Camera-Based System for Automatic Pin Mapping on Breadboards

Date: 25.06.03

Author: 21900179 / JongHyeon Kim

22000188 / ChanJung Kim

Github: repository link

Demo Video: Youtube link

Introduction

The students in the school of Mechanical and Control Engineering frequently encounter a variety of IC (integrated circuit) chips during class experiments. However, since these chips often include both manufacturer names and production dates printed together, it can be difficult to identify the exact model of the chip. Additionally, the process of manually searching for the appropriate datasheet for each chip can be inconvenient and time-consuming. To address these issues, this project proposes a vision-based system for automatic chip identification and pin mapping. This vision machine aims to identify 5 IC chips.

Problem Statement

- The presence of manufacturer names and production dates makes it difficult to accurately identify IC chip models.
- Manually searching for datasheets to determine pin configurations is time-consuming and inefficient.

Project Objectives

- Filter out unnecessary printed information and accurately identify the 5 IC chip models.
- Detect chip pins and label each corresponding node.

Project Overview

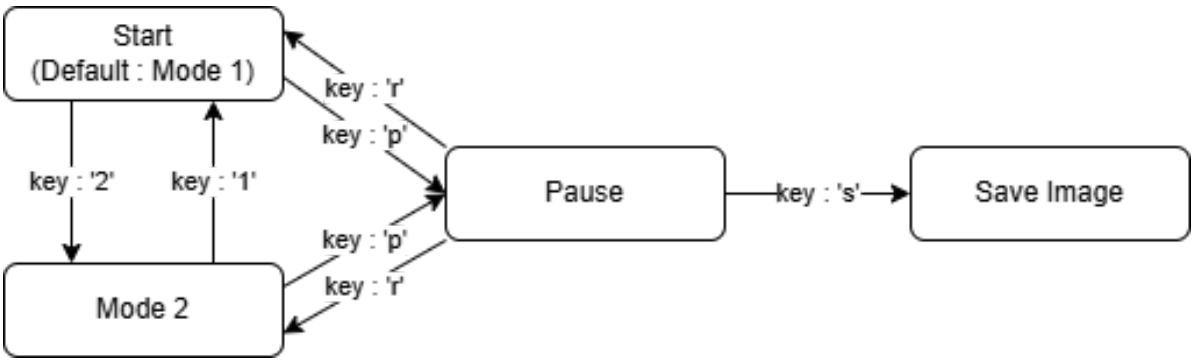
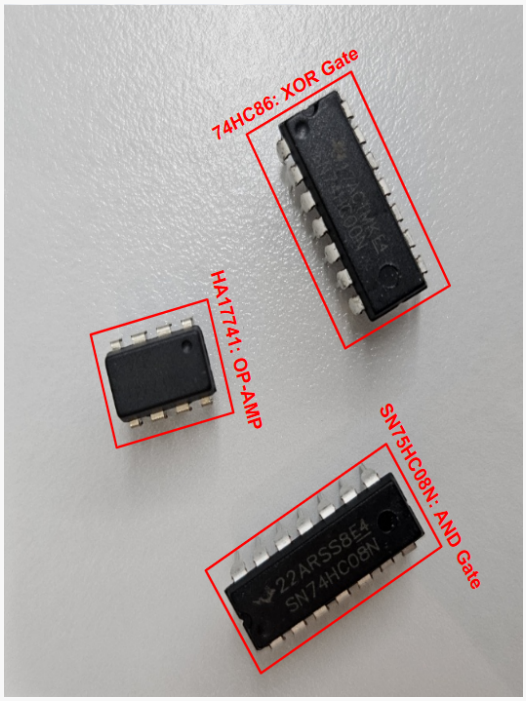


Figure 1. Overall Status Chart

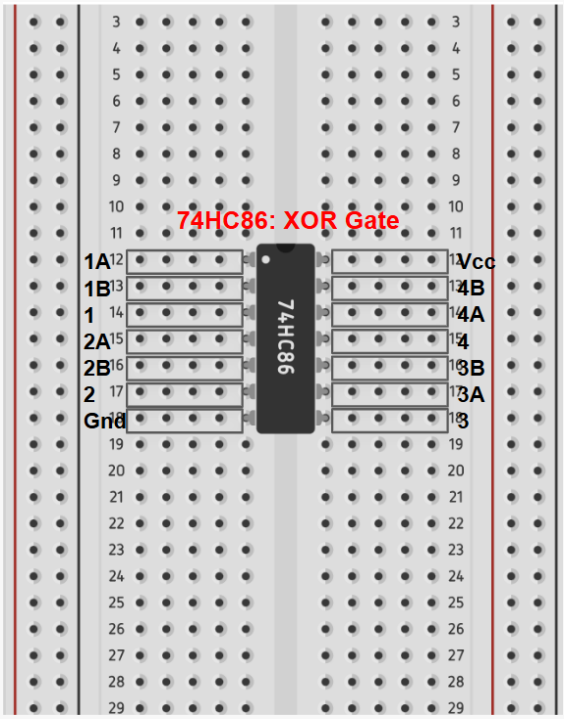
Mode	Command Key	Description
Mode 1	1	Detects scattered IC chips and identifies what chip it is.
Mode 2	2	Identifies IC chips and displays detailed information about each pin node.
Pause	p	Pauses the machine vision and freezes the current frame.
Restart	R	Restarts real-time operation of the machine vision system.
Save Image	S	Saves the currently paused image for reporting or documentation purposes.

When the program is launched, it starts in **Mode 1** by default, where IC chips are detected and their models are displayed on screen. The camera feed is provided by a smartphone connected to the laptop. Users can switch between **Mode 1** and **Mode 2** by pressing the corresponding numeric keys (`1` or `2`) on the keyboard. Pressing the `P` key pauses the video stream to allow closer inspection of a specific chip. To resume the program, the `R` key can be used. Additionally, while the video is paused, pressing `S` enables users to save the current frame for later use in reports or documentation.

Expected Output



Mode 1



Mode 2

In **Mode 1**, the system identifies the IC chip's model name and its general application, while in **Mode 2**, the system provides additional information, such as detailed pin descriptions and node labeling.

Members' roles

- JongHyeon Kim : Pre-processing / Sub Function / Smartphone Webcam build
- ChanJung Kim : OCR processing / Detecting and Displaying / Tesserect OCR build

---

Requirement

Hardware List

- Smart phone (used as a webcam via Window11 webcam Connection)

Software Installation

- OpenCV-python 4.11
- Python 3.9.21
- Pytorch 2.1.2
- Pytesseract 0.3.13

Evaluation Environment

Evaluate whether the IC chip is displayed well through the image. The information indication of the IC chip is based on OCR technology. Currently, the accuracy with the default Tesseract setup and minor tweaks is around 80-90% on good-quality images, about 60% on medium-quality ones, and 0% on poor-quality images. These results are affected by factors such as the small size of the chip text, reflections caused by lighting, and the use of a smartphone camera without additional optical aids. Given these constraints, the project aims to achieve an overall accuracy of **at least 70%**.

The accuracy is calculated by following equation:

$$Accuracy = \frac{TN + TP}{ALL},$$

The evaluation must be conducted under the following conditions:

- The text printed on the IC chip must be clearly visible.
- The chip must not be rotated upside down.

The evaluation method will determine the accuracy according to whether the information is well-marked or not. After instantaneously capturing the information of one IC chip, the system saves the image, and the evaluator labels it as true or false. By counting true images, the evaluator may confirm its accuracy.

Misson Output

1. Make sure each individual IC chip is correctly recognized and identified when scanned with a smartphone.
2. Check that all IC chips are accurately detected and identified when multiple chips are present.
3. Verify that each pin's information (such as GND, VSS, etc.) is correctly matched and displayed on the corresponding node.
4. Confirm that the labels for each pin are accurately positioned on the correct pin location without any misalignment.
5. Validate the proper functionality of all operating modes as specified in the system design.

Installation Guide

How to Install Pytesseract?

1. Download the Tesseract installer from the following link: [Tesseract Github (UB Mannheim)](). You may find the same page as below.

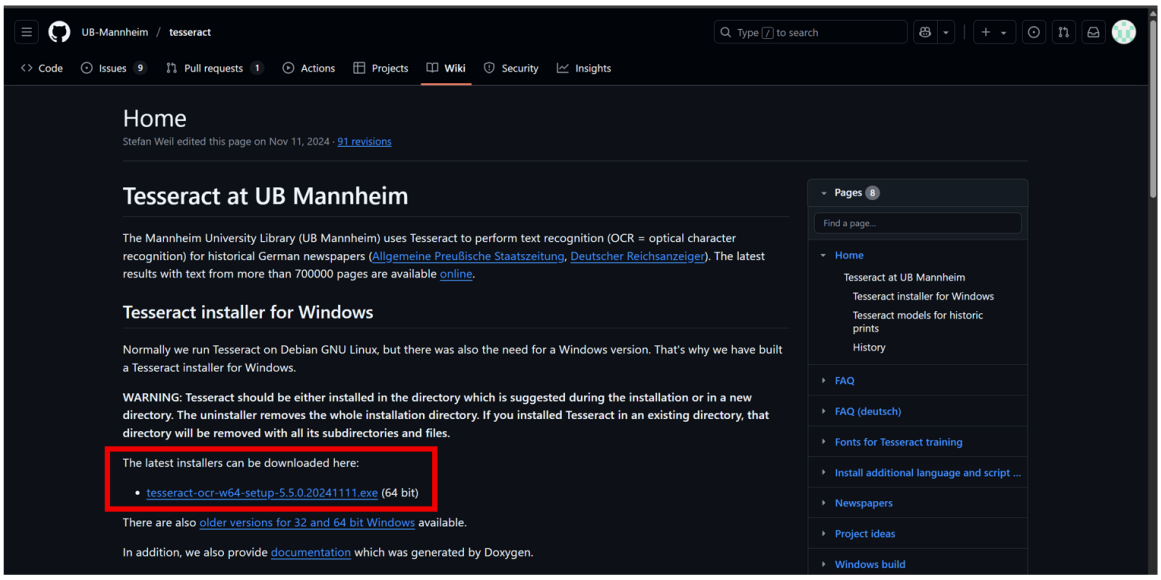


Figure 1. How to Download Tesseract

2. Click the `.exe` file highlighted in blue to begin the download, and install it as instructed. **Important:** During installation, **take note of the installation path** (eg. `C:\Program Files\Tesseract-OCR`). This path will be required when configuring system path.

3. Open the **Start Menu**, search for `환경 변수`, and open the corresponding settings window.

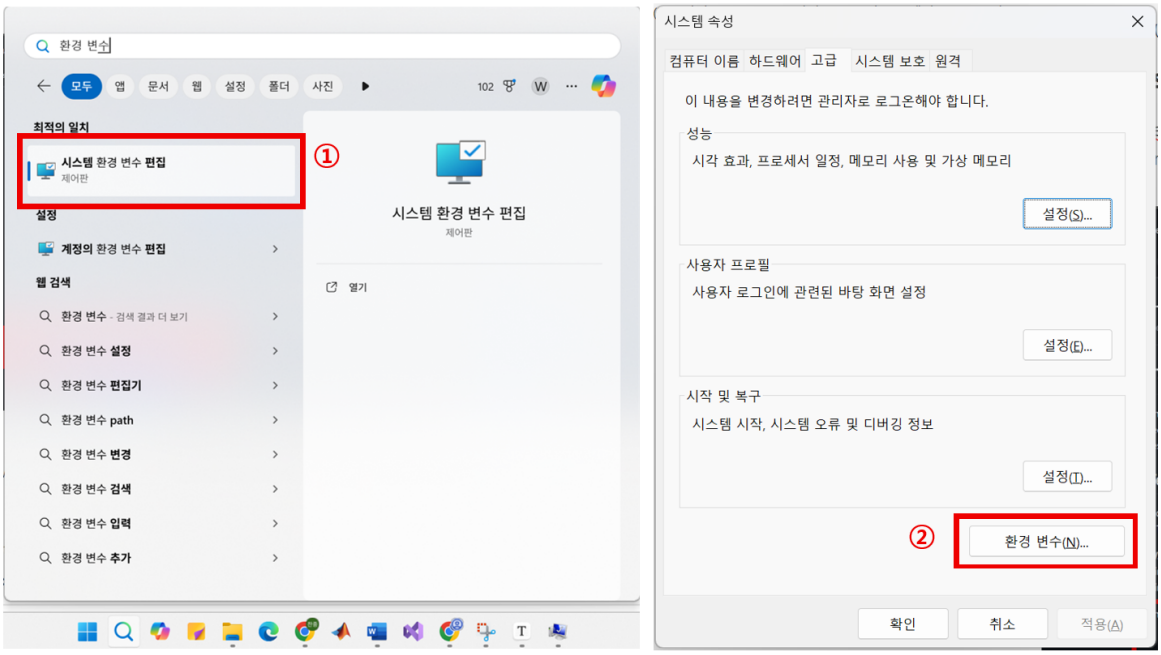


Figure 2. System Variable

4. In the User Variables section, find and select the variable named `path`, and add the path to the directory where `Tesseract-OCR` was installed.

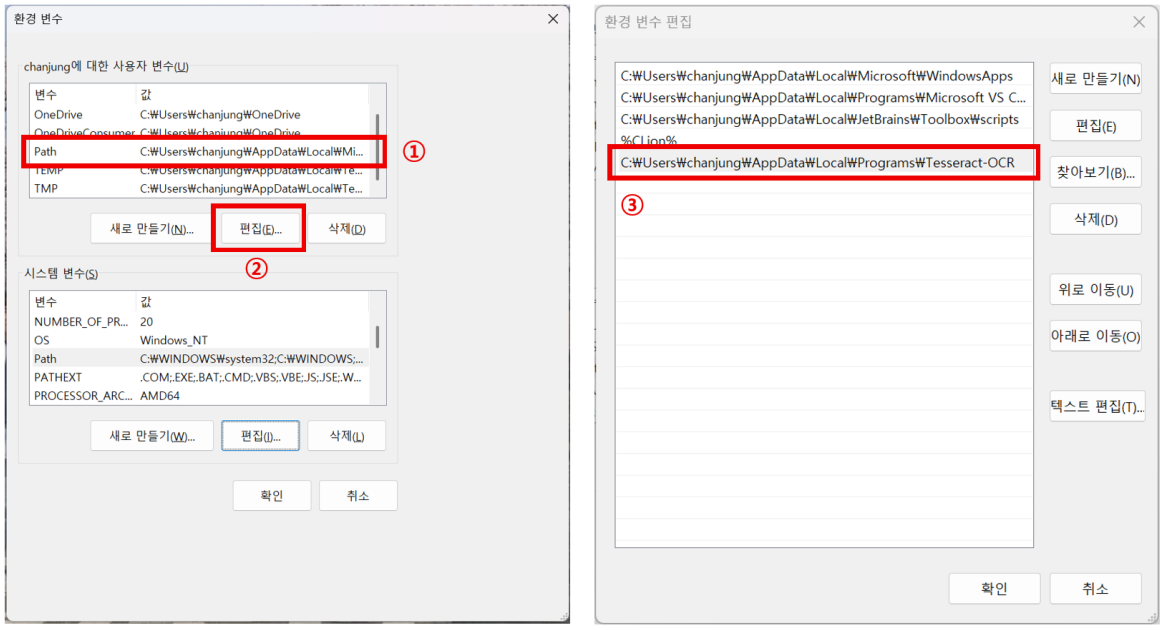


Figure 3. Edit Path to System Variable

If you encounter any difficulties during installation, please refer to YouTube for visual guidance.

5. Now, you can install `pytesseract` to your python environment by commanding the following codes to anaconda prompt.

```
(base) C:₩Windows₩System32>conda activate py39

(py39) C:₩Windows₩System32>pip install pytesseract
Collecting pytesseract
  Downloading pytesseract-0.3.13-py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: packaging>=21.3 in c:₩users₩user₩anaconda3₩envs₩py39₩lib₩site-packages (from pytesseract) (24.2)
Requirement already satisfied: Pillow>=8.0.0 in c:₩users₩user₩anaconda3₩envs₩py39₩lib₩site-packages (from pytesseract) (11.1.0)
Downloading pytesseract-0.3.13-py3-none-any.whl (14 kB)
Installing collected packages: pytesseract
Successfully installed pytesseract-0.3.13
```

Figure 4. Install Pytesseract

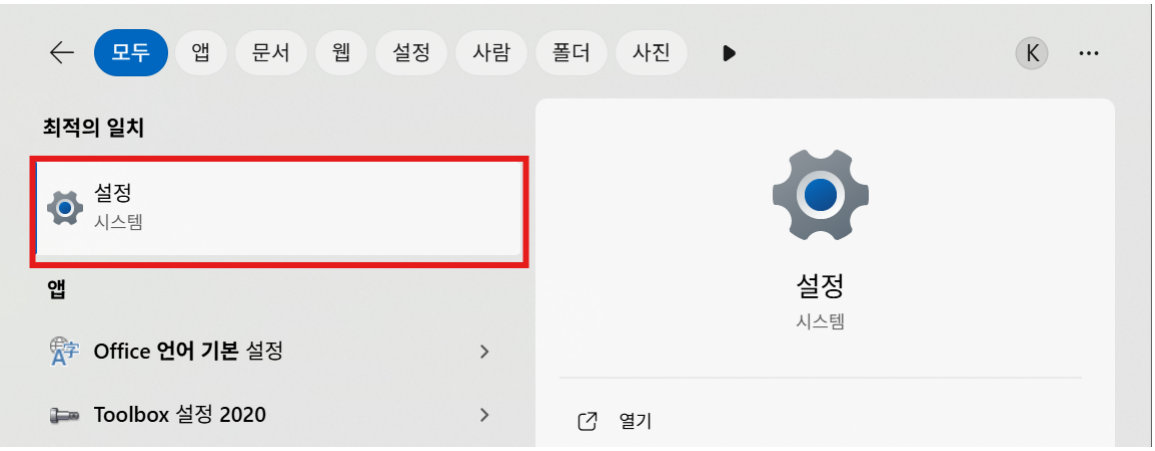
Then, in your Python script, you can set up and used `pytesseract` as follows:

```
import pytesseract as py
py.pytesseract.tesseract_cmd = r"your_path/Tesseract-OCR/tesseract.exe"
custom_config = r'--oem 3 --psm 6'

details = py.image_to_data(img, output_type=py.Output.DICT,
config=custom_config)
n_boxes = len(details['text'])
for i in range(n_boxes):
    if int(details['conf'][i]) > 60:  # Above 60% Accuracy
        (x, y, w, h) = (details['left'][i], details['top'][i], details['width']
[i], details['height'][i])
        text = details['text'][i]
        cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)
        cv2.putText(img, text, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.6, (36,
255, 12), 2)
```

How to connect a smartphone in Window11 as a webcam?

1. Search '**Settings**' in the Windows search box and press it.

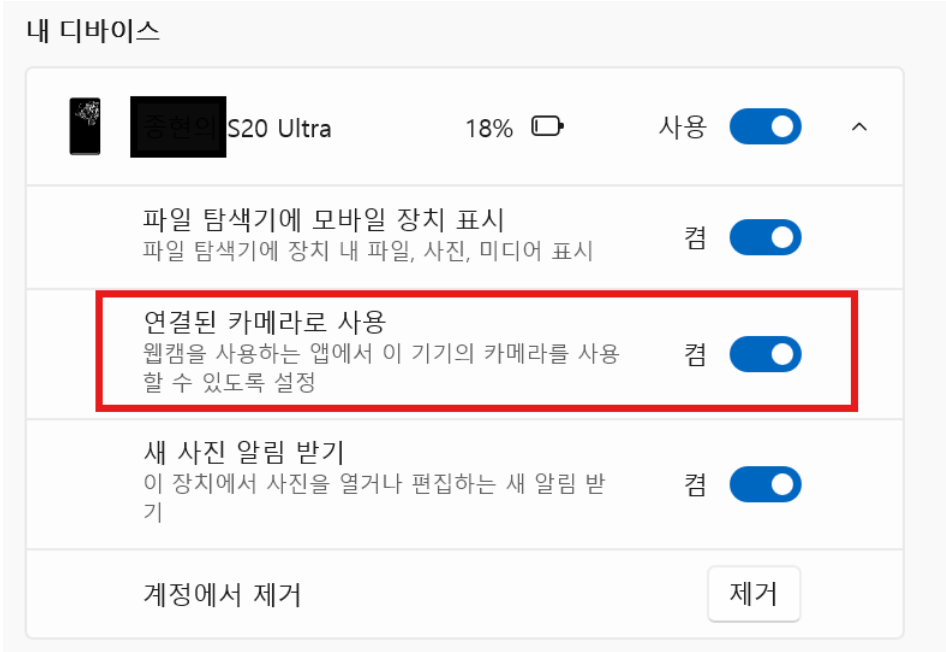


2. In the "**Bluetooth and Devices**" section, click "**Mobile Devices**".

3. Leave Allow to access mobile devices from your PC as "**Turn On**" and click "**Manage Devices**" in the mobile device section as shown in the following figure

4. If a device is not registered on the mobile device, click "**Add a device**" to add a device. When clicking, take a QR code with a smartphone and enter an authentication number as guided.

5. When you add a device, your device will be displayed as in the photo and change it to "**Turn on**" in the "**Use as a Connected Camera**" part.



The code below is the code that brings up the screen of smartphone webcam from Python code.

```
import cv2
cap = cv2.VideoCapture(1)
```

Method

OCR(Optical Character Recognition)

Explain briefly how you have implemented the project.

- Explain how you have done preprocessing
- Explain the deep learning models you have used
- Explain how you have tested for evaluation

Procedure

Explain the whole procedure step by step with proper headings and images.

Installation

Tutorials

Results and Analysis

Show the final results visually (images, graph, table etc)

Analyze the results in terms of accuracy/precision/recall etc..

Explain wheather you have achieved the project objectives

Reference

```
YouTube - How to Install and Use Tesseract OCR on Windows - Optical Character
Recognition [Link](https://www.youtube.com/watch?v=2kWvk4C1pMo)

Github - Tesseract-Ocr [Link](https://github.com/tesseract-ocr/tesseract)

Blog - 윈도우11 안드로이드 스마트폰 웹캠 사용 방법 [Link]
(https://blog.naver.com/kwy3000/223715853751)
```