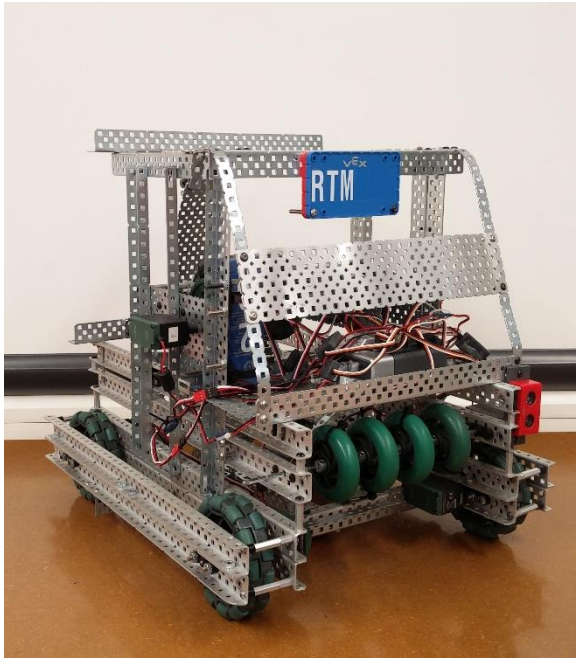
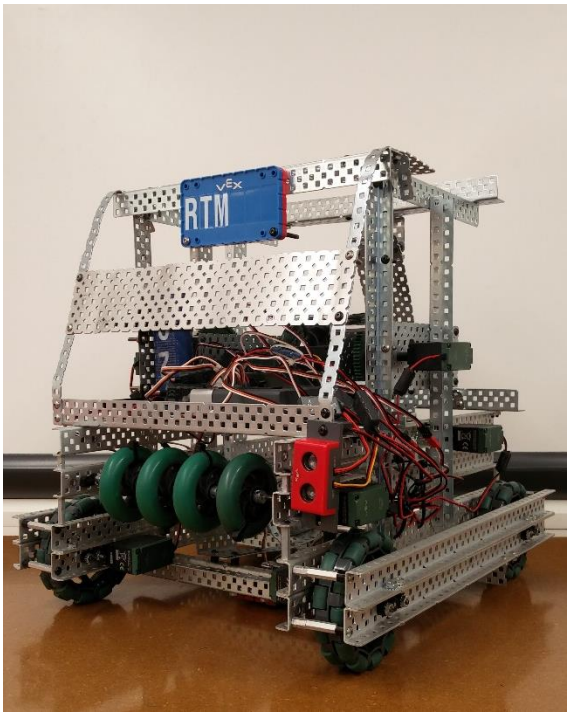



13/03/2016

# VRC Middle/High School
# ROBO-TECH MUNDET

*SPAIN - Barcelona*

Carles Núñez, Jordi González, Carolina Arias,
Hugo Limoso, Adrià Sánchez, Dani Suñé,
Joan Caballero, Alexis Molina

# VRC Middle/High School ROBO-TECH MUNDET

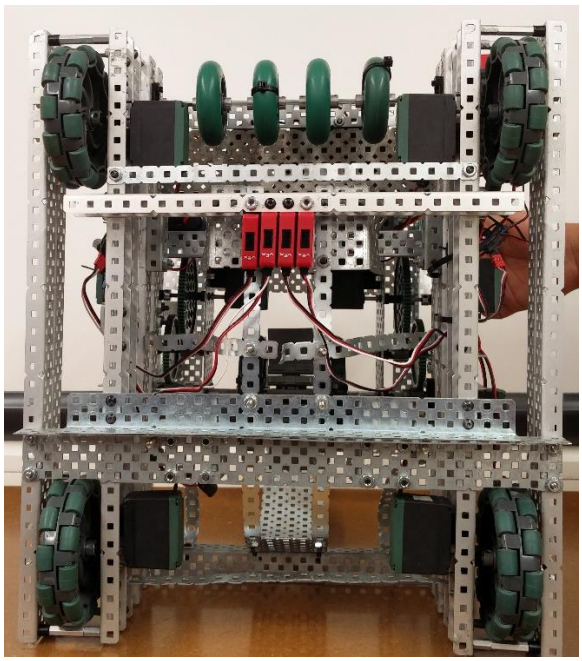
*SPAIN - Barcelona*

## Introducción

El robot que presentamos a la competición nacional es el resultado de tres meses de investigación y aprendizaje, el cual ha ido mejorando día tras día.

### La base

En primer lugar se diseñó la base y se instalaron cuatro ruedas motrices, vimos que tenía un giro muy aparatoso, eso nos llevó a evolucionar el sistema motriz a cuatro ruedas omnidireccionales de 4". Usamos para el desplazamiento cuatro motores. El chasis, principalmente era en H completamente, a medida que hemos adecuado el robot al juego se puede apreciar menos el tipo de chasis.



Esta base decidimos usarla después de hacer un estudio a las diferentes bases que otros aficionados han enseñado por internet. Primero de todo recopilamos la información y nos pusimos a estudiar cual sería la forma que más se adecuaría al juego presentado este año 2016.

Para controlar la posición del Robot hemos puesto un sensor sónar y cuatro seguidores de línea, más adelante explicaremos el porqué de esta configuración sensórica.

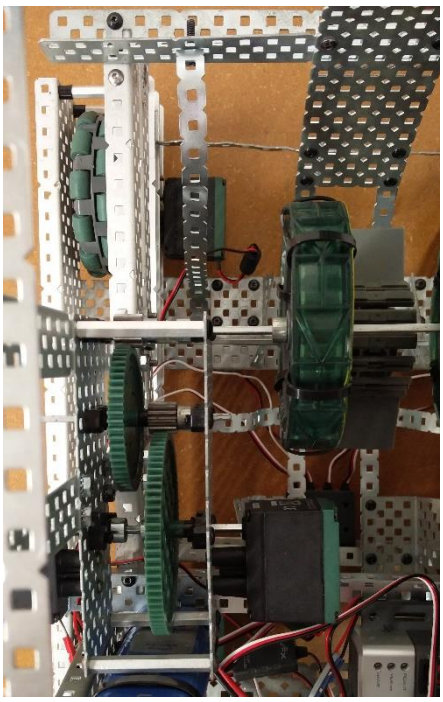
## El lanzador

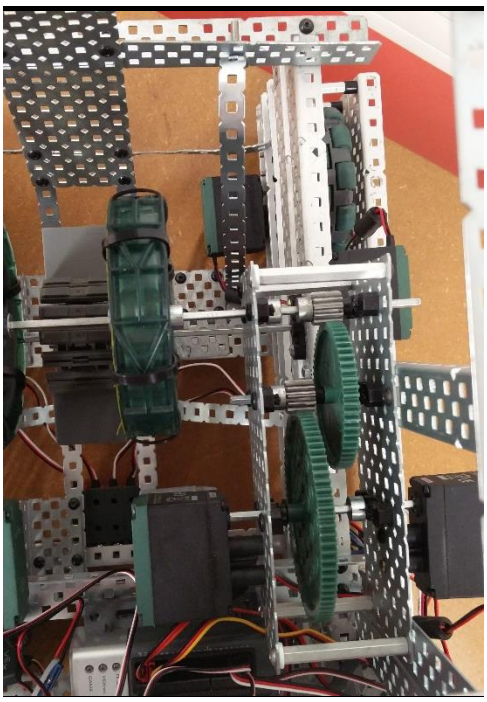
El lanzador ha sido una de las cosas que más nos ha costado recopilar información y decidir el sistema final, se han probado diferentes configuraciones para conseguir tirar desde Base.

Todos compartimos la idea de que el mejor sistema consistía en usar una secuencia de engranajes 7:1 * 5:1 proporcionando en teoría hasta 3500 revoluciones por minuto. Impulsados por 4 motores de 100RPM y el uso de dos ruedas de 5 “hace que las pelotas lleguen con facilidad y un ángulo adecuado para encestar.

Debido a la falta de material no hemos podido recrear a la perfección la configuración descrita, por tal cosa primero usamos una configuración de 5:1 * 5:1 y ruedas de 4”. Eso nos limitaba mucho el lanzamiento a solo llegar desde justo tocar la red, eso nos llevó a intentar conseguir el material adecuado y pudimos arreglar una de las dos cosas, cambiamos el sistema de engranajes al ideal 7:1 * 5:1. Con eso ya hubo suficiente para llegar desde base aunque de una manera bastante apurada.

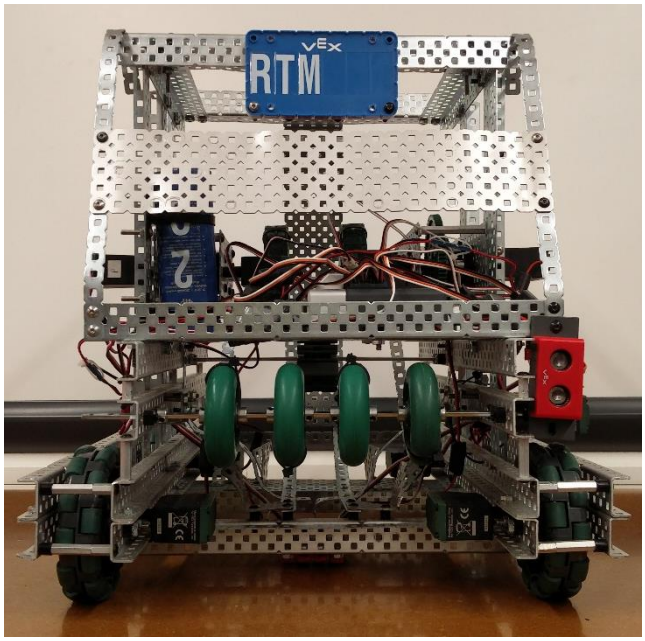
Aquí vemos el lanzador final:





## El recogedor de Pelotas delantero

Se ha realizado un recogedor de pelotas delantero que funciona con un motor a 100RPM. Permite coger las pelotas que están debajo a mucha velocidad e impide que una vez dentro salgan.




## Alineador de Pelotas Ascendente

Se ha realizado un alineador de pelotas ascendente que permiten hacer llegar un máximo de 3 pelotas al lanzador, es muy rápido ya que la distancia que recorre es muy pequeña, logrando lanzar una pelota por segundo.

## Sensórica y Programación

En este apartado no hablaremos simplemente de los sensores utilizados sino también de nuestra idea referente al control del robot autónomamente.

La idea principal era poner tres sensores encoders de motor, cuatro seguidores de línea y un sónar.

Actualmente de esa idea principal hemos puesto el sensor sónar y los cuatro seguidores de línea.

El porqué de este cambio de decisión es que por el tiempo que tenemos de autónomo, 15”, son los sensores que tenemos tiempo de usar. Todo y el cambio de normativa a la evaluación de la programación, os queremos argumentar que desde nuestro punto de vista, no es mejor programación la que más sensores tenga sino la que mejor partido saque a los sensores utilizados.

Primero hablaremos del sónar:

El sónar nos ha evitado la instalación de los dos sensores encoders para controlar el movimiento. Nos basamos a decir esto ya que en nuestra programación nuestro robot siempre está apuntando a la red cual cosa podemos detectar el tubo de PVC de la zona baja de puntuación.

Para su programación simplemente hemos usado el siguiente código:

```
#pragma config(Sensor, dgtl11,  sonarSensor,    sensorSONAR_cm)

while(SensorValue(sonarSensor) > 25 || SensorValue(sonarSensor) == -1)

{

}
```

Explicamos el siguiente código. El sensor sónar tenemos puesto que nos detecte en centímetros, por lo tanto en esta sentencia WHILE(cuando) le estamos declarando dos cosas, la primera es que nos adquiera el valor del sónar y nos haga el contenido del while siempre que el valor del sónar sea mayor a 25cm, por otra parte, cuando el sónar no recibe la onda enviada, su resultado es -1, es por eso que le hemos declarado que si su valor es -1 no continúe dentro del while ya normalmente tiene micro cortes.

Por acabar este apartado, hablaremos del seguimiento de línea:

¿Por qué tenemos cuatros sensores seguidores de línea? Definitivamente no es por la normativa de cuantos más sensores mejor. Nuestra explicación es la siguiente:

Para aprovechar el máximo el tiempo autónomo, buscamos que el robot pudiera tirar pelotas mientras sigue la línea, es por eso que queremos que aparte de seguir la línea lo haga de la forma más recta posible. Calculamos la separación entre seguidores y el ancho de la línea y vimos que puede detectar dos sensores a la vez de forma muy justa, así cuando nuestros seguidores de línea centrales detecten línea la precisión es absoluta.

Ahora entraremos a su programación, para ellos nos hemos basado en el concepto de lógica difusa. Consiste en enseñar al robot todas las opciones posibles. Después de haber estudiado su comportamiento vimos 9 posibilidades, la cual cosa la hemos declarado en una misma función.

El código es el siguiente:

```
#pragma config(Sensor, in1,    Derecho1,       sensorLineFollower)
#pragma config(Sensor, in4,    Izquierdo1,     sensorLineFollower)
#pragma config(Sensor, in3,    CentralIzq1,    sensorLineFollower)
#pragma config(Sensor, in2,    CentralDer1,    sensorLineFollower)
                int Derecho = (SensorValue(Derecho1));
                int CentralDer = (SensorValue(CentralDer1));
```

```
int CentralIzq = (SensorValue(CentralIzq1));

int Izquierdo = (SensorValue(Izquierdo1));


// Sensor Derecho ve fuera de la linea

if(  Derecho>threshold && CentralDer>threshold && CentralIzq>threshold &&
Izquierdo<threshold )

{


    // Corregir Derecho


    motor[MotorIzquierdoFrontal]  = -40;

    motor[MotorIzquierdoTrasero]  = -40;

    motor[MotorDerechoFrontal] = 20;

    motor[MotorDerechoTrasero] = 20;


}

// Sensor Central ve linea negra

if(  Derecho>threshold && CentralDer>threshold && CentralIzq<threshold &&
Izquierdo<threshold )

{


    // Corregir Izquierda

    motor[MotorIzquierdoFrontal]  = 20;

    motor[MotorIzquierdoTrasero]  = 20;

    motor[MotorDerechoFrontal] = 30;

    motor[MotorDerechoTrasero] = 30;

}

if(  Derecho>threshold && CentralDer>threshold && CentralIzq<threshold &&
Izquierdo>threshold )
```

```
{


    // Corregir Izquierda
    motor[MotorIzquierdoFrontal]  = 30;
    motor[MotorIzquierdoTrasero]  = 30;
    motor[MotorDerechoFrontal] = 50;
    motor[MotorDerechoTrasero] = 50;
}
if(  Derecho>threshold && CentralDer<threshold && CentralIzq<threshold &&
Izquierdo>threshold )
{
    // Corregir Derecha e Izquierda
    motor[MotorIzquierdoFrontal]  = 50;
    motor[MotorIzquierdoTrasero]  = 50;
    motor[MotorDerechoFrontal] = 50;
    motor[MotorDerechoTrasero] = 50;
}
if(  Derecho>threshold && CentralDer<threshold && CentralIzq>threshold &&
Izquierdo>threshold )
{


    // Corregir Derecha e Izquierda
    motor[MotorIzquierdoFrontal]  = 50;
    motor[MotorIzquierdoTrasero]  = 50;
    motor[MotorDerechoFrontal] = 30;
    motor[MotorDerechoTrasero] = 30;
}
// Sensor Izquierdo ve Linea
```

```
if(  Derecho<threshold && CentralDer<threshold && CentralIzq>threshold &&
Izquierdo>threshold )

{

        // Corregir Derecha

motor[MotorIzquierdoFrontal]  = 30;

motor[MotorIzquierdoTrasero]  = 30;

motor[MotorDerechoFrontal] = 20;

motor[MotorDerechoTrasero] = 20;

}


if(  Derecho<threshold && CentralDer>threshold && CentralIzq>threshold &&
Izquierdo>threshold )

{

//Corregir Izquierda

motor[MotorIzquierdoFrontal]  = 50;

motor[MotorIzquierdoTrasero]  = 50;

motor[MotorDerechoFrontal] = 20;

motor[MotorDerechoTrasero] = 20;

}

if(  Derecho<threshold && CentralDer<threshold && CentralIzq<threshold &&
Izquierdo>threshold )

{

        // Corregir Derecha

motor[MotorIzquierdoFrontal]  = -5;

motor[MotorIzquierdoTrasero]  = -5;

motor[MotorDerechoFrontal] = -5;

motor[MotorDerechoTrasero] = -5;

}
```

```
if(  Derecho<threshold && CentralDer<threshold && CentralIzq<threshold && Izquierdo<threshold )
{
    motor[MotorIzquierdoFrontal]  = 0;
    motor[MotorIzquierdoTrasero]  = 0;
    motor[MotorDerechoFrontal] = 0;
    motor[MotorDerechoTrasero] = 0;
}
```

Como se puede observar, tenemos las 9 posibilidades diferenciadas en los distintos IF.

Para empezar a explicar el código, hemos puesto el valor de los sensores en variables para así trabajar de una forma más cómoda.

Después de ello, hemos tenido que crear un threshold (umbral) para interpretar el funcionamiento de seguidor de línea. El valor de sensor al detectar blanco es de 180, y cuando detecta negro es de 1700, por lo tanto hemos cogido un número aproximado a la mitad para hacer la distinción de blanco o negro.

El funcionamiento de los diferentes IF consiste en decirle si el sensor está por encima del umbral (detecta negro) o por debajo (detecta blanco).

Para acabar esta apartado, explicamos cómo hacemos para encontrar una línea y así poder usar la función anterior:

Para ello hemos creado otra función a parte que consiste en hacer avanzar el robot hacia adelante hasta detectar una línea y entonces se para así dando paso a la función para seguir la línea.

El código es el siguiente:

```
void buscalinea()
{

int threshold = 600;

while(SensorValue(Derecho1)>threshold && SensorValue(CentralDer1)>threshold &&
SensorValue(CentralIzq1)>threshold && SensorValue(Izquierdo1)>threshold )
{

                motor[MotorIzquierdoFrontal]  = 100;
                motor[MotorIzquierdoTrasero]  = 100;
                motor[MotorDerechoFrontal] = 100;
                motor[MotorDerechoTrasero] = 100;
}
                motor[MotorIzquierdoFrontal]  = 0;
                motor[MotorIzquierdoTrasero]  = 0;
                motor[MotorDerechoFrontal] = 0;
                motor[MotorDerechoTrasero] = 0;
}
```

## Conclusiones

Primero de todo queremos agradecer la oportunidad que se nos da a los estudiantes con competiciones a nivel mundial, nos ha ayudado a completar gran parte de los estudios pudiendo ver, tocar y montar un robot.

Llegar hasta el día de la competición ha sido un trabajo duro. Primero estuvimos aprendiendo en que consiste el montaje de un robot haciendo diferentes diseños como el clawbot. Una vez conseguida la primera etapa, nos documentamos sobre los objetivos y normas de la competición e hicimos una lluvia de ideas para poner en común nuestros conocimientos y poder crear un robot al gusto de los ocho que somos.

Para completar el brainstorming investigamos las soluciones de diferentes aficionados encontrados por foros y videos. Allí acabamos de decidir nuestro primer diseño competitivo, separando las partes principales, el movimiento, la recolección de pelotas y su lanzamiento.

A partir de aquí vinieron los problemas: ¿Por qué el robot no hace exactamente el resultado teórico? Esta pregunta nos ayudó a sincronizar todas nuestras ideas y adaptarlas a la realidad. El primer proyecto, se movía pero no giraba, el recogedor solo cogía una pelota y el lanzador no llegaba a la zona alta de puntuación.  Poco a poco empezamos a refinar la mecánica y el robot iba subiendo niveles.

Siguiendo las mejoras llegamos al campeonato de Barcelona, el cual teníamos un robot aceptable, ya era capaz de coger varias pelotas a la vez tanto como priorizar la pelota naranja, el lanzador conseguimos que llegara a la zona alta de puntuación pero estando muy cerca de ella, y para acabar, no pudimos mejorar mucho el giro del robot.

En esta primera competición aprendimos como nunca. Solo decir que el lunes siguiente desmontamos casi todo el robot con nuevas ideas mucho más perfeccionadas. Nos pusimos manos a la obra y en dos semanas teníamos robot nuevo, el cual de manera satisfactoria arreglamos todos los problemas anteriores.

Para acabar, no teníamos suficiente con tener un robot funcionalmente competitivo, queríamos más. Entonces nos centramos en hacer una parte autónoma resolutiva. Para ello nos informamos de todos los tipos de sensores existentes en Vex, así escogiendo dos de ellos, Sónar y seguido de línea. No nos ha sido nada fácil programar el seguidor de líneas, hemos tenido que sincronizar sensores con el movimiento del robot.

Gracias a estas modificaciones, en 15 segundos de autónomo conseguimos lanzar 8 pelotas a la zona alta de puntuación entre ellas 2 naranjas, sumando un total de 50 puntos como mejor marca. Después en el modo manual siempre dependiendo de las habilidades de nuestro driver, estimamos lanzar 3 pelotas naranjas y diez verdes, siempre del campo, así logrando 80 puntos más.

Nosotros mismos nos hemos sorprendido de todo lo que hemos aprendido paso a paso sin darnos cuenta, creemos que tenemos posibilidades de ganar, tenemos ganas de conseguir el máximo reconocimiento.