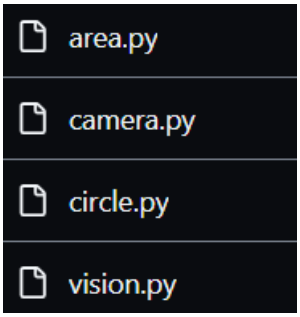
Dokumentasi Pengerjaan Final Project Bayucaraka 2024

Muhammad Zia Alhambra | 5024231059

Pengerjaan saya mulai pada hari Kamis, 15 Februari 2024, satu hari setelah pengumuman final project. Seperti pengerjaan tugas berbasis ROS2 yang telah saya lakukan, saya membuat workspace final project lalu membuat dua package, satu package python (vision) dan satu lagi package CPP (control), sesuai perintah final project.

Pada awalnya saya berencana ingin membuat satu node untuk masing-masing package, tetapi untuk package vision, untuk mengolah pecintraan gambar, saya merasa satu node akan terasa terlalu berantakan, maka saya membuat empat node pada package vision; camera.py, area.py, circle.py, vision.py.



Camera.py

Node ROS2 ini berfungsi untuk menyalakan camera dan mem-*publish*-nya menjadi topic “camera” agar dapat digunakan node lain. Perlu diketahui saya melakukan cropping besar display karena saya menggunakan camera laptop untuk mengambil display. Jika display terlalu besar dapat menyebabkan node pencitraan gambar untuk mendapatkan *contour* yang tidak diinginkan.

```python
class CameraPublisher(Node):
    def __init__(self):
        super().__init__('camera_publisher')
        self.publisher_ = self.create_publisher(Image, 'camera', 10)
        self.timer_ = self.create_timer(0.1, self.publish_image)
        self.bridge = CvBridge()

    def publish_image(self):
        cap = cv2.VideoCapture(0)
        if not cap.isOpened():
            self.get_logger().error("Failed to open camera")
            return

        ret, frame = cap.read()
        if not ret:
            self.get_logger().error("Failed to capture frame from camera")
            return

        crop = frame[250:480, 100:500]

        ros_image = self.bridge.cv2_to_imgmsg(crop, "bgr8")
        self.publisher_.publish(ros_image)
        cap.release()

def main(args=None):
    rclpy.init(args=args)
    camera_publisher = CameraPublisher()
    rclpy.spin(camera_publisher)
    camera_publisher.destroy_node()
    rclpy.shutdown()
```

Area.py

Node ini berfungsi untuk mendeteksi area (kotak) dari camera. Code saya menampilkan webcam dengan cara subscribe kepada topic “camera” yang telah kita buat dalam node camera.py. selanjutnya, code melakukan masking camera dan mencari contour yang muncul serta membuat bounding box mengelilingi area yang terdapat contour tersebut. Lalu, code saya melakukan kalkulasi titik maximum, minimum, dan titik tengah pada bidang x dan y pada setiap bounding box yang ada dan memasukkan masing-masing kedalam sebuah lists sesuai namanya. Perlu dicatat bahwa kebanyakan code yang saya digunakan pada node ini berasal dari tugas 1 pelatihan OpenCV oleh Mas Reza.

```
def image_callback(self, msg):
    cv_image = self.bridge.imgmsg_to_cv2(msg, desired_encoding="bgr8")

    # Define the color range
    lowerBound = np.array([150, 150, 100], dtype=np.uint8)
    upperBound = np.array([255, 255, 255], dtype=np.uint8)

    # Create a mask using the color range
    mask = cv2.inRange(cv_image, lowerBound, upperBound)

    # Find contours in the mask
    contours, _ = cv2.findContours(mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

    max_x_points = []  # List to store maximum x-coordinates
    min_x_points = []  # List to store minimum x-coordinates
    max_y_points = []  # List to store maximum y-coordinates
    min_y_points = []  # List to store minimum y-coordinates
    middle_x_points = []
    middle_y_points = []

    # Iterate through each contour
    for contour in contours:
        x, y, w, h = cv2.boundingRect(contour)
        area = cv2.contourArea(contour)

        # If the area of the contour is within a certain range, draw a bounding box
        if 1000 < area < 10000:
            color = (255, 0, 0)  # Red color for the bounding box
            thickness = 2
            cv2.rectangle(cv_image, (x, y), (x + w, y + h), color, thickness)

            max_x_points.append(x + w)
            min_x_points.append(x)
            max_y_points.append(y + h)
            min_y_points.append(y)

            middle_x = x + w // 2
            middle_y = y + h // 2
            middle_x_points.append(middle_x)
            middle_y_points.append(middle_y)


    cv2.imshow("area", cv_image)
    key = cv2.waitKey(1)
```

Karena kita nanti akan publish lists tersebut, ditakutkan akan publish setiap saat, maka saya buat baris agar code publish lists setiap kali saya menekan tombol ‘q’ pada keyboard. Setelah itu, node akan publish area/max_x, area/min_x, area/max_y, area/min_y, area/middle_x, dan area/middle_y ke beragai macam node lainnya.

```
if key & 0xFF == ord('q'):
    # Publish middle points as integer values
    self.publish_points(max_x_points, min_x_points, max_y_points, min_y_points, middle_x_points, middle_y_points)
    # Print the number of areas and their coordinates
    print(f"{len(max_x_points)} area(s) found when 'q' was clicked:")
    for idx, (max_x, min_x, max_y, min_y) in enumerate(zip(max_x_points, min_x_points, max_y_points, min_y_points)):
        print(f"Area {idx+1}: Max X = {max_x}, Min X = {min_x}, Max Y = {max_y}, Min Y = {min_y}")
```

Circle.py

Node ini kurang lebih sama dengan area.py. Node ini berfungsi untuk mendeteksi warna merah yang merupakan Gerakan-gerakan oleh player dan menyetornya pada list middle_x_points dan middle_y_points yang nantinya dipublish. Nantinya node ini juga dapat menampilkan Gerakan bot dengan bulat biru.

Node subscribe kepada topic “selected_integers” dari node control.cpp serta “area/middle_x” dan “area/middle_y” dari node area.py lalu menyimpannya pada sebuah list. Lalu sebuah algoritma menampilkan bulat berwarna biru sesuai kotak bernomor sesuai integer dari list selected_integers_list pada display webcam dengan titik tengah berasal dari middle_x_list dan middle_y_list.

Contohnya:

Selected_integers_list = [1, 3]

Middle_x_list = [a, b, c, …]

Middle_y_list = [m, n, o, …]

⇨ [MEMBUAT 2 LINGKARAN PADA KOTAK 1 DAN 3]
#cv2.circle(cv_image, (middle_x, middle_y), 25, blue_color, -1)

cv2.circle(cv_image, (a, m), 25, blue_color, -1)

cv2.circle(cv_image, (c, o), 25, blue_color, -1)

```
for integer in self.selected_integers_list:
    index = integer - 1
    # Check if the integer is within the range of available middle points
    if 0 <= integer < len(self.middle_x_list):
        # Get the corresponding middle point
        middle_x = self.middle_x_list[index]
        middle_y = self.middle_y_list[index]

        # Define the color blue
        blue_color = (255, 0, 0)

        # Draw circle
        cv2.circle(cv_image, (middle_x, middle_y), 25, blue_color, -1)
```

Vision.py

Node ini berfungsi untuk menggabungkan node area.py dan circle.py untuk mencari posisi bulat merah pada kotak nomor berapa yang lalu disimpan dalam list available_area. Dalam available_area ada integer dari 1 sampai 9 (karena banyak area dalam tic tac toe ada 9). Jika terdapat bulat dalam sebuah area maka akan menghilangkan integer tersebut dari list available_area. Seperti node sebelumnya, message dari topic disimpan dalam sebuah list untuk digunakan nantinya. Ada sedikit kendala saat publish available_area, saat publish akan juga publish list yang belum terupdate sebelumnya sebanyak nomor bulat merah yang tersedia. Untuk menanggulani hal tersebut saya tambahkan if statement sebelum node publish list available_area.

```
def assign_circle_to_area(self):
    previous_available_area = self.available_area.copy()  # Make a copy of the previous available areas
    try:
        for i in range(len(self.circle_x)):
            circle_x = self.circle_x[i]
            circle_y = self.circle_y[i]
            for j in range(len(self.area_maxX)):
                max_x = self.area_maxX[j]
                min_x = self.area_minX[j]
                max_y = self.area_maxY[j]
                min_y = self.area_minY[j]
                if min_x <= circle_x <= max_x and min_y <= circle_y <= max_y:
                    print(f"Circle {i+1} on Area {j+1}")
                    try:
                        self.available_area.remove(j+1)
                    except ValueError:
                        pass  # If the area is already removed or not in the list, do nothing
            print("Available Areas:", self.available_area)
    except IndexError:
        print("No circles or areas detected")

    # Check if there are changes in available areas
    if previous_available_area != self.available_area:
        self.publish_available_area()  # Publish only if there are changes
```

Control.cpp

Node ini berfungsi untuk menggerakkan bot untuk mengambil salah satu integer dari topic available_area yang di-subscribe dan disimpan dalam sebuah list dan di-publish agar dapat menampilkan bulat biru dalam node circle.py. Karena waktu terbatas, saya tidak dapat mengimplementasikan algoritma untuk gerak bot. Untuk gantinya, bot akan mengambil salah satu area kosong dari available_area dan disimpan pada list baru bernama selected_integer. Setelah itu, node akan publish topic selected_integer ke node circle.py.

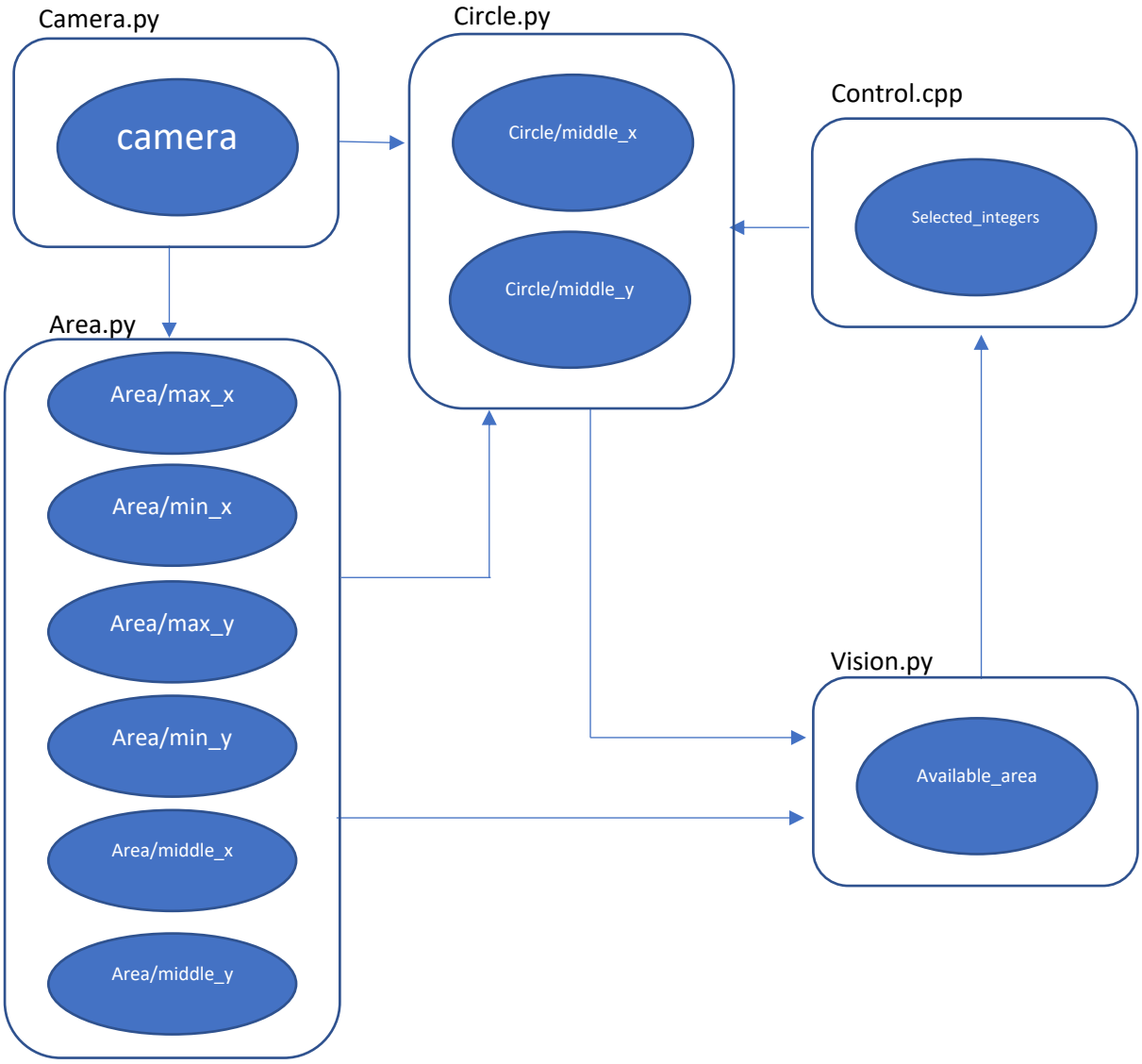
```
if (!received_data.empty()) {
    srand(time(nullptr)); // Seed the random number generator
    int random_index;
    int selected_integer;

    do {
        random_index = rand() % received_data.size(); // Generate a random index
        selected_integer = received_data[random_index]; // Select the integer at the random index
    } while (std::find(selected_integers.begin(), selected_integers.end(), selected_integer) != selected_integers.end());

    selected_integers.push_back(selected_integer); // Store the selected integer in the new list
    RCLCPP_INFO(this->get_logger(), "Selected integer: %d", selected_integer);

    // Publish the selected_integers list
    auto msg = std_msgs::msg::Int32MultiArray();
    msg.data = selected_integers;
    publisher_->publish(msg);
}
```

Alur topic dalam final project



Video: https://drive.google.com/file/d/1DpNAg2lSk-h9yAbpDIZOkf2skNuurnnW/view?usp=sharing