applied sciences

MDPI

Article

HSV Color-Space-Based Automated Object Localization for Robot Grasping without Prior Knowledge

Hyun-Chul Kang [1], Hyo-Nyoung Han [1], Hee-Chul Bae [1], Min-Gi Kim [1], Ji-Yeon Son [1] and Young-Kuk Kim [2,*]

[1] Electronics and Telecommunications Research Institute, Daejeon 34129, Korea; kauni@etri.re.kr (H.-C.K.); hyonyoung.han@etri.re.kr (H.-N.H.); hessed@etri.re.kr (H.-C.B.); kmk82@etri.re.kr (M.-G.K.); jyson@etri.re.kr (J.-Y.S.)

[2] Department of Computer Engineering, Chungnam National University, Daejeon 34134, Korea

* Correspondence: ykim@cnu.ac.kr; Tel.: +82-42-821-5450

Abstract: We propose a simple and robust HSV color-space-based algorithm that can automatically extract object position information without human intervention or prior knowledge. In manufacturing sites with high variability, it is difficult to recognize products through robot machine vision, especially in terms of extracting object information accurately, owing to various environmental factors such as the noise around objects, shadows, light reflections, and illumination interferences. The proposed algorithm, which does not require users to reset the HSV color threshold value whenever a product is changed, uses ROI referencing method to solve this problem. The algorithm automatically identifies the object's location by using the HSV color-space-based ROI random sampling, ROI similarity comparison, and ROI merging. The proposed system utilizes an IoT device with several modules for the detection, analysis, control, and management of object data. The experimental results show that the proposed algorithm is very useful for industrial automation applications under complex and highly variable manufacturing environments.

Keywords: object localization; HSV color space; teaching-less robot; smart factory

Citation: Kang, H.-C.; Han, H.-N.; Bae, H.-C.; Kim, M.-G.; Son, J.-Y.; Kim, Y.-K. HSV Color-Space-Based Automated Object Localization for Robot Grasping without Prior Knowledge. *Appl. Sci.* **2021**, *11*, 7593. https://doi.org/10.3390/app11167593

Academic Editor: Manuel Armada

Received: 30 June 2021
Accepted: 16 August 2021
Published: 18 August 2021

1. Introduction

The smart factory describes a future state of fully connected and functioning manufacturing systems conducting all the required tasks, from customer orders to production, without the need for human involvement [1].

Recently, due to the COVID-19 pandemic, factories have accelerated the transition to contactless services in the manufacturing industry and smart factory transformation by utilizing advanced ICT technologies such as AI, IoT, cloud, robots, and big data. Owing to the different characteristics of the manufacturing and information fields, there are still many technical problems to be solved to accelerate the path of smart factories [2].

In the manufacturing environment of the future, the manufacturing trend is changing from mass production systems to personalized production systems [3,4].

Robots, which serve as advanced manufacturing equipment in smart factories, are important to solve problems such as the decrease in the labor force and specialized human resources due to aging, and the increase in the demand for human convenience. Robots are widely used, with high repeatability, in various industrial processes, such as transfer, picking, painting, welding, and assembly processes, to optimize factory productivity. However, in a flexible production environment, changes in the product lead to a need for frequent robot teaching. The operator manually performs the robot teaching, which requires considerable time and effort [5]. The role of machine vision is important for the intelligent and automated teaching tasks of robots without human intervention.

This is because robot machine vision improves factory productivity in the manufacturing processes (defect finding, transfer, material handling, welding, assembly, etc.); it also enables factories to meet high levels of automation and high-quality manufacturing

requirements because it provides visual and judgment information (environmental recognition, information acquisition, autonomous behavior, etc.) that enables a robot to act similar to a human being.

Computer vision technology is being used and developed in various industries, such as autonomous driving, medical care, manufacturing, and logistics [6]. However, in manufacturing factories with high variability, it is difficult to accurately detect object localization and information due to various environmental factors, such as noise around objects, light reflections, and illumination interference.

To solve this problem, we propose a robust algorithm for object localization that can easily and quickly respond to changes in the external environment by improving a previous work [7].

The remainder of this paper is organized as follows: In Section 2, we describe the research trends related to object detection for changes in the external environment in computer vision. Section 3 introduces the robot grasping scenario in the structure of a personalized manufacturing service. Section 4 proposes an automated object localization algorithm based on the HSV color space. In Section 5, we analyze the performance evaluation and test results. Finally, Section 6 presents the conclusions.

2. Related Work

2.1. Background Subtraction Algorithms

Background subtraction [8] is a method of extracting the region of interest of an object by using the difference between the background image and the current input image. This algorithm is widely used in image processing and video applications [9]. The background subtraction must accurately model the background to detect an object. However, the background changes over time, resulting in changes in the background values. Background subtractions are sensitive to external environment changes (brightness of light, reflection, shadow, etc.) and noise, making it difficult to detect objects [10]. Representative methods of this algorithm include the mean filter, median filter method, adaptive thresholding, and Gaussian mixture methods that make the average of the previous image the background of the current image [11].

Adaptive thresholding [12,13] is a variable binarization algorithm that considers spatial changes in lighting. It divides an image into a plurality of areas and then processes each region by specifying a threshold. Background modeling that is based on the Gaussian mixture model [14], which uses a statistical probability distribution, responds to changes in the external environment and models the distribution of data with multiple Gaussian probability density functions. However, there are many difficulties in minimizing noise and accurately detecting objects in the background due to various flow changes over time (lighting conditions, shadows, weather changes due to sunlight, etc.).

There are various background subtraction techniques, such as optical flow and shadow detection [15–20].

2.2. Feature Extraction and Region-Proposal-Based Algorithms

The features of the object extracted from the image mainly utilize color and structural feature points (keypoints). A typical method that utilizes color characteristics is the color histogram method [21]. It has the advantage that color information can be grouped and mapped in object similarity discrimination and is not affected by rotation or changes in the position [22]. However, if the color information does not include spatial information, it is discriminated from another image. The structural features of an object can become its local invariance features, mainly using corners (i.e., keypoints, interest points, salient points, and feature points) [23]. To be a good feature point, it must be resilient to changes in the shape, size, and position of the object, and it must be detectable even if the camera's viewpoint and lighting change.

The Harris corner [24] detection has robust performance against changes in translation, rotation, intensity shift, and illumination. However, because it is sensitive to image scaling,

it has the disadvantage of the necessity to extract feature points from images of various scales. The histogram of oriented gradients (HOG) [25] is an algorithm that divides the target area into cells of a certain size and extracts and classifies objects with a histogram of the direction components of the cell gradation. HOG has some robust characteristics against local changes. The histogram is normalized by its neighboring cells to increase the robustness to texture and illumination variations [26].

The scale-invariant feature transform (SIFT) is an algorithm that extracts features that do not change in image size and rotation. This algorithm extracts SIFT features from two different images, matches the features closest to each other, finds the corresponding part, and maps them. It is robust against distortion, noise, and lighting changes, but it has the disadvantages of complicated computation and slow speed [27,28].

Otsu [29] found the optimal boundary value that divides the image into two classes based on the observed distribution of pixel values. The threshold is set by maximizing the dispersion between the background and object. This algorithm is very effective when a single object has a different brightness distribution than the background, but when the internal brightness distribution of the object is not constant, only a part of the object is detected.

GrabCut [30] is a graph-cut-based partitioning algorithm used to separate the foreground and background of an image. This algorithm undertakes image segmentation based on the Grab cut [31,32] algorithm, which uses the distribution of user-specified regions to minimize the energy of each pixel and find zones. However, GrabCut is not an automatic object extraction algorithm but one that requires the user to intervene and extract the required object.

The convolutional neural network (CNN) [33] technique is a widely used deepening algorithm used to detect objects such as images, videos, texts, and voices. Face recognition algorithms can be applied to discover changes in lighting angles, brightness, facial expressions, poses, etc. based on CNN [34]. However, large visual variations of faces, such as occlusions, pose variations, and illumination changes, pose great challenges for this task in real applications [35].

To minimize external environmental changes, deep learning algorithms such as R-CNN [36], Fast R-CNN [37], and YOLO [38] have attracted attention in recent years [39–41]. However, deep-learning-based object detection has many constraints, such as a large amount of training and test data, a long learning time, an annotation task, and a large amount of calculation.

Object detection determines object classification and localization in an image. Owing to large variations in viewpoints, poses, occlusions, and lighting conditions, it is difficult to perfectly accomplish object detection with an additional object localization task [39].

The various algorithms to improve performance degradation due to noise and lighting changes have been suggested. However, the models need to be updated through mathematical modeling techniques such as machine learning or deep learning to recognize the location and the shape of an object accurately whenever there are some significant changes in the external environment. Thus, we propose a new algorithm to improve these problems, which is robust to the noise around objects, shadows, light reflections, and illumination interferences because it can separate the brightness information and the color information in the HSV color model. In addition, it does not require user intervention and prior knowledge, and it responds immediately to some changes in the external environment.

Our work focuses on object localization in complex and ever-changing manufacturing environments.

2.3. RGB and HSV Color Space Overview

Color space is a mathematical model that represents color information as three or four different color components [42]. In the RGB color model, all other colors can be produced by adding the primary colors: red, green, and blue [43]. It can combine color expression with these three primary colors of light in different proportions. RGB is primarily used to

represent colors on computer monitor screens, televisions, and color printers. The HSV color space is represented by H (hue), S (saturation), and V (value). The HSV color space corresponds closely to the human perception of colors [44]. It is suitable as a color space that deals with human visual perception. HSV color-space-based image segmentation is better than RGB color space [45].

The HSV model, also known as hue, saturation, and brightness (HSB), defines a color space in terms of three constituent components [46]:

- Hue—the color type (such as red, blue, or yellow) ranges from 0 to 360°;
- Saturation—the "vibrancy" of the color ranges from 0 to 100%;
- Value—the brightness of the color ranges from 0 to 100%.

The transformation of color images in the RGB into the HSV color space is achieved as in Equations (1)–(3) [42].

$$H = \arccos \frac{\frac{1}{2}(2R - G - B)}{\sqrt{(R - G)^2 - (R - B)(G - B)}} \quad (1)$$

$$S = \frac{\max(R, G, B) - \min(R, G, B)}{\max(R, G, B)} \quad (2)$$

$$V = \max(R,\ G,\ B) \quad (3)$$

3. HSV Color-Space-Based Automated Object Localization Scenario for Robot Grasping in Factory-as-a-Service (FaaS) Platform

3.1. FaaS Platform Architecture and Line Configuration

FaaS (factory as a service: personalized manufacturing service) is a concept that provides an IoT-based smart factory that supports personalized production to individuals or companies in the form of a service [4,7]. The FaaS platform provides open manufacturing services based on a smart factory that provides flexible production for personalized products based on ICT and digital facilities (3D printers, robots, etc.).

Figure 1 depicts the structure of the FaaS platform. The FaaS platform [4] consists of three layers: a cloud layer, a manufacturing operation layer, and an execution control layer. The FaaS cloud layer provides individuals or small- and medium-sized enterprises with a web-based personalized manufacturing service that supports the entire process from ordering prototypes or products to manufacturing and distribution. The FaaS operation layer is a smart factory dynamic manufacturing operation technology (dynamic planning/scheduling, virtual simulation, production monitoring, real-time process control, situational awareness/analysis, etc.). The FaaS execution control layer connects and controls manufacturing equipment such as post-processing equipment, e.g., polishing, robots, and inspection equipment, based on the Internet of Things (IoT).

The FaaS testbed was designed and built in a micro smart factory to enable small batch production of various types for individual idea realization. The main facilities of the FaaS testbed consist of 34 IoT-based manufacturing facilities, including robots, CNCs, and vision inspection machines [7]. The FaaS production line consists of a line that produces customized products and a line for post-processing in the form of an octagon modular factory. The first product line consists of additive manufacturing facilities (3D printers) to produce a variety of customized products. The second line for post-processing consists of facilities that perform processes such as inspection, assembly, and cutting of products produced from additive manufacturing facilities. In the first production line, the robot handler transfers the produced product to the buffer process. The tower handler in the second post-processing line transfers the products loaded in the buffer to the next process for post-processing in the second line.

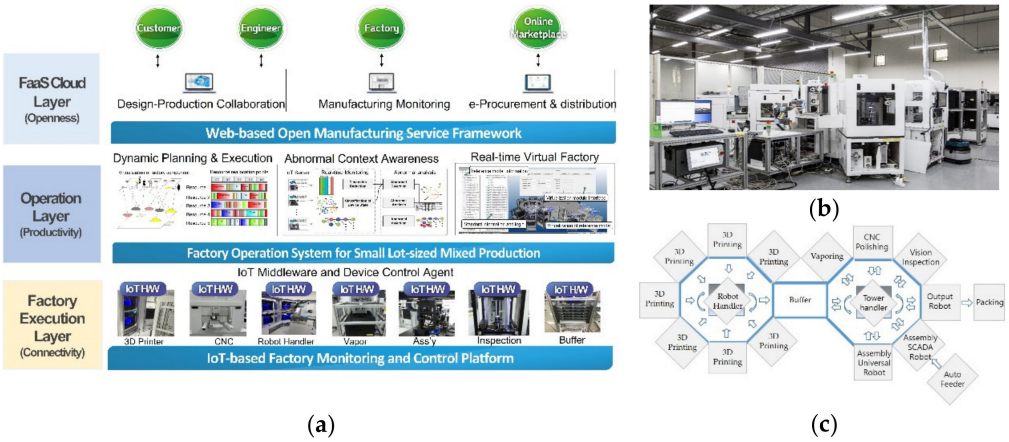


Figure 1. FaaS platform configuration: (**a**) FaaS platform architecture; (**b**) real manufacturing line; (**c**) octagonal modular factory.

3.2. IoT-Based Device Management for Robot Grasping in FaaS Platform

Major facilities are managed through the FaaS Manager in the execution control layer of the FaaS platform. FaaS manager manages IoT-based manufacturing facilities (initial setting, control, event, status information provision, etc.). IoT devices integrate with sensor devices (cameras, pressure, illuminance sensors, etc.) and manufacturing facilities (3D printers, robot handlers, etc.) to collect facility status information, receive events, and control and monitor functions [7].

Figure 2 shows the structure and function blocks of the IoT device connected to the robot handler for product transfer [7].

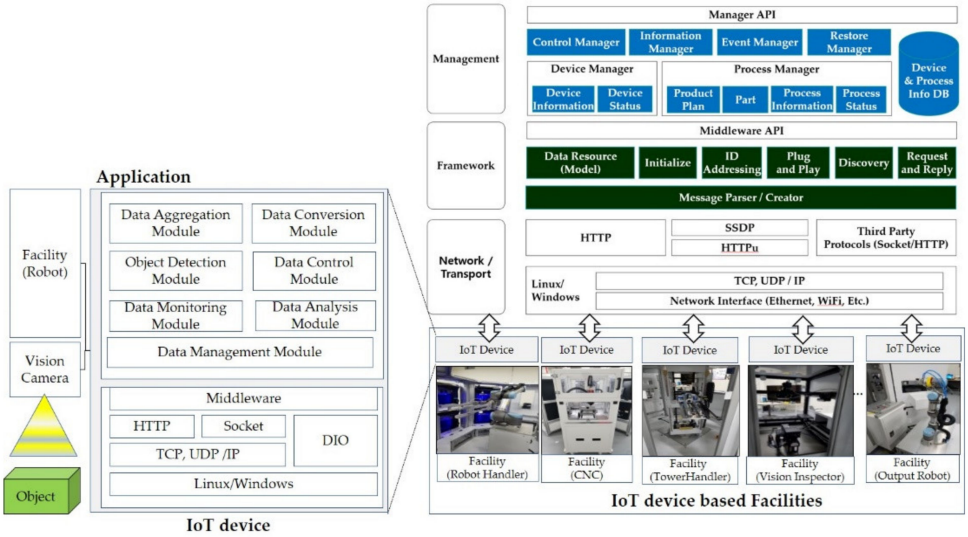


Figure 2. IoT device management and FaaS manager in the factory execution layer.

3.3. Robot Grasping Scenario in FaaS Platform

Figure 3 depicts a sequence flow diagram for automated object feature extraction when the robot handler transfers the product to the buffer process after the product is manufactured in an additive manufacturing facility [7]. With regard to the robot grasping scenario of the previous work [7], we propose an HSV color-space-based automated object localization algorithm without prior knowledge.

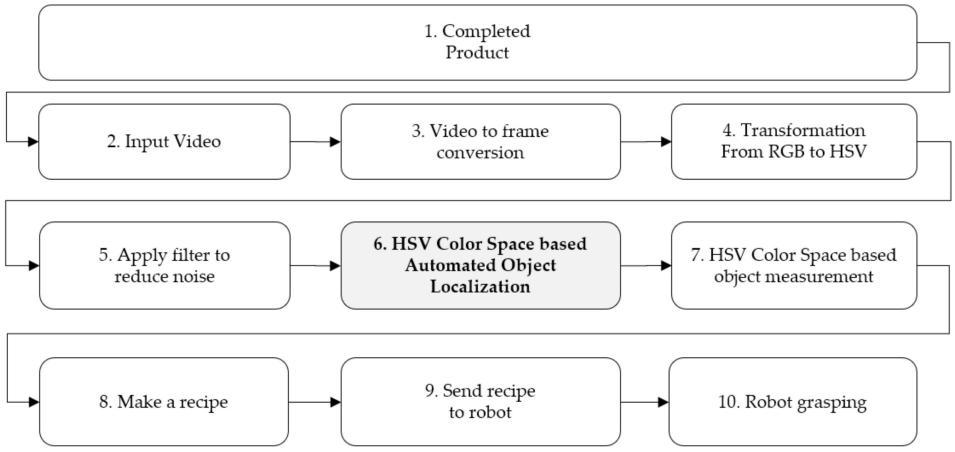


Figure 3. Sequence scenario of automated object feature extraction for robot grasping.

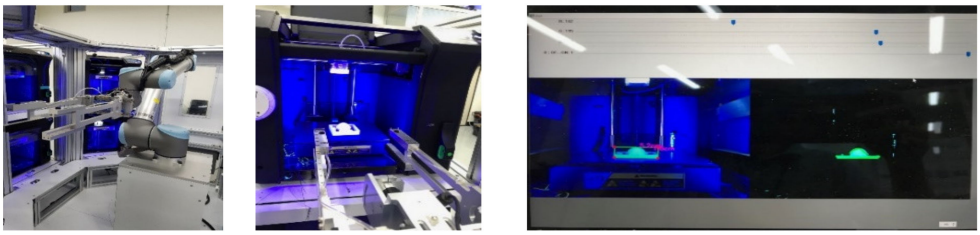
The IoT-based robot handler collects an image of the product from the data aggregation module through a vision camera. The data conversion module converts the collected videos into images and converts them into an HSV.

The object detection and analysis module works in HSV color-space-based automatic object localization and measurement. The data control module transmits the created recipe to the robot through the TCP socket and performs the grasping function.

4. HSV Color-Space-Based Automated Object Feature Extraction

4.1. Motivation

In a previous study [7], HSV color-space-based robot grasping was conducted by measuring object detection and size. However, the HSV color-space-based product detection algorithm requires human intervention and preset settings (color designation, setting threshold value for color identification, etc.), in addition to product size measurement, whenever a product is changed. In addition, even if the user designates a specific color for the product to be recognized in advance, the color of the target product appears to be deformed due to external environmental changes (weather, illumination interference, shadows, reflections, etc.). In a complex and highly volatile factory environment, users have many difficulties in automating object recognition based on color. To solve this problem, we propose a robust and simple HSV color-space-based automated object localization algorithm that can respond in real time to changes in the external environment without prior knowledge by improving the contents of previous work. The previous work is shown in Figure 4.



(**a**) (**b**) (**c**)

Figure 4. HSV color-space-based robot grasping in previous work in FaaS platform: (**a**) robot handler; (**b**) robot grasping; (**c**) user's HSV color configuration and monitoring GUI.

4.2. HSV Color-Space-Based Automated Object Localization Algorithm

The HSV color-space-based automated object localization algorithm consists of four steps, as shown in Figure 5:

- HSV color-space-based auto HSV color picking;
- Region of interest (ROI) random sampling;
- HSV similarity comparison;
- ROI merge operation.

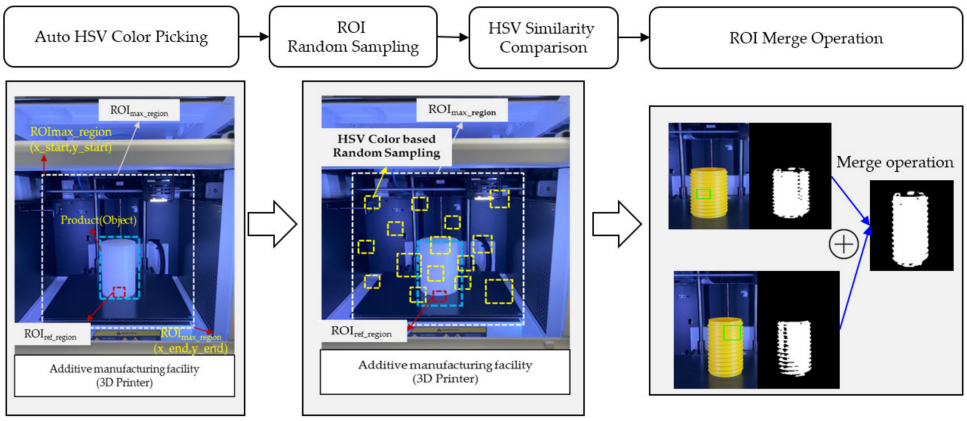


Figure 5. Process HSV color-space-based automated object localization algorithm.

The process is described in Sections 4.2.1 and 4.2.3.

4.2.1. Configuration of Reference ROI and HSV Color-Space-Based Auto HSV Color Picking

The first step of the proposed algorithm is to set the reference region of interest (ROI) and maximum ROI information to automatically recognize the product feature information (color, position) based on the HSV color space in the additive manufacturing facility (3D printer) from the camera device of the robot handler. Each of the maximum ROIs consists of one large bounding box. The bounding box is rectangular, which is determined by the x_start and y_start coordinates of the upper-left corner of the rectangle and the x_end and y_end coordinates of the lower-right corner. Algorithm 1 presents the algorithm for HSV color-space-based auto color picking. ROI_{max_region} and ROI_{ref_region} represent the bounding box size of the maximum ROI and the reference ROI, respectively. The reference ROI is set around the center of the bottom of the 3D printer, which is the reference position information where the additive manufacturing to produce the product is first started. The maximum ROI is set slightly higher than the maximum size of the product produced by the additive manufacturing facility. The reference ROI extracts reference color data to automatically identify the color of an object in the input video from the vision camera of the robot handler after product production is completed using a 3D printer.

Algorithm 1. Auto HSV Color Picking

1: Configuration of ROI_{max_region} from Robot Handler
2: Set ROI_{max_region} location info from bounding box (x_start, y_start, x_end, y_end)
3: Input Video from IoT-based Robot Handler
4: Video to image frame conversion and transformation from RGB to HSV through Equations (1)–(3)
5: Set ROI_{ref_region} location info from bounding box (x_start, y_start, x_end, y_end)
6: Picking automated HSV Color Extraction (HSV values) from ROI_{ref_region}
7: Registration color threshold of ROI_{ref_region}
8: Set H, S, V upper and lower bound threshold, ROI_{ref_region} location

The reason for setting the reference ROI is to solve the problem of the user's need to set the HSV threshold value every time because the color of the product varies depending on the order.

Auto HSV color picking extracts and stores the H, S, and V information through the camera of the robot handler for the color information of the reference ROI after the production is completed by the 3D printer.

Auto HSV color picking sets the upper and lower thresholds of the stored H, S, and V information. The upper and lower threshold values are set as the maximum and minimum ranges based on the extracted HSV values. It detects changes in product color owing to illumination interference or changes in light conditions.

4.2.2. HSV Color-Based ROI Random Sampling and HSV Color Similarity Comparison

In this study, we propose an HSV color-space-based ROI random sampling and HSV color similarity algorithm to automatically recognize and accurately reflect the shape features of objects that are affected by lighting conditions, shadows, reflections, etc.

ROI random sampling is used to automatically recognize the shape of an object from complex and volatile external environment changes such as shadows, reflections, or lighting interferences, etc. in factories. ROI random sampling is advantageous for accurate positioning and recognition of an object because it responds quickly and easily to some changes in the lighting environment without any human intervention or prior learning.

HSV color similarity can be defined as the extent of color matching with the reference ROI by calculating the distances for H, S, and V between the colors of the randomly retrieved ROI and reference ROI. The definition of the similarity threshold means the boundary value of whether the HSV color similarity matches. Algorithm 2 presents the algorithm of the HSV color-based ROI random sampling and similarity. The $ROI_{sampling\_max}$ variable is the ROI max count of the ROI random sampling. $ROI_{sampling\_i}$ represents the initial value of the number of sampling ROIs. Similarity threshold refers to the threshold of HSV color similarity. The $ROI_{TempList}$ variable is a temporary array list that registers random sampling ROIs with high HSV color similarity, compared to the reference ROI. ROI random sampling randomly searches for the ROI to detect the location of the product in various places inside the 3D printer from the maximum ROI. ROI random sampling extracts the position values of the searched random ROI. Then, the RGB of the random ROI is converted into an HSV, and the HSV value is extracted.

Algorithm 2. ROI Random Sampling and Similarity Comparison

Start ROI Random Sampling and Similarity Comparison in ROI_{max_region}
Set $ROI_{sampling\_max}$ = Max_sample_count
For $ROI_{sampling\_i}$ = 0: $ROI_{sampling\_max}$
Extract Random $ROI_{sampling\_i}$ location info from bounding box (x_start, y_start, x_end, y_end)
$ROI_{sampling\_i}$ → Transformation from RGB to HSV through Equations (4)–(6)
Picking automated HSV Color Extraction (HSV values) from $ROI_{sampling\_i}$
Compute HSV Color Similarity of ROI_{ref_region} between $ROI_{sampling\_i}$
Retrieve HSV Color Values → (H1, S1, V1 of ROI_{ref_region} and H0, S0, V0 $ROI_{sampling\_i}$)
  Compute HSV Color Distance by Equation (7)
IF($Similarity_{threshold}$ > Distance) then
  Set HSV Upper(max) and Low(min) bound threshold from $ROI_{sampling\_i}$
  Add HSV value, Upper/Lower bound threshold of $ROI_{sampling\_i}$ in $ROI_{TempList}$
Else
  $ROI_{sampling\_i}$ = $ROI_{sampling\_i}$ + 1
IF($ROI_{sampling\_i}$ > Max_sample_count) then
    Action $ROI_{TempList}$ Merge Operation
Else
    Continue:
End

The HSV color similarity compares the similarity between the HSV color of the reference ROI and the searched HSV color value of the random ROI. HSV color similarity is used to accurately classify product shape (position, color) as the HSV value of a random ROI can be a shape property inside the object or a background property outside the object shape

due to changes in the external environment (lighting condition, reflection, illumination, etc.). The HSV color similarity algorithm compares the HSV value of the reference ROI with the HSV color value of the random ROI and stores only the HSV value of the random ROI closest to the distance.

The following is an algorithm for the HSV color-space-based similarity distance, as in Equation (7):

$$h = \frac{\min(|h1 - h0|, 360 - |h1 - h0|)}{180.0} \tag{4}$$

$$s = |s1 - s0| \tag{5}$$

$$v = \frac{|v1 - v0|}{255.0} \tag{6}$$

$$\text{distance} = \sqrt{h^2 + s^2 + v^2} \tag{7}$$

where H (hue) ranges from 0 to 360°, and the distance between the two colors can be calculated from 0 to 1 in Equation (4). S (saturation) is in the 0–1 range in Equation (5). V (value) ranges from 0 to 255 in Equation (6), and when divided by 255, it has a value of 0 to 1. H1, S1, and V1 are the HSV values of the reference ROI, and H0, S0, and V0 are the HSV values of the randomly sampled ROI. If the distance value of color similarity is close to 0, compared to the threshold, it means that the HSV color value of the reference ROI and the HSV value of the randomly searched ROI are the most similar.

If the threshold value of the HSV color similarity is less than or equal to the distance value, the HSV upper and lower limit values of the randomly sampled ROI are added to the temporary ROI buffer list and stored. Otherwise, the similarity of the HSV values is compared by executing the maximum number of randomly sampled ROIs.

4.2.3. HSV Color-Based ROI Merge Operation

Algorithm 3 shows the algorithm of the ROI merge operation. The ROI merge operation retrieves the temporary ROI list for random sampling of ROI values with HSV similar to the reference ROI. $ROI_{TempList\_i}$ represents the initial value of the index of the temporary array list that registers random sampling ROIs with high HSV color similarity, compared to the reference ROI. $ROI_{TempList\_max}$ denotes the maximum index length of the temporary array. The ROI merge operation performs mask operation (OR bit operation) to merge the searched random sampling ROI and reference ROI data.

The ROI merge operation is used to detect the shape (color, position) of the object of the product more accurately from the interference of light, illumination, etc., based on the HSV color base in the real environment. The final ROI merge operation is followed by the move to the automated object measurement.

Algorithm 3. ROI Merge Operation

1: Start ROI Merge Operation
2: Set $ROI_{TempList\_i}$, Retrieve $ROI_{TempList\_max}$
3: Set $ROI_{TempList\_i}$ = 0
4: **For** i = 0: $ROI_{TempList\_max}$
5: Retrieve HSV Value, Upper and Lower threshold for $ROI_{TempList\_i}$
6: $ROI_{TempList\_i}$ → Image Masking According to Upper and Lower threshold
7: Bit Operation ROI_{ref_region} and $ROI_{TempList\_i}$
8: Merge operation of ROI_{ref} and $ROI_{TempList\_i}$
9: **IF**($ROI_{TempList\_i}$ > $ROI_{TempList\_max}$) then
10: Go to Automated Object Measurement through Equations (8) and (9)
11: **Else**
12: $ROI_{TempList\_i}$ = $ROI_{TempList\_i}$ +1
13: Continue:
14: **End**

4.3. HSV Color-Space-Based Object Measurement Algorithm

The object measurement algorithm is calculated and used as the ratio of the "pixels per metric" ratio of the reference object [7]. The object width describes the value measured in the pixel as in Equation (8). We calculated the size of the bounding box (dHeight, dWidth) based on the camera image of the HSV model [7].

The size of the real object (width, height) is calculated by dividing dHeight and dWidth by the pixels per metric, as in Equation (9).

$$\text{PixelsPerMetric} = \text{object width (measured in pixels)}/\text{reference object width} \quad (8)$$

$$\text{Height} = \text{dH}/\text{pixelsPerMetric}, \text{Width} = \text{dW}/\text{pixelsPerMetric} \quad (9)$$

4.4. Automated Recipe Generation of Robot Grasping

The robot handler needs the loading position of the target object, the unloading position to be transferred to, the trajectory of the robot manipulator, the control information of grasping, etc. for the product transfer work. The robot control information is included in the recipe and executes grasping or manipulation. In ISA-88.01 [47], the robot control information was formatted as "Recipe." The user should modify the recipe information whenever the shape of the product is changed. It takes considerable time and effort for the user to teach the operator directly to modify the recipe for control by moving the robot manually. Figure 6 shows the recipe type for robot grasping in the FaaS platform. ISA-88.01 [47] deals with the entire product cycle and is divided into four recipe types. This study deals with a specific facility (robot handler), which is executed by dividing it into two steps: master recipe and control recipe. The motion recipe information for robot grasping consists of Product.json and Move.xml files, which are the master and control recipes, respectively. The master recipe includes basic feature information and process information of the target product, and the control recipe includes the trajectory information of the robot grasping. The master recipe created through object localization and measurement is delivered to the robot handler from the FaaS Manager. It includes the product ID (partID), source loading equipment (fromFacility), and destination unloading equipment (toFacility) information. The control recipe is managed within the robot handler facility; the two-recipe information is combined to control the robot handler [7].

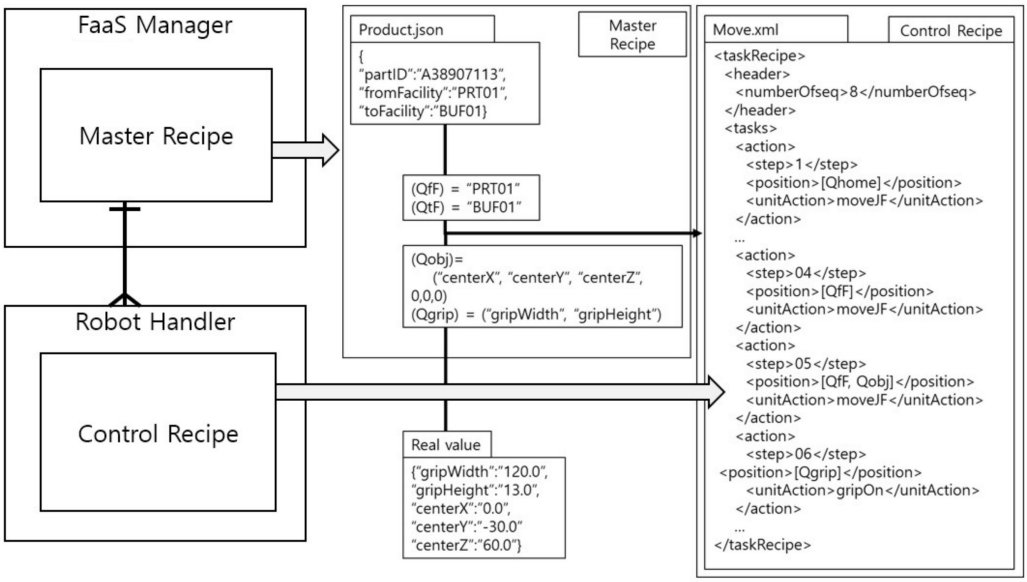


Figure 6. Recipe type for robot grasping in FaaS platform.

The FaaS manager transmits the master recipe to the robot handler, and the IoT-based robot handler performs the grasping operation to transfer the product through the master and control recipes. (QfF) and (QtF) convert source loading equipment and destination unloading equipment in the master recipe to equipment parameters. (Qobj) of the master recipe is calculated by (Qobj) as the center point of the object. (Qgrip) of the master recipe reflects the product's horizontal and vertical size information extracted as HSV color-space-based automatic object measurement information. <header> of the control recipe indicates the total number of steps <numberOfseq>, and <task> denotes the task information for each step. <step> represents the work progress step, <position> represents the target position of the trajectory that the robot needs to move, and <unitAction> represents the robotic unit performing tasks such as movement and grasping.

5. Experiment and Result Analysis

The experiment for evaluation was based on OpenCV [48], and the proposed algorithm was tested and verified in an environment with a lot of ambient noise, light reflection, and lighting interference of the product in the additive manufacturing facility.

IoU [49] refers to the value obtained by dividing the area of the intersection of two regions by the value of the sum region and is used as one of the indicators to evaluate the accuracy of the predicted bounding box in object localization. Generally, for two finite sample sets A and B, their IoU is defined as the intersection (A∩B) divided by the union (A∪B) of A and B [49].

$$\mathrm{IoU}(A, B) = \frac{A \cap B}{A \cup B} = \frac{A \cap B}{|A| + |B| - A \cap B} \tag{10}$$

Comparison of Algorithms

The algorithm was evaluated by testing three products produced on the FaaS platform with different colors. Figure 7 shows the actual appearance of the product and the appearance of the product in the additive manufacturing facility (3D printer). It can be observed that the color of the product is deformed by shadows, reflections, etc., owing to the blue LED lighting in the additive manufacturing facility.

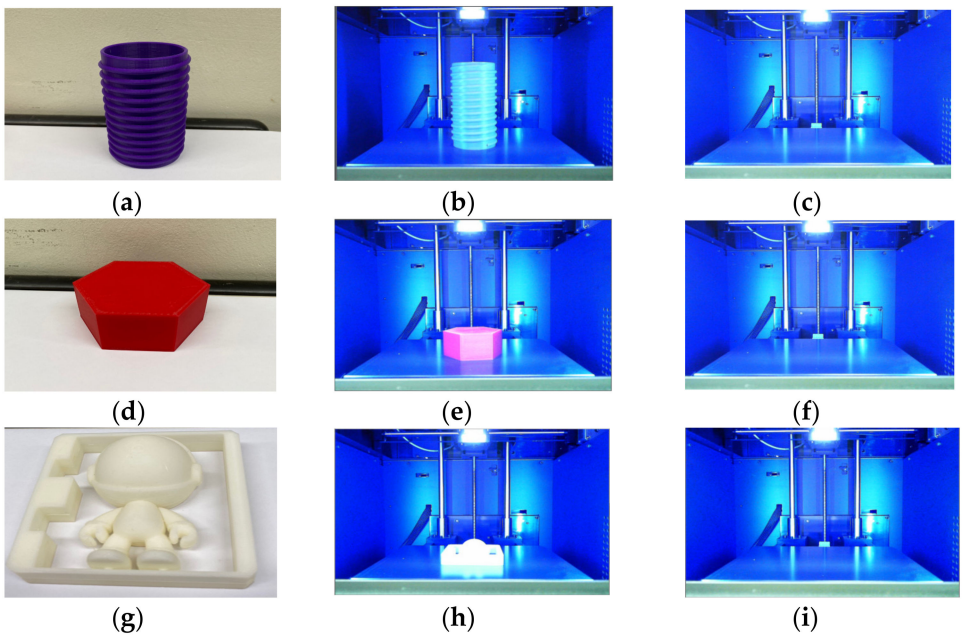


Figure 7. Test products and test environments after production is completed in 3D printers: (**a**,**d**,**g**) original Test 1,2,3 products; (**b**,**e**,**h**) foreground of original Test 1,2,3; (**c**,**f**,**i**) background of original Test 1,2,3.

Figures 8–10 show the results of applying the background subtraction by the Otsu and GMM (Gaussian Mixture Model) algorithms and applying GrabCut and HOG in test products 1, 2, and 3. The green line represents the ground-truth bounding boxes, and the purple line represents the predicted bounding boxes.

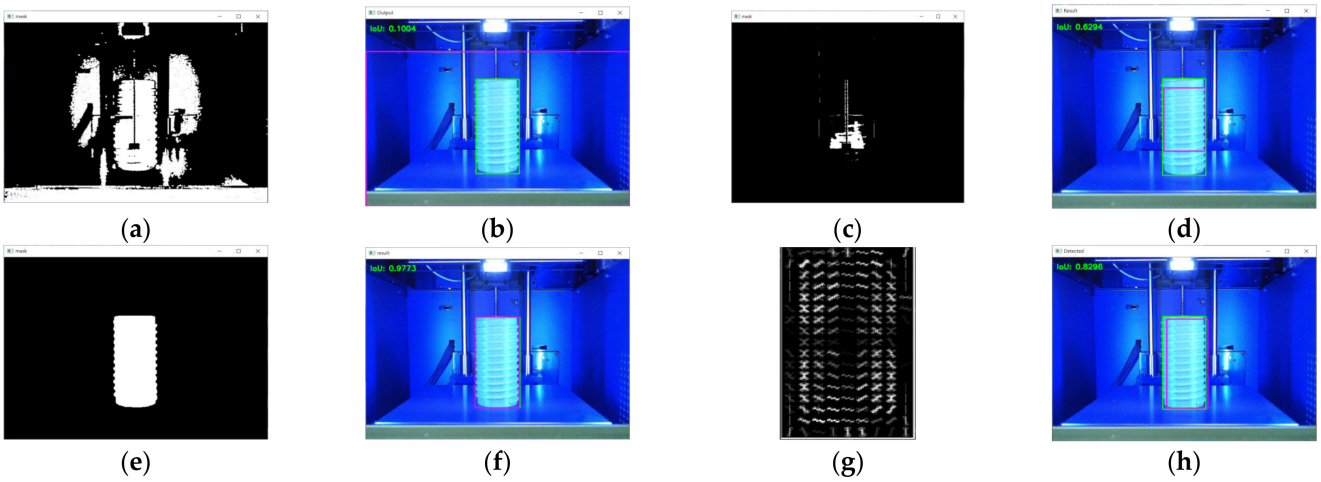


Figure 8. Test 1 product result of background subtraction by the Otsu and GMM algorithms: (**a**) foreground mask of the background subtraction by the Otsu algorithm; (**b**) test result of the background subtraction by the Otsu algorithm; (**c**) foreground mask of the background subtraction by the GMM algorithm; (**d**) test result of the background subtraction by the GMM algorithm; (**e**) foreground mask of the GrabCut algorithm; (**f**) test result of the GrabCut algorithm; (**g**) gradient vector of the HOG algorithm; (**h**) test result of the HOG algorithm.

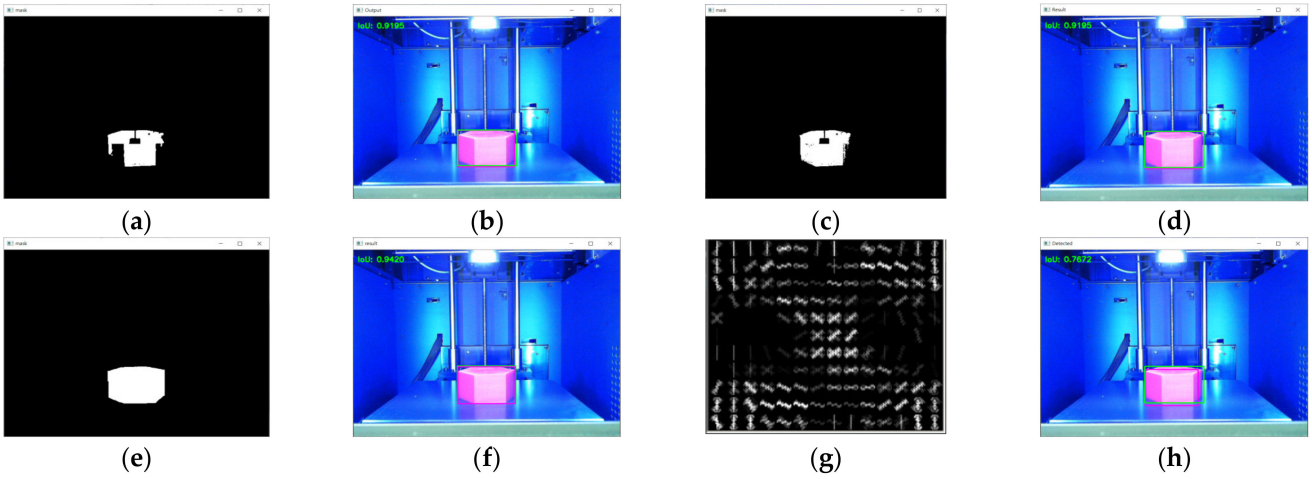


Figure 9. Test 2 product result of background subtraction by the Otsu and GMM algorithms: (**a**) foreground mask of the background subtraction by the Otsu algorithm; (**b**) test result of the background subtraction by the Otsu algorithm; (**c**) foreground mask of the background subtraction by the GMM algorithm; (**d**) test result of the background subtraction by the GMM algorithm; (**e**) foreground mask of the GrabCut algorithm; (**f**) test result of the GrabCut algorithm; (**g**) gradient vector of the HOG algorithm; (**h**) test result of the HOG algorithm.

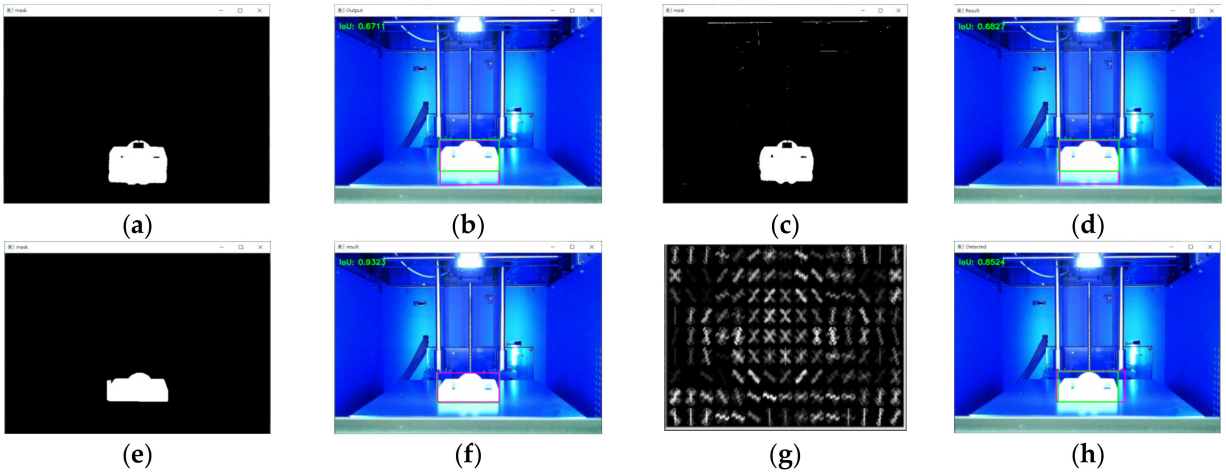


Figure 10. Test 3 product result of background subtraction by the Otsu and GMM algorithms: (**a**) foreground mask of the background subtraction by the Otsu algorithm; (**b**) test result of the background subtraction by the Otsu algorithm; (**c**) foreground mask of the background subtraction by the GMM algorithm; (**d**) test result of the background subtraction by the GMM algorithm; (**e**) foreground mask of the GrabCut algorithm; (**f**) test result of the GrabCut algorithm; (**g**) gradient vector of the HOG algorithm; (**h**) test result of the HOG algorithm.

The background subtraction by the Otsu algorithm in Figure 8 does not detect the product properly because of the long height of product 1 and severe noise owing to the influence of the top illumination. Even background subtraction by the GMM algorithm is not robust to changes in lighting, and only a part of the product background is detected.

The GrabCut algorithm exhibits high performance because the user directly sets the product's bounding box in the foreground. However, GrabCut extracts the foreground step by step through user interaction. The HOG algorithm is insensitive to changes in light and illumination owing to the characteristic that distinguishes the amount and direction of the edge; thus, its IoU is higher than the background subtraction by the Otsu and GMM algorithms, but it has difficulty in accurate object localization.

Test 2 in Figure 9 is a case where the background color and product color are clearly different. In the test product, the color of the product changes owing to the lighting in the 3D printer, but the colors are easily distinguished and separated, so the Otsu and GMM algorithms and IoU show good performance.

Figure 10 shows that the background subtraction by the Otsu and GMM algorithms recognizes the reflected image of the Test 3 product, produced in the additive manufacturing facility (3D printer) as a product object. Although the intersection over union (IoU) performance is not high, GraphCut set by humans shows the same excellent performance as Test 1 and 2 products.

Figure 11 shows the performance evaluation results of the proposed algorithm for the Test 1 product. The maximum ROI is displayed as a cyan line. Random sampling ROI sets to search 70 or 150 ROI and are indicated by a red rectangle within the maximum ROI of the 3D printer, and the reference ROI is represented by a yellow line. Figure 11b shows the mask image of the product that is merged by selecting only ROIs with high similarity to the HSV of the reference region among the HSV values of the random sampling ROI. The proposed algorithm shows good performance in IoU by extracting object localization by reflecting the effect of lighting in test product 1.

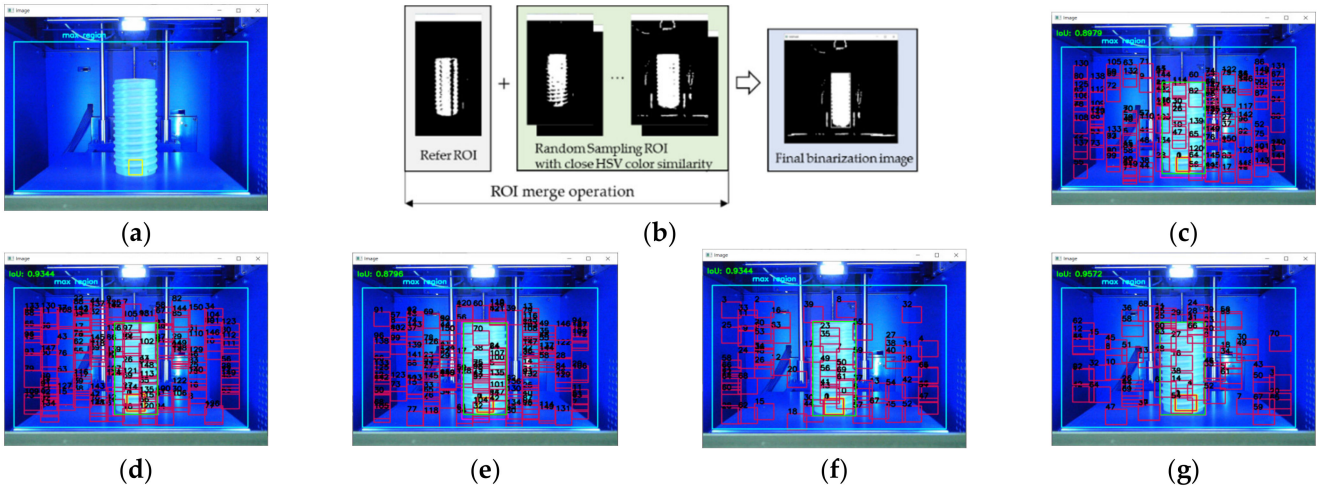


Figure 11. Test 1 product result of the proposed algorithm: (**a**) reference ROI for auto HSV color picking; (**b**) merge operation through random sample ROIs with close HSV Color similarity; (**c**) Test 1 product result (random sampling ROI: 150); (**d**) another Test 1 product result (random sampling ROI: 150); (**e**) another Test 1 product result (random sampling ROI: 150); (**f**) another Test 1 product result (random sampling ROI: 70); (**g**) another Test 1 product result (random sampling ROI: 70).

Figure 12 shows the performance evaluation results of the proposed algorithm for the Test 2 product. The proposed algorithm extracts the HSV values of the reference ROI and of the random ROI and compares their similarity. Then, the merge operation is performed by selecting only the HSV values of the ROIs with a close distance. As a result of the HSV color-based similarity analysis, the similarity of ROI in the Test 1 product is high when the distance value is small (approximately 30 or less).

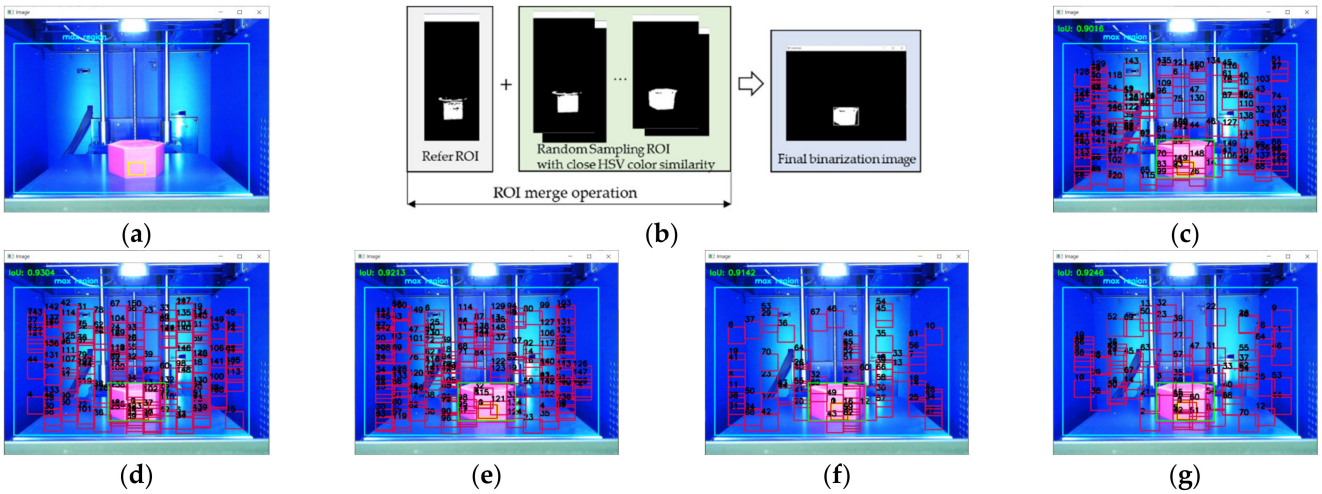


Figure 12. Test 2 product result of proposed algorithm: (**a**) reference ROI for auto color picking; (**b**) merge operation through random sample ROIs with close HSV color similarity; (**c**) Test 2 product result (random sampling ROI: 150); (**d**) another Test 2 product result (random sampling ROI: 150); (**e**) another Test 2 product result (random sampling ROI: 150); (**f**) another Test 2 product result (random sampling ROI:70); (**g**) another Test 2 product result (random sampling ROI:70).

Figure 13 shows the performance evaluation results of the proposed algorithm for the Test 3 products. Depending on the position of the random sampling ROI, as shown in Figure 13c, some of the reflected shadows of the product's lighting were recognized, resulting in a decrease in IoU performance. As the HSV value of the shadow ROI of the product was measured similarly to that of the reference ROI of the product, it was difficult to clearly distinguish the boundary between the background and the foreground.

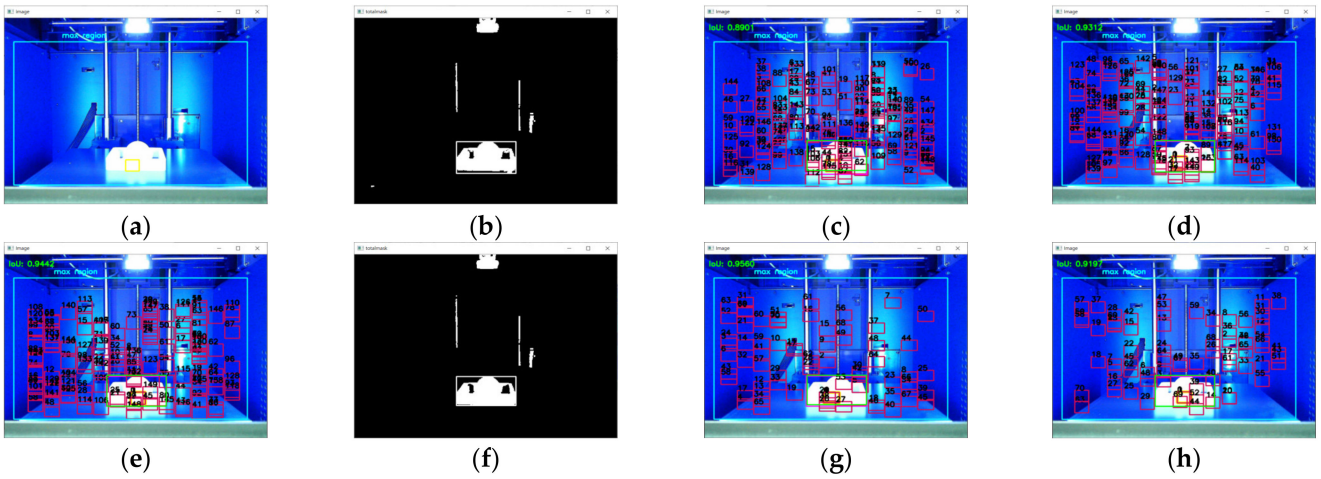


Figure 13. Test 3 product result of proposed algorithm: (**a**) reference ROI for auto color picking; (**b**) final mask binarization image after merge operation (random sampling ROI: 150); (**c**) Test 3 product result (random sampling ROI: 150); (**d**) another Test 3 product result (random sampling ROI: 150); (**e**) another Test 3 product result (random sampling ROI: 150); (**f**) final mask binarization image after merge operation (random sampling ROI: 70) of another Test 3 product in Figure 12g; (**g**) another Test 3 product result (random sampling ROI:70); (**h**) another Test 3 product result (random sampling ROI:70).

Figure 14 shows the performance evaluation results when the size of the reference ROI is large and the reference ROI contains a different color (e.g., blue, similar to the bottom surface). This shows that the object localization result is poor, including the shadow of the bottom surface of the product. The proposed algorithm randomly samples an ROI within the background space from a single HSV color range and compares the HSV color similarity with a reference ROI. In addition, HSV color-based object localization selects only ROIs from random sampling that contain a color similar to the HSV color of the reference ROI as the candidate HSV colors. The Test 3 product consists of a product and a supporting external frame with an empty space between the two, where the color of the bottom surface (blue) of the 3D printer is expressed. Therefore, in auto HSV color picking, if the HSV of the reference ROI is mixed with multiple colors, it is difficult to clearly distinguish the product boundary, as shown in Figure 14.

Figure 14. Test 3 product result of proposed algorithm: (**a**) reference ROI for auto color picking; (**b**) final mask binarization image after merge operation (random sampling ROI: 70); (**c**) Test 3 product result (random sampling ROI: 70).

Table 1 summarizes the test comparison results of the algorithms used for object localization. The GrabCut algorithm shows the highest performance of the overall algorithms. The GrabCut algorithm exhibits good results because the rectangular window set by the user is recognized as the foreground, and anything other than the rectangle is recognized as the background; however, it has the drawback as the user must intervene directly and set the bounding box every time the product is changed. Background subtraction using Otsu and GMM algorithms showed high performance when not affected by light and lighting conditions but did not otherwise find the position of the object accurately. As shown in

the results Figure 8a–d, when the background includes a significant noise, its substation algorithm cannot identify object localization accurately. Although the performance of the HOG algorithm is not higher than that of the GrabCut algorithm, it shows stable results that are somewhat robust to changes in lighting. The values without and with parentheses in Table 1 show the IoU performance results when the number of random sampling ROIs is 70 and 150, respectively.

Table 1. Summary of test comparison results.

Object	BS [1] Using Otsu	BS [1] Using GMM	GrabCut	HOG	Proposed Algorithm
	IoU	IoU	IoU	IoU	IoU
Test Product 1	0.1004	0.6294	0.9733	0.8296	0.9039 (0.9458)
Test Product 2	0.9195	0.9195	0.9420	0.7672	0.9178 (0.9194)
Test Product 3	0.6711	0.6827	0.9323	0.8524	0.9218 (0.9378)

[1] Background subtraction (BS).

The performance of the proposed algorithm is evaluated to be good because the ROI merge operation is executed by comparing the HSV color similarity of the reference ROI through an ROI random sampling that can cover the part affected by lighting. The proposed object localization algorithm shows good performance results based on the HSV color space by reflecting the color change of the original product due to light interference, shadows, or light in a single HSV color space.

6. Conclusions

Smart factories are evolving to become intelligent by combining major ICT technologies such as IoT, robots, and AI to connect all processes of production, distribution, service, and consumption. Smart factories for manufacturing intelligence beyond production automation still have many problems to be solved on the production floor to improve productivity, reduce costs, and optimize processes. To respond quickly to flexible production environments and rapidly changing consumer demands, robots, the core equipment of smart factories, require automation and intelligence that can operate on their own without human intervention. The robot's machine vision technology is one of the core technologies of an important smart factory that can automate robots.

This study proposes a simple and robust HSV color-space-based object localization algorithm for external environmental changes (illumination interference and shadow, reflections, lighting conditions, etc.) in a highly variable manufacturing environment.

The proposed algorithm does not require the user to specify the HSV color in advance for object detection and localization, even if the product is changed. The HSV color of the reference ROI is automatically extracted from the vision of the robot connected to the IoT device. In addition, through random sampling, the ROI merge operation is performed on only the ROI with high HSV similarity to the reference ROI, so that the object localization of the product can be automatically identified within the 3D printer. This means that object localization can be extracted quickly and automatically without user intervention and prior knowledge (e.g., learning and pre-setting).

Further work is needed to recognize the position of objects quickly and easily for a variety of products that contain multiple colors, not just a single color. In addition, when HSV color values are distributed differently in the ROI, it is necessary to develop an algorithm that can accurately measure and analyze the distance of HSV similarity by spatially subdividing the ROI. It is necessary to consider the optimization of the HSV color similarity distance and random sampling ROI size. Moreover, interesting research [50–52] such as artistic robot painting and real-time grasping detection for the intelligence and automation of robots have been reported. We expect that the time and effort required for robot setting and teaching work, which are approached by human manual work, can be reduced.

Author Contributions: Conceptualization, H.-C.K. and Y.-K.K.; methodology, H.-C.K. and J.-Y.S.; investigation, H.-N.H., H.-C.B., and H.-C.K.; data curation, H.-N.H., H.-C.B., and M.-G.K.; writing—original draft preparation, H.-C.K.; writing—review and editing, Y.-K.K.; project administration, Y.-K.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the project for manufacturing data commonly used platform R&D, funded by the Korea Ministry of SMEs and Startups in 2021 (S3125303. Development of Ultra Lightweight Platform for Production Process and Safety Management of Small and Medium Manufacturing Companies). Additionally, this work was supported by Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean government (21ZR1100, A Study of Hyper-Connected Thinking Internet Technology by autonomous connecting, controlling and evolving ways).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Lasi, H.; Fettke, P.; Kemper, H.G.; Feld, T.; Hoffmann, M. Industry 4.0. *Bus. Inf. Syst. Eng.* **2014**, *6*, 239–242. [CrossRef]
2. Chen, B.; Wan, B.; Shu, L.; Li, P.; Mukherjee, M.; Yin, B. "Smart Factory of Industry 4.0: Key Technologies, Application Case, and Challenges.". *IEEE Access* **2018**, *6*, 6505–6519. [CrossRef]
3. Park, K.T.; Son, Y.H.; Ko, S.W.; Noh, S.D. Digital Twin and Reinforcement Learning-Based Resilient Production Control for Micro Smart Factory. *Appl. Sci.* **2021**, *11*, 2977. [CrossRef]
4. Son, J.; Kang, H.C.; Bae, H.C.; Lee, E.S.; Han, H.Y.; Kim, H. IoT-based open manufacturing service platform for mass personalization. *J. Korean Inst. Commun. Sci.* **2015**, *33*, 42–47.
5. Wang, W.; Chen, Y.; Li, R.; Jia, Y. Learning and Comfort in Human–Robot Interaction: A Review. *Appl. Sci.* **2019**, *9*, 5152. [CrossRef]
6. Okarma, K. Applications of Computer Vision in Automation and Robotics. *Appl. Sci.* **2020**, *10*, 6783. [CrossRef]
7. Kang, H.C.; Han, H.Y.; Bae, H.C.; Lee, E.S.; Kim, M.G.; Son, J.; Kim, H.; Kim, Y.K. HSV Color Space Based Robot Grasping for Personalized Manufacturing Services. In Proceedings of the International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea, 16–18 October 2019; IEEE: Jeju, Korea, 2019; pp. 1010–1012.
8. Poppe, C.; Martens, G.; de Bruyne, S.; Lambert, P.; van de Walle, R. Robust spatio-temporal multimodal background subtraction for video surveillance. *Opt. Eng.* **2008**, *47*, 107203. [CrossRef]
9. Chiu, S.-Y.; Chiu, C.-C.; Xu, S.S.-D. A Background Subtraction Algorithm in Complex Environments Based on Category Entropy Analysis. *Appl. Sci.* **2018**, *8*, 885. [CrossRef]
10. Piccardi, M. Background subtraction techniques: A review. In Proceedings of the 2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583), Hague, The Netherlands, 10–13 October 2004; pp. 3099–3104.
11. Tamersoy, B. *Background Subtraction*; The University of Texas at Austin: Austin, TX, USA, 2009.
12. Roy, P.; Dutta, S.; Dey, N.; Dey, G.; Chakraborty, S.; Ray, R. Adaptive thresholding: A comparative study. In Proceedings of the International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), Kanyakumari, India, 10–11 July 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 1182–1186.
13. Bradley, D.; Roth, G. Adaptive thresholding using the integral image. *J. Graph. Tools* **2007**, *12*, 13–21. [CrossRef]
14. Bouttefroy, P.L.M.; Bouzerdoum, A.; Phung, S.L.; Beghdadi, A. On the analysis of background subtraction techniques using Gaussian mixture models. In Proceedings of the 2010 IEEE International Conference on Acoustics, Speech and Signal Processing, Dallas, TX, USA, 14–19 March 2010; pp. 4042–4045.
15. Stauffer, C.; Grimson, W.E.L. Adaptive background mixture models for real-time tracking. In Proceedings of the 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Fort Collins, CO, USA, 23–25 June 1999; pp. 246–252.
16. Kim, W.; Jung, C. Illumination-invariant background subtraction: Comparative review, models, and prospects. *IEEE Access* **2017**, *5*, 8369–8384. [CrossRef]
17. Jacques, J.C.S.; Jung, C.R.; Musse, S.R. A background subtraction model adapted to illumination changes. In Proceedings of the 2006 International Conference on Image Processing, Atlanta, GA, USA, 8–11 October 2006; pp. 1817–1820.
18. Parks, D.H.; Fels, S.S. Evaluation of Background Subtraction Algorithm with Post-Processing. In Proceedings of the 2008 5th International Conference on Advanced Video & Signal Based Surveillance, Santa Fe, NM, USA, 1–3 September 2008; pp. 192–199.
19. Salvador, E.; Cavallaro, A.; Ebrahimi, T. Cast shadow segmentation using invariant color features. *Comput. Vis. Image Underst.* **2004**, *95*, 238–259. [CrossRef]
20. Horn, B.K.; Schunck, B.G. Determining optical flow. *Artif. Intell.* **1981**, *17*, 185–203. [CrossRef]

21. Ren, X.; Ramanan, D. Histograms of sparse codes for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 3246–3253.
22. Yoon, H.S.; Bae, Y.L.; Yang, Y.K. A Study on Image Retrieval Using Space Information of Color Histogram. In Proceedings of the Korea Information Processing Society Conference, Daejeon, Korea, 13–14 October 2000; pp. 867–870.
23. Tuytelaars, T.; Mikolajczyk, K. *Local Invariant Feature Detectors: A Survey*; Now Publishers Inc.: Boston, MA, USA, 15 June 2008; pp. 177–280.
24. Harris, C.G.; Stephens, M. A combined corner and edge detector. In Proceedings of the Alvey Vision Conference, Manchester, UK, 31 August–2 September 1988; pp. 147–151.
25. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, USA, 20–25 June 2005; pp. 886–893.
26. Suleiman, A.; Sze, V. Energy-efficient HOG-based object detection at 1080HD 60 fps with multi-scale support. In Proceedings of the 2014 IEEE Workshop on Signal Processing Systems (SiPS), Belfast, UK, 20–22 October 2014; pp. 1–6.
27. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
28. Lowe, D.G. Object recognition from local scale-invariant features. In Proceedings of the Seventh IEEE International Conference on Computer Vision(ICCV), Kerkyra, Greece, 20–27 September 1999; pp. 1150–1157.
29. Otsu, N. A threshold selection method from gray-level histograms. *IEEE Trans. Syst. Man Cybern.* **1979**, *9*, 62–66. [CrossRef]
30. Rother, C.; Kolmogorov, V.; Blake, A. GrabCut–Interactive Foreground Extraction using Iterated Graph Cut. *ACM Trans. Graph.* **2004**, *23*, 309–314. [CrossRef]
31. Boykov, Y.; Kolmogorov, V. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell.* **2004**, *26*, 1124–1137. [CrossRef]
32. Yi, F.; Moon, I. Image segmentation: A survey of graph-cut methods. In Proceedings of the IEEE International Conference on Systems and Informatics (ICSAI2012), Yantai, China, 19–20 May 2012; pp. 1936–1941.
33. Albawi, S.; Mohammed, T.A.; Al-Zawi, S. Understanding of a convolutional neural network. In Proceedings of the IEEE International Conference on Engineering and Technology (ICET), Antalya, Turkey, 21–23 August 2017; pp. 1–6.
34. Kim, Y.H.; Park, S.W.; Kim, D.Y. Research on Robust Face Recognition against Lighting Variation using CNN. *J. Korea Inst. Electron. Commun. Sci.* **2017**, *12*, 325–330.
35. Varghese, A.; Gubbi, J.; Ramaswamy, A.; Balamuralidhar, P. ChangeNet: A deep learning architecture for visual change detection. In Proceedings of the European Conference on Computer Vision (ECCV) Workshops, Munich, Germany, 8–14 September 2018.
36. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
37. Girshick, R. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
38. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
39. Zhao, Z.Q.; Zheng, P.; Xu, S.T.; Wu, X. Object detection with deep learning: A review. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 3212–3232. [CrossRef]
40. Koščević, K.; Subašić, M.; Lončarić, S. Deep Learning-Based Illumination Estimation Using Light Source Classification. *IEEE Access* **2020**, *8*, 84239–84247. [CrossRef]
41. Clement, L.; Kelly, J. How to train a cat: Learning canonical appearance transformations for direct visual localization under illumination change. *IEEE Robot. Autom. Lett.* **2018**, *3*, 2447–2454. [CrossRef]
42. Shaik, K.B.; Ganesan, P.; Kalist, V.; Sathish, B.S.; Jenitha, J.M.M. Comparative study of skin color detection and segmentation in HSV and YCbCr color space. *Procedia Comput. Sci.* **2015**, *57*, 41–48. [CrossRef]
43. Ganesan, P.; Rajini, V.; Sathish, B.S.; Shaik, K.B. HSV color space based segmentation of region of interest in satellite images. In Proceedings of the International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), Kanyakumari, India, 10–11 July 2014; pp. 101–105.
44. Herodotou, N.; Plataniotis, K.N.; Venetsanopoulos, A.N. A color segmentation scheme for object-based video coding. In Proceedings of the IEEE Symposium on Advances in Digital Filtering and Signal Processing, Symposium Proceedings, Victoria, BC, Canada, 5–6 June 1998; pp. 25–29.
45. Sural, S.; Qian, G.; Pramanik, S. Segmentation and histogram generation using the HSV color space for image retrieval. In Proceedings of the IEEE International Conference on Image Processing, Rochester, NY, USA, 22–25 September 2002.
46. Li, D. (Ed.) *Encyclopedia of Microfluidics and Nanofluidics*; Springer Science & Business Media: Berlin, Germany, 2008.
47. Instrument Society of America (ISA), ANSI/ISA-88.01-1995. *"Batch Control Part1: Models and Terminology"*; Instrument Society of America (ISA): Durham, NC, USA, 1995; pp. 35–51.
48. Bradski, G.; Kaehler, A. OpenCV. *Dr. Dobb's J. Softw. Tools* **2000**, *3*.
49. Zhou, D.; Fang, J.; Song, X.; Guan, C.; Yin, J.; Dai, Y.; Yang, R. IoU Loss for 2D/3D Object Detection. In Proceedings of the IEEE International Conference on 3D Vision (3DV), IEEE, Quebec, QC, Canada, 16–19 September 2019; pp. 85–94.
50. Scalera, L.; Seriani, S.; Gasparetto, A.; Gallina, P. Non-Photorealistic Rendering Techniques for Artistic Robotic Painting. *Robotics* **2019**, *8*, 10. [CrossRef]

51. Karimov, A.; Kopets, E.; Kolev, G.; Leonov, S.; Scalera, L.; Butusov, D. Image Preprocessing for Artistic Robotic Painting. Inventions. *Appl. Sci.* **2021**, *6*, 19.
52. Zhang, J.; Li, M.; Feng, Y.; Yang, C. Robotic grasp detection based on image processing and random forest. *Multimed. Tools Appl.* **2020**, *79*, 2427–2446. [CrossRef]