RS 5

RAPID side

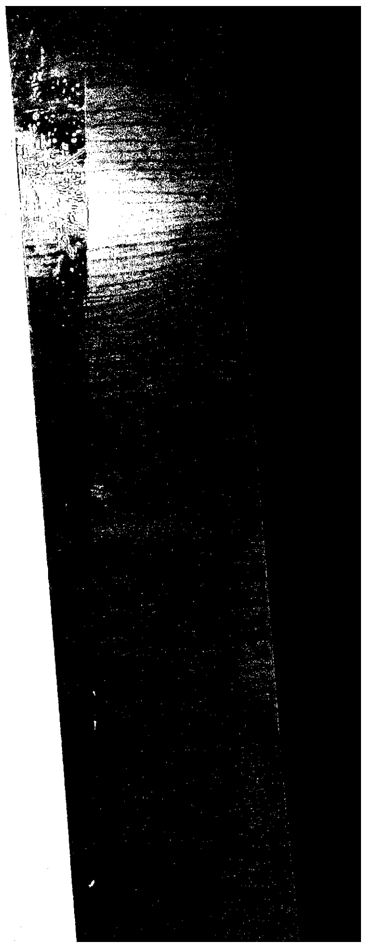
PythonCom

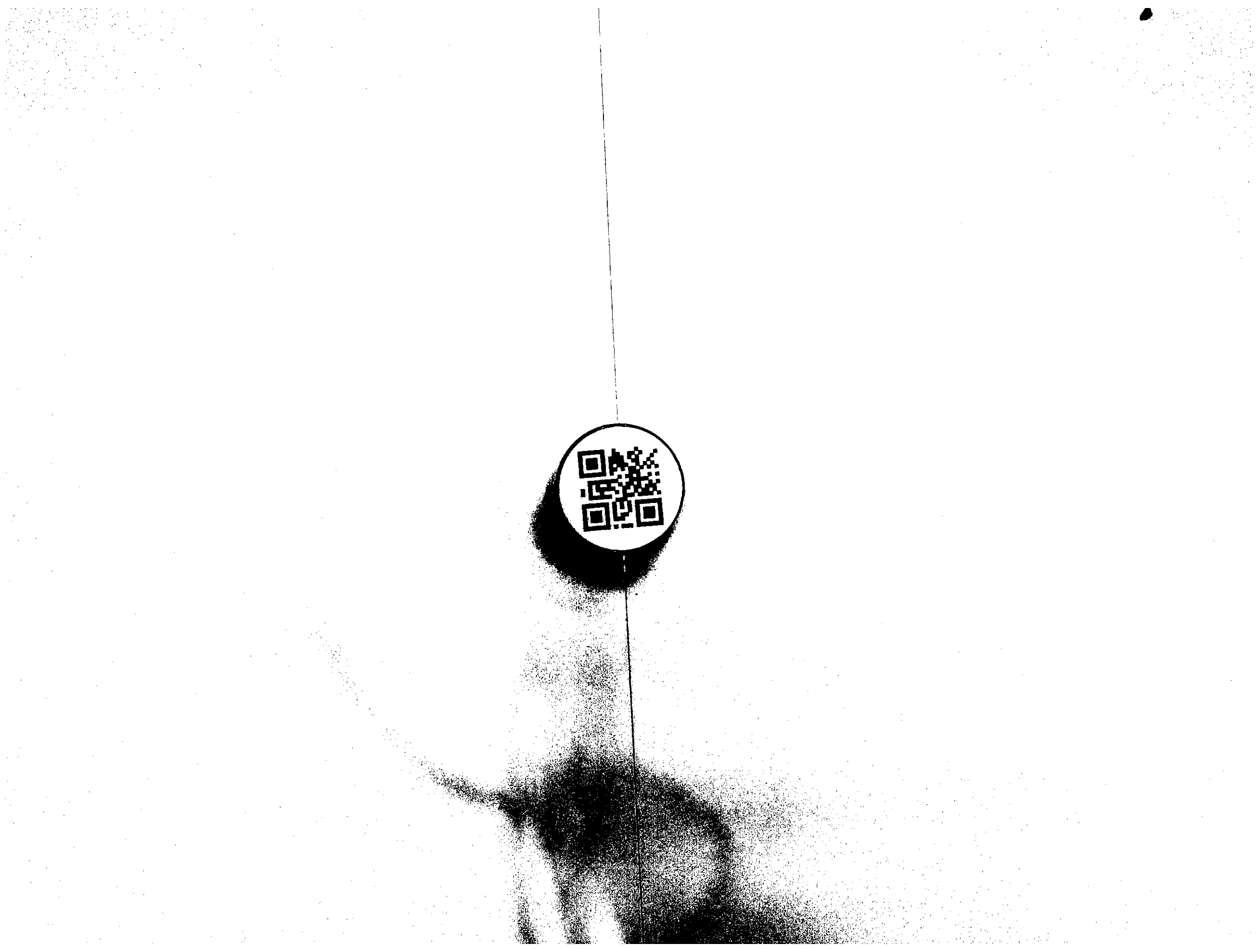
```
MODULE PythonCom
    VAR num WPW:=0;  ! What Python Wants
    VAR num WRD:=0;  ! What RAPID Does
    VAR bool ready_flag:= FALSE;  ! RAPID's ready flag
    VAR bool image_processed:= FALSE;  ! Python's ready flag
    CONST num gripHeight:= 10;
    CONST num safeHeight:= 60;  ! Close-up image height

    ! Where the selected puck is located
    VAR robtarget puck_target := [[0,0,200],[0,1,0,0],[-
1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];

    VAR robtarget point_a := [[0,0,500],[0,1,0,0],[-
1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    VAR robtarget point_b := [[0,0,500],[0,1,0,0],[-
1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    VAR robtarget current_point := [[0,0,500],[0,1,0,0],[-
1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];

    VAR robtarget put_puck_target;  ! Target used to place puck
    VAR num puck_angle;  ! Angle puck should be rotated (optional)

    CONST robtarget middleOfTable:=[[0,0,0],[0,1,0,0],[-
1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];

    ! Overview image position
    VAR robtarget overview:=[[0,0,500],[0,1,0,0],[-
1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];

    TASK PERS wobjdata wobjTableN:=[FALSE,TRUE,"",[[150,-
500,8],[0.707106781,0,0,-0.707106781]],[[0,0,0],[1,0,0,0]]];
    PERS tooldata tGripper:=[TRUE,[[0,0,114.25],[0,0,0,1]],[1,[-
0.095984607,0.082520613,38.69176324],[1,0,0,0],0,0,0]];

    PROC main()

        closeGripper(FALSE);
        MoveJ overview,v100,z10,tGripper\WObj:=wobjTableN;

        ready_flag:=TRUE;
        WHILE TRUE DO
            IF WPW <> 0 THEN
                WRD:=WPW;
                WPW:=0;

                TEST WRD
                    CASE 1:
                    move_to_point_a;
                    CASE 2:
                    move_to_point_b;
                    CASE 3:
                    pick_up_puck;
                    CASE 4:
                    place_puck;
                ENDTEST
            ENDIF
            WRD:=0;
        ENDWHILE
    ENDPROC


    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
    ! Gather/Pick up puck function is needed !
    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
    !------------insert code here------------!
    PROC pick_up_puck()
        MoveL Offs(current_point, 0, 0,
-30),v20,z5,tGripper\WObj:=wobjTableN;
        WaitRob \ZeroSpeed;
        current_point := Offs(current_point, 0, 0, -30);
        MoveL Offs(current_point, 55, 0,
0),v20,z5,tGripper\WObj:=wobjTableN;
        WaitRob \ZeroSpeed;
        current_point := Offs(current_point, 55, 0, 0);
        closeGripper(TRUE);
        MoveL Offs(current_point, 0, 0,
90),v50,z5,tGripper\WObj:=wobjTableN;
        WaitRob \ZeroSpeed;
        current_point := Offs(current_point, 0, 0, 90);
        ready_flag := TRUE;
    ENDPROC

    PROC place_puck()
        MoveL Offs(middleOfTable, 0, 0,
100),v50,z5,tGripper\WObj:=wobjTableN;
        WaitRob \ZeroSpeed;
        MoveL Offs(middleOfTable, 0, 0,
10),v50,z5,tGripper\WObj:=wobjTableN;
        WaitRob \ZeroSpeed;
        closeGripper(FALSE);
        MoveL overview,v50,z5,tGripper\WObj:=wobjTableN;
        WaitRob \ZeroSpeed;
        current_point := overview;
        ready_flag := TRUE;
    ENDPROC

    PROC move_to_point_a()
        MoveL point_a,v50,z5,tGripper\WObj:=wobjTableN;
        WaitRob \ZeroSpeed;
        current_point := point_a;
        ready_flag := TRUE;
    ENDPROC

    PROC move_to_point_b()
        MoveL point_b,v50,z5,tGripper\WObj:=wobjTableN;
        WaitRob \ZeroSpeed;
        current_point := point_b;
        ready_flag := TRUE;
    ENDPROC

    ! wait for python to finish processing image
    PROC wait_for_python()
        WHILE NOT image_processed DO
        ENDWHILE
        image_processed:=FALSE;
    ENDPROC

    ! Function to open / close the gripper
    PROC closeGripper(bool state)
        WaitTime 0.1;
        IF state=TRUE THEN
          setDO AirValve1, 1;
          setDO AirValve2, 0;
        ELSEIF state=FALSE THEN
          setDO AirValve1, 0;
          setDO AirValve2, 1;
        ENDIF
        WaitTime 0.2;
    ENDPROC

ENDMODULE
```

Python side

This is where the interesting stuff happens.

Before doing anything, we took pictures like this at different positions with height 500:





Finding the center of the puck in each image for each position gives a list of "mappings" from one coordinate system to the other.

With this information, we crafted an inverse matrix to go from a translation in the image to a corresponding translation in the x,y-plane of the robot.

Using this matrix, we can approximately center the camera above the puck.

After centering, we move down to 150mm above the table, then just do the same process again. In the program, you can see the different inverse matrices for different heights, named `inv500`, `inv150`, `inv40`.

After centering above the puck at 150mm, the robot moves straight down to 40mm above the table and runs the centering twice. This puts the robot in a close enough position to run the `pick_up_puck` routine on the robot.

Picking up the puck is proceeded by `place_puck` which simply just puts the puck in the center of the table.

The code is very messy, but the essence of it is captured in this video:
https://youtu.be/TU6okolpxZQ

Python script

```
from rwsuis import RWS as rws
import numpy as np
import cv2
import time
from pyueye import ueye
from pyzbar.pyzbar import decode

pcMem = ueye.c_mem_p()
memId = ueye.int()
pitch = ueye.INT()
hCam = ueye.HIDS(0)

width = ueye.int(2592)
height = ueye.int(1944)
bpp = ueye.int(24)

ueye.is_InitCamera(hCam, None)
rect_aoi = ueye.IS_RECT()
rect_aoi.s32X = ueye.int(0)
rect_aoi.s32Y = ueye.int(0)
rect_aoi.s32Width = width
rect_aoi.s32Height = height
ueye.is_AOI(hCam, ueye.IS_AOI_IMAGE_SET_AOI, rect_aoi,
ueye.sizeof(rect_aoi))

ueye.is_SetColorMode(hCam, ueye.IS_CM_RGB8_PACKED)
ueye.is_AllocImageMem(hCam, width, height, bpp, pcMem, memId)
ueye.is_AddToSequence(hCam, pcMem, memId)
ueye.is_InquireImageMem(hCam, pcMem, memId, width, height, bpp,
pitch)

focus = ueye.INT(200)
ueye.is_Focus(hCam, ueye.FOC_CMD_SET_MANUAL_FOCUS, focus,
ueye.sizeof(focus))
ueye.is_Focus(hCam, ueye.FOC_CMD_GET_MANUAL_FOCUS, focus,
ueye.sizeof(focus))
ueye.is_CaptureVideo(hCam, ueye.IS_WAIT)


def take_picture():
    arr = [0]
    while len(arr) < 20:
        arr = ueye.get_data(pcMem, width, height, bpp, pitch, False)
    frame = np.reshape(arr, (height.value, width.value,
bpp.value//8))
```

```
    return frame


def find_pucks(image):
    frame = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    _, frame = cv2.threshold(frame, 90, 255, cv2.THRESH_BINARY)
    return decode(frame)


def get_center_of_puck(puck):
    center_x = sum(map(lambda p: p.x, puck.polygon))/4
    center_y = sum(map(lambda p: p.y, puck.polygon))/4
    return (center_x, center_y)


robot = rws.RWS("http://152.94.0.38")

robot.request_mastership()


def wait_for_rapid():
    while robot.get_rapid_variable("ready_flag") == "FALSE":
        time.sleep(0.5)
    robot.set_rapid_variable("ready_flag", "FALSE")


aorb = True


def set_point_a(x, y, z):
    robot.set_robtarget_translation("point_a", [x, y, z])


def set_point_b(x, y, z):
    robot.set_robtarget_translation("point_b", [x, y, z])


def move_to_point_a():
    wait_for_rapid()
    robot.set_rapid_variable("WPW", 1)
    wait_for_rapid()
    robot.set_rapid_variable("ready_flag", "TRUE")


def move_to_point_b():
    wait_for_rapid()
    robot.set_rapid_variable("WPW", 2)
    wait_for_rapid()
    robot.set_rapid_variable("ready_flag", "TRUE")
```

```
def pick_up_puck():
    wait_for_rapid()
    robot.set_rapid_variable("WPW", 3)
    wait_for_rapid()
    robot.set_rapid_variable("ready_flag", "TRUE")


def place_puck():
    wait_for_rapid()
    robot.set_rapid_variable("WPW", 4)
    wait_for_rapid()
    robot.set_rapid_variable("ready_flag", "TRUE")


center = np.matrix([[1296], [972]])

mat500 = np.matrix([[-6, 1019], [1027.75, 12]])
inv500 = np.linalg.inv(mat500) * 200

mat150 = np.matrix([[-6.75, 1213.25], [1214.25, 8.75]])
inv150 = np.linalg.inv(mat150) * 80

mat40 = np.matrix([[-2, 395.25], [395.5, 4.25]])
inv40 = np.linalg.inv(mat40) * 10


def offset_from_center_as_col_vec(x, y):
    return np.matrix([[x], [y]]) - center


def image_to_robot_translation_500(trans):
    return np.matmul(inv500, trans)


def center_in_image_500(x, y):
    return image_to_robot_translation_500(
        -offset_from_center_as_col_vec(x, y)
    )


def image_to_robot_translation_150(trans):
    return np.matmul(inv150, trans)


def center_in_image_150(x, y):
    return image_to_robot_translation_150(
        -offset_from_center_as_col_vec(x, y)
    )
```

```
def image_to_robot_translation_40(trans):
    return np.matmul(inv40, trans)


def center_in_image_40(x, y):
    return image_to_robot_translation_40(
        -offset_from_center_as_col_vec(x, y)
    )


if __name__ == "__main__":
    # move to overview
    set_point_a(0, 0, 500)
    move_to_point_a()

    # take picture, find puck
    img = take_picture()
    pucks = find_pucks(img)

    if len(pucks) < 1:
        print("ERROR: no pucks in image")
        exit(-1)

    puck = pucks[0]

    # find center of puck in image
    (x, y) = get_center_of_puck(puck)

    test = center_in_image_500(x, y)
    x_1 = test[0, 0]
    y_1 = test[1, 0]

    print(x_1)
    print(y_1)

    # move above puck
    set_point_b(x_1, y_1, 500)
    move_to_point_b()

    # move down
    set_point_a(x_1, y_1, 150)
    move_to_point_a()

    img = take_picture()
    pucks = find_pucks(img)

    if len(pucks) < 1:
        print("ERROR: no pucks in image")
        exit(-1)
    puck = pucks[0]

    (x, y) = get_center_of_puck(puck)

    test = center_in_image_150(x, y)
    x_1 = x_1 + test[0, 0]
    y_1 = y_1 + test[1, 0]

    print(x_1)
    print(y_1)

    # center on puck
    set_point_b(x_1, y_1, 150)
    move_to_point_b()

    # move down to 40mm
    set_point_a(x_1, y_1, 40)
    move_to_point_a()

    img = take_picture()
    pucks = find_pucks(img)

    if len(pucks) < 1:
        print("ERROR: no pucks in image")
        exit(-1)

    puck = pucks[0]

    (x, y) = get_center_of_puck(puck)

    test = center_in_image_40(x, y)
    x_1 = x_1 + test[0, 0]
    y_1 = y_1 + test[1, 0]

    print(x_1)
    print(y_1)

    # center at height 40 first time
    set_point_b(x_1, y_1, 40)
    move_to_point_b()

    img = take_picture()
    pucks = find_pucks(img)

    if len(pucks) < 1:
        print("ERROR: no pucks in image")
        exit(-1)

    puck = pucks[0]
    (x, y) = get_center_of_puck(puck)

    test = center_in_image_40(x, y)
    x_1 = x_1 + test[0, 0]
    y_1 = y_1 + test[1, 0]

    print(x_1)
    print(y_1)

    # center at height 40 second time
    set_point_b(x_1, y_1, 40)
    move_to_point_b()

    # pick up and place puck
    pick_up_puck()
    place_puck()

ueye.is_ExitCamera(hCam)
```