

UNIVERSIDAD ECCI

SISTEMAS DE EMBEBIDOS

Josimar Hernandez

Sergio Fonseca

Juan Aldana

Jhon Vanegas

ARAÑA CUADRUPEDA LINKI

El presente proyecto consiste en el diseño, fabricación y control del robot cuadrúpedo imprimible llamado LINKI. Ha sido realizado en el departamento de Ingeniería mecatrónica de la Universidad colombiana de carreras industriales (ECCI) como Trabajo de fin de semestre.

OBJETIVOS

1. Estudio y aplicaciones de los distintos robots 'n-podos' (robots andadores de 'n' patas) para la elección del modelo físico óptimo sobre el que se ha basado el desarrollo de todo el proyecto.
2. Diseño del robot cuadrúpedo LINKI.
3. Impresión de las piezas diseñadas en plástico gracias a impresoras 3D.
4. Montaje de la estructura física del robot e instalación de la electrónica necesaria para su control.
5. Control remoto por medio de un tarjeta núcleo F446ZE conectado al PC.
6. Implementación y evaluación del diseño Programación de movimientos básicos

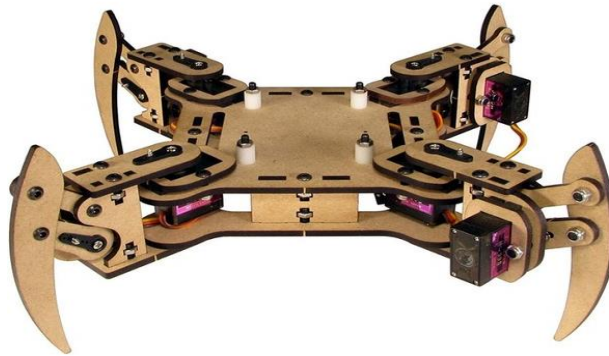
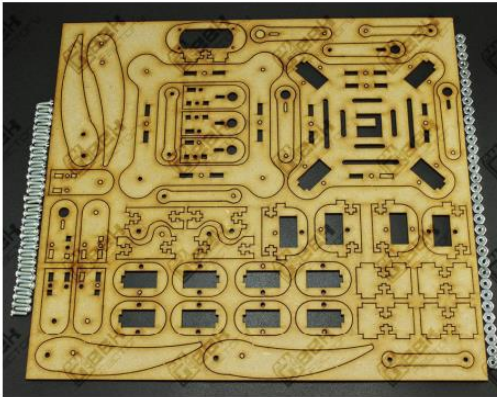
El LINKI es un proyecto de robot eficiente, porque con pocos recursos técnicos y económicos se ha conseguido una base útil y completa de robot andador. Además, es un proyecto de robot flexible ya que su diseño está abierto a muchas posibilidades de evolución para trabajos futuros.

FABRICACION Y MONTAJE

En este apartado se dan detalles y pasos de cómo se han montado todos los elementos que componen el LINKI, tanto las piezas impresas y servomotores que forman la estructura física como las conexiones eléctricas necesarias para el funcionamiento completo del robot.

Para la ejecución de este robot cuadrúpedo tipo araña hemos dispuesto de diferentes componentes y materiales así:

- 8 servomotores
- tarjeta nucleo- 446ZE
- estructura en acrílico de 3mm de espesor.
- 40 tornillos m3 x 10
- 16 tornillos m3 x 12
- 56 tuercas m3
- JOYSTICK



El código inicia con un serie de telecomandos, primero asignando las entradas de los servomotores

```
9 void MoverConjunto(uint8_t motores,  
0  
1  
2 DigitalIn entrada (PE_15);  
3 DigitalOut S3 (PB_0);  
4 DigitalOut S2 (PE_0);  
5 AnalogIn analog_1(A0);  
6 DigitalOut led(LED1);  
7  
8 PwmOut Servo1(PB_11);  
9 PwmOut Servo2(PB_10);  
0 PwmOut Servo3(PA_0);  
1 PwmOut Servo4(PE_14);  
2 PwmOut Servo5(PE_12);  
3 PwmOut Servo6(PE_10);  
4 PwmOut Servo7(PD_12);  
5 PwmOut Servo8(PD_13);  
6  
7
```

Asignación para poder mover el motor

```
101  
102  
103 void MoverMotor(uint8_t motor, uint8_t grados){  
104  
105 Transmi.printf("Se movera solo un motor. \n");  
106 if (grados <= 180){  
107  
108 AnchoPulso= ((grados*9.8)+535);  
109  
110  
111 switch (motor){  
112  
}  
}
```

MOTORXMOTOR

```
12
13 case 0x01:{
14   Servo1.pulsewidth_us(AnchoPulso);
15   Transmi.printf("Se mueve el motor 1.\n");
16   break;
17 }
18
19
20 case 0x02:{
21   Servo2.pulsewidth_us(AnchoPulso);
22   Transmi.printf("Se mueve el motor 2.\n");
23   break;
24 }
25
26 case 0x03:{
27   Servo3.pulsewidth_us(AnchoPulso);
28   Transmi.printf("Se mueve el motor 3.\n");
29   break;
30 }
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49 case 0x07:{
50   Servo7.pulsewidth_us(AnchoPulso);
51   Transmi.printf("Se mueve el motor 7.\n");
52   break;
53 }
54
55 case 0x08:{
56   Servo8.pulsewidth_us(AnchoPulso);
57   Transmi.printf("Se mueve el motor 8.\n");
58   break;
59 }
60
61 default:{
62   Transmi.printf("Motor no cargado. \n");
63 }
64 }
65
66 }
67
```

```
32 case 0x04:{
33   Servo4.pulsewidth_us(AnchoPulso);
34   Transmi.printf("Se mueve el motor 4.\n");
35   break;
36 }
37
38 case 0x05:{
39   Servo5.pulsewidth_us(AnchoPulso);
40   Transmi.printf("Se mueve el motor 5.\n");
41   break;
42 }
43 case 0x06:{
44   Servo6.pulsewidth_us(AnchoPulso);
45   Transmi.printf("Se mueve el motor 6.\n");
46   break;
47 }
48
49 case 0x07:{
50   Servo7.pulsewidth_us(AnchoPulso);
51   Transmi.printf("Se mueve el motor 7.\n");
```

PARA PODER MOVER UN COJUNTO DE MOTORES SE TIENE UNA ASIGNACION

DD

```
73
74
75 void MoverConjunto(uint8_t motores,uint8_t posicion){
76 Transmi.printf("Se movera un conjunto de motores. \n");
77
78 if (posicion==0xDD){
79
80     switch(motores){
81         case 1:{
82             Servo1.pulsewidth_us(550);
83             Servo2.pulsewidth_us(550);
84             Transmi.printf("Mover primer conjunto, ambos cerrados. \n");
85             break;
86         }
87
88         case 2:{
89             Servo3.pulsewidth_us(550);
90             Servo4.pulsewidth_us(550);
91             Transmi.printf("Mover segundo conjunto, ambos cerrados. \n");
92
93         }
94
95         case 3:{
96             Servo5.pulsewidth_us(550);
97             Servo6.pulsewidth_us(550);
98             Transmi.printf("Mover tercer conjunto, ambos cerrados. \n");
99             break;
100        }
101
102        case 4:{
103            Servo7.pulsewidth_us(550);
104            Servo8.pulsewidth_us(550);
105            Transmi.printf("Mover cuarto conjunto, ambos cerrados. \n");
106            break;
107        }
108
109        default: {
110            Transmi.printf("Motores no definidos. \n");
111        }
112    }
113 }
```

DA

```
else{
  if (posicion==0xDA) {

switch(motores){
  case 1:{
    Servo1.pulsewidth_us(2100);
    Servo2.pulsewidth_us(550);
    Transmi.printf("Mover primer conjunto, uno abierto, dos cerrado. \n");
    break;
  }

  case 2:{
    Servo3.pulsewidth_us(2100);
    Servo4.pulsewidth_us(550);
    Transmi.printf("Mover segundo conjunto, uno abierto, dos cerrado. \n");
    break;
  }

  case 3:{
    Servo5.pulsewidth_us(2100);
    Servo6.pulsewidth_us(550);

case 4:{
    Servo7.pulsewidth_us(2100);
    Servo8.pulsewidth_us(550);
    Transmi.printf("Mover cuarto conjunto, uno abierto, dos cerrado. \n");
    break;
  }

  default: {
    Transmi.printf("Motores no definidos. \n");
  }
}

else{
```

```
else{

if (posicion==0x1D){

switch(motores){
    case 1:{
        Servo1.pulsewidth_us(550);
        Servo2.pulsewidth_us(2100);
        Transmi.printf("Mover primer conjunto, uno cerrado, dos abierto. \n");
        break;
    }

    case 2:{
        Servo3.pulsewidth_us(550);
        Servo4.pulsewidth_us(2100);
        Transmi.printf("Mover segundo conjunto, uno cerrado, dos abierto. \n");
        break;
    }

    case 3:{
        Servo5.pulsewidth_us(550);
        Servo6.pulsewidth_us(2100);
        Transmi.printf("Mover tercer conjunto, uno cerrado, dos abierto. \n");
        break;
    }

    case 4:{
        Servo7.pulsewidth_us(550);
        Servo8.pulsewidth_us(2100);
        Transmi.printf("Mover cuarto conjunto, uno cerrado, dos abierto. \n");
        break;
    }

    default: {
        Transmi.printf("Motores no definidos. \n");
    }
}
else{
```

1ª

```
else{
  if (posicion==0x1A){

switch(motores){
  case 1:{
    Servo1.pulsewidth_us(2100);
    Servo2.pulsewidth_us(2100);
    Transmi.printf("Mover primer conjunto, ambos abiertos. \n");
    break;
  }

  case 2:{
    Servo3.pulsewidth_us(2100);
    Servo4.pulsewidth_us(2100);
    Transmi.printf("Mover segundo conjunto, ambos abiertos. \n");
    break;
  }

  case 3:{
    Servo5.pulsewidth_us(2100);
    Servo6.pulsewidth_us(2100);

    ,

  case 4:{
    Servo7.pulsewidth_us(2100);
    Servo8.pulsewidth_us(2100);
    Transmi.printf("Mover tercer conjunto, ambos abiertos. \n");
    break;
  }

  default: {
    Transmi.printf("Motores no definidos. \n");
  }
}

}

else{
  Transmi.printf("Movimiento no definido. \n");
}
```

VARIABLES PRINCIPALES PARA LA DETECCION DEL COLOR

```

360
361 void RutColor(){
362
363     S2=0;
364     S3=0;
365     rojo=Detectar();
366
367     S2=0;
368     S3=1;
369     azul=Detectar();
370
371     S2=1;
372     S3=1;
373     verde=Detectar();
374
375     if ((rojo>2)&(rojo<25) & (azul>35)&(azul<54) & (verde>53)&(verde<73)){
376         Transmi.printf("rojo. \n");
377         Servo1.pulsewidth_us(1025);
378         wait (0.5);
379         Servo2.pulsewidth_us(1319);

```

MOVIMIENTOS ASIGNADOS POR CADA COLOR

ROJO	<pre> if ((rojo>2)&(rojo<25) & (azul>35)&(azul<54) & (verde>53)&(verde<73)){ Transmi.printf("rojo. \n"); Servo1.pulsewidth_us(1025); wait (0.5); Servo2.pulsewidth_us(1319); wait (0.5); Servo1.pulsewidth_us(1412); wait (0.5); Servo7.pulsewidth_us(1025); wait (0.5); Servo8.pulsewidth_us(1809); wait (0.5); Servo7.pulsewidth_us(1417); wait (0.5); Servo2.pulsewidth_us(1515); Servo8.pulsewidth_us(1515); wait (0.5); Servo3.pulsewidth_us(1221); wait (0.5); Servo4.pulsewidth_us(1613); wait (0.5); </pre>
------	---

AZUL	<pre> else{ if ((rojo>130)&(rojo<152) & (azul>35)&(azul<56) & (verde>91)&(verde<114)) Transmi.printf("azul. \n"); Servo1.pulsewidth_us(1025); wait (0.5); Servo2.pulsewidth_us(1711); wait (0.5); Servo1.pulsewidth_us(1412); wait (0.5); Servo7.pulsewidth_us(1025); wait (0.5); Servo8.pulsewidth_us(1319); wait (0.5); Servo7.pulsewidth_us(1417); wait (0.5); Servo2.pulsewidth_us(1515); Servo8.pulsewidth_us(1515); wait (0.5); Servo3.pulsewidth_us(1221); </pre>
VERDE	<pre> if ((rojo>10)&(rojo<24) & (azul>25)&(azul<36) & (verde>23)&(verde<34)){ Transmi.printf("verde. \n"); Servo1.pulsewidth_us(700); wait (1); Servo3.pulsewidth_us(700); wait (1); Servo5.pulsewidth_us(700); wait (1); Servo7.pulsewidth_us(700); wait (1); Servo3.pulsewidth_us(700); wait (1); Servo1.pulsewidth_us(1500); wait (1); Servo3.pulsewidth_us(1500); wait (1); Servo5.pulsewidth_us(1500); wait (1); </pre>

El ingreso para la programación del LINKI inicia con unos Comandos establecidos estos “comandos” deben estar en código ascii el cual nos ayudara a configurar los movimientos que deseamos

FF= INICIO DE COMANDO
01/02/03/04= TIPO DE COMANDO
1D/DD/DA/1A= NUMERO DE MOTOR
ASCII= NUMERO DE GRADOS
FD= FIN DE COMANDO



ARAÑA LINKI FINAL

