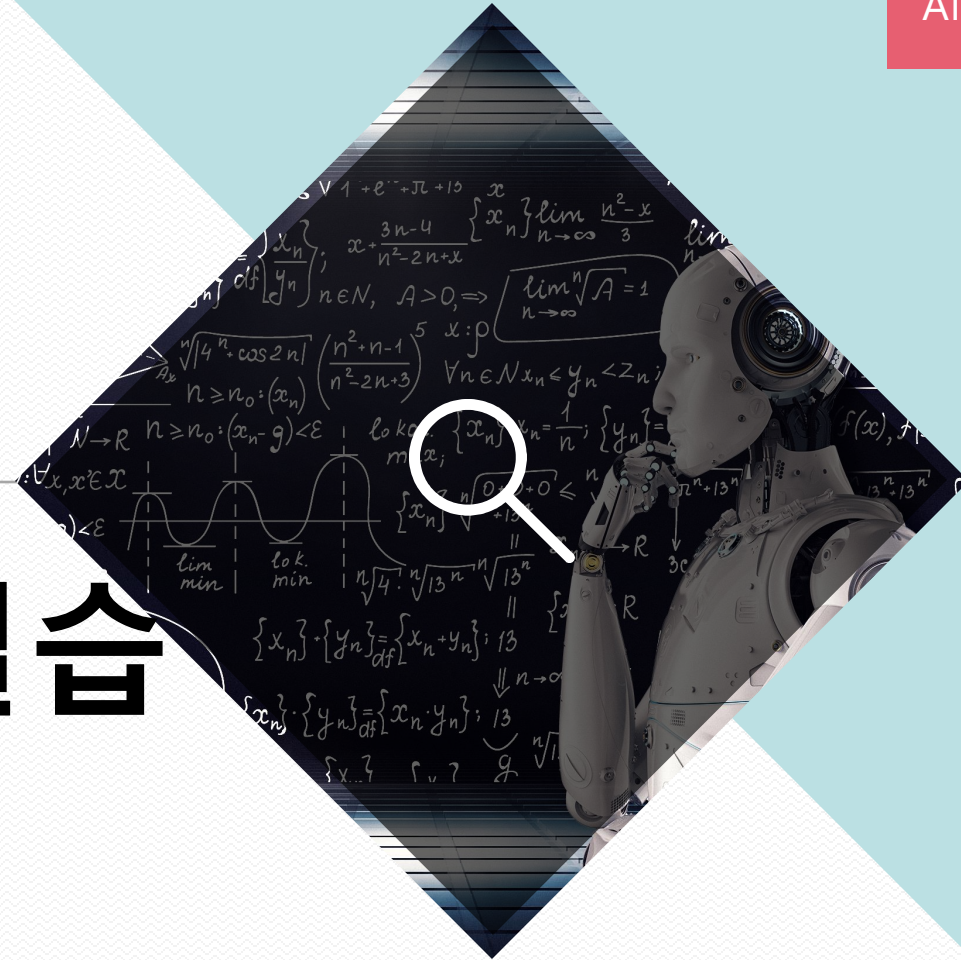


Smart Factory실습

Hands on for simulator



Agenda

- ***Smart Factory Workflow***
 - *First Scenario (Defect detection)*
 - *Second Scenario (Color detection)*
 - *ETC Scenario (System Start/Stop)*
- ***What to do on the Smart Factory***
 - *Planning*
 - *Gathering Dataset*
 - *Training using OTX*
 - *Inferencing on the Localhost*
 - *Utilize Arduino LED with Inferencing Result*
- ***Using Tools***
 - *Motion Detect Tool*
 - *Color Detect Tool*

Smart Factory Workflow

Overview

NUC

NUC10i7FN

10th Generation Intel® Core™ i7-10710U

1.1 GHz – 4.7 GHz Turbo, 6 core, 12 thread, 12MB Cache

25W Intel® UHD Graphics, 300 MHz – 1.15 GHz

16GB RAM / 512GB SSD

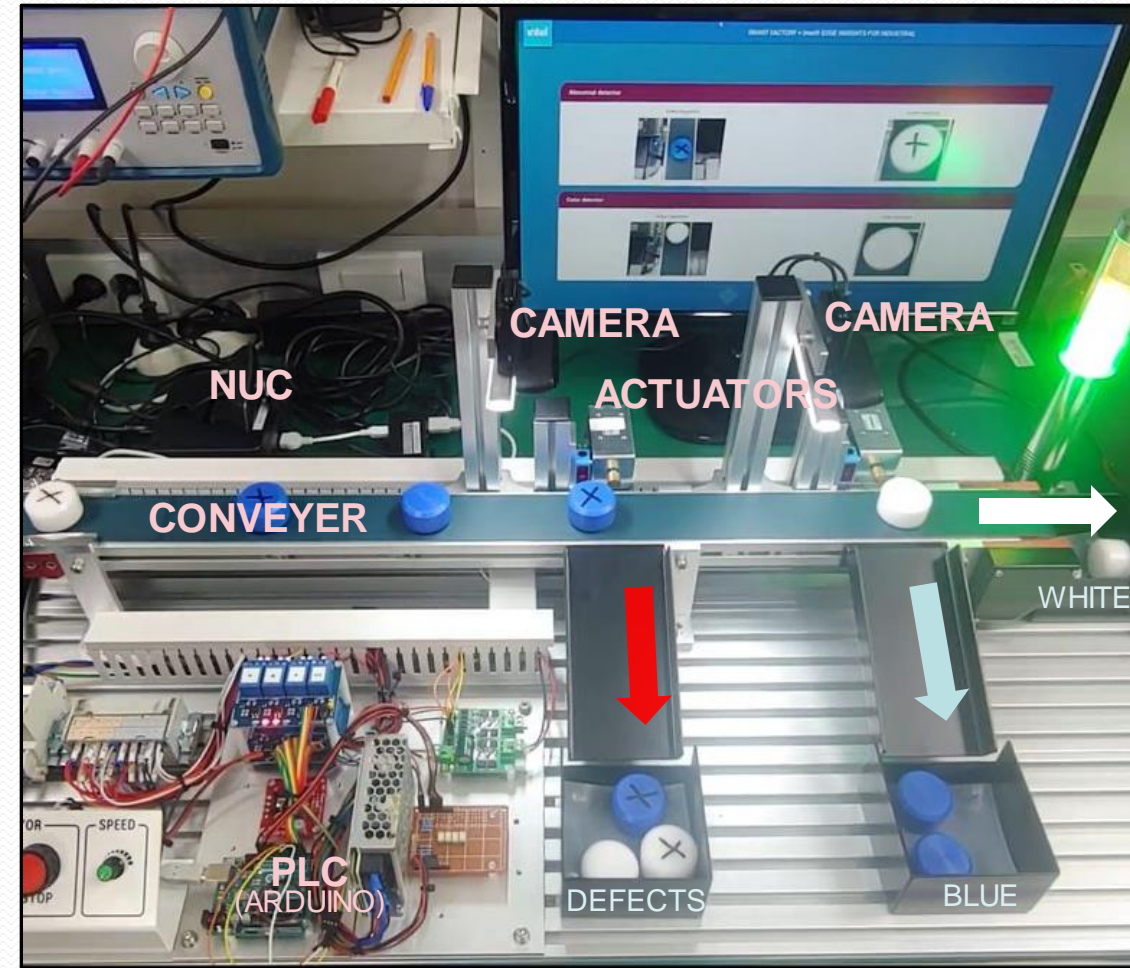
CAMERA

LOGITECH HD PRO WEBCAM C920N

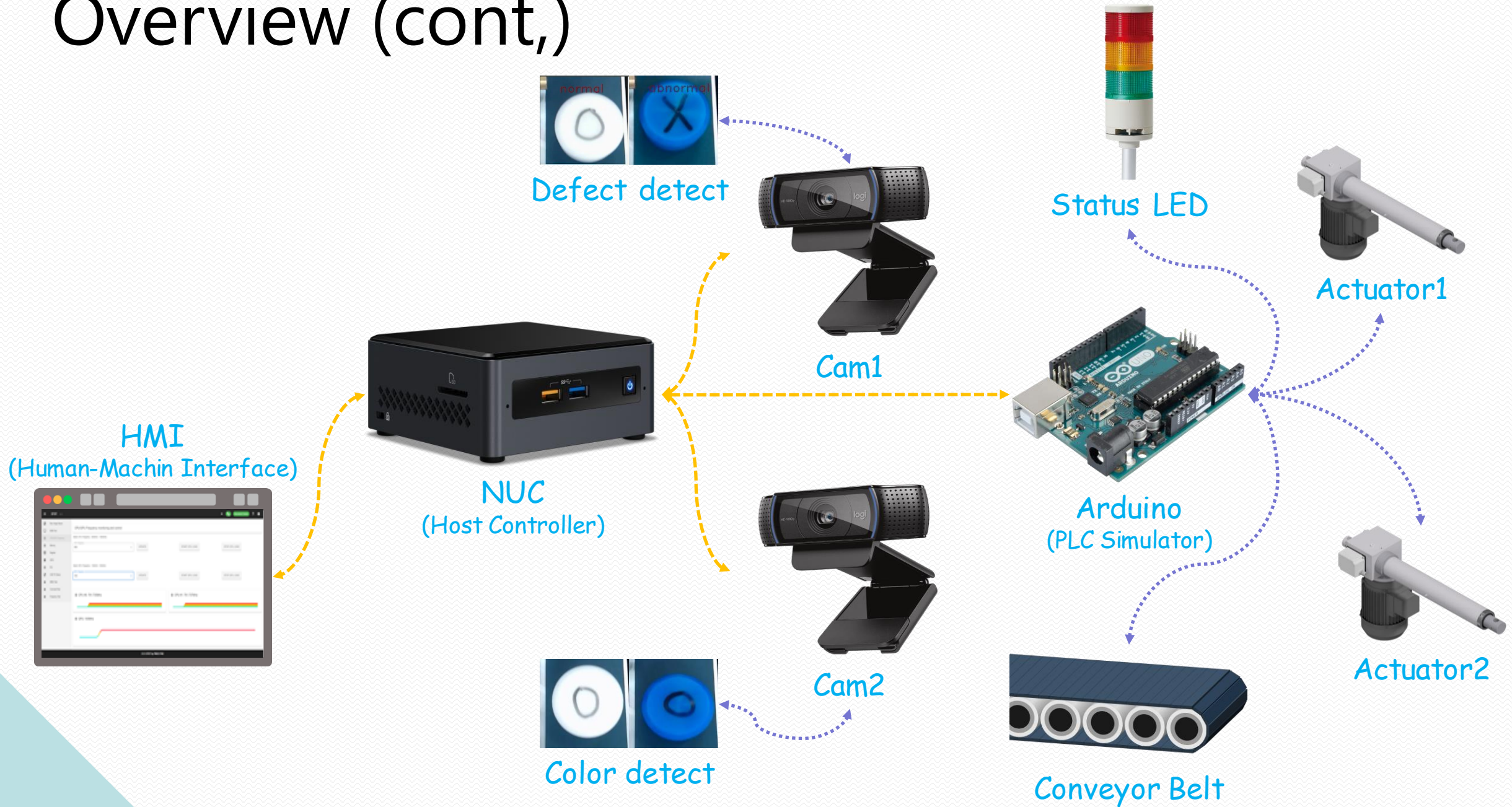
640x480 30 fps / up to 1920x1080 30 fps

ARDUINO

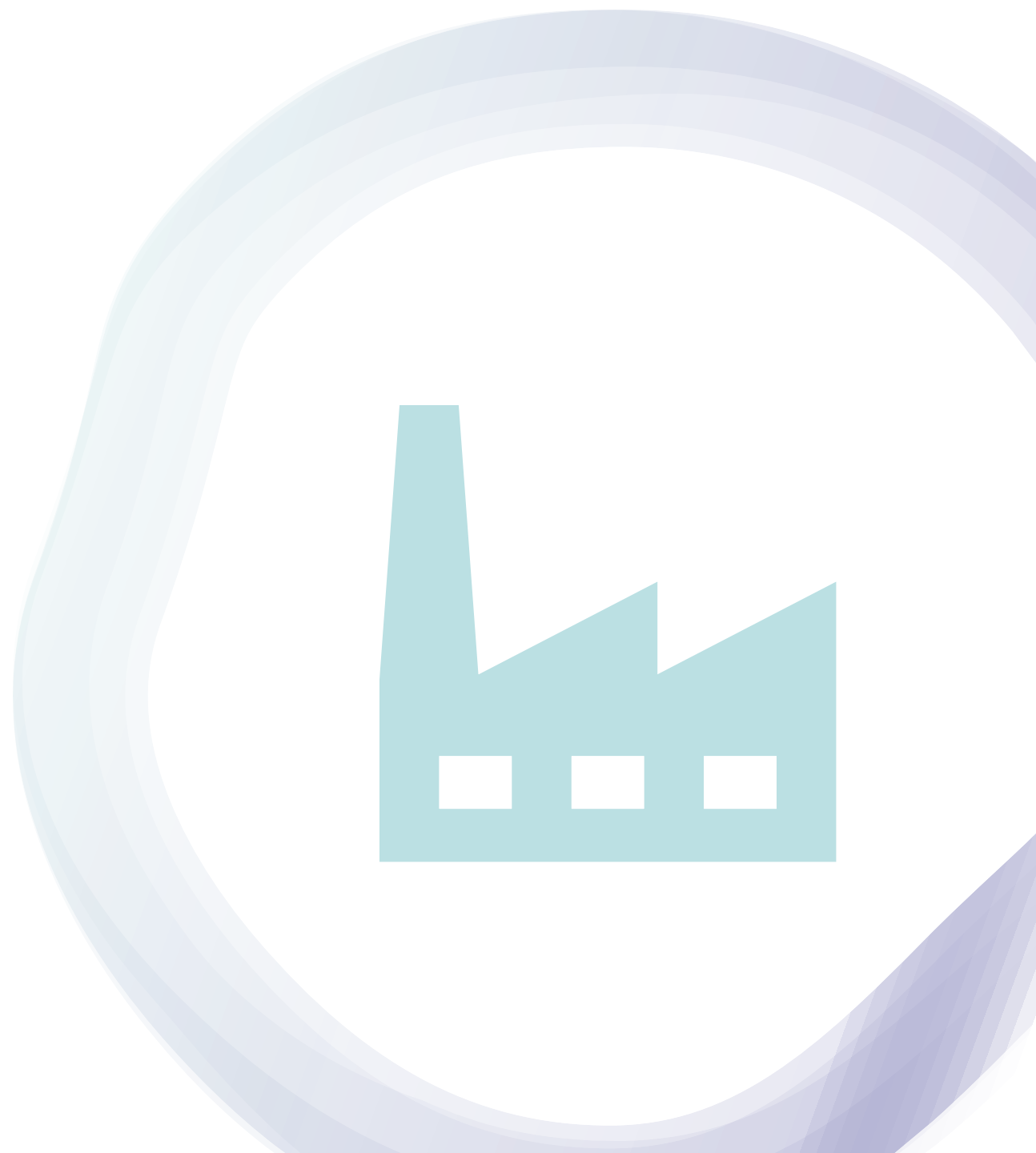
Uno



Overview (cont,)

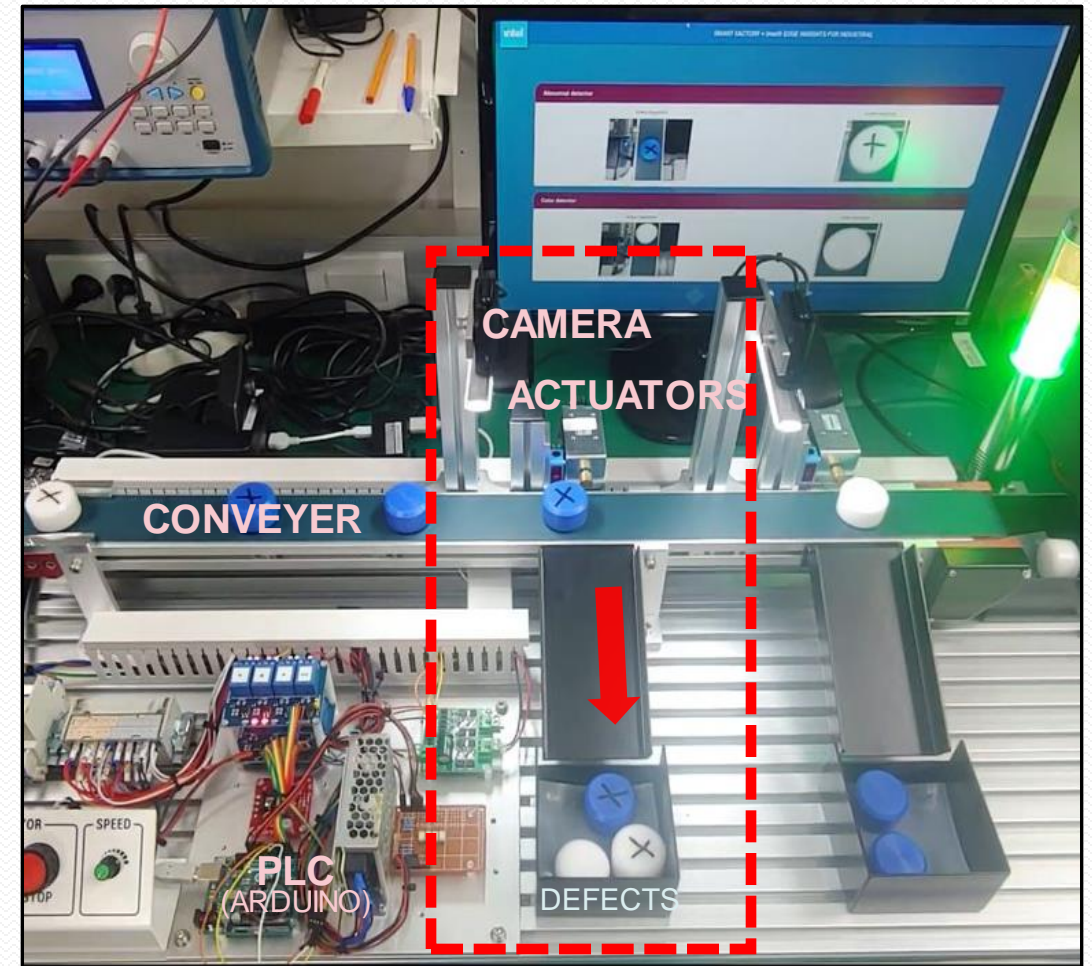


First Scenario



First Scenario

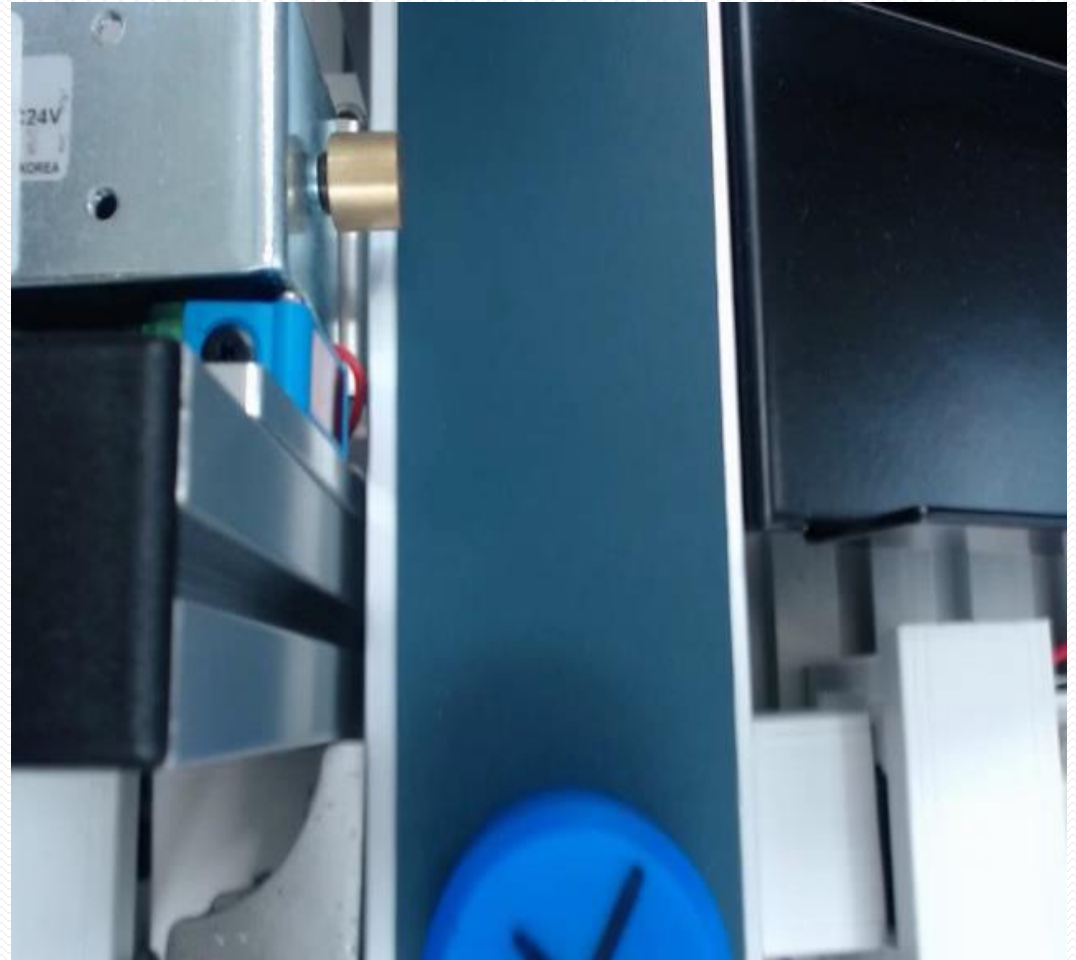
- When the conveyor belt moved with items,
 1. First camera detect the item
 2. In real-time video frame, get one image for detecting the item
 3. Using that one image, get the inferencing to result from localhost
 4. If an item is defective, an actuator will actuate and sort the item down
 5. If it is judged to be good, no action is taken



Camera (1)

- Video frame in real-time through the first camera on the Smart Factory
- Using OpenCV, we can catch the video frame in real-time
- There is no reason to acquire and use an image every frame
- So, need to idea that gets the specific one frame for inferencing
- Play the saved video

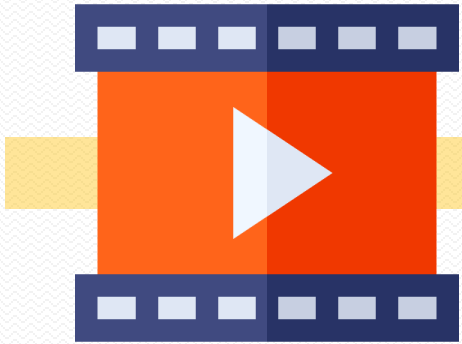
```
$ cd <root_dir>  
$ mplayer ./resource/factory/conveyor.mp4
```



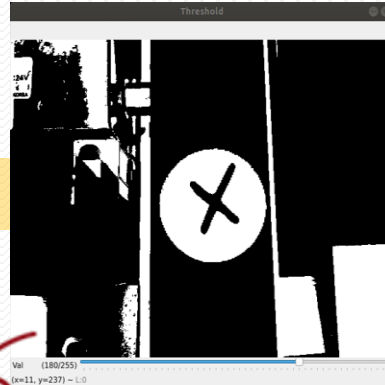
[`<root_dir>/resource/factory/conveyor.mp4`](#)

Motion Detect

Camera Input



Binarization

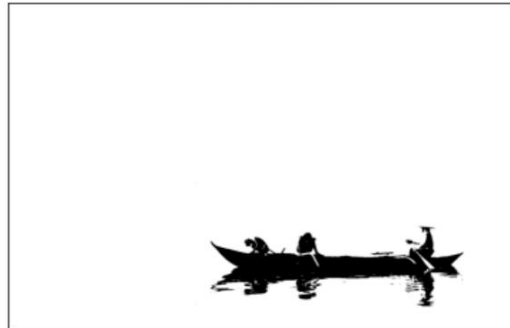


To drastically reduce the amount of information you have to work with

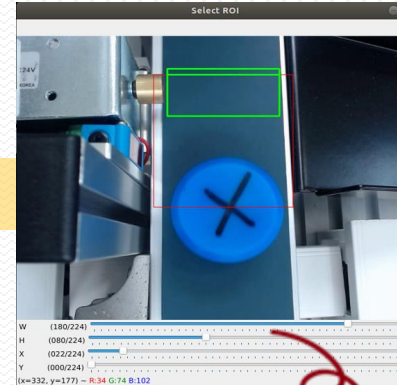
original image



filtered image



Region of Interesting
ROI Setting



Video Stream
with 640x480 resolution

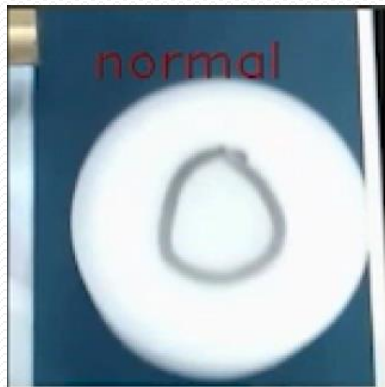
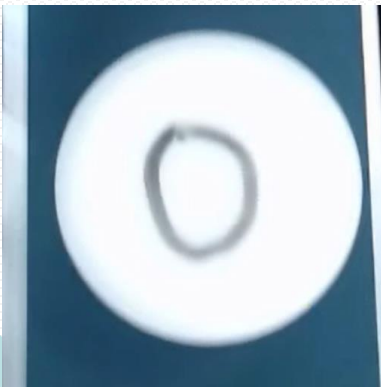
Motion Detect



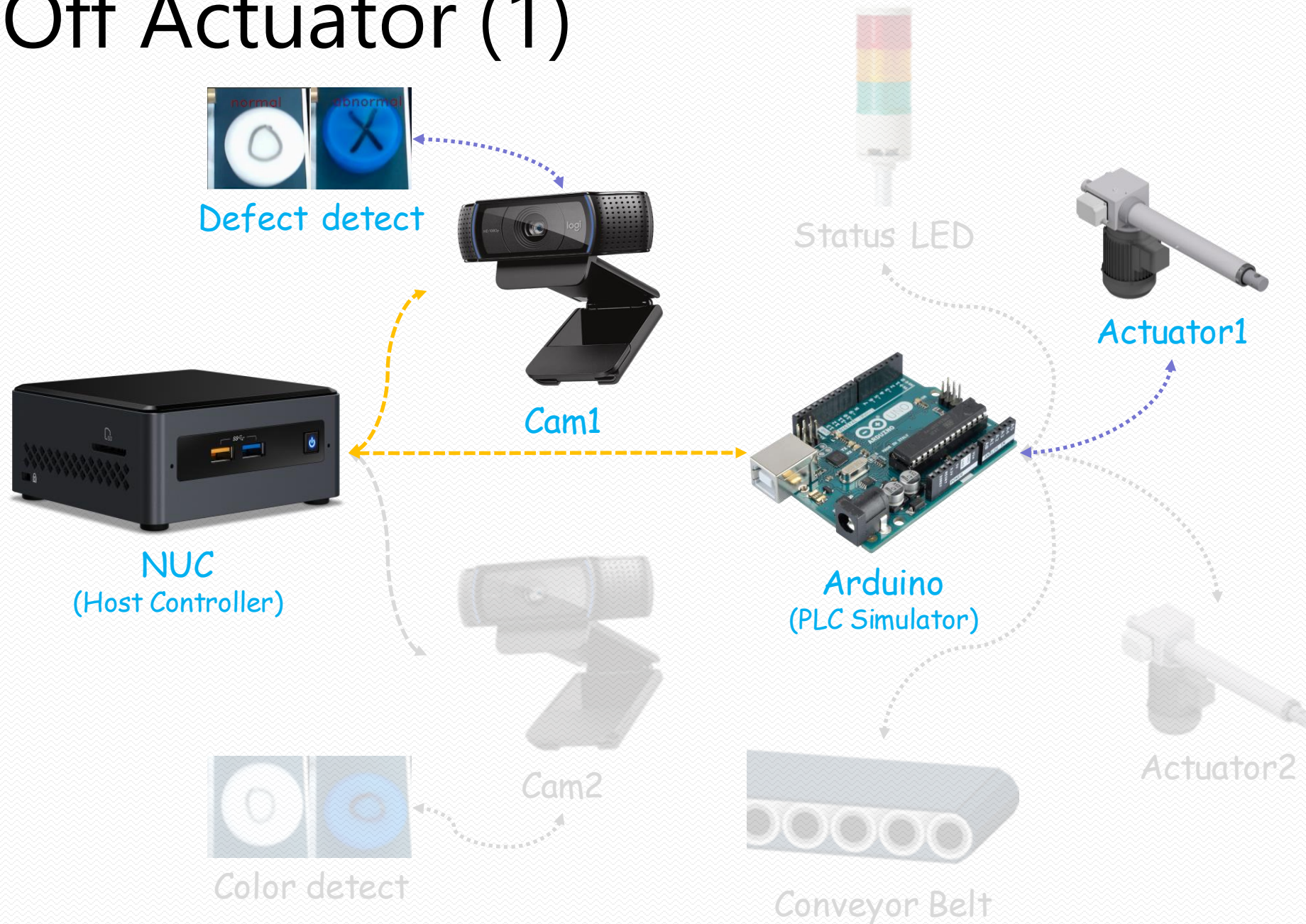
224x224 image

Inferencing

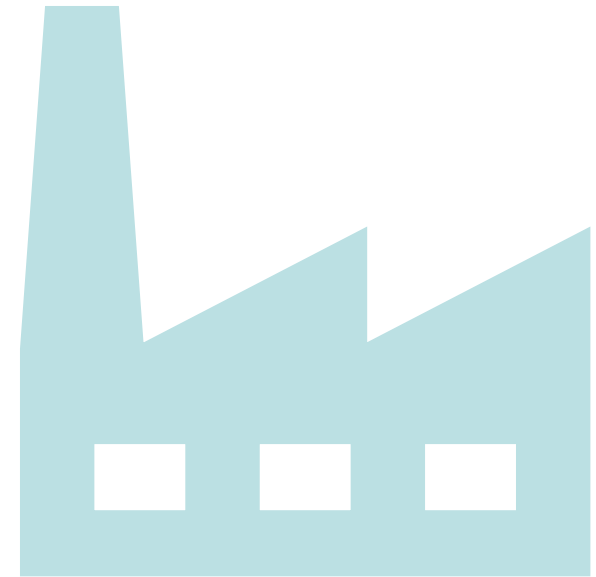
1. Load the trained model
2. Inferencing with video stream in real-time
3. Show the video stream with result



On/Off Actuator (1)

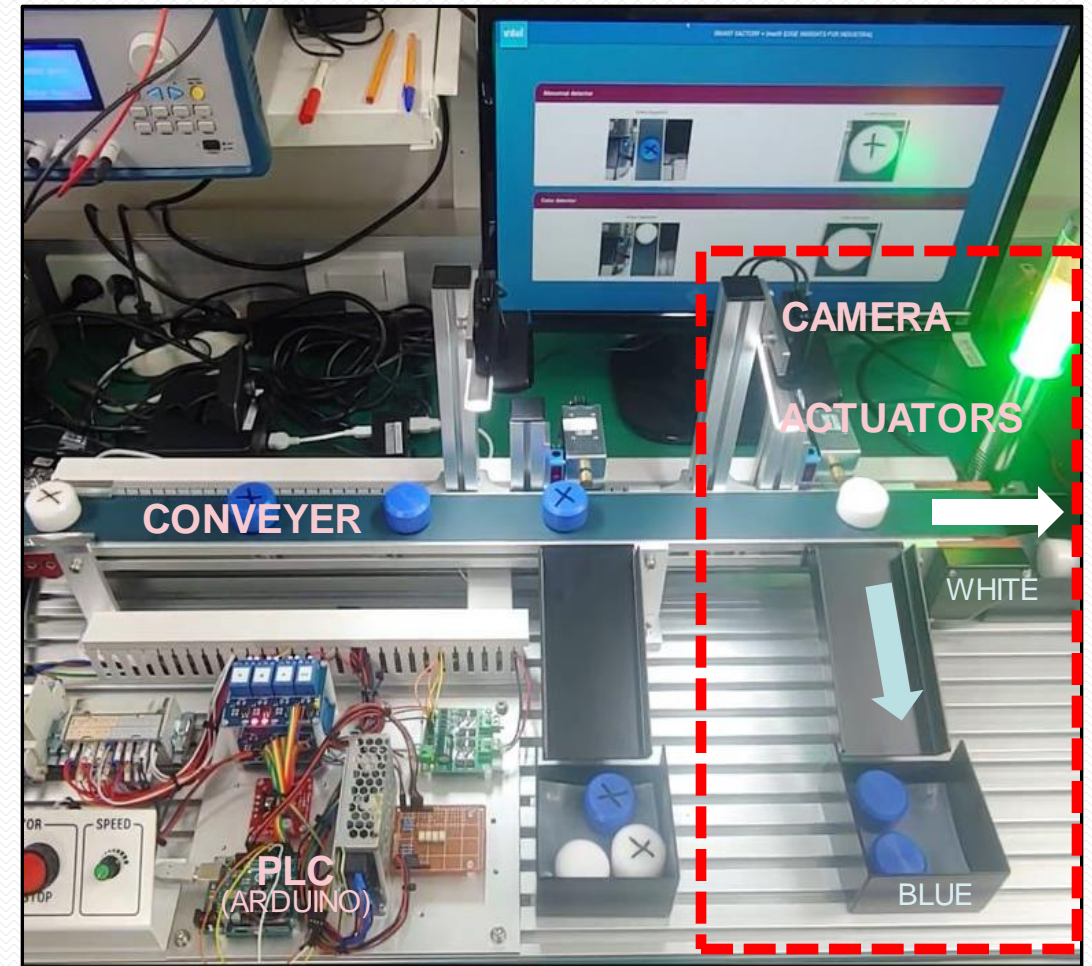


Second Scenario



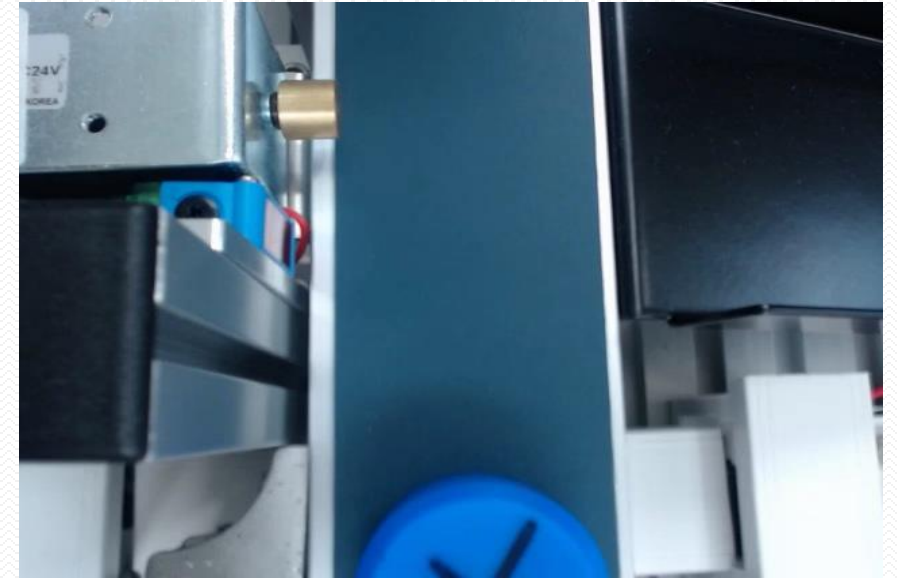
Second Scenario

- When the conveyor belt moved with items,
 1. Second camera detect the item
 2. In real-time video frame, get one image for detecting the item
 3. Using that one image, get the color result from color detection tool
 4. If an item has blue color, an actuator will actuate and sort the item down
 5. If it is judged to white color of item, no action is taken

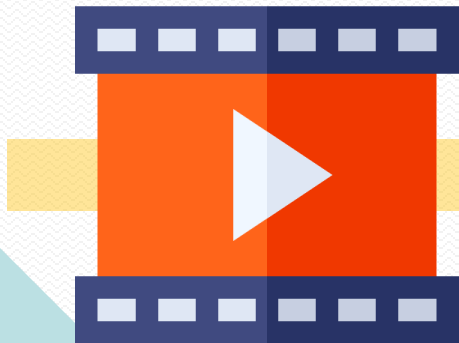


Camera (2) & Motion Detect

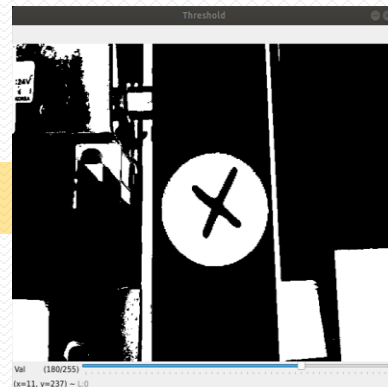
- Same scenario with the Camera (1)
- Need to idea that gets the specific one frame for color division
- Also, motion detection to get just one image from a video stream is the same scenario as before



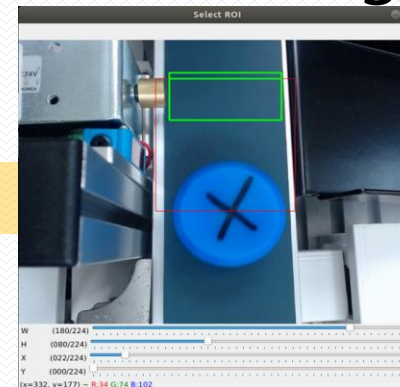
Camera Input



Binarization



Region of Interesting
ROI Setting



Motion Detect



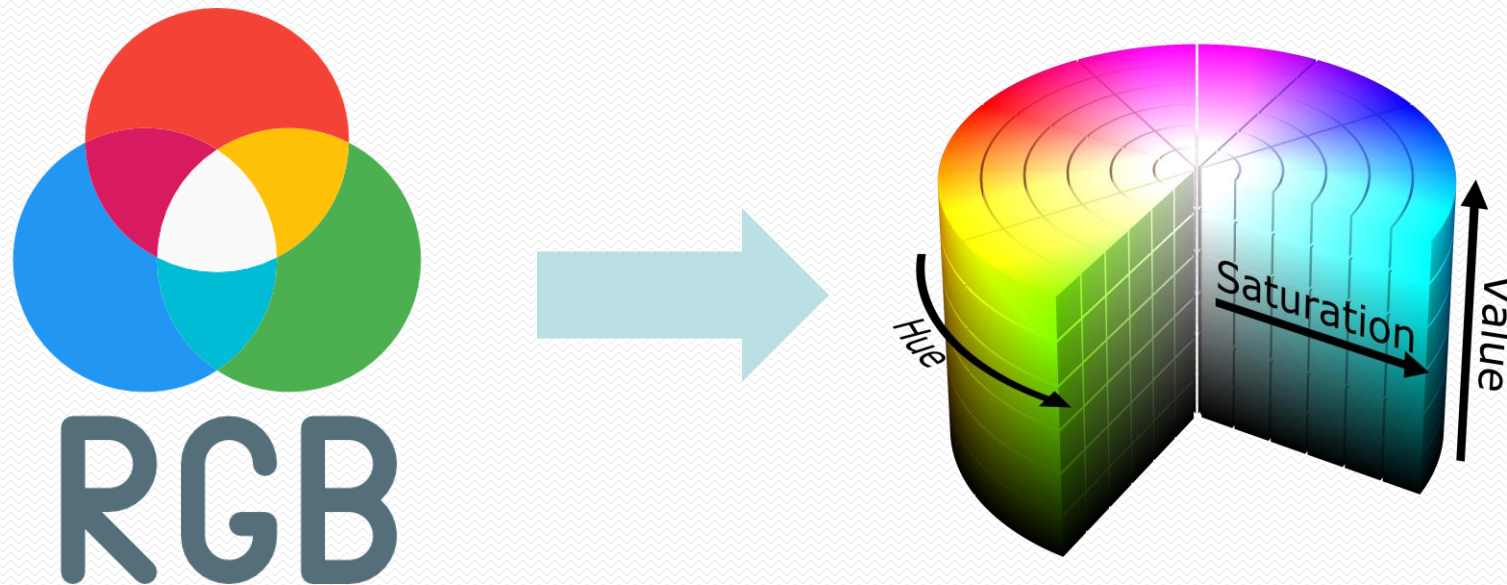
Color Detect

- *RGB (Red, Green, Blue)*

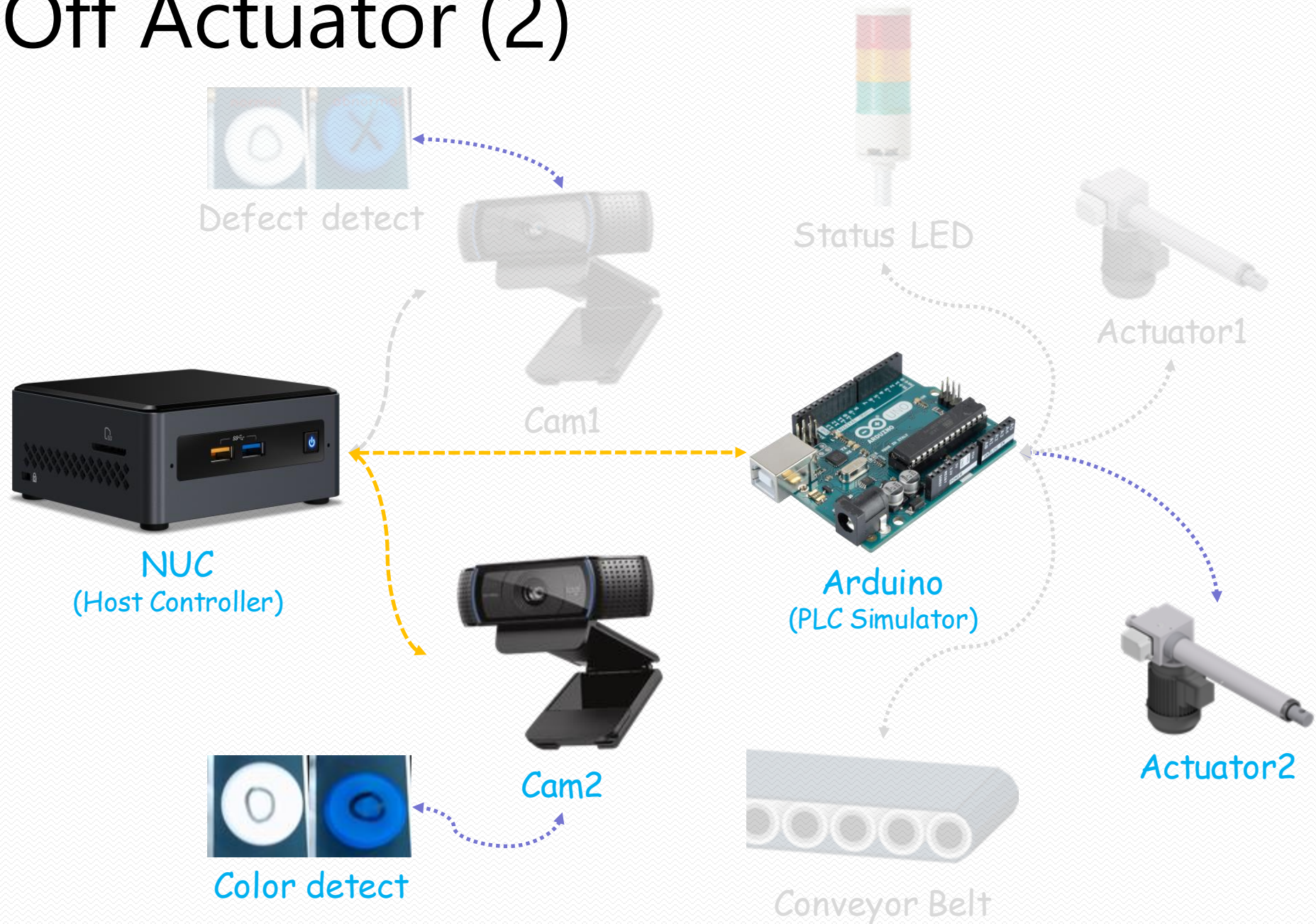
- additive color model in which the RGB primary colors of light are added together in various ways to reproduce a broad array of colors

- *HSV (Hue, Saturation, Value)*

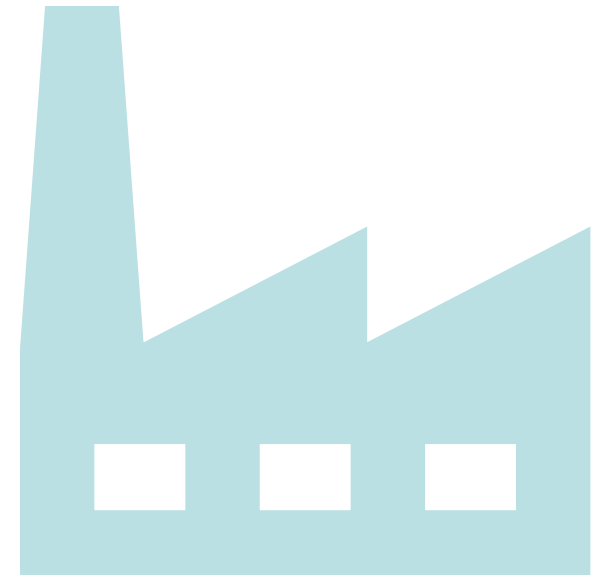
- more closely aligned with the way human vision perceives color-making attributes



On/Off Actuator (2)

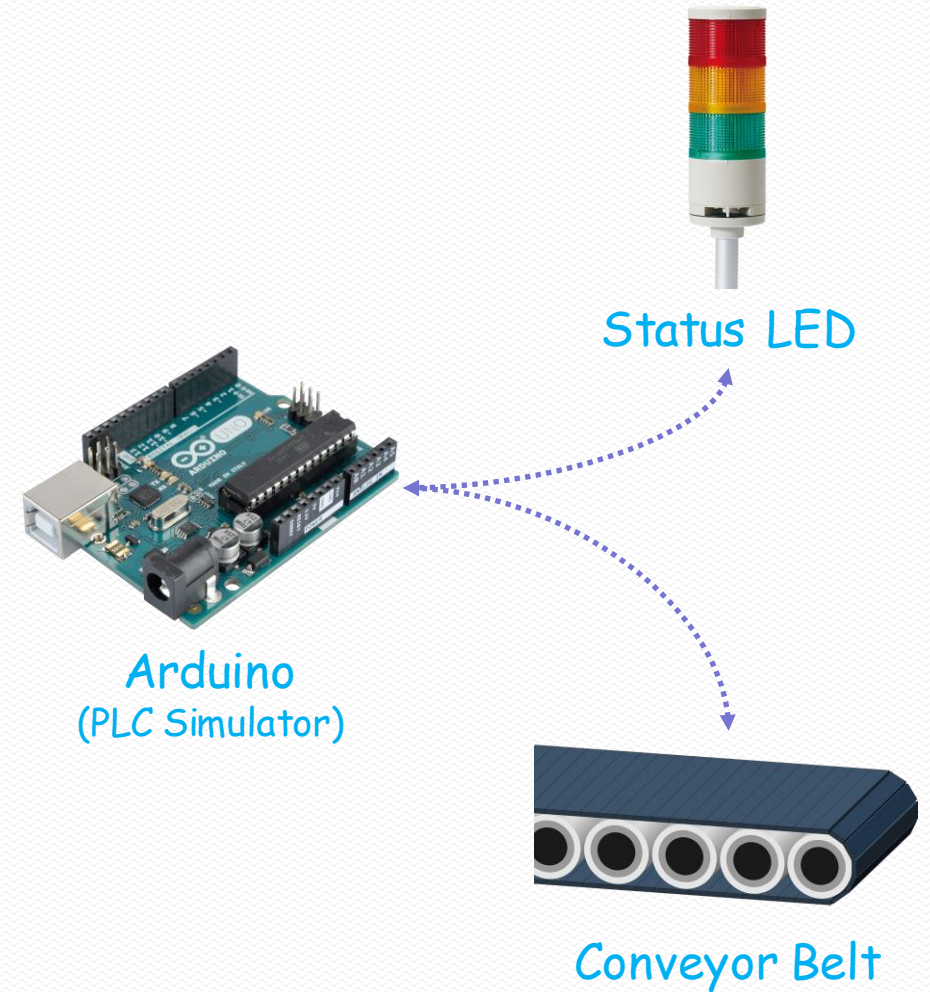
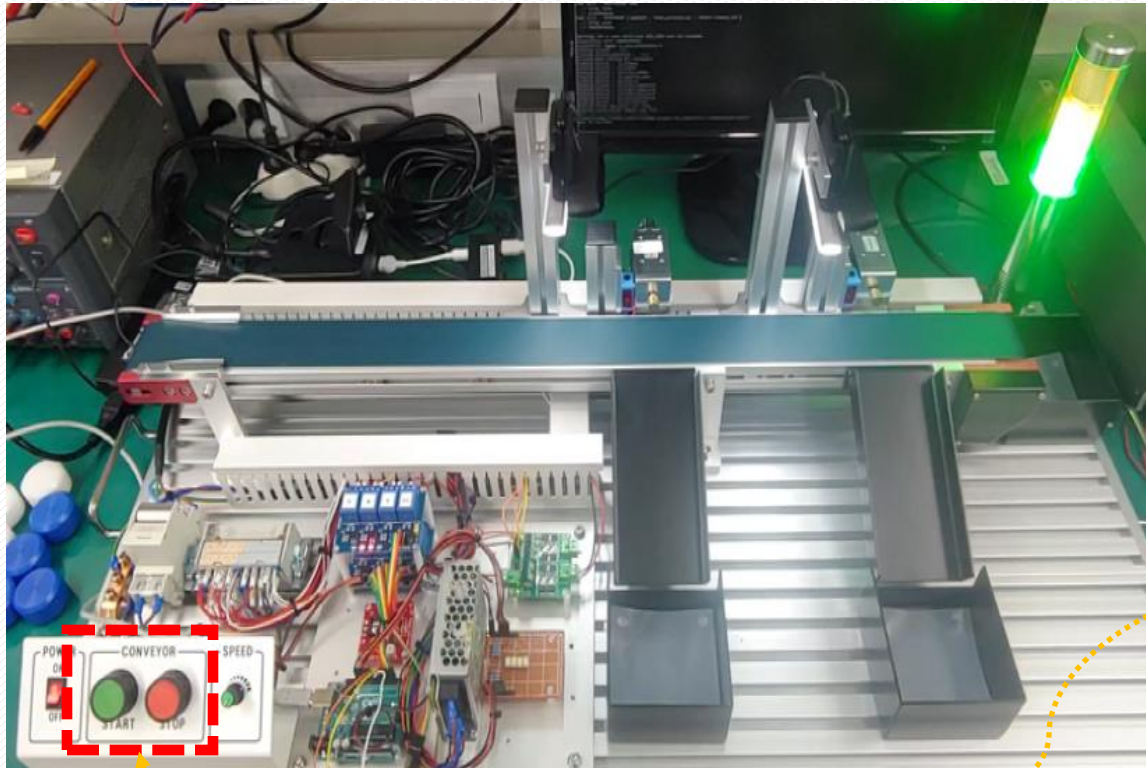


ETC Scenario



ETC Scenario

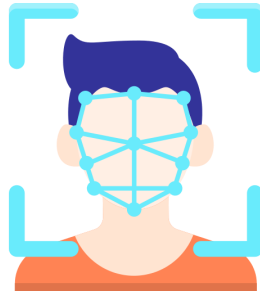
- Start/Stop button



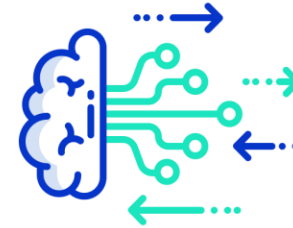
What to do on the Smart Factory



How do you prepare dataset?



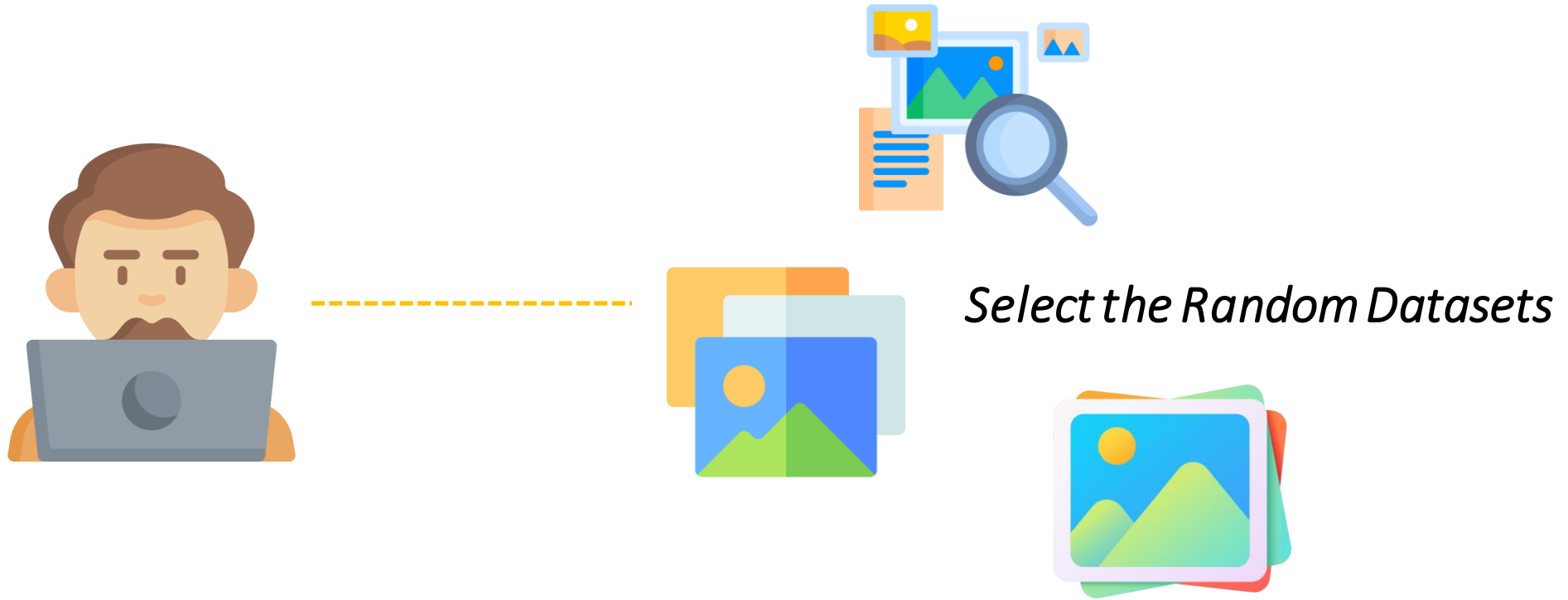
What do you want to recognize?



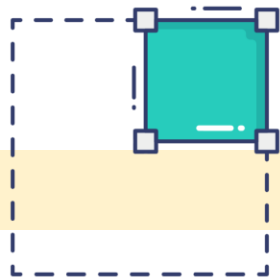
Which models do you want to use?

- Image Classification
- Object Detection
- Segmentation

Planning



Gathering Dataset



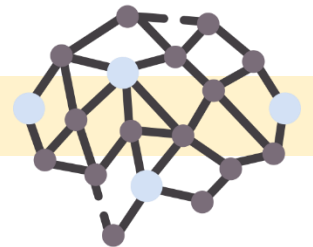
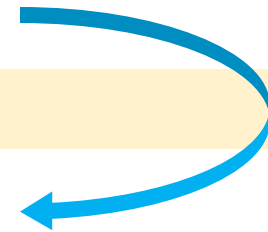
Reduce Image Size



Upload Images



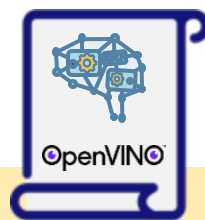
OTX Training



Training using OTX

OpenVINO™ Training Extensions

Export Model



OpenVINO IR File
*.XML, *.BIN

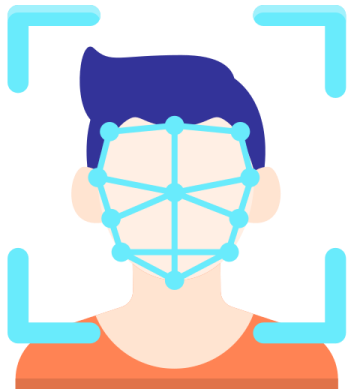


*Localhost
Machine*

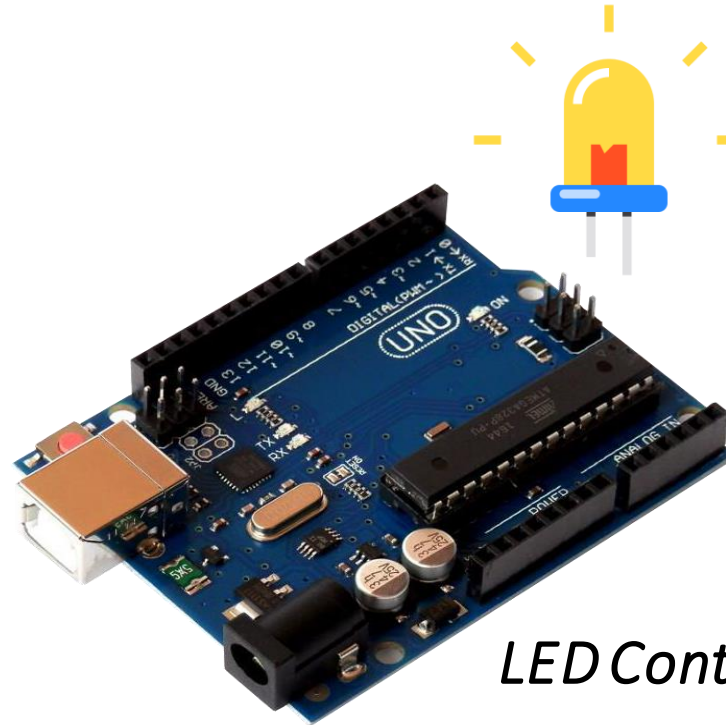


Inferencing Result

Inferencing on the Localhost



Inferencing Result

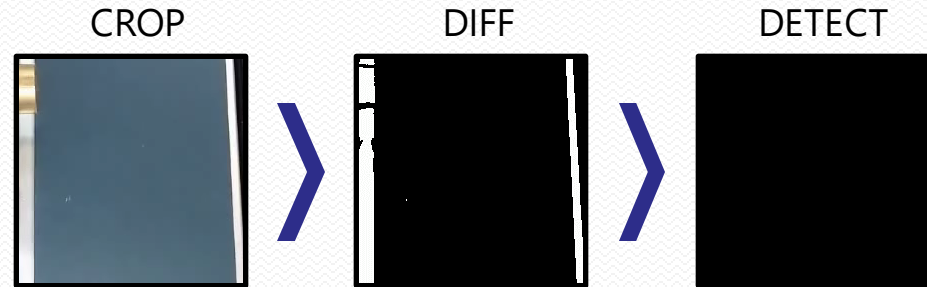
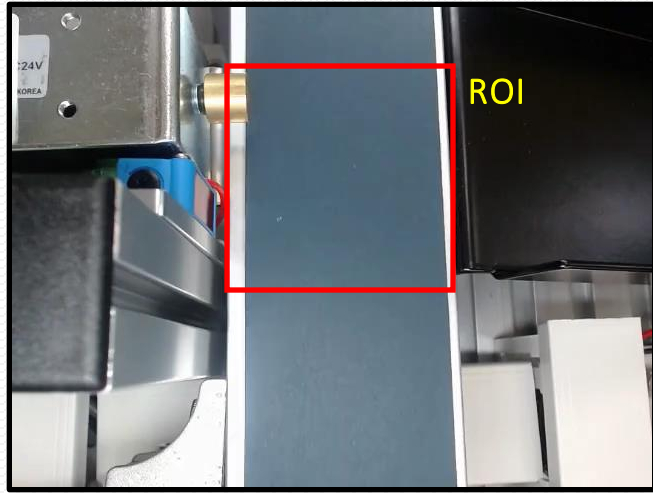


LED Control with Arduino

Utilize Arduino LED with Inferencing Result

Using Tools

Detailed Flow

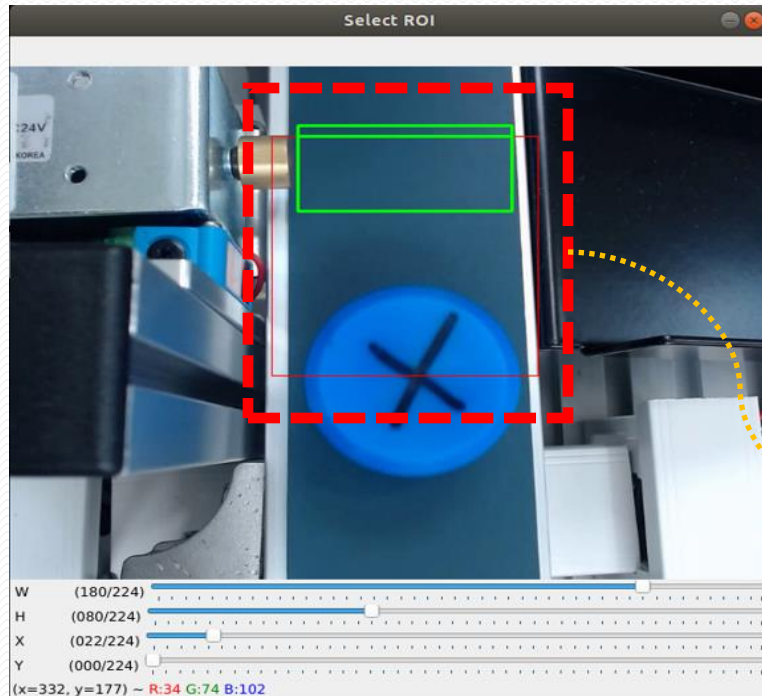


- *Crop the frame per selected ROI*
- *Calculate the frame difference*
 - Apply custom threshold with brightness value
- *Choose the best frames to pass*

Select ROI to Detect Motion

Python Tool (*iotdemo-motion-detector*)

*iotdemo-motion-detector -l
./resource/factory/conveyor.mp4*



Pre-Implemented Tool

Python Main (*factory.py*)

Detect the Frame (each Thread)

- *detected* = *det.detect(frame)*

if detected is None:
continue

Enqueue (each Thread)

- *q.put(('VIDEO: Cam1 detected', *detected*))*

- *q.put(('VIDEO: Cam2 detected', *detected*))*

```
[default]
inverted = False
flipped = False
top_margin = 10
threshold = 127
top_ratio = 0.6
mid_ratio = 0.4
skip_time = 0.099
roi_top_left = (12, 0)
roi_bottom_right = (212, 90)
crop_top_left = (0, 10)
crop_bottom_right = (224, 234)
```

motion.cfg





Cropped Image

Python API for Motion Detect

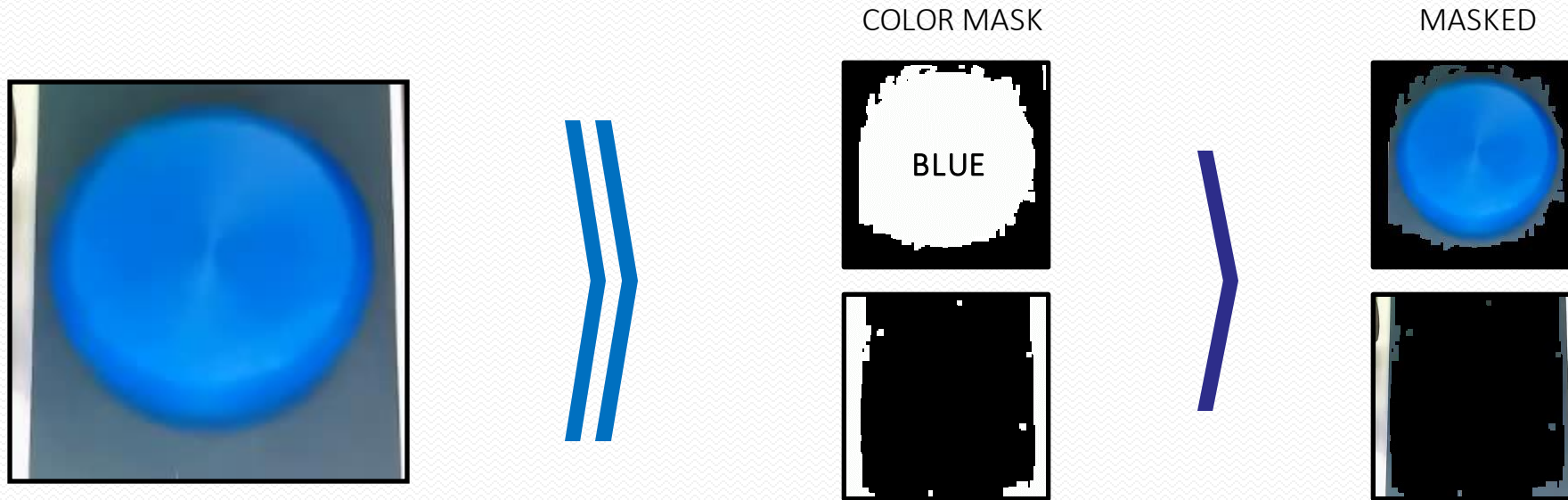
- Refer to the file below

[<root_dir>/iotdemo/motion/motion_detector.py](#)

- How to use the python API?

- Import
- Load “motion.cfg”  [Method: load_preset](#)
- Detect the frame  [Method: detect](#)

Color Detection Flow



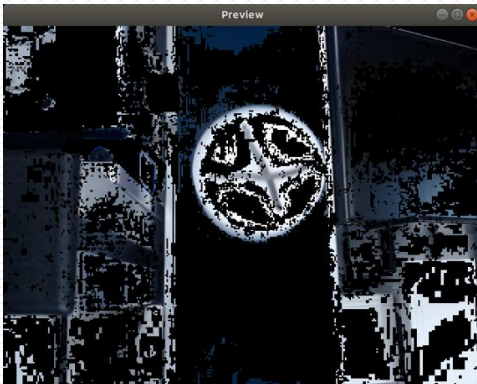
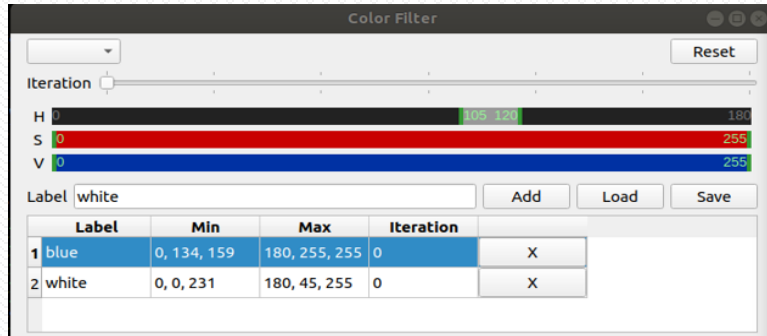
- *Apply color masks to the frames*
 - Using the HSV color model to specify the color position and color "purity"
- *Count the masked pixels and predict colors*

Color Detection

Python Tool (*iotdemo-color-detector*)

iotdemo-color-detector

./resource/factory/conveyor.mp4



Python Main (*factory.py*)

Color Detect (Thread 2)

- *predict = color.detect(detected)*

Get the Predict Ratio (Thread 2)

- *name, ratio = predict[0]*

*ratio = ratio * 100*

Print the Predict Ratio (Thread 2)

- *print(f"{name}: {ratio:.2f}")*

Python API for Color Detect

- Refer to the file below

[<root_dir>/iotdemo/color/color_detector.py](#)

- How to use the python API?

- Import

- Load "color.cfg"

- Detect the frame

- What command send to Arduino?

Method: load_preset

Method: detect

predict[0] -> (name, ratio)

In-queue with name as 'PUSH', data as 2



THANK YOU

Factory setup 방법

1. On
2. Camera check 1개씩 순차적으로 각각 확인 1,2/3,4 번 으로 잡혔는지 확인 필요함. : `ls -al /dev/video`
3. Intel02/class02/smart_factory로 이동
4. 가상환경 진입 : `source sf_env/bin/activates`
5. `iotdemo-motion-detector -h` 로 옵션확인
6. `iotdemo-motion-detector -c 2` 설정
7. 뚜껑놓고 Camera 1설정
8. `cp` 이용해서 `motion.cfg` resource로 copy후 `sync`실행
9. `iotdemo-color-detector /dev/video4` 실행후 `white, blue add+save`
10. `cp color.cfg smart_factory/resources/`로 copy후에 `sync`실행
11. Aduino mega/uno확인후 `python` 파일 수정
12. 수정을 위해 아두이노 권한 변경 `ls /dev/ttyACM0` → `aduno`로 이동
13. `Sudo chmod a+rw /dev/ttyACM0`
14. Bin파일 삭제 `rm -rf ./bin` `rm -rf ./build`
15. `make init` → `make build` → `make flash`
16. 동작확인 → `factory_ref.py`
17. Vi `factory_ref.py`에서 `video`를 `mp4`로 바꾸는건 2개의 threa에서 `/dev/video2` , `/dev/video4`로 수정