# JoyMan

-Deepak Soni

Objective:

Controlling a bot wirelessly by reading motion and gestures of a person using

Microsoft Kinect.

Prerequisites:

* Basics of object oriented programming in C#
* Basics of UART communication.
* Arduino

Hardware Requirements:

* Microsoft Kinect 1.0
* Arduino Mega 2560
* HC-05 Bluetooth module
* An Omni or Mecanum drive
* A Windows computer

Approach:

In this project, Kinect is used as a 3D camera to detect a person’s head’s position in space so we get the x,y and z positions. When the programme initialises the first position of the head is taken as the reference for the rest of the programme. With this reference and current head position we calculate the inverse tan of the angle and send it to the bot. In the bot side the programme is written to move the bot at an angel received. Along with the head person’s hand’s y-coordinate is also monitored if it is raised enough the hit command is given to the bot, In this case it is a badminton playing bot and moves the racket on hit command.

Kinect

Windows Computer

With Bluetooth

HC-05 Bluetooth module

Arduino

Person

Camera

Wired USB

Wireless Bluetooth

UART at Serial1

Bot side

Kinect side

Request data and wait until it comes

Sends latest data and wait until next request

Key Codes Snippets:

In KinectSkeleton.cs

foreach (JointMapping joint in this.jointMappings.Values)

{

Joint myrightHand = this.currentSkeleton.Joints[JointType.Head];

Joint myStallRead = this.currentSkeleton.Joints[JointType.HandLeft];

Joint myHitRead = this.currentSkeleton.Joints[JointType.HandRight];

float rightX = myrightHand.Position.X;

float rightY = myrightHand.Position.Y;

float rightZ = myrightHand.Position.Z;

float hitZ = myHitRead.Position.Y;

float handY = myStallRead.Position.Y;

Reference set for the first time than bl becomes true

if (!bl)

{

refX = rightX;

refY = rightY;

refZ = rightZ;

Getting the relative positions

refHandY = handY;

refHitZ = hitZ;

bl = true;

cc.writeStr("\n");

//break;

Getting atan value to get direction

}

rightX = rightX - refX;

rightY = rightY - refY;

rightZ = rightZ - refZ;

handY = handY - refHandY;

hitZ = hitZ - refHitZ;

tanVal = (float)Math.Atan2(rightZ, rightX);

Indicates data request from arduino

if (tanVal < 0)

{

tanVal += (float)(Math.PI \* 2);

}

int j = 0;

Stalling if movement is not significant enough

if (cc.isAvailable())

{

str = cc.readStr();

j = (int)(tanVal \* 100);

if (Math.Sqrt(Math.Pow(rightZ, 2) + Math.Pow(rightX, 2)) <0.1)

{

j = j + 1000;

Addition of 1000 indicates stall command

}

else

if (handY > 0.7)

{

j = j + 1000;

}

Addition of 10000 indicates hit command

if (hitZ > 0.7)

{

j = j + 10000;

}

str = j.ToString();

Sending data from ComCon class

cc.writeStr(str);

cc.writeStr("\n");

}

}

In ComCon.cs

class ComCon

{

public static bool isavailable = false;

public static String buffer = "";

static SerialPort \_serialPort = new SerialPort();

static Thread readThread = new Thread(Read);

public ComCon()

{

StringComparer stringComparer = StringComparer.OrdinalIgnoreCase;

\_serialPort.PortName = "COM40";

\_serialPort.BaudRate = 38400;

Setting Port information, our us COM40 at 38400 baud rate, else is left default

\_serialPort.Parity = \_serialPort.Parity;

\_serialPort.DataBits = \_serialPort.DataBits;

\_serialPort.StopBits = \_serialPort.StopBits;

\_serialPort.Handshake = \_serialPort.Handshake;

\_serialPort.ReadTimeout = 500;

\_serialPort.WriteTimeout = 500;

try

{

\_serialPort.Open();

}

catch

{

}

There is no inbuilt isAvailable() function so we run a thread to continuously read and update the variables

readThread.Start();

}

public void writeStr(String message)

{

\_serialPort.Write(message);

}

public String readStr()

{

String tmpp = buffer;

buffer = "";

isavailable = false;

On data read isAvailable variable is cleared

variable is cleared

return tmpp;

}

public bool isAvailable()

{

return isavailable;

}

Thread to continuously read data and set is available if new data comes storing data in buffer variable

public static void Read()

{

while (true)

{

try

{

buffer = buffer + \_serialPort.ReadByte();

isavailable = true;

}

catch (TimeoutException) { }

}

}

}

In remote.ino (bot side)

void remoteread()

Sends character ‘a’ to indicate data request

{

Serial1.write('a');

while(!Serial1.available())

{

}

Waits until new data comes

angle = Serial1.parseInt();

if(angle>700)

{

//Serial.print("Stall");

stall();

Stalls if 1000 is added as angle goes to from 0-628

}

else

{

angle = ((angle)/100);

magFeed(60,angle);

}

magFeed is the navigation function with pwm and angle ans arguements

//Serial.println(angle);

}

Links:

<https://github.com/Deepak61900/JoyMan>

<https://www.youtube.com/watch?v=8Ln2ocw7RKQ>

Email id: deepak61900@gmail.com

The Verdict:

This project is currently reading the motion and gestures of the body parts, sound input can also be implemented easily. Kinect is a very useful piece of hardware and does the job quiet perfectly. The concept used in this project can be used to develop things like Surrogate soldiers, Shadow mode from the movie ‘Real Steel’ and much more.