**Se adicionan las librerías en la carpeta raíz de arduino y luego se importan en el arduino para poderlas utilizar**

**#include <FuzzyRule.h>**

**#include <FuzzyComposition.h>**

**#include <Fuzzy.h>**

**#include <FuzzyRuleConsequent.h>**

**#include <FuzzyOutput.h>**

**#include <FuzzyInput.h>**

**#include <FuzzyIO.h>**

**#include <FuzzySet.h>**

**#include <FuzzyRuleAntecedent.h>**

**#include "Ultrasonic.h"**

**Asignamos los pines del Arduino donde estarán conectados a los pines de entrada de nuestro puente-H, de esta manera podremos controlar ambos motoreductores.**

**const int enPin = 5; // PWM se conecta al pin 1 del puente-H**

**const int in1Pin = 2; // Entrada 2 del puente-H**

**const int in2Pin = 4; // Entrada 7 del puente-H**

**Y el sensor ultrasónico.**

**Ultrasonic ultrasonico(7,8);**

**Fuzzy\* fuzzy = new Fuzzy();        //Iniciando objeto**

**Creamos los conjuntos difusos.**

**FuzzySet\* demcerca = new FuzzySet(-20, -10, 5, 9);      //Conjunto difuso Distancia demasiado cerca**

**FuzzySet\* muycerca = new FuzzySet(14.5, 15, 15.5, 16);    //Conjunto difuso Distancia muy cerca**

**FuzzySet\* cerca = new FuzzySet(15.5, 20, 40, 50);     //Conjunto difuso Distancia cerca**

**FuzzySet\* media = new FuzzySet(45, 50, 70, 80);     //Conjunto difuso Distancia promedio**

**FuzzySet\* lejos = new FuzzySet(75, 80, 103, 103);   //Conjunto difuso Distancia lejos**

**FuzzySet\* Alto = new FuzzySet(0, 0, 0, 0);                // Velocidad para detener**

**FuzzySet\* Suave = new FuzzySet(0, 200, 500, 700);         // Velocidad suave**

**FuzzySet\* Media = new FuzzySet(600, 1200, 1800, 2000);    // Velocidad medio**

**FuzzySet\* Rapida = new FuzzySet(1900, 2200, 3000, 3500);  // Velocidad rápido**

**int dist;**

**Después de haber creado los conjuntos difusos, dentro del void setup() crearemos las entradas y  las salidas de nuestros conjuntos DISTANCIA y VELOCIDAD.**

**void setup(){**

**Serial.begin(9600);**

**pinMode(in1Pin, OUTPUT); //Configura in1Pin como salida**

**pinMode(in2Pin, OUTPUT); //Configura in2Pin como salida**

**//Crear composición de entrada difusa distancia**

**FuzzyInput\* distance = new FuzzyInput(1);**

**distance->addFuzzySet(demcerca); //Agrega fuzzyset demasiado cerca a distancia**

**distance->addFuzzySet(muycerca); //Agrega fuzzyset muy cerca a distancia**

**distance->addFuzzySet(cerca);    //Agrega fuzzyset cerca a distancia**

**distance->addFuzzySet(media);    //Agrega fuzzyset media a distancia**

**distance->addFuzzySet(lejos);    //Agrega fuzzyset lejos a distancia**

**fuzzy->addFuzzyInput(distance);  //Agrega entrada difusa al objeto difuso**

**//Crear composición de entrada difusa vel**

**FuzzyInput\* vel = new FuzzyInput(2);**

**vel->addFuzzySet(Alto); // Agregar fuzzyset paro**

**vel->addFuzzySet(Suave);// Agregar fuzzyset lento**

**vel->addFuzzySet(Media);// Agregar fuzzyset medio**

**vel->addFuzzySet(Rapida);//Agregar fuzzyset rapido**

**fuzzy->addFuzzyInput(vel);// Agrega entrada difusa al objeto difuso**

**//Crear composición de salida difusa velocidad**

**FuzzyOutput\* salPWM = new FuzzyOutput(1);  //Crea salida difusa velocidad**

**FuzzySet\* rev = new FuzzySet(-4, -4, -3, 0);**

**salPWM->addFuzzySet(rev);// Agregar fuzzyset paro a velocidad**

**FuzzySet\* paro = new FuzzySet(-.5, 1, 1.5, 4);**

**salPWM->addFuzzySet(paro);// Agregar fuzzyset paro a velocidad**

**FuzzySet\* lenta = new FuzzySet(3.9, 5, 5.5, 6);**

**salPWM->addFuzzySet(lenta);// Agregar fuzzyset lento a velocidad**

**FuzzySet\* medio = new FuzzySet(5.5, 6, 7, 8);**

**salPWM->addFuzzySet(medio);// Agregar fuzzyset medio a velocidad**

**FuzzySet\* rapido = new FuzzySet(7.5, 9, 9, 9);**

**salPWM->addFuzzySet(rapido);//Agregar fuzzyset rapido a velocidad**

**fuzzy->addFuzzyOutput(salPWM);//  Agrega entrada difusa al objeto difuso**

**Una vez realizada la fuzzificación evaluamos las reglas difusas.**

**// Arma las reglas difusas. (Distancia)**

**// FuzzyRule 1 "IF distancia =  muy cerca THEN salPWM = Paro"**

**FuzzyRuleAntecedent\* ifdistancemuycerca = new FuzzyRuleAntecedent(); // Inicializando antecedente de la expresión**

**ifdistancemuycerca->joinSingle(muycerca); // Agregando FuzzySet correspondiente al antecedente**

**FuzzyRuleConsequent\* thensalPWMparo = new FuzzyRuleConsequent(); // Inicializando consecuencia de la expresión**

**thensalPWMparo->addOutput(paro); // Agregando FuzzySet correspondiente a la consecuencia**

**// Instantiating a FuzzyRule object**

**FuzzyRule\* fuzzyRule01 = new FuzzyRule(1, ifdistancemuycerca, thensalPWMparo);**

**fuzzy->addFuzzyRule(fuzzyRule01); // Agrega regla difusa al objeto difuso.**

**// FuzzyRule "IF distance = cerca THEN salPWM = lenta"**

**FuzzyRuleAntecedent\* ifdistancecerca = new FuzzyRuleAntecedent(); // Inicializando antecedente de la expresión**

**ifdistancecerca->joinSingle(cerca); // Agregando FuzzySet correspondiente al antecedente**

**FuzzyRuleConsequent\* thensalPWMlenta = new FuzzyRuleConsequent(); // Inicializando consecuencia de la expresión**

**thensalPWMlenta->addOutput(lenta);// Agregando FuzzySet correspondiente a la consecuencia**

**// Instantiating a FuzzyRule object**

**FuzzyRule\* fuzzyRule02 = new FuzzyRule(2, ifdistancecerca, thensalPWMlenta);**

**fuzzy->addFuzzyRule(fuzzyRule02); // Agrega regla difusa al objeto difuso.**

**de la misma manera creamos las reglas difusas para la distancia media, distancia cerca y demasiado cerca.**

**Programación del Sensor Ultrasónico SFR05**

**void loop(){**

**float dist = ultrasonico.Ranging(CM);**

**int rpm=getRPM();**

**Serial.println(rpm, DEC);**

**fuzzy->setInput(1, dist); //Agrega variable de entrada dist al objeto difuso**

**fuzzy->setInput(2, rpm); //Agrega variable de entrada dist al objeto difuso**

**Asignamos las variables flotantes.**

**float a;**

**float b;**

**float c;**

**float d;**

**a= muycerca->getPertinence();**

**b= cerca->getPertinence();**

**c= media->getPertinence();**

**d= lejos->getPertinence();**

**fuzzy->fuzzify(); // Fuzificación**