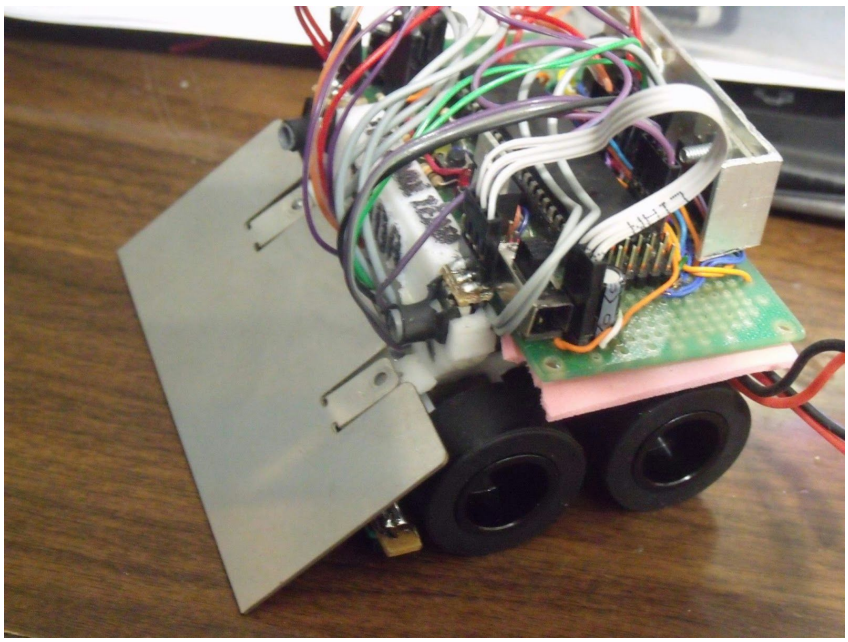
Competencia de robótica

JOHNNY 5

~2015~

Categoría: Mini-Sumo

Nombre robot: **Yoda**



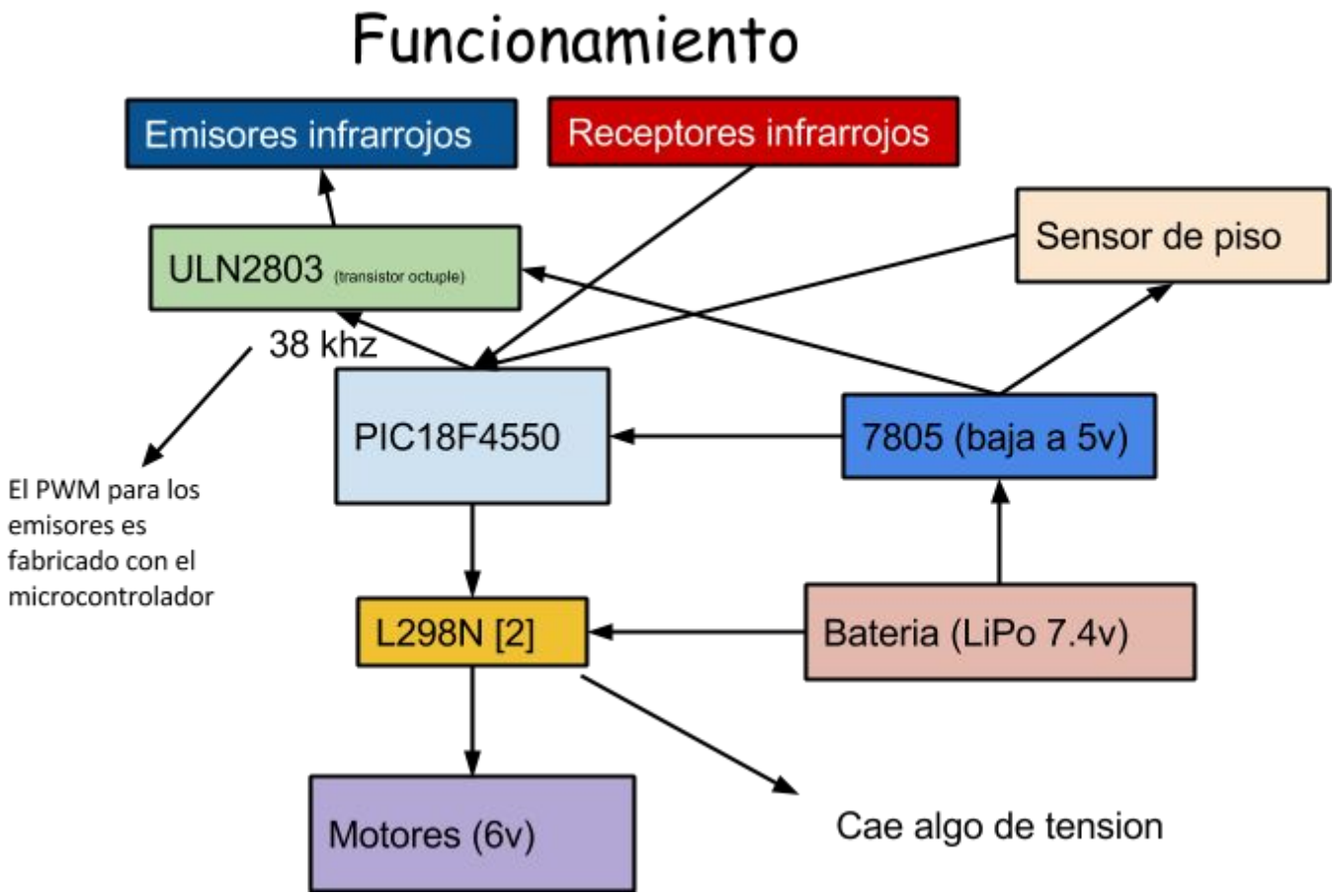
Institución: **ORT**

Participantes: (Nombre completo y mail)

- **Ariel Nowik (ariel.nowik@gmail.com)**
- **Mateo Salvatto (mateo.nicolas.salvatto@gmail.com)**
- **Agustin Chiari (rulitoslocos@gmail.com)**
- **Dylan Tasat (dylantasat11@gmail.com)**

Introduccion

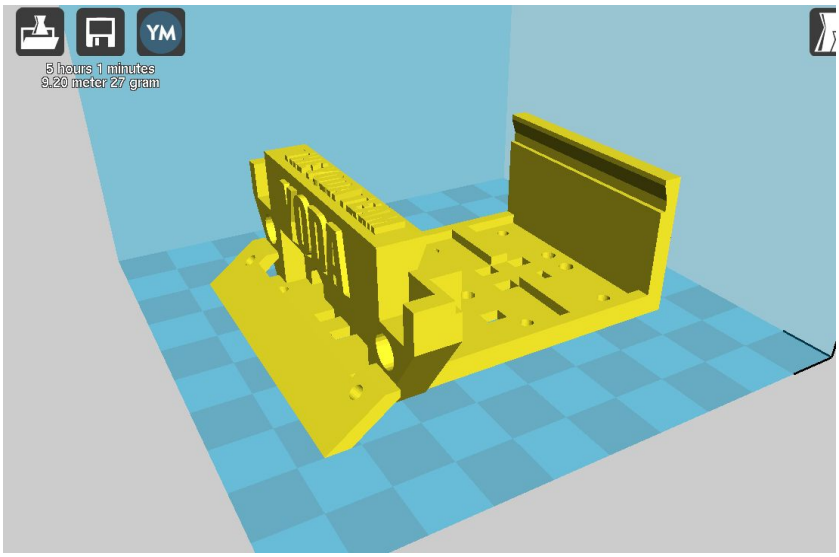
Ninguna de las derrotas constantes que tuvimos en los años anteriores, donde construimos entre otros robots a “**Adefesio**” y a “**Ussop**” fueron en vano. Aprendimos muchísimas cosas de los errores del pasado, y muchas de ellas fueron aplicadas en este nuevo robot “Yoda”. Por ejemplo, como en el robot “Ussop” el hecho usar una placa universal para soldar los componentes fue tan positivo ya que no sufrió jamás averías en ninguna competencia, decidimos nuevamente utilizar esta clase de placa nuevamente, pero esta vez medida en 5cmx10cm en vez de 10cmx10cm para ahorrar espacio valioso, a pesar de que esto significa no poder reproducir el mismo circuito muchas veces, pero es un costo que estamos dispuestos a afrontar. Otra de las particularidades de este robot es que fue impreso en 3D, lo cual nos permitió mayor precisión para colocar los motores y las rampas en el lugar indicado, además de poder marcar un lugar a presión muy preciso para lograr colocar los emisores infrarrojos y el ahorro de una gran cantidad de tiempo que previamente era gastado en la construcción de un chasis.



A diferencia de USSOP decidimos utilizar un único microcontrolador que genera la frecuencia necesaria para los emisores infrarrojos (38khz), para ocupar menos espacio en el circuito, y porque también el pic18f4550 tiene muchas más entradas y salidas libres de las que podemos llegar a usar por ahora (más de 32), considerando que el espacio del robot es muy limitado y no es posible agregar demasiadas cosas. Utilizamos en esta ocasión motores de 6v ya que si bien tienen *corriente mayor* en el caso promedio, es posible que funcionen con una batería de solo 7,4v, más chica que baterías similares de mayor tensión. De hecho las lipo 7.4v estan diseñadas para esta clase de motores ya que consideran que en el componente de conmutación (l298) suele caer la tensión necesaria para llegar a los 6v nominal de los motores.

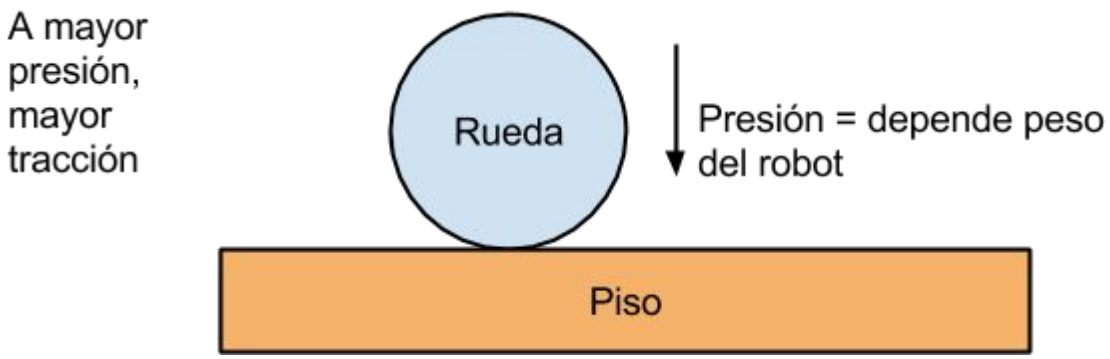
Mecanica

El chasis del robot fue diseñado en una impresora 3D, fueron necesarias muchísimas impresiones para lograr que quedara la medida exacta de los tornillos. El diseño contó también con un área para colocar la plaqueta electrónica y la batería *a presión*, más agujeros para colocar los emisores infrarrojos y los receptores



Tanto la rampa como los motores fueron atornillados a esta estructura.

- Se utilizaron motores de corriente continua, de 6v y 1000rpm, controlados con un pwm de una frecuencia relativamente baja (1 khz) para regular velocidades. El programa en sí permite diez diferentes tipos de velocidad para cada motor, de las cuales 0 es la mínima y 10 la máxima.
- La rampa fue encargada por internet (Pololu), y afilada en nuestra escuela con diferentes herramientas
- La batería utilizada fue una LiPo de 7,4v 1320 mah, consideramos que dura demasiado tiempo para lo que consume el robot, y que en un futuro podríamos utilizar una batería de menor mAh, que dure un poco menos pero que ocupe menos espacio del robot. No obstante la ventaja clara que tenemos es que es muy cómodo el robot en este sentido ya que hay que cargar la batería muy cada tanto.
- Queda pendiente en estos días encontrar alguna forma de agregarle más peso al robot (el robot de por si pesa tan solo 300 gr), puesto que de esta forma podemos incrementar la traccion de las ruedas del robot, ya que las mismas realizan de esta forma un mejor contacto contra el suelo

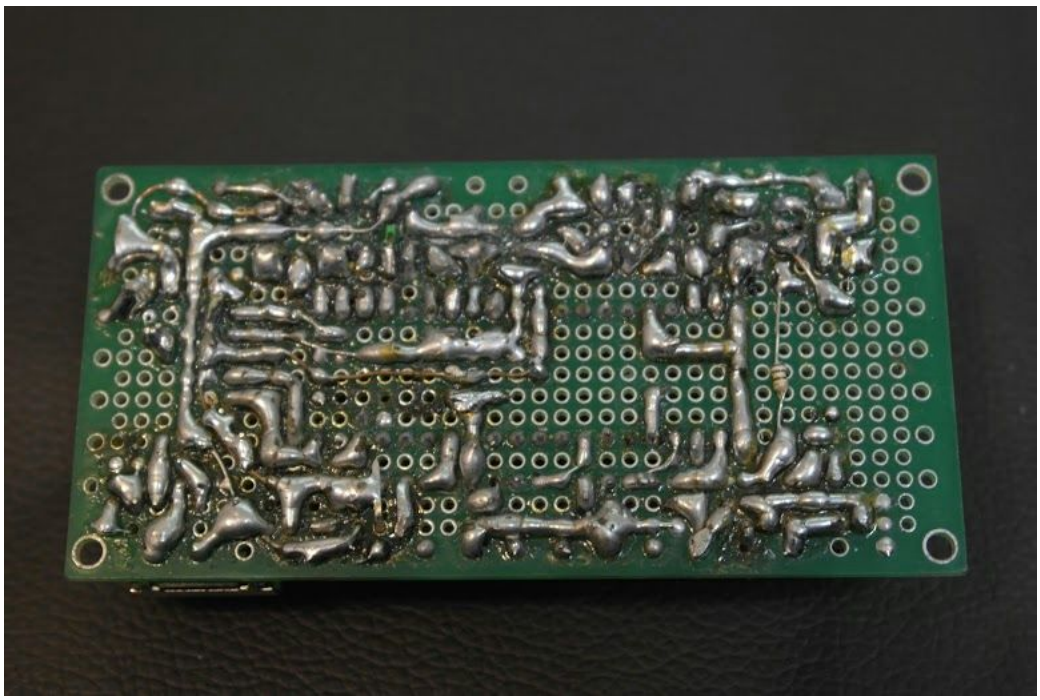


- Nos dimos cuenta entonces que la limitación de la fuerza que nuestro robot puede ejercer no está en los motores sino en la tracción de las ruedas, ya que si por ejemplo nuestro robot choca contra una

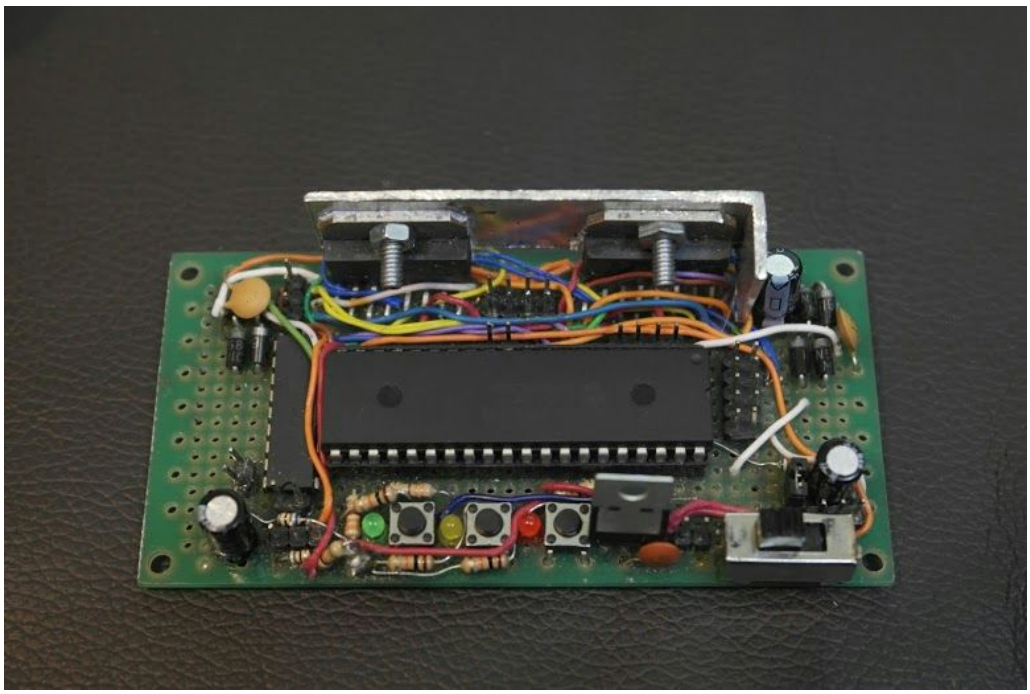
pared, las ruedas giran en el aire, no se traban los motores. No obstante, mientras más peso tenga nuestro robot la tracción es mejor.

Electronica

La electrónica fue montada en una plaqueta universal de 5x10, la cual consistió principalmente de un microcontrolador PIC18F4550, dos conmutadores de motores (l298n) más un ULN2802 preparado para conmutar los dos sensores infrarrojos a 38 khz con la corriente necesaria. El hecho fundamental que nos convenció de utilizar una plaqueta universal fue la solidez característica de las soldaduras resultantes, puesto que se elimina la posibilidad de levantamiento de pistas, ademas de la capacidad de resoldar el circuito constantemente.



Soldar esta placa en su totalidad nos llevó aproximadamente una semana de trabajo.



Utilizamos un 7805 para conseguir los 5v necesarios para el microcontrolador y los emisores, ademas de 3 leds y 3 pulsadores para testear el circuito.
Para disminuir el desgaste de nuestros motores utilizamos diodos (Ocho en total), los cuales impiden que, al cambiar de estado, el motor sea desgastando por picos de corriente.
Utilizamos dos puentes H (L298N) para ser capaces de duplicar la corriente maxima que utilizamos en nuestros motores

Programacion

Para programar nuestro PIC18F4550 utilizamos un programador comercial Pick It 2, proveido por nuestra institución y el software gratuito de MicroChip, MPLABX.

La lógica fundamental y principal de nuestro programa consiste en cuatro estados principales:

- Inicio: En este estado testeamos los sensores de distancia (Utilizando los botones y leds previamente mencionados). En caso de ser presionado el boton de activación, se pasa estado de espera.
- Espera: Se esperan cinco segundos (Reglamentarios) y se pasa directamente al estado de búsqueda
- Búsqueda: En este estado se procede a girar a la derecha o la izquierda según el estado de una variable, que puede cambiar entre uno y cero. En caso de que alguno de los sensores infrarrojos de frente detecte al enemigo, se pasará directamente al último estado: Ataque.
- Ataque: En este estado el robot avanzará directamente, lo más rápido que pueda, en direccion al enemigo, en caso de que ambos sensores esten detectando simultáneamente. De no ser así, girara hacia el lado del sensor que esté detectando (Y si ninguno de los detecta, girara hacia el último lado que haya detectado, ya que puede recordar los estados previos de los sensores).

El lenguaje que utilizamos para programar fue “C”, con el compilador gratuito XC8 de MicroChip.

Conclusiones

Estamos muy satisfechos de haber logrado combinar todas nuestras experiencias con robots MiniSumo aprendiendo de nuestros errores y haber creado un robot sólido, rápido y perfectamente funcional.
Creemos que podemos realizar algunas modificaciones en el futuro para mejorar el rendimiento de nuestro robot, como por ejemplo agregar sensores de piso, diseñar un compartimento para el peso extra (Plomo), agregar más sensores infrarrojos, entre otras cosas.
Los componentes que utilizamos en este caso (En su gran mayoría) los importamos personalmente desde Estados Unidos (Comprados en www.pololu.com)

Los componentes que utilizamos se dividieron entre los comprados en el exterior y los que nos proporcionó nuestra institución.
Por un lado, compramos nuestra rampa, los motores, tuercas y tornillos especiales, las ruedas, la placa universal y utilizamos nuestra propia impresora 3D para la construcción del chasis.

Por otro lado, el colegio nos proporcionó tanto el microcontrolador como el resto de los componentes (Resistencias, diodos, L289N, entre otros)

Para finalizar, adjuntamos el codigo fuente de nuestra programacion:
(Siendo el siguiente el archivo main.c)

```
#include "ybot.h"

int sa;
int suma[4];
int a,b,mode;
int giro=0;
int flag;

//mayor a 900 es blanco

enum {ST , ADELANTE , WAIT ,LINE_A ,LINE_B , BUSCAR , ENCONTRADO , ADELANTE_TEST};
enum {NODETECT , DETECT };

/*** Funcion que decide si el sensor de piso ve que nos fuimos de la pista ***/
char PisoDetect(int sensor){
  if (PISO[sensor] > 990){
    if (suma[sensor] < 30){
      suma[sensor]++;
    }
  }else{
    suma[sensor] = 0;
  }
  if (suma[sensor] >= 30){
    return DETECT;
  }
  return NODETECT;
}
void CheckLine(){
  if (PisoDetect(A)){
    SetStatus(LINE_A);
  }
  if (PisoDetect(B)){
    SetStatus(LINE_B);
  }
}

int main(int argc, char** argv) {
  initYBOT();
```

```
sa = 0;
mode = 0;
a = 0;
flag = 0;

SetStatus(ST);
int x;
for (x = 0;x < 4;x++){
   suma[x] = 0;
}

while (1){
   UpdateYBOT();

   switch (status){
      case ST:
         L_AMARILLO = VIST[A][LEJOS] >= 5;
         L_VERDE = 1;
         L_ROJO = VIST[B][LEJOS] >= 5;
         if (B_VERDE_PRESS()){
            giro=0;
         }
         if (B_ROJO_PRESS()){
            giro=1;
         }
         if (flag == 0){
            if (ACTIVACION == 0){
               flag = 1;
            }
         }else if(flag == 1){
            if (ACTIVACION == 1){
               SetStatus(WAIT);
            }
         }
         if (B_AMARILLO_PRESS()){
            SetStatus(ADELANTE_TEST);
         }
      break;
      case ADELANTE_TEST:
         MotorsSpeed(10,10);
         L_ROJO = CLOCK.miliseconds < 500;
         L_AMARILLO = CLOCK.miliseconds < 500;
         L_VERDE = CLOCK.miliseconds < 500;
```

```
    if (B_AMARILLO_PRESS()){
      SetStatus(ST);
      MotorsSpeed(0,0);
    }
  break;
  case WAIT:
    L_VERDE = CLOCK.miliseconds % 200 > 100;
    L_AMARILLO = CLOCK.miliseconds % 200 <= 100;
    L_ROJO = CLOCK.miliseconds % 200 > 100;
    if (CLOCK.seconds >= 5){
      SetStatus(BUSCAR);
    }
  break;
  case BUSCAR:
    L_VERDE = 0;
    L_AMARILLO = 0;
    L_ROJO = 0;
    if (giro == 0){
      MotorsSpeed(0,9);
    }else{
      MotorsSpeed(9,0);
    }
    if (VIST[A][LEJOS] >= 1 || VIST[B][LEJOS] >= 1){
      SetStatus(ENCONTRADO);
    }
  break;
  case ENCONTRADO:
    L_VERDE = 1;

    if (VIST[A][LEJOS] >= 1 && VIST[B][LEJOS] < 1){
      mode = 0;
      MotorsSpeed(7,-7); //4,0
      L_ROJO = 0;
      L_AMARILLO = 1;
    }else if (VIST[A][LEJOS] < 1 && VIST[B][LEJOS] >= 1){
      mode = 1;

      MotorsSpeed(-7,7); //4,0
      L_ROJO = 0;
      L_AMARILLO = 0;
    }else if (VIST[A][LEJOS] >= 1 && VIST[B][LEJOS] >= 1){
      L_ROJO = 1;
      L_AMARILLO = 0;
      MotorsSpeed(10,10);
```

```
            }else{
               if (mode == 0){
                  MotorsSpeed(0,7); //4,1
               }else if(mode == 1){
                  MotorsSpeed(7,0); //4,1
               }
            }
         break;
      }

   }
}
```