LEGO - DEFECT SORTING MECHANISM

LEGO Production Line

Abstract

This document is written for educational purpose of the Industrial and Systems Engingeering of KAIST. It contains a detail explanation of the LEGO Production line with defect sorting mechanism using a convolutional neural network.

Vina Sari Yosephine

vinasari@kaist.ac.kr

Contents

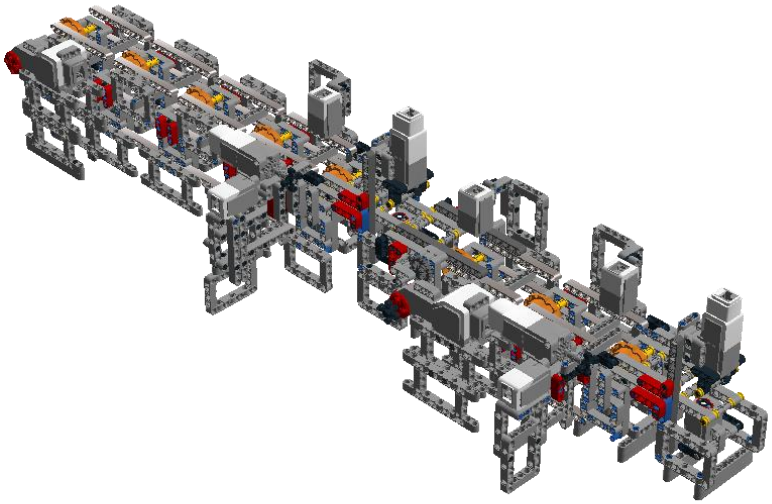
1. Introduction

a. Learning Goal

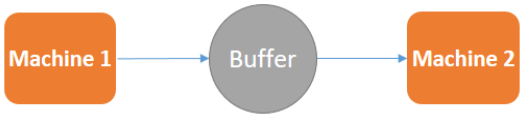
The era of cyber technology has benefited the manufacturing sector. It is therefore important for the engineering education to incorporate the technology while teaching the core engineering knowledge to the students. Industrial engineering students must be equipped with the learning for the future (or rather recent development) manufacturing. Therefore, the goal of this module is twofold: manufacturing system engineering fundamental and the introduction of recent manufacturing technology in the smart Factory.

b. LEGO-Based Production Line[1] Overview

Since 2015, Industrial and Systems Engineering Department of KAIST has been developing a problem-based-learning teaching apparatus focusing on the manufacturing systems engineering. The fundamental apparatus is called the LEGO Production Line (LPL), which consists of two machines and one intermediate buffer in the form of a conveyor belt. Each machine in the LPL subjects to a failure, which creates a variability in the production time and machine's availability. Furthermore, the interaction between machines and the finite intermediate buffer creates a system dynamic that



contributes to the starvation and blockage of an individual machine. These production line configurations reinforce students to learn about how each individual component influences the performance measures of the production line in a realistic way.



According to the Little's Law[2], the relation between performance measures of the production line can be expressed as follows:
*WIP = TH * CT*
WIP = Work in progress, *TH* = Throughput, *CT* = Cycle Time
Learning through LPL, students are asked to define and measure their own performance measures.

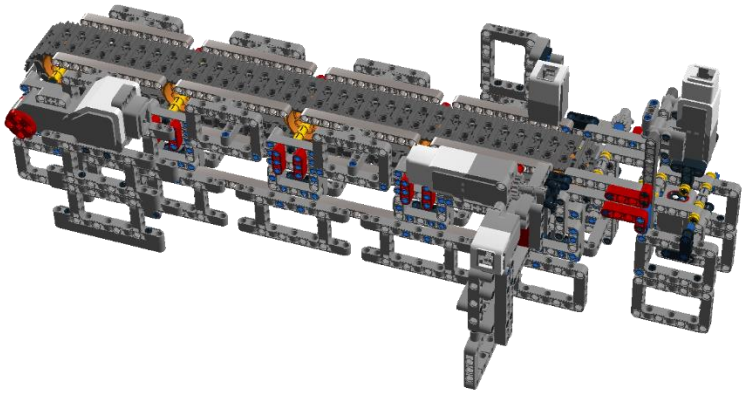
In this module, the functionality of the line is extended by adding a computer-vision quality control to the second machine.

c. Lab Apparatus

The main tools in the project are LEGO EV3, LEGO Technic, a webcam, and a 3D Printing.

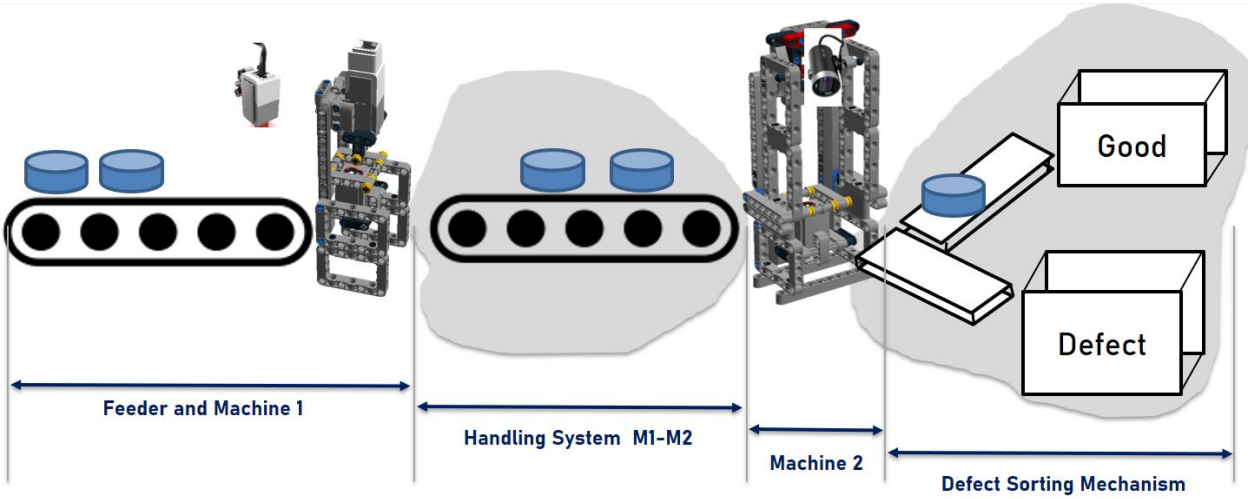
d. The First Machine and the Feeder Conveyor

The first machine and the feeder conveyor have an identical design to the LPL.



2. Defect Sorting Mechanism

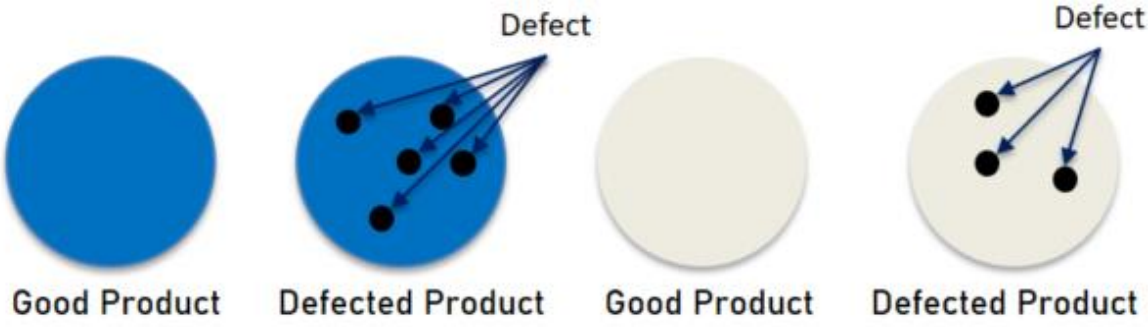
The defect sorting mechanism is to be completed in the second half of the semester, in 7 weeks period. Each group was asked to design the handling system from the first machine to the second machine, and also the defect sorting mechanism, the gray areas in the figure.



The main features of Machine 2 are as follows:

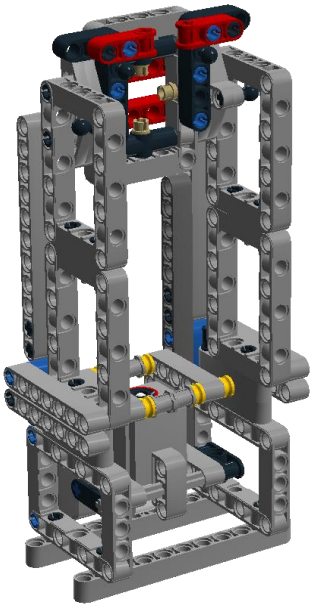
- Machine 2 functions as a quality control that is able to automatically distinguish the good product and the defected product.
- Machine 2 needs to identify the defected product and store it in a different area.
- At least three main components have to be designed and fabricated by a 3D printing technology.

Shown here, the shape of the product is circular chips with two colors, white and blue. A defected product has random dots pattern on its surface.



1. Machine Building

Machine 2 is an adaptation of machine 1 that is controlled by a color sensor and a medium motor. However, since color sensor can only recognize a light intensity, it is not enough to identify a defected product. Therefore, a camera is installed on top of machine two to capture the image of the product, and the motor is installed on the side part of the machine.



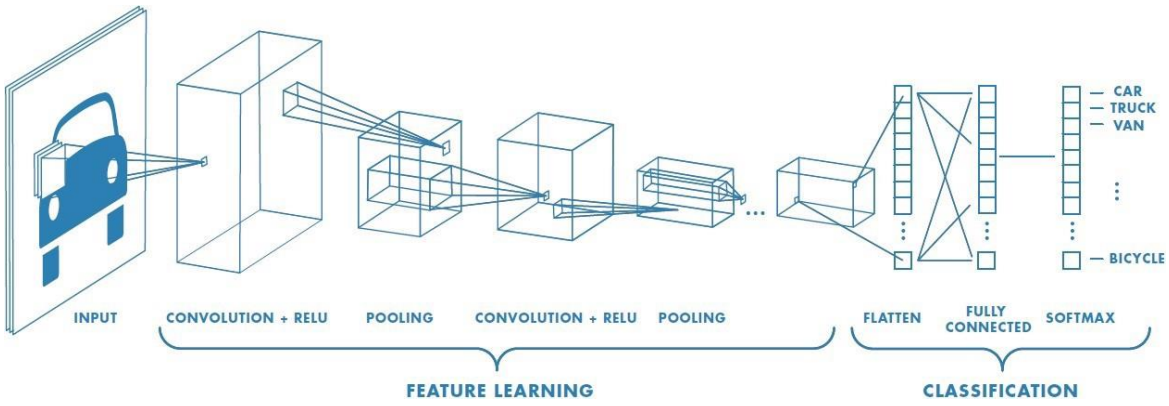
For a detail step-by-step instruction, open **Machine-camera.pdf**

2. Deep Learning – Convolutional Neural Network

The requirement of the production line is to capture an image to be classified as good or defect during the second process. The deep learning – convolutional neural network (CNN) is suitable to do perform the classification.

CNN is a deep learning based algorithm that is able to recognize the pattern and classify the image effectively. It is a class of deep neural network, that is also inspired by human biological process in learning/recognizing an object, in this case the defect/good product.

The algorithm send an image into a network, and it will return the class of the image. In LPL the final class is either good product or defected product. Additional classes can be added to the image when needed.

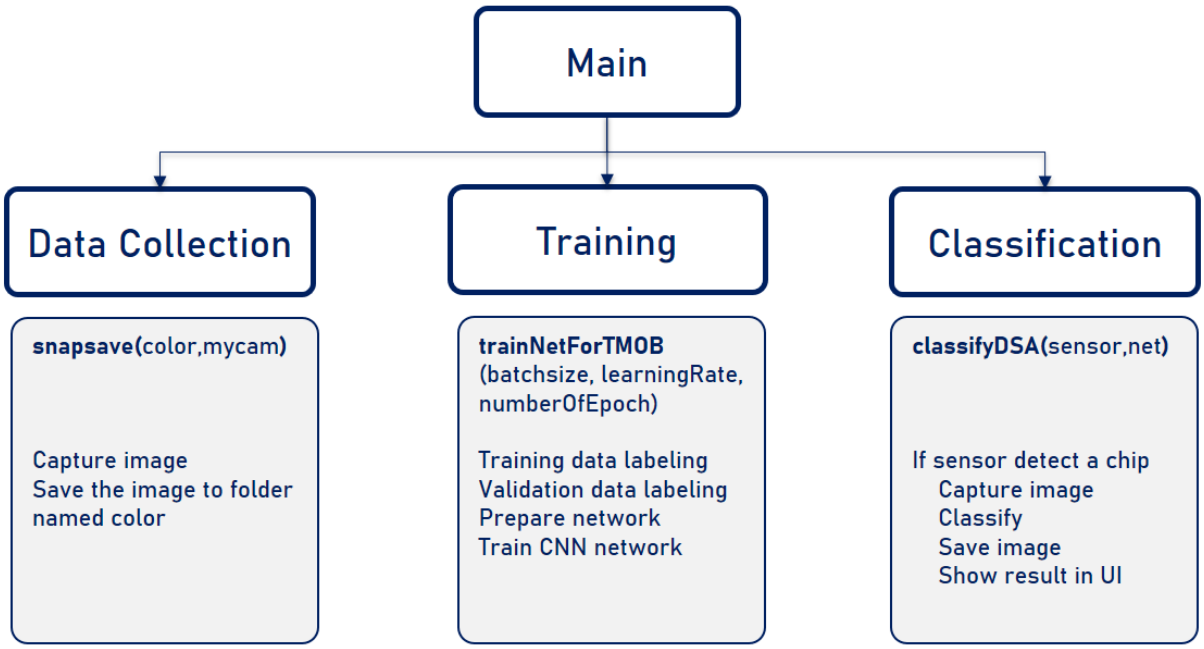


The network consists of multiple layers that detects different features of the image, such as edge, brightness, or complexity of the image. There are a lot of available networks, and in LPL the VGG16 is chosen as the network architecture.

For learning convenience, a MATLAB program has been pre-develop for the use of LPL. Details can be found in the next three sections.

3. The Algorithm Explained

The following section describes the code that controls the LEGO factory. Following the structure of CNN transfer learning, there are three main functions in this system: data collection, training, and classification.



a. Main Program

The main program provides an interface to select the step that a user is going to perform. When the user selects operation mode 1,2,or 3, the screen is directed to the data collection, training, and the classification screen respectively.

b. Data Collection

When mode 1 is selected, another screen will appear with the live visualization of the camera. This will allow the user to insert the chip and select the appropriate button to save the image for data training.

When the button **Capture White Image** is pressed, a function called *snapsave* is called to capture the image (therefore "snap") and save the image to the folder according to the pressed button, in this case the folder with the title WHITE.

c. Training

When mode 2 is selected, a screen will prompt for user to fill in the batch size, learning rate, and number of epochs for the training.

Before the actual training, a network preparation has to be performed first. This is important so we can use the pre-made MATLAB *trainingNetwork* function. This function requires the image, the label, layers, and all the training parameters.

First the function will create three temporary folder consists of the three colors that are going to be trained. The image to be trained will then be labeled as 'WHITE', 'DEFECT', or 'BLUE' in an array called *trainLabelArray* for training data. Same treatment goes to the validation data. The system uses the VGG16 network to train the data.

d. Classification

Classification is only possible when the file *trainedNetwork.mat* presents in the folder. The input image is classified by using the trained network.

The output of the classification is a variable called *labelnum* consists of three categorical value: 1, 2, and 3 which stands for white, blue, and defected.

4. Guideline for the LEGO Smart Factories User Interface

1. Data Collection

The first step of implementing CNN on the machine is by collecting the good and defected images. Each product was placed in the machine and the image is captured by the machine's camera.

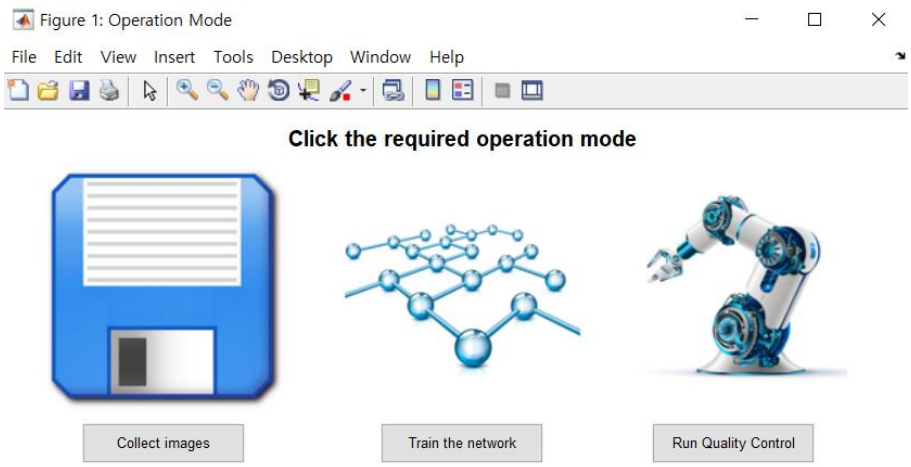
It is important to collect multiple images to achieve high accuracy during classification. Special attention must be taken especially because the position of a chip in the machine may be different, depends on the parameters of the line (conveyor belt speed, machine's distance, etc.).

Another consideration is the light ambience during training. It is important to make sure that the image we use for data training accommodate all the possibility of image to be classified.

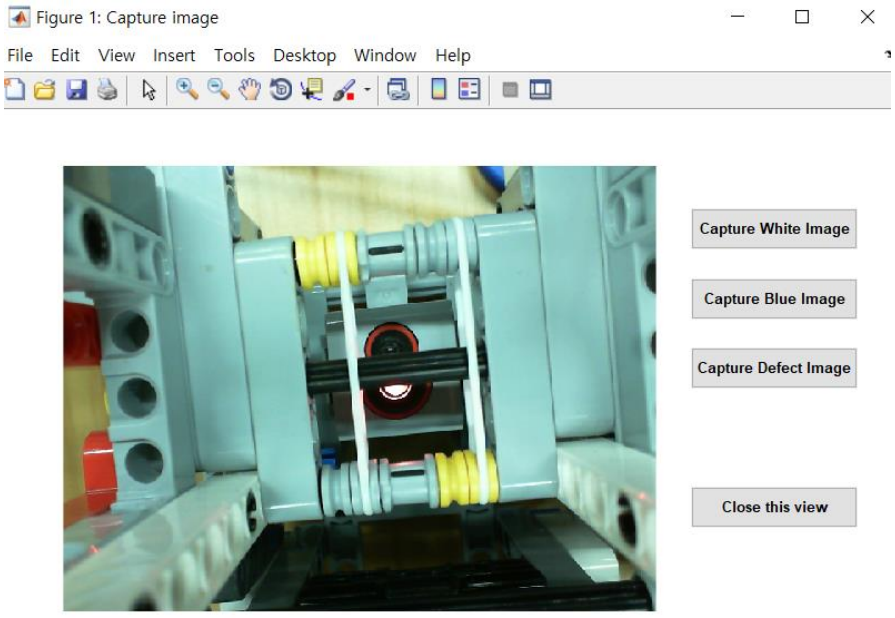
The recommendation is to capture 200-300 images for each category: good and defected.

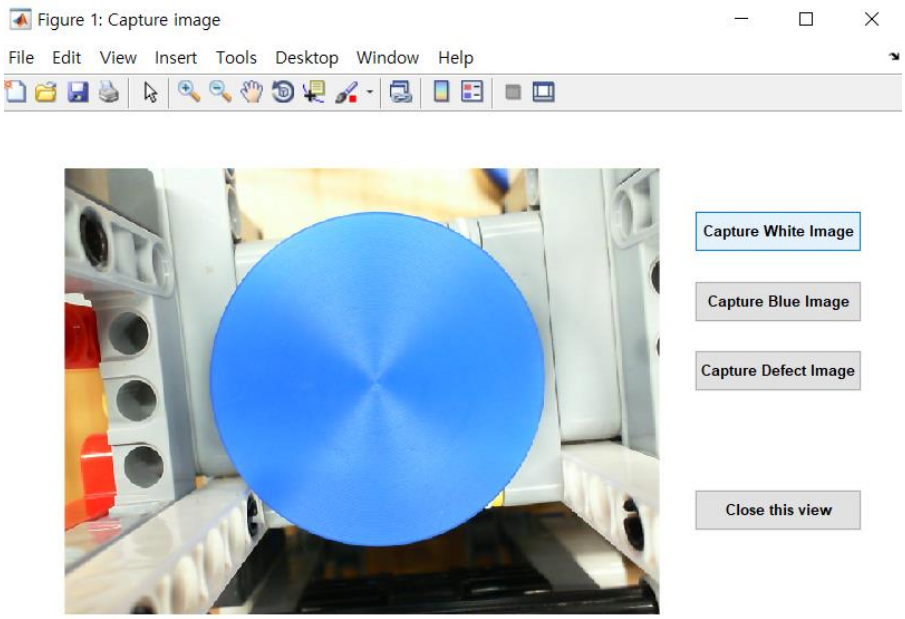
To run the user interface:

1. Unzip the Matlab codes and open in the respected folder.
2. Adjust the path in the snapsave.m and classifyDSA.m according to the folder location.
3. Make sure the camera is connected through USB cable and run the file mainDSA.m, and a window will appear as follows.
4. Click "collect images"



An interface showing the inside part of the machine will appear, with three options for selecting the color of the product. The camera can be tested by placing a random object in the machine.

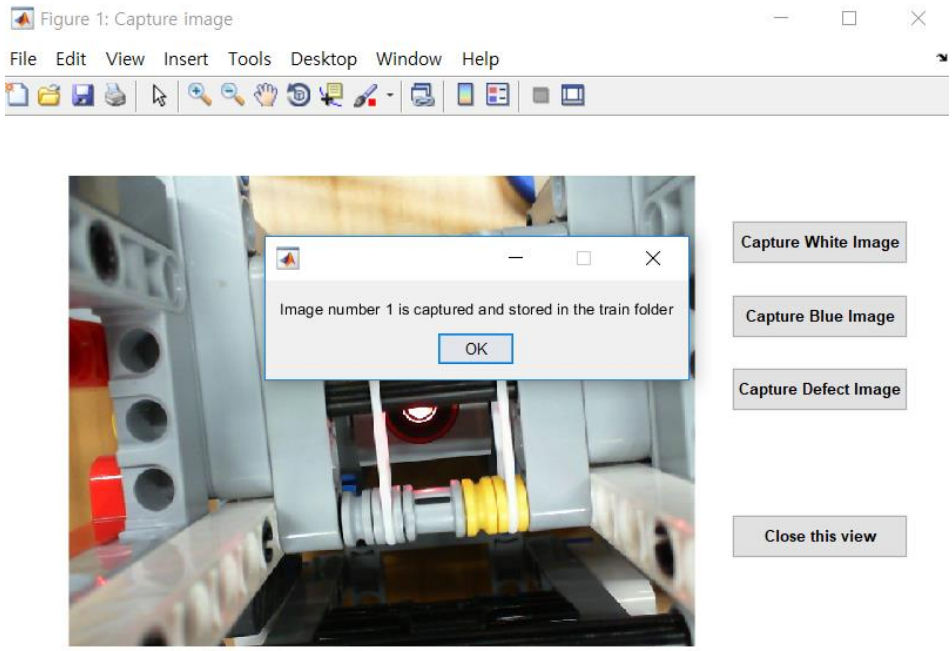




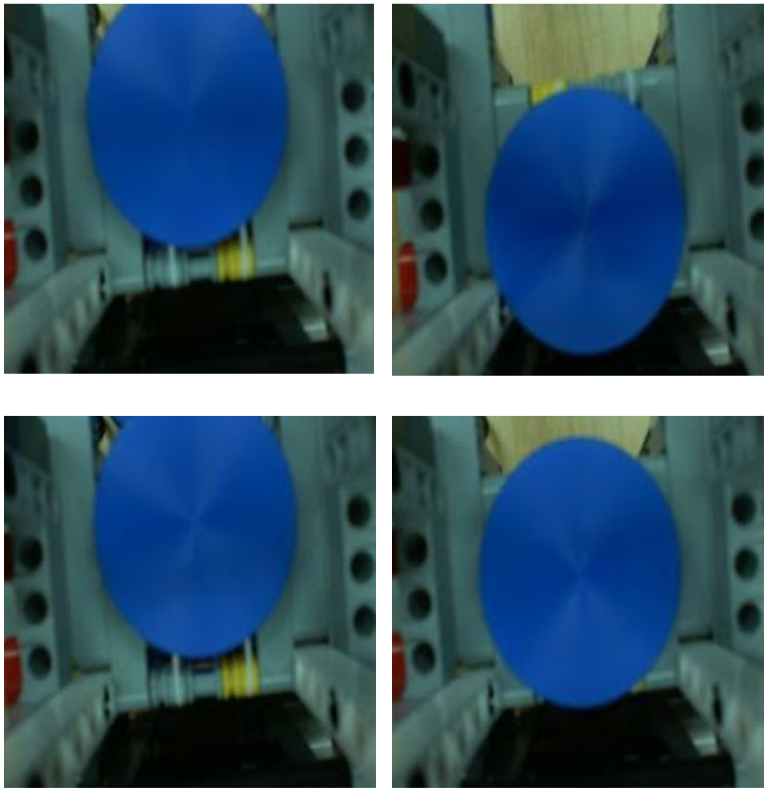
The data collection is performed manually. First, insert the chips to be capture and click the button based on its color. In this example, click the button “Capture Blue Image”.

Upon successful image collection, a message will pop in the window. This message can be closed or ignored to be close later. The program will randomly store the image 80% to training folder and 20% to the validation folder. The same repetition should be performed to another color as well.

To collected images are stored in the same folder where the file mainDSA.m is located.

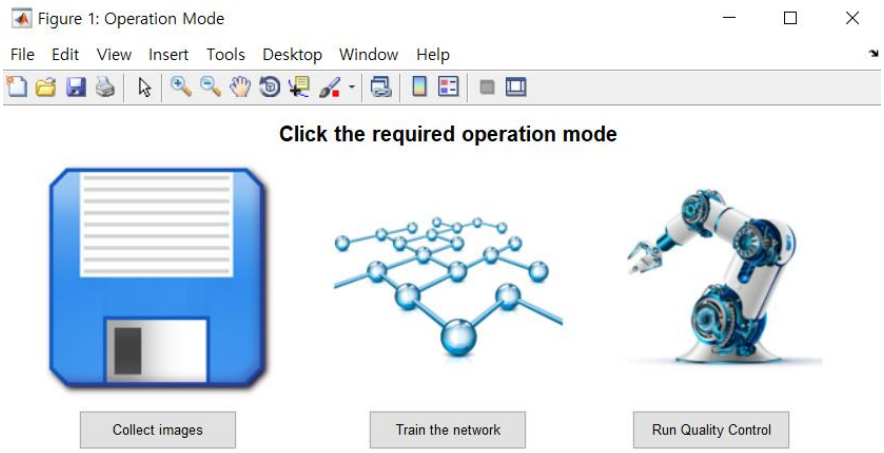


It is important to collect images with multiple position in the picture. This is to accommodate the potential location of the image.



2. CNN Feature Training

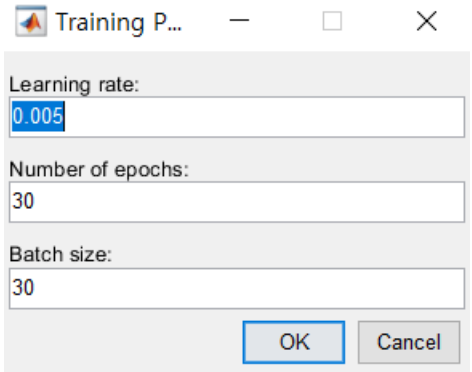
To train the network, open the script mainDSA.m and select the button “Train the network”.



A prompt message will appear to ask for Learning rate, Number of epochs, and Batch size. The following is the definition of each parameter.

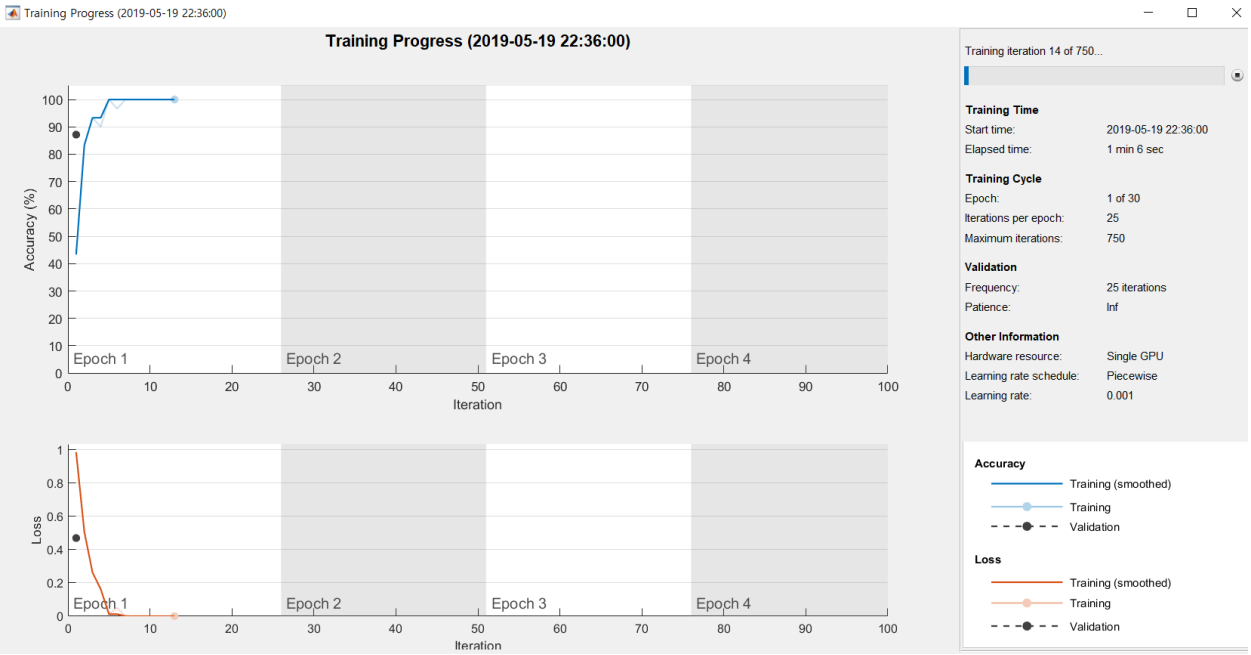
1. Learning rate, which influences the rate of the algorithm in training the neural network in finding the best weights.
2. Number of epochs is the number of the data passed to the algorithm.
3. Batch size

There is no specific rule to determine these parameters, except for a trial & error experiment.



After the parameters are placed. A new window that shows the training progress will appear. In this case, we want to aim for accuracy 100% and Loss of 0.

1. Training accuracy: the accuracy between the training and validation data.
2. Loss: the difference between the prediction classification and the true value of the training data.

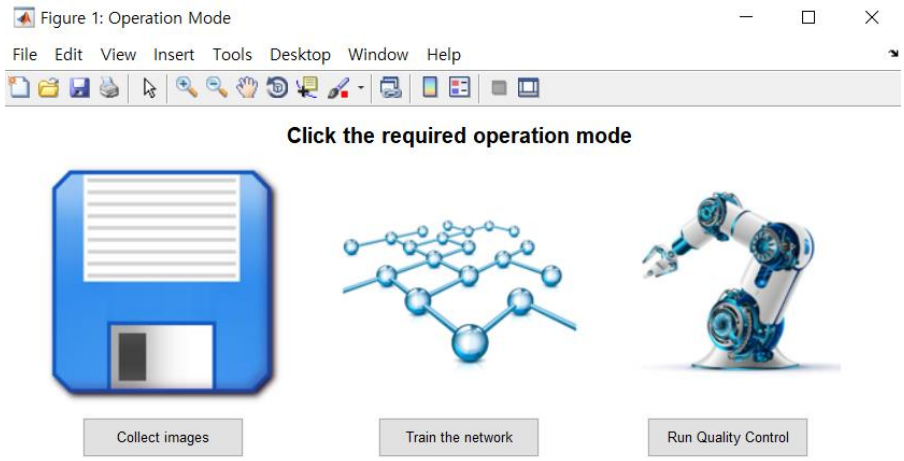


When the training is finished, a new file named trainedNetwork.mat will appear in the same folder. This file will be used in the image classification

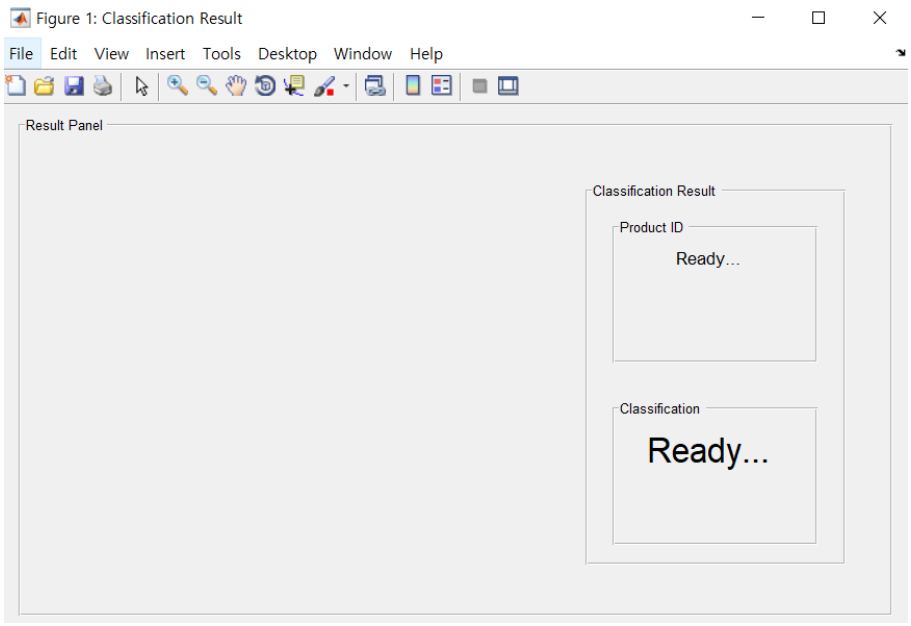
5. Classification

This section describes the image classification procedure after data collection and training. The program can be inserted to the main program for future use.

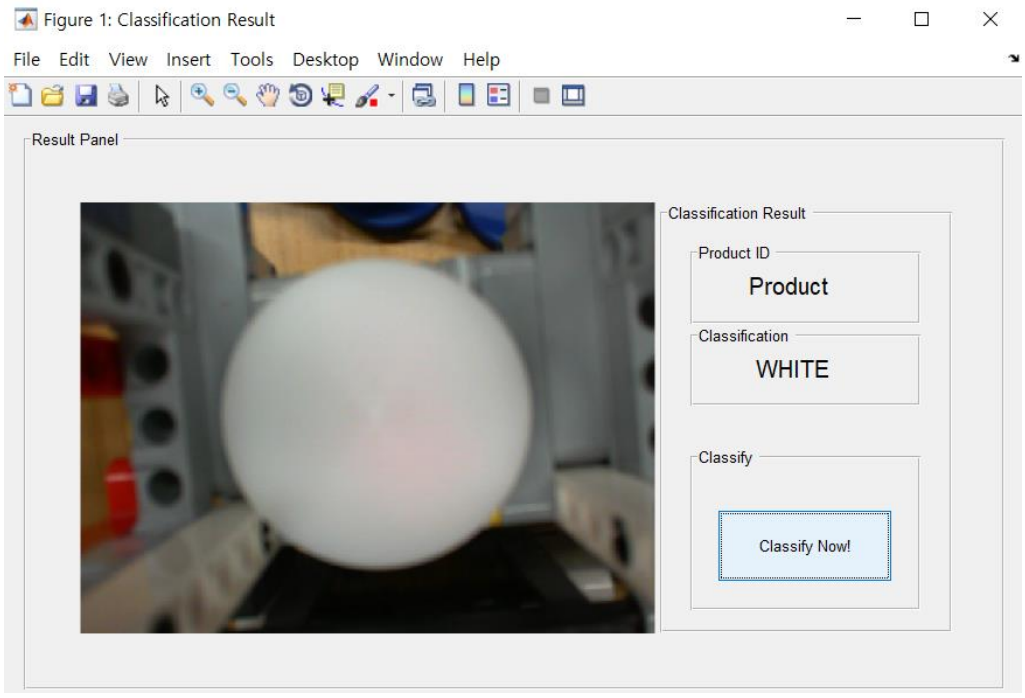
To start classification, run the script mainDSA.m and click “Run Quality Control”.

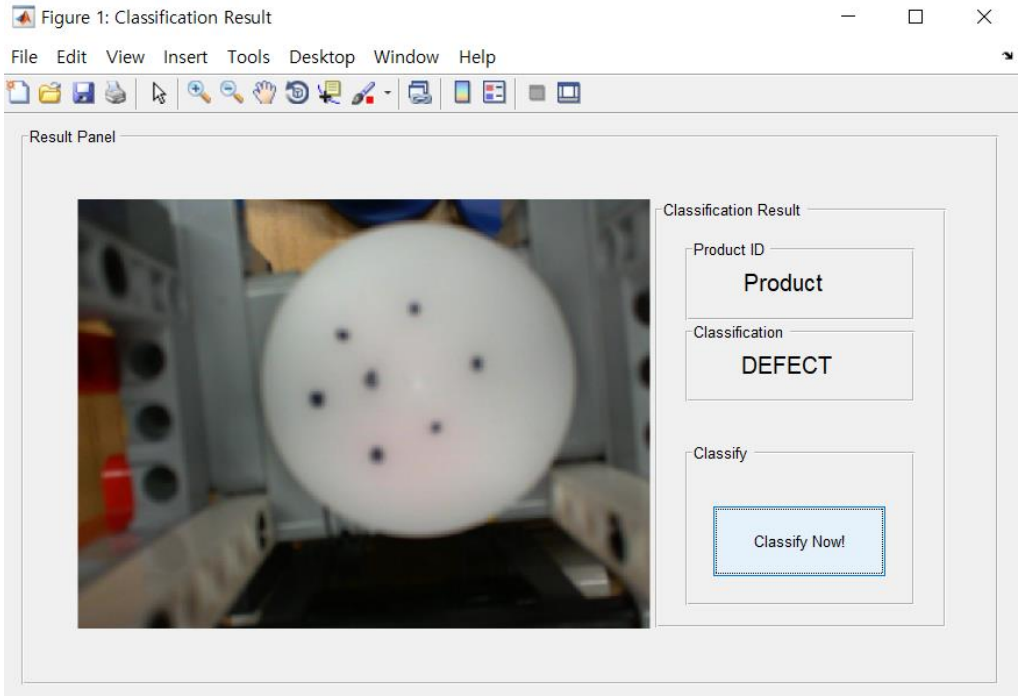


The default classification window is as follows. A result panel will show the captured image. The program keeps the product ID as a record. Both image and the classification result will be stored in the DATA folder in the main folder.



Once the product enters the machine, click “Classify Now!” button, and the program will classify the result.





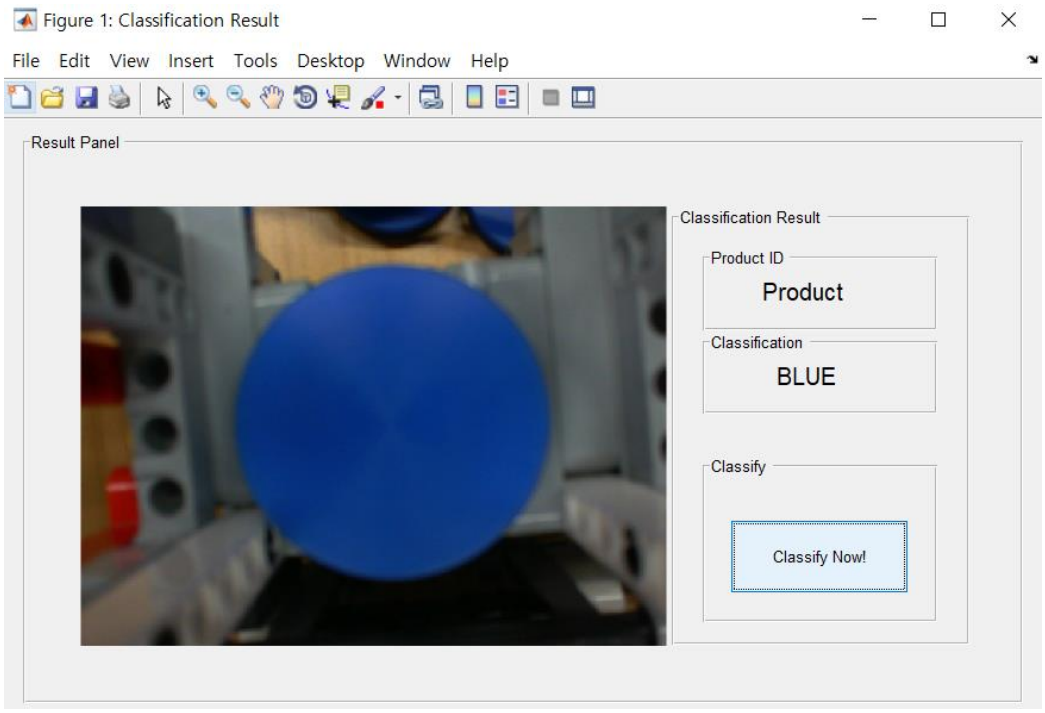


Figure 1: Classification Result
File Edit View Insert Tools Desktop Window Help
Result Panel
Classification Result
Product ID
Product
Classification
BLUE
Classify
Classify Now!

5. Production Line Configuration

a. Integrated Production Line Structure

The final goal of implementing CNN is to have the second machine as a quality control. The result of training can now be used for the classification. A coordination between the first machine, defect-sorting machine, and the sorting algorithm has to be built. An example of a complete configuration is as shown in the following figure.

- A chip enters the system through the feeder conveyor, and queue behind the gate.
- If M1 is available, the gate will be open, and the chip enters the machine.
- If the machine fails, the chip waits for repair in the machine.
- Upon completion, the chip is transferred to a conveyor to the defect-sorting machine.
- In the machine, the chip will be classified.
- If the chip is classified as good, it will be stored in the good storage, otherwise it will transported to the defected storage.

For visualization, a real-time GUI shows the camera-view and the classification of the products.

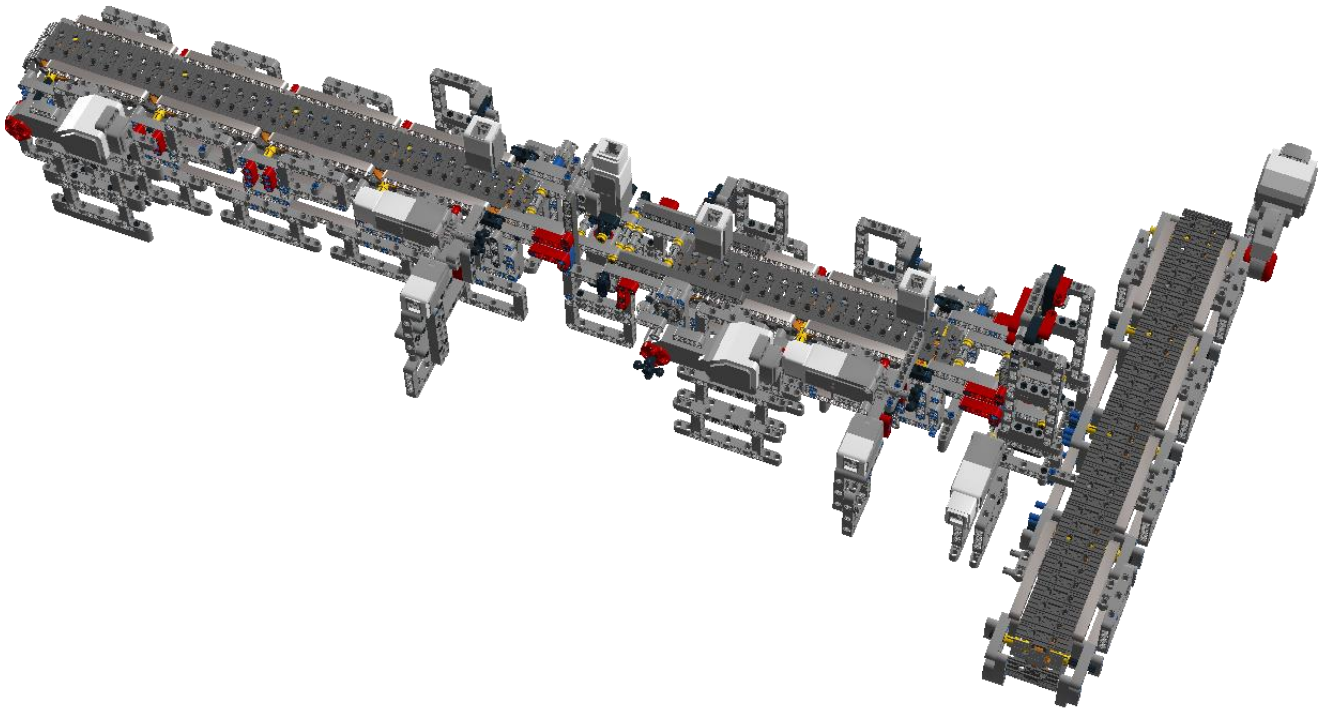
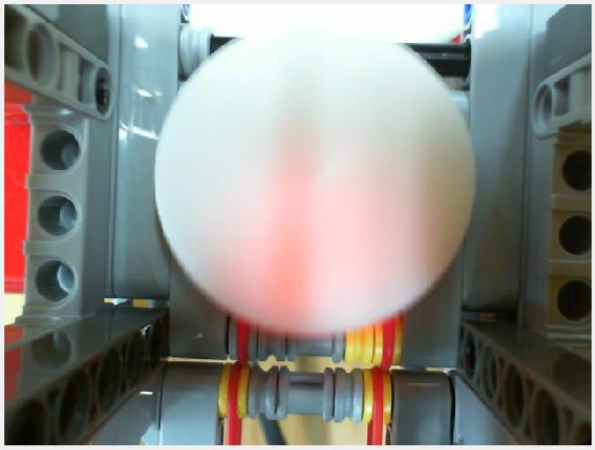


Figure 1: Classification Result

파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)

LEGO Two-Machine One-Buffer Deep Learning Based Vision Test



Result Panel

Classification Result

Product ID

5

Classification

WHITE

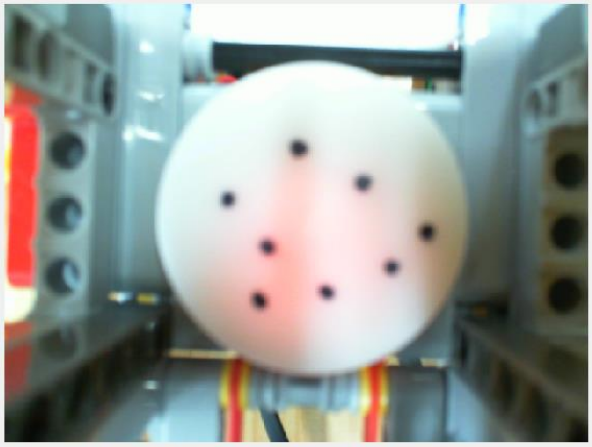
Close

Close this view

Figure 1: Classification Result

파일(F) 편집(E) 보기(V) 삽입(I) 툴(T) 데스크탑(D) 창(W) 도움말(H)

LEGO Two-Machine One-Buffer Deep Learning Based Vision Test



Result Panel

Classification Result

Product ID

7

Classification

DEFECT

Close

Close this view

Reference

[1] KAIST-LEGO Production Line Instruction, https://www.sdm.kaist.ac.kr/education-ie251