Capítulo 4 – Localización Monte Carlo VBR.

4.1 Introducción.

En este capítulo se presenta una breve discusión acerca de las debilidades del método de localización Monte Carlo, así como las modificaciones propuestas a este método con el fin de acelerar el tiempo de convergencia, la exactitud del mismo y la reducción en el número de muestras utilizadas. Se profundiza en la implementación del método de localización Monte Carlo incluyendo las modificaciones propuestas, a este método se le ha denominado Monte Carlo VBR (*Vision Based Reseting*). La ingeniería de software necesaria para llevar a cabo la implementación, puede ser consultada en el apéndice B.

4.2 Discusión sobre Monte Carlo.

Una de las debilidades del método de localización Monte Carlo, es que no aprovecha las lecturas de los sensores para generar las muestras de la nueva población. Por lo tanto cuando estamos captando una *Baliza* con los sensores, es posible que aún existan muestras en el otro extremo del entorno donde es muy poco probable que el robot pueda captar dicha *Baliza*.

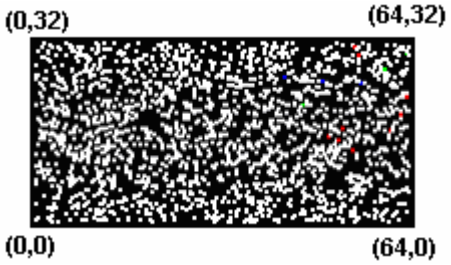

Figura 4.1 Evolución de la población utilizando MCL

Por ejemplo, en la figura 4.1 el robot se encuentra en la pose $P = (x, y, \theta) = (50, 16, 45)$ y capta una baliza “C”, pero aún después de el proceso de remuestreo (resamplig) se siguen conservando poses en las que no es posible captar la baliza “C”. En la figura anterior, cada punto representa una posible pose del robot. Los puntos de color blanco representan las poses muy poco probables, los de color verde a las poco probables, los de color azul a las poses probables y los de color rojo corresponden a las muy probables.

Otra debilidad del método radica en el número de iteraciones necesarias para estabilizar el proceso. Es decir, que el área que ocupa la población dentro del entorno sea lo suficientemente compacta como para que la diferencia en el error entre la posición real y la posición estimada no sea muy variable entre iteraciones subsecuentes. En estudios efectuados con el método de localización Monte Carlo se ha descubierto que son necesarias de 13 a 16 iteraciones para que el método se estabilice [18], pero depende de factores como el número de muestras utilizadas, así como el tamaño y forma del entorno.

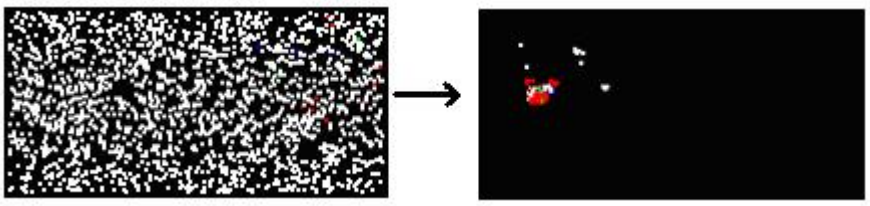

Figura 4.2 Estabilización del conjunto muestral.

En la figura 4.2 el cuadro de la izquierda muestra el inicio del proceso y el de la derecha muestra la evolución de la población cuando se considera que el proceso ya se estabilizó.

Para la localización global, el método Monte Carlo requiere una cantidad suficiente de muestras, de tal forma que la población de individuos no pierda representatividad. Si en la localización global se utilizan pocas muestras, la probabilidad de densidad local colapsa prematuramente sobre las pocas muestras más probables. La cantidad de muestras está directamente ligada con el tamaño y la forma del entorno.

El método es capaz de recuperarse después de que el robot ha sido raptado del entorno y tras un lapso de tiempo se ha vuelto a colocar en una posición totalmente diferente, el problema radica en que este proceso es muy lento. En las pruebas efectuadas por [22] se descubrió que son necesarias más de 35 iteraciones para que el robot pueda localizarse dentro de un radio de error de 10 centímetros.

4.3 Modificaciones Propuestas.

Observando las desventajas descritas anteriormente, se proponen las siguientes modificaciones del método Monte Carlo:

- Modificar el proceso de remuestreo (*resamplig*) de tal forma que cuando se capte por lo menos una baliza, se utilice Monte Carlo Dual para generar las muestras.
- Definir una caja envolvente de la población y generar las nuevas muestras a través del proceso de remuestreo (*resamplig*) propuesto, dentro de esta caja envolvente.

- Verificar en cada fase de remuestreo (*resamplig*) la calidad de la población. La calidad se medirá en base a los puntajes o pesos de los individuos de la población. Para esto, es necesario definir un umbral, el cual, definirá el peso promedio de la población para que se considere de buena calidad.
- Si la calidad de la población es mala, entonces reemplazar ésta con una nueva obtenida de manera aleatoria y distribuida uniformemente a través del entorno. Por el contrario, si la población es de buena calidad, hacer el remuestreo (*resamplig*) de la manera propuesta.

4.4 Algoritmo de Monte Carlo VBR.

Fase de predicción.

1. Para cada muestra $X_{t-1}^{(i)}$ en Φ_{t-1}
    2. Generar una muestra $\tilde{X}_t^{(i)}$ desde $p(X_t^{(i)} \mid X_{t-1}^{(j)}, a_{t-1})$
    3. Reemplazar $X_{t-1}^{(i)}$ con $\tilde{X}_t^{(i)}$

Fase de actualización.

1. [Paso opcional] reemplazar algunas muestras de Φ'_t con muestras aleatorias.
2. Para cada muestra $\tilde{X}_t^{(i)}$ en Φ'_t
    3. Actualizar el peso de la muestra con la probabilidad de la lectura sensorial $\omega_t^{(i)} = p(o_t \mid X_t^{(i)})$
4. Para cada muestra $X_t^{(i)}$ en Φ_t

5. Calcular y almacenar el peso acumulado de todas las muestras debajo de la muestra actual $(cw(X_t^{(i)}))$

6. Calcular el peso total de todas la muestras (tw)

7. Para cada muestra $\tilde{X}_{t+i}^{(i)}$ deseada en Φ_{t+1}

    8. Generar un número aleatorio (r) entre 0 y (tw)

    9. Encontrar la muestra con el máximo $(cw(X_t^{(i)})) < r$

    10. Agregar la muestra encontrada a Φ_{t+1}

11. Si el área que ocupa $\Phi_{t+1} > Ua$ y $AvgProb > UmbralCalidad$ y se capta por lo menos una baliza en la lectura sensorial.

    12. Para cada muestra $\tilde{X}_{t+1}^{(i)}$ en Φ_{t+1}

        13. Si su peso $\omega_{t+1}^{(i)} < ProbabilidadDeseada$

            14. Obtener $\hat{X}_{t+1}^{(i)}$ a partir de $p(o_t \mid X_t^{(i)})$ dentro del área de Φ_{t+1}

            15. Remplazar $\tilde{X}_{t+1}^{(i)}$ con $\hat{X}_{t+1}^{(i)}$

16. De lo contrario

    17. Si $AvgProb < UmbralCalidad$

        18. Para cada muestra $\tilde{X}_{t+1}^{(i)}$ en Φ_{t+1}

            19. Obtener $\hat{X}_{t+1}^{(i)}$ de manera aleatoria con una distribución uniforme.

            20. Reemplazar $\tilde{X}_{t+1}^{(i)}$ con $\hat{X}_{t+1}^{(i)}$

El parámetro *Ua* se refiere al umbral del área, con éste se decide que tan pequeña debe ser el área que ocupa la población para dejar de reemplazar muestras con baja probabilidad con otras obtenidas a partir de Monte Carlo Dual. *AvgProb* se refiere al promedio de los pesos de la población .Con *UmbralCalidad* se decide cual deberá ser por lo menos el peso promedio de la población para que siga siendo representativa. Con *ProbabilidadDeseada* se decide el peso mínimo que deberá tener la muestra para no ser reemplazada.

4.5 Entorno de simulación.

En esta sección se presentan los detalles de los elementos del entorno de simulación utilizado para la implementación del método de localización Monte Carlo VBR.

A continuación se presentan los distintos componentes del sistema.

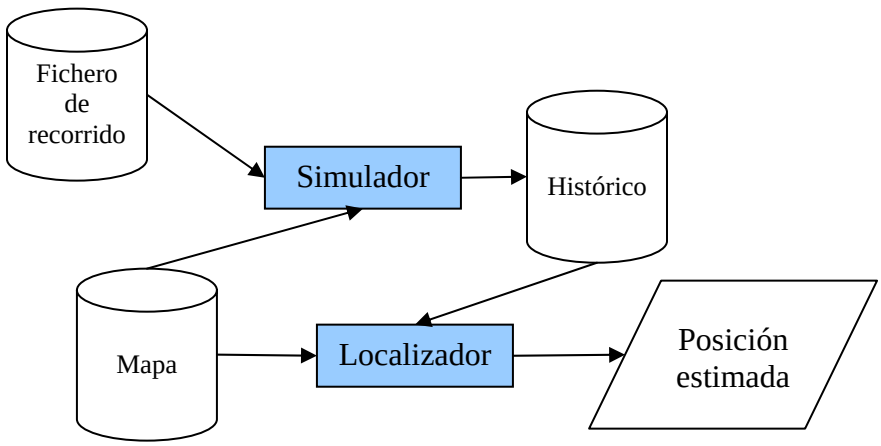

Figura 4.3 Diagrama de Bloques del sistema.

4.5.1 Fichero de recorrido.

El *Fichero de recorrido* es el archivo de entrada al sistema, contiene la posición inicial y las acciones que el robot deberá ejecutar. En la figura 4.4 se muestra un ejemplo de este archivo.

Cada línea del *Fichero de recorrido* inicia con un identificador. **P** es el identificador de la posición inicial del robot, la preceden tres números (x, y, θ) los cuales son las coordenadas de la posición inicial y rotación del robot. **A** índica la acción que el robot deberá ejecutar, la preceden tres números (x, y, θ) los cuales indican la traslación en x, y y la rotación en θ a partir de la posición actual del robot.

```
P 50    16    135
A 0     0     -90
A 0     0     -45
A -10   0     0
A 0     -8    225
A -11   0     -45
A 0     8     0
A 0     0     -45
```

Figura 4.4 Archivo de entrada al sistema.

4.5.2 Simulador.

Es el encargado de leer el archivo en el cual se encuentra la sucesión de acciones que el robot tendrá que ejecutar. Genera las imágenes que se ven desde cada posición después de que se ejecutó la acción correspondiente y las almacena en el histórico. Proporciona la información necesaria al algoritmo localizador en cada iteración.

4.5.3 Histórico.

El *Histórico* contiene la misma información que el archivo de entrada, pero se le han anexado las imágenes unidimensionales que simulan las capturadas por la cámara desde las posiciones recorridas. A continuación se muestra el Histórico resultante del archivo de entrada de la figura 4.4

Acción	Posición	Imagen
----------	50, 16, 135	B(34)
0,0,-90	50, 16, 45	C(33)
0, 0, -45	50, 16, 0	E(32) E(40) E(47)
0, -8, 225	40, 8, 225	G(40)
-10, 0, 45	30, 8, 270	G(15)
-11, 0, -45	19, 8 , 225	F(79)
0, 8, 0	19, 16, 225	F(48)
0, 0, -45	19, 16, 180	D(45) D(39) D(34)

Tabla 4.1 Ejemplo de Histórico.

4.5.4 Localizador.

Recibe como parámetros de entrada el Mapa y el histórico, para cada iteración ejecuta el método de localización Monte Carlo VBR y presenta en pantalla la posición estimada por el método.

4.5.5 Mapa.

El mapa en el cual el robot se localiza es una versión reducida del campo de juego de la RoboCup. El campo de juego oficial de la RoboCup cuenta con ocho posibles balizas, una en cada esquina del campo de juego, una al final de cada uno de los extremos de la línea central y las dos porterías. Las seis balizas que rodean el campo de juego son cilindros con dos colores, cada uno de estos cilindros tiene una configuración de color única. Cada portería tiene su propio color. Las dimensiones del campo de juego son las siguientes: 280 centímetros de largo, 180 centímetros de ancho. Las porterías se encuentran colocadas en el centro de los extremos del campo. Las dimensiones de las porterías son: 60 centímetros de largo y 30 centímetros de alto [23].

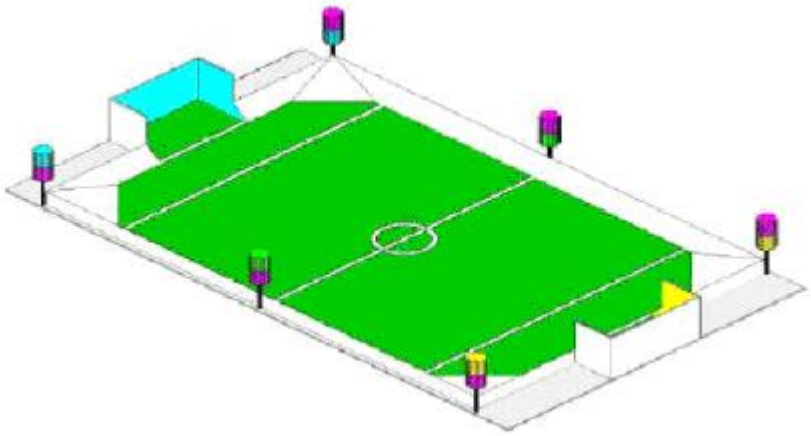

Figura 4.5 Campo de juego de la RoboCup.

La versión reducida del campo de juego utilizado para la implementación del método tiene las siguientes dimensiones: 64 centímetros de largo y 32 de centímetros de ancho. Se cuenta con las 8 balizas para que el robot lleve a cabo la tarea de localización.

En la figura 4.6 se muestra una representación del campo de juego utilizado. Esta versión reducida también cuenta con las ocho balizas del campo oficial de la RoboCup. Cada una de las balizas está representada de forma única por las letras A, B, C, D, E, F y G. Donde las balizas E y F corresponden a las porterías. Este es el mapa utilizado por los algoritmos de simulación y localización.

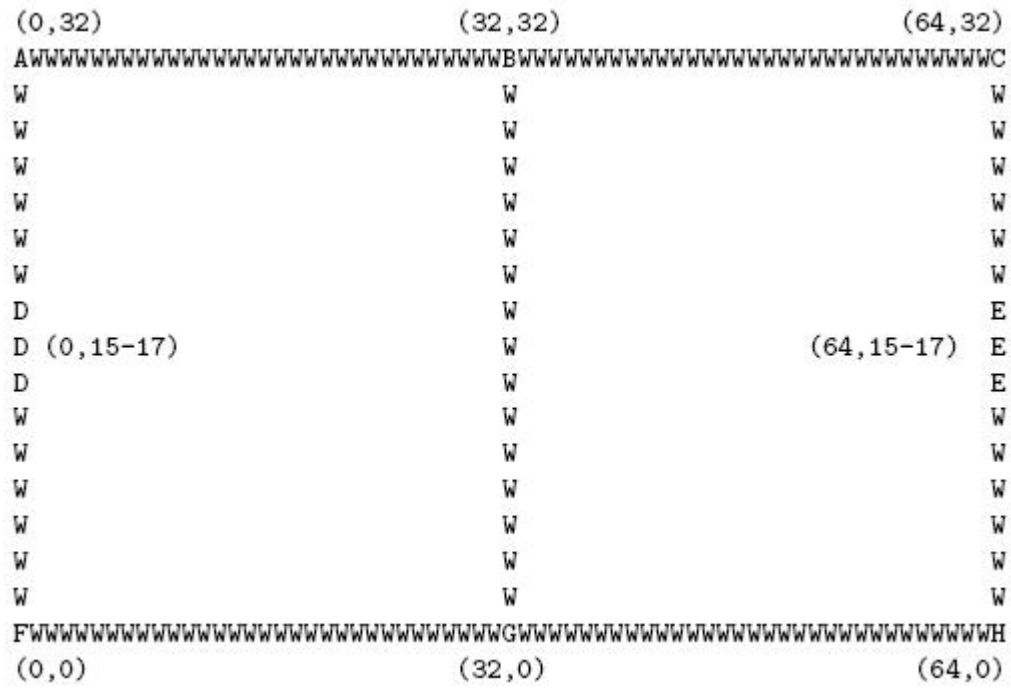

Figura 4.6 Mapa del entorno utilizado para la implementación.

4.5.6 Modelo de actuación.

Con el fin de hacer lo más realista posible la simulación, el porcentaje de ruido utilizado para x, y va desde 0 hasta 5% y el porcentaje de ruido utilizado para θ va desde 0 hasta 10%. Se decidió tener el porcentaje de error variable en este rango, debido a que en situaciones reales existen muchos factores que impiden que el robot se desplace siempre de la misma manera. Entre estos factores podemos mencionar el deslizamiento de las ruedas

debido a la superficie, si el robot utiliza pilas, existen factores como la falta de voltaje constante, variaciones en la potencia de los motores, entre otros. La forma en que se obtiene el error de movimiento es el siguiente:

- Para x se obtiene de manera aleatoria el porcentaje de error (de 0 a 5%) que se le aplicará a la muestra.
- Se obtiene otro número aleatorio entre 0 y 1, si el número aleatorio obtenido es menor a 0.5, entonces a la acción en x se le resta el error; si el número aleatorio es mayor o igual a 0.5, entonces a la acción en x se le suma el error.
- Para y se obtiene de manera aleatoria el porcentaje de error (de 0 a 5%) que se le aplicará a la muestra.
- Se obtiene otro número aleatorio entre 0 y 1, si el número éste es menor a 0.5, entonces a la acción en y se le resta el error; si el número aleatorio es mayor o igual a 0.5, entonces a la acción en y se le suma el error.
- Para θ se obtiene de manera aleatoria el porcentaje de error (de 0 a 10%) que se le aplicará a la muestra.
- Se obtiene otro número aleatorio entre 0 y 1, si dicho número es menor a 0.5, entonces a la acción en θ se le resta el error; si por el contrario es mayor o igual a 0.5, entonces a la acción en θ se le suma el error.

Por ejemplo, si la acción fuese desplazarse 10 unidades en x, se obtiene de manera aleatoria el porcentaje de error entre 0 y 5% que se obtendrá a partir de esas 10 unidades (el error máximo sería 0.5 unidades). Supongamos que el número obtenido es 2.5%, entonces

la cantidad de error correspondiente es de 0.25 unidades. Posteriormente se obtiene otro número de forma aleatoria entre 0 y 1, para efecto del ejemplo, supongamos que este número es 0.75, entonces, a la acción se le deberá sumar el error correspondiente 10 + 0.25 = 10.25 y este es el desplazamiento en x que se le aplicará a la muestra. Los valores y y θ se obtienen de manera análoga, la diferencia para obtener θ radica en el porcentaje de error máximo. A medida que el robot avanza, aumenta la incertidumbre en la pose debido a la acumulación del error.

4.5.7 Modelo de observación.

La cámara de video simulada se definió con un campo visual de 45° y alcance de 25 centímetros, estos parámetros pueden ser fácilmente modificables. Las imágenes que proporciona dicha cámara son unidimensionales de 80 pixeles. Se decidió utilizar este modelo simplificado ya que es suficiente para capturar toda la información necesaria para que el robot pueda llevar a cabo el proceso de localización debido a que este se mueve en un plano. El robot utiliza la cámara para detectar balizas que le ayudarán a localizarse dentro del entorno. En la siguiente figura se muestra el alcance de la cámara superpuesta al entorno.

Figura 4.7 Campo visual de la cámara.

En la figura 4.8 se muestra la forma en que se realizan los cálculos para verificar si un *Baliza* se encuentra dentro del campo visual de la cámara. De esta forma, podemos generar la imagen que el robot vería desde cualquier punto determinado del entorno. El campo visual se encuentra definido como un ángulo α (que para este modelo es de 45°) y un alcanze L (en nuestro caso 25 centímetros).

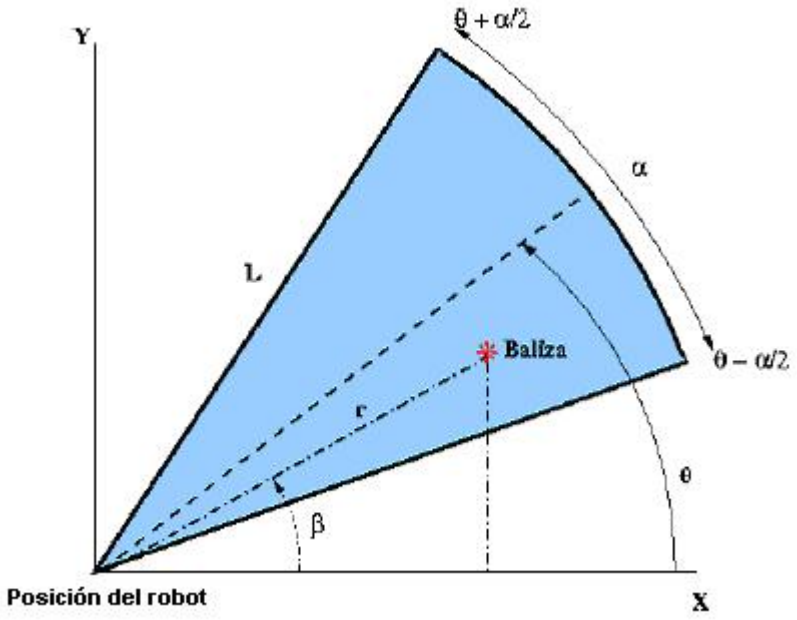

Figura 4.8 Detección de balizas.

Sea P la *Posición del robot* y b la posición de la *Baliza*. Se trazlada P al origen de tal forma que:

$$r = P - b = \sqrt{(b_y - P_y)^2 + (b_x - P_x)^2} \tag{5}$$

$$\beta = arctg \frac{b_y - P_y}{b_x - P_x} \tag{6}$$

La *Baliza* b se encuentra dentro del campo visual de la cámara si $r < L$ y $\beta \in \left[\theta - \frac{\alpha}{2}, \theta + \frac{\alpha}{2}\right]$.

Si la *Baliza* se encuentra dentro del campo visual de la cámara, entonces el píxel de la imagen en el que se captura dicha baliza es calculado como:

$$pixel = 80 * \frac{\left(\left(\theta + \frac{\alpha}{2}\right) - \beta\right)}{\alpha} \text{ para } \beta \leq \left(\theta + \left(\frac{\alpha}{2}\right)\right) \quad (7)$$

$$pixel = 80 - \frac{80 * \left(\beta - \left(\theta - \frac{\alpha}{2}\right)\right)}{\alpha} \text{ para } \beta > \left(\theta + \left(\frac{\alpha}{2}\right)\right) \quad (8)$$

En la siguiente figura (figura 4.9) se muestra un ejemplo de la imagen obtenida cuando el robot se encuentra en la posición de coordenadas (x, y, θ) = (19, 16, 180). Desde esta posición, la cámara capta tres balizas de "color" D en los pixeles 34, 39 y 45.

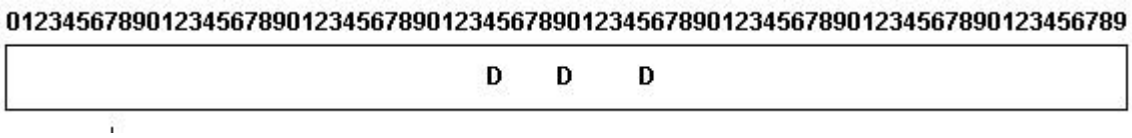

Figura 4.9 Ejemplo de una imagen capturada por la cámara.

4.6 Implementación.

Se parte de una población limitada de muestras que representan las posibles poses del robot. Las muestras se generan en cada instante a partir de la población anterior junto con las mediciones del entorno (para este caso en específico imágenes) y las acciones ejecutadas por el robot (desplazamiento en x, y y rotación en θ). Inicialmente al no conocer la posición del robot, se parte de una distribución uniforme de muestras que se obtienen de manera aleatoria. En cada instante de tiempo se ejecutan los tres pasos del método de localización Monte Carlo VBR:

1. **Resamplig.** Elegir N nuevos $\tilde{X}_t^{(i)}$ a partir de los $X_{t-1}^{(j)}$, en base a su $\omega_{t-1}^{(j)}$. Algunos $X_{t-1}^{(j)}$ pueden desaparecer y otros multiplicarse.
2. **Samplig.** $X_t^{(i)} = \tilde{X}_t^{(i)} + \Delta X_{t-1}^{(i)}$

   Donde $\Delta X_{t-1}^{(i)}$ depende de a_{t-1} y sigue una distribución de probabilidad normal que se deriva de $p(X_t^{(i)} \mid X_{t-1}^{(j)}, a_{t-1})$.
3. **Importance Samplig.** $\omega_t^{(i)} = \dfrac{p(o_t \mid X_t^{(i)})}{\sum_{\Phi_t} p(o_t \mid X_t^{(j)})}$

4.6.1 Resamplig.

En este paso del método se obtiene una nueva población en base a la población anterior, de tal forma que las nuevas muestras se concentren en las zonas de mayor probabilidad. Así se garantiza la convergencia del conjunto muestral en torno a la posición real del robot.

Este paso consta de dos fases, en la primera obtenemos una nueva población provisional de la siguiente manera: se generan N números aleatorios, donde N corresponde al tamaño de la población. Para cada número se selecciona la muestra cuyo puntaje ω está inmediatamente por encima del número calculado. De esta forma, las muestras con mayor probabilidad serán las que se seleccionen mayor número de veces.

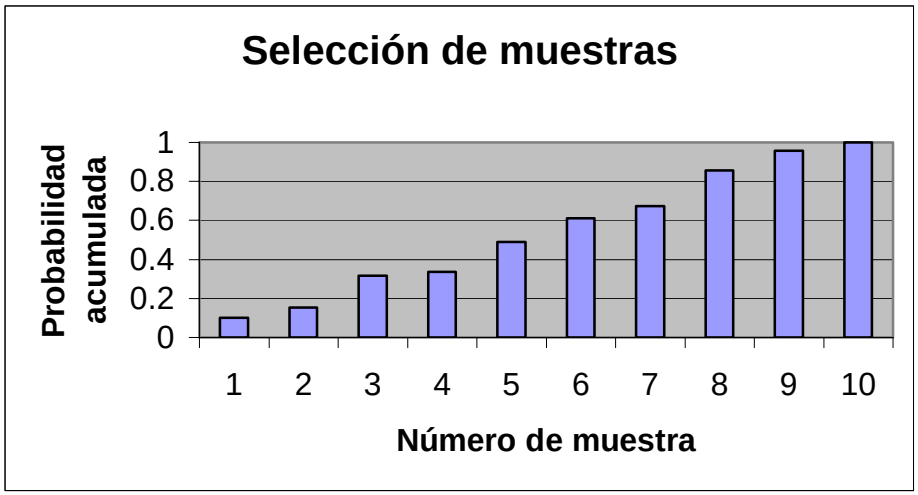

Figura 4.10 Probabilidades acumuladas del conjunto muestral.

En la figura 4.10 se ilustra con un ejemplo la forma en que se realiza la implementación de este paso para 10 muestras con probabilidades 0.1020, 0.0510, 0.1632, 0.0204, 0.1530, 0.1224, 0.0612, 0.1836, 0.1020, 0.0408 (cuyos valores se van acumulando y representando en el eje y) para las muestras 1 a 10 respectivamente (representadas en el eje x). Para el ejemplo anterior, se calcularían diez números aleatorios entre 0 y 1, ya que los puntajes de las muestras están normalizados y se elegiría la muestra del eje x cuyo

valor corresponde al tramo en el que se encuentra dicho número sobre el eje y. De esta forma, si el número aleatorio calculado fuese, por ejemplo 0.2653, se elegiría la muestra 3; si el valor fuese 0.5730, se elegiría la muestra 6 y así sucesivamente. Se puede observar en la gráfica del ejemplo (figura 4.10), que las muestras de mayor probabilidad están reflejadas en mayores tramos sobre el eje y, por lo tanto son elegidas un mayor número de veces que las de menor probabilidad.

En la segunda fase se verifica la calidad de la población para saber si no se ha perdido representatividad. Si la población sigue siendo representativa entonces se define una caja envolvente de la población.

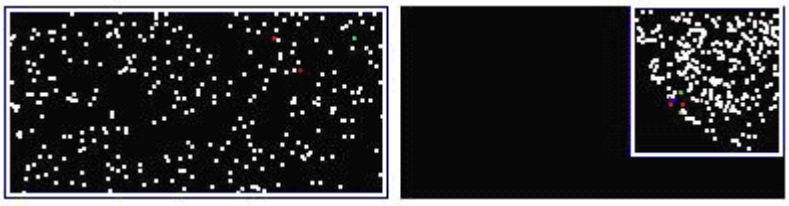

Figura 4.11 Caja envolvente de la población.

Si existe por lo menos una baliza en la imagen que está captando el robot, todas las muestras de la población provisional que tengan asociado un peso o puntaje bajo son reemplazadas con muestras donde la información sensorial así lo indique, estas muestras deben estar dentro de la caja envolvente que define el área de la población. Al utilizar esta caja envolvente, obtenemos una reducción acelerada en el área de la población, esto se

puede apreciar en la figura anterior. De esta forma se eliminan muestras poco verosímiles con muestras con mayor probabilidad y que están más cercanas a la posición real del robot.

4.6.2 Samplig.

En el fichero de entrada al sistema (fichero de recorrido) las acciones posibles consisten en desplazamientos en x, y y rotaciones en θ. Si una acción en el fichero de recorrido contiene instrucciones de desplazamiento y rotación, se realiza primero el desplazamiento y luego la rotación. Si se desea rotar primero, entonces se deberán realizar dos acciones: primero la rotación sin agregar instrucciones de desplazamiento y después el desplazamiento sin agregar instrucciones de rotación.

Cuando es la primera iteración, la población de muestras se distribuye uniformemente en el entorno ya que se desconoce la posición inicial del robot. En las siguientes iteraciones, las nuevas muestras se corresponden con las obtenidas en el paso de remuestreo (*resamplig*) de la iteración anterior tras aplicar la acción ejecutada por el robot a cada muestra añadiendo un cierto error de movimiento. Se aplican dos porcentajes de error: uno para x, y y otro para θ (la forma en que se obtuvo este error fue explicada en la sección 4.5.6). Este error de movimiento permite que el algoritmo no se bloquee si dentro del conjunto de muestras no se encuentra una cuyas coordenadas correspondan con las de la posición del robot. Sin error, el bloqueo se produciría cuando el conjunto muestral inicial no incorporase ninguna muestra representativa de la posición del robot. Al introducir este error de movimiento en cada desplazamiento y rotación, se incorporan muestras al conjunto muestral que pueden ser mejores o peores que las ya seleccionadas. Las mejores muestras

ayudarán a que la convergencia se produzca correctamente en torno a la posición real del robot, mientras que las peores serán desechadas en el siguiente remuestreo (*resamplig*).

4.6.3 Importance Samplig.

La probabilidad de que una imagen hay sido vista desde una determinada posición se calcula como $p(position \mid obs) = e^{-d^2/256}$. Donde d corresponde a la distancia entre la imagen teórica y la imagen observada. La distancia se define como la diferencia en pixeles entre las dos imágenes comparadas. Al hacer uso de este modelo satisfacemos la siguiente idea intuitiva: cuando la distancia es mayor entre las imágenes, menor es la probabilidad asignada a la pose desde la cual se captó la imagen teórica y viceversa.

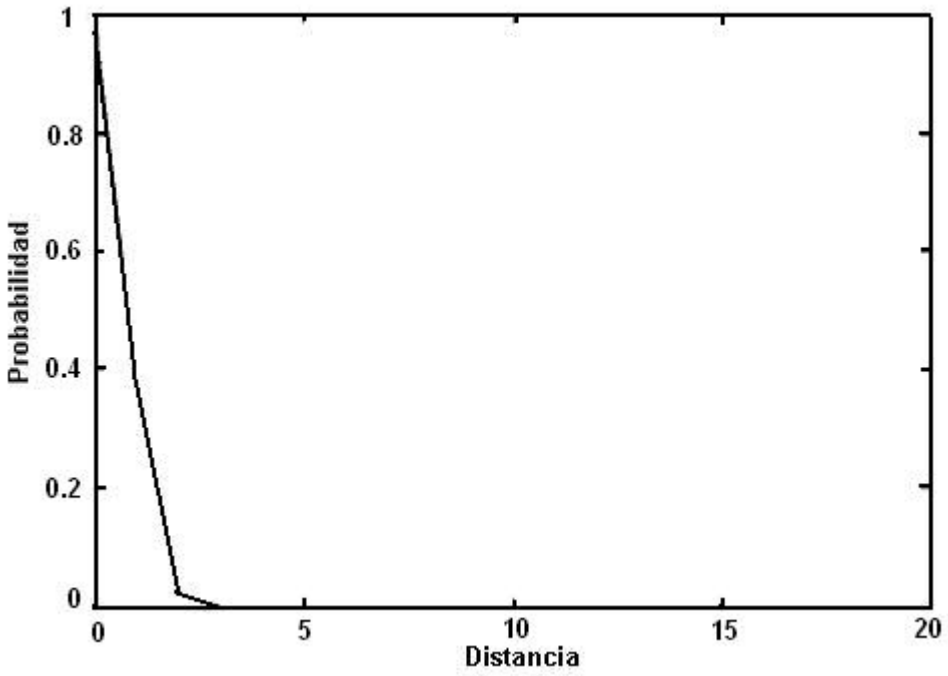

Figura 4.12 Verosimilitud de la posición en función de la distancia.

La función distancia se implementó comparando la imagen real, es decir, la imagen que se capta con la cámara desde la posición real del robot, y la imagen teórica, estimada por el Localizador para cada muestra o individuo de la población. Para cada baliza encontrada en la imagen real, se busca la más cercana que corresponda en color en la imagen teórica, entre ellas existe una diferencia d_i de pixeles. La distancia de la imagen real a la imagen teórica, se obtiene sumando las distancias de cada una de las balizas encontradas en las imágenes de tal forma que queda definida de la siguiente manera:

Sea b_r el número de balizas en la imagen real. La distancia de la imagen real a la imagen teórica es calculada como: $d_{12} = \sum_{i=1}^{br} di$.

Dado que las imágenes no son simétricas (el número de balizas encontradas en ambas imágenes puede ser distinto), se calcula de manera análoga la distancia de la imagen teórica a la real y se combinan ambos resultados.

Sea b_t el número de balizas en la imagen teórica. La distancia de la imagen real a la imagen teórica es calculada como: $d_{21} = \sum_{j=1}^{bt} dj$.

La distancia d utilizada para calcular la probabilidad $p(position \mid obs) = e^{-d^2/256}$, se obtiene a partir de la combinación de d_{12} y d_{21} de la siguiente forma:

1. Si b_r y b_t son 0, entonces $d = 0$.

2. Si b_r es 0, entonces $d = \frac{d_{21}}{b_t}$.

3. Si b_t es 0, entonces $d = \frac{d_{12}}{br}$

4. En cualquier otro caso la distancia es calculada como:

$$d = \frac{\left(\left(\frac{d_{12}}{b_r}\right) + \left(\frac{d_{21}}{b_t}\right)\right)}{2}$$

El rango de valores para la distancia va desde 0 (cuando las imágenes son idénticas) a 80 (cuando las imágenes son completamente diferentes).

4.7 Conclusiones del capítulo.

En este capítulo se presentó una breve discusión acerca de las debilidades encontradas en método Monte Carlo presentado en el capítulo anterior, se presentaron las modificaciones propuestas al método Monte Carlo y se propuso el algoritmo de localización Monte Carlo VBR. Además, se presentaron aspectos de la implementación como el modelos de actuación, el modelo de observación. También se muestra la forma en que se implementaron las tres pasos del método de localización Monte Carlo VBR (resamplig, samplig e importance samplig).