**PREGUNTA 1: MAPEADO CON POSICIONES CONOCIDAS (fichero)**

Subir el fichero .m para el mapeado del entorno con posiciones conocidas utilizando el simulador STDR (fichero "MappingWithKnownPoses\_Simulator.m")

*\*Insertar Fichero\**

**PREGUNTA 2: MAPEADO CON POSICIONES CONOCIDAS (resultados)**

Subir un fichero .zip con las dos imágenes de los mapas obtenidos en simulación:

Sin ruido de odometría (llamar "mapa\_sinruido.jpg")

Con ruido de odometría (llamar "mapa\_conruido.jpg")

*\*Insertar Fichero\**

**PREGUNTA 3: MAPEADO CON POSICIONES CONOCIDAS (comentarios)**

Comente brevemente las siguientes cuestiones: ¿Cómo se realiza este tipo de mapeado? Comente los resultados obtenidos en función de la presencia o no de ruidos en la odometría, y explique brevemente a qué se deben las diferencias.

**Respuesta:**

Podríamos decir que este mapeado ha sido realizado mediante el método de rejilla por ocupación, en el que el mapa, está dividido en una seria de celdas, las cuales tienen contadores, en el caso en el que un sensor del robot, en este caso el láser, atraviese una celda, el contador de esta disminuye, pero en caso de que choque, este incrementa. De tal forma que las celdas con un contador mayor son elementos que el robot no puede atravesar.

Esto lo podemos ver a medida que el robot va avanzando en el simulador, ya que podíamos observar cómo, a medida que el robot avanzaba, los lugares por los que pasaba el laser eran coloreados de un color claro indicando las zonas por las que puede pasar, y en un color más oscuro las que no puede atravesar.

En cuanto a los resultados, aquellos realizados sin ruido, nos devuelven un mapa fiel a la realidad sin ninguna modificación o alteración. Esto se debe a que se han simulado condiciones ideales en las que no existe ningún error de medida u odometría.

En lo que respecta al mapa con ruido. Este tienes grandes errores, dándonos un mapa totalmente distinto a la realidad en el que apenas se pueden distinguir las habitaciones y el pasillo del mapa real. De esta forma, hemos comprobado que, en condiciones normales, existen una gran cantidad de factores que por un minimo error que tengan, ya sean los sensores o la odometría, pueden generar grandes errores en nuestros mapas.

*\*TODO: Completar\**

**PREGUNTA 4: SLAM (fichero)**

Subir el fichero OnlineSLAM\_2022a.m

*\*Insertar Fichero\**

**PREGUNTA 5: SLAM (resultados)**

Subir un fichero .zip con las tres imágenes siguientes:

Mapa obtenido en simulación sin ruido de odometría (llamar "online\_sim\_sinruido.jpg")

Mapa obtenido en simulación con ruido de odometría (llamar "online\_sim\_conruido.jpg")

Mapa real del pasillo OL21 (llamar "mapa\_real.jpg")

*\*Insertar Fichero\**

**PREGUNTA 6: SLAM (comentarios)**

Comente brevemente las conclusiones obtenidas respecto al uso de técnicas de SLAM (comparándolo con el uso de técnicas de mapeado con posición conocida).

**Respuesta:**

Se ha observado que, como bien enuncia el nombre de la técnica, hacer uso del algoritmo SLAM ha permitido obtener una mejor localización del robot habiendo ejecutado el robot con baja velocidad. A pesar de que en algunos momentos de la ejecución el resultado parecía “enloquecer”, al aplicar técnicas intrínsecas del código como el cierre de lazos (hacer pasar al robot por el mismo punto una vez ha recorrido parte del mapa), se observa que la obtención del mapa final lo realiza de una forma más exacta, apenas viéndose afectado por posibles ruidos del entorno que afecten a los sensores, a diferencia de lo que se veía en los ejercicios anteriores. Aunque SLAM pueda requerir mayor coste computacional, gracias a la potencia de los ordenadores actuales con los que cuenta el alumnado, apenas se ha notado una bajada de rendimiento, y por tanto se opina que este método funciona mejor que el anterior.

**PREGUNTA 7: LOCALIZACIÓN CON AMCL (fichero)**

Subir en un archivo .zip los programas de localización para el simulador y para el robot real:

AMCL\_Localization\_2022a\_Simulador.m

AMCL\_Localization\_2022a\_RobotReal.m

*\*Insertar Fichero\**

**PREGUNTA 8: LOCALIZACIÓN CON AMCL (resultados)**

Subir en un fichero .zip los dos videos siguientes:

Localización en simulación (mapa simple\_rooms), con posición inicial desconocida.

Localización del robot real en el pasillo OL21, con posición inicial desconocida.

*\*Insertar Fichero (mandar capturas que tenemos y explicarle que nos lo ha permitido como dijimos en clase)\**

**PREGUNTA 9: LOCALIZACIÓN CON AMCL (comentarios)**

Describa brevemente el proceso de localización utilizando filtros de partículas. Comente los resultados obtenidos tanto en simulación, como con el robot real. ¿Qué características del entorno permiten que el robot se localice? ¿Cuándo consigue localizarse el robot y por qué? ¿Cuáles son las ventajas e inconvenientes de este tipo de localización? ¿Funciona siempre correctamente (converge siempre el filtro)? En caso contrario, ¿cómo se podría solucionar?

**Respuesta:**

El proceso de localización utilizando filtros de partículas sigue el siguiente proceso iterativo: primero, comprueba si conoce la posición inicial del robot. Si lo hace, entonces todos los puntos quedan centralizados en dicha posición. De lo contrario, las partículas quedan distribuidas aleatoriamente por el entorno del mapa. A medida que pasan las iteraciones, se selecciona una partícula del conjunto anterior (donde tienen más probabilidad de salir las que mayor peso tengan), y se propaga dicha partícula a su nueva posición. Se recalculan los pesos de cada una de ellas y el proceso se repite hasta que el robot queda localizado. Aunque no afecte mucho en este algoritmo, que el pasillo contenga asimetrías ayuda a localizar el robot, al igual que el hecho de que el robot avance en línea recta en mitad del mapa, permitiendo a los puntos que converjan de manera más rápida. De hecho, hasta que el robot no llega a lo que se podría decir que es el punto medio del mapa, el algoritmo no llega a converger, demostrando lo que se acaba de explicar. La ventaja principal de este tipo de filtro es que se consiga encontrar la posición del robot en prácticamente todas las iteraciones de dicho algoritmo. Sin embargo, cuanta mayor cantidad de partículas generadas, mayor carga computacional se generará sobre la máquina en la que se está trabajando. No siempre funciona bien el filtro, quedándose en ocasiones estancado sin conseguir llegar a la posición solución. Para poder arreglar esto, se podría reservar un número determinado de partículas que queden siempre aleatoriamente distribuidas por el mapa, de forma que “desatasque” el algoritmo.