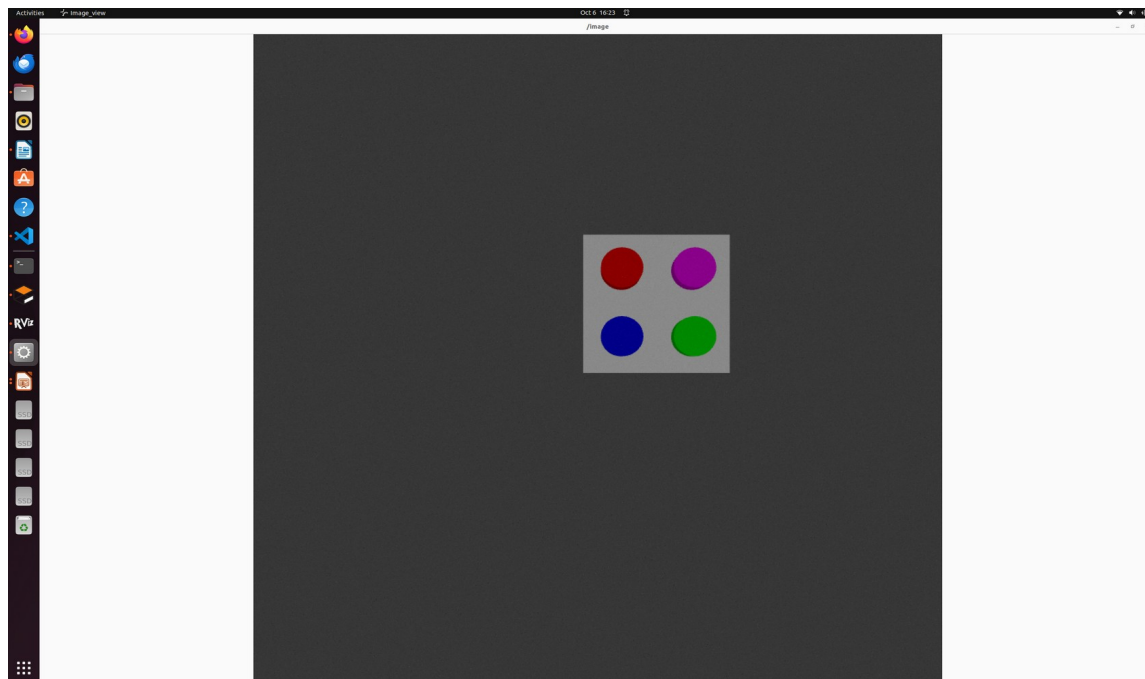
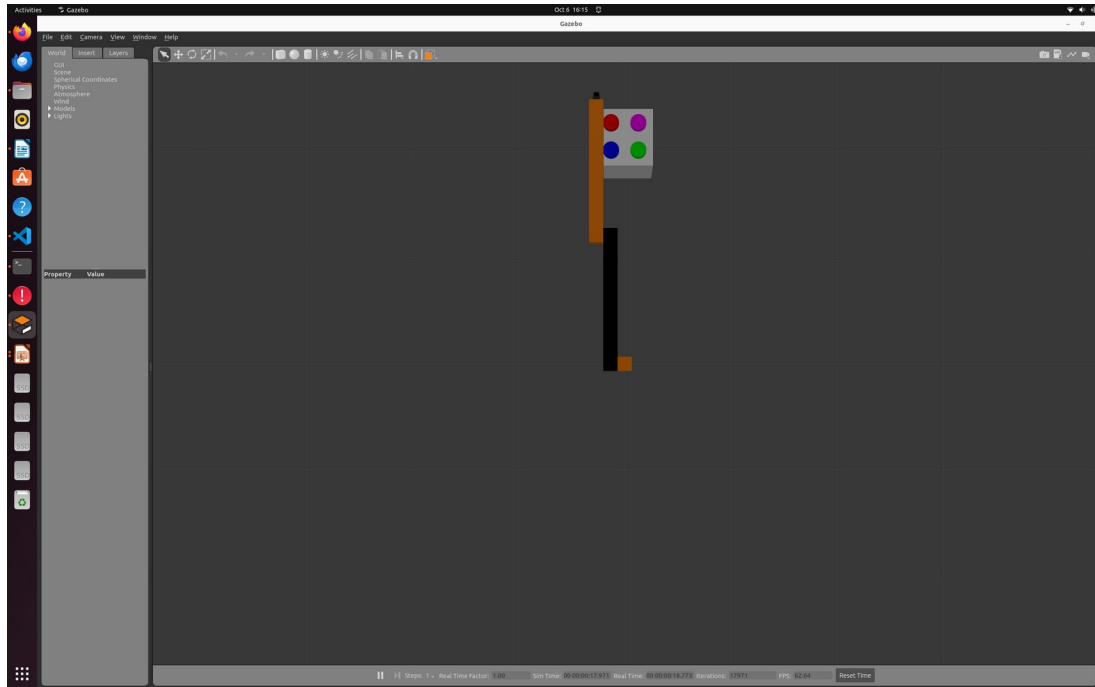


VBM – HW-5

Swati V. Shirke

1. image and detected feature location



2. moved to location = q1, q2 = [0.3,0.3]

ran controller logic without activating velocity control, just to check reference feature values

```
[INFO] [1728366516.099602863] [image_subscriber]: reference features: [(187, 342), (221, 418), (297, 384), (262, 307)]
[INFO] [1728366516.099881217] [image_subscriber]: current features: [(150, 465), (103, 396), (219, 418), (172, 349)]
[INFO] [1728366516.100217741] [image_subscriber]: error: [-37 123 -118 -22 -78 34 -90 42]
[INFO] [1728366516.100805486] [image_subscriber]: image_jacobian: [[-1.00000e+00 0.00000e+00 1.50000e+02 6.97500e+04 -2.25010e+04
4.65000e+02]
```

code – part of test1.py

function for color detection

```
[INFO] [1728366516.584855370] [image_subscriber]: joint_vel: [[-0.08302689 -0.07232028]]

[INFO] [1728366516.582979515] [image_subscriber]: reference features: [(187, 342), (221, 418), (297, 384), (262, 307)]
[INFO] [1728366516.583186601] [image_subscriber]: current features: [(150, 465), (103, 396), (218, 418), (172, 349)]
[INFO] [1728366516.583447354] [image_subscriber]: error: [-37 123 -118 -22 -79 34 -90 42]
[INFO] [1728366516.583890730] [image_subscriber]: image_jacobian: [[-1.00000e+00 0.00000e+00 1.50000e+02 6.97500e+04 -2.25010e+04
4.65000e+02]
[ 0.00000e+00 -1.00000e+00 -1.50000e+02]
[-1.00000e+00 0.00000e+00 -1.00000e+00]
[ 3.96000e+02]
[ 0.00000e+00 -1.00000e+00 -1.00000e+00]
[-1.03000e+02]
[-1.00000e+00 0.00000e+00 -1.00000e+00]
[ 4.18000e+02]
[ 0.00000e+00 -1.00000e+00 -1.00000e+00]
[-2.18000e+02]
[-1.00000e+00 0.00000e+00 -1.00000e+00]
[ 3.49000e+02]
[ 0.00000e+00 -1.00000e+00 -1.00000e+00]
[-1.72000e+02]]

[INFO] [1728366516.595533649] [1.78]
[ 0. 0.]
[ 0. 0.]
[ 0. 0.]
[ 1. 1.]

[INFO] [1728366516.606810508e-02]
[ 3.06810508e-02]
[ 5.07720880e-04]
```

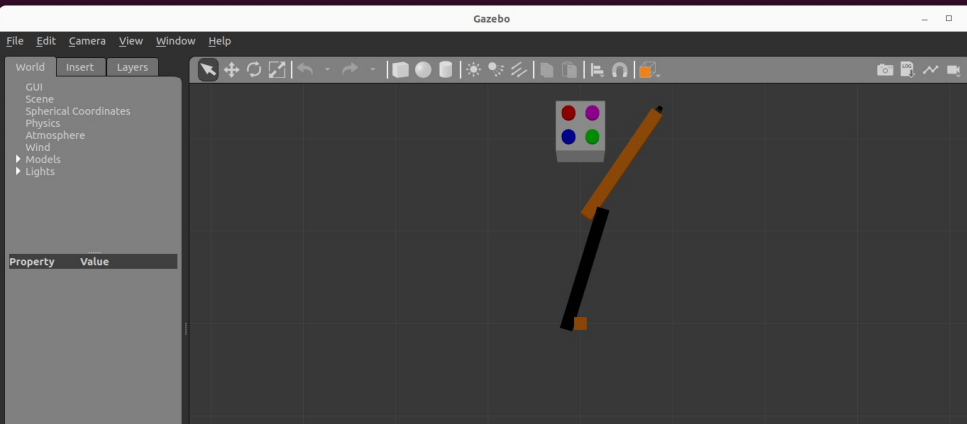
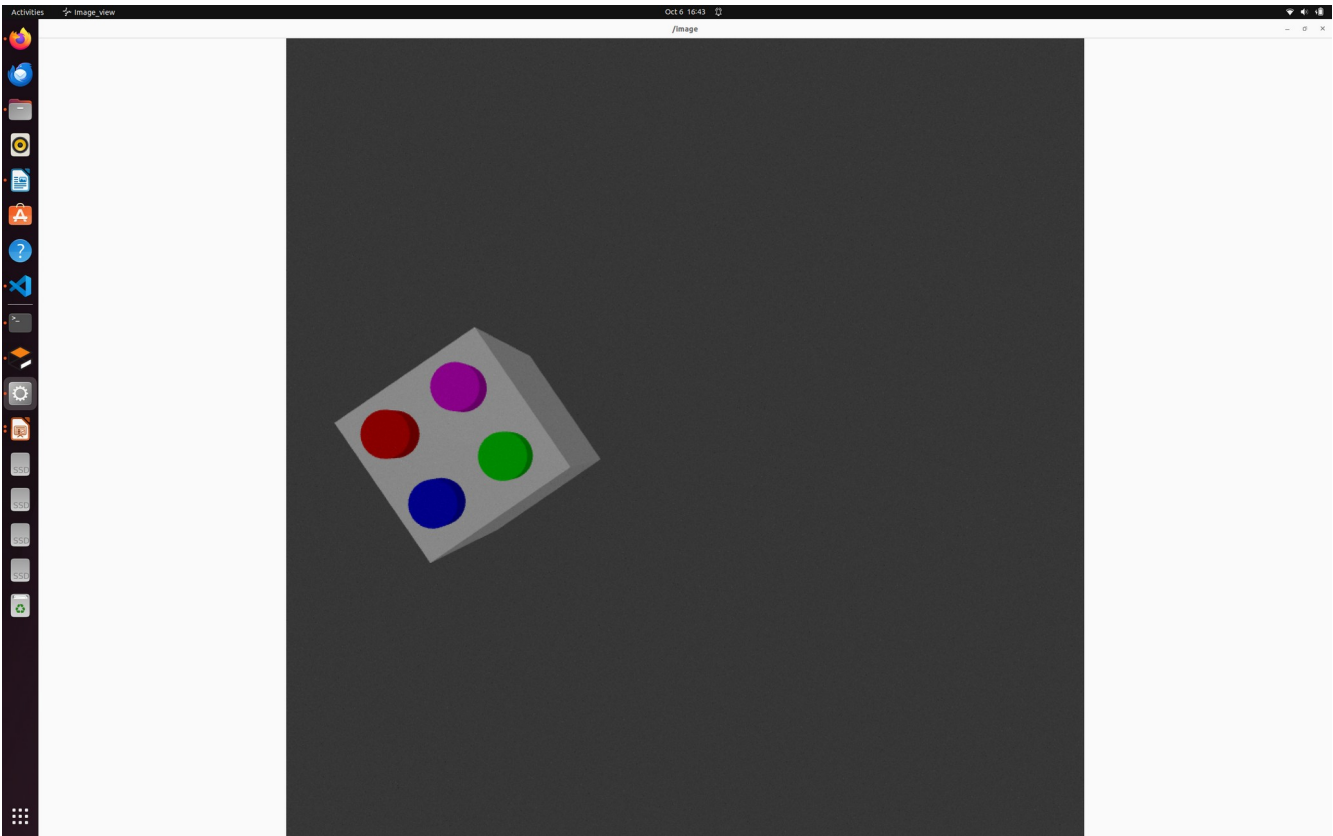


image view output



Visual Studio Code interface showing the `color_detection.py` file in the `src` directory. The file contains a `detect_colors` function that uses OpenCV to detect colors in an image. The function iterates through a list of colors (blue, red, green, pink) and detects them using HSV color space and contour detection. The terminal output shows the execution of the script.

```
def detect_colors(img):
    # The order of the colors is blue, green, red
    #img = cv2.cvtColor(cv2.imread('./image1.jpg'), cv2.COLOR_BGR2RGB)
    center_list = []
    #detect blue
    #define HSV range for blue color
    lower_blue = np.array([100, 150, 50])
    upper_blue = np.array([140, 255, 255])
    result, mask_blue, num_pixels, center = detect_blue(img, lower_blue, upper_blue)
    center_list.append(center)
    # cv2.imshow('blue detected', result)
    # print("num blue pixels: ", num_pixels)
    # print("Center of blue circle: ", center)

    #detect Red
    #define HSV range for blue color
    print("")
    lower_red = np.array([0, 100, 100])
    upper_red = np.array([10, 255, 255])
    result, mask_red, num_pixels, center = detect_blue(img, lower_red, upper_red)
    center_list.append(center)
    # cv2.imshow('red detected', result)
    # print("num red pixels: ", num_pixels)
    # print("Center of red circle: ", center)

    ##detect green
    #define HSV range for green color
    print("")
    lower_green = np.array([40, 100, 100])
    upper_green = np.array([80, 255, 255])
    result, mask_green, num_pixels, center = detect_blue(img, lower_green, upper_green)
    center_list.append(center)
    # cv2.imshow('green detected', result)
    # print("num green pixels: ", num_pixels)
    # print("Center of green circle: ", center)

    ##detect pink
    #define HSV range for pink color
    print("")
    lower_pink = np.array([140, 100, 100])
    upper_pink = np.array([170, 255, 255])
    result, mask_pink, num_pixels, center = detect_blue(img, lower_pink, upper_pink)
    center_list.append(center)
    # cv2.imshow('pink detected', result)
    # print("num pink pixels: ", num_pixels)
    # print("Center of pink circle: ", center)

    return center_list
```

Terminal output:

```
(myenv) swati@swati-Legion-Slim-5-16IRH8:~/VBM/ros2_ws/src/opencv_test_py/opencv_test_py$ python color_detection.py
```

Visual Studio Code interface showing the `color_detection.py` file in the `src` directory. The file contains a `detect_colors` function that uses OpenCV to detect colors in an image. The function iterates through a list of colors (blue, red, green, pink) and detects them using HSV color space and contour detection. The terminal output shows the execution of the script.

```
def detect_blue(img, lower_value, upper_value):
    # Convert the image from BGR to HSV color space
    hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
    # Create a mask for blue color
    mask = cv2.inRange(hsv, lower_value, upper_value)
    # Apply the mask to get the blue regions in the original image
    result = cv2.bitwise_and(img, img, mask=mask)
    # Calculate the number of pixels of the color
    num_pixels = cv2.countNonZero(mask)

    # Find contours of the detected color
    contours, _ = cv2.findContours(mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

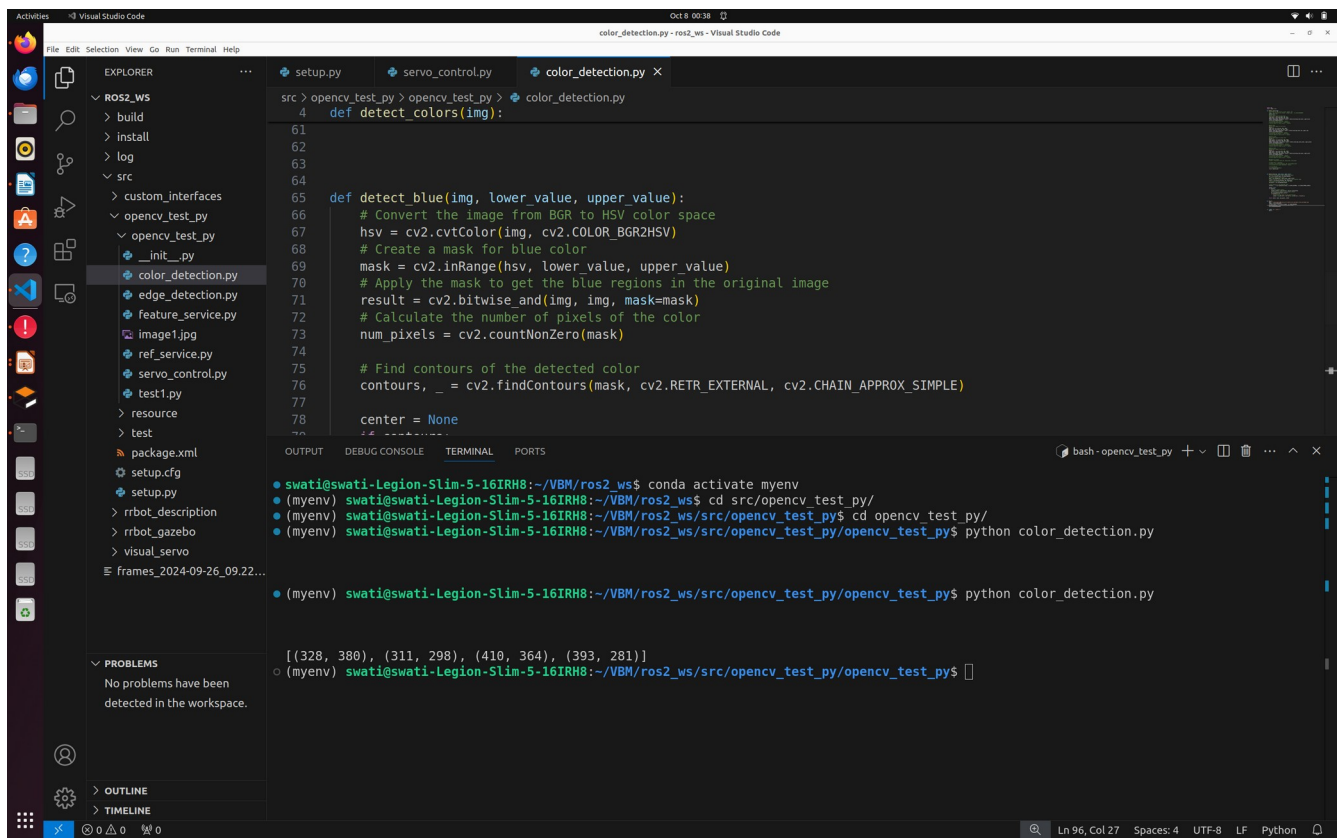
    center = None
    if contours:
        # Get the largest contour
        largest_contour = max(contours, key=cv2.contourArea)
        # Calculate the moments of the largest contour
        M = cv2.moments(largest_contour)
        if M["m00"] != 0:
            # Calculate the center of the detected part
            center = (int(M["m10"] / M["m00"]), int(M["m01"] / M["m00"]))

    return result, mask, num_pixels, center

def main():
    path = '/home/swati/VBM/ros2_ws/src/opencv_test_py/opencv_test_py/image1.jpg'
    image = cv2.imread(path)
    current_frame = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    ref_feature_list = detect_colors(current_frame)
    print(ref_feature_list)
```

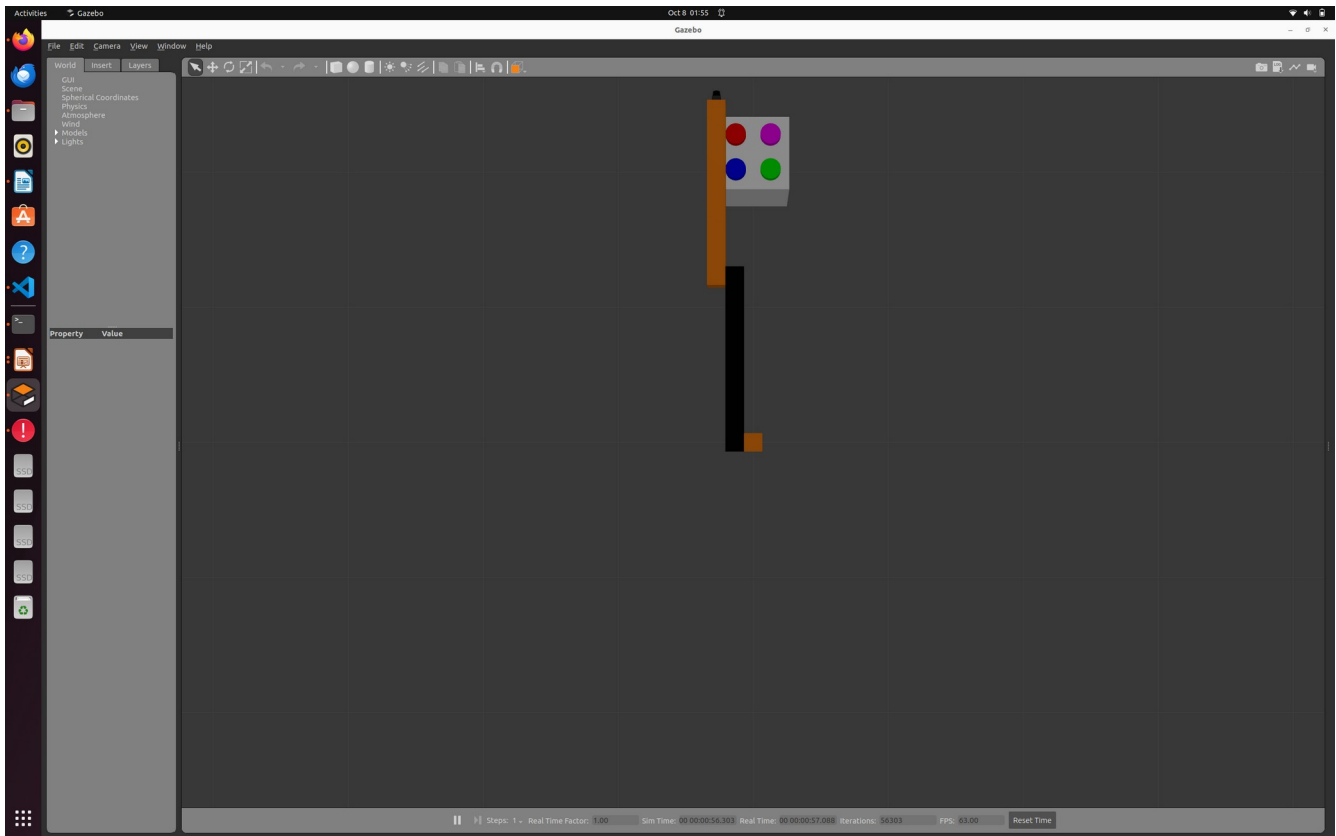
Terminal output:

```
(myenv) swati@swati-Legion-Slim-5-16IRH8:~/VBM/ros2_ws/src/opencv_test_py/opencv_test_py$ python color_detection.py
```

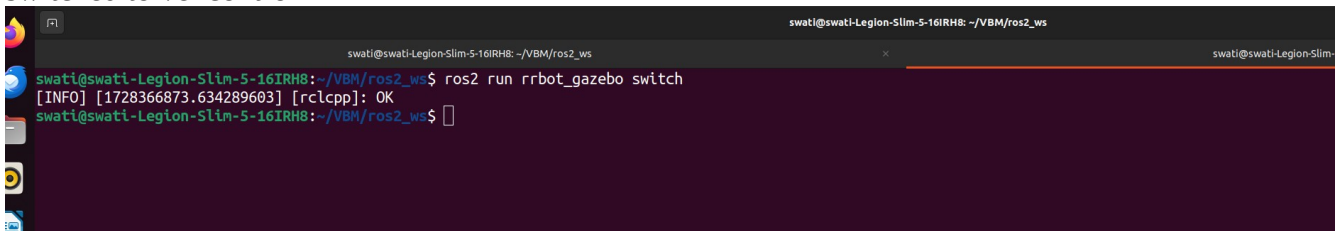


3. Applying visual control

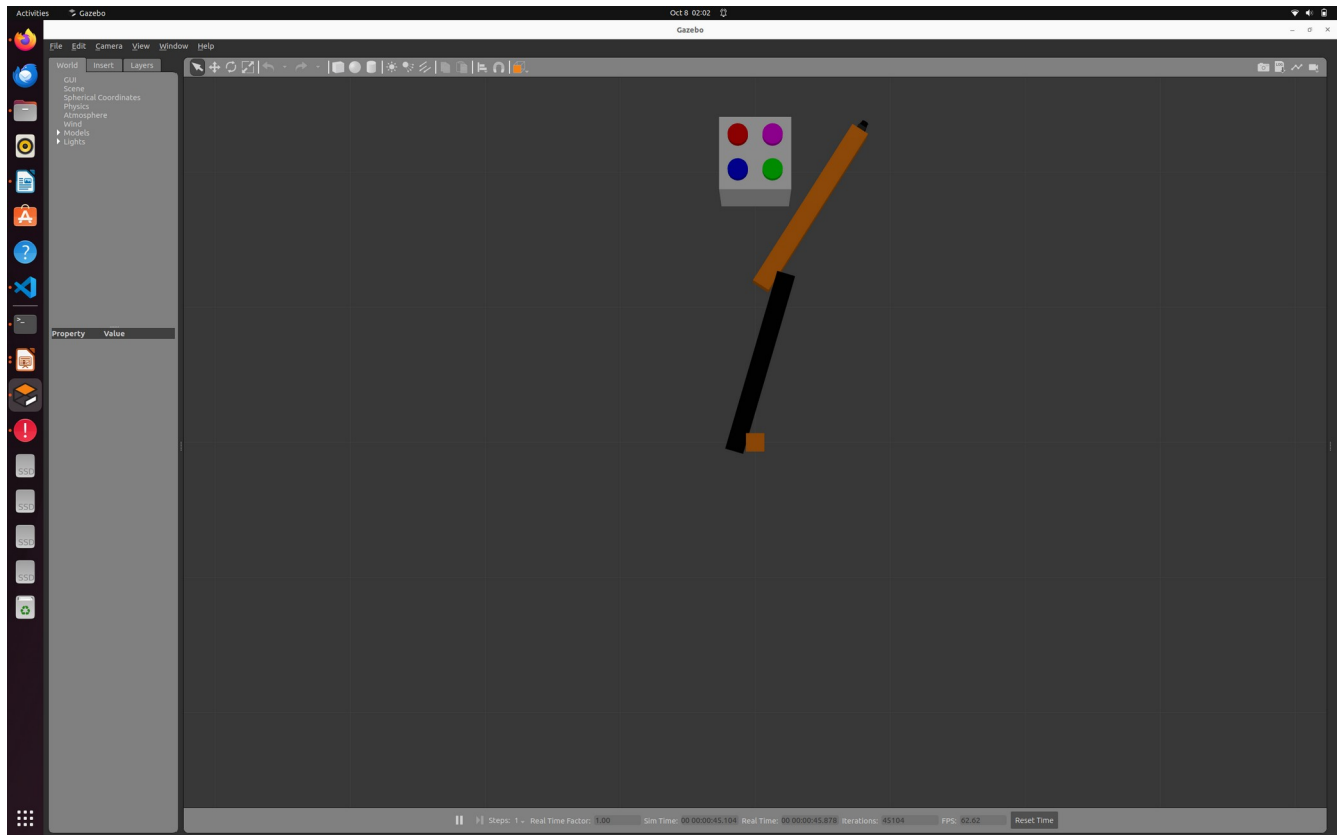
started simulation and bot joint angle [0,0]



switched to vel control



Bot moved to ref position and stopped there



```
[INFO] [1728367305.065047689] [image_subscriber]: reference features: [(187, 342), (221, 418), (297, 384), (262, 307)]
[INFO] [1728367305.065288060] [image_subscriber]: current features: [(189, 437), (149, 364), (262, 397), (222, 324)]
[INFO] [1728367305.065575460] [image_subscriber]: error: [ 2 95 -72 -54 -35 13 -40 17]
[INFO] [1728367305.066055142] [image_subscriber]: image_jacobian: [[-1.00000e+00  0.00000e+00  1.89000e+02  8.25930e+04 -3.57220e+04
 4.37000e+02]
 [ 0.00000e+00 -1.00000e+00  4.37000e+02  1.90970e+05 -8.25930e+04
 -1.89000e+02]
 [-1.00000e+00  0.00000e+00  1.49000e+02  5.42360e+04 -2.22020e+04
 3.64000e+02]
 [ 0.00000e+00 -1.00000e+00  3.64000e+02  1.32497e+05 -5.42360e+04
 -1.49000e+02]
 [-1.00000e+00  0.00000e+00  2.62000e+02  1.04014e+05 -6.86450e+04
 3.97000e+02]
 [ 0.00000e+00 -1.00000e+00  3.97000e+02  1.57610e+05 -1.04014e+05
 -2.62000e+02]
 [-1.00000e+00  0.00000e+00  2.22000e+02  7.19280e+04 -4.92850e+04
 3.24000e+02]
 [ 0.00000e+00 -1.00000e+00  3.24000e+02  1.04977e+05 -7.19280e+04
 -2.22000e+02]]
[INFO] [1728367305.066356165] [image_subscriber]: robot_jacobian: [[-0.25004509 -0.73232149]
 [ 0.96823419  1.8442533 ]
 [ 0.          0.          ]
 [ 0.          0.          ]
 [ 0.          0.          ]
 [ 1.          1.          ]]
[INFO] [1728367305.066659969] [image_subscriber]: robot vel: [[-3.15550731e-03]
 [-5.71233875e-03]
 [ 7.83377021e-04]
 [-1.82477048e-06]
 [ 4.56747551e-07]
 [ 7.26006766e-05]]
[INFO] [1728367305.066958287] [image_subscriber]: joint_vel: [[ 0.00107908 -0.00229974]]
```

