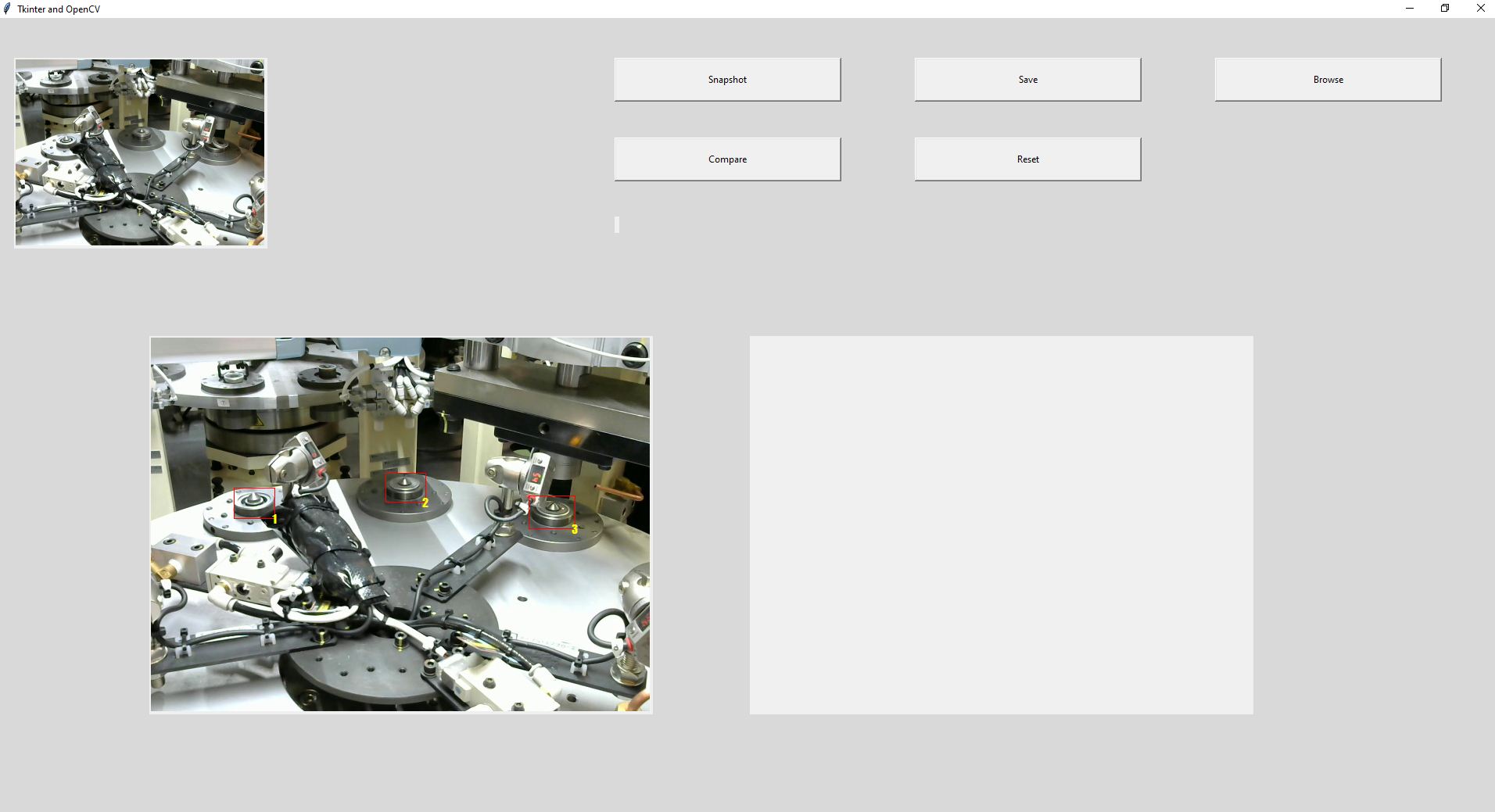
# Product’s Line ERROR Detection

## Introduction

**Product’s Line ERROR Detection** is the python software which use to check error process in pipeline by checking similarities between 2 images. Currently, this software work on both of Local PC and Mamos that you can easily capture and control GPIO input/output. 

## Developer Guide

### Requirements

Python 3.8.5

Pip 20.2.4

OS: Windows/ Linux/ Debian

### Libraries

certifi==2020.6.20  
imutils==0.5.3  
numpy==1.19.4  
olefile==0.46  
opencv-contrib-python==4.4.0.46  
opencv-python==4.4.0.46  
wincertstore==0.2  
Pillow==7.2.0  
pygubu==0.10.2

(requirement.txt)

### Installation

1. To download the code.

>> git clone xxx.git

>> cd aska\_products

1. Check your python version.

>> python3 –v

Python 3.8.5

>> pip –version  
Pip 20.2.4 from C:\...\pip (python 3.8)

1. Create python environments.

python3 –m pip install –user virtualenv

python3 –m venv venv

1. Run virtual environment.

source venv/bin/activate

1. To install python libraries.

>> pip install –r requirements.txt

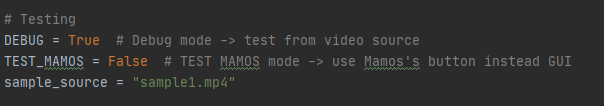
1. pip install –r requirements.txt

### Run Software

(venv) >> python3 camera\_show.py

### Overall

After python library was installed, it’s time to run the software. You can easily test this software on your own PC with DEBUG mode. This mode will be use video source (sample1.mp4) instead video source from camera.



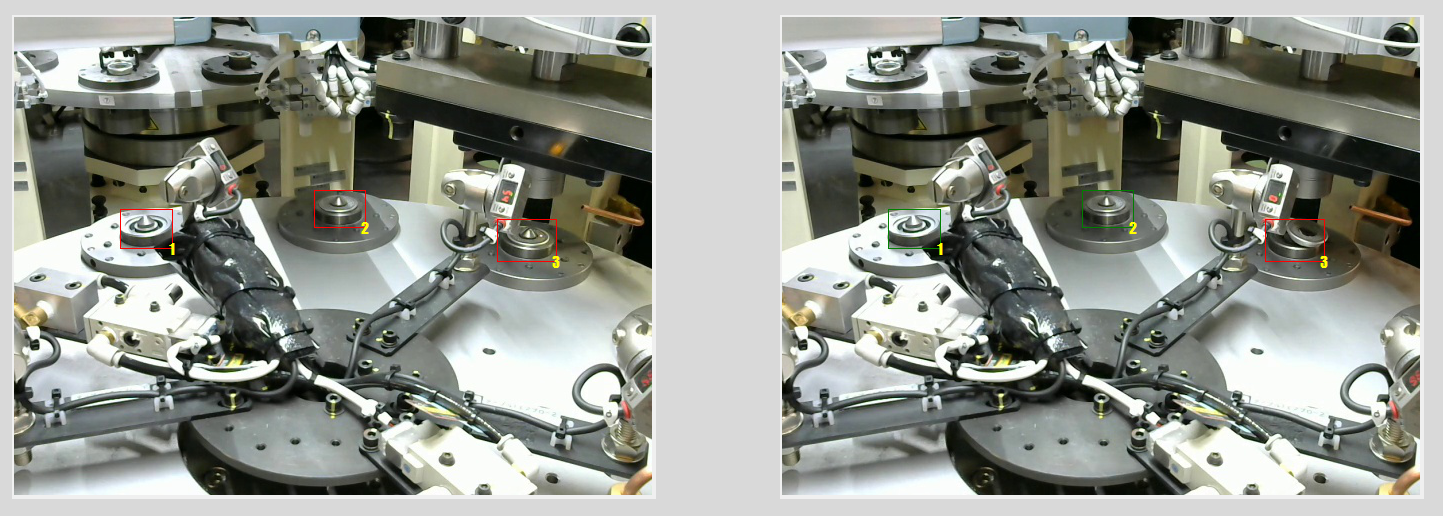
After program was started, it will automatically call the last json data that contains of “image name” and “rectangle position” .

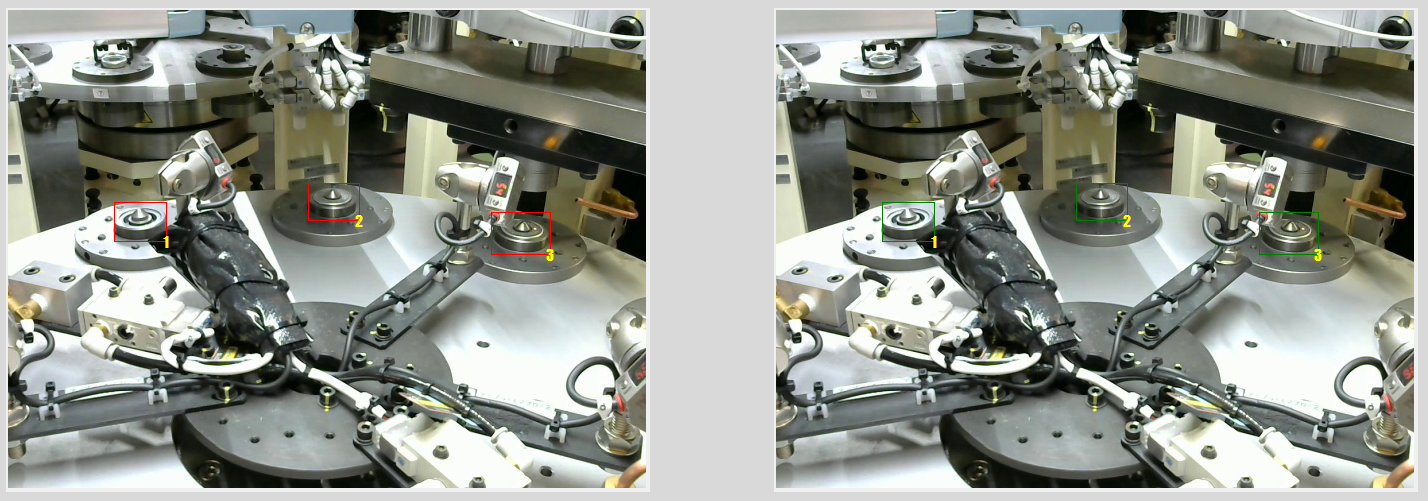


(camera\_show.py)



Finally, you can click **“Compare”** button anytime you want to check error from the current image from camera.





### GPIO

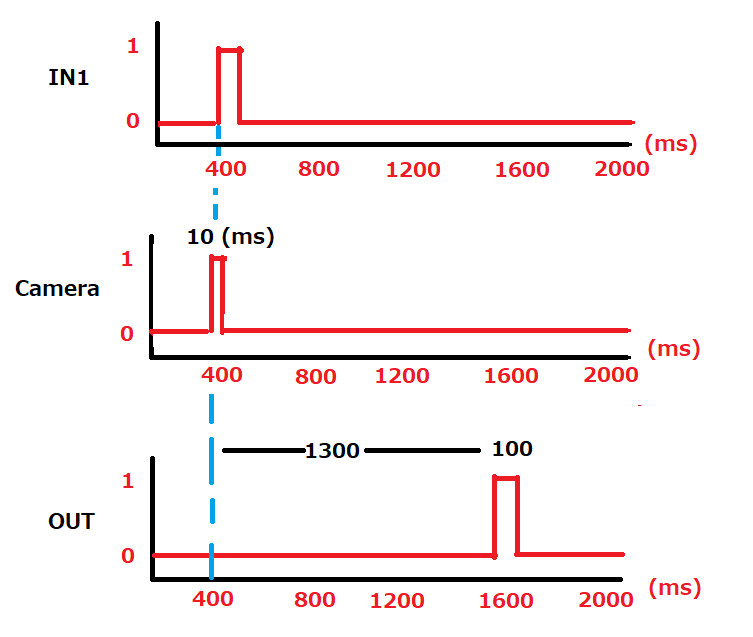
Input = pin 167

Output (ok) = pin 161

Output (ng) = pin 184

### Running Time

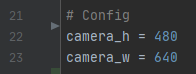
After you press button (PIN 167), it will capture current image from camera, then it will use the image in crop area to calculate similarity and show the result to output pin in 1300 millisecond (crop area = 640x480 pixel). This calculate time is depend on the size of crop area. For example, If size of crop area is 320x240, calculate time is about 650 milisecond.



(Crop area = 640x480 pixel)

### Camera Configuration

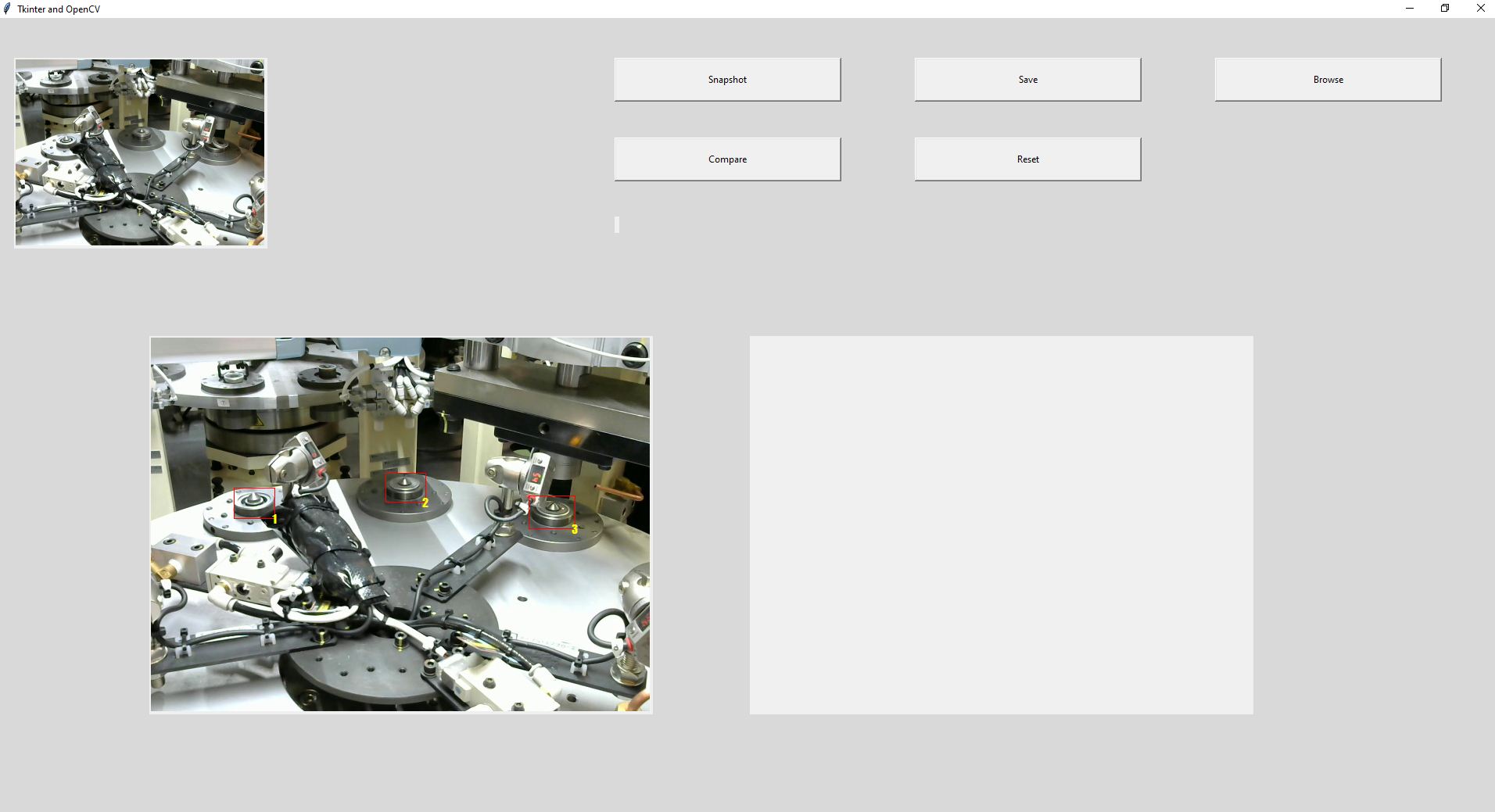
You can change the camera resolution in “camera\_show.py” here.



(camera\_show.py)

## User Guide

### Overall



**Snapshot Button:** Use to take a photo from current image from camera or video (DEBUG mode)

**Save:** Use to save a snap image and rectangle in left canvas with json file.

**Browse:** To load json data, then show the image on the left canvas.

**Compare:** To take a photo and show it on the right canvas, then compare image between left and right image.

**Reset:** To reset the all parameters and images

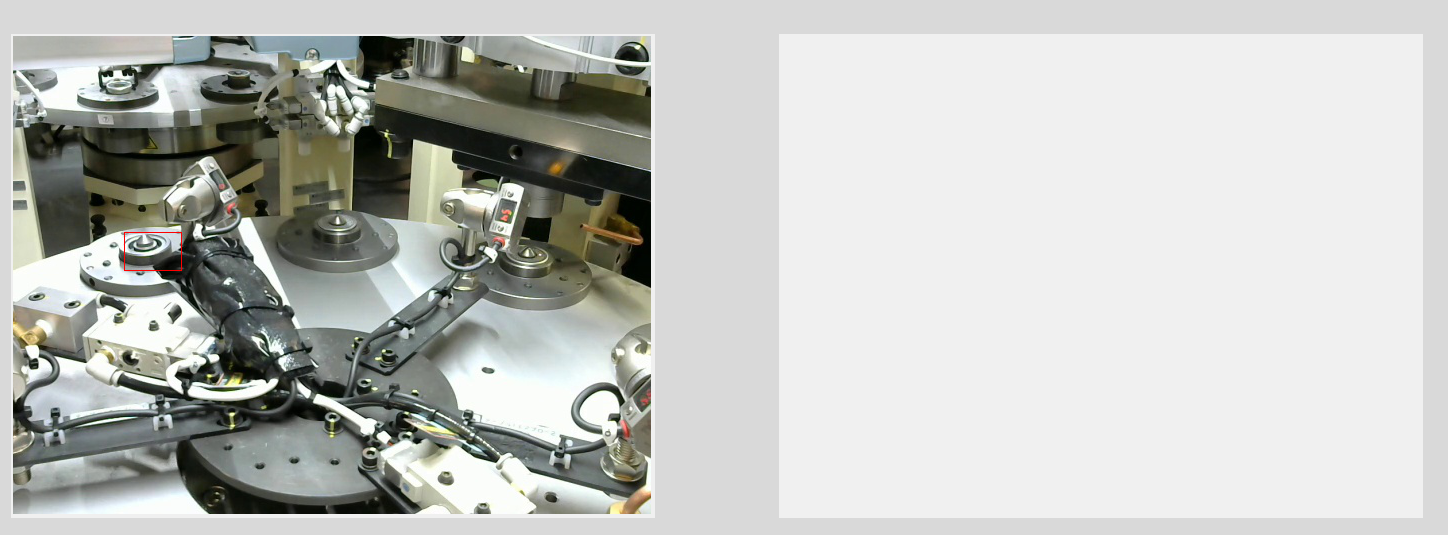
### FlowchartC:\Users\thirat\Documents\git\plastic_detection\doc\Plastic_detection.png

### Create Json Data

You can create new Json data with reset the screen by click “Reset” button, then the canvas on the left will be blank like this.



You can click the **“Snapshot”** button to get some original image, then draw some rectangle to crop detection area. Finally Click **“Save”** button to save Json file.



## Reference

1. Similarities checking (OpenCV)

https://pysource.com/2018/07/20/find-similarities-between-two-images-with-opencv-and-python/