Sistemas de Control para Robots
PRÁCTICA 3

PLANIFICACION LOCAL

Pedro Barquín Ayuso
Miguel Ballesteros García

Algoritmo de evitación de obstáculos ND (NearnessDiagramNavigation)

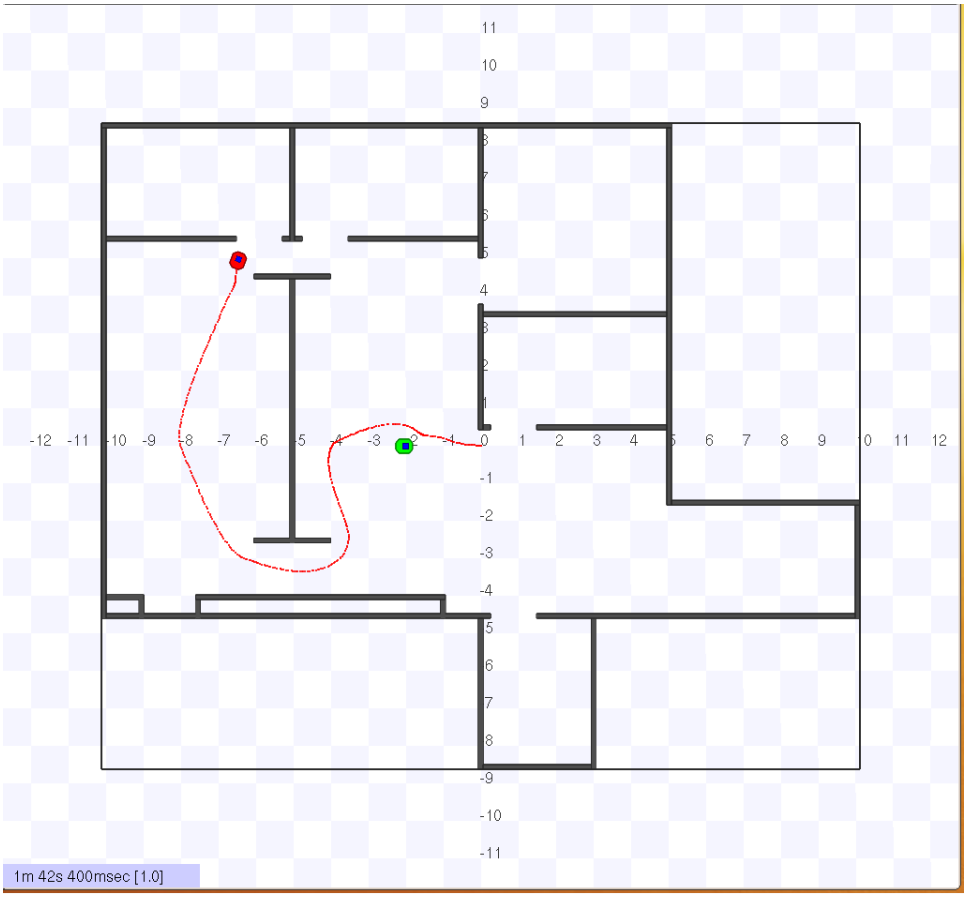
-Describe el funcionamiento de dicho algoritmo, empleando para ello los artículos que seencuentran en la página del propio driver dentro del proyecto Player/Stage.

El algoritmo se utiliza en entornos altamente poblados o densos. Por lo que vemos en los artículos nos explica la estrategia “Divide and Conquer”. Esta estrategia empieza por la localización de los obstáculos por medio de los sensores. Siguiendo ciertas características de los obstáculos se crean regiones. Con los obtaculos, el robot y la posición de destino se crean las entradas a un árbol que se recorrerá en base a la posición actual y unos criterios. En función de la situación final del árbol efectuamos una acción para avanzar.

-Comprobar su funcionamiento, empleando 4 ó 5 posiciones de destino sobre el entorno yempleando otro robot como obstáculo.

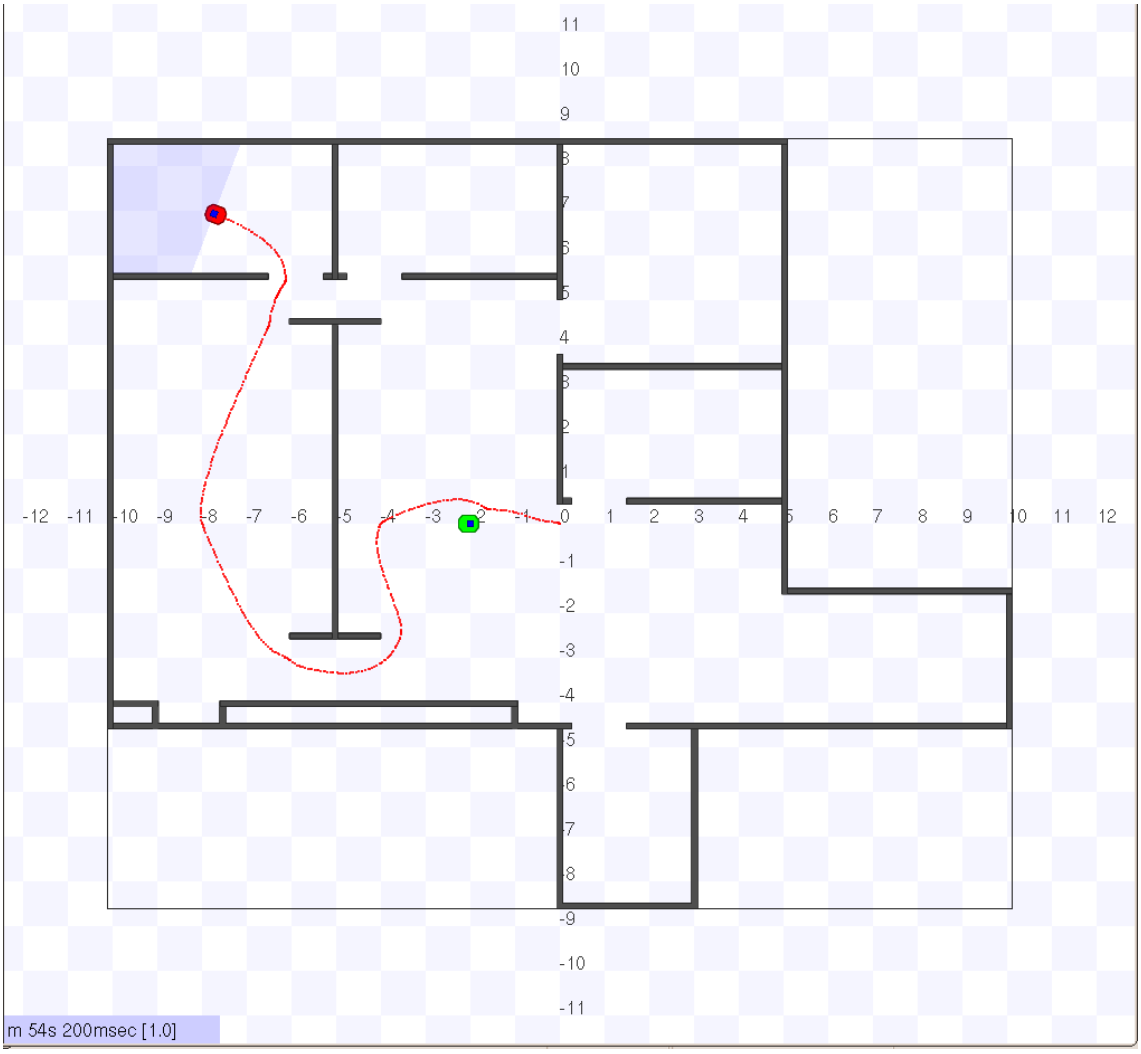
En esta parte hemos realizado dos recorridos compuestos por puntos estratégicamente puestos para el robot realice diferentes pruebas.

Recorrido 1Fallo:

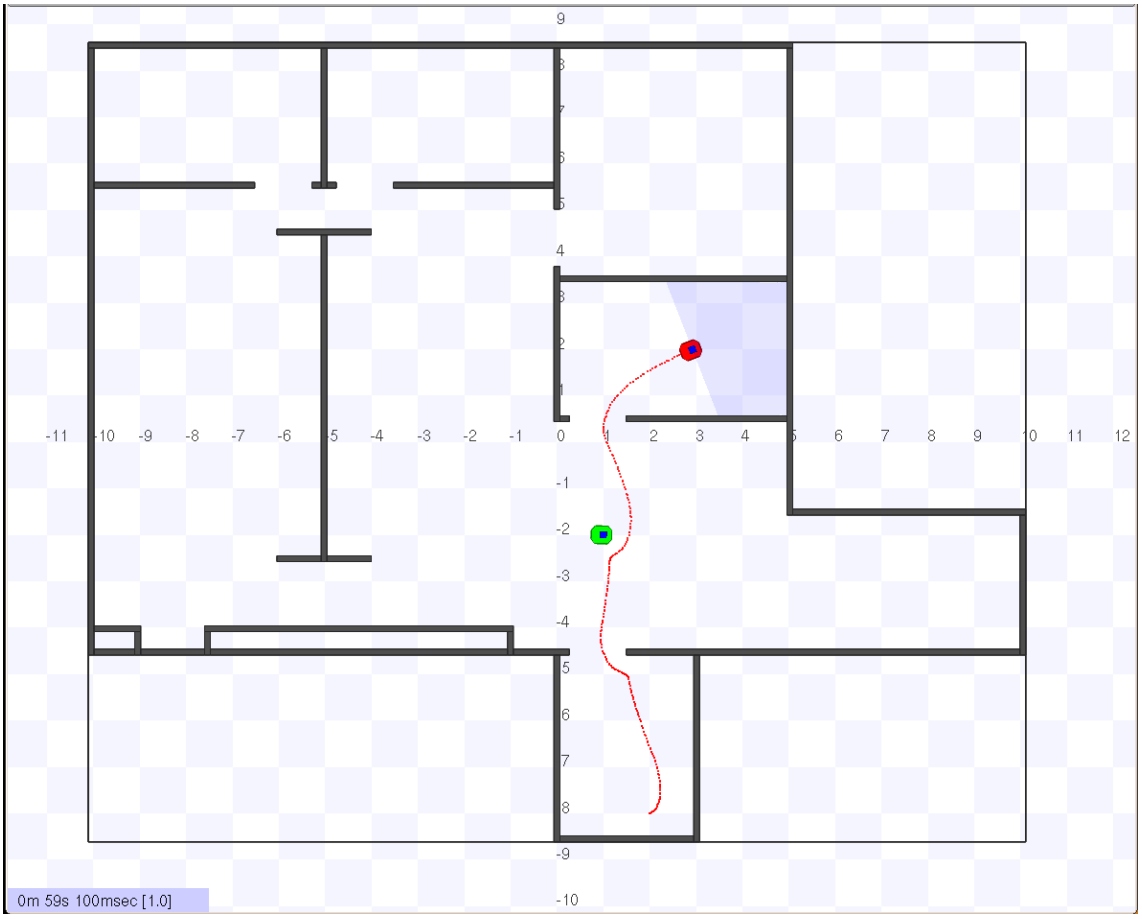


El robot debería entrar en la habitación de arriba pero debido al paso estrecho no continua y se queda bloqueado. Tras encontrar este fallo modificamos los valores del algoritmo para conseguir que el robots alcanzara la posición de destino

Recorrido 1:



Recorrido 2:



-Configurar el dispositivo nd. Empleando los parámetros disponibles en el mismo.

```
driver(
  name "nd"
  provides ["position2d:1"]
  requires ["output:::position2d:0" "input:::position2d:0" "laser:0"]

  max_speed [0.3 30.0]
  min_speed [0.1 10.0]
  goal_tol [0.1 15.0]
  wait_on_stall 1

  rotate_stuck_time 5.0
  translate_stuck_time 5.0
  translate_stuck_dist 0.15
  translate_stuck_angle 10.0

  avoid_dist 0.25
  safety_dist 0.1

  laser_buffer 1
  sonar_buffer 1
)
```

-¿Qué efecto tiene cada uno sobre el comportamiento del robot?

-goal_tol: limites lineales y de rotación, una vez que el robot supera estos límites se para.

-max_speed: velocidad máxima lineal y de rotación absolutas del robot.

-min_speed: velocidad minima lineal y de rotación absolutas del robot (no puede ser 0)

-avoid_dist: distancia a la que se empieza a evitar el obtaculo.

-safety_dist: medida extra alrededor del robot al evitar obstáculos.

-rotate_stuck_time**:** tiempo máximo que pueda rotar el robot sin realizar un avance hacia la ubicación de destino

-translate_stuck_time: tiempo máximo de avance del robot sin acercarse a la ubicación de destino
-translate_stuck_dist:

-translate_stuck_angle:

-wait_on_stall: se para la navegación local al saltar este flag.

-laser_buffer: número de escaneos del laser a tener en cuanta en la navegación local.

-sonar_buffer: número de escaneos del sonar a tener en cuanta en la navegación local.

-sonar_bad_transducers: información a desechar de cierto sonar

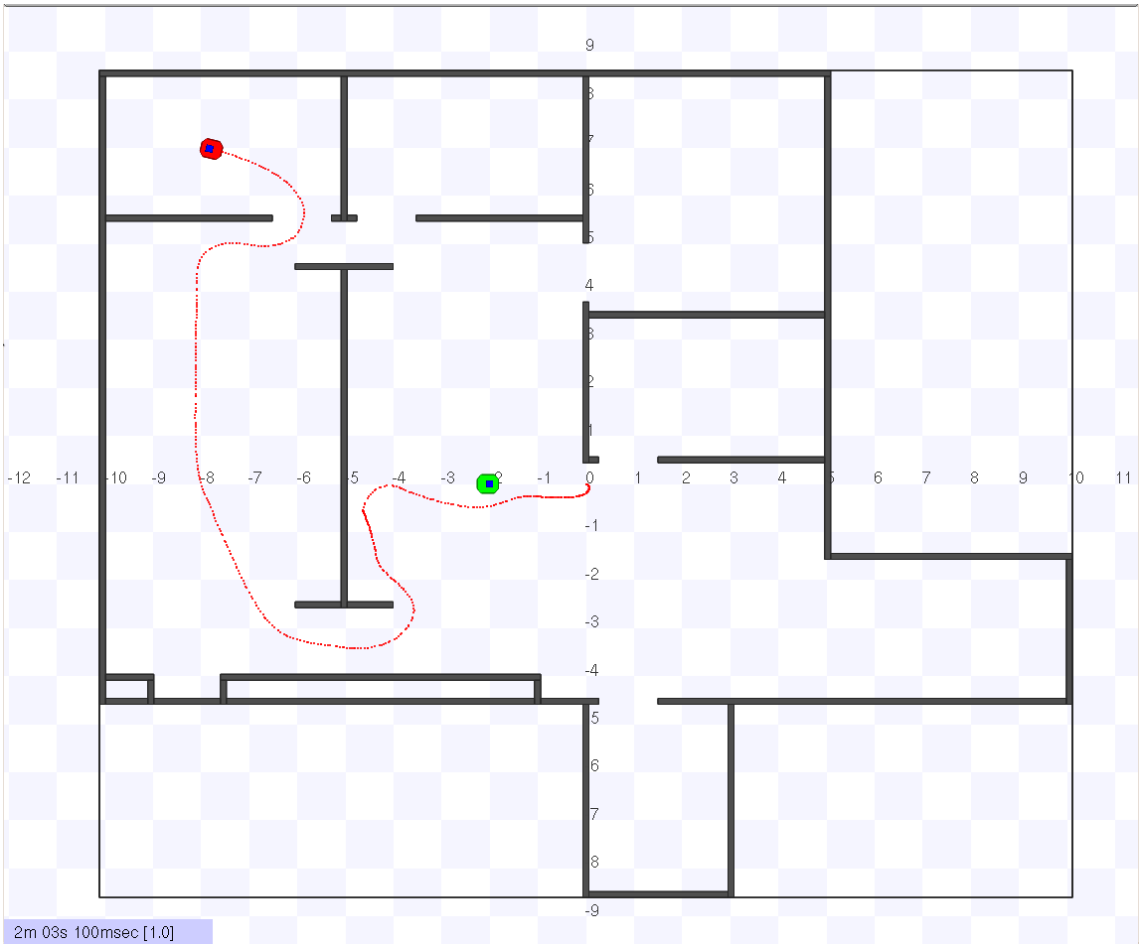
<u>Problemas Encontrados</u>

El robot no ve el otro robot y se estrella: la altura a la que está el laser es más alta que la altura del otro robot y al hacer pruebas con playerv vemos que efectivamente el laser no lo ve. Solucionamos esto añadiendo el laser al robot que está quieto para que pueda verle el robot que se mueve

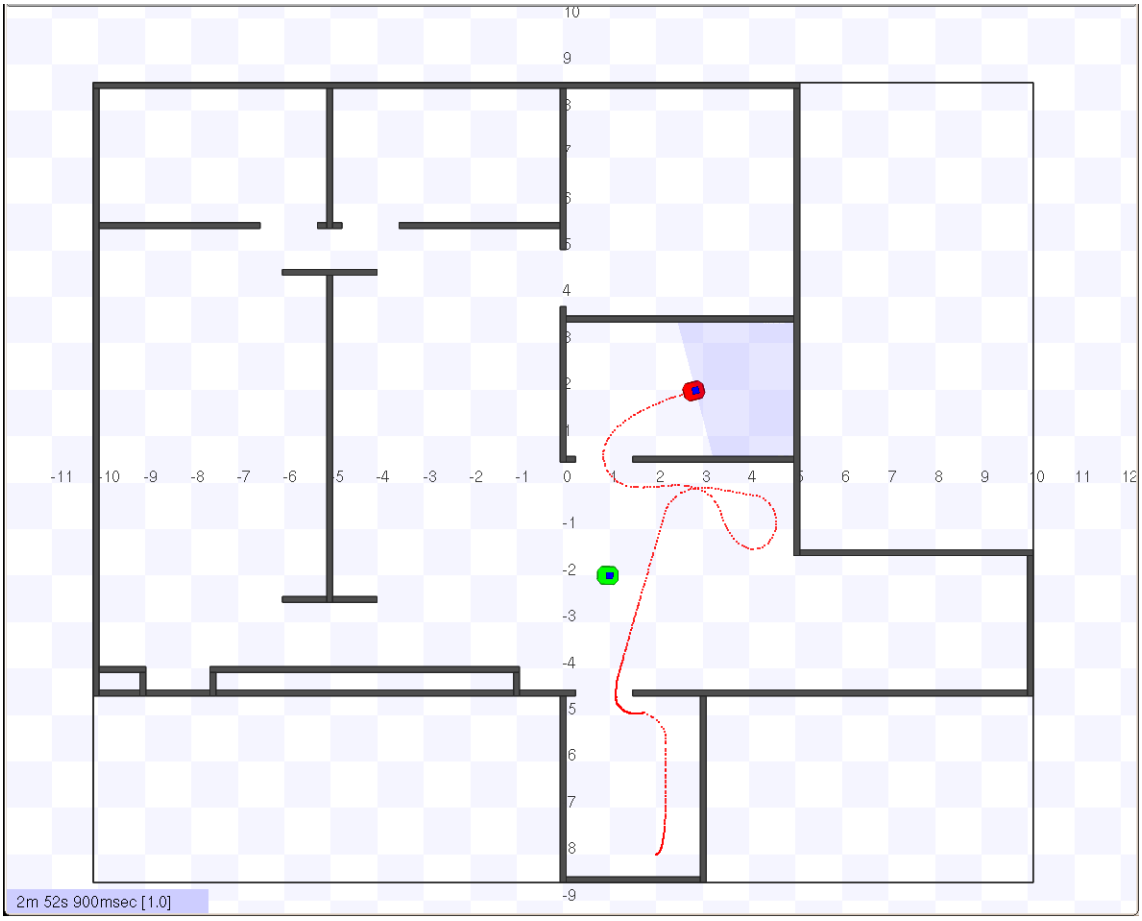
Algoritmo de evitación de obstáculos VFH+ (Vector Field Histogram)

Al realizar estas pruebas hemos hecho los mismos recorridos que en el apartado anterior para poder comparar los dos algoritmos.

Recorrido 1:



Recorrido 2:



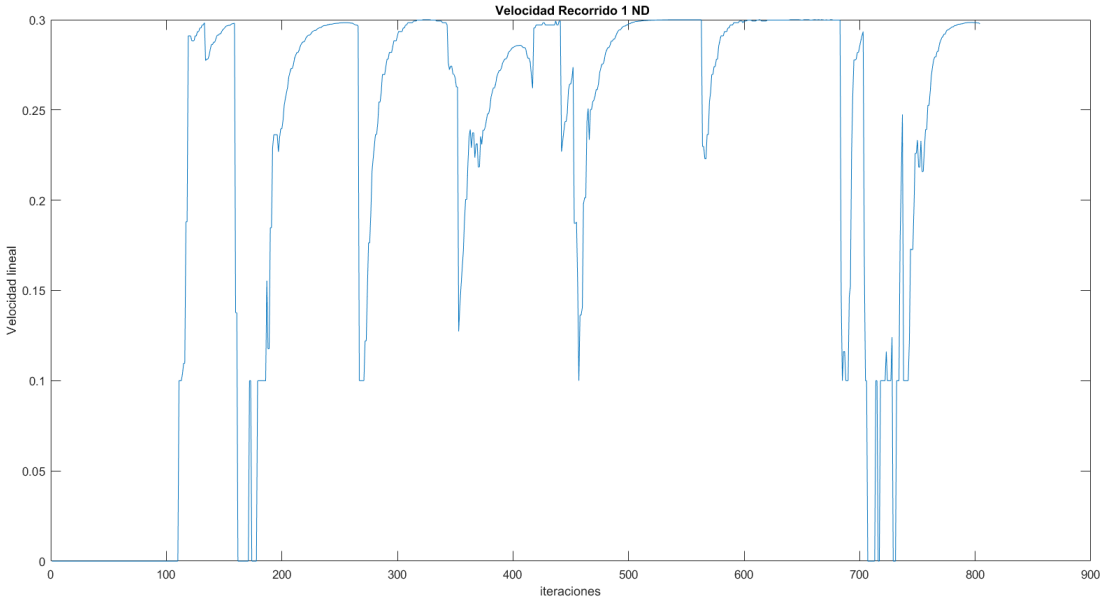

```
driver
(
  name "vfh"
  requires ["position2d:0" "laser:0"]
  provides ["position2d:1"]
  safety_dist_0ms 0.15
  safety_dist_1ms 0.25
  max_speed 0.20
  distance_epsilon 0.15
  angle_epsilon 10
)
```

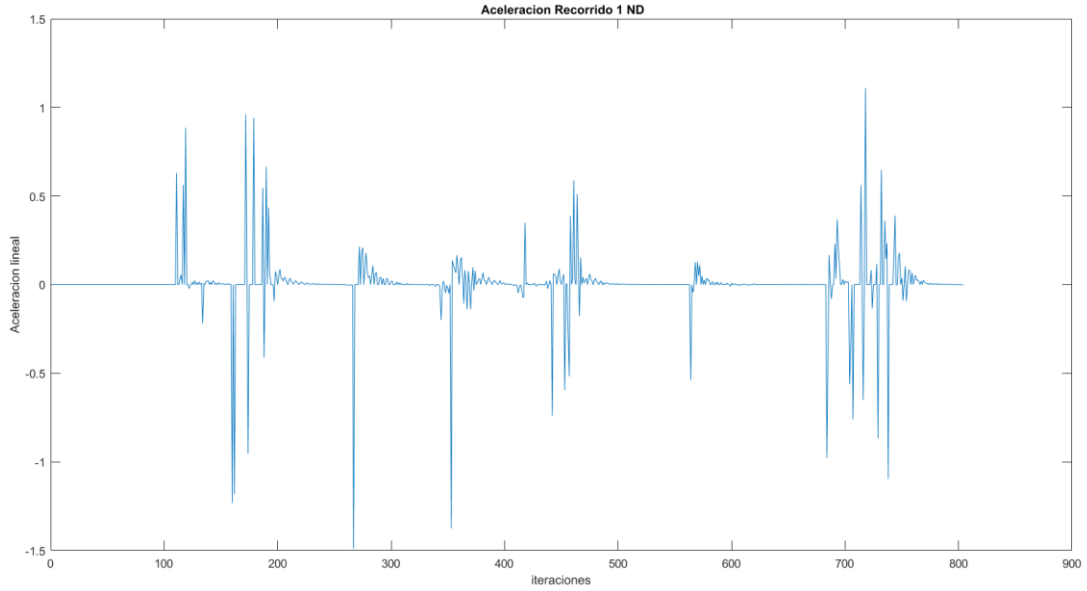
Con los valores por defecto apreciamos que en ciertas zonas del mapa el robot está a punto de entrar en un mínimo local sin llegar a entrar por lo que realizamos modificaciones en los parámetros del algoritmo. Con las modificaciones conseguimos agilizar el movimiento del robot aunque en algunas zonas como se puede apreciar en la segunda curva del recorrido 2 le sigue costando y la línea que dibuja el camino del robot es mucho más intensa por la acumulación de puntos.

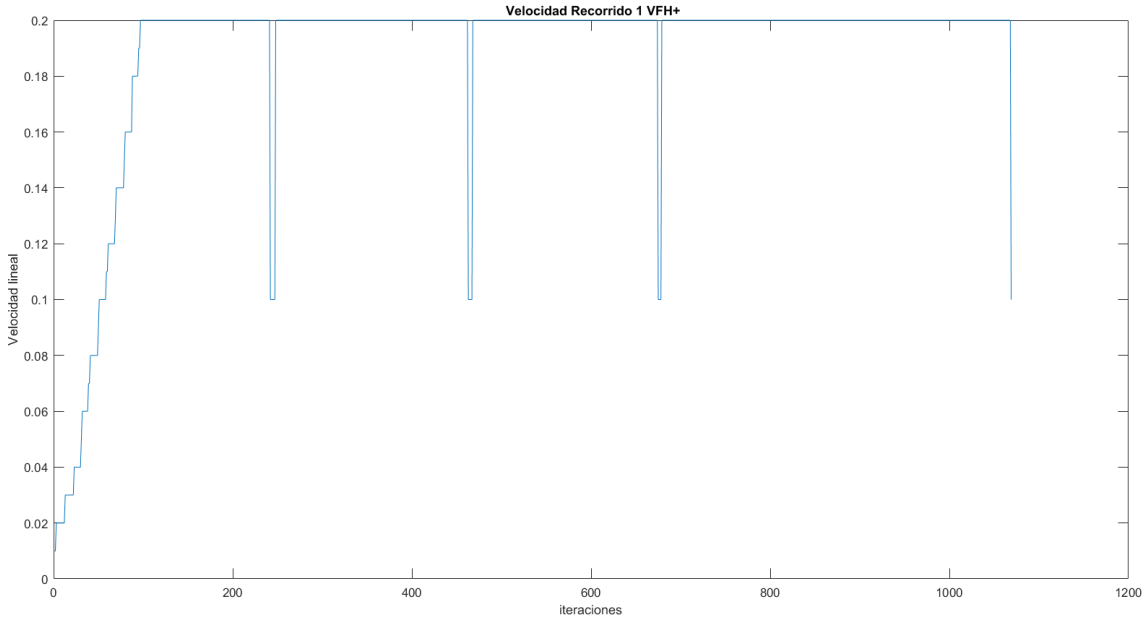
Comparativa de algoritmos de evitación de obstáculos

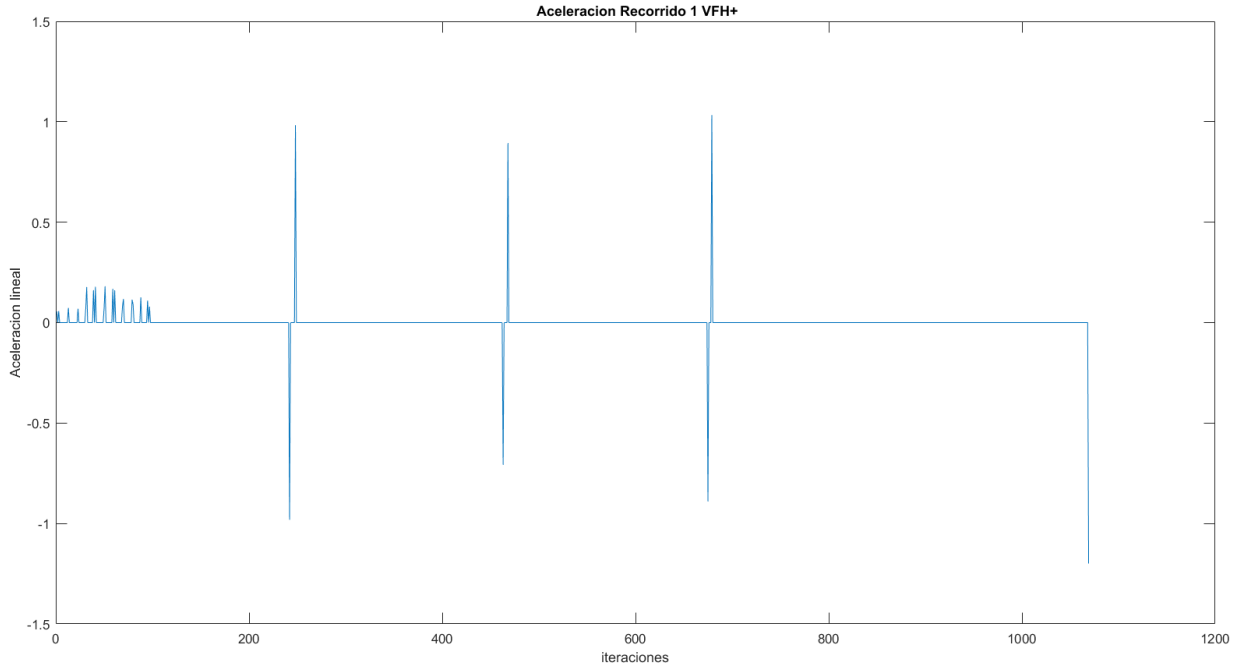
Recorrido 1:

	D. seguridad mínima	Tiempo al objetivo	Velocidad media	Velocidad máxima	Aceleración media	Aceleración máxima
ND	0.1	100.23	0.218	0.3	0.0036	1.110
VFH+	0.15	97.84	0.188	0.2	0.0012	1.033



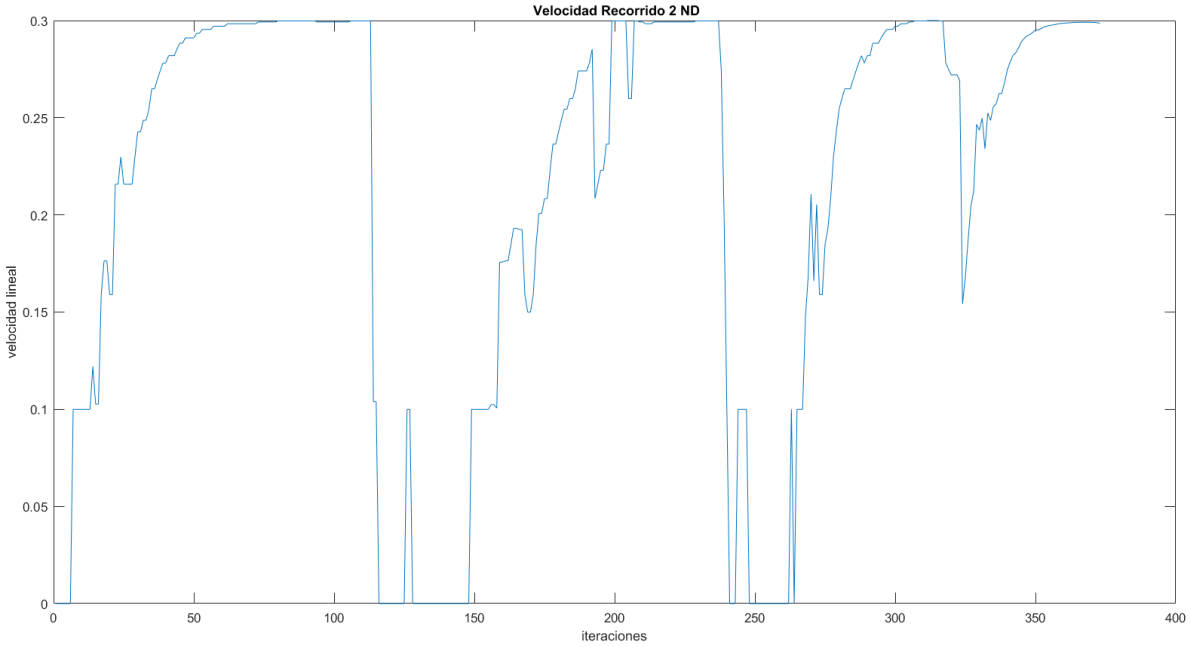


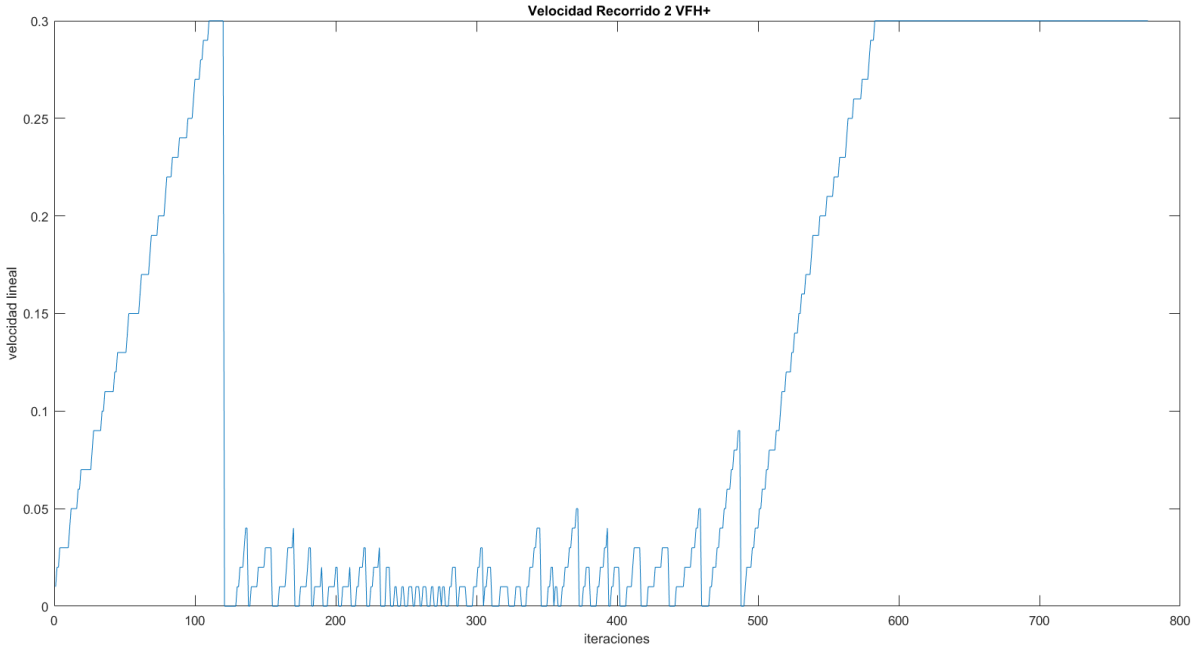




Recorrido 2:

	D. seguridad mínima	Tiempo al objetivo	Velocidad media	Velocidad máxima	Aceleración media	Aceleración máxima
ND	0.1	52.89	0.214	0.3		
VFH+	0.15	149.59	0.126	0.3		


Velocidad Recorrido 2 ND
velocidad lineal
iteraciones
0
0.05
0.1
0.15
0.2
0.25
0.3
0
50
100
150
200
250
300
350
400


Velocidad Recorrido 2 VFH+
velocidad lineal
iteraciones
0
0.05
0.1
0.15
0.2
0.25
0.3
0
100
200
300
400
500
600
700
800