ДОДАТОК А

Код програми керування

Код програми для Raspberry Pi

```
#include <pigpio.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <unistd.h>
#include <string.h>
// Motor 1 - Right motor
#define M1_B 17 /* Physical pin 11 */
#define M1_A 27 /* Physical pin 13 */
#define M1_PWM 13 /* Physical pin 33 */
/* Motor 2 - Left motor*/
#define M2_B 22 /* Physical pin 15 */
#define M2_A 23 /* Physical pin 16 */
#define M2_PWM 12 /* Physical pin 32 */
#define HIGH 1
#define LOW 0

void stopRobot() {
  gpioWrite(M1_A, HIGH);
  gpioWrite(M1_B, HIGH);
  gpioWrite(M2_A, HIGH);
  gpioWrite(M2_B, HIGH);
  gpioPWM(M1_PWM, 0);
  gpioPWM(M2_PWM, 0);
}
void shutdownRobot(){
```

```
  stopRobot();
  gpioTerminate();
}
void setupRobot() {
  if (gpioInitialise() < 0) {
    printf("Initialisation failed!\n");
    exit(1);
  }
  gpioSetMode(M1_A, PI_OUTPUT);
  gpioSetMode(M1_B, PI_OUTPUT);
  gpioSetMode(M2_A, PI_OUTPUT);
  gpioSetMode(M2_B, PI_OUTPUT);
  gpioSetMode(M1_PWM, PI_ALT0);
  gpioSetMode(M2_PWM, PI_ALT0);
}
void go_forward(unsigned short PWM) {
  gpioWrite(M1_A, HIGH);
  gpioWrite(M1_B, LOW);
  gpioWrite(M2_A, LOW);
  gpioWrite(M2_B, HIGH);
  gpioPWM(M1_PWM, PWM);
  gpioPWM(M2_PWM, PWM);
}
void go_back(unsigned short PWM) {
  gpioWrite(M1_A, LOW);
  gpioWrite(M1_B, HIGH);
  gpioWrite(M2_A, HIGH);
  gpioWrite(M2_B, LOW);
  gpioPWM(M1_PWM, PWM);
  gpioPWM(M2_PWM, PWM);
}
void turn_left(unsigned short PWM) {
  gpioWrite(M1_A, HIGH);
  gpioWrite(M1_B, LOW);
  gpioWrite(M2_A, LOW);
```

```
  gpioWrite(M2_B, HIGH);
  gpioPWM(M1_PWM, PWM);
}
void turn_right(unsigned short PWM) {
  gpioWrite(M1_A, LOW);
  gpioWrite(M1_B, HIGH);
  gpioWrite(M2_A, LOW);
  gpioWrite(M2_B, HIGH);
  gpioPWM(M2_PWM, PWM);

int main(int argc, char *argv[]){
    const int port = 8080;
    int server_socket;
    server_socket = socket(AF_INET, SOCK_STREAM, 0);
    /* Specify adress for the socket */
    struct sockaddr_in server_address, client_address;
    server_address.sin_family = AF_INET;
    server_address.sin_addr.s_addr = INADDR_ANY;
    server_address.sin_port = htons(port);
    if (bind (server_socket, (struct sockaddr *) &server_address,
sizeof(server_address)) == -1){
        perror ("Binding error!");
        close (server_socket);
        exit (1);
    }
    if (listen(server_socket, 1) == -1){
        perror("Listening error!");
        close(server_socket);
        exit(1);
    }

    int client_socket;
    socklen_t sin_len = sizeof(client_address);
    char request_buffer[1]; // buffer for working with client's requests
    // Prepare robot
```

```
        setupRobot();
        while(1){
          /* Getting size of cliens address */
          client_socket = accept(server_socket, (struct sockaddr *) &client_address,
&sin_len);
          if(client_socket == -1){
            perror("Connection failed\n");
            //exit(1);
            continue;
          }
          /* Recieve command from client */
          memset(request_buffer,0,1); // zeroing out the buffer
          read(client_socket,request_buffer,1);
          //printf("Recieved: %d\n",request_buffer[0]);
          /* Perform movement with robot */
          switch (request_buffer[0]) {
            case 1:
              go_forward(255);
              break;
            case 2:
              turn_right(255);
              break;
            case 3:
              go_back(255);
              break;
            case 4:
              turn_left(255);
              break;
            case 5:
              stopRobot();
              break;
            case 6:
              printf("Recived shutdown request. Quitting...");
              shutdownRobot();
              printf("Closing server socket...\n");
```

```
                close(server_socket);
                exit(0);
                break;
            default:
                printf("Error: Unrecognized command: %d\n",request_buffer[0]);
                break;
        }
        close(client_socket);
    }
    printf("Closing server socket...\n");
    close(server_socket);
    return 0;
}
```

Код програми на Java

```
import java.awt.event.*;
import java.awt.*;
import java.awt.BorderLayout;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextArea;
import javax.swing.JTextField;

public class MyFrame extends JFrame{

    private static final long serialVersionUID = 1L;
    byte previousRequest = 0;
    MyFrame(String host, int port){
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JTextArea log = new JTextArea("Use WASD keys to control the
robot\n");
        JPanel centralPanel = new JPanel();
        ImagePanel imagePanel = new ImagePanel();
```

```
            JTextField commandField =new JTextField();
            //JLabel label = new JLabel("Label");

            /**
             * 1 - forward
             * 2 - right
             * 3 - back
             * 4 - left
             * 5 - stop
             * 6 - stop server
             */
            commandField.addKeyListener(new KeyListener() {
               public void keyPressed(KeyEvent e) {
                  commandField.setText("");
                  if (e.getKeyCode() == KeyEvent.VK_W && previousRequest !=1) {
                     log.append("Forward key pressed: Moving forward\n");
                     previousRequest = 1;
                     Connection.connect(host, port,previousRequest);
                  } else if (e.getKeyCode() == KeyEvent.VK_D && previousRequest
!=2) {
                     log.append("Right key pressed: Moving right\n");
                     previousRequest = 2;
                     Connection.connect(host, port,previousRequest);
                  } else if (e.getKeyCode() == KeyEvent.VK_S && previousRequest
!=3) {
                     log.append("Back key pressed: Moving back\n");
                     previousRequest = 3;
                     Connection.connect(host, port,previousRequest);
                  } else if (e.getKeyCode() == KeyEvent.VK_A && previousRequest
!=4) {
                     log.append("Left key pressed: Moving left\n");
                     previousRequest = 4;
                     Connection.connect(host, port,previousRequest);
```

```
                } else if (e.getKeyCode() == KeyEvent.VK_Q && previousRequest
!=6) {
                    log.append("Sent shutdown request to server.\n");
                    previousRequest = 6;
                    Connection.connect(host, port,previousRequest);
                }
            }

            public void keyReleased(KeyEvent e) {
                if ((e.getKeyCode() == KeyEvent.VK_W || e.getKeyCode() ==
KeyEvent.VK_D || e.getKeyCode() == KeyEvent.VK_S ||
                        e.getKeyCode() == KeyEvent.VK_A) && previousRequest
!=5) {
                    log.append("Key released: Stoping\n");
                    previousRequest = 5;
                    Connection.connect(host, port,previousRequest);
                }
            }

            public void keyTyped(KeyEvent e) { ; }
        });

        //imagePanel.setBackground(Color.BLUE);
        //imagePanel.setSize(600, 600);
        imagePanel.setMinimumSize(new Dimension(400,400));
        //imagePanel.add(canvas, null);
        //imagePanel.setVisible(true);
        //frame.getContentPane().add(panel);

        centralPanel.setLayout(new BorderLayout());
        centralPanel.add(log, BorderLayout.EAST);
        centralPanel.add(imagePanel, BorderLayout.CENTER);
        setSize(700,600);
        setLayout(new BorderLayout());
        add(centralPanel,BorderLayout.CENTER);
```

```
            //add(centralPanel, BorderLayout.NORTH);

            add(commandField,BorderLayout.SOUTH);
            setVisible(true);
        }
    };

    import java.io.*;
    import java.net.*;

    class Connection{
        public static void connect(String host, int port, byte command){
            try{
                Socket mySocket;
                mySocket = new Socket(host,port);
                /* Send the command to server */
                DataOutputStream clientOutput = new
DataOutputStream(mySocket.getOutputStream());
                clientOutput.writeByte(command);
                mySocket.close();
            } catch(Exception e){
                System.out.println(e);
            }
        }
    };

    public class Main {
        public static void main(String[] args){
            final String host = "192.168.0.10";
            final int port = 8080;
            new MyFrame(host, port);
        }
    };
```

ДОДАТОК Б

Демонстраційні матеріали

ДОДАТОК В

Відомість атестаційної роботи

№	Позначення	Найменування	Дод. відомості
		Текстові документи	
1	ГЮИК. 442576.005 ПЗ	Пояснювальна записка	82 с., ф. А4
2		Додаткові документи	
		Демонстраційні матеріали	19 с., у форматі *.ppt

					ГЮИК. 442576.005 ВД			
Змін.	Арк.	№ документа	Підп.	Дата				
Розробив		Коротенко І.В.			Система дистанційного аналізу теплових властивостей технічних об'єктів – апаратна частина Відомість атестаційної роботи	Літера	Аркуш	Аркушів
Перевірив		Цимбал О. М.						1
						ХНУРЕ кафедра КІТАМ		
Н. контроль		Яшков І.О.						
Затвердив		Невлюдов І.Ш.						