FINAL PROJECT BAYUCARAKA

CATHLEEN GRACIA - 5025231018
TEKNIK INFORMATIKA

VISION

- GET_CORNERS_POINT
- PLAYER_MOVE
- COMPUTER_MOVE
- STRING_CALLBACK
- CALLBACK
- CAMERA_FRAME

CONTROL

- CHECK_WIN
- MINIMAX
- COMPUTER_MOVE
- TOPIC_CALLBACK

VISION

GET_CORNERS_POINT

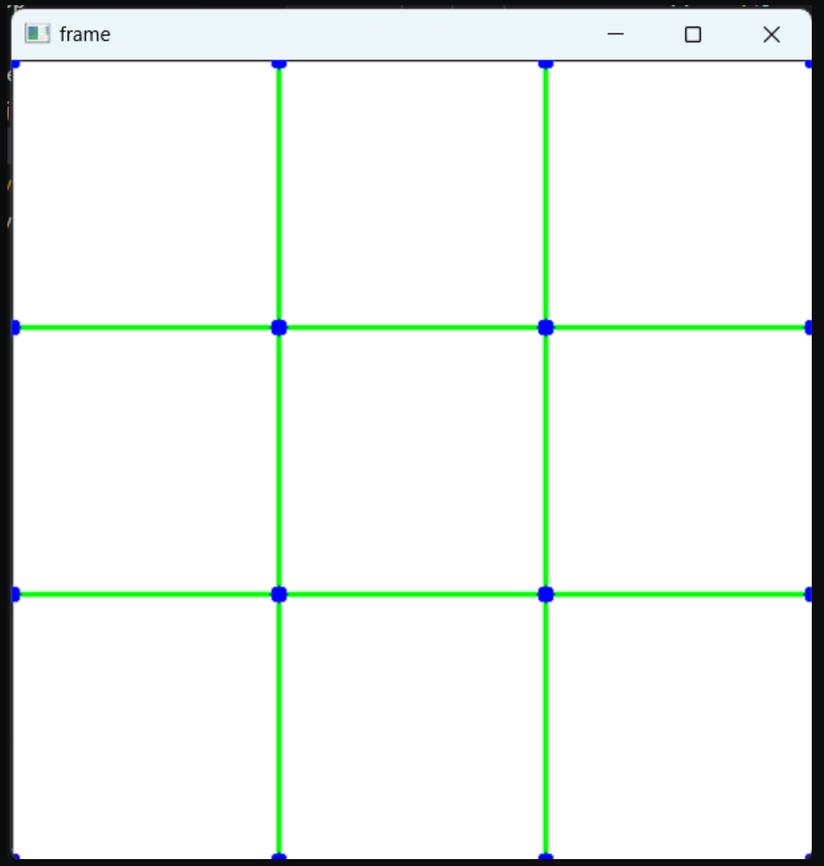
```python
# Mendapatkan corner points dari grid tictactoe
def get_corner_points(self, frame_shape):
        height, width = frame_shape
        corners = []

        for i in range(4):
            for j in range(4):
                corners.append((j * width // 3, i * height // 3))

        return corners
```

Fungsi untuk mendapatkan corner point atau titik-titik dari grid tictactoe yang telah dibuat

VISION

GET_CORNERS_POINT

```python
# Mendapatkan corner points dari grid tictactoe
def get_corner_points(self, frame_shape):
        height, width = frame_shape
        corners = []

        for i in range(4):
            for j in range(4):
                corners.append((j * width // 3, i * height // 3))

        return corners
```



Fungsi untuk mendapatkan corner point atau titik-titik dari grid tictactoe yang telah dibuat

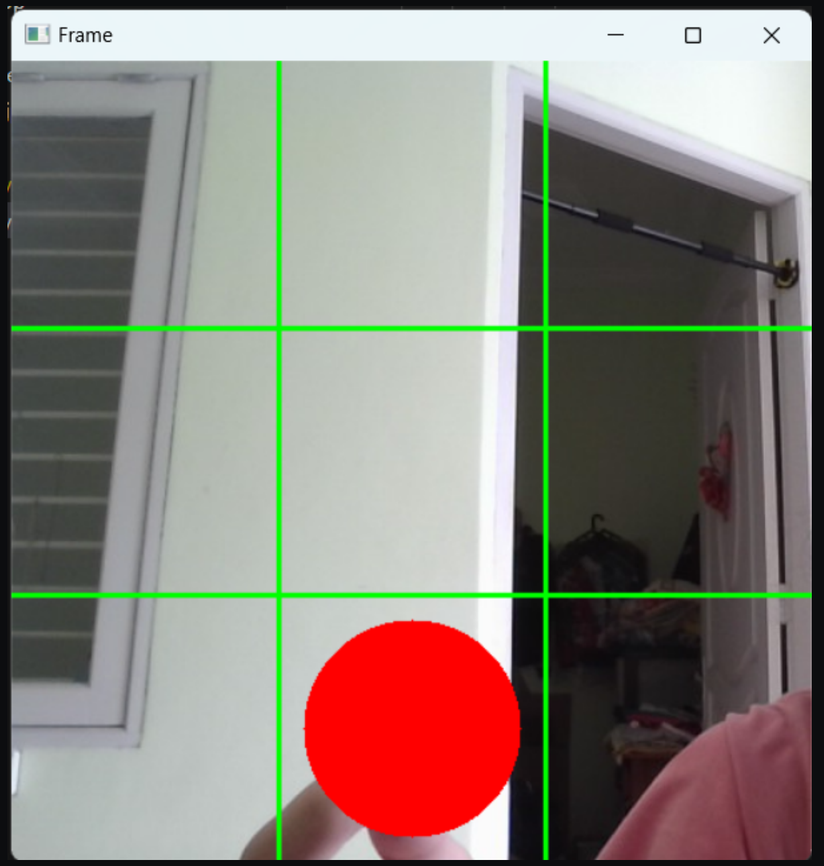
```
Corner points: [(0, 0), (160, 0), (320, 0), (480, 0), (0, 160), (160, 160), (320, 160), (480, 160), (0, 320), (160, 320), (320, 320), (480, 320), (0
, 480), (160, 480), (320, 480), (480, 480)]
```

VISION

PLAYER_MOVE

```python
# Menggambar langkah dari player
def player_move(self, corners, cropped_frame):
    if self.player[1] == 1:
        a = corners[5][0] // 2
        b = corners[5][1] // 2
        cv2.circle(cropped_frame, (a, b), 70, (0, 0, 255), -1)
```

Fungsi untuk menggambar lingkaran berwarna merah pada kotak yang dipilih sebagai langkah player

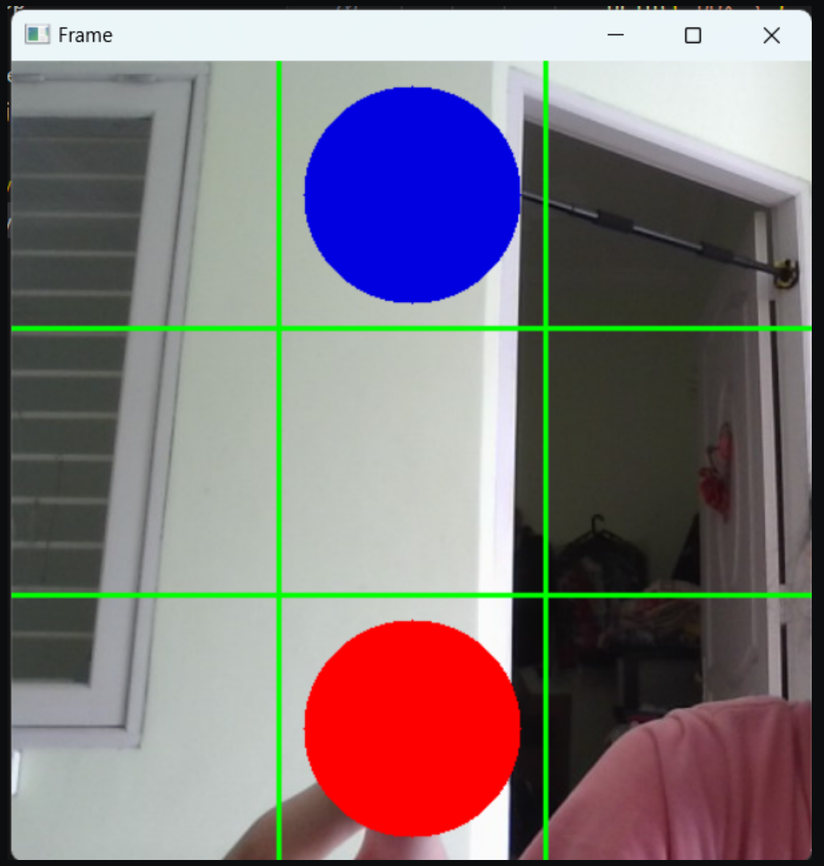


VISION

COMPUTER_MOVE

```python
# Menggambar langkah dari computers
def computer_move(self, corners, cropped_frame):
    if self.computer[1] == 1:
        a = corners[5][0] // 2
        b = corners[5][1] // 2
        cv2.circle(cropped_frame, (a, b), 70, (225, 0, 0), -1)
```

Fungsi untuk menggambar lingkaran berwarna biru pada kotak yang merupakan langkah dari computer.

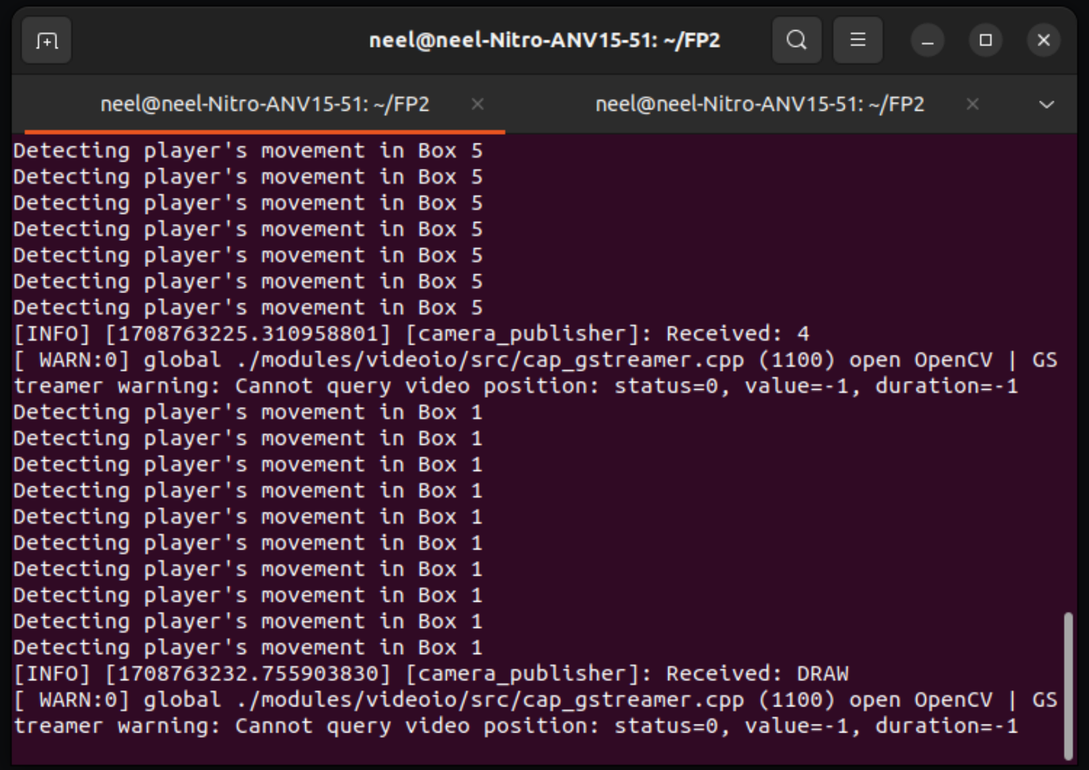


VISION

STRING_CALLBACK

```python
# Menerima pesan berupa String dari control
def string_callback(self, msg):
    self.get_logger().info('Received: %s' % msg.data)

    self.end = f'{msg.data}'
    self.endflag = 1

    if self.moves == 9:
        self.camera_frame()
```

Fungsi untuk menerima pesan berbentuk String yang berisi pernyataan apabila player menang, kalah, atau seri.

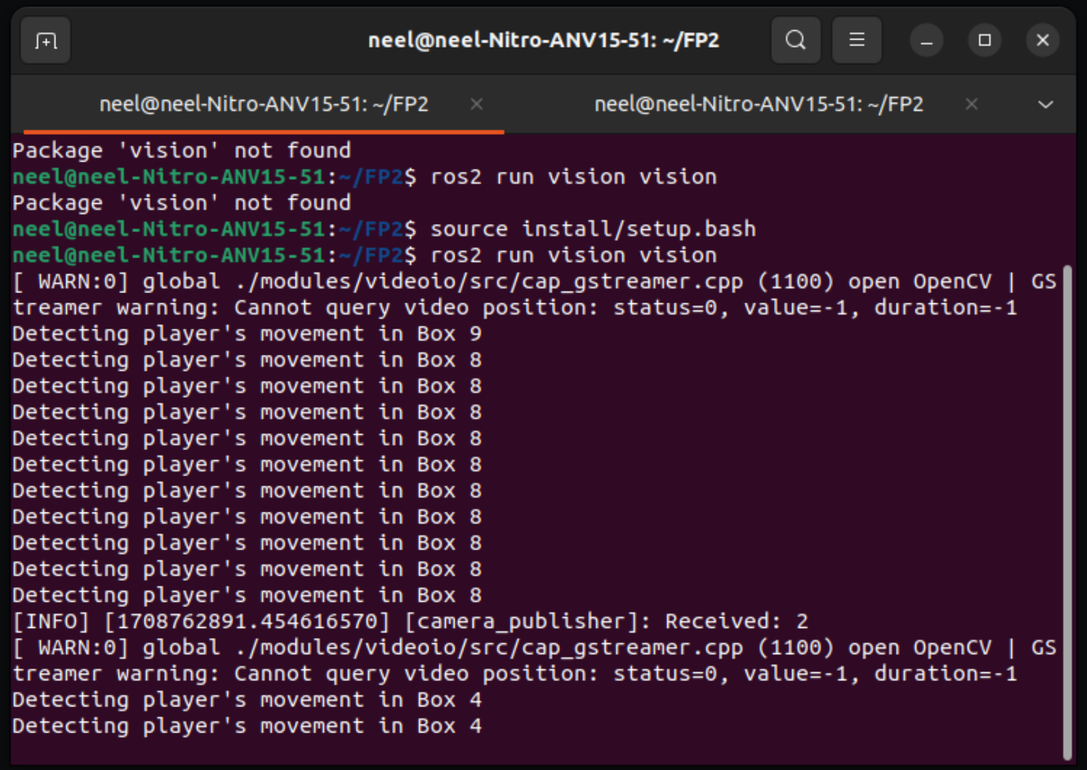


VISION

CALLBACK

```python
# Menerima pesan berupa Int dari control
def callback(self, msg):
    self.get_logger().info('Received: %d' % msg.data)
    computer = msg.data

    self.computer[computer] = 1

    self.moves += 1

    self.camera_frame()
```

Fungsi untuk menerima pesan berbentuk Integer yang berisi langkah computer hasil pengolahan dari control.



VISION

Fungsi camera_frame berguna untuk mendeteksi dimana langkah player dengan menggunakan OpenCV.

```python
# Mendeteksi langkah player dengan menggunakan OpenCV
def camera_frame(self):
    cap = cv2.VideoCapture(0)

    count = np.zeros(10)
    index = 0

    while True:
        ret, frame = cap.read()

        if not ret :
            break

        flag = 0

        height, width = frame.shape[:2]

        size = min(height, width)

        flipped = cv2.flip(frame, 1)

        cropped_frame = flipped[0:size, 0:size]
```

Sintaks ini digunakan untuk membuat objek kamera untuk menangkap video dan membacanya. Lalu frame di *mirror* dan diubah ukurannya agar bentuknya menjadi persegi..

VISION

CAMERA_FRAME

```
for i in range(1, 3):
    cv2.line(cropped_frame, (0, i * cropped_frame.shape[0] // 3), (cropped_frame.shape[1], i * cropped_frame.shape[0] // 3), (0, 255, 0), 2)
    cv2.line(cropped_frame, (i * cropped_frame.shape[1] // 3, 0), (i * cropped_frame.shape[1] // 3, cropped_frame.shape[0]), (0, 255, 0), 2)
```

Sintaks ini digunakan untuk menggambar grid dari tictactoe.

```
corners = self.get_corner_points(cropped_frame.shape[:2])
```

Sintaks ini digunakan untuk memanggil fungsi get_corners_points dan mendapatkan nilai corners atau koordinat titik pada grid.

VISION

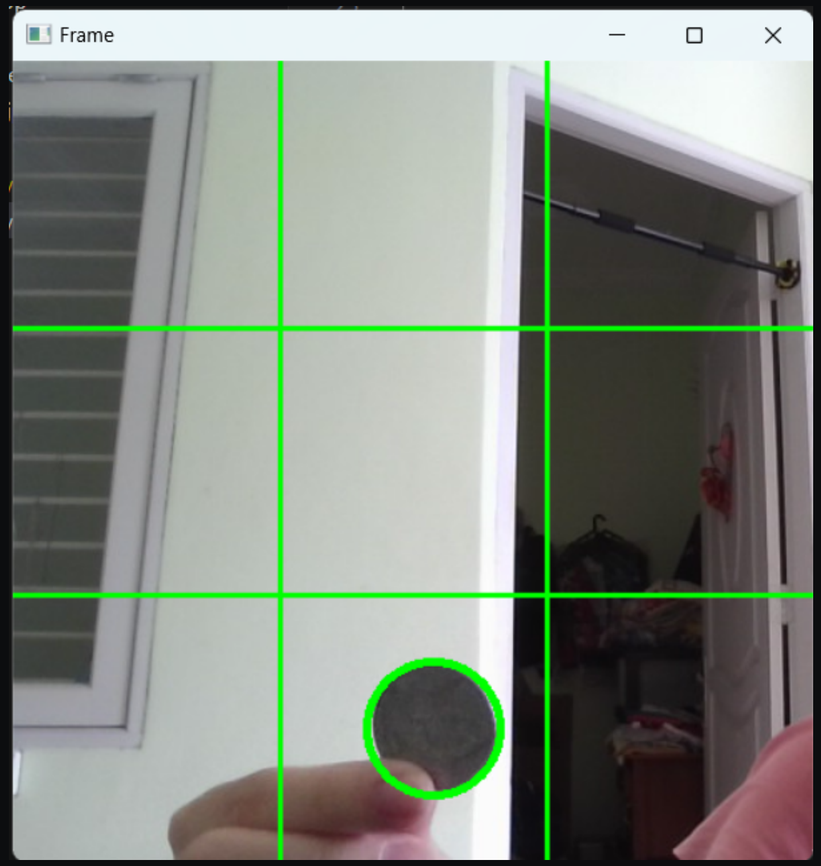
CAMERA_FRAME

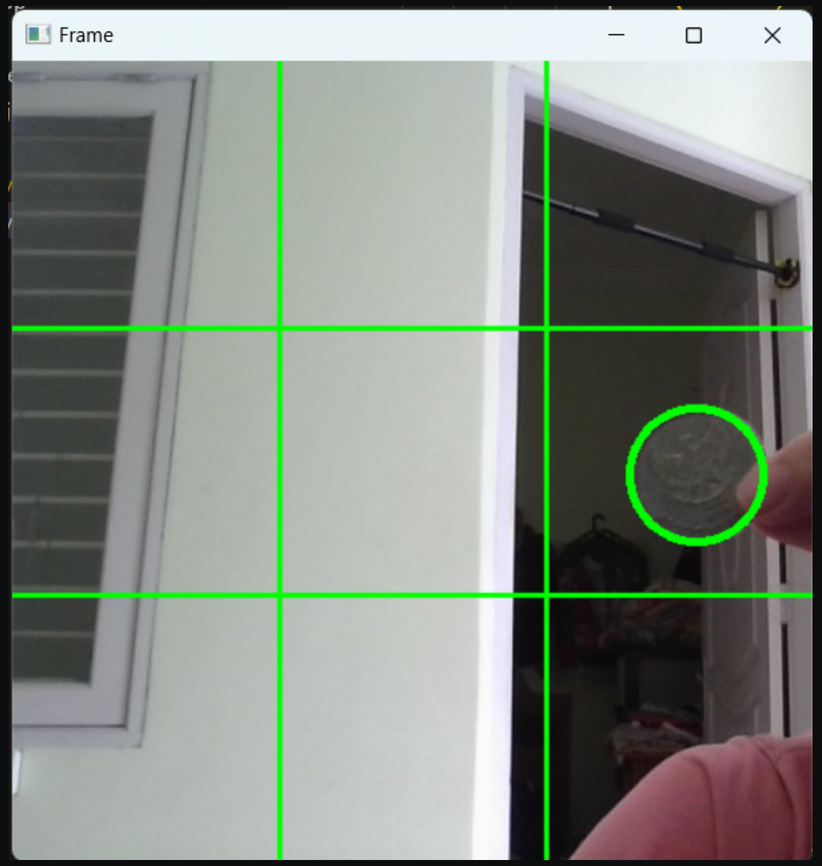
```
circles = cv2.HoughCircles(gray, cv2.HOUGH_GRADIENT, dp=0.1, minDist=30, param1=150, param2=30, minRadius=35, maxRadius=100)

if circles is not None and self.endflag == 0:
    circles = np.round(circles[0, :]).astype("int")

    for (x, y, r) in circles:
        cv2.circle(cropped_frame, (x, y), r, (0, 255, 0), 4)

        if corners[0][0] <= x <= corners[5][0] and corners[0][1] <= y <= corners[5][1]:
            print("Detecting player's movement in Box 1")
            count[1] += 1
            if count[1] == 10.0:
                flag = 1
                self.player[1] = 1
                index = 1
```

Sintaks ini digunakan untuk mendeteksi lingkaran pada frame.

VISION

CAMERA_FRAME





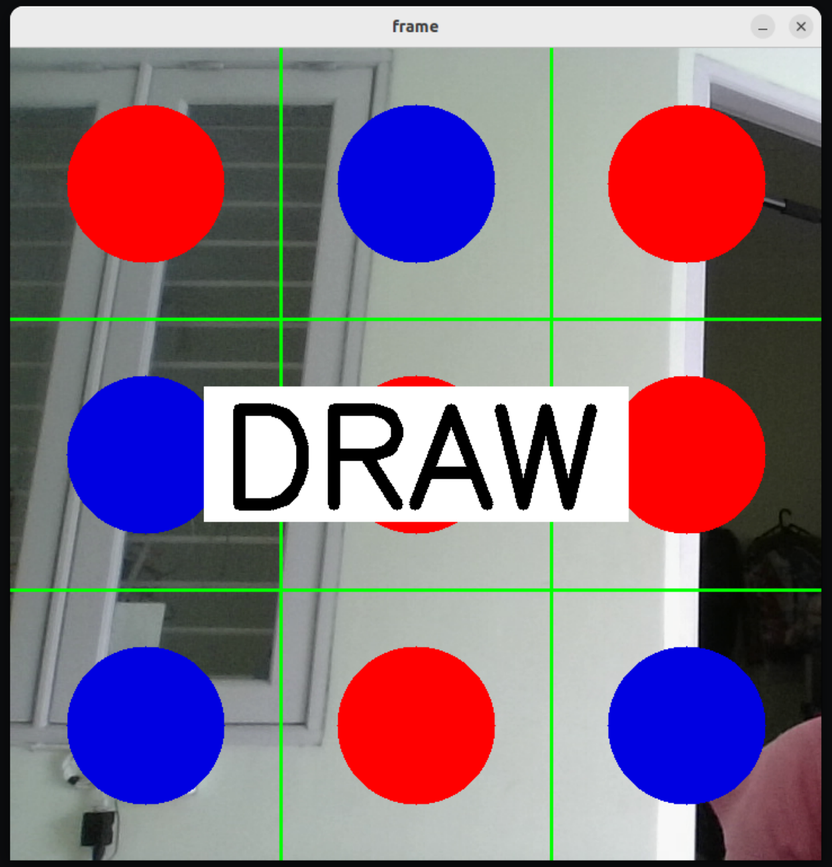
VISION

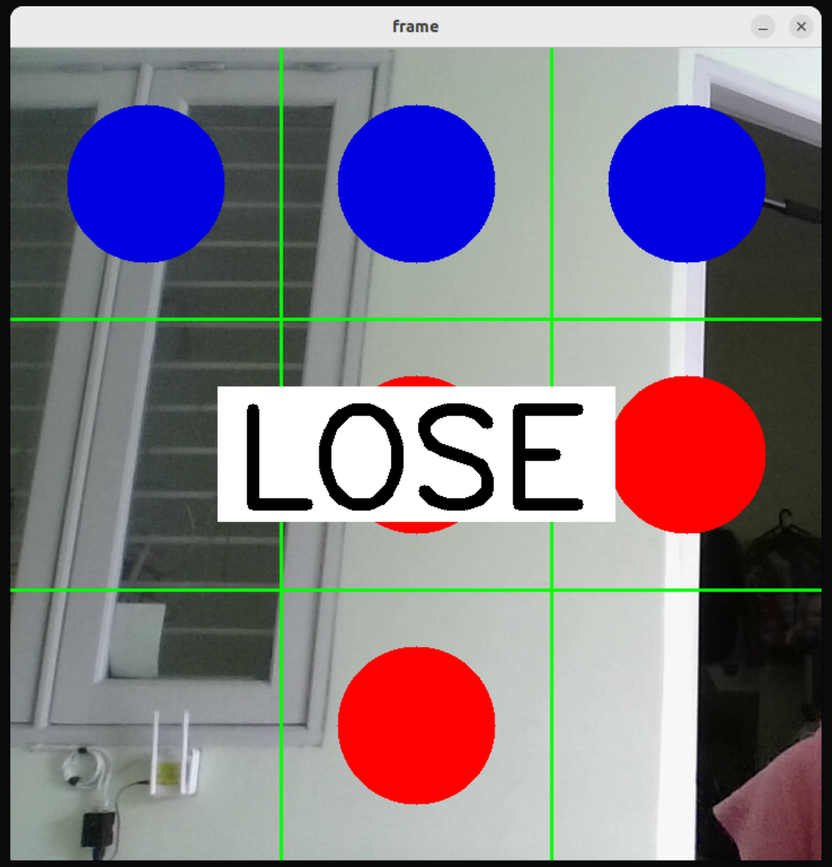

```
if self.endflag == 1:
    font = cv2.FONT_HERSHEY_SIMPLEX
    font_scale = 4
    font_color = (0, 0, 0)
    thickness = 10

    frame_size = cropped_frame.shape

    text_size = cv2.getTextSize(self.end, font, font_scale, thickness)

    text_x = int((frame_size[0] - text_size[0][0]) / 2)
    text_y = int((frame_size[1] + text_size[0][1]) / 2)

    rectangle_start = (int(text_x - 15), int(text_y + 15))
    rectangle_end = (int(text_x + text_size[0][0] + 15), int(text_y - text_size[0][1] - 15))

    cv2.rectangle(cropped_frame, rectangle_start, rectangle_end, (255, 255, 255), -1)
    cv2.putText(cropped_frame, self.end, (text_x, text_y), font, font_scale, font_color, thickness)
```

Sintaks ini digunakan untuk menampilkan text pada frame apabila pesan String sudah diterima.

VISION

CAMERA_FRAME





VISION

CAMERA_FRAME

```
if flag == 1 and self.endflag == 0:
    player_msg = Int16()
    player_msg.data = index

    self.publisher_.publish(player_msg)

    self.moves += 1

    break
```

Sintaks ini digunakan untuk mengirimkan pesan berupa Integer kepada control yang berisi langkah dari player.

CONTROL

CHECK_WIN

```
// Memeriksa apakah ada yang menang atau tidak
int check_win(){
  int wins[8][3] = {{0, 1, 2}, {3, 4, 5}, {6, 7, 8}, {0, 3, 6}, {1, 4, 7}, {2, 5, 8}, {0, 4, 8}, {2, 4, 6}};
  for(int i = 0; i < 8; ++i){
    if(board_[wins[i][0]] != 0 && board_[wins[i][0]] == board_[wins[i][1]] && board_[wins[i][1]] == board_[wins[i][2]]){
      return board_[wins[i][2]];
    }
  }
  return 0;
}
```

Fungsi ini digunakan untuk mengecek apakah ada yang menang antara player atau computer. Ketika player menang, fungsi akan mengembalikan nilai -1. Ketika computer menang, fungsi akan mengembalikan nilai 1. Ketika tidak ada yang menang atau kalah, fungsi akan mengembalikan nilai 0.

CONTROL


```
int minimax(int player){
  int winner = check_win();

  if(winner != 0){
    return winner * player;
  }

  int move = -1;
  int score = -2;

  for(int i = 0; i < 9; i++){
    if(board_[i] == 0){
      board_[i] = player;
      int tempscore = -minimax(player*-1);

      if(tempscore > score){
        score = tempscore;
        move = i;
      }

      board_[i] = 0;
    }
  }
  if(move == -1){
    return 0;
  }
  return score;
}
```

Fungsi ini digunakan untuk mencoba semua kemungkinan langkah yang bisa dilakukan untuk menghasilkan langkah terbaik yang dapat dilakukan oleh computer.

CONTROL

COMPUTER_MOVE

```
int computer_move(){
  int move = -1;
  int score = -2;

  for(int i = 0; i < 9; i++){
    if(board_[i] == 0){
      board_[i] = 1;
      int tempscore = -minimax(-1);
      board_[i] = 0;

      if(tempscore > score){
        score = tempscore;
        move = i;
      }
    }
  }
  return move;
}
```

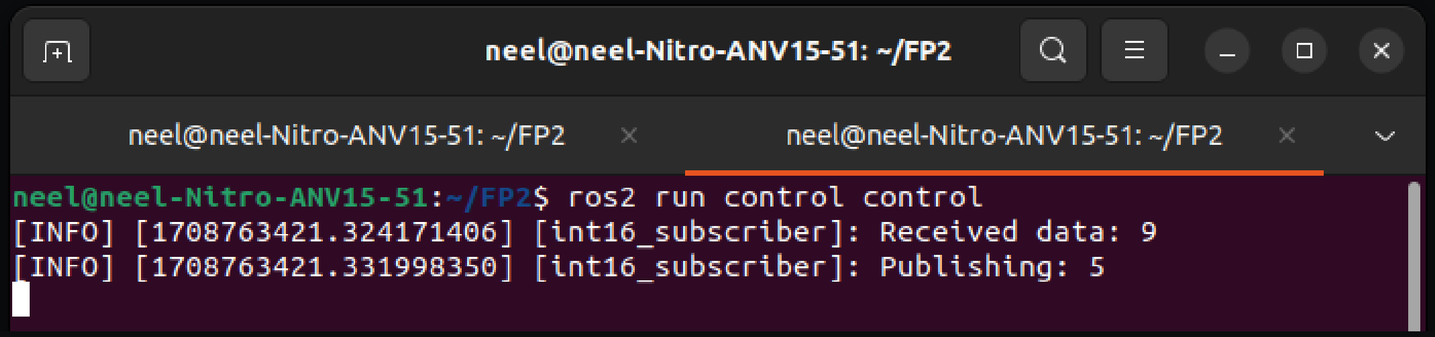
Fungsi ini digunakan untuk menggerakan computer berdasarkan langkah terbaik yang mungkin untuk dilakukan.

CONTROL

TOPIC_CALLBACK

```
void topic_callback(const std_msgs::msg::Int16::SharedPtr msg) {
  RCLCPP_INFO(this->get_logger(), "Received data: %d", msg->data);

  int k;
  while(moves_ < 9){
    int player = msg->data - 1;

    if(board_[player] == 0){
      board_[player] = -1;
```

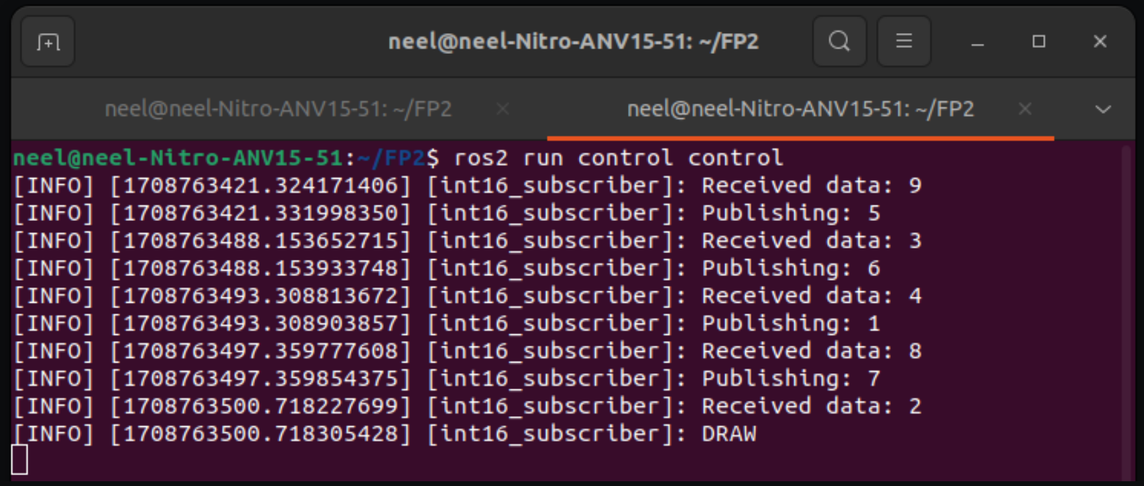
Fungsi ini digunakan untuk menerima pesan dari vision.



CONTROL


```
if(check_win() == -1){
  RCLCPP_INFO(this->get_logger(), "WIN");

  std_msgs::msg::String msg;
  msg.data = "WIN";
  publisher2_->publish(msg);
  break;

}else if(check_win() == 0 && moves_ == 8){
  RCLCPP_INFO(this->get_logger(), "DRAW");

  std_msgs::msg::String msg;
  msg.data = "DRAW";
  publisher2_->publish(msg);
  break;
}

moves_++;
```

Sintaks ini digunakan untuk mengecek apakah player menang atau seri. Apabila menang, pesan String berisi “WIN“ akan dikirim ke vision. Apabila seri. Apabila menang, pesan String berisi “WIN“ akan dikirim ke vision. Apabila seri, pesan String berisi “DRAW“ akan dikirim ke vision.

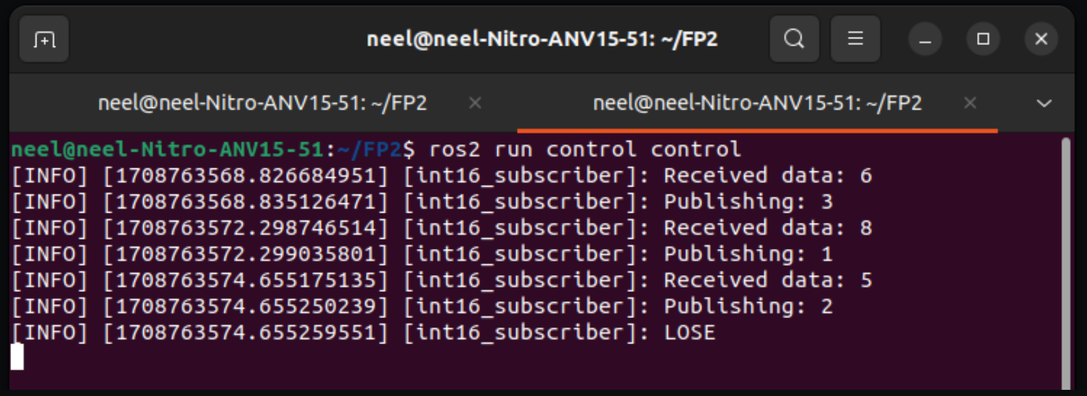


CONTROL

TOPIC_CALLBACK

```
if(check_win() == 0){
  k = computer_move();
  board_[k] = 1;

  RCLCPP_INFO(this->get_logger(), "Publishing: %d", k+1);

  if(check_win() == 1){
    RCLCPP_INFO(this->get_logger(), "LOSE");

    std_msgs::msg::String msg;
    msg.data = "LOSE";

    publisher2_->publish(msg);
  }

  std_msgs::msg::Int16 computer_msg;
  computer_msg.data = k + 1;
  publisher_->publish(computer_msg);

  moves_++;

  break;
```

Sintaks ini digunakan untuk memanggil fungsi computer_move untuk mendapatkan langkah dari computer dan mengirimkan pesan Integer berisi langkahnya kembali ke vision. Setelah itu, akan di cek apakah player kalah atau tidak. Apabila kalah, pesan String berisi “LOSE“ akan dikirim ke vision. Selain itu, pesan Integer yang berisi langkah dari computer juga akan dikirim ke vision.



THANK YOU!