

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



```
# 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

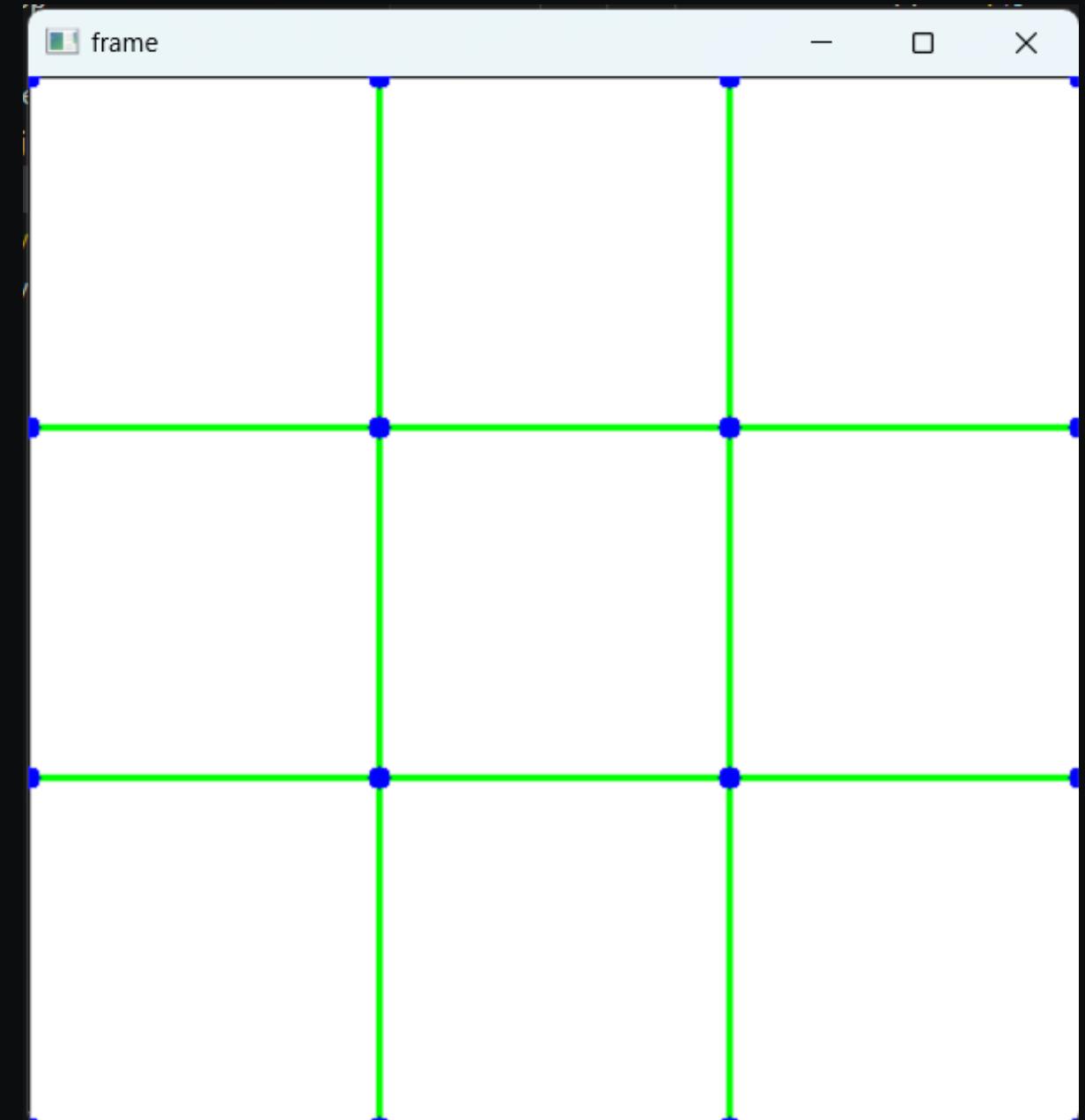
| VISION

GET_CORNERS_POINT

```
# 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

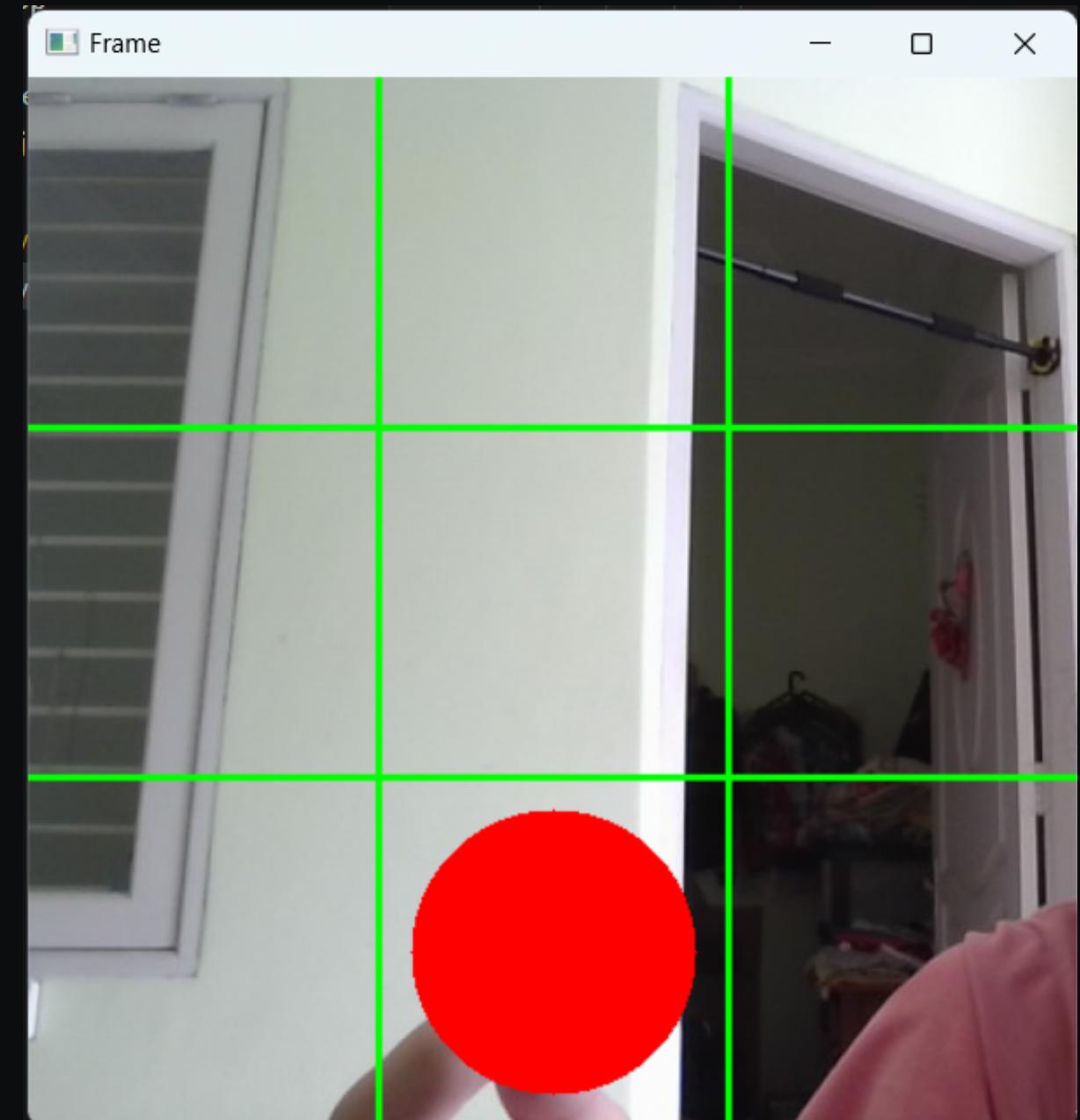
```
Corner points: [(0, 0), (160, 0), (320, 0), (480, 0), (0, 160), (160, 160), (320, 160), (480, 160), (0, 320), (160, 320), (320, 320), (480, 320), (0, 480), (160, 480), (320, 480), (480, 480)]
```

| VISION

PLAYER_MOVE

```
# 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

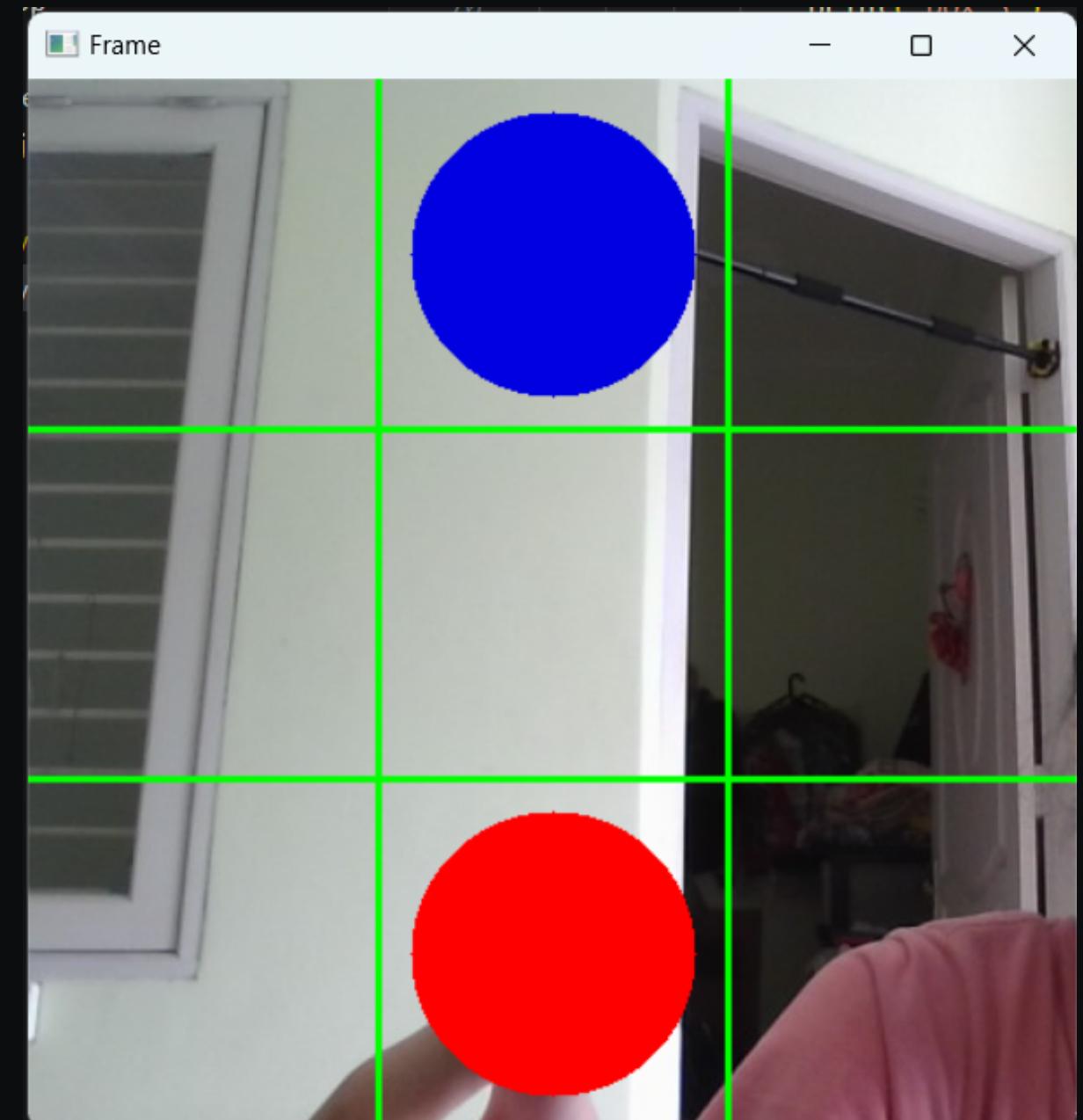


| VISION

COMPUTER_MOVE

```
# 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

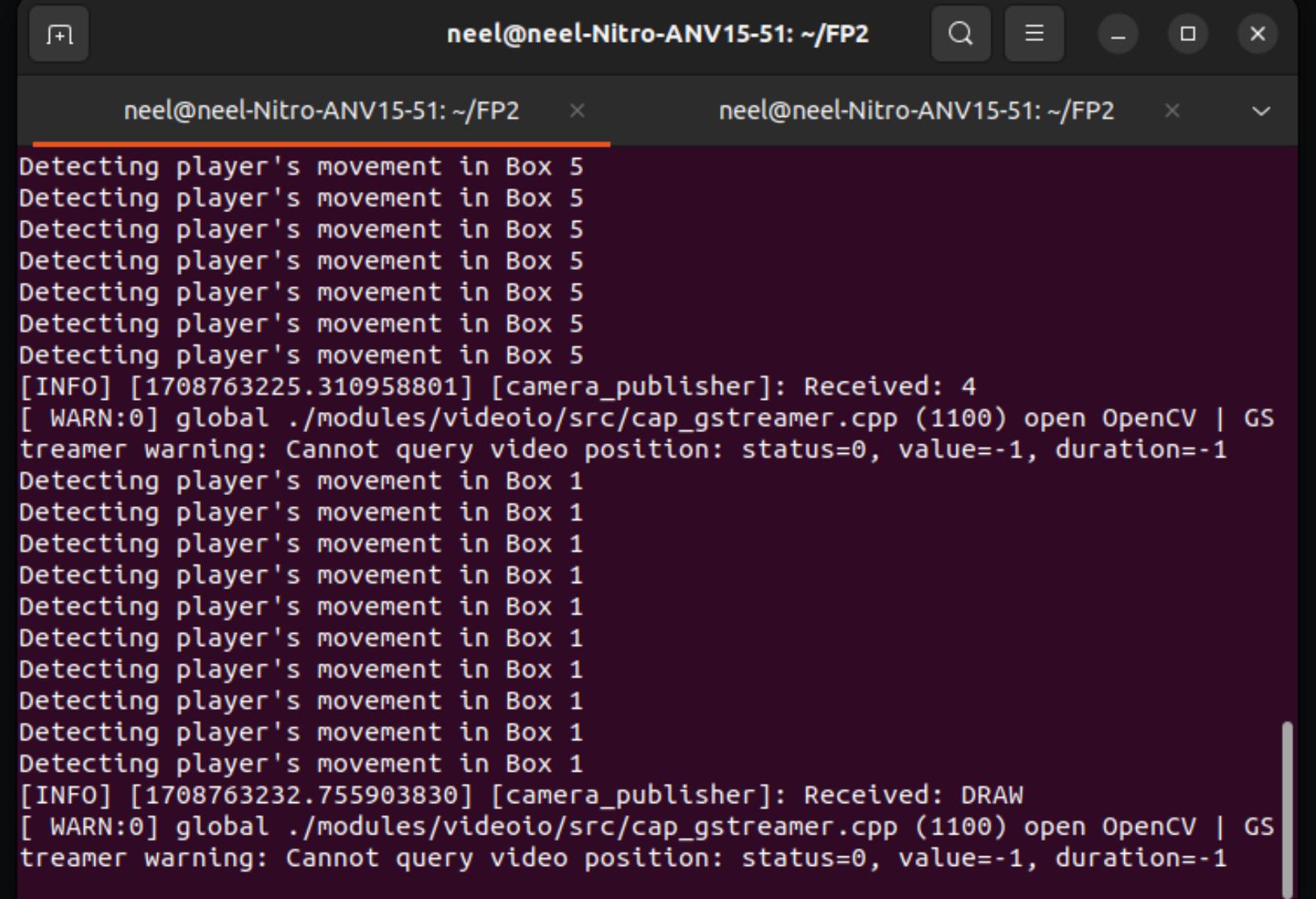
STRING_CALLBACK

```
# 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.



The terminal window displays two sessions. The left session shows repeated log entries from the 'camera_publisher' node: 'Detecting player's movement in Box 5'. The right session shows a log entry from the same node: '[INFO] [1708763225.310958801] [camera_publisher]: Received: 4'. A warning message follows: '[WARN:0] global ./modules/videoio/src/cap_gstreamer.cpp (1100) open OpenCV | GStreamer warning: Cannot query video position: status=0, value=-1, duration=-1'. The terminal has a dark theme with light-colored text and a dark background.

```
Detecting player's movement in Box 5
[INFO] [1708763225.310958801] [camera_publisher]: Received: 4
[ WARN:0] global ./modules/videoio/src/cap_gstreamer.cpp (1100) open OpenCV | GStreamer warning: Cannot query video position: status=0, value=-1, duration=-1
Detecting player's movement in Box 1
[INFO] [1708763232.755903830] [camera_publisher]: Received: DRAW
[ WARN:0] global ./modules/videoio/src/cap_gstreamer.cpp (1100) open OpenCV | GStreamer warning: Cannot query video position: status=0, value=-1, duration=-1
```

VISION

CALLBACK

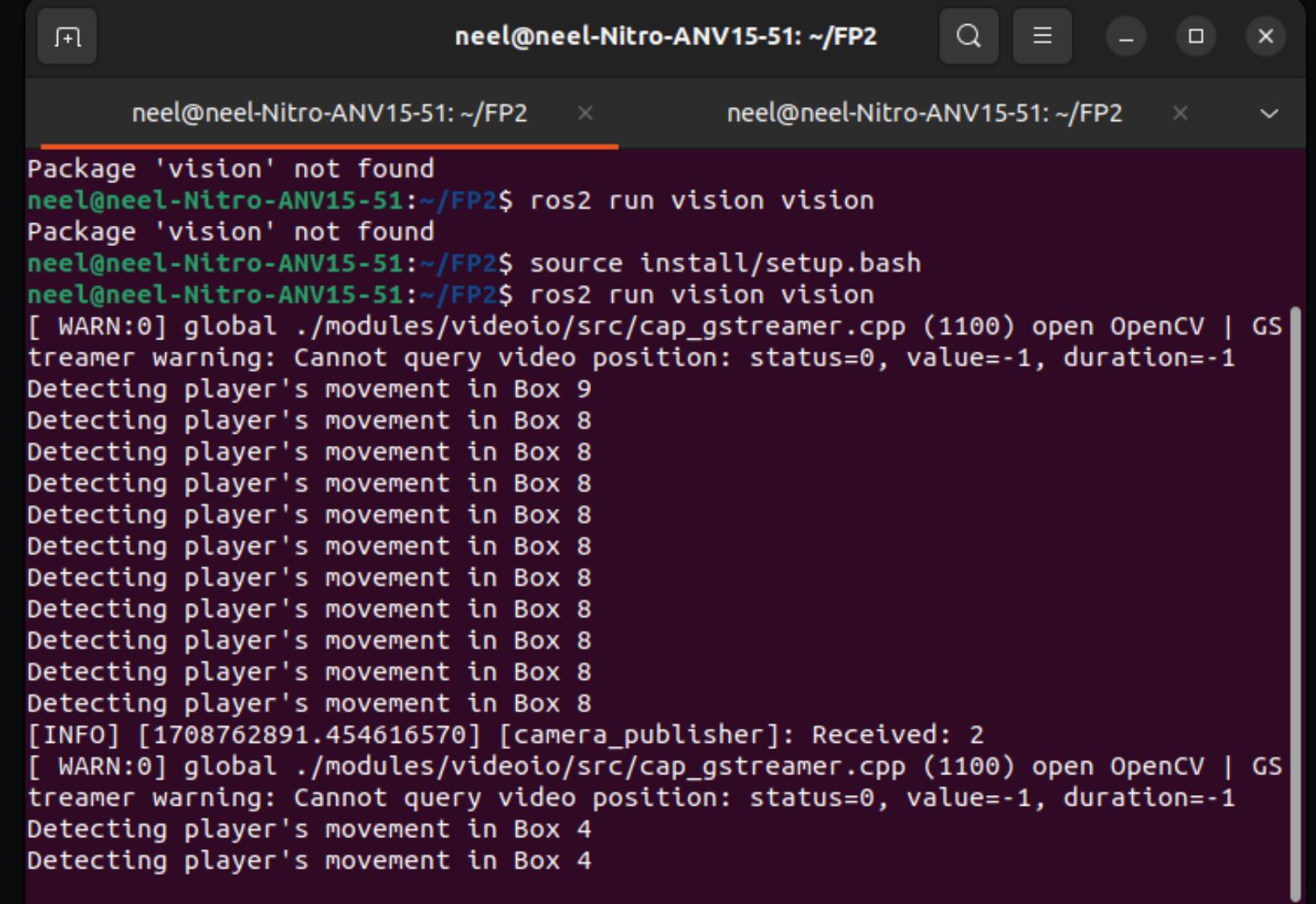
```
# Menerima pesan berupa Int dari control
def callback(self, msg):
    self.get_logger().info('Received: %d' % msg.data)
    computer = msg.data

    self.computer[computer] = 1

    self.moves += 1

    self.camera_frame()
```

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



The screenshot shows a terminal window with three tabs, each titled 'neel@neel-Nitro-ANV15-51: ~/FP2'. The first tab contains the command 'ros2 run vision vision'. The second tab contains the command 'source install/setup.bash'. The third tab contains the command 'ros2 run vision vision'. The output of the third tab shows several lines of text indicating the detection of player movement in boxes 9 and 8, followed by an INFO message about receiving a value of 2 from a camera publisher.

```
neel@neel-Nitro-ANV15-51: ~/FP2
neel@neel-Nitro-ANV15-51: ~/FP2
neel@neel-Nitro-ANV15-51: ~/FP2

Package 'vision' not found
neel@neel-Nitro-ANV15-51:~/FP2$ ros2 run vision vision
Package 'vision' not found
neel@neel-Nitro-ANV15-51:~/FP2$ source install/setup.bash
neel@neel-Nitro-ANV15-51:~/FP2$ ros2 run vision vision
[ WARN:0] global ./modules/videoio/src/cap_gstreamer.cpp (1100) open OpenCV | GSTREAMER warning: Cannot query video position: status=0, value=-1, duration=-1
Detecting player's movement in Box 9
Detecting player's movement in Box 8
[INFO] [1708762891.454616570] [camera_publisher]: Received: 2
[ WARN:0] global ./modules/videoio/src/cap_gstreamer.cpp (1100) open OpenCV | GSTREAMER warning: Cannot query video position: status=0, value=-1, duration=-1
Detecting player's movement in Box 4
Detecting player's movement in Box 4
```

VISION

CAMERA_FRAME

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

```
# 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")

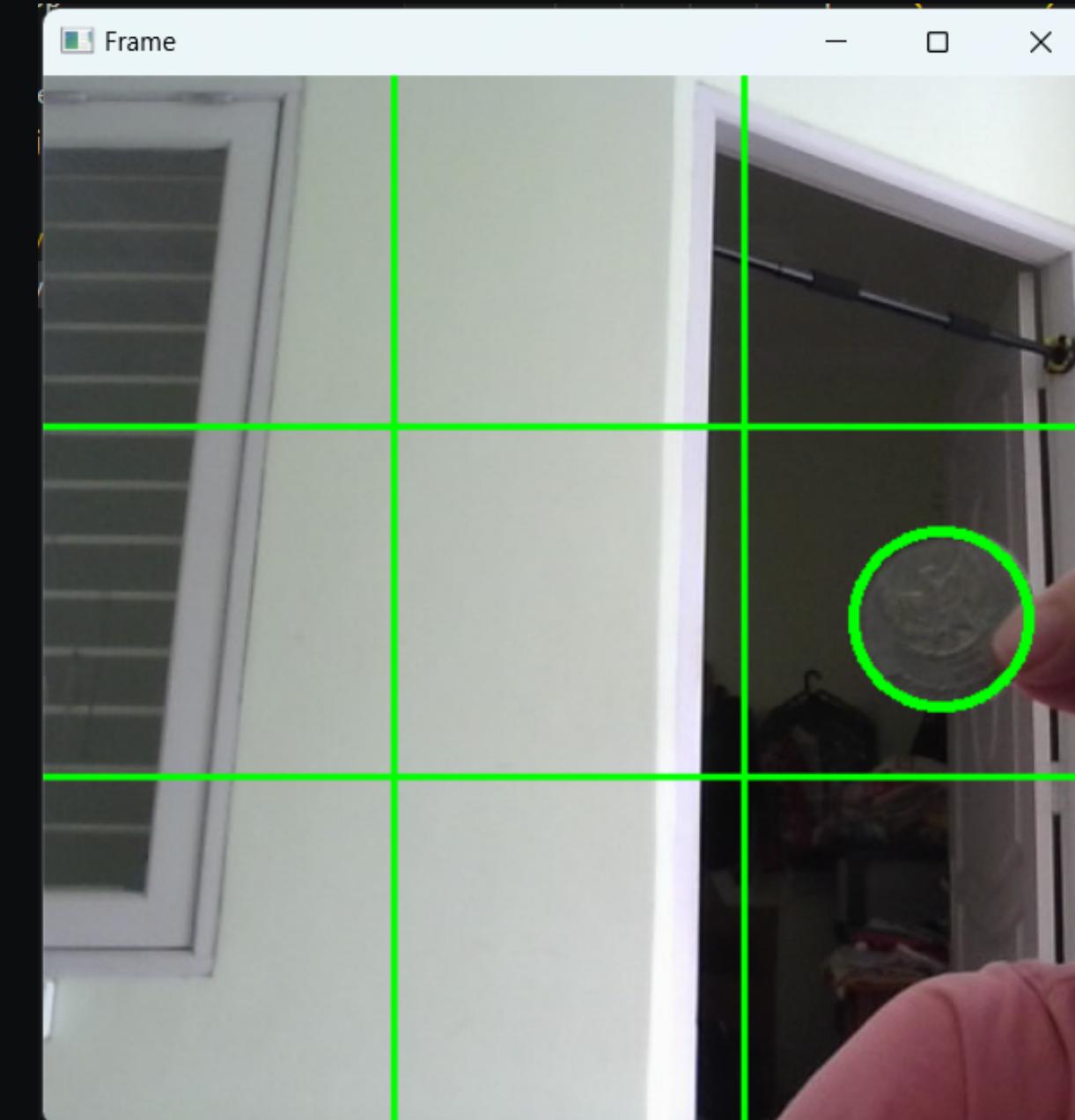
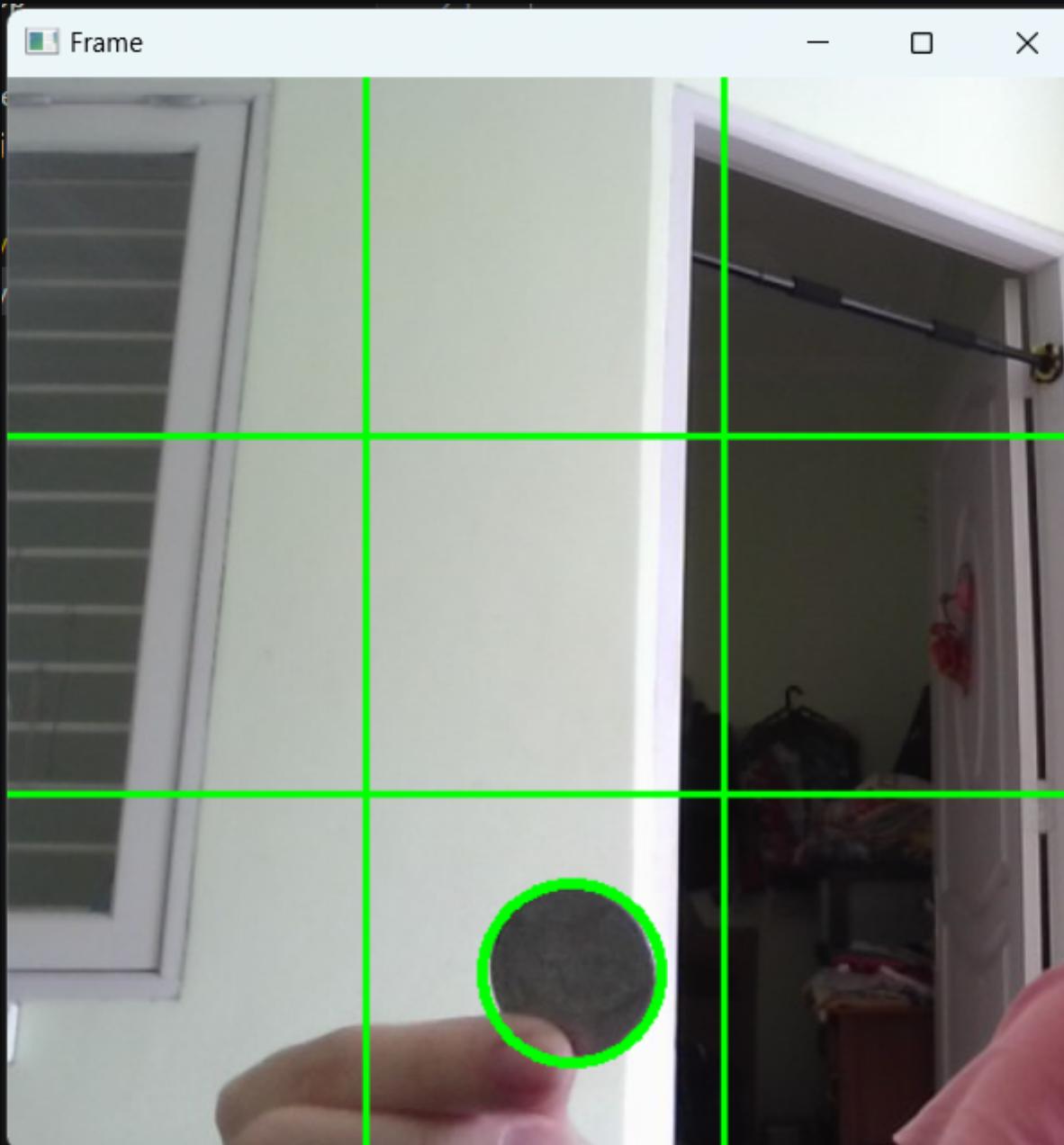
    for (x, y, r) in circles:
        cv2.circle(cropped_frame, (x, y), r, (0, 255, 0), 4)

        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

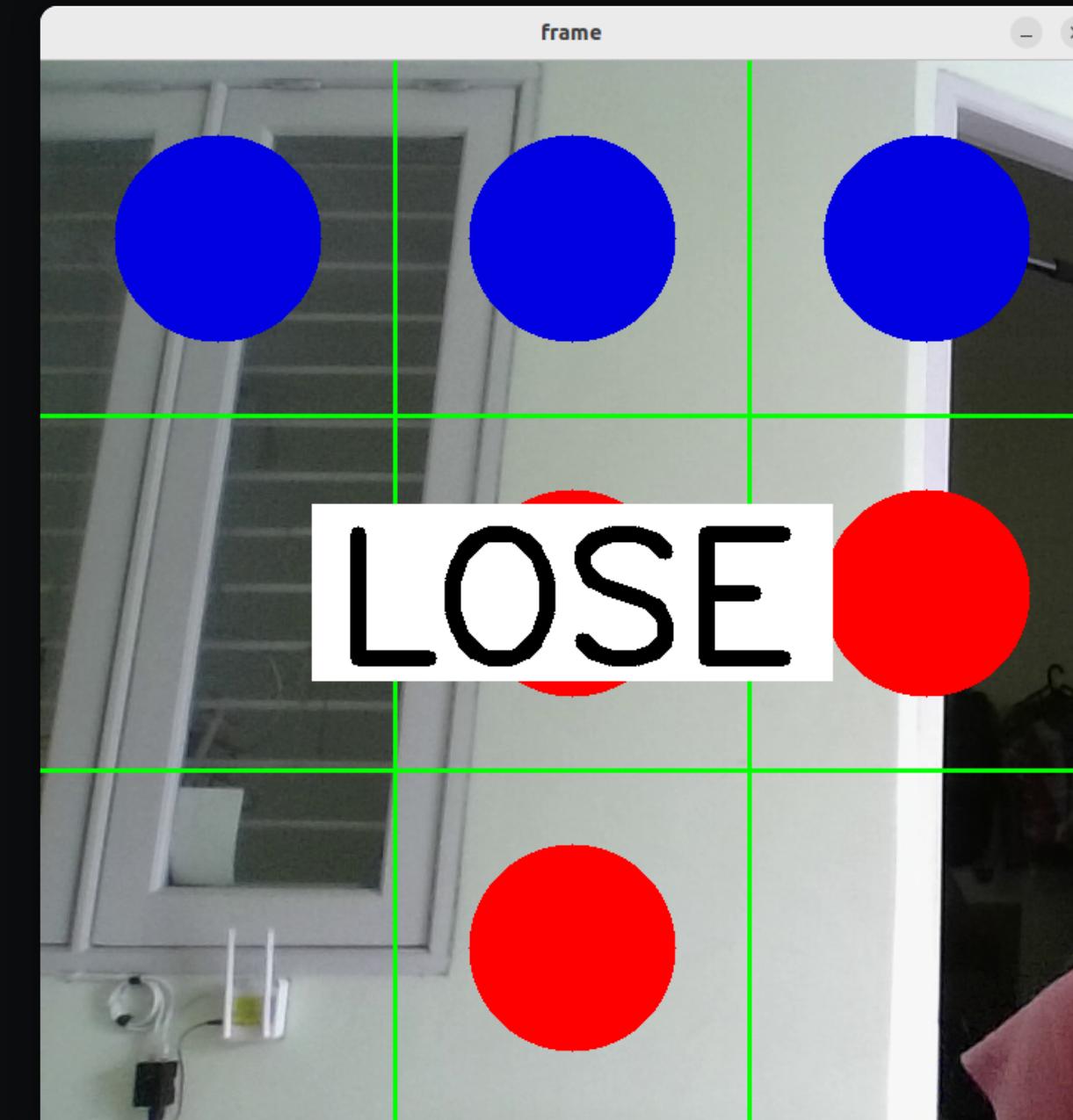
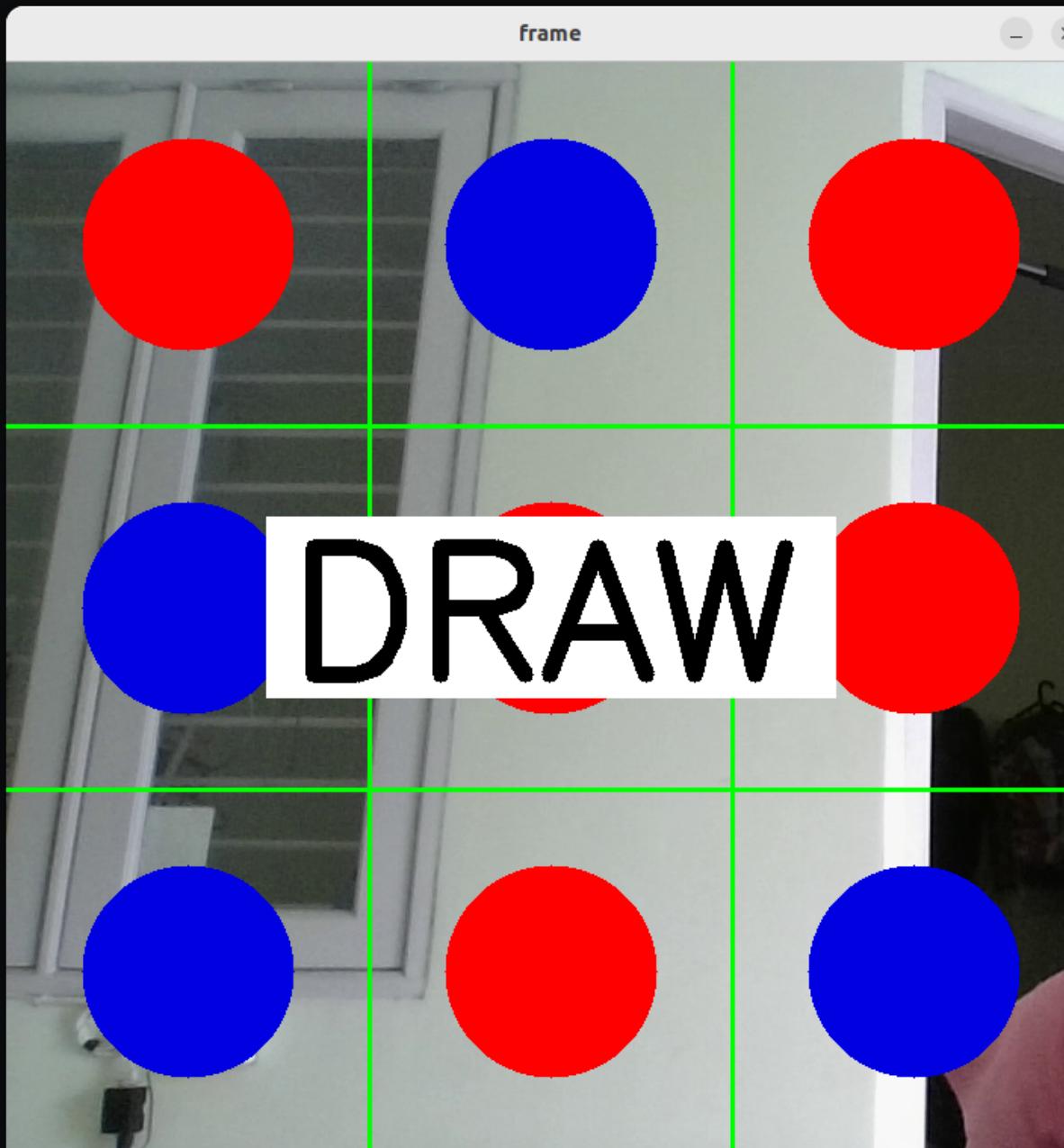
CAMERA_FRAME

```
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

MINIMAX

```
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

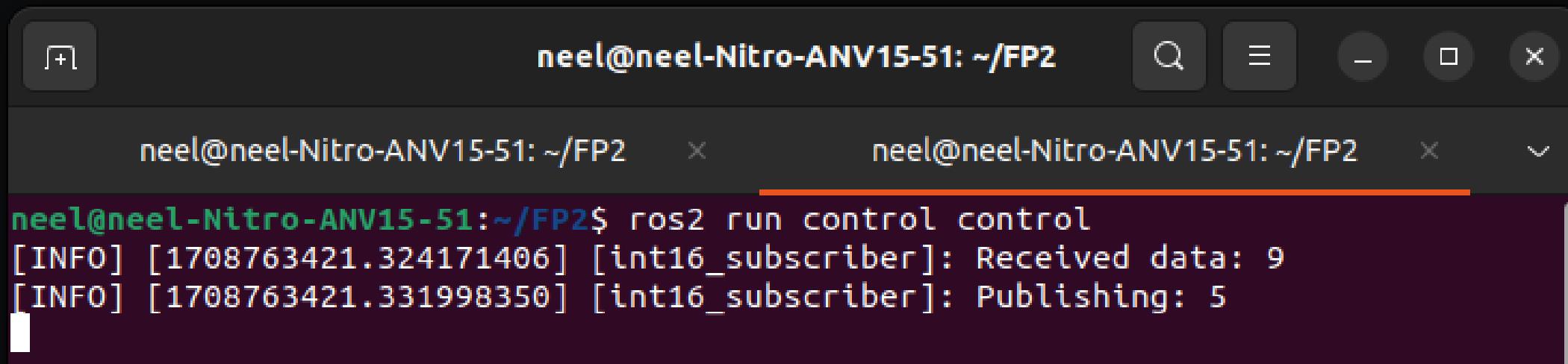
TOPIC_CALLBACK

```
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.



The screenshot shows a terminal window with two tabs open. The title bar of the window reads "neel@neel-Nitro-ANV15-51: ~/FP2". The left tab also has the same title. The right tab is active and shows the command "neel@neel-Nitro-ANV15-51:~/FP2\$ ros2 run control control". The output of the command is displayed below the command line, showing two INFO messages: "[INFO] [1708763421.324171406] [int16_subscriber]: Received data: 9" and "[INFO] [1708763421.331998350] [int16_subscriber]: Publishing: 5".

```
neel@neel-Nitro-ANV15-51: ~/FP2
neel@neel-Nitro-ANV15-51: ~/FP2 × neel@neel-Nitro-ANV15-51: ~/FP2 ×
neel@neel-Nitro-ANV15-51:~/FP2$ ros2 run control control
[INFO] [1708763421.324171406] [int16_subscriber]: Received data: 9
[INFO] [1708763421.331998350] [int16_subscriber]: Publishing: 5
```

CONTROL

TOPIC_CALLBACK

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

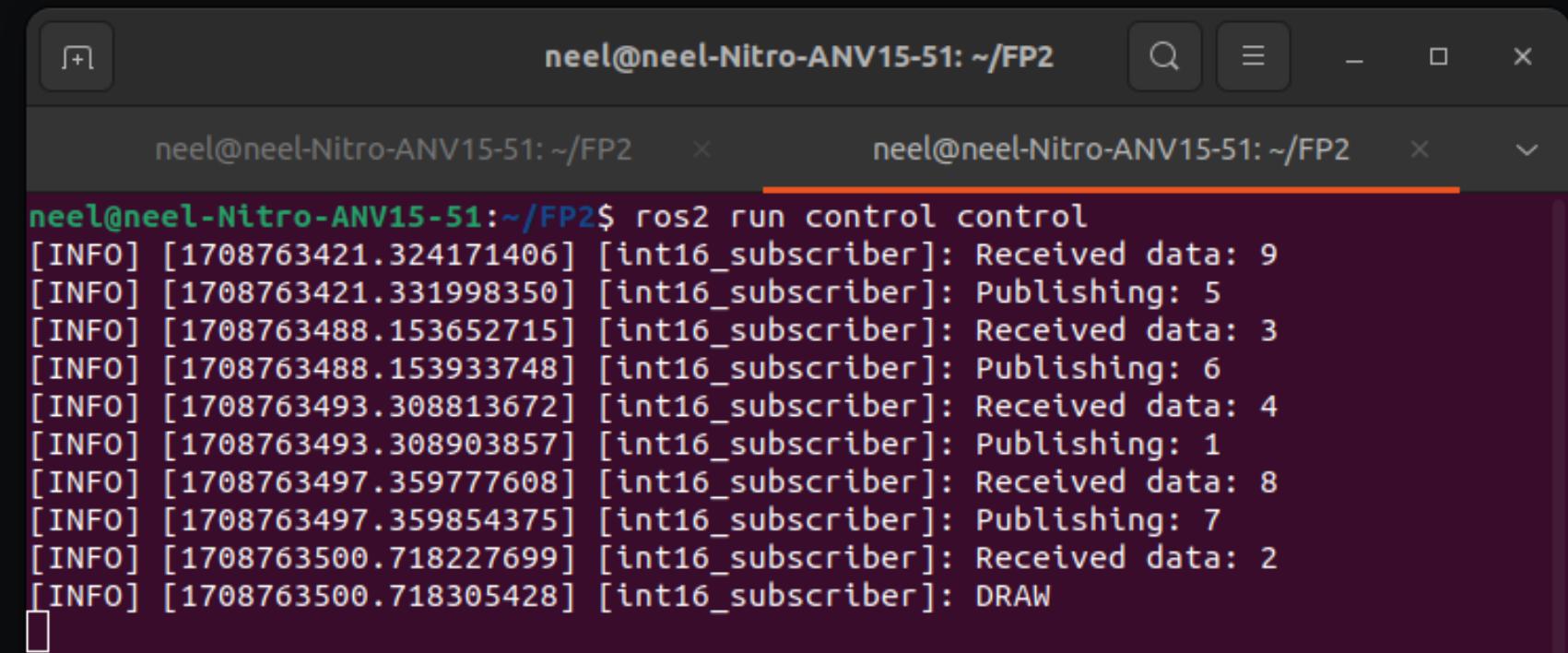
    std_msgs::msg::String msg;
    msg.data = "WIN";
    publisher2_->publish(msg);
    break;

}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.



A terminal window titled "neel@neel-Nitro-ANV15-51: ~/FP2" showing ROS2 log output. The log shows a sequence of messages from an int16_subscriber node, alternating between "Received data" and "Publishing" messages, with values ranging from 1 to 9. The last message is "INFO [1708763500.718305428] [int16_subscriber]: Publishing: DRAW".

```
neel@neel-Nitro-ANV15-51:~/FP2$ ros2 run control control
[INFO] [1708763421.324171406] [int16_subscriber]: Received data: 9
[INFO] [1708763421.331998350] [int16_subscriber]: Publishing: 5
[INFO] [1708763488.153652715] [int16_subscriber]: Received data: 3
[INFO] [1708763488.153933748] [int16_subscriber]: Publishing: 6
[INFO] [1708763493.308813672] [int16_subscriber]: Received data: 4
[INFO] [1708763493.308903857] [int16_subscriber]: Publishing: 1
[INFO] [1708763497.359777608] [int16_subscriber]: Received data: 8
[INFO] [1708763497.359854375] [int16_subscriber]: Publishing: 7
[INFO] [1708763500.718227699] [int16_subscriber]: Received data: 2
[INFO] [1708763500.718305428] [int16_subscriber]: Publishing: DRAW
```

CONTROL

```
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);
    }

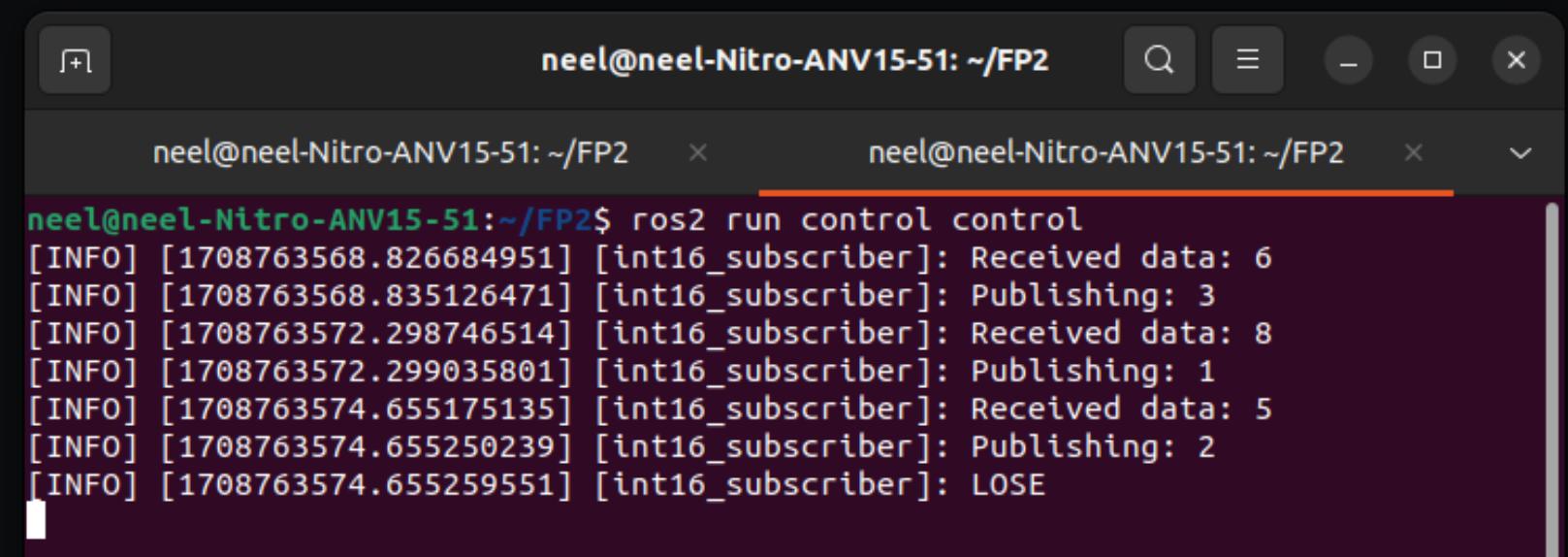
    std_msgs::msg::Int16 computer_msg;
    computer_msg.data = k + 1;
    publisher_->publish(computer_msg);

    moves_++;
}

break;
```

TOPIC_CALLBACK

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.



The terminal window shows three tabs, all titled 'neel@neel-Nitro-ANV15-51: ~/FP2'. The bottom tab is active and displays the following ROS2 log output:

```
neel@neel-Nitro-ANV15-51:~/FP2$ ros2 run control control
[INFO] [1708763568.826684951] [int16_subscriber]: Received data: 6
[INFO] [1708763568.835126471] [int16_subscriber]: Publishing: 3
[INFO] [1708763572.298746514] [int16_subscriber]: Received data: 8
[INFO] [1708763572.299035801] [int16_subscriber]: Publishing: 1
[INFO] [1708763574.655175135] [int16_subscriber]: Received data: 5
[INFO] [1708763574.655250239] [int16_subscriber]: Publishing: 2
[INFO] [1708763574.655259551] [int16_subscriber]: LOSE
```



**THANK
YOU!**