Sistemas de Control para Robots
PRÁCTICA 2

LOCALIZACIÓN MEDIANTE FILTROS DEPARTÍCULAS

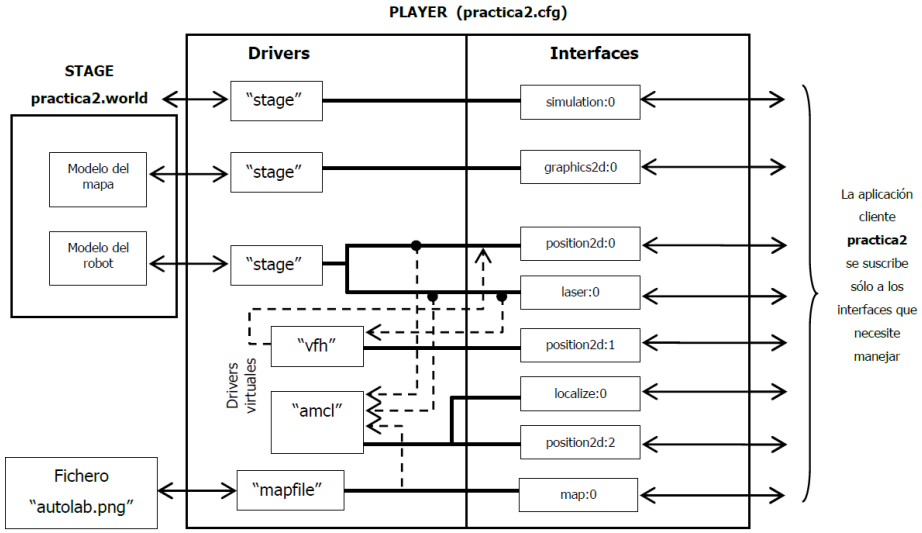
PLAYERNAV

Pedro Barquín Ayuso
Miguel Ballesteros García

Índice

Introducción

Para la realización de esta práctica hemos seguidos las fases de desarrollo indicadas en el manual modificando el Archivo.world primero y el Archivo.cfg con los drivers que debían añadirse y las modificaciones pertinentes en los que ya existían, acorde al esquema de conexión.



En cuanto a los archivos de configuración el resultado es el siguiente:

Archivo.cfg

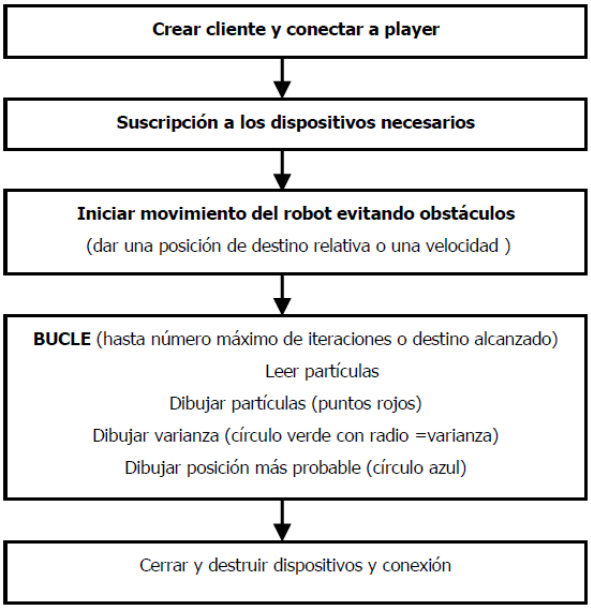
```
driver
(
  name "stage"
  provides ["simulation:0"]
  plugin "stageplugin"
worldfile "ejemplo1.world"
)
driver
(
  name "stage"
  provides ["graphics2d:0"]
  model "mimapa"
)
driver
(
  name "stage"
  provides ["position2d:0" "laser:0" ]
model "robot1"
)
driver
(
  name "vfh"
  provides [ "position2d:1" ]
  requires [ "position2d:0" "laser:0" ]
)
driver
(
  name "mapfile"
  provides ["map:0"]
  filename "autolab.png"
  resolution 0.025
)
driver
(
  name "amcl"
  provides ["position2d:2" "localize:0"]
```

```
requires ["odometry:::position2d:0" "laser:::map:0" "laser:0" ]
init_pose [0.0 0.0 0.0] #Estimacion inicial de la posicion del robot en [m m rad]
init_pose_var [1 1 6.28]#Varianzas para caracterizar la incertidumbre inicial de la estimacion
anterior+
pf_min_samples 100              #minimo numero de particulas a mantener en el filtro
pf_max_samples 10000            #maximo numero de particulas a manterner en el filtro
)
```

Archivo.world

```
include "pioneer.inc"
include "map.inc"
include "sick.inc"
window
(
size [ 809 689 ]     # Tamaño de la ventana de visualización de Stage en píxeles
  center [0.0 0.0]     # Coordenadas en metros (mundo real) del centro de la ventana
scale 40             # Relación entre coordenadas en píxeles y en metros.
show_grid 1
show_data 1
)
floorplan
(
  name "mimapa"
  bitmap "autolab.png"
size [20.225 17.225 1]  # Tamaño en metros del bitmap. Lo carga centrado en la ventana de Stage
)
pioneer2dx
(
  name "robot1"
  color "red"
localization "odom"             #tipo obtencionposicion
  pose [0.0 0.0 0.0 0.0]        #posicion del robot en el mapa
sicklaser()                     #añade el laser
)
```

Una vez hecho esto, procedemos a modificar el Archivo.c siguiendo el pseudocódigo propuesto en el enunciado, haciendo especial hincapié en la etapa del bucle ya que el resto son similares a las practicas anteriores.



Primero creamos el cliente y suscribimos los drivers necesarios tal y como se ha venido realizando en las prácticas anteriores, salvo que en este caso hay que prestar mas atención a la hora de realizar las suscripciones para no equivocarnos al haber varios drivers iguales.
Después iniciamos el movimiento del robot apoyándonos en el código de la práctica anterior salvo ligeras modificaciones para realizar las pruebas.
Creamos el bucle con las funcionalidades especificadas en la imagen anterior, quedando tal como se puede ver en la porción de código extraída del Archivo.c adjunto.

```
//leer particulars
playerc_localize_get_particles(localize);

//declaración dinámica para los contenedores en función del número de partículas
printf("numero de particulas %d\n",localize->num_particles);
puntos=(player_point_2d_t *)malloc(sizeof(player_point_2d_t)*(localize->num_particles));

//almacenamos y representamos las partículas
playerc_graphics2d_setcolor (graficos, colorR);
for (dd=0 ;dd<localize->num_particles-1 ; dd++){
        puntos[dd].px=localize->particles[dd].pose[0];
        puntos[dd].py=localize->particles[dd].pose[1];
}

//limpiamos los puntos anteriores
playerc_graphics2d_clear(graficos);

//dibujamos las particulas
playerc_graphics2d_draw_points (graficos, puntos, localize->num_particles);

/obtenemos información para dibujar el circulo
variacion=fabs(localize->variance);
printf("valor de la variacion %f\n",localize->variance);
circuloX=localize->mean[0];
circuloY=localize->mean[1];

//dibujamos el Circulo varianza
DibujaCirculo(circuloX, circuloY, variacion*10, colorG, graficos);

//posición más probable
DibujaCirculo(circuloX, circuloY, 0.3, colorB, graficos);
```

La función DibujarCirculo suministrada por el profesor se hamodificadopara completar mayor superficie (aumentamos el número de puntos dados a 16 para cerrar un poco más la circunferencia dibujada).

```
void    DibujaCirculo(double    xc,    double    yc,    double    radio,    player_color_t    color,
playerc_graphics2d_t *graficos){
        doublex[16], senos[16], cosenos[16], inc;
        inty,i;
        player_point_2d_t puntos[16];

        inc=2*3.14/16;
        x[0]=0.0;
        for(i=1;i<16;i++){
                x[i]=x[i-1]+inc;
        }
        for(i=0;i<16;i++){
                senos[i]=sin(x[i]);
                cosenos[i]=cos(x[i]);
        }
        for(i=0;i<16;i++){
                puntos[i].px=xc+radio*cosenos[i];
                puntos[i].py=yc+radio*senos[i];
        }
        playerc_graphics2d_setcolor (graficos,color);
        playerc_graphics2d_draw_polyline (graficos,puntos,16);
}
```

Problemas encontrados:

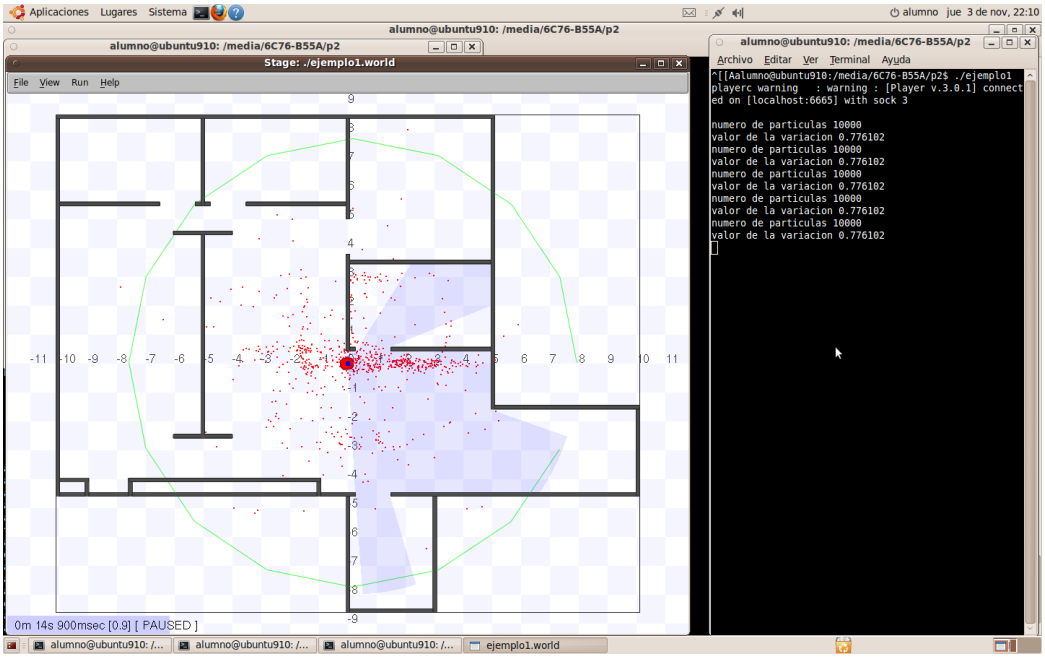
- Al comprobar si el código funcionaba correctamente nos encontramos primero con que el robot se estrellaba con la pared situada en la dirección del movimiento. Tras repasar el Archivo.cfg vimos que el driver amcl no estaba funcionando como debería porque estábamos usando un position2d que no era el suyo.

- Una vez solucionado este problema y comprobando que el robot sorteaba las paredes, tuvimos problemas al reservar memoria para poder representar los puntos de las partículas, ya que al ser un algoritmo adaptativo el cual va cambiando según la situación la reserva hay que hacerla de forma dinámica. Esto lo solucionamos reservando memoria dinámicamente antes del bucle.

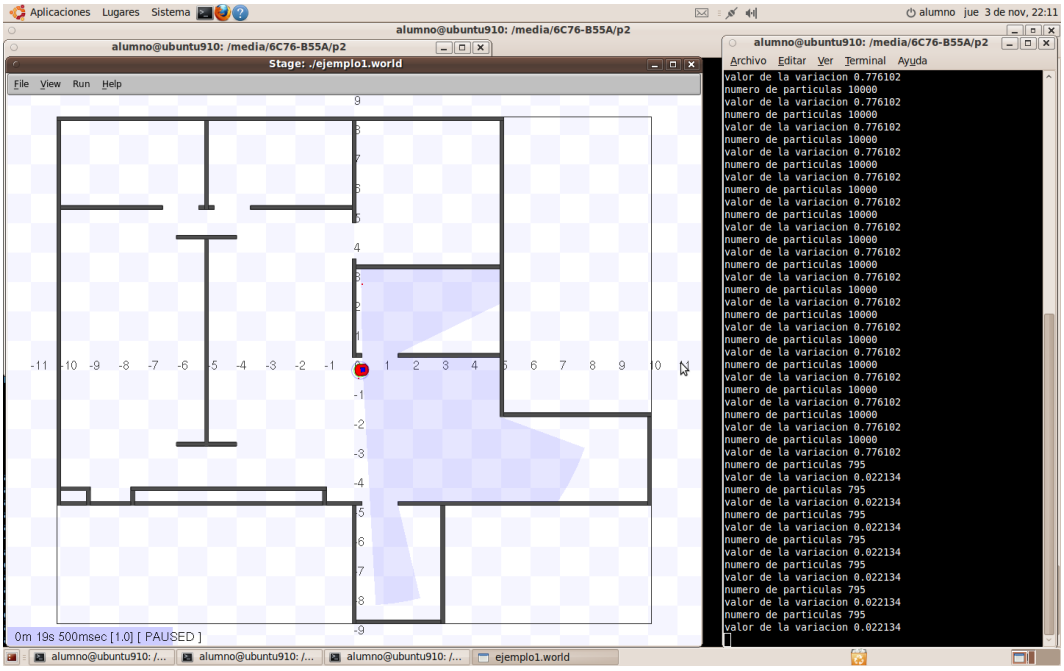
Pruebas (AMCL+VHL)

-**Compruebe el funcionamiento del sistema de localización cuando la hipótesis inicial del filtro se coloca aproximadamente en la posición real del robot.**

En esta prueba se lanzan las partículas en las proximidades de la ubicación real del robot, por lo que en primera instanciaaparecen todas las partículasal rededor y el radio de ubicación posible es grande ya que aún no ha realizado ningún cálculo para para obtener la ubicación real.



En la siguiente captura se aprecia que tras pocas iteraciones (casi instantáneo) a aproximado casi completamente la posición del robot y además coincide con la real, esto es debido a que al lanzar las partículas concentradas y próximas al robot en pocos pasos realiza la ubicación.

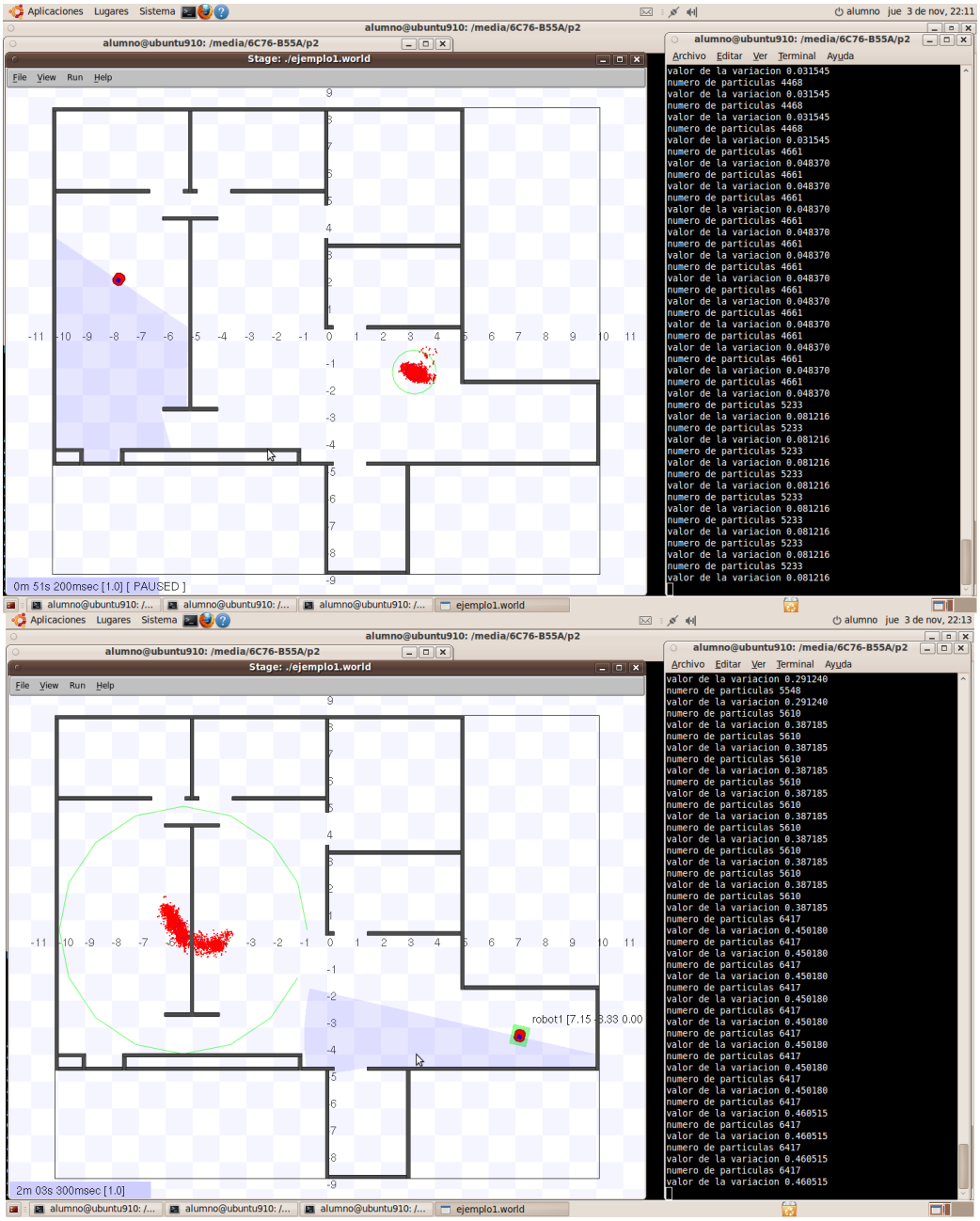


Como segunda prueba realizamos simulacro de un “secuestro” (kidnapping) para ver cómo se comporta, para ello una vez está el robot localizado correctamente lo desplazamos de posición, cuando ocurre esto se pueden dar dos situaciones:

-Primer caso es que si la sala es similar a la ubicación donde estaba antes de desplazarlo ocurre que continúa pensando que está en el mismo sito y casi no varía, hasta que empieza a ver cosas distintas.

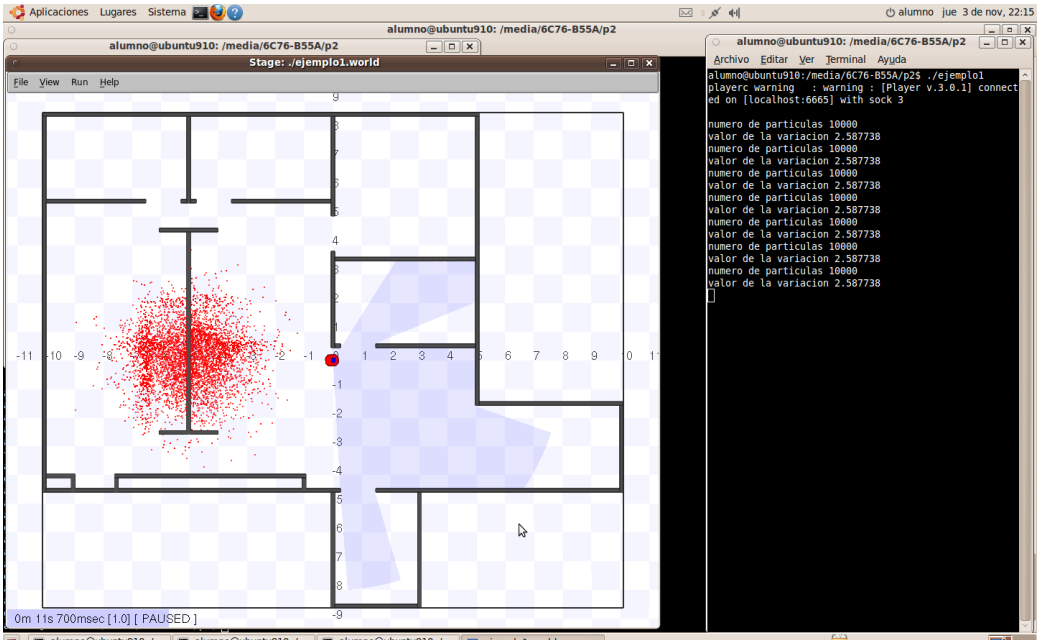
-Segundo caso ocurre cuando el entorno es distinto en ese caso empieza a aumentar la incertidumbre tal como se ve en la segunda imagen

En ambos casos la posibilidad de recuperación es prácticamente nula a no ser que el secuestro sea muy próximoa la ubicación (no se consideraría secuestro) en ese caso al aumentar el número de partículas es posible que recupere.

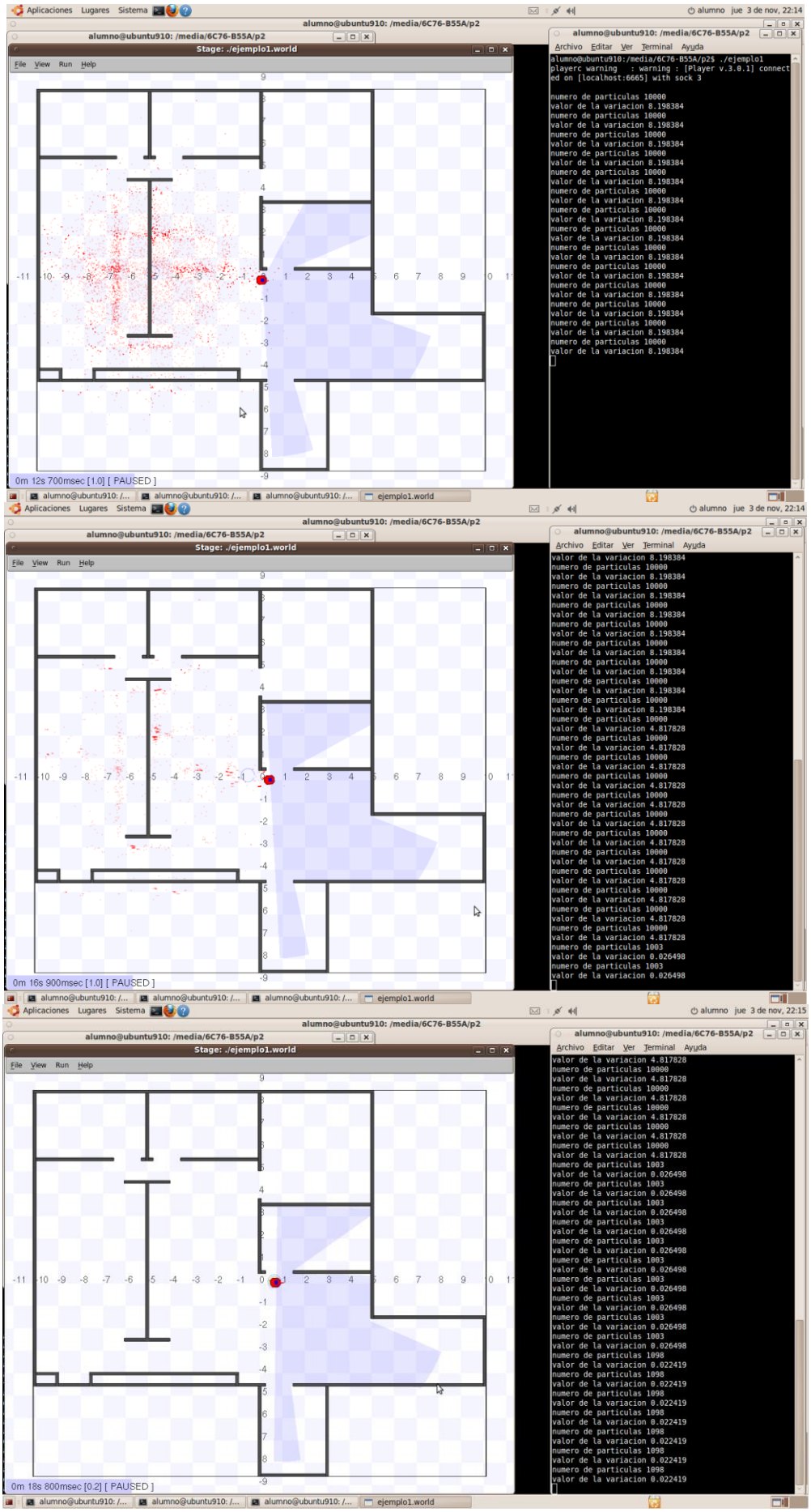


-Compruebe el funcionamiento del sistema de localización cuando la hipótesis inicial se coloca en una posición errónea.

Primero caso, comprobamos una posición errónea y las partículas centralizadas en el origen. Como se puede apreciar por las capturas al no tocar las partículas con la ubicación real del robot no consigue localizarlo, pudiendo dar lugar incluso a falsas localizaciones si el entorno es similar.



Segundo caso ahora comparamos con una posición errónea, pero con las partículas más dispersas (imagen 1), en este caso las partículas llegan hasta la posición del robot por lo que tras varias iteraciones empieza a encontrar posiciones posibles (imagen 2) y tras unas cuantas iteraciones más logra aproximas a la posición real (imagen 3).

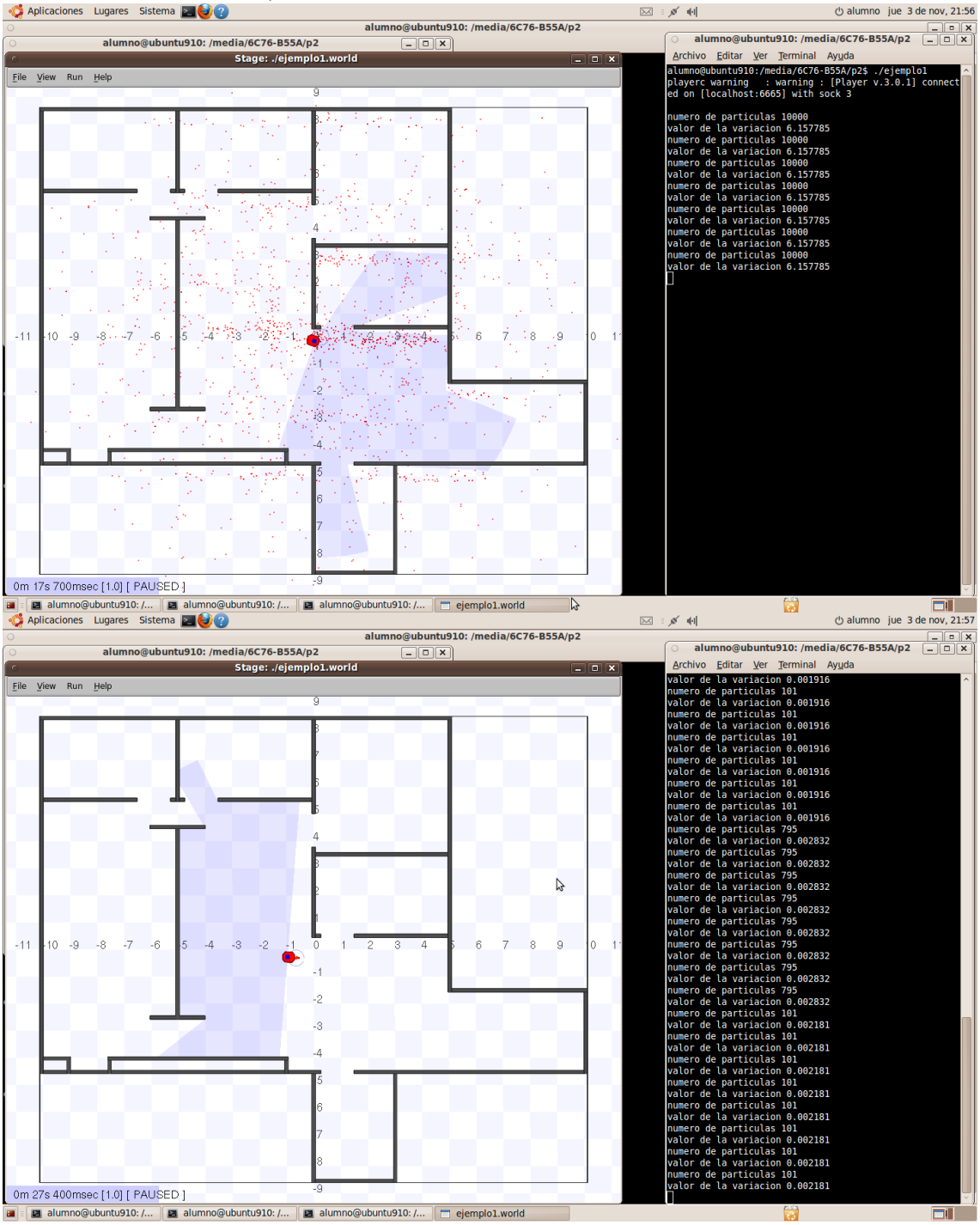

Aplicaciones Lugares Sistema
alumno jue 3 de nov, 22:14
alumno@ubuntu910: /media/6C76-B55A/p2
Stage: ./ejemplo1.world
File View Run Help
0m 12s 700msec [1.0] [PAUSED]
alumno@ubuntu910: /media/6C76-B55A/p2
Archivo Editar Ver Terminal Ayuda
alumno@ubuntu910:/media/6C76-B55A/p2$ ./ejemplo1
playerc warning : warning : [Player v.3.0.1] connected on [localhost:6665] with sock 3
numero de particulas 10000
valor de la variacion 8.198384
ejemplo1.world
0m 16s 900msec [1.0] [PAUSED]
valor de la variacion 4.817828
numero de particulas 1003
valor de la variacion 0.026498
0m 18s 800msec [0.2] [PAUSED]
alumno jue 3 de nov, 22:15
numero de particulas 1098
valor de la variacion 0.022419

Como podemos ver cuando le damos una posición buena en seguida se recupera y coge la posición en la que se encuentra el robot. Sin embargo cuando el damos una posición errónea tarda más en localizarse y dependiendo del número y dispersión de las partículas puede que no se encuentre nunca.

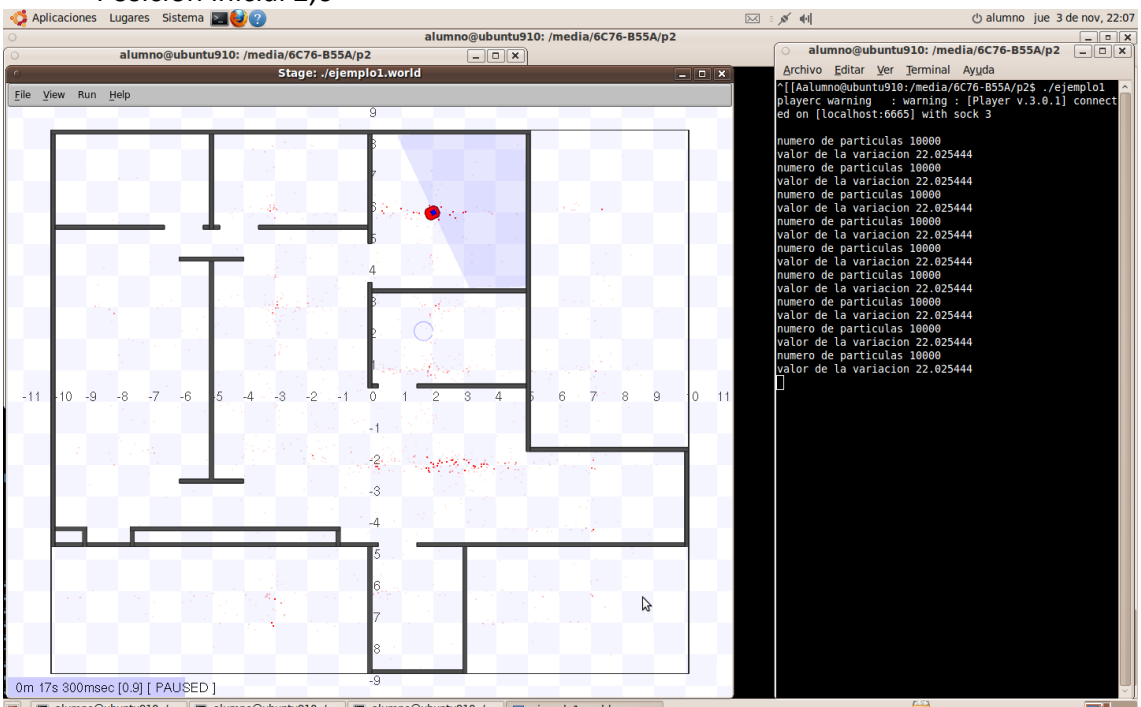
-Compruebe la capacidad del sistema para resolver el problema de la localización global. Para ello, inicialice el filtro siempre con posición inicial [0 0 0] y una varianza elevada que cubra todo el mapa de entorno. Realice múltiples pruebas con el robot endiferentes posiciones iniciales para comprobar la convergencia del filtro.

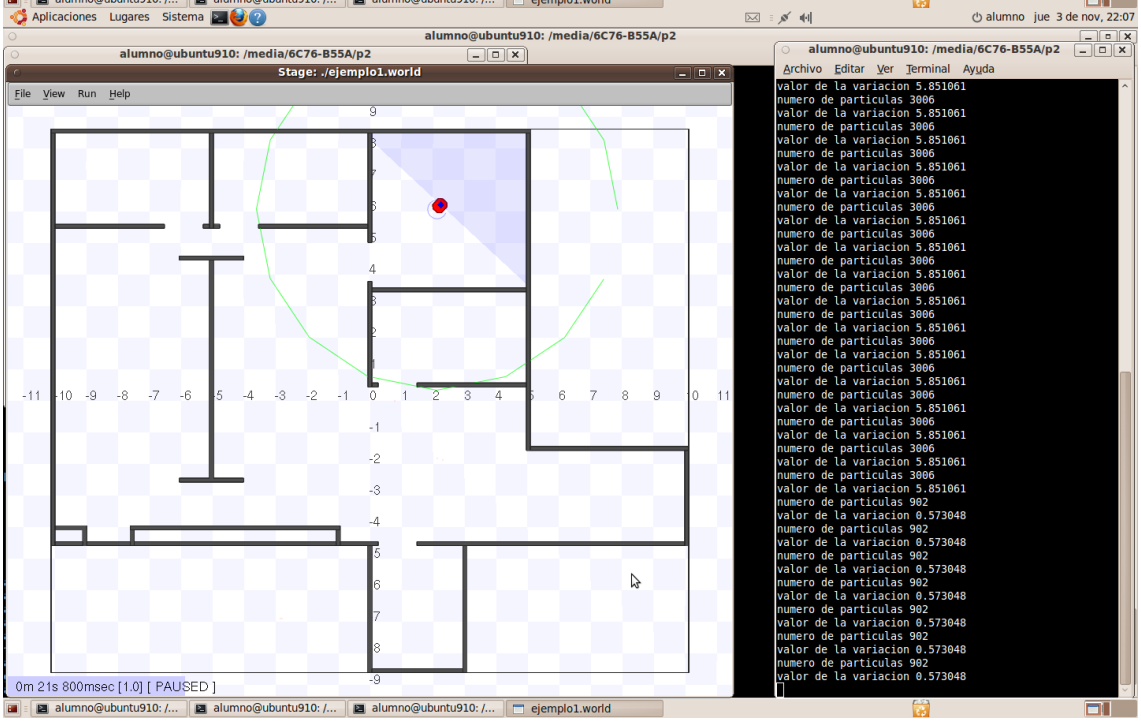
En este caso al dar mayor dispersión a las partículas es capaz de encontrarse independientemente del lugar de comienzo del robot (localización global), tal y como se puede aprecias en los casos que se enseñan a continuación, dependiendo de la posición y el entono tardara más o menos, pero al final siempre se ubica.

- Posición iniciar 0,0



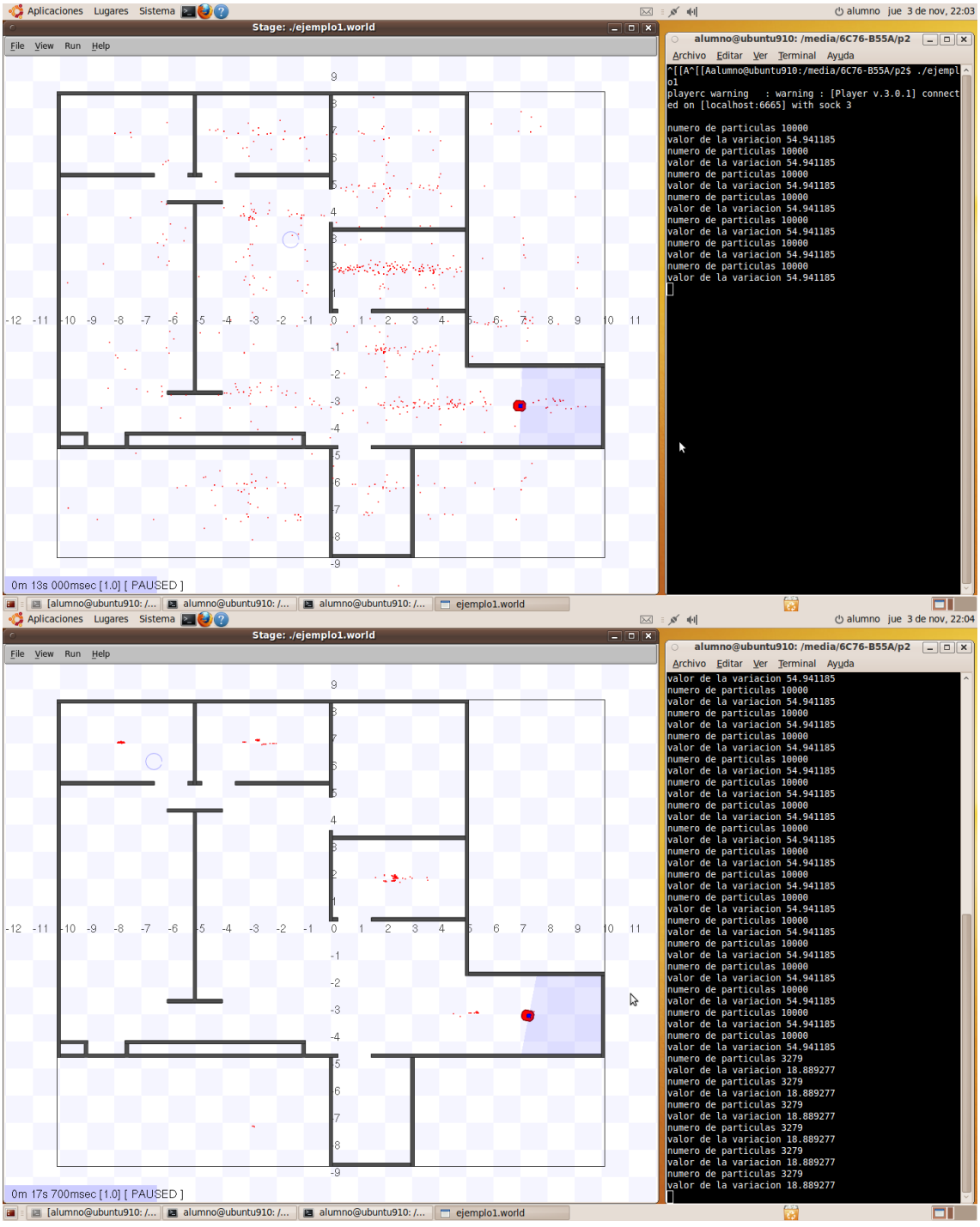
- Posición inicial 2,6

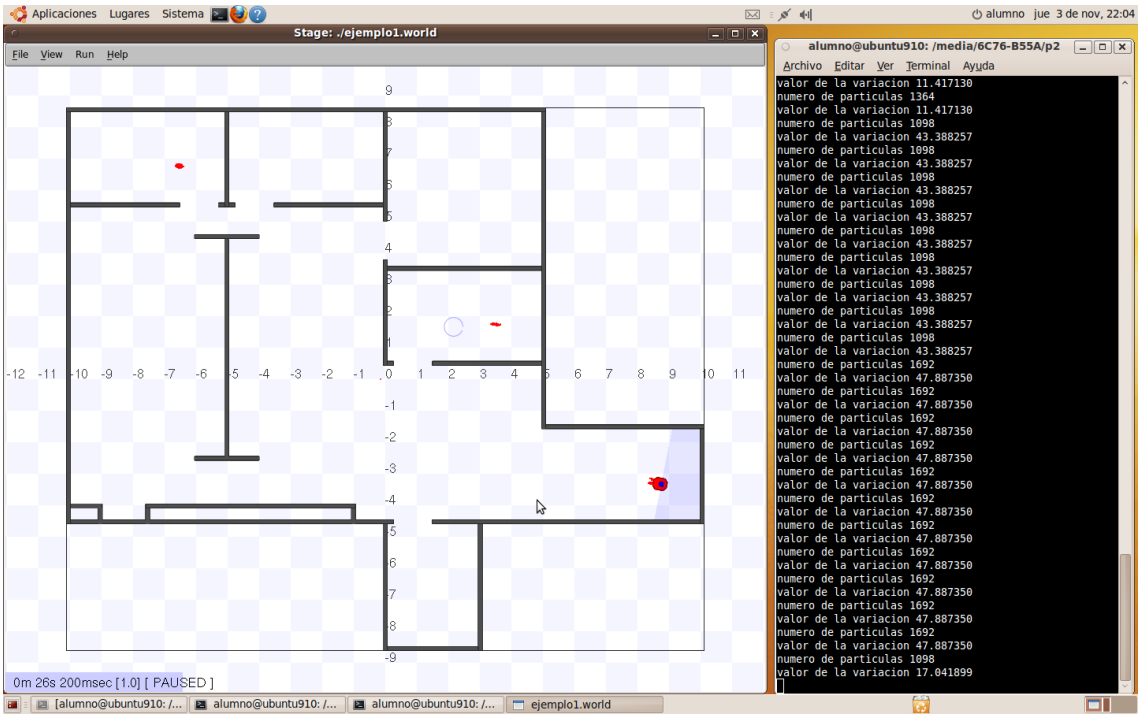




- Posición inicial 7,-3

Este caso es el más completo, ya que a medida que avanza se puede ver que empieza a descubrir la posición, pero tiene dudas, ya que hay varios puntos del mapa muy similares (imagen 2), por lo que empieza a centrar las partículas en ellos, hasta que finalmente a base de avanzar y obtener nuevas medida, empieza a descartarlos hasta quedarse con el correcto.




Aplicaciones Lugares Sistema
alumno jue 3 de nov, 22:04
Stage: ./ejemplo1.world
File View Run Help
0m 26s 200msec [1.0] [PAUSED]
alumno@ubuntu910: /media/6C76-B55A/p2
Archivo Editar Ver Terminal Ayuda
valor de la variacion 11.417130
numero de particulas 1364
valor de la variacion 11.417130
numero de particulas 1098
valor de la variacion 43.388257
numero de particulas 1098
valor de la variacion 43.388257
numero de particulas 1098
valor de la variacion 43.388257
numero de particulas 1098
valor de la variacion 43.388257
numero de particulas 1098
valor de la variacion 43.388257
numero de particulas 1098
valor de la variacion 43.388257
numero de particulas 1098
valor de la variacion 43.388257
numero de particulas 1098
valor de la variacion 43.388257
numero de particulas 1098
valor de la variacion 43.388257
numero de particulas 1692
valor de la variacion 47.887350
numero de particulas 1692
valor de la variacion 47.887350
numero de particulas 1692
valor de la variacion 47.887350
numero de particulas 1692
valor de la variacion 47.887350
numero de particulas 1692
valor de la variacion 47.887350
numero de particulas 1692
valor de la variacion 47.887350
numero de particulas 1692
valor de la variacion 47.887350
numero de particulas 1692
valor de la variacion 47.887350
numero de particulas 1692
valor de la variacion 47.887350
numero de particulas 1692
valor de la variacion 47.887350
numero de particulas 1098
valor de la variacion 17.041899
[alumno@ubuntu910: /...
alumno@ubuntu910: /...
alumno@ubuntu910: /...
ejemplo1.world

Pruebas (PLAYERNAV)

Para este apartado se ha intentado utilizar la aplicación playernav según indica el manual, pero no se ha logrado un resultado satisfactorio posiblemente por algún error de configuración, por ello no se pueden comparar los resultados con las pruebas anteriores.

Aun así se adjunta el proceso y los datos de las pruebas.

Para la utilización de la interface playernav hemos modificado tenido que modificar Archivo.cfg y Archivo.world con la información pertinente para poder utilizarlo ya que según el manual playernav necesita elementos del tipo planner y localize este último ha sido usado en la parte anterior pero el planner se ha añadido para esta.

Archivo.cfg

```
driver
(
  name "stage"
  provides ["simulation:0"]
  plugin "stageplugin"
worldfile "ejemplo1.world"
)
driver
(
  name "stage"
  provides ["position2d:0" "laser:0" ]
  model "robot0"
)
driver
(
  name "vfh"
  provides [ "position2d:1" "planner:0"]
  requires [ "position2d:0" "laser:0" ]
)
driver
(
  name "amcl"
  provides ["localize:0"]
requires ["odometry:::position2d:0" "laser:::map:0" "laser:0" ]
init_pose [0.0 0.0 0.0] #Estimación inicial de la posición del robot en [m m rad]
init_pose_var [2 2 6.28]#Varianzas para caracterizar la incertidumbre inicial de la V
pf_min_samples 100              #mínimo numero de partículas a mantener en el filtro
pf_max_samples 10000            #máximo numero de partículas a mantener en el filtro
)
driver
(
  name "mapfile"
  provides ["map:0"]
filename "autolab.png"
resolution 0.025
)
```

Archivo.world

```
include "pioneer.inc"
include "map.inc"
include "sick.inc"

window
(
size [ 809 689 ]     # Tamaño de la ventana de visualización de Stage en píxeles
  center [0.0 0.0]     # Coordenadas en metros (mundo real) del centro de la ventana
scale 40             # Relación entre coordenadas en píxeles y en metros.
show_grid 1
show_data 1
)
floorplan
```

```
(
  name "mimapa"
  bitmap "autolab.png"
size [20.225 17.225 1]  # Tamaño en metros del bitmap. Lo carga centrado en la ventana de Stage
)
pioneer2dx
(
  name "robot0"
  color "red"
localization "odom"             #tipo obtenciónposición
  pose [0.0 0.0 0.0 0.0]        #posición del robot en el mapa
sicklaser()                     #añade el laser
)
```

Se han repetido estos archivos con distinto port para el uso de un segundo robot.

En cuanto a los comandos de ejecución para el programa son los siguientes:

```
Player -p 6665 Archivo1.cfg
Player -p 6666 Archivo2.cfg

Playernav localhost:6665 localhost:6666
```

El problema radica en que no aparece la ruta a seguir cuando fijamos la posición de destino para los robots pero sí que realizan el movimiento por lo que no sabemos a ciencia cierta si está correctamente configurado.