IAIA Final Project

ShoeBot : automated shoes organizing system

2023.06.14

Prof. Young-keun Kim

21700435 Tae-Gyun Yang

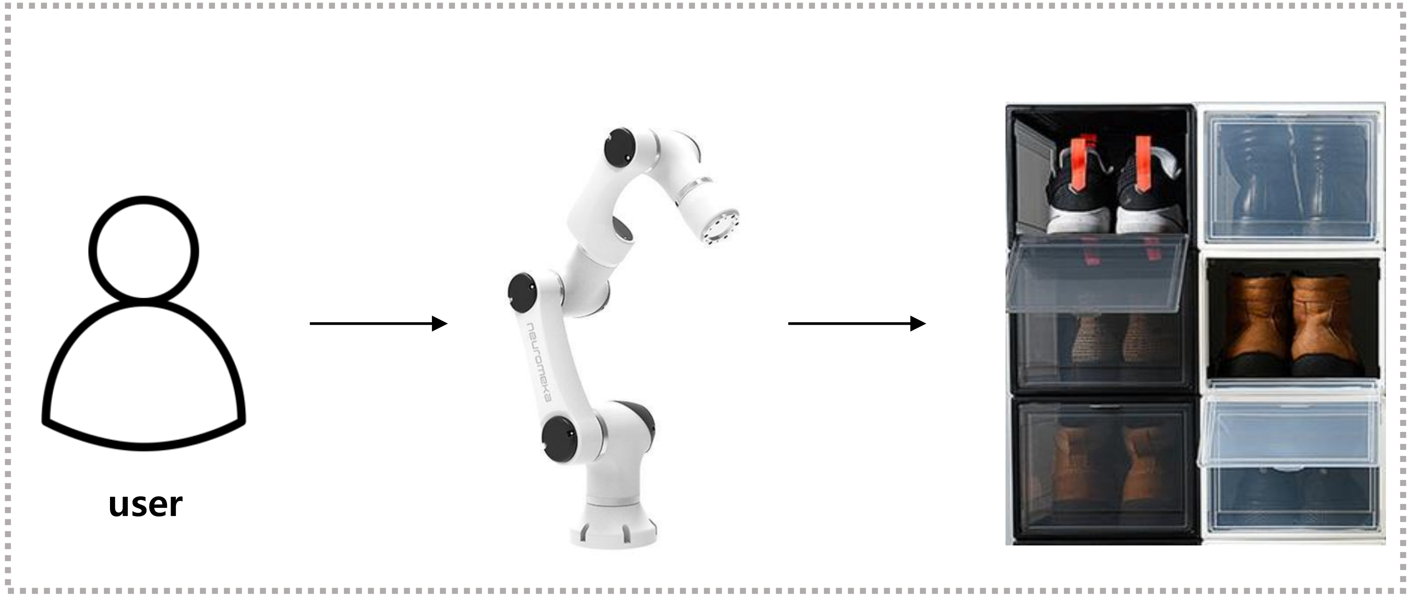
21800773 Min-Woong Han

21901057 Jun-Young Choi

▣ Problem definition

Background and proposal

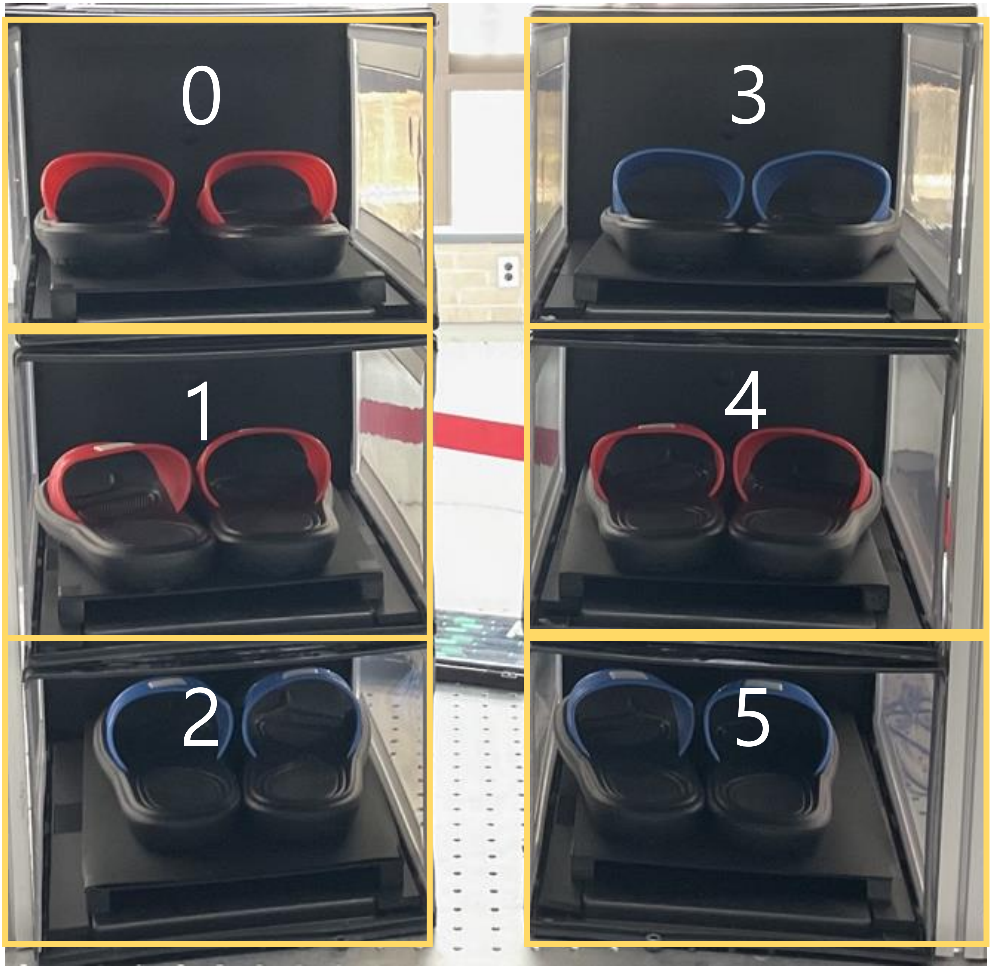
[source : A shoe rack in a lap]



Implementation of automated shoes organizing system based on robot arm

▣ System architecture

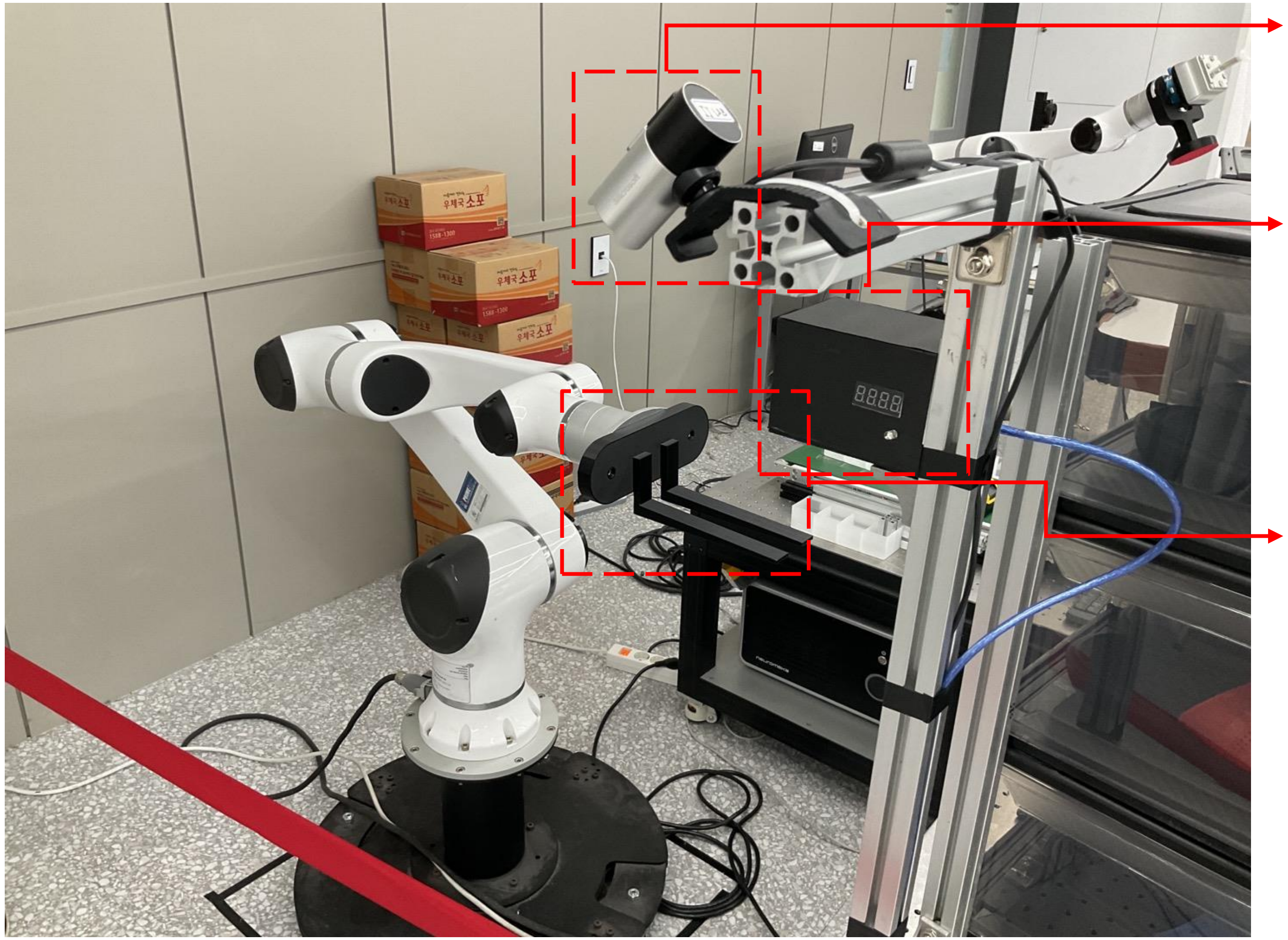
Hardware architecture



- Specify index for each shoe rack
- Configure the software to work with assigned indexes as well
- To continuously update the presence or absence of shoes in the shoebox corresponding to each index

▣ System architecture

Hardware architecture

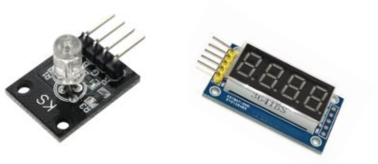


Microsoft life cam studio

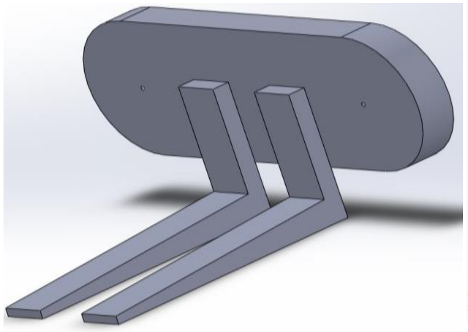
- Utilized to determine user behavior

Visual display using arduino uno

- 4-bit digital tube module
- RGB LED sensor

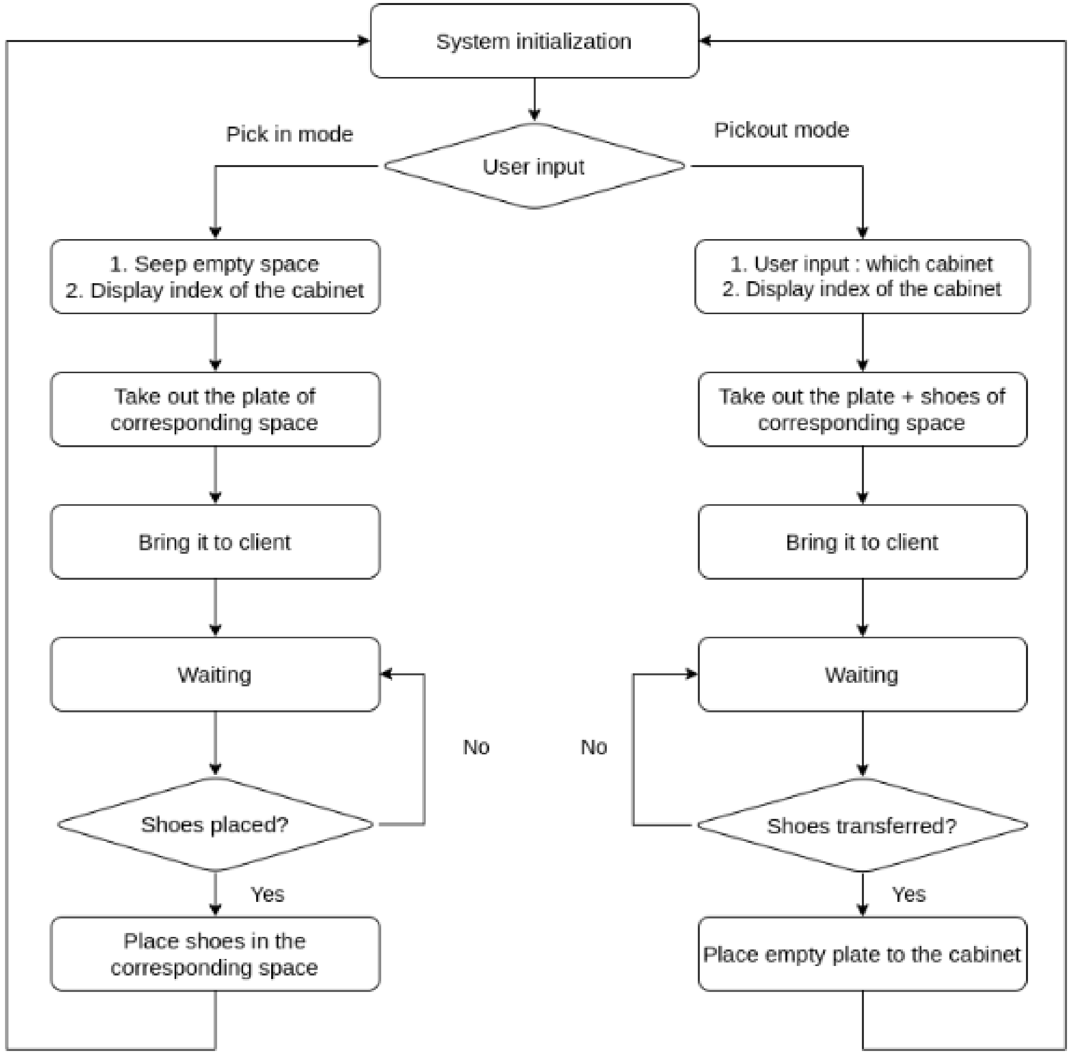


Manufacture of robot end effecter joint

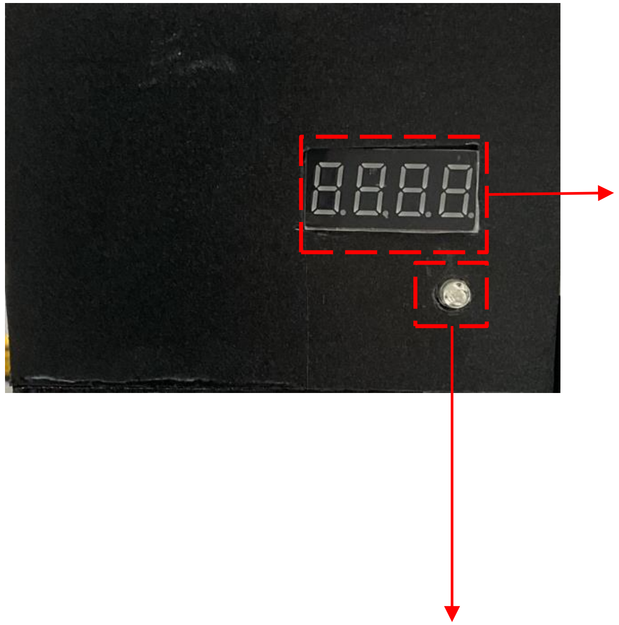


▣ System architecture

Flow chart


System initialization
Pick in mode
User input
Pickout mode
1. Seep empty space
2. Display index of the cabinet
1. User input : which cabinet
2. Display index of the cabinet
Take out the plate of corresponding space
Take out the plate + shoes of corresponding space
Bring it to client
Bring it to client
Waiting
Waiting
No
No
Shoes placed?
Shoes transferred?
Yes
Yes
Place shoes in the corresponding space
Place empty plate to the cabinet

- Microprocessor behavior during manipulation :



4 bit digital tube module

1) Pickin mode

- Display which cabinet is assigned

2) Pickout mode

- Display which cabinet is selected

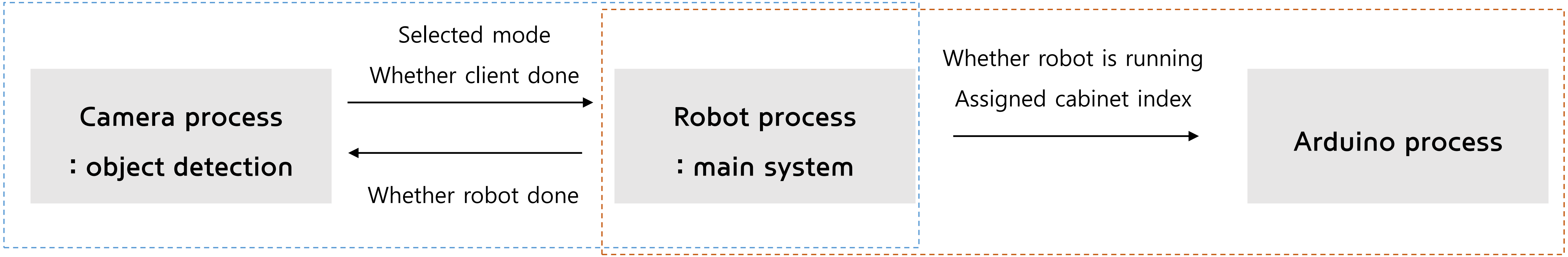
RGB LED module

1) When robot is running : red light as danger alarm

2) When robot is stopped : blue light for safety alarm

▣ System architecture

Software architecture



Camera process

: object detection

Selected mode

Whether client done

Whether robot done

Robot process

: main system

Whether robot is running

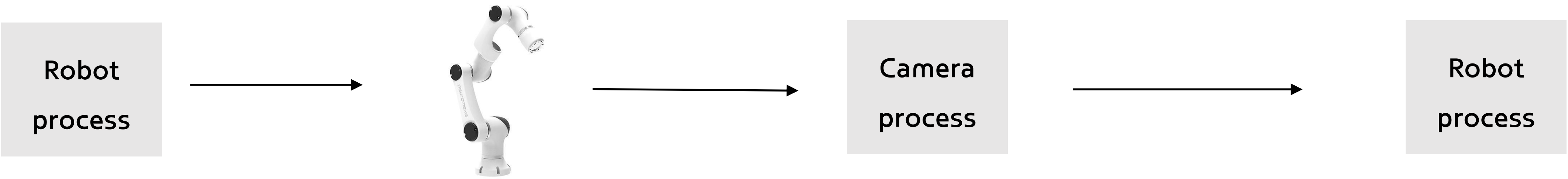
Assigned cabinet index

Arduino process

Publisher – Subscriber ROS internal communication

USB port serial communication

Example : shoes pickin mode



Robot process

Camera process

Robot process

Check empty space

Pickout empty plate

Transfer plate to client

Receive workdone flag

If completed, detect shoes

is started

Receive & wait for a flag whether the user has

finished putting the shoes

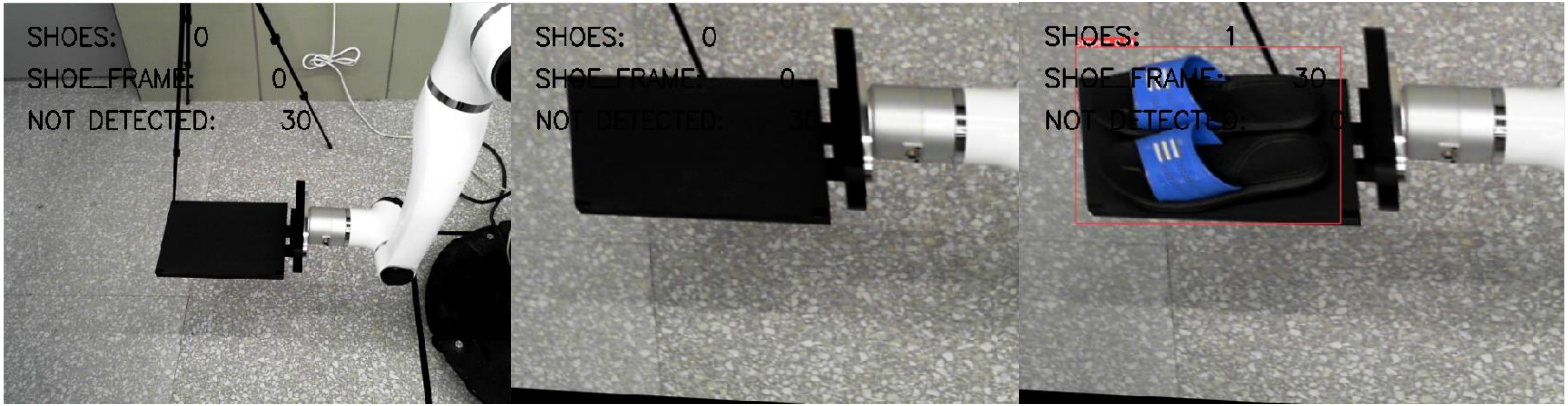
Put the shoes back in the cabinet when finished

▣ Camera

Object detection

- Object detection based on YOLO V5
- Training data : 147 images of shoes
- opencv inverse perspective mapping

  : Change frame to top-down view

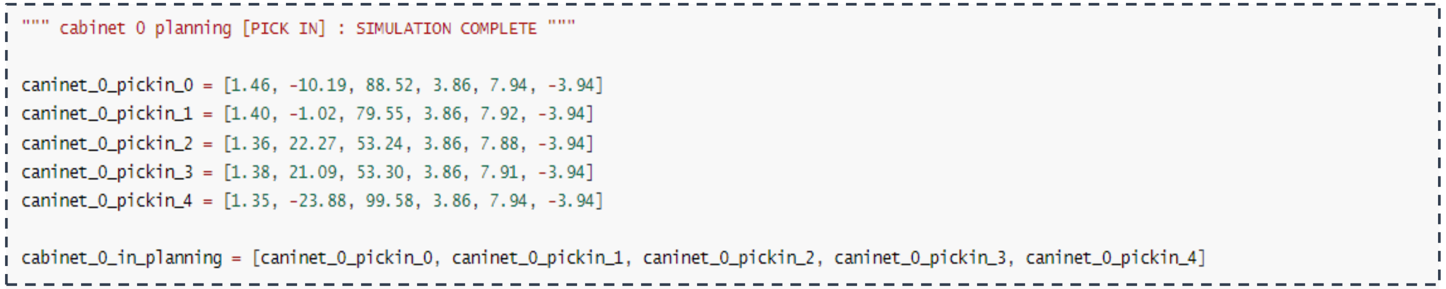


[frame with inverse perspective mapping]

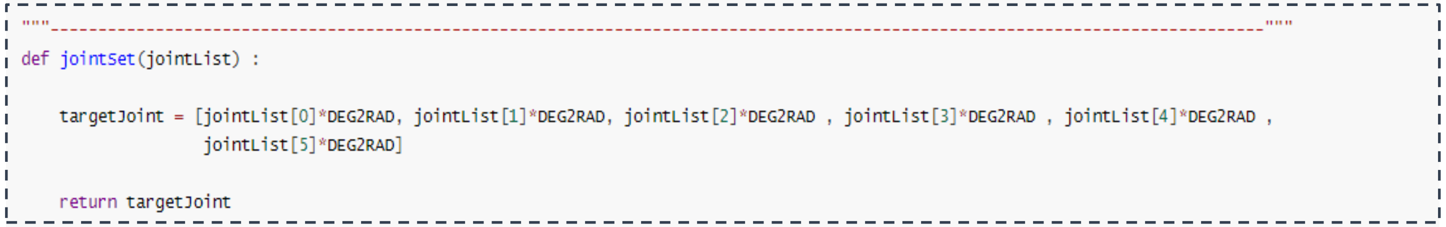
▣ Robot

Robot manipulation

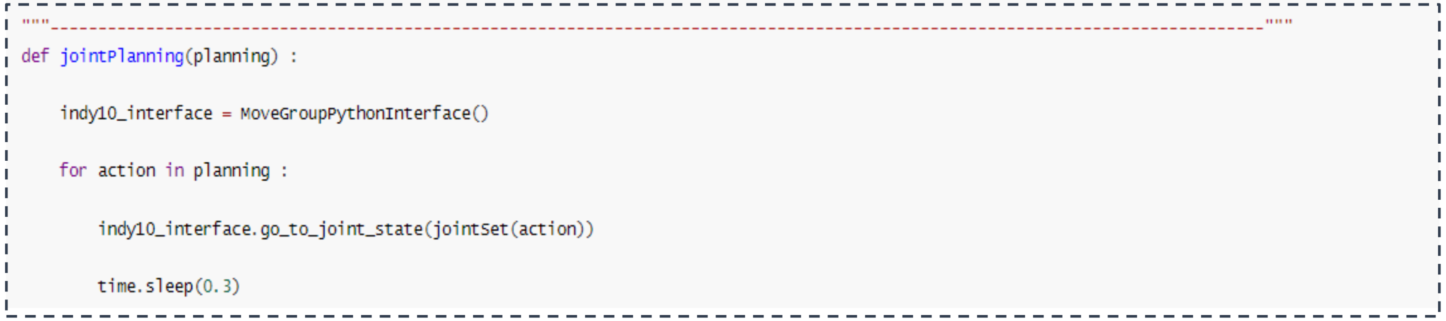
- Instability for path setting with absolute, relative coordinate → determined to use joint command
- Organized system to save the joint commands required for each path planning and sequentially applied
- Example)



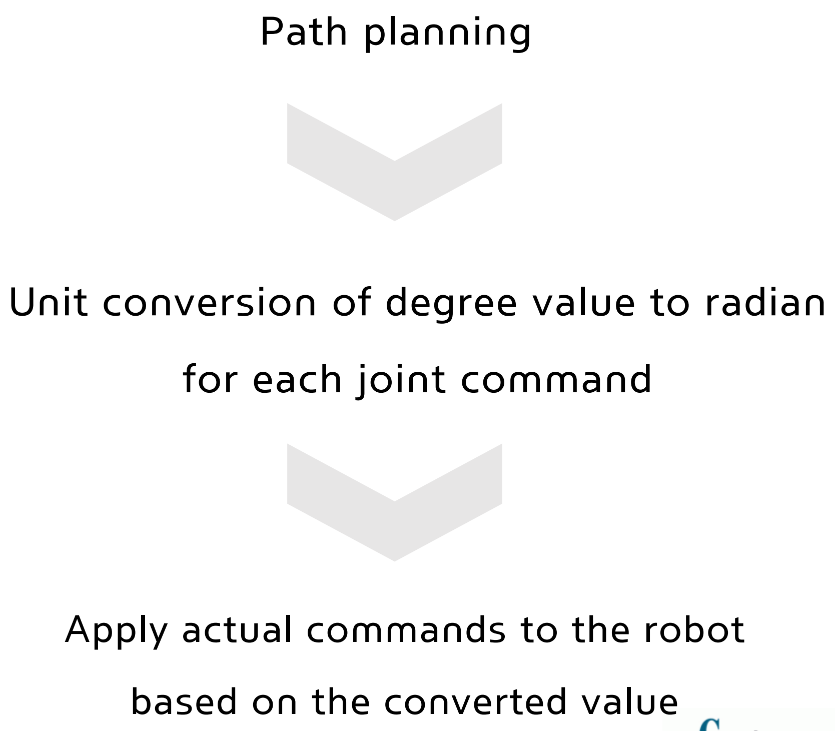
```
""" cabinet 0 planning [PICK IN] : SIMULATION COMPLETE """

caninet_0_pickin_0 = [1.46, -10.19, 88.52, 3.86, 7.94, -3.94]
caninet_0_pickin_1 = [1.40, -1.02, 79.55, 3.86, 7.92, -3.94]
caninet_0_pickin_2 = [1.36, 22.27, 53.24, 3.86, 7.88, -3.94]
caninet_0_pickin_3 = [1.38, 21.09, 53.30, 3.86, 7.91, -3.94]
caninet_0_pickin_4 = [1.35, -23.88, 99.58, 3.86, 7.94, -3.94]

cabinet_0_in_planning = [caninet_0_pickin_0, caninet_0_pickin_1, caninet_0_pickin_2, caninet_0_pickin_3, caninet_0_pickin_4]
```



```
"""------------------------------------------------------------------------------------------------------------------------------"""
def jointSet(jointList) :

    targetJoint = [jointList[0]*DEG2RAD, jointList[1]*DEG2RAD, jointList[2]*DEG2RAD , jointList[3]*DEG2RAD , jointList[4]*DEG2RAD ,
                   jointList[5]*DEG2RAD]

    return targetJoint
```



```
"""------------------------------------------------------------------------------------------------------------------------------"""
def jointPlanning(planning) :

    indy10_interface = MoveGroupPythonInterface()

    for action in planning :

        indy10_interface.go_to_joint_state(jointSet(action))

        time.sleep(0.3)
```



Path planning

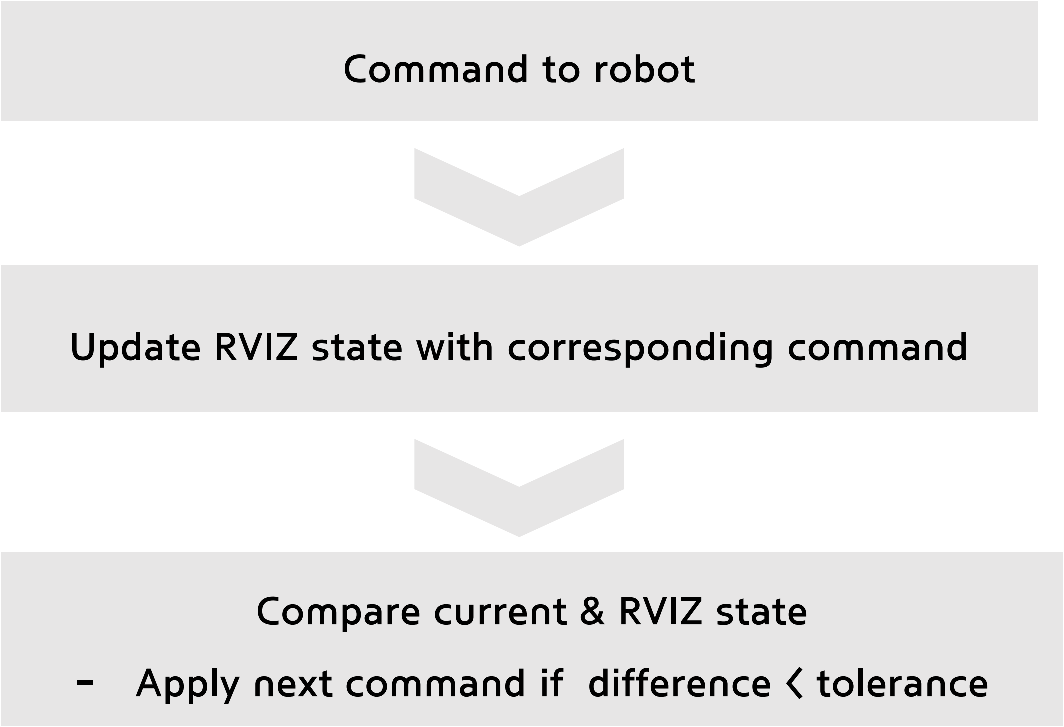
Unit conversion of degree value to radian for each joint command

Apply actual commands to the robot based on the converted value

▣ Robot manipulation

Original manipulation

- Long time consumed for RVIZ update
- Problem with the next command taking too long



Command to robot

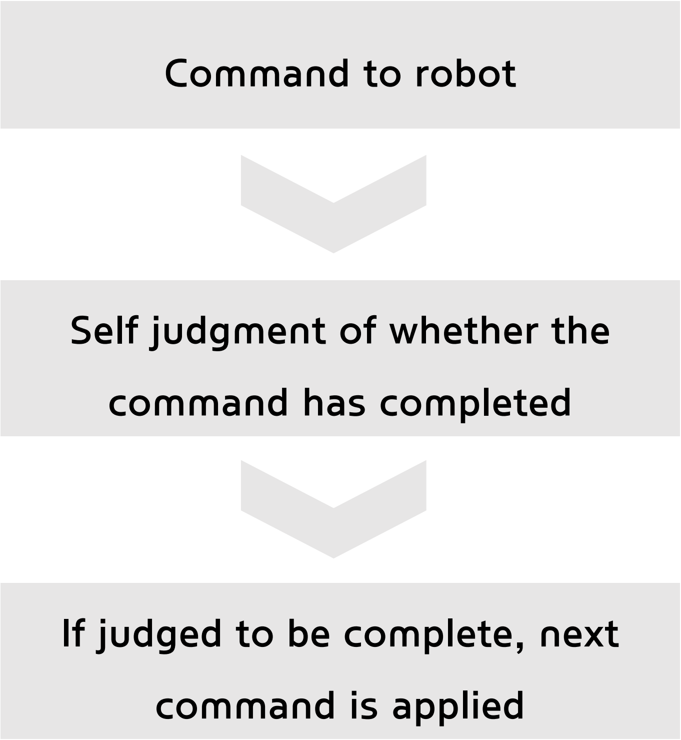
Update RVIZ state with corresponding command

Compare current & RVIZ state

- Apply next command if difference < tolerance

Changed manipulation

- Sequential command applying got faster
- Slowing down problem as ROS running time increases still exists



Command to robot

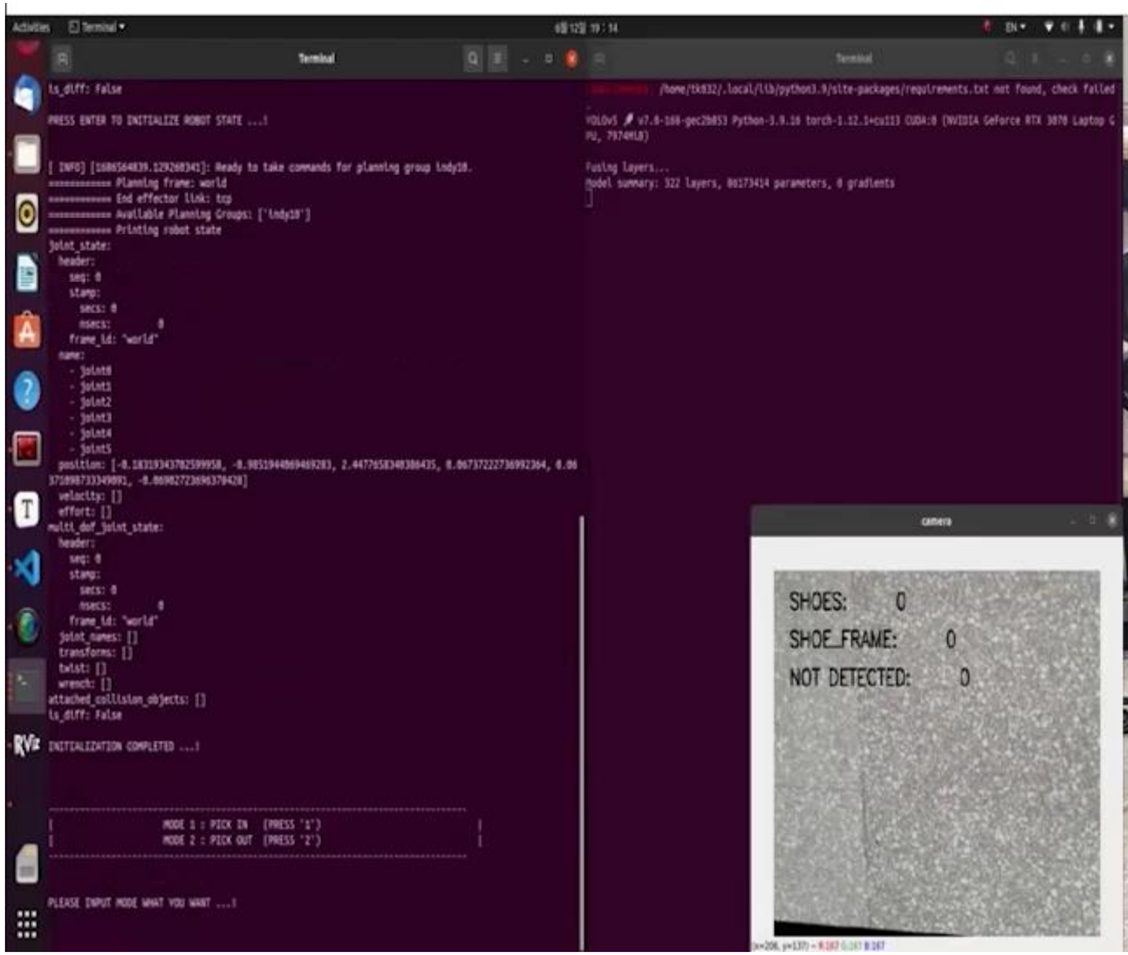
Self judgment of whether the command has completed

If judged to be complete, next command is applied

▣ Result (Demo video)

Pickin mode

Demo video (left : terminal / right : robot manipulation)

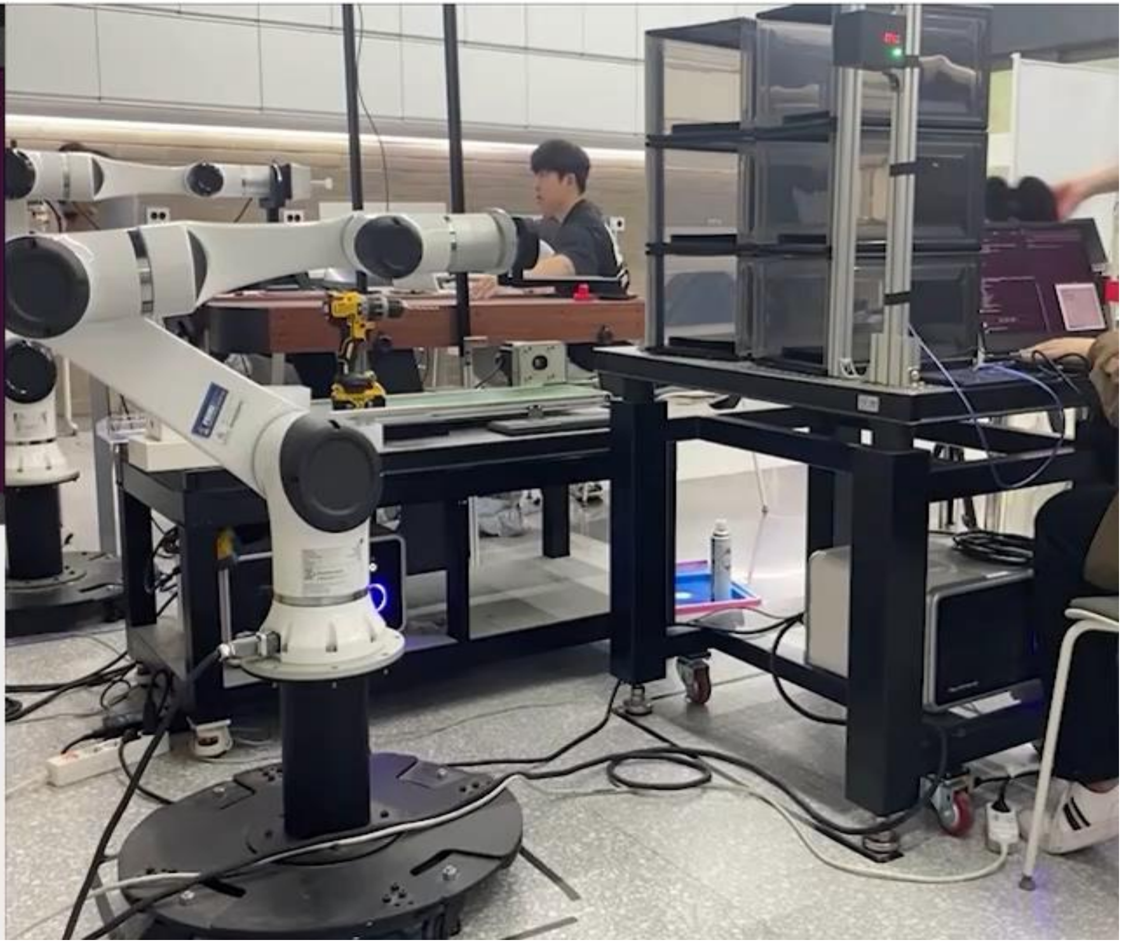


```
is_diff: False

PRESS ENTER TO INITIALIZE ROBOT STATE ...!


[ INFO] [1686564839.129268341]: Ready to take commands for planning group indy10.
============ Planning frame: world
============ End effector link: tcp
============ Available Planning Groups: ['indy10']
============ Printing robot state
joint_state:
  header:
    seq: 0
    stamp:
      secs: 0
      nsecs:         0
    frame_id: "world"
  name:
    - joint0
    - joint1
    - joint2
    - joint3
    - joint4
    - joint5
  position: [-0.18319343702599958, -0.9851944069469283, 2.4477658340380435, 0.06737222736992364, 0.06
371098733349091, -0.06982723696370428]
  velocity: []
  effort: []
multi_dof_joint_state:
  header:
    seq: 0
    stamp:
      secs: 0
      nsecs:         0
    frame_id: "world"
  joint_names: []
  transforms: []
  twist: []
  wrench: []
attached_collision_objects: []
is_diff: False

INITIALIZATION COMPLETED ...!



------------------------------------------------------------------------------
[                   MODE 1 : PICK IN   (PRESS '1')                            ]
[                   MODE 2 : PICK OUT  (PRESS '2')                            ]
------------------------------------------------------------------------------


PLEASE INPUT MODE WHAT YOU WANT ...!
```

```
[illegible] /home/tk032/.local/lib/python3.9/site-packages/requirements.txt not found, check failed
.
YOLOv5 🚀 v7.0-188-gec2b853 Python-3.9.16 torch-1.12.1+cu113 CUDA:0 (NVIDIA GeForce RTX 3070 Laptop G
PU, 7974MiB)

Fusing layers...
Model summary: 322 layers, 86173414 parameters, 0 gradients
```

```
SHOES:        0
SHOE_FRAME:        0
NOT DETECTED:        0
```

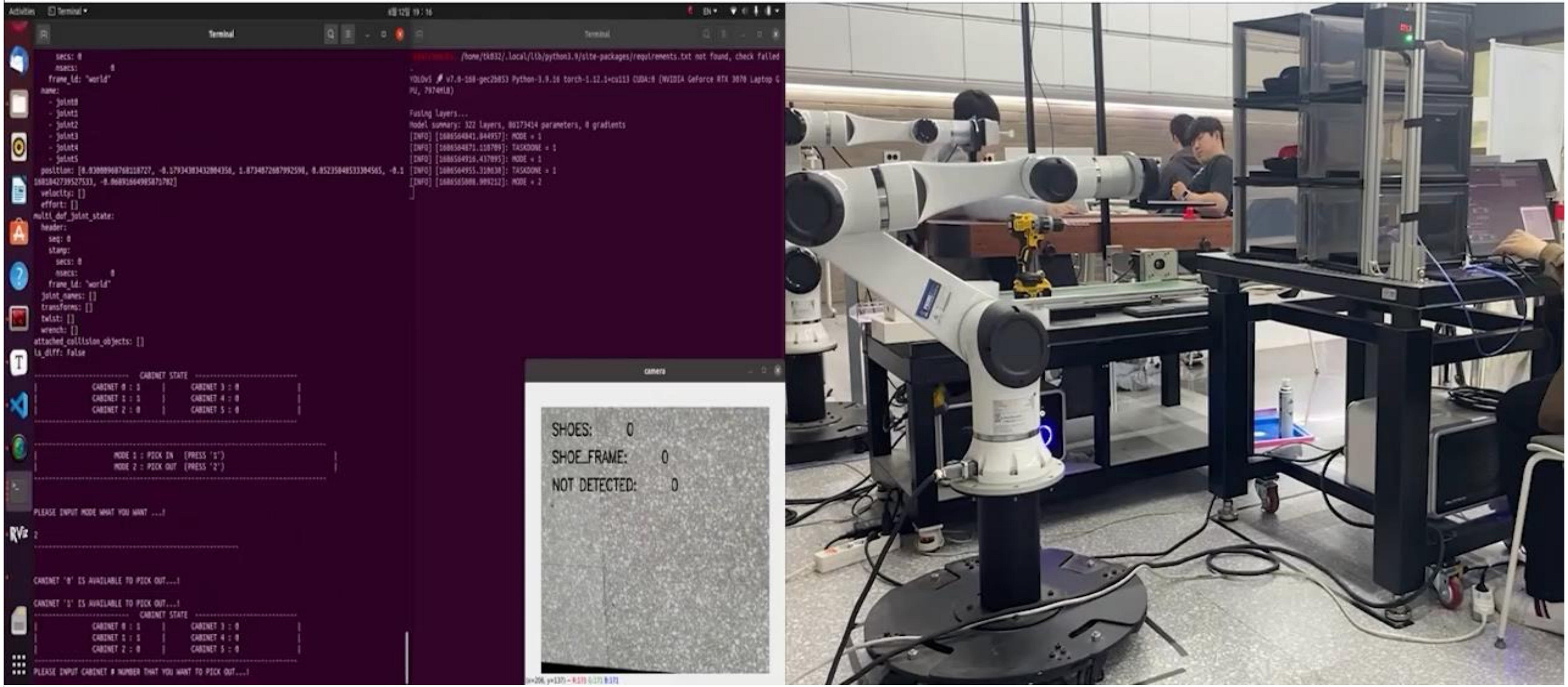
(x=206, y=137) ~ R:187 G:187 B:187



▣ Result (Demo video)

Pickout mode

Demo video (left : terminal / right : robot manipulation)



```
      secs: 0
      nsecs:         0
    frame_id: "world"
  name:
    - joint0
    - joint1
    - joint2
    - joint3
    - joint4
    - joint5
  position: [0.030009687681187 27, -0.17934303432004356, 1.8734872687992598, 0.05235048533384565, -0.1
1681842739527533, -0.06891664985871782]
  velocity: []
  effort: []
multi_dof_joint_state:
  header:
    seq: 0
    stamp:
      secs: 0
      nsecs:         0
    frame_id: "world"
  joint_names: []
  transforms: []
  twist: []
  wrench: []
attached_collision_objects: []
is_diff: False

-------------------------   CABINET STATE   -------------------------
|              CABINET 0 : 1       |       CABINET 3 : 0              |
|              CABINET 1 : 1       |       CABINET 4 : 0              |
|              CABINET 2 : 0       |       CABINET 5 : 0              |
-----------------------------------------------------------------------


-------------------------------------------------------------------------------
|                    MODE 1 : PICK IN    (PRESS '1')                          |
|                    MODE 2 : PICK OUT   (PRESS '2')                          |
-------------------------------------------------------------------------------



PLEASE INPUT MODE WHAT YOU WANT ....!

2
-----------------------------------------------------


CABINET '0' IS AVAILABLE TO PICK OUT....!

CABINET '1' IS AVAILABLE TO PICK OUT....!
-------------------------   CABINET STATE   -------------------------
|              CABINET 0 : 1       |       CABINET 3 : 0              |
|              CABINET 1 : 1       |       CABINET 4 : 0              |
|              CABINET 2 : 0       |       CABINET 5 : 0              |
-----------------------------------------------------------------------
PLEASE INPUT CABINET # NUMBER THAT YOU WANT TO PICK OUT....!
```

```
[illegible] /home/tk032/.local/lib/python3.9/site-packages/requirements.txt not found, check failed
.
YOLOv5 🚀 v7.0-168-gec2b853 Python-3.9.16 torch-1.12.1+cu113 CUDA:0 (NVIDIA GeForce RTX 3070 Laptop G
PU, 7974MiB)

Fusing layers...
Model summary: 322 layers, 86173414 parameters, 0 gradients
[INFO] [1686564841.044957]: MODE = 1
[INFO] [1686564871.110789]: TASKDONE = 1
[INFO] [1686564916.437895]: MODE = 1
[INFO] [1686564955.310638]: TASKDONE = 1
[INFO] [1686565008.909212]: MODE = 2
```

```
SHOES:        0
SHOE_FRAME:      0
NOT DETECTED:      0
```

(x=206, y=137) ~ R:171 G:171 B:171

▣ Result

Performance evaluation

- A total of 5 insertion and removal operations are performed for each compartment analysis is performed
- Analysis of camera recognition accuracy
- Evaluation index : success rate [%]

	Pickin	Pickout	Object detection
cabinet 0	100 [%]	100 [%]	100 [%] Average confidence : 0.93
cabinet 1	100 [%]	100 [%]	
cabinet 2	100 [%]	100 [%]	
cabinet 3	100 [%]	100 [%]	
cabinet 4	100 [%]	100 [%]	
cabinet 5	100 [%]	100 [%]	

- High accuracy joint commands & High accuracy achieved by applying deep learning in static situations

Thank you