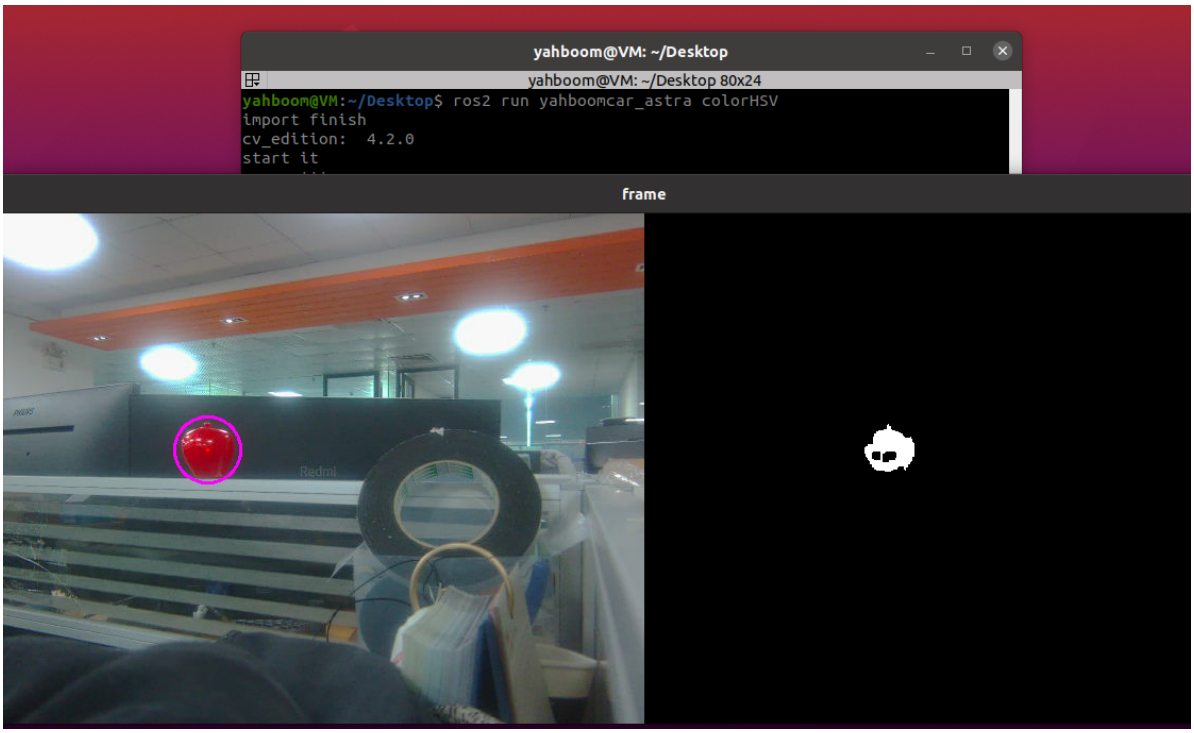
3.Color tracking

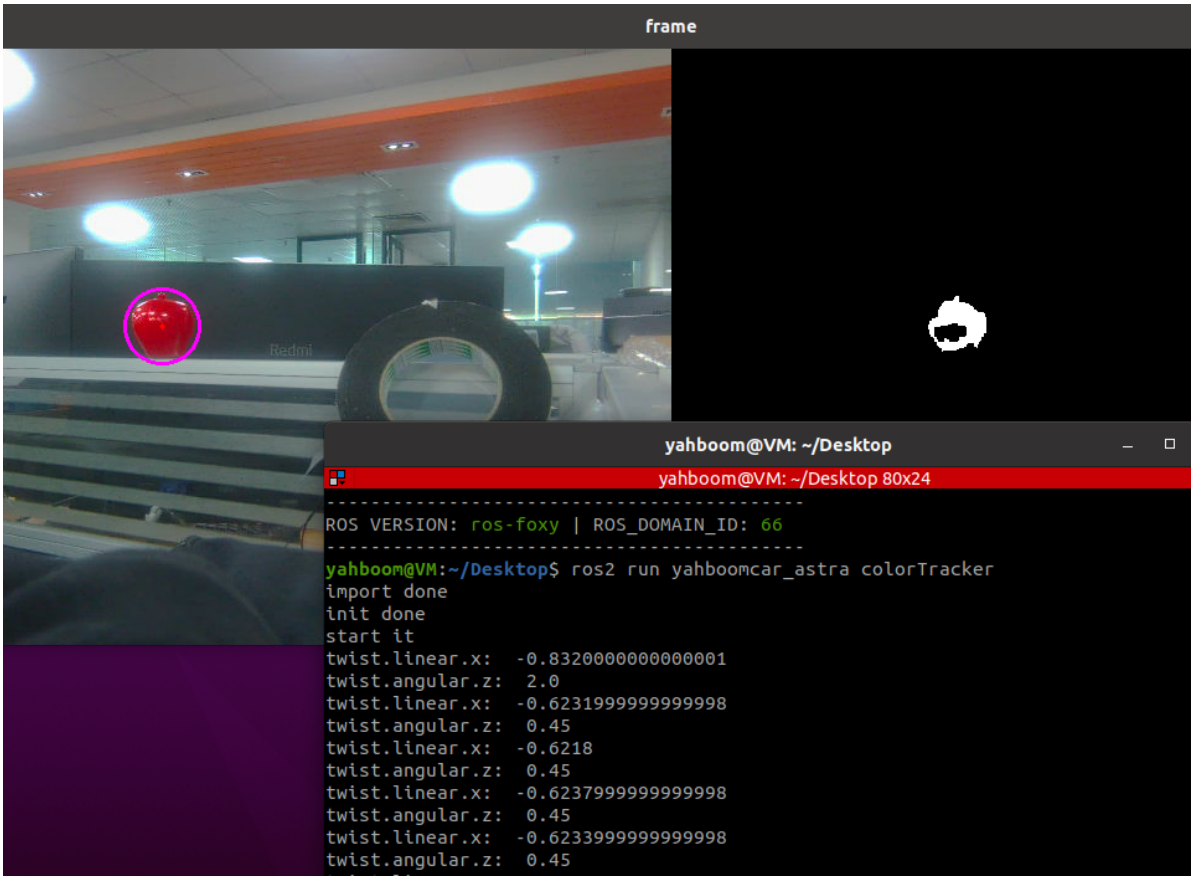
1.Start program

Input following command:

```
ros2 launch orbbec_camera dabai_dcw2.launch.py
ros2 run yahboomcar_astra colorHSV
ros2 run yahboomcar_astra colorTracker
```

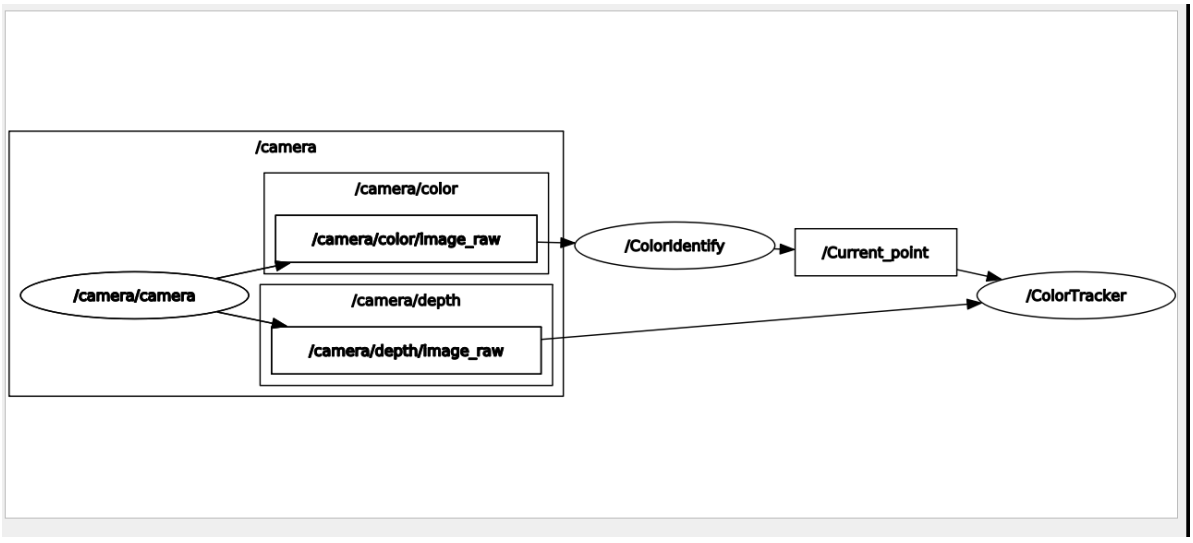


After successful startup, the above screen will be displayed, and the program will initially load the HSV value, and then display the processed image.
Press the [r] button to reselect the color and use the mouse to box out the color to be tracked. The selected area can only have one color.
After selecting the color to be tracked, the program completes the image processing and presses the spacebar to start tracking. The terminal that starts colorTracker will display.



View communication between nodes, input following command.

```
ros2 run rqt_graph rqt_graph
```



3.2 Code

Code path:

```
~/orbbec_ws/src/yahboomcar_astra/yahboomcar_astra/colorHSV.py
~/orbbec_ws/src/yahboomcar_astra/yahboomcar_astra/colorTracker.py
```

3.2.1、colorHSV.py

This program mainly has the following functions:

- Subscribe to camera image data;
- Obtain keyboard and mouse events for switching modes and color selection;
- Process images and publish the center coordinates of tracked objects and publish them

Some core codes are as follows:

```
#创建发布者和订阅者
self.pub_position = self.create_publisher(Position,"/Current_point", 10)
self.sub_img
=self.create_subscription(Image,'/camera/color/image_raw',self.handleTopic,1)
#订阅图像回调函数把图像传process函数
frame, binary =self.process(frame, action)
#获取键盘鼠标事件，得到hsv的值;
if action == 32: self.Track_state = 'tracking'
elif action == ord('i') or action == ord('I'): self.Track_state =
"identify"
elif action == ord('r') or action == ord('R'): self.Reset()
elif action == ord('q') or action == ord('Q'): self.cancel()
if self.Track_state == 'init':
cv.namedWindow(self.windows_name, cv.WINDOW_AUTOSIZE)
cv.setMouseCallback(self.windows_name, self.onMouse, 0)
if self.select_flags == True:
cv.line(rgb_img, self.cols, self.rows, (255, 0, 0), 2)
cv.rectangle(rgb_img, self.cols, self.rows, (0, 255, 0), 2)
if self.Roi_init[0] != self.Roi_init[2] and self.Roi_init[1] !=
self.Roi_init[3]:
rgb_img, self.hsv_range = self.color.Roi_hsv(rgb_img,
self.Roi_init)
self.gTracker_state = True
self.dyn_update = True
else: self.Track_state = 'init'
#计算中心坐标的值，self.circle存放xy值
rgb_img, binary, self.circle = self.color.object_follow(rgb_img, self.hsv_range)
#发布中心坐标的消息
threading.Thread(target=self.execute, args=(self.circle[0], self.circle[1],
self.circle[2])).start()
def execute(self, x, y, z):
position = Position()
position.anglex = x * 1.0
position.angley = y * 1.0
position.distance = z * 1.0
self.pub_position.publish(position)
```

3.2.2、colorTracker.py

The main function of this program is to receive/Current_ Point and depth image topic data, calculate speed size, and then publish speed data.

```
#定义订阅者接收需要的话题数据
self.sub_depth =
self.create_subscription(Image,"/camera/depth/image_raw",self.depth_img_Callback
```

```
, 1)
self.sub_position
=self.create_subscription(Position,"/Current_point",self.positionCallback,1)
#定义速度发布者
self.pub_cmdVel = self.create_publisher(Twist,'/cmd_vel',10)
#两个重要的回调函数，获取到self.Center_x值和distance_值
def positionCallback(self, msg):
def depth_img_Callback(self, msg):
    #self.Center_x值和distance_值根据计算线速度，角速度
    self.execute(self.Center_x, distance_)
    def execute(self, point_x, dist):
    self.get_param()
    if abs(self.prev_dist - dist) > 300:
    self.prev_dist = dist
    return
    if abs(self.prev_angular - point_x) > 300:
        self.prev_angular = point_x
        return
    if self.Joy_active == True: return
    linear_x = self.linear_pid.compute(dist, self.minDist)
    angular_z = self.angular_pid.compute(320, point_x)
    if abs(dist - self.minDist) < 30: linear_x = 0
    if abs(point_x - 320.0) < 30: angular_z = 0
    twist = Twist()
    if angular_z>2.0:
        angular_z = 2.0
    if angular_z<-2.0:
        angular_z = -2.0
    if linear_x > 1.0:
        linear_x = 1.0
    if linear_x <-1.0:
        linear_x = -1.0
    twist.angular.z = angular_z * 1.0
    twist.linear.x = linear_x * 1.0
```