HTML Document (Interface Structure Layer)

<!DOCTYPE html>

<html lang="ar" dir="ltr">

<head>

<title>

Graduation Project

</title>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="shortcut icon" type="imagex-icon" href="{{ url\_for("static", filename="favicon.ico") }}" />

<link rel="stylesheet" href="{{ url\_for("static", filename="libs/normalize.min.css") }}" />

<link rel="stylesheet" href="{{ url\_for("static", filename="libs/bootstrap.min.css") }}" />

<link rel="stylesheet" href="{{ url\_for("static", filename="styles.css") }}" />

</head>

<body class="">

<div class="container">

<div class="row">

<div class="col-xs-12 section">

<h1>Graduation Project</h1>

</div>

</div>

<div class="row">

<div class="col-sm-12 col-md-6 stream-container section">

<img src="{{ url\_for('video\_feed') }}" data-src="{{ url\_for('video\_feed') }}">

</div>

<div class="col-sm-12 col-md-6">

<div class="row">

<div class="col-xs-12 section table-wrapper">

<table class="table arrows">

<tr>

<td class="sides left-side" rowspan="2">

<span class="btn" id="left-arrow" data-arrow="left">

left

</span>

</td>

<td>

<span class="btn" id="up-arrow" data-arrow="up">

up

</span>

</td>

<td class="sides right-side" rowspan="2">

<span class="btn" id="right-arrow" data-arrow="right">

right

</span>

</td>

<td class="sides" rowspan="2">

<span class="btn" id="recognize" data-arrow="enter">

detect

</span>

</td>

</tr>

<tr>

<td>

<span class="btn" id="down-arrow" data-arrow="down">

down

</span>

</td>

</tr>

</table>

</div>

<div class="col-xs-12 section console-wrapper">

<div class="console">

<pre></pre>

</div>

</div>

</div>

</div>

</div>

<div class="row">

<div class="col-xs-12" id="error\_container">

</div>

</div>

</div>

<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

<script type="text/javascript" src="{{ url\_for("static", filename="scripts.js") }}"></script>

</body>

</html>

Cascade Style Sheet (Interface Presentation Layer)

body {

min-height: 100vh;

}

img {

max-width: 100%;

}

.section {

position: relative;

margin-bottom: 20px;

}

.stream-container {

text-align: center;

}

.stream-container img {

width: 100%;

height: 100%;

border:1px solid #16191d;

border-radius: 4px;

}

.table>tbody>tr>td, .table>tbody>tr>th, .table>tfoot>tr>td, .table>tfoot>tr>th, .table>thead>tr>td, .table>thead>tr>th {

border: 0;

padding: 5px

}

.arrows {

table-layout: fixed;

margin-bottom: 0

}

.arrows tr:first-child td {

padding-top: 0;

}

.arrows tr:last-child td {

padding-bottom: 0;

}

.arrows .sides {

padding-bottom: 0

}

.arrows .sides.left-side {

padding-left: 0;

}

.arrows .sides.right-side {

padding-right: 0;

}

.arrows .btn {

width: 100%;

background: #21252b;

color: white;

line-height: 50px;

padding: 0;

font-size: 2em;

border: 0;

}

.arrows .btn:hover, .arrows .btn:focus, .arrows .btn.pushed {

background: #16191d;

}

.arrows .sides .btn {

line-height: 110px;

}

.console-wrapper .console {

background: #16191d;

color: #ffffff;

width: 100%;

height: 290px;

border-radius: 4px;

padding: 10px;

text-align: left

}

.console-wrapper .console pre {

background-color: inherit;

color: inherit;

border: 0;

border-radius: 0;

padding: 0;

padding-right: 5px;

margin: 0;

width: 100%;

max-height: 100%;

overflow-x: hidden;

overflow-y: auto;

white-space: pre-wrap; /\* css-3 \*/

white-space: -moz-pre-wrap; /\* Mozilla, since 1999 \*/

white-space: -pre-wrap; /\* Opera 4-6 \*/

white-space: -o-pre-wrap; /\* Opera 7 \*/

word-wrap: break-word;

}

.console-wrapper .console pre::-webkit-scrollbar {

background-color: #282c34;

border-radius: 2px;

width: 6px;

}

.console-wrapper .console pre::-webkit-scrollbar-thumb{

background-color: #06090d;

border-radius: 2px;

}

@media(min-width: 991px) {

.console-wrapper.section {

top: -130px;

}

.table-wrapper.section {

bottom: -310px;

}

.stream-container {

height: 420px;

}

}

Javascript (Interface Behavior Layer)

/\* Global Functions \*/

Object.compare = function (obj1, obj2) {

    //Loop through properties in object 1

    for (var p in obj1) {

        //Check property exists on both objects

        if (obj1.hasOwnProperty(p) !== obj2.hasOwnProperty(p)) return false;

        switch (typeof (obj1[p])) {

            //Deep compare objects

            case 'object':

                if (!Object.compare(obj1[p], obj2[p])) return false;

                break;

            //Compare function code

            case 'function':

                if (typeof (obj2[p]) == 'undefined' || (p != 'compare' && obj1[p].toString() != obj2[p].toString())) return false;

                break;

            //Compare values

            default:

                if (obj1[p] != obj2[p]) return false;

        }

    }

    //Check object 2 for any extra properties

    for (var p in obj2) {

        if (typeof (obj1[p]) == 'undefined') return false;

    }

    return true;

};

/\* Global Variables \*/

// Elements selectors

selectors = {

console: ".console pre",

stream: ".stream-container img",

errorContainer: "#error\_container",

arrows: {

general: ".arrows .btn",

left: ".arrows #left-arrow",

up: ".arrows #up-arrow",

right: ".arrows #right-arrow",

down: ".arrows #down-arrow",

enter: ".arrows #recognize"

}

};

// pressed buttons

pressedArrows = {

left: 0,

up: 0,

right: 0,

down: 0

};

recognizeKeyPressed = false;

// states

baseURL= window.location.protocol + "//" + window.location.host + "/";

/\* Bind functions to events when document is ready \*/

$( document ).ready(function() {

// Check if stream kicked off

checkStream();

//Bind keys to callbacks

arrowKeysBinding({

left: arrowKeyBehavior("left"),

up: arrowKeyBehavior("up"),

right: arrowKeyBehavior("right"),

down: arrowKeyBehavior("down"),

enter: {

press: function() {

if(!recognizeKeyPressed) {

recognizeKeyPressed = true;

$(selectors.arrows.enter).addClass("pushed");

recognize();

}

},

release: function() {

recognizeKeyPressed = false;

$(selectors.arrows.enter).removeClass("pushed");

}

}

});

window.setInterval(function(){

checkStream();

synchronize();

}, 1000);

});

/\* Global Functions \*/

function writeToConsole(textToWrite) {

var $console = $(selectors.console),

startSymbol = '>',

scrollBottomPosition = $console.prop("scrollHeight") - $console.scrollTop() - $console.outerHeight();

$console.append(startSymbol+" "+textToWrite+"\n");

// scroll to bottom if the console is already at bottom

if(scrollBottomPosition < 20) {

$console.scrollTop($console.prop("scrollHeight"));

}

return true;

}

checkStream = (function(){

var streamIsUp = false,

img = new Image(),

streamURL = $(selectors.stream).attr('data-src');

img.onload = function() {

if(!streamIsUp){

streamIsUp = true;

$(selectors.stream).attr('src', streamURL);

writeToConsole('Stream is Ready!');

}

}

img.onerror = function(){

if(streamIsUp){

streamIsUp = false;

writeToConsole('Stream has Error!');

}

}

return function() {

img.src = streamURL;

return true;

}

})();

function arrowKeyBehavior(arrow) {

return {

press: function() {

if(!pressedArrows[arrow]) {

pressedArrows[arrow] = 1;

$(selectors.arrows[arrow]).addClass("pushed");

synchronize();

}

},

release: function() {

if(pressedArrows[arrow]) {

pressedArrows[arrow] = 0;

$(selectors.arrows[arrow]).removeClass("pushed");

synchronize();

}

}

};

}

// This functions sends current states to the server and ask for Robot state to display on console

synchronize = (function () {

var syncResponse = {

waiting: false,

hadError: false,

state: ''

};

var moveInstructions = {};

var oldmoveInstructions = {};

var secondsCounter = 0;

return function(){

if(syncResponse.waiting) { return }

secondsCounter++;

moveInstructions = calculateMoveInstructions();

if(secondsCounter < 4 && Object.compare(oldmoveInstructions, moveInstructions)) {return}

oldmoveInstructions = jQuery.extend({}, moveInstructions);

if(secondsCounter <= 4) {

secondsCounter=0;

}

syncResponse.waiting = true;

$.ajax({

method: "POST",

url: baseURL + "move",

data: moveInstructions

})

.done(function(result) {

syncResponse.hadError = false;

if(syncResponse.state != result.state) {

syncResponse.state = result.state;

writeToConsole(result.state);

}

})

.fail(function(requestObject, error, errorThrown) {

if(!syncResponse.hadError) {

syncResponse.hadError = true;

writeToConsole("An error happened! check browser console for more info");

$(selectors.errorContainer).html(requestObject.responseText);

console.log(requestObject);

console.log(error);

console.log(errorThrown);

}

})

.always(function() {

syncResponse.waiting = false;

});

}

})();

recognize = (function () {

var recognizeResponse = {

waiting: false

};

return function(){

if(recognizeResponse.waiting) { return }

recognizeResponse.waiting = true;

$.ajax({

method: "POST",

url: baseURL + "recognize"

})

.done(function(result) {

console.log(result);

writeToConsole(result.state);

})

.fail(function(requestObject, error, errorThrown) {

writeToConsole("An error while recognizing happened! check browser console for more info");

$(selectors.errorContainer).html(requestObject.responseText);

console.log(requestObject);

console.log(error);

console.log(errorThrown);

})

.always(function() {

recognizeResponse.waiting = false;

});

}

})();

function calculateMoveInstructions() {

var moveInstructions = jQuery.extend({}, pressedArrows);

if(moveInstructions.up && moveInstructions.down) {

moveInstructions.up = 0;

moveInstructions.down = 0;

}

if(moveInstructions.left && moveInstructions.right) {

moveInstructions.left = 0;

moveInstructions.right = 0;

}

return moveInstructions;

}

function arrowKeysBinding(callback) {

$(document).keydown(function(e) {

switch(e.which) {

case 13: // enter

callback.enter.press();

break;

case 37: // left

callback.left.press();

break;

case 38: // up

callback.up.press();

break;

case 39: // right

callback.right.press();

break;

case 40: // down

callback.down.press();

break;

default: return; // exit this handler for other keys

}

e.preventDefault(); // prevent the default action (scroll / move caret)

});

$(document).keyup(function(e) {

switch(e.which) {

case 13: // enter

callback.enter.release();

break;

case 37: // left

callback.left.release();

break;

case 38: // up

callback.up.release();

break;

case 39: // right

callback.right.release();

break;

case 40: // down

callback.down.release();

break;

default: return; // exit this handler for other keys

}

e.preventDefault(); // prevent the default action (scroll / move caret)

});

// Mouse click events

$(selectors.arrows.general).mousedown(function(event) {

var clickedArrow = $(this).attr("data-arrow");

switch (event.which) {

case 1: // left mouse button only

callback[clickedArrow].press();

break;

default: return; // exit this handler for other keys

}

});

$(selectors.arrows.general).mouseup(function(event) {

var clickedArrow = $(this).attr("data-arrow");

switch (event.which) {

case 1: // left mouse button only

callback[clickedArrow].release();

break;

default: return; // exit this handler for other keys

}

});

}

Flask Application (Server-side script)

#!/usr/bin/env python

from importlib import import\_module

import json

import os

from flask import Flask, render\_template, Response, request, url\_for, jsonify, g

# from camera.camera\_opencv import Camera

# Raspberry Pi camera module (requires picamera package)

from camera.camera\_pi import Camera

import smbus

import time

import sys

from PIL import Image

import picamera

import io

'''

camera = picamera.PiCamera()

camera.resolution = (300, 300)

camera.framerate = 10

'''

bus = smbus.SMBus(1)

app = Flask(\_\_name\_\_)

frame = 0

@app.route('/')

def index():

"""Video streaming home page."""

return render\_template('index.html')

def gen(camera):

"""Video streaming generator function."""

global frame

while True:

global frame

frame = camera.get\_frame()

yield (b'--frame\r\n'

b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')

@app.route('/video\_feed')

def video\_feed():

"""Video streaming route. Put this in the src attribute of an img tag."""

return Response(gen(Camera()),

mimetype='multipart/x-mixed-replace; boundary=frame')

@app.route('/move', methods=['POST'])

def move\_robot():

state = ''

move = {

'left': int(request.form['left']),

'up': int(request.form['up']),

'right': int(request.form['right']),

'down': int(request.form['down'])

}

if move['up'] and move['down']:

move['up']=0

move['down']=0

if move['left'] and move['right']:

move['left']=0

move['right']=0

# Move the car

if move['up'] and move['right']:

state="Moving: up-right"

bus.write\_byte\_data(0x21, 0x00, 9)

elif move['up'] and move['left']:

state="Moving: up-left"

bus.write\_byte\_data(0x21, 0x00, 7)

elif move['down'] and move['right']:

state="Moving: down-right"

bus.write\_byte\_data(0x21, 0x00, 3)

elif move['down'] and move['left']:

state="Moving: down-left"

bus.write\_byte\_data(0x21, 0x00, 1)

elif move['up']:

state="Moving: up"

bus.write\_byte\_data(0x21, 0x00, 8)

elif move['down']:

state="Moving: down"

bus.write\_byte\_data(0x21, 0x00, 2)

elif move['left']:

state="Moving: left"

bus.write\_byte\_data(0x21, 0x00, 4)

elif move['right']:

state="Moving: right"

bus.write\_byte\_data(0x21, 0x00, 6)

else:

state="Stopped"

bus.write\_byte\_data(0x21, 0x00, 5)

return jsonify({

'state': state

})

@app.route('/recognize', methods=['POST'])

def recognize\_picture():

global first

state = 'Recognizing...'

global frame

image = Image.open(io.BytesIO(frame))

image.save('/home/pi/Pics/2','jpeg')

sys.stdout.write("qq")

sys.stdout.flush()

text = ''

length = sys.stdin.read(4)

sys.stdout.write("ZZzz")

sys.stdout.flush()

sys.stdout.write(str(length))

sys.stdout.flush()

rec = sys.stdin.read(int(length))

state = rec

return jsonify({

'state': state

})

if \_\_name\_\_ == '\_\_main\_\_':

app.debug = False

app.run(host='0.0.0.0', threaded=True, port=5000)

TensorFLow Image Classification Script

from \_\_future\_\_ import absolute\_import

from \_\_future\_\_ import division

from \_\_future\_\_ import print\_function

import threading

import sys

import time

import argparse

import os.path

import re

import sys

import tarfile

import threading

import numpy as np

from app import frame

import io

from six.moves import urllib

import tensorflow as tf

FLAGS = None

from PIL import Image

# pylint: disable=line-too-long

DATA\_URL = 'http://download.tensorflow.org/models/image/imagenet/inception-2015-12-05.tgz'

# pylint: enable=line-too-long

text = ''

class NodeLookup(object):

def \_\_init\_\_(self,

label\_lookup\_path=None,

uid\_lookup\_path=None):

if not label\_lookup\_path:

label\_lookup\_path = os.path.join(

FLAGS.model\_dir, 'imagenet\_2012\_challenge\_label\_map\_proto.pbtxt')

if not uid\_lookup\_path:

uid\_lookup\_path = os.path.join(

FLAGS.model\_dir, 'imagenet\_synset\_to\_human\_label\_map.txt')

self.node\_lookup = self.load(label\_lookup\_path, uid\_lookup\_path)

def load(self, label\_lookup\_path, uid\_lookup\_path):

if not tf.gfile.Exists(uid\_lookup\_path):

tf.logging.fatal('File does not exist %s', uid\_lookup\_path)

if not tf.gfile.Exists(label\_lookup\_path):

tf.logging.fatal('File does not exist %s', label\_lookup\_path)

# Loads mapping from string UID to human-readable string

proto\_as\_ascii\_lines = tf.gfile.GFile(uid\_lookup\_path).readlines()

uid\_to\_human = {}

p = re.compile(r'[n\d]\*[ \S,]\*')

for line in proto\_as\_ascii\_lines:

parsed\_items = p.findall(line)

uid = parsed\_items[0]

human\_string = parsed\_items[2]

uid\_to\_human[uid] = human\_string

# Loads mapping from string UID to integer node ID.

node\_id\_to\_uid = {}

proto\_as\_ascii = tf.gfile.GFile(label\_lookup\_path).readlines()

for line in proto\_as\_ascii:

if line.startswith(' target\_class:'):

target\_class = int(line.split(': ')[1])

if line.startswith(' target\_class\_string:'):

target\_class\_string = line.split(': ')[1]

node\_id\_to\_uid[target\_class] = target\_class\_string[1:-2]

# Loads the final mapping of integer node ID to human-readable string

node\_id\_to\_name = {}

for key, val in node\_id\_to\_uid.items():

if val not in uid\_to\_human:

tf.logging.fatal('Failed to locate: %s', val)

name = uid\_to\_human[val]

node\_id\_to\_name[key] = name

return node\_id\_to\_name

def id\_to\_string(self, node\_id):

if node\_id not in self.node\_lookup:

return ''

return self.node\_lookup[node\_id]

def create\_graph():

"""Creates a graph from saved GraphDef file and returns a saver."""

# Creates graph from saved graph\_def.pb.

with tf.gfile.FastGFile(os.path.join(

FLAGS.model\_dir, 'classify\_image\_graph\_def.pb'), 'rb') as f:

graph\_def = tf.GraphDef()

graph\_def.ParseFromString(f.read())

\_ = tf.import\_graph\_def(graph\_def, name='')

def run\_inference\_on\_image(image):

if not tf.gfile.Exists(image):

tf.logging.fatal('File does not exist %s', image)

image\_data = tf.gfile.FastGFile(image, 'rb').read()

# Creates graph from saved GraphDef.

with tf.Session() as sess:

softmax\_tensor = sess.graph.get\_tensor\_by\_name('softmax:0')

predictions = sess.run(softmax\_tensor,

{'DecodeJpeg/contents:0': image\_data})

predictions = np.squeeze(predictions)

# Creates node ID --> English string lookup.

node\_lookup = NodeLookup()

top\_k = predictions.argsort()[-FLAGS.num\_top\_predictions:][::-1]

global text

for node\_id in top\_k:

human\_string = node\_lookup.id\_to\_string(node\_id)

score = predictions[node\_id]

#print('%s (score = %.5f)' % (human\_string, score))

text = text + ('%s (score = %.5f)' % (human\_string, score)) + "\n"

def maybe\_download\_and\_extract():

"""Download and extract model tar file."""

dest\_directory = "/home/pi/tar"

if not os.path.exists(dest\_directory):

os.makedirs(dest\_directory)

filename = DATA\_URL.split('/')[-1]

filepath = os.path.join(dest\_directory, filename)

if not os.path.exists(filepath):

def \_progress(count, block\_size, total\_size):

sys.stdout.write('\r>> Downloading %s %.1f%%' % (

filename, float(count \* block\_size) / float(total\_size) \* 100.0))

sys.stdout.flush()

filepath, \_ = urllib.request.urlretrieve(DATA\_URL, filepath, \_progress)

print()

statinfo = os.stat(filepath)

print('Successfully downloaded', filename, statinfo.st\_size, 'bytes.')

tarfile.open(filepath, 'r:gz').extractall(dest\_directory)

#def main():

if \_\_name\_\_ == '\_\_main\_\_':

parser = argparse.ArgumentParser()

parser.add\_argument(

'--model\_dir',

type=str,

default='/home/pi/tar',

help="""\

Path to classify\_image\_graph\_def.pb,

imagenet\_synset\_to\_human\_label\_map.txt, and

imagenet\_2012\_challenge\_label\_map\_proto.pbtxt.\

"""

)

parser.add\_argument(

'--image\_file',

type=str,

default='',

help='Absolute path to image file.'

)

parser.add\_argument(

'--num\_top\_predictions',

type=int,

default=5,

help='Display this many predictions.'

)

FLAGS, unparsed = parser.parse\_known\_args()

#tf.app.run(main=main, argv=[sys.argv[0]] + unparsed)

#main()

maybe\_download\_and\_extract()

#image = (FLAGS.image\_file if FLAGS.image\_file else

# os.path.join(FLAGS.model\_dir, 'cropped\_panda.jpg'))

image = os.path.join('/home/pi/Pics', '1.jpg')

sys.stdout.write("Wait Some Minutes \n")

sys.stdout.flush()

create\_graph()

#thread\_1.start()

run\_inference\_on\_image(image)

sys.stdout.write("Ready!!!!!!!! \n")

sys.stdout.flush()

#run\_inference\_on\_image()

#sys.stdout.write("Ready!!!!!!!! \n")

#sys.stdout.flush()

image = os.path.join('/home/pi/Pics', '2')

global text

sys.stdout.write("{{")

sys.stdout.flush()

while True:

text =''

request = sys.stdin.read(2)

if request == "re":

run\_inference\_on\_image(image)

if len(text)%2 != 0:

text = text + ' '

text = str(len(text)) + " " + text

sys.stdout.write(text)

sys.stdout.flush()

text = ''

#sys.stdout.write("an")

#sys.stdout.flush()

x = 1

Pulse Width Modulation Driver

#include "PWM.h"

void PWM\_init\_OC0(void){

    TCCR0 = (1<<WGM00) | (1<<WGM01) | (1<<COM01) | (1<<CS01);//fast PWM

                                                        // Non-inverted

                                                        // Pre-scaler = 8

//About 490 HZ

    TCNT0 = 0;

    OCR0 = 0;

    DDRB |= (1<<PB3);

}

void PWM\_init\_OC2(void){

    TCCR2 = (1 << WGM20) | (1 << WGM21) | (1 << COM21) | (1 << CS21);

                                                        //fast PWM with

                                                        // Non-inverted

                                                        // Pre-scaler = 8

    TCNT2 = 0;

    OCR2 = 0;

    DDRD |= (1 << PD7);

}

void PWM\_init\_OC1A(void){

    TCCR1B = (1 << WGM12) | (1 << CS11); // presacler = 8

    TCCR1A = (1 << WGM10) | (1 << COM1A1) | (1 << COM1B1);

    TCNT1 = 0;

    DDRD |= (1 << PD5);

}

void PWM\_init\_OC1B(void){

    TCCR1B = (1 << WGM12) | (1 << CS11); // presacler = 8

    TCCR1A = (1 << WGM10) | (1 << COM1A1) | (1 << COM1B1);

    TCNT1 = 0;

    DDRD |= (1 << PD4);

}

void PWM\_Duty\_Cycle\_OC1A(unsigned char Duty\_Cycle){

    OCR1A = Duty\_Cycle;

}

void PWM\_Duty\_Cycle\_OC1B(unsigned char Duty\_Cycle){

    OCR1B = Duty\_Cycle;

}

void PWM\_Duty\_Cycle\_OC0(unsigned char Duty\_Cycle){

    OCR0 = Duty\_Cycle;

}

void PWM\_Duty\_Cycle\_OC2(unsigned char Duty\_Cycle){

    OCR2 = Duty\_Cycle;

}

Twin Wires Interface (I2C Protocol)

#include "TWI.h"

#include <avr/io.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*I2c initialization\*/

void TWI\_init(void){

    PORTC |= (1 << PC0);

    PORTC |= (1 << PC1);

#if TWI\_SPEED == 100

    TWBR=32; // to set the speed 100kb/s ,,, Speed 400khz TWBR=2;

#elif TWI\_SPEED == 400

    TWBR=2; // to set the speed 100kb/s ,,, Speed 400khz TWBR=2;

#endif

    TWSR=0; // TWPS=0; //PreScaler 1:1

    TWCR=(1<<TWEN)|(1<<TWEA); // enable the TWI peripheral

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*I2c initialization\*/

void TWI\_init\_slave(unsigned char address){

    TWBR = 32;

    TWSR = 0;

    TWCR = (1 << TWEN); // enable the TWI peripheral

    TWAR = ( (address << 1) & (0xfe)); // address of the slave

    TWCR = (1 << TWINT) | (1 << TWEN) | (1 << TWEA);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*I2c recieve\*/

unsigned char TWI\_recieve(){

    TWCR = (1 << TWINT) | (1 << TWEN) | (1 << TWEA); // enable the TWI peripheral

    while(!(TWCR & (1<<TWINT)) );

    return TWDR;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*I2c listen\*/

void TWI\_listen(){

    while(!(TWCR & (1<<TWINT)) );

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Device Address\*/

void TWI\_address(unsigned char address){

    TWAR = ((address<<1)&(0xfe));

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Sending the Address\*/

void TWI\_send\_DeviceAddress\_Read(unsigned char address){

    TWDR=((address<<1)|(0x01));//Address+ReadBit

    TWCR=(1<<TWEN)|(1<<TWINT)|(1<<TWEA);

    while((!(TWCR & (1<<TWINT)))&& ((((TWSR&0xf8)!=0x40))));

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void TWI\_send\_DeviceAddress\_Write(unsigned char address){

    TWDR=((address<<1)&(0xfe));//Address+WriteBit

    TWCR=(1<<TWEN)|(1<<TWINT);

    while((!(TWCR & (1<<TWINT)))&&((TWSR&0xf8)!=0x18));

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Sending a byte of data\*/

void TWI\_send\_data(unsigned char Data){

    TWDR=Data;

    TWCR=(1<<TWEN)|(1<<TWINT);

    while((!(TWCR & (1<<TWINT)))&&( (((TWSR&0xf8)!=0x28))||(((TWSR&0xf8)!=0xB8))));

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Sending a stop bit\*/

void TWI\_stop(void){

    TWCR = (1<<TWINT)|(1<<TWEN)|(1<<TWSTO);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Receive with ACK\*/

unsigned char TWI\_recieve\_data\_ACK(void){

    /\*while the TWIF is zero (job not finished) & status register not telling u that you received anything ....wait\*/

    while(((!(TWCR & (1<<TWINT))))&&((((TWSR&0xf8)!=0x80))||((TWSR&0xf8)!=0x50)));

    TWCR=(1<<TWEN)|(1<<TWINT)|(1<<TWEA);

    return TWDR;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*Receive with NACK\*/

unsigned char TWI\_recieve\_data\_NACK(void)

{

    while(((!(TWCR & (1<<TWINT))))&&((((TWSR&0xf8)!=0x88))||((TWSR&0xf8)!=0x58)));

    TWCR=(1<<TWEN)|(1<<TWINT);

    return TWDR;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Main Motor Control AVR Code

#include <avr/io.h>

#include <util/delay.h>

#include "PWM.h"

#include "TWI.h"

#define SPEED 100

void MotorBR\_forward(unsigned char speed);

void MotorBR\_backward(unsigned char speed);

void MotorBL\_forward(unsigned char speed);

void MotorBL\_backward(unsigned char speed);

void MotorFR\_forward(unsigned char speed);

void MotorFR\_backward(unsigned char speed);

void MotorFL\_forward(unsigned char speed);

void MotorFL\_backward(unsigned char speed);

void Move\_right(unsigned char speed);

void Move\_left(unsigned char speed);

void Move\_backward(unsigned char speed);

void Move\_forward(unsigned char speed);

void Move\_up\_right(unsigned char speed);

void Move\_up\_left(unsigned char speed);

void Move\_down\_right(unsigned char speed);

void Move\_down\_left(unsigned char speed);

void Stop();

int main(){

    unsigned char data\_TWI = 0;

    DDRD = 0xff;

    DDRB = 0xff;

    while(1){

        TWI\_init\_slave(33);

        TWI\_listen();

        data\_TWI = TWI\_recieve\_data\_ACK();

        if(data\_TWI == 8){

            Move\_forward(SPEED);

        }

        else if(data\_TWI == 6) {

            Move\_right(SPEED);

        }

        else if(data\_TWI == 5) {

            Stop();

        }

        else if(data\_TWI == 4) {

            Move\_left(SPEED);

        }

        else if(data\_TWI == 2) {

            Move\_backward(SPEED);

        }

        else if(data\_TWI == 9) {

            Move\_up\_right(SPEED);

        }

        else if(data\_TWI == 7) {

            Move\_up\_left(SPEED);

        }

        else if(data\_TWI == 3) {

            Move\_down\_right(SPEED);

        }

        else if(data\_TWI == 1) {

            Move\_down\_left(SPEED);

        }

        else{

            Stop();

        }

    }

    return 0;

}

void MotorBR\_backward(unsigned char speed){

    DDRD |= (1 << PD3);

    PORTD &= ~(1 << PD3);

    PWM\_init\_OC0();

    PWM\_Duty\_Cycle\_OC0(speed);

}

void MotorBR\_forward(unsigned char speed){

    DDRD |= (1 << PD3);

    PORTD |= (1 << PD3);

    PWM\_init\_OC0();

    PWM\_Duty\_Cycle\_OC0(speed);

}

void MotorBL\_forward(unsigned char speed){

    DDRD |= (1 << PD2);

    PORTD &= ~(1 << PD2);

    PWM\_init\_OC1A();

    PWM\_Duty\_Cycle\_OC1A(speed);

}

void MotorBL\_backward(unsigned char speed){

    DDRD |= (1 << PD2);

    PORTD |= (1 << PD2);

    PWM\_init\_OC1A();

    PWM\_Duty\_Cycle\_OC1A(speed);

}

void MotorFR\_backward(unsigned char speed){

    DDRD |= (1 << PD1);

    PORTD &= ~(1 << PD1);

    PWM\_init\_OC1B();

    PWM\_Duty\_Cycle\_OC1B(speed);

}

void MotorFR\_forward(unsigned char speed){

    DDRD |= (1 << PD1);

    PORTD |= (1 << PD1);

    PWM\_init\_OC1B();

    PWM\_Duty\_Cycle\_OC1B(speed);

}

void MotorFL\_forward(unsigned char speed){

    DDRD |= (1 << PD0);

    PORTD &= ~(1 << PD0);

    PWM\_init\_OC2();

    PWM\_Duty\_Cycle\_OC2(speed);

}

void MotorFL\_backward(unsigned char speed){

    DDRD |= (1 << PD0);

    PORTD |= (1 << PD0);

    PWM\_init\_OC2();

    PWM\_Duty\_Cycle\_OC2(speed);

}

void Move\_forward(unsigned char speed){

    MotorBL\_forward(speed);

    MotorBR\_forward(speed);

    MotorFR\_forward(speed);

    MotorFL\_forward(speed);

}

void Move\_backward(unsigned char speed){

    MotorBL\_backward(speed);

    MotorBR\_backward(speed);

    MotorFR\_backward(speed);

    MotorFL\_backward(speed);

}

void Move\_right(unsigned char speed){

    MotorBL\_forward(speed);

    MotorBR\_backward(speed);

    MotorFR\_backward(speed);

    MotorFL\_forward(speed);

}

void Move\_left(unsigned char speed){

    MotorBL\_backward(speed);

    MotorBR\_forward(speed);

    MotorFR\_forward(speed);

    MotorFL\_backward(speed);

}

void Stop(){

    TCCR2 = 0;

    TCCR0 = 0;

    TCCR1B = 0;

    TCCR1A = 0;

    PORTD &= ~( (1 << PD4) | (1 << PD5) | (1 << PD7) );

    PORTB &= ~(1 << PB3);

}

void Move\_up\_right(unsigned char speed){

    MotorBL\_forward(speed + 30);

    MotorBR\_forward(speed - 30);

    MotorFR\_forward(speed- 30);

    MotorFL\_forward(speed + 30);

}

void Move\_up\_left(unsigned char speed){

    MotorBL\_forward(speed - 30);

    MotorBR\_forward(speed + 30);

    MotorFR\_forward(speed + 30);

    MotorFL\_forward(speed - 30);

}

void Move\_down\_right(unsigned char speed){

    MotorBL\_backward(speed + 30);

    MotorBR\_backward(speed - 30);

    MotorFR\_backward(speed- 30);

    MotorFL\_backward(speed + 30);

}

void Move\_down\_left(unsigned char speed){

    MotorBL\_backward(speed - 30);

    MotorBR\_backward(speed + 30);

    MotorFR\_backward(speed + 30);

    MotorFL\_backward(speed - 30);

}