

6.Color interaction

1.1 Introduction to gameplay

How to play:

```
1) First close the large program;  
2) Start a command line window as the anti-decryption server and keep running.  
cd ~/dofbot_ws/ # Enter the workspace  
catkin_make # Compile  
source devel/setup.bash # Update system environment  
roslaunch dofbot_info dofbot_server.launch # Start server node  
3) Color interaction gameplay path:  
dofbot_ws/src/dofbot_snake_follow/scripts/Color interaction.ipynb
```

(1) Start the code block and display the interface as shown in the figure. Click the color button to select a color at will. The selected color is displayed in the middle above the image.

(2) When the selected color appears in the field of view, the robot arm will estimate the location of the block based on its area in the image.

(3) When the area becomes smaller, the robotic arm drives forward until the front end completes the grasping action. When the area becomes larger, it retreats and shakes its head in fear.

(4) If the color that appears on the screen is not the selected color, the robot arm moves back and shakes its head.

Note: Use the color block here to control the robotic arm back and forth, and then slowly move away to trigger it. This case requires multiple attempts



1.2 Code design

- Use polygon approximation method to obtain the area of color objects in the field of view

```
for i, cnt in enumerate(contours):
    # Calculate the moments of a polygon
    mm = cv.moments(cnt)
    cx = mm['m10'] / mm['m00']
    cy = mm['m01'] / mm['m00']
    # Calculate the area of the contour
    area = cv.contourArea(cnt)
```

- Obtain the current pose through the correct solution

```
def get_Posture(self):
    '''
    Publish joint angles and obtain positions
    '''
    self.read_joint()
    # Wait for the server to start
    self.client.wait_for_service()
    # Create message package
    request = kinemaricsRequest()
    ...    ...
```

- Estimate location based on area size

```
# Estimate the position of the block based on the camera
distance = 27.05 * math.pow(area, -0.51) - 0.2
# Estimate the position of the block in the base coordinate system
target_dist = distance + self.Posture[1]
```

- Use the inverse solution to find the angle at which each joint needs to rotate

```
def snake_run(self, point_y):
    '''
    Post a position request and obtain the joint rotation angle
    '''
    pass
```

- Anti-solution design

When we play Color interaction, we need to control the end posture of the robotic arm, always facing forward. When obtaining the inverse solution, just write the end posture of the robotic arm.

```
////////// Color interaction //////////  
if (request.tar_z >= 0.2) {  
    x=request.tar_x;  
    y=request.tar_y;  
    z=request.tar_z;  
    Roll= -90;  
}  
Code path dofbot_ws/src/dofbot_info/src/dofbot_server.cpp
```

For detailed codes, please view snake_ctrl.py;snake_move.py;snake_target.py in the dofbot_ws/src/dofbot_snake_follow/scripts