations and future work. Section 10 concludes the article.

2 RELATED WORK

In this section, we review the related works on industrial visual inspection, bubble detection, and multi-camera system.

■ **Industrial visual inspection:** Visual inspection is widely adopted to inspect the quality of manufactured products in factories. For instance, the studies in [7] developed a visual sensing system that aims to rectify millimeter-level edge deviation during production at the factory. The developed system can help increase the material utilization rate and save manpower. In [17], the authors designed the FC-assisted camera systems to detect defects of the tile products in real time. The study in [20] designed a vision inspection system to replace manual inspection for detecting the liquid level and segmenting the bubble area in the liquor distillation. Moreover, the study in [27] developed an automated visual inspection system that employs the CNN models (i.e., YOLOV5 and DeepLabV3Plus) to detect the surface defects of the molded pulp products. Similarly, BubCam is a visual sensing system that aims to inspect the quality of the air bubbles of the ink bags manufactured in HP's ink factories.

■ **Bubble detection:** Bubble detection is important in many industrial applications. The existing studies [9, 11, 14, 26] mainly applied the object detection algorithms for the bubble detection, which can be divided into the following two main categories. The first category consists of the conventional CV-based object detection approaches [14, 26] that generally use the edge detection algorithms to extract the air bubbles from the images. Then, the extracted areas are fit to the geometry shapes (e.g., circles or ellipse) whose size is considered as the bubble volume. For instance, the study in [26] adopted a hough circle detection algorithm to extract the bubble areas from the image. Then, a concentric circular arrangement algorithm was used to determine the circles that best fit to the extracted bubble areas. The second category consists of the DL-based object detection approaches [9, 11]. For instance, the study in [11] developed a **region-based CNN (RCNN)** model to detect the bounding boxes of the bubbles from the image. Then, a shape regression CNN was adopted to transform the extracted boxes to circles or ellipses whose total number of pixels is considered as the bubble volume. In [9], the authors proposed a **generative adversarial network (GAN)** model to generate the augmented images for efficiently training the RCNN model with a limited labeled images dataset. However, as shown by our experiments in Section 8, these existing CV-based and DL-based object detection approaches have inferior performance in estimating the volume of the small bubbles in the inspected ink bags in our application. Thus, we employ a DL-based segmentation model [29] for the bubble volume estimation in BubCam.



a) An ink bag manufacturing line. b) Manual quality inspection.

Fig. 1. An illustration of factories' ink bag manufacturing and manual inspection.

■ **Multi-camera system:** Existing studies [4, 18, 21, 30] proposed the use of multiple cameras for product quality inspection in the production lines. For instance, the study in [4] developed a multi-camera system to inspect defects on the stereo skeleton of the car. Specifically, multiple cameras are deployed to cover different parts of the car for detecting the tiny defects in real time. In [21], the authors designed a multi-camera system to achieve a highly accurate 3D profile measurement. Moreover, the study in [30] proposed a multi-camera system to inspect the quality of the product parts moving on the conveyor belt in the factories. The proposed multi-camera BubCam uses the multiple wireless cameras to improve the bubble volume estimation accuracy under the dynamic ambient light reflection conditions. Similar to [22, 30], we adapt the configuration for the camera's parameters to achieve the desired performance with minimum energy consumption under the time-varying environment conditions. Furthermore, in [22], the author adapts the frame rate of wireless cameras by measuring the similarity of local frames and frames from other wireless cameras. Differently, in BubCam, we control the activation and adapt the frame rate of the wireless camera base on the position of bubbles and reflection in the main camera. Furthermore, we formulate the adaptation problem as an MDP problem and train a DRL model to control the wireless cameras.

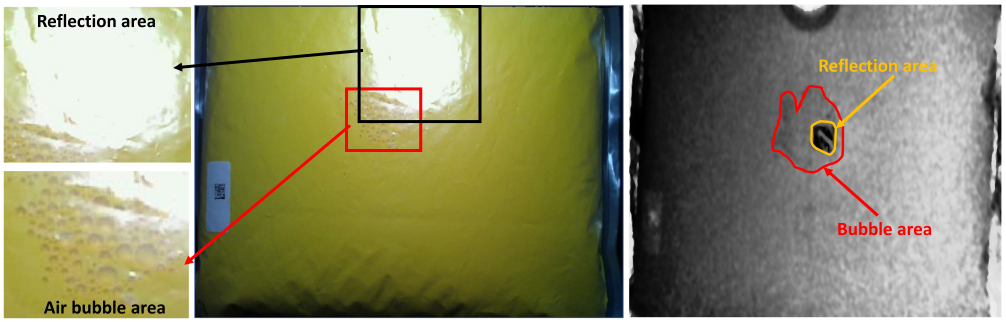
The preliminary version of this work [5] presented the design of a SADRL approach to learn the optimal camera configuration policy. It does not consider the retraining costs of the DRL agent during the production line conversion process. In this article, we make the following new contributions. First, we formulate a Markov game problem and present the design of a MADRL approach that aims to reduce the training costs to learn the optimal configuration policy for the repositioned and new wireless cameras during the conversion process. Second, we conduct new experiments to compare the prior approaches including the SADRL approach [5] and the MADRL approach in this article. The evaluation results show that the MADRL approach can achieve the similar camera's energy usage and inspection accuracy with reduced retraining costs.

3 BACKGROUND, MOTIVATION, & APPROACH

In this section, we present the background of the ink bag quality inspection at HP's factories. Then, we describe the design approach and challenges of BubCam.

3.1 Background and Motivation

As discussed in Section 1, the current protocol of HP's factories adopts a manual quality inspection approach to estimate the volume of the air bubbles inside the manufactured ink bags. Figure 1 illustrates how the ink bags are inspected in the HP factories. Specifically, the empty ink bags with two plastic layers are filled by the ink-filling machines. Then, the filled ink bags



(a) RGB image. (b) Depth image.

Fig. 2. Samples of captured RGB and deep images.

are moved through the conveyor belt to the packaging process. However, the filling machines may inject the air bubbles into the bags. The air bubbles with a sufficiently large size can reduce the printing quality of the industrial printers that use the manufactured ink bags [8]. Thus, the quality inspection is performed to ensure that the ink bags with a large bubble volume are not distributed to customers. The inspection results also provide information to determine the need for corrective actions in the ink-filling process. To achieve this goal, during the production process, the technicians continuously monitor and perform human eye tracking to examine the ink bags moving on the conveyor belt. The ink bags detected with the air bubbles are taken for a manual measurement procedure, which begins with removing the outer layer of the inspected ink bag for better visibility. Then, the air bubbles are manually squeezed to a corner of the bag, which approximately forms an air cone, as illustrated in Figure 1(b). A backlight is placed behind the ink bag to better visualize the boundary of the formed air cone. Finally, a vernier caliper is used to measure the radius and height of the cone for determining the volume of the air bubbles.

The above manual quality inspection approach has two main drawbacks as follows. First, it is a destructive test because the outer layer of the inspected ink bags is removed for better visibility of the air bubbles. As such, all inspected ink bags are destroyed even they do not have large air bubbles. Second, it has high latency. From historical records, the manual inspection procedure including the outer layer removal, bubble squeezing, and air cone size measurement can take about 5 minutes to 10 minutes per ink bag. Thus, to avoid the destructive tests and increase the inspection throughput, it is desirable to develop an automated inspection system that can reliably estimate the air bubble volume of the ink bags with low latencies.

3.2 Design Approach and Challenges

In this work, we design and implement an automated visual quality inspection system, called BubCam, to replace the factories' manual inspection approach. The key design approach of BubCam is to deploy a single wall-powered camera to continuously capture images of the manufactured ink bags moving on the conveyor belt. The camera is deployed at a relatively far distance from the conveyor belt such that a captured image can contain the entire ink bag. Meanwhile, the actual air bubbles often have a size of millimeter level. As a result, the air bubbles appear with an extremely small size in the captured images. For instance, as shown in a sample in Figure 2(a), an air bubble has a size of around 1,200 pixels, which only accounts for 1/768 of the entire 1-megapixel image.

It is nontrivial to accurately estimate the volume of such small bubbles in the captured images. A possible approach to address this problem is to place the camera closer to the ink bags. However, in the production lines, the manufactured ink bags can have three different sizes of 36.5cm × 31cm,

41.5cm × 37cm, or 42cm × 49cm. Multiple cameras are needed to capture the entire ink bags with these large sizes at a short distance. Deploying multiple wall-powered cameras is undesirable in industrial settings, especially in the scheme of a *reconfigurable manufacturing system* [16] that can adjust the layout and configuration of the production lines in response to variations of the regulatory requirements or market demands. Moreover, the use of multiple high-quality cameras also increases the system cost. Thus, in BubCam, we use a single wall-powered camera.

Furthermore, due to the buoyancy of the ink and the curved bag surface, the air bubbles often cluster into multiple groups on the bag's internal surface. This observation suggests that the air bubbles in the same group should be detected together. Thus, we develop a DL-based image segmentation approach that extracts the air bubbles into multiple image regions for the bubble volume estimation. We also investigate the feasibility of the existing CV-based [26] and CNN-based [11] object detection approaches for our bubble estimation problem. However, from our experiments (cf. Section 8), these existing approaches have inferior performance compared with BubCam. The main reason is that these approaches aim to recognize the bubbles as individual objects. Thus, they fail to accurately detect the small bubbles that are squeezed into multiple groups in the ink bag. Moreover, the DL-based segmentation model generally requires less effort for image labeling than the DL-based object detection approaches. Specifically, determining the ground-truth label of the bubble regions is often easier than determining the ground-truth label of the individual bubbles.

Furthermore, due to the complex production settings and requirements, the design of BubCam also faces the following two additional challenges. First, the bubble volume estimation accuracy suffers from the ambient light reflection problem. Specifically, in the factories, the production systems are often set up in an open space or an enclosed space covered by the transparent frames. The ambient light can reflect on the surface of the ink bags on certain locations, which may change over time due to the motion of the ink bags on the conveyor belt and the surrounding industrial objects and humans. As a result, the air bubbles can be blocked by the light reflection on the captured RGB images, as illustrated in Figure 2(a). The blocked bubbles cannot be detected based on the optical visual sensing information only. Second, as the ink bags move on the conveyor belt during the inspection process, the quality of captured images can be low due to the motion blur effect. The low-quality images result in low accuracy in detecting small air bubbles.

To address the above challenges, BubCam fuses multiple consecutive RGB images that may have different quality and reflected areas under the dynamic light reflection and motion blur conditions. Furthermore, we prototype BubCam by a depth camera that can provide both the RGB and depth images. The hardware components of BubCam will be described in Section 4. Figures 2(a) and 2(b) show the RGB and depth images captured by the camera under the light reflection. As shown in Figure 2, the air bubble areas that are blocked in the RGB frame are still visible in the depth image. Thus, to achieve better accuracy, we fuse the RGB and depth images when a reflected area is detected on the RGB image. However, as shown in Figure 2(b), the depth image also has a reflection area, which is caused by the infrared light emitted from itself. Although the reflection area is small compared with the reflection area in RGB images, the bubbles blocked in both the RGB and depth images cannot be recognized.

We further develop a multi-camera BubCam system that aims to improve accuracy, especially under the dynamic lighting condition. The multi-camera BubCam additionally uses multiple wireless cameras to capture the inspected ink bag from different view angles. These images are fused with the images captured by the main camera to generate the final bubble volume estimation result. Specifically, due to the low cost and easy installation of the wireless cameras, the multi-camera BubCam can be deployed in an *ad hoc* fashion for accuracy improvement, especially when the strong light reflection is observed. Moreover, we formulate a camera configuration adaptation problem and propose a SADRL-based learning approach to save the camera's battery power.

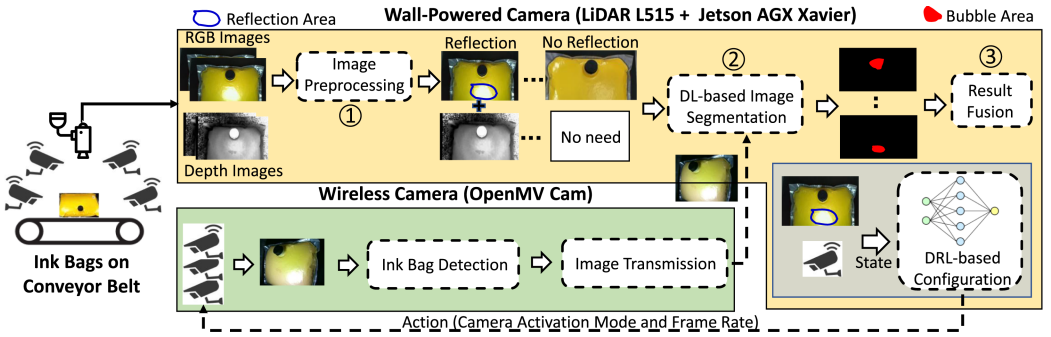


Fig. 3. BubCam system overview. A main wall-powered camera is supported by multiple wireless cameras.

In addition, a production line is often reconfigured four or five times per month to manufacture ink bags in different sizes and filled with inks in different colors. Specifically, the injection speed of the ink-filling machine and the movement speed of the conveyor belt are adjusted to accommodate the changes in ink bag size and ink color. The positions of the existing wireless cameras may need to be adjusted accordingly. The learned configuration policy of the DRL agent that is trained with the wireless cameras located at the previous positions might not maintain high performance for the configuration of the cameras at the new positions. Moreover, additional cameras may be deployed to achieve sufficient coverage and accuracy for inspecting the ink bags with larger sizes. Therefore, the DRL agent needs to be retrained to learn the new configuration policy for the repositioned and new cameras. In general, the DRL agent retraining process incurs high training costs in terms of the number of required data samples and training latency. To address this issue, we further develop a MADRL approach to train multiple DRL agents, each of which is used for the configuration adaptation of a separate wireless camera. During the production line reconfiguration process, we only retrain the agent for the camera with the position adjustment distance higher than a threshold. New DRL agents can also be trained for newly added cameras.

Lastly, data labeling is a challenging task for developing BubCam. Specifically, labeling the images of the manufactured ink bags cannot be performed by normal persons based on their instinct and/or basic knowledge. The labeling process is time-consuming and requires collaboration with experienced technicians. Thus, it takes a lot of effort to create a big labeled image dataset for training and testing the BubCam's DL-based image segmentation model. To address this issue, we design the knowledge-based labeling approach that utilizes the prior knowledge about the motion of the ink bags to facilitate the labeling process.

4 SYSTEM OVERVIEW AND HARDWARE

In this section, we overview the design and hardware of BubCam, as illustrated in Figure 3.

4.1 Overview of Single-camera System

For the wall-powered camera, we choose L515 [15], an off-the-shelf LiDAR camera that can provide both RGB and depth images with a resolution of 1920×1080 pixels and 1024×768, respectively, at a frame rate up to 30 **frames per second (FPS)**. Moreover, it has a sensing range of from 0.25 meters to 9 meters. The L515 is connected to a fog node prototyped by a Jetson AGX Xavier unit that is equipped with a 2.03 GHz CPU and an embedded 1.2 GHz GPU. The images are processed at the fog node for the bubble volume estimation. Specifically, the image processing pipeline of the single-camera BubCam consists of the following three main steps.

■ **Image preprocessing:** As illustrated in Figure 3, the main camera L515 captures the ink bags moving through its view on the conveyor belt. Per image capturing, the L515 provides both RGB and depth image frames. Upon receiving the captured consecutive RGB frames, the fog node calculates the pixel-wise absolute difference between them and then selects the key frames whose pixel difference is higher than a certain threshold. Moreover, a color filtering algorithm is implemented to detect the ink bag and light reflection areas in each key RGB frame. When any reflection area is detected, the corresponding depth frame is also selected. The selected RGB and depth images are fed to the segmentation module.

■ **DL-based image segmentation:** We adopt a deep CNN-based segmentation model, called **pyramid scene parsing network (PSPNet)** [29], to extract the bubble areas from the images. However, due to the nature of a deep CNN, training the PSPNet often requires a large labeled image dataset to achieve satisfactory inference accuracy. As discussed earlier, determining the ground truth of the bubble areas is nontrivial. Thus, we develop a knowledge-based image labeling approach that uses the prior knowledge about the motion speed of the ink bags to facilitate the labeling process. We also design a new loss function that utilizes the relationship between the bubble volume in the consecutive frames to improve the training accuracy.

■ **Result fusion:** The PSPNet takes an RGB/depth image as input to predict a pixel score map of the image. For the RGB key frame with reflected areas, its score is fused with the score of its corresponding depth frame to generate the final score map of the frame. Then, we consider the total number of pixels with a score greater than a certain threshold as the bubble volume of the frame. Finally, the bubble volumes of all frames of the same ink bag are aggregated to yield the bubble volume of the inspected ink bag.

The detailed design of the above three steps is described in Section 5.

4.2 Overview of Multi-camera System

For the wireless camera, we choose the lower-power OpenMV H7 Plus that includes an OV5640 module providing an image resolution up to 2592×1944 (5 megapixels). The OpenMV is equipped with an ARM Cortex M7 480 MHz processor, a 32 MBs SDRAM, and a 1 MB SRAM, which allows us to implement simple algorithms to preprocess the images locally. Moreover, it is capable of adjusting its frame rate from 0 to 30 FPS.

■ **Local preprocessing:** Due to the movement of the ink bags, the captured images may only contain the conveyor belt. Thus, to reduce the image processing and communication overheard, the wireless camera runs a color filtering algorithm to detect the presence of the ink bag on its images. Then, it only sends the images with the ink bag to the fog node.

■ **DRL-based configuration adaptation:** At the fog node, we implement a DRL-based controller that aims to adapt the configuration for the wireless cameras's parameters in response to variations of the lighting condition and bubble presence. Specifically, the DRL controller periodically observes a system state including the presence of bubble and light reflection in the image of the main camera and the residual battery energy levels of the wireless cameras. Then, it selects an action for configuring the activation mode and frame rate of the wireless cameras. The main objective is to maintain the desired accuracy of the bubble volume estimation while maximizing the system lifetime. In Section 6 and Section 7, we formally formulate the configuration adaptation problems and present the proposed SADRL- and MADRL-based solutions.

5 DESIGN OF SINGLE-CAMERA BUBCAM

In this section, we describe details of the image preprocessing, segmentation, and fusion modules of the single-camera BubCam.

5.1 Image Preprocessing

5.1.1 Key Frame Selection. The captured consecutive RGB image frames may have similar contents. To reduce the image processing overheard, BubCam selects the key frames only for the image segmentation and fusion. Specifically, among the captured frame, the first frame is always selected as a key frame. Then, the pixel-wise absolute difference between the first and second frames is calculated to generate a pixel difference map. The difference map is converted to a gray-scale image. Finally, a median blur algorithm is used to filter out the noises from the gray-scale image.

A pixel is considered as a changed pixel from the first frame to the second frame if the difference of its value in the gray-scale image is greater than a threshold. The value of this threshold is chosen empirically under which BubCam can achieve a good balance between the image processing overhead and the inspection accuracy. A higher threshold value reduces the processing overhead but increases the miss detection rate of the ink bags. Specifically, if an image with the conveyor belt only is selected as a key frame, the first image with the ink bag among incoming images should be selected as the next key frame. Thus, the threshold value is set to be 80, which is the lowest pixel difference value between the image with the conveyor belt and the image with the ink bag.

The ratio of the number of changed pixels to the total number of a frame's pixels is used to represent the difference degree between the contents of the two frames. Then, the second frame is selected as a key frame if its difference degree is higher than 20%. Given a new frame, the above processing pipeline is repeated to determine whether it should be selected as a key frame based on its difference degree compared with the latest selected key frame.

5.1.2 Ink Bag and Reflection Detection. BubCam aims to fuse the key frames of the same ink bag to generate the final volume estimation of the bag. Thus, the third preprocessing step is to determine groups of the key frames that contain the same ink bag. Specifically, in the production lines, the ink bags are moving on the conveyor belt one by one with a certain distance. As a result, the groups of the same bag images are interleaved with the images with the conveyor belt only. Thus, we use the presence of the conveyor belt images to determine the images of the same group.

As the color of the ink bags is different from that of the conveyor belt, we implement a color filtering algorithm to detect the presence of the ink bag in the images. Specifically, the algorithm converts the RGB image to the **hue, saturation, value (HSV)** color map. A pixel is considered belonging to the ink bag area if its H, S, V values are within specific ranges that are determined based on the ink bag color. For the yellow ink bag we use the H, S, V value ranges of $[20, 40]$, $[40, 255]$, and $[40, 255]$, respectively. The image is considered as an image with no ink bag if yellow pixel is detected.

We also use the above algorithm to detect the presence of reflection areas on the ink bag to determine the need of depth fusion. Specifically, the color of the reflection areas is white. Thus, we use the H, S, V value ranges of $[222, 256]$, $[0, 256]$, and $[0, 256]$, respectively, to detect pixels of the reflection areas.

5.2 Image Segmentation

5.2.1 Segmentation Model. BubCam adopts the PSPNet [29], a state-of-the-art deep segmentation model that uses a ResNet50 [12] as the backbone CNN model to extract the feature map of the input image. Then, the feature map is forwarded into a pyramid pooling module in which the features are fused to generate a feature representation. Finally, the representation is fed into a convolutional layer to yield a predicted score map.

5.2.2 Knowledge-based Labeling. To label the images for training the PSPNet, we developed a manual labeling approach that requires a collocation with technicians in the production lines. This

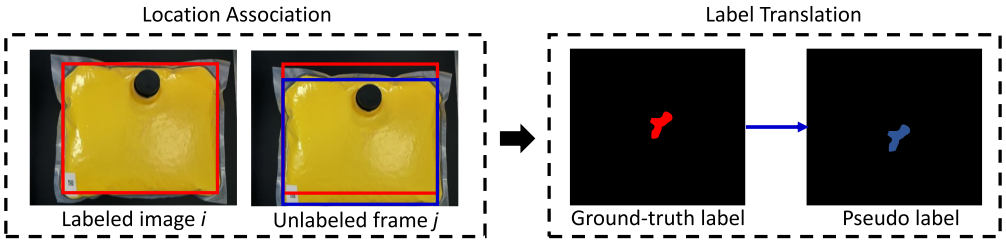


Fig. 4. Knowledge-based labeling approach of BubCam.

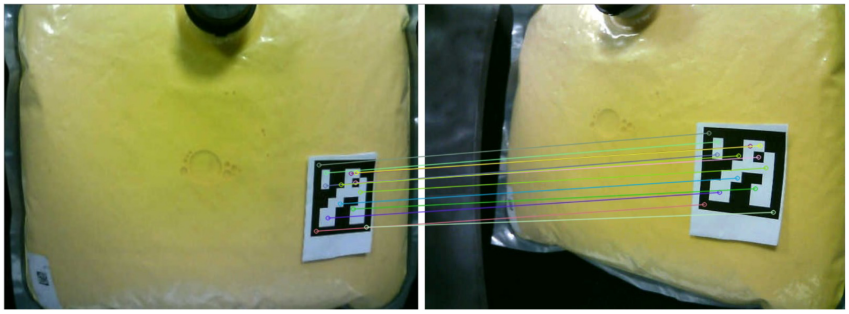
manual labeling approach is labor intensive and time-consuming. More details of this approach will be described in Section 8.1.2. Thus, we design a knowledge-based labeling approach that utilizes the prior knowledge about the motion of the ink bags to label the images based on the ground-truth labels obtained by the manual labeling. Figure 4 illustrates our knowledge-based labeling approach.

Specifically, in the production lines, the motion speed of the ink bags is known due to the constant rotation speed of the conveyor belt. We consider the consecutive images of the same ink bag that have the same ink bag area size but a different location. First, we adopt the manual labeling to generate the ground-truth label (i.e., the pixel score map) of a frame (e.g., frame i in Figure 4) among these frames. Our proposed labeling approach aims to generate a pseudo label of the unlabeled frame j based on the ground-label of frame i. First, we use the color filtering algorithm as described in Section 5.1.2 to detect bubble pixels in frames i and j. Then, a connected component analysis algorithm [1] is used to extract the ink bag area as a rectangle bounding box from the frames. Due to the constant motion speed of the ink bag, the relative distance between locations of two extracted boxes in frames i and j can be determined. The pseudo label of frame j is generated based on the calculated distance and the ground-truth label of frame i.

5.2.3 Model Training. We train the PSPNet with a loss function, denoted by L, which is defined as $L = L_{ce} + L_s$, where L_{ce} is the pixel-wise cross-entropy loss function of the PSPNet, and L_s is a new loss function that utilizes the prior system knowledge for improving the accuracy. Specifically, the consecutive images of the same ink bag should have a consistent volume of the air bubbles. Thus, the L_s is designed to minimize the difference between the predicted score maps of these frames. Let β denote the number of image training groups, each of which consists of the consecutive images of the same ink bag with the same bubble volume in the training dataset. If we define $\Omega_i = 1, \ldots, |\Omega_i|$ as the set of images in group i, then $L_s = \sum_{i=1}^{\beta} \sum_{j=1}^{|\Omega_i|} \sum_{k=1}^{|\Omega_i|} |s_j - s_k|$, where s_j and s_k are the predicted total score of images j and k in group i, respectively.

5.3 Result Fusion

The PSPNet predicts a pixel score map for each input key RGB/depth frame. First, the score map of the key RGB frame with the light reflection is fused with the score map of its corresponding depth frame. The L515 of BubCam provides the RGB and depth images with a resolution of 1920×1080 pixels and 1024×768 pixels, respectively. Fusing the entire low-quality depth frame with the high-quality RGB frame can reduce the image quality. Moreover, the purpose of the depth fusion is to achieve better accuracy in detecting the bubbles blocked by the reflection in the RGB image. Thus, we only fuse the score of RGB pixels in the reflected areas with those in the depth image. Specifically, the average fusion is adopted to generate the final score map of the reflected areas in the RGB image. Then, the score map of the fused frame is used to calculate the bubble volume as the total number of pixels with a score higher than a threshold value that is determined based on



(a) Image captured by main camera. (b) Image captured by wireless camera.

Fig. 5. Samples of images with a QR code.

the ink bag color. For instance, we use a threshold of 0.5 for the yellow ink bags. Finally, the bubble volume of an ink bag is obtained by averaging the volume estimation results of all its key frames.

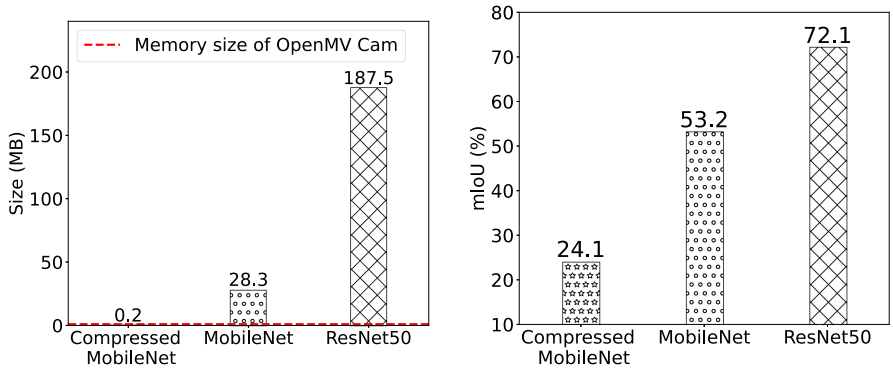
6 DESIGN OF MULTI-CAMERA BUBCAM WITH SADRL-BASED CONFIGURATION ADAPTATION

In this section, we first present the design of the multi-camera BubCam. Then, we formulate a configuration adaptation problem and propose an SADRL-based solution.

6.1 Image Preprocessing

Image processing pipeline: The images captured by the wireless cameras may contain the conveyor belt only. Thus, to save the camera's battery power, the wireless camera should only send the ink bag images to the fog node. To achieve this goal, we use the color filtering algorithm in Section 5.1.2 to detect the presence of the ink bag in the captured images. The main and wireless cameras have different view angles. Thus, we adopt a homography projection algorithm to associate the images of the wireless cameras with the images of the main camera. Let H_i denote the 3×3 homography matrix of the wireless camera i. The $P_i = [x_i, y_i, 1]$ denotes the coordinate vector of a pixel p_i in the image of the wireless camera i. Then, the pixel q_m with a coordinate vector, denoted by $Q_m = [x_m, y_m, 1]$, in the image of the main camera is associated with pixel p_i as follows: $Q_m = H_i \times P_i$. The matrix H_i for the wireless camera i is predetermined during the deployment phase. Specifically, we paste a QR code on a surface area of an ink bag. Figure 5 shows two samples of the ink bag images with the QR code captured by the main and wireless cameras i from different view angles. Then, the QR code areas in these two images are extracted and associated to determine the H_i.

In the above image processing pipeline, the wireless cameras send their captured images to the main camera, which employs the PSPNet-based framework to segment out the air bubble regions. The wireless camera's communication costs can be reduced if the PSPNet-based image segmentation framework can be implemented in the wireless camera. Thus, we investigate the performance of the PSPNet implementation in the 1 MB SRAM memory of the wireless camera (i.e., OpenMV). The wireless camera also has a memory SDRAM of 32 MBs, but this memory can be used for storing the captured images only. In the main camera, the PSPNet with a CNN backbone of ResNet50 requires a memory size of 187.5 MB. Thus, we use a lightweight CNN backbone, called MobileNet V2 [23], for the PSPNet in the wireless camera. However, the PSPNet with the original MobileNet still requires a memory size of 28.3 MB. Therefore, we adopt an existing CNN compression approach [6] to reduce the memory size of MobileNet. As a result, the memory size of the PSPNet with the compressed MobileNet has a memory of 0.2 MB. Figure 6(a) presents the



(a) Model size. (b) mIoU.

Fig. 6. The memory size and mIoU of PSPNet model with different CNN backbones.

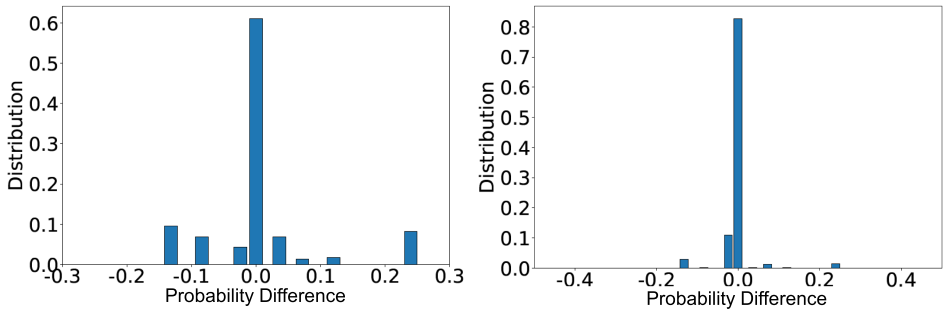
memory size of the PSPNet with different CNN backbones. We train these PSPNet models with a training dataset consisting of 800 images. Figure 6(b) shows the **mean intersection over union (mIoU)** of the evaluated PSPNet models over 200 testing images. The mIoU is the average of ratios of the overlap area to the union area between the ground-truth and predicted bubble areas in the testing images. From Figure 6, we can see that the PSPNet with the compressed MobileNet can be fit to the memory of the wireless camera. However, it can achieve a low mIoU value of 24.1% (i.e., image segmentation accuracy) only. Thus, we do not implement the PSPNet-based image segmentation framework in the wireless cameras.

6.2 SADRL-based Configuration Adaptation

To increase the lifetime of its battery, a wireless camera should be activated only when the air bubbles are blocked by the light reflection in the images captured by the main camera. Moreover, the wireless camera should increase its frame rate when the air bubbles appear in its captured images. Otherwise, it can keep a minimum frame rate. To achieve the goal, we develop a DRL-based solution to adapt the configuration for the activation mode and frame rate of the wireless cameras in response to the two exogenous stochastic processes, i.e., the time-varying presence of the air bubbles and light reflection.

6.2.1 Assessment of Markov Property. **Markov assumption (MA)** is a basic property of the systems where RL is applicable. Thus, we conduct experiments to assess if the above two stochastic processes satisfy the MA, i.e., $\mathbb{P}[X_t|X_{t-1}] = \mathbb{P}[X_t|X_{t-1}, \ldots, X_{t-M}]$ with $M \geq 0$, where the X_t represents the reflection/bubble presence at time t. Specifically, we consider the probability difference, denoted by $\Delta P = \mathbb{P}[X_k|X_{k-1}] - \mathbb{P}[X_k|X_{k-1}, \ldots, X_{k-M}]$, as an MA compliance metric. A lower absolute value of ΔP indicates better compliance. Figure 7 shows the distribution of ΔP with $M = 2$ for the reflection and bubble presence in a dataset consisting of 1,000 consecutive images that we captured in the factory. From Figure 7, we can see that these two stochastic processes have good compliance with the MA because their values of ΔP concentrate at zero. Thus, we formulate the configuration adaptation as an MDP problem and propose a DRL-based solution to learn the optimal adaptation policy.

6.2.2 MDP Formulation. Time is divided into identical intervals of τ seconds, which is referred to as the adaptation period. We divide the last image captured by the main camera into a number of equal sub-areas, denoted by Γ. At the beginning of every adaptation period, called a *time step*, the presence of the air bubbles and light reflection on these subareas is observed to configure the



(a) Reflection appearance. (b) Bubble appearance.

Fig. 7. Compliance with Markov assumption.

activation mode and frame rate for the wireless cameras. Let N denote the number of wireless cameras in the systems.

System state: The system state, denoted by s, is a 3-tuple: $s = (B, L, E)$, where the $B = [b_1, \ldots, b_\Gamma | b_i \in \{0, 1\}]$ and $L = [l_1, \ldots, l_\Gamma | l_i \in \{0, 1\}]$ represent the presence of the air bubbles and light reflection in the Γ image sub-areas, respectively, while the $E = [e_i, \ldots, e_N]$ is a residual battery energy level vector of N cameras.

Configuration action: The configuration action, denoted by a, is a vector $a = [\alpha_1, \ldots, \alpha_N, f]$, where $\alpha_i \in \{0, 1\}$ is the activation mode of the camera i and $f \in [f_{min}, f_{max}]$ is the frame rate of all wireless cameras. If α_i is equal to 1, the camera i is activated. Otherwise, it is set to the sleep mode. Moreover, the f_{min} and f_{max} denote the minimum and maximum frame rates, respectively.

Reward function: Let e_{min} denote the minimum of the remaining energy levels of N wireless cameras at the end of the k^{th} adaptation period and ϕ_{k} denote the accuracy of the bubble volume estimation of the images captured during the k^{th} period. Then, the immediate reward, denoted by $r(s, a)$, is defined as

$$r(s, a) = \lambda_1 \mathcal{N}(e_{\text{min}}) + \lambda_2 \mathcal{N}(\min\{\phi_k - \phi_{\text{req}}, 0\}), \tag{1}$$

where λ_1 and λ_2 are weights, ϕ_{req} is the required accuracy, and $\mathcal{N} = \max(x, 0)/x_{max}$ represents a normalization process.

The objective of the above MDP problem is to find an optimal adaptation policy that determines action a based on state s to maximize the expected reward over a long run, i.e., $\mathbb{E}[r(s, a)]$. As shown in Equation (1), the reward $r(s, a)$ is defined based on the weighted sum of the minimum remaining energy and the degree of violating the accuracy requirement. Thus, the optimal policy is to satisfy the accuracy requirement while maximizing the system lifetime, which is defined as the operational time of the system until the first wireless camera runs out of energy.

6.2.3 Offline Training. We adopt the learning framework of a DRL algorithm, called the **Proximal Policy Optimization (PPO),** to learn the optimal configuration adaptation policy, because PPO is a strong and simple baseline method in DRL algorithms. Typically, the DRL agent interacts with the system to learn the optimal policy. However, for the formulated problem, training the DRL agent at runtime faces the following two challenges. First, training needs long times to converge, which may lead to large energy consumption of the wireless cameras. Second, it is cumbersome to measure the camera's power and the bubble volume accuracy without the ground-truth labels during the online learning phase. To address these two challenges, we adopt an offline training approach, in which we collect an image dataset and measure the camera's power traces during the collection. The collected images are processed to determine the presence of the light

reflection and the air bubbles. Moreover, we use the measured power trace to model the camera's power consumed to capture an image. Then, we use the image dataset and the built power model to drive the offline training of the DRL agent. Finally, the trained DRL agent is used to adapt the configuration of the wireless cameras for the online bubble volume estimation in the production lines.

7 MADRL-BASED CONFIGURATION ADAPTATION

In this section, we formulate the multi-camera configuration adaptation problem as a Markov game and present the detailed design of our MADRL solution.

7.1 Markov Game

The **Markov game (MG)** [10] is an extension of the MDP and can be represented by the 4-tuple $(\mathbf{N}, \mathbf{S}, \mathbf{A}, \mathbf{R})$, where $\mathbf{N} = \{1, \ldots, N\}$ denotes the set of N agents for adapting the configurations of N wireless cameras, $S = \{s^1, \ldots, s^N\}$ is the set of states observed by N agents, $A = \{a^1, \ldots, a^N\}$ is the set of actions of N agents, and $R = \{r^1, \ldots, r^N\}$ is the set of reward functions of N agents. Specifically, the N agents have the same system state $s^1 = \ldots = s^N = (B, L, E)$, where the B, L, and E represent the presence of the air bubbles and light reflection in the Γ image sub-areas and residual battery energy levels of N cameras, respectively. The agent's action is a vector $a_i = [\alpha_i, f_i]$, where $\alpha_i \in \{0, 1\}$ and $f_i \in [f_{min}, f_{max}]$ are the activation mode and frame rate of the camera i, respectively. Moreover, the formulated MG is a fully cooperative game in which the N agents have the same reward $r^1 = \ldots = r^N = r$, where r is the reward function defined in Equation (1).

7.2 MADRL Solution

Existing studies [10] have proposed various DRL schemes to train and find the optimal solution of a multi-agent MG system, which can be divided into two main categories: centralized and distributed training schemes. In this article, we adopt a centralized DRL training scheme, called **multi-agent proximal policy optimization (MAPPO)** [28], to learn the optimal actions for N agents in our formulated MG. Centralized training simplifies the training process by learning only a centralized critic network and addresses the non-stationarity problem of multi-agent learning. Specifically, the MAPPO algorithm trains a critic network, denoted by $V(s; \phi)$, and N policy networks, denoted by $\pi(s; \theta_i)$ $(1 \leq i \leq N)$. The policy networks decide which configuration actions the wireless cameras should take to maximize their expected rewards. Meanwhile, the critic network evaluates the taken actions by computing the value functions. These networks interact with each other to learn the optimal configuration policy during the training phase. Algorithm 1 presents the detailed training procedure of the MAPPO algorithm.

Specifically, the training phase lasts for G epochs, each of which consists of T time steps. The first training begins with initializing the critic and policy networks with random weights of ϕ and θ_i $(1 \leq i \leq N)$. Then, a training epoch starts with initial states of N agents that are chosen randomly from the training data. At the beginning of the tth time step, the actions $a_t = [a_t^1, \ldots, a_t^N]$ of N agents are selected for state s_t using the current policy networks. The immediate reward r_t is obtained as a feedback signal obtained after executing the selected actions in a_t. The transition tuple $[s_t, a_t, r_t, s_{t+1}]$ is stored in the trajectory pool, denoted by ζ. The next step is to compute the advantage function values, denoted by A_t $(1 \leq t \leq T)$, of T transition samples stored in the pool ζ. At the end of the gth training epoch, a random mini-batch of H samples $\{(s_h, a_h, r_h, s_{h+1})\}_{h=1}^{h=H}$ is sampled from the pool ζ to update the critic and policy networks, as presented in lines 20–26 of Algorithm 1. In the first training process, all N policy networks are updated. Otherwise, the retraining process only retrains the agents for the wireless cameras with the position adjustment

ALGORITHM 1: Multi-agent proximal policy optimization (MAPPO) training framework

Input: G: number of training epoches; T: number of steps in each epoch; H: mini-batch size; ϵ: clipping parameter; γ: discount factor.

1 $\mathbf{N} = \{1, \ldots, N\}$: set of agents for N wireless cameras;
2 **if** *First Training* **then**
3 Initialize the N policy networks $\pi(s;\theta_1), \ldots, \pi(s;\theta_N)$ with random weights;
4 Initialize the critic network $V(s;\phi)$ with random weights;
5 **end if**
6 **for** $g = 1$ *to* G **do**
7 Sample initial states for N agents;
8 Initialize the trajectory pool $\zeta = []$;
9 **for** $t = 1$ *to* T **do**
10 **for** $i = 1$ *to* N **do**
11 Sample action $a_t^i \sim \pi(s_t;\theta_i)$ using the current policy;
12 **end for**
13 Execute the joint action $a_t = [a_t^1, \ldots, a_t^N]$;
14 Observe next states s_{t+1} and reward r_t;
15 Add $[s_t, a_t, r_t, s_{t+1}]$ to the pool ζ;
16 **end for**
17 **for** $t = 1$ *to* T **do**
18 Compute the advantage function value:
 $A_t = -V(s_t,\phi) + r_t + \gamma r_{t+1} + \cdots + \gamma^{T-t+1} r_{T-1} + \gamma^{T-t} V(s_T;\phi)$;
19 **end for**
20 Sample a mini-batch of $\{s_h, a_h, r_h, s_{h+1}\}_{h=1}^{H}$ with H samples;
21 Update the critic network $V(s;\phi)$ using the loss function:

$$L(\phi) = \frac{1}{H}\sum_{h=1}^{H}(-V(s_h;\phi) + r_h + \gamma r_{h+1} + \cdots + \gamma^{T-h} r_T)^2$$

  if *Retraining* **then**
22 $\mathbf{N}$ = set of agents for the wireless cameras with the position adjustment distance higher than a distance threshold;
23 **end if**
24 **for** $i = 1$ *to* $|\mathbf{N}|$ **do**
25 Update the policy network $\pi(s;\theta_i)$ using the loss function:

$$L^{CLIP}(\theta_i) = \frac{1}{H}\sum_{h=1}^{H}\hat{\mathbb{E}}_h\left[\min\left(\frac{\pi(s_h;\theta_i)}{\pi_{\text{old}}(s_h;\theta_i)}A_h, \text{clip}\left(\frac{\pi(s_h;\theta_i)}{\pi_{\text{old}}(s_h;\theta_i)}, 1-\epsilon, 1+\epsilon\right)A_h\right)\right],$$

  where $\pi_{\text{old}}(s_h;\theta_i)$ is the old policy network before the update;
26 **end for**
27 **end for**

Output: Optimal configuration policies of N agents: $\pi(s,\theta_1), \ldots, \pi(s,\theta_N)$.

distance higher than a distance threshold. Section 7.3 will present how to determine the cameras with the position adjustment for retraining. Moreover, the MAPPO algorithm uses the clipped surrogate objective function, denoted $L^{CLIP}(\theta_i)$, to update the policy network $\pi(s;\theta_i)$, as presented in line 25 of Algorithm 1. The clipped objective function helps ensure that the difference between the old policy and new policy is not too large, which is effective in trust region policy optimization [24].

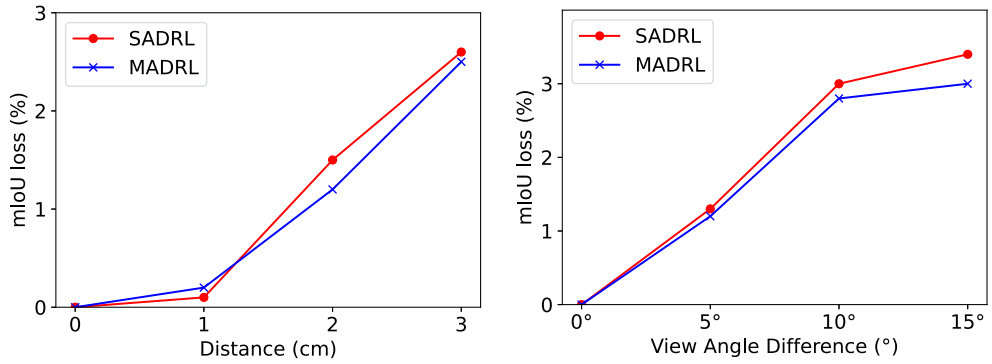


Fig. 8. mIoU loss due to the position adjustments.

7.3 MADRL Retraining

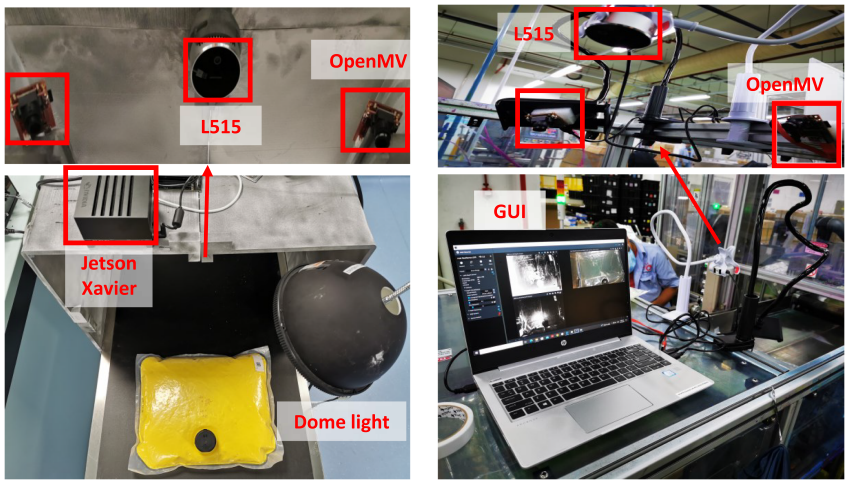
7.3.1 Impacts of Camera's Position Changes on Performance of Trained DRL Agents. Define the **6-degree-of-freedom (6DoF)** pose/position of the camera i as $P_i = (x_i, y_i, z_i, \omega_i, \gamma_i, \varphi_i)$, where x_i, y_i, z_i, ω_i, γ_i, and φ_i denote the x-coordinate/surge, y-coordinate/sway, z-coordinate/heave, roll, pitch, and yaw components of the 6DoF pose P_i, respectively. We conduct experiments to investigate impacts of the wireless camera position changes on the end-to-end bubble inspection accuracy. Specifically, we train both the SADRL and MADRL systems to learn the optimal configuration policies for $N = 2$ wireless cameras located at certain 2-DoF positions. The detailed settings of the DRL agents and training processes will be presented in Section 8.3. Then, we move one wireless camera from its original position along the x-axis direction to three new positions with a distance step size of 1 cm. We also move the camera to three more new positions whose roll angle changes are 5°, 10°, and 15°, respectively. Then, we measure the inspection accuracy loss when we apply the configuration policies learned with the original positions to adapt the parameters of the camera at the new positions. We use the mIoU as the inspection accuracy metric.

Figure 8 shows the mIoU loss of the SADRL and MADRL approaches under various changes in the camera's distance and view angle. Each mIoU loss point is the mIoU reduction due to the change of the camera position from the original position to a new position over 100 testing ink bag images. From Figure 8, we can see that the mIoU losses of both the DRL and MARL approaches increase with the distance and view angle difference between the old and new positions of the camera. Moreover, the mIoU loss is less than 1% when the distance and view angle difference are less than 1.8 cm and 3°, respectively. From these results, the DRL agents need to be retrained to maintain high inspection accuracy when the camera position change exceeds a certain threshold.

7.3.2 Retraining Process. We define $P_{\text{old}} = \{P_i | i = 1, \ldots, N\}$ and $P_{\text{new}} = \{P_j | j = 1, \ldots, N\}$ as the sets of the old and new positions of N wireless cameras. Let c_{ij}^{dist} and c_{ij}^{angle} denote the distance and view angle correlations between $P_i = (x_i, y_i, z_i, \omega_i, \gamma_i, \varphi_i) \in P_{\text{old}}$ and $P_j = (x_j, y_j, z_j, \omega_j, \gamma_j, \varphi_j) \in P_{\text{new}}$. They are calculated as

$$\begin{cases} c_{ij}^{\text{dist}} = 1 - \mathcal{N}(|x_i - x_j| + |y_i - y_j| + |z_i - z_j|), \\ c_{ij}^{\text{angle}} = 1 - \mathcal{N}(|\omega_i - \omega_j| + |\gamma_i - \gamma_j| + |\varphi_i - \varphi_j|), \end{cases} \tag{2}$$

where $\mathcal{N} = \max(x, 0)/x_{max}$ represents the normalization process that normalizes its input x to an output value in a range of $[0, 1]$. In the proposed MADRL approach, we only retrain the DRL agent of the camera whose distance and view angle correlations between its old and new position



(a) Lab testbed. (b) Factory trial.

Fig. 9. Testbed at lab and factory trials.

are higher than certain thresholds, denoted by $c_{\text{th}}^{\text{dist}}$ and $c_{\text{th}}^{\text{angle}}$, respectively. Differently, with the SADRL approach, we may need to retrain the agent to learn the new configuration policy for all wireless cameras even if only one camera is moved to the new position. Due to the use of the offline training approach, the retraining process involves the labeling process of the images captured by the cameras at the new positions. Thus, the MADRL can help reduce the training cost compared with the SADRL approach. Moreover, the MADRL retraining process also follows the procedure in Algorithm 1. However, in the end of each training epoch $1 \leq g \leq G$, we do not need to update the policy networks of the cameras whose distance and view angle correlations are less than $c_{\text{th}}^{\text{dist}}$ and $c_{\text{th}}^{\text{angle}}$, respectively.

8 TRIALS AND EVALUATION

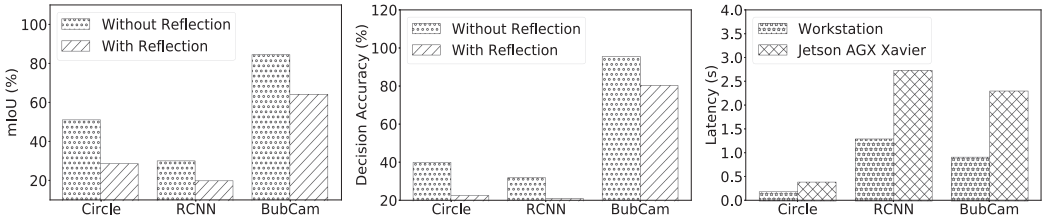
In this section, we present the deployments and evaluation of our proposed BubCam systems in lab and factory environments.

8.1 Testbed and Trials

8.1.1 Lab Testbed. We deploy a conveyor belt sized $1.5m \times 0.25m \times 0.7m$ as illustrated in Figure 9(a) to simulate the production line in a lab environment. Specifically, we manually inject different volumes of air bubbles into the ink bags. Then, we attach the cameras and fog nodes to a 3D-printed frame holder to capture images of the ink bags moving on the conveyor belt. Specifically, the deployed conveyor belt is capable of adjusting its rotation speed, which allows us to evaluate the impacts of different motion speeds on the image quality and processing accuracy. Moreover, we control the ambient lighting condition in the lab to create various light reflection conditions. This simulated system is used to drive the design and conduct controlled experiments to verify the performance of BubCam.

8.1.2 Factory Trials. We deploy the BubCam in HP's production lines to capture images of the manufactured ink bags moving through a conveyor belt as illustrated in Figure 9(b). We develop a software program based on an open source Python library, called Labelme[1], to manually label

[1]https://github.com/labelmeai/labelme



(a) mIoU (b) Decision accuracy (c) Latency.

Fig. 10. Comparison with existing bubble detection approaches.

the captured images in collaboration with HP's product engineers. Specifically, we first use the **graphical user interface (GUI)** of the developed program to manually create a polygon for cropping out the bubble areas. These bubble areas are processed to create the label of the captured images. Then, we work with the engineers to confirm the labeling results. However, this manual labeling process is tedious and extremely time-consuming. In addition, only a limited number of ink bags are available to be used for data collection. Thus, we only create a small dataset of images with confirmed ground-truth labels. Furthermore, we use our proposed knowledge-based labeling approach (cf. Section 5.2.2) to create more training images based on these confirmed labels.

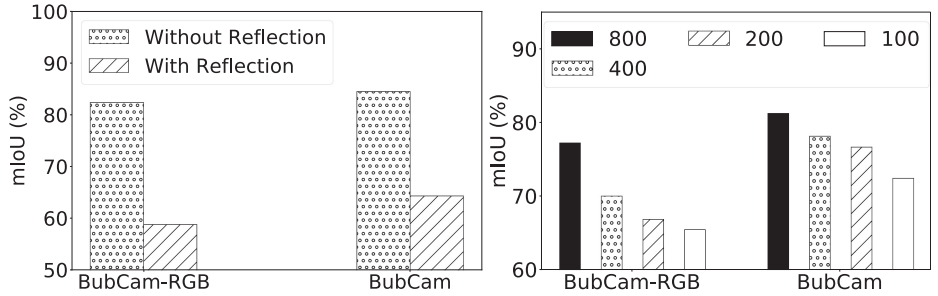
8.2 Evaluation of Single-cam BubCam at lab

8.2.1 Evaluation Settings. We use TensorFlow 2.1 and OpenCV 4.5.3 libraries to build the image processing pipeline and DRL model of BubCam in Python 3.8. We employ the following three performance metrics: *(1) mIoU* is the average of ratios of the overlap area to the union area between the ground-truth and predicted bubble areas in all testing images. A higher value of mIoU indicates better segmentation accuracy. *(2) Decision accuracy* is used to evaluate the accuracy in estimating the bubble volume. Let m_{gt} and m_{pred} denote the ground-truth and predicted numbers of the bubble pixels. Then, the volume estimation result of the ink bag is considered accurate if $|m_{\text{pred}-m_{gt}}|/m_{gt} \leq 0.2$. This accuracy metric is used for communication in the factories. *(3) Latency* is the end-to-end latency for estimating the air bubble volume of an image.

We compare BubCam with the following three baseline approaches: *(1) Circle* is a bubble detection approach proposed in [26]. It uses the CV algorithms to detect individual air bubbles as circles. *(2) RCNN* proposed in [11] employs an RCNN model to extract bounding boxes of all individual bubbles in the image. Then, a shape regression CNN is used to transform the extracted boxes into circles whose total number of pixels are considered as the bubble volume. *(3) BubCam-RGB* is a variant of our BubCam that uses the PSPNet to extract the bubble areas for the bubble volume estimation without depth fusion.

8.2.2 Evaluation Results. In this section, we evaluate the single-camera BubCam and three baseline approaches.

■ **Comparison with bubble detection approaches:** We evaluate the BubCam and two baseline approaches including the Circle and RCNN based on a dataset of 1,000 RGB images that are collected in our lab's conveyor belt system. In particular, each RGB image has one corresponding depth image. The dataset is divided into the training and testing sets by a ratio of 8:2. Figure 10 shows the mIoU, decision accuracy, and latency of the BubCam, Circle, and RCNN approaches on a total of 200 testing RGB images, which consists of 100 RGB images with/without the light reflection. From Figure 10, we can see that the presence of the light reflection reduces the image processing performance of three approaches. Moreover, BubCam always achieves the highest mIoU and decision accuracy among the three approaches. For instance, BubCam has an mIoU of 64.3% and 84.4%,



(a) Comparison with BubCam-RGB. (b) Impacts of pseudo labels.

Fig. 11. Comparison of BubCam's variants.

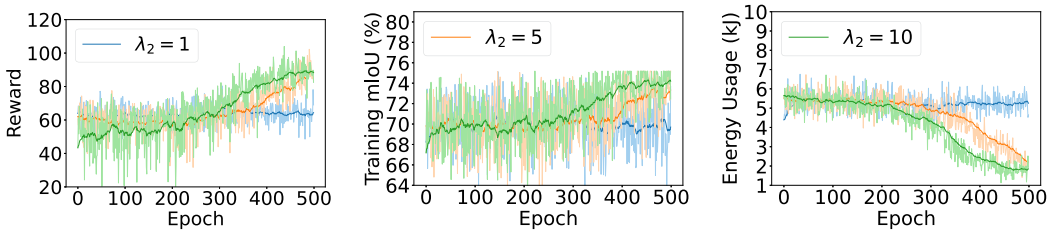
while RCNN has an mIoU of 20.3% and 30.4% in two cases with and without the light reflection, respectively. These results imply that BubCam achieves mIoU improvement of about 2x and 1.56x compared with RCNN. Furthermore, Figure 10(c) presents the latency of three approaches in the fog node (i.e., Jetson AGX Xavier) and a workstation with a 3.3 GHz CPU, a RTX8000 GPU, and a 48 GB RAM. From Figure 10(c), the Circle has the lowest latency since it employs the simple CV algorithms to detect the air bubbles. Differently, BubCam and RCNN have higher latencies due to the use of deep CNN models. However, BubCam can achieve lower latencies than RCNN. Specifically, BubCam can achieve 2.3-second latency on the Jetson AGX Xavier. These results demonstrate the superior performance of BubCam compared with the existing CV-based and CNN-based bubble detection approaches.

■ **Comparison between BubCam's variants:** Figures 11(a) and 11(b) show the impacts of the depth fusion and the number of pseudo labels on the mIoU of BubCam. Specifically, from Figure 11(a), with the presence of the light reflection, BubCam can achieve better accuracy than its BubCam-RGB (i.e., BubCam without depth fusion) in estimating the air bubble volume under the presence of the light reflection. Specifically, BubCam and BubCam-RGB have an mIoU of 64.3% and 58.8%, respectively, with the light reflection. These results demonstrate the usefulness of the depth fusion in BubCam. Moreover, Figure 11(b) shows the mIoU of the BubCam and BubCam-RGB on 200 testing images under various number of training images with the pseudo label. First, we use 400 images with ground-truth labels to create a small training dataset. Then, we additionally include a number of the pseudo images varying from 800 to 400 to 200 to 100 in the dataset for training the BubCam and BubCam-RGB. From Figure 11(b), we can see that more training pseudo images can help improve the accuracy of the BubCam and BubCam-RGB.

8.3 Evaluation of Multi-cam BubCam at Lab

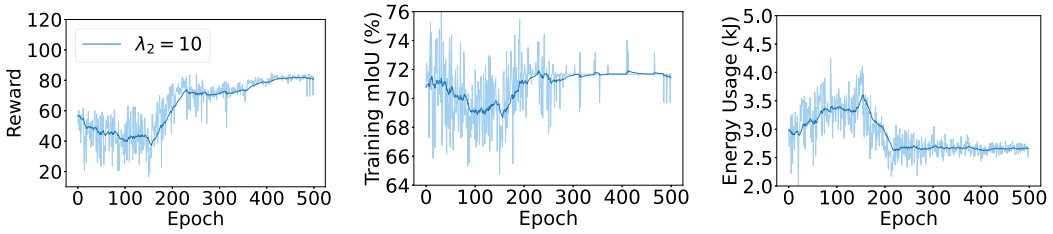
In this section, we present the evaluation settings and performance of the SADRL and MADRL approaches for adapting the parameters of the wireless cameras.

8.3.1 Evaluation Settings. We evaluate the performance of the multi-camera BubCam with two wireless cameras (i.e., $N = 2$). As mentioned in Section 4, the OpenMV Cam Plus is used to prototype the wireless cameras whose activation mode and frame rate are controlled by the DRL-based adaptation agent implemented in the fog node. Moreover, the OpenMV libraries and Micro Python 1.5.3 are used to implement the image preprocessing pipeline and camera's parameter configuration in the wireless cameras. In the fog node, we use Tensorforce 0.6.3 to implement the SADRL and MADRL approaches. Specifically, for the SADRL approach, the PPO agent consists of two neural networks, called the actor and value networks. Each network consists of an input layer, two hidden layers, and an output layer. Each hidden layer has 64 tanh units. The Adam optimizer with



(a) Training reward (b) Training mIoU (c) Training energy usage

Fig. 12. Multi-camera SADRL training results.



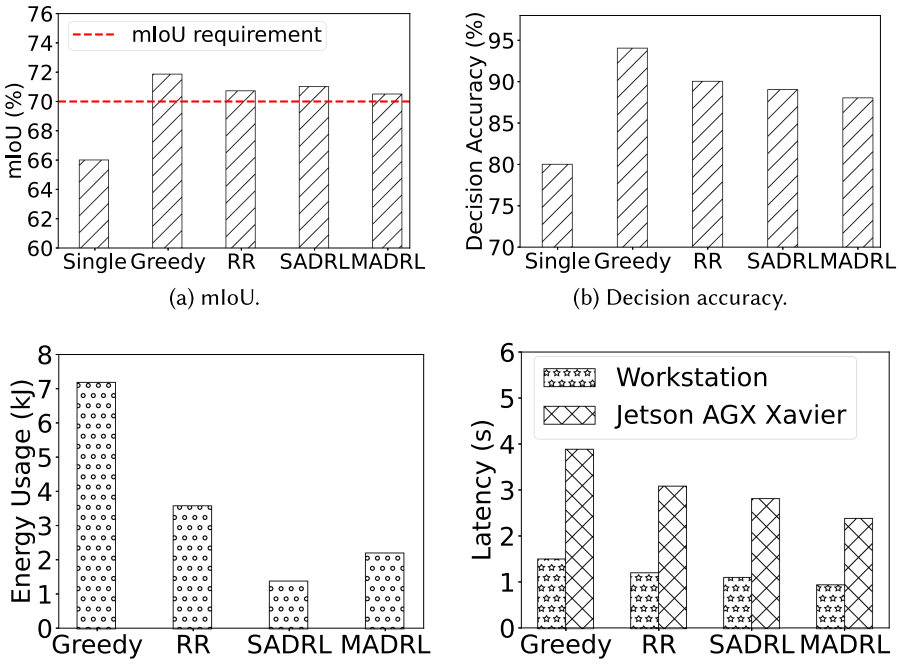
(a) Training reward (b) Training mIoU (c) Training energy usage

Fig. 13. Multi-camera MADRL training results.

a learning rate of 10^{-3} is used for training. The same neural network structure and parameters are used for each DRL agent of the MADRL approach. Moreover, the adaptation period τ is set to 2 seconds. At the beginning of every period, the DRL agent observes a system state s including the presence of bubbles and reflection in nine sub-areas (i.e., $\Gamma = 9$) of the main camera's last image and the remaining energy of the wireless cameras. Then, it selects an action a to configure the activation mode and frame rate of the two wireless cameras $f \in [f_{\min}, f_{\max}]$. The $f_{\min}$ and $f_{\max}$ are set to 0 and 30 FPS, respectively. The frame rate of the main camera L515 is fixed at 30 FPS. We use the mIoU of the labeled images captured during the k^{th} adaptation period as the accuracy ϕ_k to calculate the immediate reward $r(s, a)$.

8.3.2 Training Results. The weights λ_1 and λ_2 in Equation (1) affect the tradeoff between camera energy usage and compliance to the accuracy requirements. We evaluate the convergence of the DRL agent training under various settings for λ_1 and λ_2. Figure 12 shows the SADRL training traces of the reward, mIoU (i.e., accuracy), and total energy usage of the wireless cameras over 500 training epochs, each of which consists of 250 adaptation periods. λ_1 is set to 1, while λ_2 varies from 1 to 5 to 10. From Figure 12(a), we can see that with $\lambda_2 = 5, 10$, the reward increases and then saturates at around a certain value along the training epochs. Moreover, Figures 12(b) and 12(c) show that the accuracy and energy usage have increasing and decreasing overall trends when $\lambda_2 = 5, 10$. The results show that the training of the SADRL agent can be convergent with $\lambda_2 = 5, 10$. Differently, with $\lambda_2 = 1$, the reward, accuracy, and energy usage mostly remain stable during the training.

Figure 13 shows the training traces of the reward, mIoU, and energy usage of the MADRL approach in which λ_1 and λ_2 are set to 1 and 10, respectively. From Figures 13(a) through 13(c), the reward and mIoU increase and saturate at certain values after 150 epochs, while the energy usage has the opposite trend. Moreover, the SADRL and MADRL approaches have similar saturated values of the training reward, mIoU, and energy usage, which are about 80, 72%, and $2kJ$, respectively.



(a) mIoU. (b) Decision accuracy.

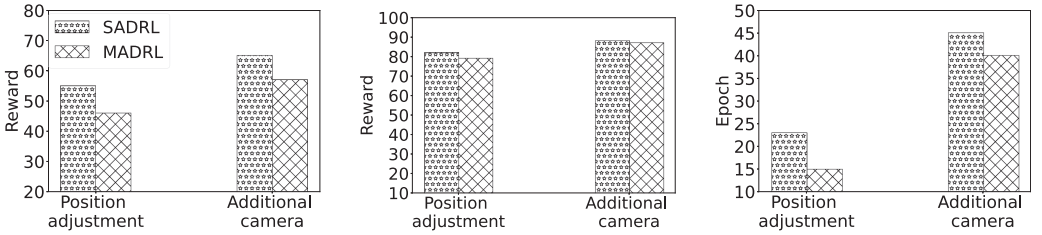
(c) Energy usage. (d) Latency.

Fig. 14. Comparison of multi-camera SADRL and MADRL approaches ($\lambda_1 = 1$ and $\lambda_2 = 10$) with baseline approaches in lab settings. Sub-figures (c) and (d) do not include the wireless cameras' total energy usage and latency of the single-camera BubCam system, respectively, because this system consists of the main camera only.

8.3.3 Execution Results at Lab. We compare the execution of the multi-camera SADRL and MADRL approaches with three baseline approaches: the single-camera BubCam, greedy, and **round-robin (RR)** approaches in the lab environment. The single-camera BubCam only uses the main camera with a constant frame rate of 30 FPS. Similar to the proposed DRL approach, the greedy and RR approaches additionally deploy the wireless cameras to assist the main camera. Specifically, in the greedy approach, all wireless cameras are always activated with a fixed frame of 30 FPS, while the RR approach activates one wireless camera in an adaptation period.

Figures 14(a) through 14(d) present the mIoU, decision accuracy, total energy usage, and latency of the single (i.e., single-camera BubCam), greedy, RR, SADRL, and MADRL approaches over 50 execution adaptation periods. From Figures 14(a) and 14(b), due to the use of the wireless cameras, the multi-camera BubCam with the two DRL approaches can achieve higher mIoU and decision accuracy than the single-camera BubCam. The greedy, RR, SADRL, and MADRL approaches can satisfy the mIoU requirement and achieve high decision accuracy. From Figure 14(c), we can see that our two DRL approaches achieve lower energy usage than two baseline approaches (i.e., the greedy and RR approaches). The MADRL approach has a slightly higher energy usage than the SADRL approach. Meanwhile, the greedy approach has the highest energy usage due to the activation of all wireless cameras over time. In summary, compared with the single-camera BubCam, the multi-camera BubCam with the DRL approaches can achieve better accuracy at the cost of higher latency and energy usage.

8.3.4 Retraining Results of DRL Approaches. In this section, we compare the retraining results of the SADRL and MADRL approaches for the multi-camera BubCam consisting of two wireless



(a) Jumpstart performance. (b) Asymptotic performance. (c) Time to threshold.

Fig. 15. Retraining performance results of multi-camera SADRL and MADRL approaches. The reward threshold is set to 80.

cameras 1 and 2 in the following two cases. In the first case, we move camera 1 from its old position to a new position at a distance of 2 cm. We also rotate the view angle of this camera by 10°. In the second case, we deploy an additional wireless camera 3 at a 6DoF pose, which has the same view angle and a 4 cm distance from camera 1's position. We calculate the distance and view angle correlations of the new camera 3 with the existing cameras 1 and 2. Then, we use the trained agent from an existing camera that has the highest sum of the distance and view angle correlations for adapting the configuration of the camera 3.

With the SADRL approach, we retrain the DRL agent to learn the new configuration policies for all cameras in both cases. With the MADRL approach, we retrain only the agent of camera 1 and 3 in two cases, respectively. We employ the following three retraining performance metrics [31]: *(1)* ***Jumpstart performance (JP)*** is the initial performance (i.e., reward) of the trained agent for the new system configuration before retraining. *(2)* ***Asymptotic performance (AP)*** is the ultimate reward of the agent after retraining. *(3)* ***Time to threshold (TT)*** is the learning time (i.e., number of retraining epochs) needed to reach a certain reward threshold, which is set to 80. A higher TT value indicates larger retraining latency and number of required training data samples. Figure 15 shows the JP, AP, and TT results of the SADRL and MADRL approaches. From Figure 15(a), the MADRL approach has lower JP than the SADRL approach in both the system reconfiguration cases. However, after the retraining process, the MADRL approach can recover the reward loss and achieve a similar AP value compared with the SADRL approach, as shown in Figure 15(b). Moreover, from Figure 15(c), the MADRL approach requires fewer training epochs to reach the reward threshold of 80 compared with the SADRL. This is because the MADRL retrains only one DRL agent for the repositioned or new camera. These results imply that the MADRL approach can achieve similar performance while requiring less training costs compared with the SADRL approach.

From Figure 15(c), the SADRL and MADRL approaches require 45 and 42 training epochs, respectively, to train a DRL agent for the newly added camera. In the evaluation, each training epoch consists of $T = 1,000$ adaptation periods, each of which lasts for 2 seconds. As such, the SADRL and MADRL approaches take about 25 and 23.33 hours, respectively, to train one DRL agent for the new camera. These results indicate high retraining costs of the DRL approaches.

8.4 Evaluation at Factory

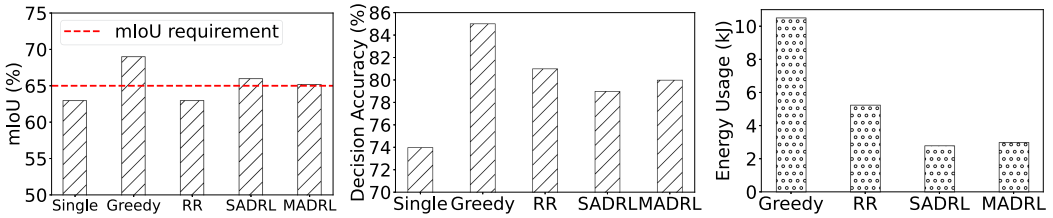
In this section, we evaluate the single-camera and multi-camera BubCam systems based on the images collected from the real ink bag production lines in the factory.

8.5 Performance of Single-camera BubCam

We compare the single-camera BubCam with the manual inspection approach in the factories (cf. Section 3.1) based on 400 RGB and 400 depth images collected in the production lines. Table 1

Table 1. Comparison of Single-camera BubCam with Manual Inspection Approach in Factory

Approach	Decision accuracy	Latency
Manual inspection	55.4%	5–10 minutes
Single-camera BubCam	74.5%	2.3 seconds



(a) mIoU. (b) Decision accuracy. (c) Energy usage.

Fig. 16. Comparison of multi-camera SADRL and MADRL approaches ($\lambda_1 = 1$ and $\lambda_2 = 10$) with baseline approaches in factory settings.

shows the decision accuracy and latency of the single-camera BubCam and manual inspection approaches on 200 testing RGB images. Due to the manual measurement procedure, the factory's current inspection approach achieves a low accuracy of 55.4% only. Moreover, it takes from 5 minutes to 10 minutes to manually inspect an ink bag in the factory. The single-camera BubCam can achieve accuracy improvement of 34% and latency reduction of up to 300x compared with the manual inspection. Moreover, the single-camera BubCam can take the place of human monitoring on the conveyor belt running 24 hours, which can help save a large amount of human resources.

8.6 Performance of Multi-camera BubCam

We compare our two multi-camera DRL approaches (i.e., SADRL and MADRL) with three baseline approaches (i.e., single-camera BubCam, greedy, and RR approaches). Specifically, we collect a total of 162,000 images that are captured by the multi-camera BubCam with the two wireless cameras at the factory. We manually label 1,200 images and use knowledge-based labeling (cf. Section 5.2.2) to generate the labels of the remaining images. We train the SADRL and MADRL approaches with a training dataset of 145,800 images. Figure 16 presents the mIoU, decision accuracy, and energy usage of the two proposed DRL and three baseline approaches over 16,200 testing images. From Figures 16(a) and 16(b), in the factory environment, all approaches suffer from an mIoU drop of about 3% to 5% compared with the lab environment results shown in Figures 14(a) and 14(b). The main reason for the mIoU and decision accuracy drop is the complex environmental conditions (e.g., the dynamic lighting condition) at the factory. Moreover, two proposed DRL approaches still can meet the mIoU requirement and achieve high decision accuracy with less energy usage compared with two baseline approaches.

9 DISCUSSION AND FUTURE WORK

The above evaluation results show that BubCam outperforms six baseline approaches, including the current manual inspection approach. Moreover, in the production lines, the ink bags move on the conveyor belt and appear in the camera's field of view one by one at a period of every 10 seconds. Thus, the automated quality inspection of each ink bag should be completed within 10 seconds. Meanwhile, BubCam can achieve an end-to-end latency of less than 3 seconds. This

indicates that BubCam can meet the latency requirement of the production lines. Regarding the inspection accuracy requirement, our design objective is that BubCam should obtain a higher accuracy than the manual inspection approach. This objective is achieved according to our evaluation results.

However, the performance of BubCam highly depends on the availability of sufficient labeled ink bag images for training and testing the BubCam's CNN-based image segmentation and DRL-based camera configuration adaptation models. In this study, we develop a manual labeling approach that requires a collocation with technicians in the production lines. This manual labeling approach is labor intensive and time-consuming. To facilitate the labeling process, we further design the knowledge-based labeling approach (cf. Section 5.2.2) that uses the motion speed of the ink bag and the ground-truth label of an image to label the consecutive images of the same ink bag. However, our knowledge-based labeling approach cannot translate a ground-truth image label to the image labels of different ink bags. Thus, it takes a lot of effort to create a big labeled image dataset of different ink bag sizes and colors for development and evaluation of BubCam.

Moreover, BubCam is susceptible to input perturbations that can be caused by the working environment noises/disturbances. In BubCam, we use a camera to capture images to train the deep CNN-based image segmentation model (i.e., PSPNet). The trained PSPNet may not show the same performance on the cameras deployed for estimating the air bubble volumes of manufactured ink bags in the production lines. This is because the quality of captured images across different cameras may be different due to the deviation in the light intensity condition of the cameras. To address this issue, we collected the training images in the production lines at different time periods such that the training dataset can cover many different lighting conditions. As future work, we will adopt the data augmentation methods proposed in [2, 19] to extend the training dataset. Specifically, we can capture the training images under different controlled lighting conditions and model the relationship between the captured images. Then, the modeled relationship can be used to augment the training dataset. As such, the trained PSPNet can have the capability to deal with images captured at unseen lighting conditions.

10 CONCLUSION

This article presents the design and implementation of a smart camera system, called BubCam, for the automated quality inspection of the ink bags manufactured in HP's ink production lines. BubCam employs a DL-based image segmentation and fusion pipeline to accurately estimate the volume of the air bubbles in the inspected ink bags under the complex settings and dynamic environment conditions of factories. Furthermore, BubCam additionally deploys multiple wireless cameras to achieve better accuracy based on the multi-view visual sensing information. To save battery power of the wireless cameras, the SADRL- and MADRL-based configuration approaches are proposed to adapt the configuration for the camera's activation mode and frame rate in response to the changes of the presence of air bubbles and light reflection. Extensive evaluation based on lab testbed experiments and factory deployment, as well as comparison with six baseline approaches, is conducted to show the effectiveness of the proposed BubCam systems.

REFERENCES

[1] Abolfazl Abdollahi and Biswajeet Pradhan. 2021. Integrated technique of segmentation and classification methods with connected components analysis for road extraction from orthophoto images. *Expert Systems with Applications* 176 (2021), 114908.

[2] Jiangshan Ai, Sihua Chen, Peng Deng, Libing Bai, Lulu Tian, and Jie Zhang. 2020. CycleGANs for semi-supervised defects segmentation. In *Proceedings of the International Conference on Sensing, Measurement & Data Analytics in the Era of Artificial Intelligence (ICSMD'20)*. 611–616.

[3] Xiao Bian, Ser Nam Lim, and Ning Zhou. 2016. Multiscale fully convolutional network with application to industrial inspection. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV'16)*. 1–8.
[4] Fei Chang, Mingyu Dong, Min Liu, Ling Wang, and Yunqiang Duan. 2020. A lightweight appearance quality assessment system based on parallel deep learning for painted car body. *IEEE Transactions on Instrumentation and Measurement* 69, 8 (2020), 5298–5307.
[5] Jiale Chen, Duc Van Le, Rui Tan, and Daren Ho. 2023. BubCam: A vision system for automated quality inspection at manufacturing lines. In *Proceedings of the ACM/IEEE 14th International Conference on Cyber-physical Systems (with CPS-IoT Week'23)*. 12–21.
[6] Jiale Chen, Duc Van Le, Rui Tan, and Daren Ho. 2024. NNFacet: Splitting neural network for concurrent smart sensors. *IEEE Transactions on Mobile Computing* 23, 2 (2024), 1627–1640.
[7] Zihao Chu, Lei Xie, Tao Gu, Yanling Bu, Chuyu Wang, and Sanglu Lu. 2022. Edge-eye: Rectifying millimeter-level edge deviation in manufacturing using camera-enabled IoT edge device. In *Proceedings of the ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN'22)*. 247–258.
[8] Jos de Jong, Roger Jeurissen, Huub Borel, Marc van den Berg, Herman Wijshoff, Hans Reinten, Michel Versluis, Andrea Prosperetti, and Detlef Lohse. 2006. Entrapped air bubbles in Piezo-driven inkjet printing: Their effect on the droplet velocity. *Physics of Fluids* 18, 12 (2006), 121511.
[9] Yucheng Fu and Yang Liu. 2019. BubGAN: Bubble generative adversarial networks for synthesizing realistic bubbly flow images. *Chemical Engineering Science* 204 (2019), 35–47.
[10] Sven Gronauer and Klaus Diepold. 2022. Multi-agent deep reinforcement learning: A survey. *Artificial Intelligence Review* 55 (2022), 1–49.
[11] Tim Haas, Christian Schubert, Moritz Eickhoff, and Herbert Pfeifer. 2020. BubCNN: Bubble detection using Faster RCNN and shape regression network. *Chemical Engineering Science* 216 (2020), 115467.
[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 770–778.
[13] Haixin Huang, Xueduo Tang, Feng Wen, and Xin Jin. 2022. Small object detection method with shallow feature fusion network for chip surface defect detection. *Scientific Reports* 12, 1 (2022), 1–9.
[14] Jarmo Ilonen, Tuomas Eerola, Heikki Mutikainen, Lasse Lensu, Jari Käyhkö, and Heikki Kälviäinen. 2014. Estimation of bubble size distribution based on power spectrum. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. 38–45.
[15] Intel. 2022. L515. https://www.intelrealsense.com/lidar-camera-l515/
[16] Yoram Koren, Xi Gu, and Weihong Guo. 2018. Reconfigurable manufacturing systems: Principles, design, and future trends. *Frontiers of Mechanical Engineering* 13 (2018), 121–136.
[17] Liangzhi Li, Kaoru Ota, and Mianxiong Dong. 2018. Deep learning for smart industry: Efficient manufacture inspection system with fog computing. *IEEE Transactions on Industrial Informatics* 14, 10 (2018), 4665–4673.
[18] Giuseppe Lisanti, Svebor Karaman, Daniele Pezzatini, and Alberto Del Bimbo. 2018. A multi-camera image processing and visualization system for train safety assessment. *Multimedia Tools and Applications* 77, 2 (2018), 1583–1604.
[19] Wenjie Luo, Zhenyu Yan, Qun Song, and Rui Tan. 2021. PhyAug: Physics-directed data augmentation for deep sensing model transfer in cyber-physical systems. In *Proceedings of the 20th International Conference on Information Processing in Sensor Networks (co-located with CPS-IoT Week'21)*. 31–46.
[20] Haocheng Ma and Lihui Peng. 2019. Vision based liquid level detection and bubble area segmentation in liquor distillation. In *Proceedings of the IEEE International Conference on Imaging Systems and Techniques (IST'19)*. 1–6.
[21] Wenyan Pai, Jin Liang, MingKai Zhang, Zhengzong Tang, and Leigang Li. 2022. An advanced multi-camera system for automatic, high-precision and efficient tube profile measurement. *Optics and Lasers in Engineering* 154 (2022), 106890.
[22] Christian Salim, Abdallah Makhoul, Rony Darazi, and Raphaël Couturier. 2019. Similarity based image selection with frame rate adaptation and local event detection in wireless video sensor networks. *Multimedia Tools and Applications* 78 (2019), 5941–5967.
[23] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. MobileNetV2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4510–4520.
[24] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv* (2017).
[25] Judi E. See, Colin G. Drury, Ann Speed, Allison Williams, and Negar Khalandi. 2017. The role of visual inspection in the 21st century. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 61. 262–266.
[26] Nataliya Strokina, Jiri Matas, Tuomas Eerola, Lasse Lensu, and Heikki Kälviäinen. 2012. Detection of bubbles as Concentric Circular Arrangements. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR'12)*. 2655–2659.

[27] Haochen Wang, Zhiwei Shi, Yafei Qiao, Fan Yang, Yuzhe He, Dong Xuan, and Wei Zhao. 2023. Autonomous and cost-effective defect detection system for molded pulp products. In *Proceedings of the ACM/IEEE 14th International Conference on Cyber-physical Systems (with CPS-IoT Week'23)*. 1–11.

[28] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. 2022. The surprising effectiveness of PPO in cooperative multi-agent games. *Advances in Neural Information Processing Systems* 35 (2022), 24611–24624.

[29] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. 2017. Pyramid scene parsing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2881–2890.

[30] Siyuan Zhou, Duc Van Le, Rui Tan, Joy Qiping Yang, and Daren Ho. 2023. Configuration-adaptive wireless visual sensing system with deep reinforcement learning. *IEEE Transactions on Mobile Computing* 22, 9 (2023), 5078–5091.

[31] Zhuangdi Zhu, Kaixiang Lin, Anil K. Jain, and Jiayu Zhou. 2023. Transfer learning in deep reinforcement learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 11 (2023), 13344–13362.

Received 14 July 2023; revised 20 November 2023; accepted 12 January 2024