Circuito centro

183 526 504 176 420 687 613 361 188 586 616 262

Circulo superior izquierda

168 586 634 302 297 671 628 583 142 479 575 287

Circulo superior derecha

151 420 466 190 294 666 650 567 163 595 711 454

Circulo inferior izquierda

270 683 590 183 674 720 601 280 185 545 526 218

Circulo inferior derecha

186 476 425 149 657 712 623 275 284 677 663 263

triángulo centro

269 265 86 75 388 588 416 211 280 322 129 110 triángulo superior izquierda

191 589 565 117 108 487 297 80 65 228 120 110

triángulo superior derecha

81 75 259 420 217 417 527 617 65 76 293 614 triángulo inferior izquierda

81 88 240 75 107 350 456 82 85 529 664 289

cuadrado centro

275 640 640 432 373 764 710 464 275 648 678 517 cuadrado superior izquierda

261 711 716 499 339 764 707 518 243 617 636 542

cuadrado superior derecha

189 506 523 517 268 749 777 779 198 732 862 856

cuadrado inferior izquierda

567 800 742 280 812 810 680 262 577 640 613 326

cuadrado inferior derecha

395 609 592 270 565 795 701 285 427 728 709 378

Imagen que contiene Texto

Descripción generada automáticamente

Warning: NEWFF used in an obsolete way.

> In nnerr.obs\_use (line 17)

In newff>create\_network (line 126)

In newff (line 101)

In RedNeuronalOr (line 28)

See help for NEWFF to update calls to the new argument list.

a =

1.0789 1.2216 1.2995 1.1713 0.8358 -0.1432 0.2631 0.1327 0.2540 -0.1220 -0.2373 -0.2844 0.1253 -1.1510

0.0676 -0.2217 -0.2581 0.1250 0.4895 1.1358 0.6769 0.9750 0.6761 1.1019 0.5168 -0.0428 0.4046 0.8622

0.1556 -0.0743 0.0357 0.3944 0.1082 -0.0669 -0.3808 -0.1581 0.1776 0.2288 1.1575 0.5484 1.0664 0.6951

pesos1 =

-0.0012 -0.0003 0.0011 -0.0008 0.0000 0.0034 0.0008 0.0005 -0.0036 0.0001 -0.0016 -0.0024

0.0011 0.0013 -0.0007 0.0016 -0.0023 0.0004 -0.0021 -0.0023 -0.0006 -0.0001 0.0019 0.0015

-0.0001 -0.0001 0.0017 0.0024 -0.0014 0.0039 0.0016 -0.0015 -0.0012 0.0002 -0.0015 0.0020

0.0010 0.0014 0.0015 0.0018 0.0014 0.0003 -0.0025 0.0015 -0.0024 -0.0017 -0.0016 0.0018

0.0011 -0.0015 -0.0031 -0.0027 -0.0007 -0.0024 -0.0009 -0.0014 -0.0035 -0.0014 -0.0035 0.0008

0.0007 -0.0000 0.0010 0.0009 0.0012 0.0020 0.0011 0.0012 -0.0005 0.0012 -0.0021 0.0024

-0.0030 0.0013 0.0020 -0.0027 0.0014 -0.0021 0.0011 -0.0003 -0.0014 0.0010 -0.0004 -0.0004

-0.0014 -0.0006 0.0001 -0.0036 0.0006 -0.0002 0.0031 0.0001 -0.0036 -0.0017 -0.0000 -0.0020

-0.0016 -0.0009 0.0029 -0.0027 0.0003 0.0018 0.0029 0.0006 -0.0004 -0.0003 0.0002 -0.0000

-0.0017 0.0021 0.0017 0.0018 0.0016 0.0008 0.0024 -0.0020 -0.0018 0.0008 0.0030 -0.0011

-0.0018 0.0025 -0.0010 0.0002 0.0026 0.0035 0.0017 0.0009 0.0004 0.0019 0.0024 -0.0013

0.0018 -0.0012 -0.0022 -0.0026 -0.0012 0.0010 0.0027 -0.0004 0.0012 0.0001 0.0020 0.0013

bias1 =

2.7489

-2.7504

0.0377

-1.5307

0.9476

-4.4908

0.2960

1.5686

-2.8369

-2.8393

-3.7961

-0.2157

peso2 =

-0.3974 -0.0659 0.6844 -0.3042 -0.6458 0.7970 0.0800 -0.4243 0.5279 -0.6438 0.0438 -0.5821

-0.4089 0.2964 0.1181 -0.1079 0.3256 -0.7637 0.4138 -0.1710 0.6364 -0.2807 -0.3283 0.8103

-0.3341 -0.9495 0.7082 -0.8915 -0.3383 0.9768 0.9990 -0.0703 -0.7995 -0.8866 -0.6487 0.3508

bias2 =

-0.0631

0.8242

-0.7920

>>

///////////////////////////////////////////////////////////////////////////////////////////

Warning: NEWFF used in an obsolete way.

> In nnerr.obs\_use (line 17)

In newff>create\_network (line 126)

In newff (line 101)

In RedNeuronalOr (line 28)

See help for NEWFF to update calls to the new argument list.

a =

0.9907 1.3549 0.8872 1.5332 0.9836 0.0371 -0.2198 -0.0355 0.0036 -0.3813 0.2436 -0.1014 0.1767 -0.0779

0.3606 0.1905 0.1326 0.1696 0.4169 0.9292 0.7234 0.8276 0.8089 1.0788 0.6535 -0.0699 0.2412 -0.0119

-0.0824 -0.1434 0.1717 0.1171 0.0584 0.0424 -0.0523 0.0456 -0.0695 0.2842 1.2071 0.8777 1.1062 0.6956

pesos1 =

-0.0010 -0.0029 -0.0010 0.0019 0.0023 -0.0025 -0.0019 -0.0001 -0.0008 -0.0019 0.0011 0.0002

0.0026 -0.0004 -0.0016 -0.0001 -0.0004 0.0019 -0.0020 0.0024 0.0028 -0.0011 0.0006 -0.0017

-0.0007 -0.0028 -0.0006 0.0012 0.0009 0.0025 0.0024 -0.0001 0.0003 0.0018 -0.0014 -0.0021

-0.0018 0.0008 0.0014 0.0015 0.0004 0.0028 -0.0013 0.0013 -0.0000 0.0005 -0.0012 0.0020

0.0005 -0.0001 -0.0013 0.0002 -0.0001 -0.0033 0.0017 -0.0031 -0.0005 -0.0003 0.0000 0.0019

-0.0009 -0.0001 -0.0015 0.0006 0.0005 0.0016 -0.0027 0.0025 -0.0022 0.0018 0.0006 -0.0027

0.0030 -0.0003 0.0013 0.0028 0.0007 0.0015 -0.0020 0.0021 0.0031 0.0009 -0.0008 -0.0000

0.0031 0.0004 0.0021 0.0012 0.0002 -0.0005 0.0031 0.0005 0.0033 0.0005 -0.0013 0.0016

0.0024 -0.0006 -0.0004 0.0035 0.0001 0.0006 0.0034 0.0004 0.0001 -0.0013 -0.0020 -0.0008

0.0014 -0.0015 -0.0017 -0.0000 0.0017 0.0002 0.0033 -0.0005 0.0029 0.0006 -0.0001 0.0016

0.0022 -0.0009 -0.0009 -0.0026 0.0001 0.0023 0.0036 0.0016 -0.0000 -0.0011 0.0007 0.0019

0.0022 -0.0015 -0.0015 -0.0021 -0.0016 0.0016 -0.0023 -0.0014 -0.0003 -0.0016 -0.0009 0.0011

bias1 =

4.6375

-0.4083

-0.5184

-4.5037

-1.0816

0.2354

-1.0763

-4.2194

0.9089

-1.4390

1.2116

6.3967

peso2 =

-0.9955 0.3607 -0.5135 0.6994 -0.2837 -0.4994 0.0625 -0.7167 -0.0430 0.8107 0.1318 0.3714

-0.2405 -0.2424 0.1428 -0.4331 0.9739 0.6227 0.6012 -0.1242 0.1748 0.2804 0.8632 -0.0676

0.8088 0.2639 0.9635 0.3649 -0.8320 -0.8312 0.4776 -0.2992 -0.7084 -0.6741 0.5662 -0.4794

bias2 =

0.1385

-0.5025

-0.3614

>>

/////////////////////////////////////////////////////////////////////////////////

Warning: NEWFF used in an obsolete way.

> In nnerr.obs\_use (line 17)

In newff>create\_network (line 126)

In newff (line 101)

In RedNeuronalOr (line 28)

See help for NEWFF to update calls to the new argument list.

a =

0.9832 0.9580 0.9191 1.0305 1.0692 0.0000 -0.0002 -0.0098 0.0167 -0.0692 0.1080 0.0085 0.2638 0.0580

0.0572 0.1036 0.1856 0.1484 0.0143 0.9348 1.0053 0.9000 0.9820 0.9339 -0.0586 -0.1001 -0.0028 0.0668

0.2253 0.1721 0.1068 0.1526 0.2725 0.0439 -0.1426 0.0347 -0.0416 0.0759 0.9622 1.0497 1.0417 0.8173

pesos1 =

-0.0027 0.0012 -0.0006 0.0019 0.0024 0.0009 0.0005 -0.0016 0.0032 -0.0005 0.0007 -0.0004

-0.0011 0.0010 0.0007 0.0032 -0.0015 0.0017 0.0045 0.0000 0.0024 -0.0009 0.0004 -0.0008

-0.0007 -0.0017 -0.0034 0.0023 0.0001 0.0021 0.0024 -0.0014 0.0023 -0.0002 0.0006 0.0004

0.0021 0.0023 0.0012 -0.0015 -0.0031 -0.0023 0.0010 -0.0004 0.0020 -0.0021 -0.0001 -0.0000

-0.0034 0.0014 0.0009 -0.0021 0.0014 -0.0019 -0.0029 0.0025 0.0009 -0.0005 0.0014 -0.0005

0.0016 -0.0029 0.0006 0.0013 0.0006 -0.0042 -0.0008 -0.0014 0.0014 -0.0018 -0.0019 -0.0024

-0.0017 -0.0031 -0.0029 0.0028 -0.0003 0.0015 0.0021 -0.0015 -0.0006 -0.0017 -0.0003 -0.0001

-0.0025 0.0024 -0.0005 -0.0019 0.0003 -0.0016 0.0001 0.0009 0.0014 0.0003 0.0004 -0.0001

-0.0007 0.0008 -0.0021 -0.0033 0.0015 -0.0031 0.0012 0.0012 -0.0012 -0.0006 -0.0019 -0.0009

0.0020 -0.0001 0.0015 0.0030 0.0008 -0.0021 0.0037 0.0016 0.0017 0.0003 -0.0012 -0.0019

-0.0015 -0.0003 0.0019 0.0023 0.0002 0.0014 0.0018 -0.0005 0.0039 0.0008 -0.0012 0.0008

-0.0023 0.0003 -0.0014 0.0031 -0.0017 -0.0002 -0.0025 -0.0005 0.0001 -0.0028 0.0014 -0.0014

bias1 =

-0.8071

0.6439

2.0218

-1.2393

1.0027

1.0989

-2.2818

-0.0355

-0.9654

0.7761

-3.6886

1.1725

peso2 =

-0.5930 -0.8929 -0.6694 -0.3404 -0.8600 -0.4271 0.0242 0.4642 -0.5210 0.6848 0.5878 0.3752

0.6275 -0.2499 0.8244 -0.5915 0.9001 0.3743 0.4427 0.4997 0.0417 0.3258 -0.0618 0.9737

-0.2132 0.5500 -0.3616 0.5344 -0.6835 -0.7177 0.8577 -0.1854 -0.5618 0.6325 -0.3810 0.5399

bias2 =

0.6591

0.4122

0.1907

>>

Arduino programa anterior

#include "math.h"

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

//Crear el objeto lcd dirección 0x3F y 16 columnas x 2 filas

LiquidCrystal\_I2C lcd(0x3F,16,2); //

//ENTRADAS SENSORES

int s0 = A0;

int s1 = A1;

int s2 = A2;

int s3 = A3;

int s4 = A4;

int s5 = A5;

int s6 = A6;

int s7 = A7;

int s8 = A8;

int s9 = A9;

int s10 = A10;

int s11 = A11;

int p1 = 0;

int p2 = 0;

int p3 = 0;

int p4 = 0;

int p5 = 0;

int p6 = 0;

int p7 = 0;

int p8 = 0;

int p9 = 0;

int p10 = 0;

int p11 = 0;

int p12 = 0;

float n1 , n2 , n3, n4, n5, n6, n7, n8 , n9, n10 , n11, n12,n13,n14,n15; // variables de las salidas de las neuronas

float a1 , a2 , a3, a4, a5, a6, a7, a8 , a9, a10 , a11, a12 ; // las salidas de las funciones de activacion

float e=2.7182818 ; // Definimos e para trabjar con la tanjente hiperbolica

void setup() {

pinMode (s0 , INPUT);

pinMode (s1 , INPUT);

pinMode (s2 , INPUT);

pinMode (s3 , INPUT);

pinMode (s4 , INPUT);

pinMode (s5 , INPUT);

pinMode (s6 , INPUT);

pinMode (s7 , INPUT);

pinMode (s8 , INPUT);

pinMode (s9 , INPUT);

pinMode (s10 , INPUT);

pinMode (s11 , INPUT);

Serial.begin(9600);

lcd.print("Hola Mundo");

}

void loop()

{

p1 = analogRead(s0);

p2 = analogRead(s1);

p3 = analogRead(s2);

p4 = analogRead(s3);

p5 = analogRead(s4);

p6 = analogRead(s5);

p7 = analogRead(s6);

p8 = analogRead(s7);

p9 = analogRead(s8);

p10 = analogRead(s9);

p11 = analogRead(s10);

p12 = analogRead(s11);

Serial.print(p1);

Serial.print(" ");

Serial.print(p2);

Serial.print(" ");

Serial.print(p3);

Serial.print(" ");

Serial.print(p4);

Serial.print(" ");

Serial.print(p5);

Serial.print(" ");

Serial.print(p6);

Serial.print(" ");

Serial.print(p7);

Serial.print(" ");

Serial.print(p8);

Serial.print(" ");

Serial.print(p9);

Serial.print(" ");

Serial.print(p10);

Serial.print(" ");

Serial.print(p11);

Serial.print(" ");

Serial.print(p12);

Serial.println(" ");

// definimos la sumatoria de nuestra neurona

n1= p1 \* (0.0003) + p2 \* (0.0005) + p3 \* (0.0018) + p4 \* (0.0008) + p5 \* (-0.0008) + p6\* (-0.0014) + p7 \* (-0.0023) + p8 \* (0.0020) + p9 \* (0.0033) + p10 \* (-0.0006) + p11 \* (0.0001) + p12 \* (-0.0003) + (-4.3898) ;

n2= p1 \* (0.0025 ) + p2 \* (0.0013) + p3 \* (0.0032 ) + p4 \* (-0.0033) + p5 \* (-0.0014) + p6\* (-0.0008) + p7 \* (-0.0001) + p8 \* (-0.0039) + p9 \* (0.0022) + p10 \* (0.0003) + p11 \* (0.0044) + p12 \* (-0.0038) + (-1.6034) ;

n3= p1 \* (-0.0024) + p2 \* (0.0025) + p3 \* (0.0035) + p4 \* (0.0036) + p5 \* (0.0006) + p6\* (-0.0008) + p7 \* (-0.0007) + p8 \* (-0.0009 ) + p9 \* (0.0013) + p10 \* (0.0007) + p11 \* (0.0007) + p12 \* (-0.0006) + (-4.2599) ;

n4= p1 \* (-0.0015) + p2 \* (0.0015) + p3 \* (-0.0014) + p4 \* (0.0064) + p5 \* (-0.0020) + p6\* (0.0011) + p7 \* (-0.0045 ) + p8 \* (-0.0030) + p9 \* (0.0045) + p10 \* (0.0038) + p11 \* (0.0021) + p12 \* (-0.0005) + (-0.0802) ;

n5= p1 \* (-0.0001) + p2 \* (0.0019) + p3 \* (0.0012) + p4 \* (0.0006) + p5 \* (-0.0004) + p6\* (-0.0028) + p7 \* (0.0029) + p8 \* (-0.0029) + p9 \* (-0.0023) + p10 \* (0.0046) + p11 \* (0.0016) + p12 \* (-0.0016) + (-0.4693) ;

n6= p1 \* (-0.0010) + p2 \* (-0.0005) + p3 \* (-0.0012) + p4 \* (0.0017) + p5 \* (-0.0003) + p6\* (-0.0029) + p7 \* (-0.0010) + p8 \* (-0.0014) + p9 \* (-0.0006) + p10 \* (0.0011) + p11 \* (-0.0011) + p12 \* (0.0007) + (-2.2345) ;

n7= p1 \* (0.0015) + p2 \* (-0.0049) + p3 \* (0.0038) + p4 \* (-0.0033) + p5 \* (0.0017) + p6\* (0.0027) + p7 \* (0.0014) + p8 \* (-0.0010) + p9 \* (-0.0050 ) + p10 \* (-0.0009) + p11 \* (0.0049) + p12 \* (-0.0006) + (-1.1653) ;

n8= p1 \* (-0.0075) + p2 \* (-0.0067) + p3 \* (0.0027) + p4 \* (-0.0068) + p5 \* (-0.0025 ) + p6\* (0.0021) + p7 \* (0.0034) + p8 \* (0.0025) + p9 \* (0.0072) + p10 \* (-0.0055) + p11 \* (-0.0005) + p12 \* (-0.0044) + (2.5014) ;

n9= p1 \* (0.0045) + p2 \* (0.0016) + p3 \* (0.0000) + p4 \* (-0.0026) + p5 \* (0.0028) + p6\* (-0.0016) + p7 \* (-0.0009) + p8 \* (-0.0007) + p9 \* (-0.0006) + p10 \* (-0.0025) + p11 \* (-0.0009) + p12 \* (-0.0015) + (-2.1101) ;

n10= p1 \* (0.0005) + p2 \* (0.0016) + p3 \* (0.0036) + p4 \* (0.0018) + p5 \* (0.0029) + p6\* (-0.0002) + p7 \* (0.0036) + p8 \* (0.0004) + p9 \* (0.0003) + p10 \* (0.0037) + p11 \* (0.0023) + p12 \* (0.0019) + (-1.4202) ;

n11= p1 \* (0.0013) + p2 \* (0.0001) + p3 \* (0.0012) + p4 \* (0.0002) + p5 \* (-0.0003) + p6\* (0.0002) + p7 \* (0.0006) + p8 \* (-0.0024) + p9 \* (0.0016) + p10 \* (-0.0007) + p11 \* (-0.0010) + p12 \* (0.0010) + (-3.5385) ;

n12= p1 \* (-0.0025) + p2 \* (-0.0027) + p3 \* (-0.0018 ) + p4 \* (-0.0050) + p5 \* (-0.0021) + p6\* (-0.0029) + p7 \* (-0.0002) + p8 \* (0.0035) + p9 \* (-0.0057) + p10 \* (0.0002) + p11 \* (-0.0015) + p12 \* (0.0029) + (0.5888) ;

a1 = tansig ( n1 ) ;

a2 = tansig ( n2 );

a3 = tansig ( n3 );

a4 = tansig ( n4);

a5 = tansig ( n5);

a6 = tansig ( n6);

a7 = tansig ( n7);

a8 = tansig ( n8);

a9 = tansig ( n9);

a10 = tansig ( n10);

a11 = tansig ( n11);

a12 = tansig ( n12);

n13 = a1 \*(-0.1396) + a2 \*(0.5684) + a3 \*(-0.2203) + a4 \*(-0.8993) + a5 \*(-0.9686) + a6 \*(0.3379) + a7 \*(0.1434) + a8 \*(0.1993) + a9 \*(-0.6952)+ a10 \*(0.6646)+ a11 \*(0.7276) + a12 \*(-0.7840) + (0.1189) ;

n14 = a1 \*(0.3874) + a2 \*(0.4109) + a3 \*(0.1817) + a4 \*(-0.5427) + a5 \*(0.7274) + a6 \*(0.0004) + a7 \*(-0.7556) + a8 \*(-0.8880) + a9 \*(-0.9608)+ a10 \*(0.2348)+ a11 \*(-0.8047) + a12 \*(0.0340) + (-0.9908) ;

n15 = a1 \*(0.8903) + a2 \*(-0.7815) + a3 \*(-0.0814) + a4 \*(0.6685) + a5 \*(-0.8437) + a6 \*(-0.5642) + a7 \*(0.3424) + a8 \*(-0.8870) + a9 \*(-0.1298 )+ a10 \*(0.0404)+ a11 \*(0.8159) + a12 \*(-0.7139) + (0.5335) ;

n13=round(n13); //redondeamos para que no salga - negativos

n14=round(n14); //redondeamos para que no salga – negativos

n15=round(n15); //redondeamos para que no salga - negativos

/\*

Serial.println(n13);

Serial.println(n14);

Serial.println(n15);

if(n13==1)

{

Serial.println("cuadrado");

lcd.print("cuadrado");

}

if(n14==1)

{

Serial.println("circulo");

lcd.print("circulo");

}

if(n15==1)

{

Serial.println("triangulo");

lcd.print("trinagulo");

}

\*/

delay(1000);

}

// definimos la funcion tangente hiperbolica

// funcion tansig

float tansig ( float x ) {

float a ;

a = ( pow ( e , x ) -pow ( e , -x ) ) / ( pow ( e , x ) + pow ( e , -x ) ) ;

return a ; // regreseamos el valor de a

}