MONTE CARLO - ALGORITMO DE LOCALIZACION

ELVIS ALDEAN P.[1], KEVIN CHOEZ M.[1], JOHN GARCIA R.[1]

[1]Carrera de Ingeniería en Sistemas Computacionales
Universidad de Guayaquil

Resumen

En la actualidad muchas personas no saben ¿Qué es el Método Montecarlo? ¿Cómo se aplica? ¿Para qué sirve? Entre otras interrogantes, pero sin saberlo muchos de ellos lo han aplicado en diversas situaciones de la vida cotidiana. Por ejemplo:

Generar un número aleatorio a partir de la función RAN de la calculadora

Algoritmo de Las Vegas es un algoritmo de computación de carácter aleatorio (random) que no es aproximado: es decir, da el resultado correcto o informa que ha fallado.

El método de Monte Carlo en sí, es un método no determinista o estadístico numérico, usado para aproximar expresiones matemáticas complejas y costosas de evaluar con exactitud.

En este artículo mostramos como las técnicas Monte Carlo y la realización de experimentos por ordenador con números aleatorios, puede ser muy útil en la asimilación de estas materias.

Por la cual vamos a ver algunos métodos de monte Carlo que hay y la cual también nos va ayudar a realizar una pequeña simulación con número aleatorio en el lenguaje de programación python y también la metodología para realizar aprueba en un robot de localización y también vamos a explicar el cálculo de la simulación que realizada con los números aleatorio paso a paso y debemos tener un poco de conocimiento en la área de la estadística para así entender el articulo realizado para luego implementa la simulación deseada y también vamos observar el resultado final de una pequeña simulación con números aleatorios.

I. INTRODUCCIÓN

El problema de la localización en la robótica móvil se puede subdividir en dos: la estimación de la posición absoluta global del robot y la capacidad del robot de hacer un seguimiento de su posición de forma local. La posición absoluta, que es el caso que se aborda en este artículo, se define como la ubicación del robot dentro de un espacio acotado y conocido, sin más información disponible a priori para el robot, que el conocimiento de que se encuentra dentro de dicho mapa. Si se considera que el robot se encuentra localizado dentro del mapa, la segunda parte del problema, consiste en poder realizar de forma local un seguimiento de su posición. La subdivisión del problema en dos, permite utilizar el posicionamiento global para que, haciendo uso del mapa disponible, sea posible planificar una serie de movimientos que el robot debe realizar para cumplir con su misión, mientras que el seguimiento local de su posición le permite realizar una navegación eficiente. Esto, le facilita el poder tener en cuenta imprevistos no reflejados en el mapa, como por ejemplo cualquier tipo de obstáculo móvil.

Otros aspectos clave de los enfoques probabilísticos es que proporcionan una localización global más precisa que con modelos de localización basados en la subdivisión del mapa en celdas fijas, permitiendo además una fácil implementación. Sin embargo, esto no está exento de contrapartidas. En el caso del método Montecarlo expuesto en este trabajo, éste puede dar lugar a falsas localizaciones, sobre todo en mapas simétricos, donde es imposible diferenciar la posición del robot exclusivamente con información geométrica. También, por norma general, el méto-

do probabilístico aquí descrito requiere de una mayor potencia computacional para la realización de los múltiples cálculos necesarios para localizar al robot.

II. Trabajos relacionados

II-A. Monte Carlo Dual

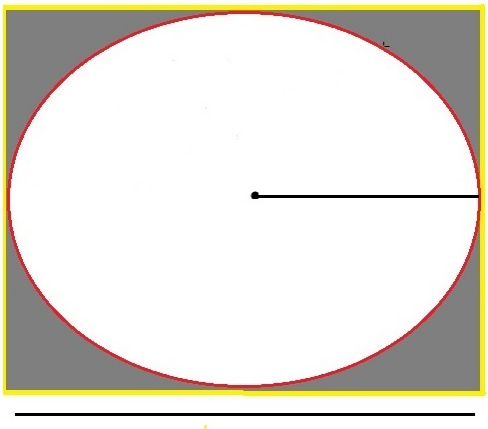
Este es una versión alternativa al metodo monte carlo.El principio de este metodo es invertir el proceso de muestreo. Monte carlo genera las nuevas particulas según la observacion mas reciente y luego ajusta los factores de importancia segun la creencia anterior. Por esta razon el algoritmo dual tiene fortalezas y debilidades complementarias a las de monte carlo, es ideal para sensores altamente precisos pero es muy sensible al ruido en las mediciones[1].

II-B. Mixure - Monte Carlo

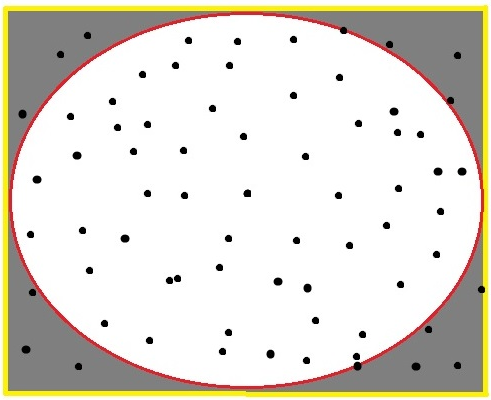
Este algortimo pretende heredar las fortalezas de monte carlo y monte carlo dual haciendo un algoritmo hibrido. La idea central es que cada muestra sea generada usando uno de los dos algoritmos, siendo aleatoria la decision acerca del cual de ellos se usa para generar la muestra[1].

III. Datos

Para realizar el cálculo de las posibles rutas que tome el robot se basara en el área del circulo y del cuadrado para por medio de ellos buscar el valor de pi.



Para la localizacion del robot se tomara en cuenta los puntos que se generen dentro del circulo



El método monte carlo consiste en una distribución de numeros aleatorios , el numero de iteraciones serán determinados por el programador, el procesamiento de las interacciones ocurren de forma independiente[3].

*III-A. **Calculo de distancias***

- Para realizar el calculo de la distancia siempre nos basamos en el punto de origen O = (0.5,0.5)
- El otro punto que se tomara en cuanta será los que estén representado dentro del circulo representado por (x,y)[2].
- Tomando en referencia las coordenadas de (x,y) calculamos el valor delta de X y Y

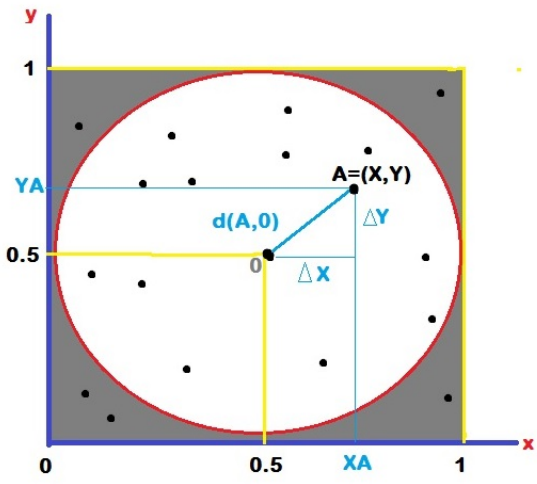
$$\Delta X = XA - 5 \tag{1}$$

$$\Delta Y = YA - 5 \tag{2}$$

- Una vez obtenido dichos valores calculamos la distancia ”FORMULA DE DISTANCIA”

$$d(A,0) = \sqrt{(\Delta X)^2 + (\Delta Y)^2} \tag{3}$$

- Si el resultado de la distancia es menor a 0,5 será el punto mas optimo para la ubicación del robot[4]



IV. METODOLOGÍA

IV-A. Filtros de Bayes

Los filtros de Bayes apuntan a problema de estimar el estado (x) de un sistema dinámico, es decir variante en el tiempo. Esta densidad de probabilidad generalmente es llamada la creencia[4].

$$bel(x_t) = p(x_t)|o_t, a_t - 1, o_t - 1, a_t - 2, ..., o_0) \tag{4}$$

El método Monte Carlo no requiere una definición analítica del modelo de movimiento; basta con tener un modelo de muestreo que sea compatible con

$$p(x', |X, a) \tag{5}$$

IV-B. Resultado con Python

Aquí vemos una pequeña simulación donde vamos a localizar el valor de Pi con el método monte-Carlo por la cual utilizamos los datos anteriores de la fórmula de la distancia.

Python

```
import math
import random
=========================================
    ***  HOLA  Grupo #4   ***
El valor que se acerca a Pi  es:==>  3.192
=========================================
None


    Px = random.random()
```

IV-C. Representación de la función de probabilidad

Dado que es imposible calcular las probabilidades de todas las infinitas posiciones posibles del robot[4], lo que se hace es de escoger un numero N de posiciones (en adelante partículas) representativas del espacio. Este conjunto de posiciones permite establecer la densidad de la probabilidad (o lo que es lo mismo la posición del robot y su incertidumbre)[5].

IV-D. Cálculo de la función de probabilidad

La forma de calcular la función de densidad de la probabilidad se hace partiendo de un estado inicial y realizando un cálculo en dos fases de forma iterativa. La primera, se denomina fase de predicción y, la segunda, fase de actualización[5].

Así pues, inicialmente, se reparte de forma aleatoria y homogénea el número de partículas por el espacio del mapa. Esto es así ya que inicialmente, como se había comentado anteriormente, la única información de la que se dispone es que el robot se encuentra dentro del mapa conocido a priori. Seguidamente se procede a la fase de predicción. Durante esta fase, basándose en el modelo de movimiento del control local del robot, se aplica a cada una de las partículas el movimiento realizado por el propio robot durante el intervalo de tiempo entre el instante inicial y el actual[1]. A dicho desplazamiento se le aplica un pequeño ruido gaussiano que tiene como objetivo modelar el error de un desplazamiento calculado exclusivamente a través de la odometría del robot, así como dispersar las partículas que por azar se encuentren muy próximas[6].

Seguidamente se realiza la fase de la actualización. Durante esta fase se cotejan las medidas del sensor láser con las medidas de un sensor láser virtual aplicado a cada partícula. Dicho láser virtual traza rectas en base a la posición y orientación de la partícula y devuelve la distancia mínima a la que intersectan con el mapa[3].

V. RESULTADOS

Se ha presentado el problema de la localización de un robot móvil en un marco de referencia absoluto[1], explicándose cómo se pueden utilizar funciones de densidad de probabilidad para

modelar el posicionamiento del robot con una cierta incertidumbre y se ha expuesto un algoritmo basado en el método de Montecarlo que permite el cómputo de dicha densidad[3].

Los diversos experimentos planteados, han demostrado que el algoritmo implementado es capaz de dotar al robot de capacidad para localizarse en escenarios complejos como los que componen los espacios arquitectónicos por los que nos movemos a diario con una incertidumbre sobre su posición muy baja[5].

Por todo lo expuesto, asumiendo que a día de hoy existen métodos que permiten obtener mapas del entorno fiables de forma sencilla y, dada la dificultad de encontrar espacios perfectamente simétricos dentro de las construcciones actuales, es posible afirmar que el método propuesto resuelve de forma sencilla, fiable y eficiente el problema de la localización en interiores.

VI. Conclusiones

En base a las pruebas realizadas sobre los métodos implementados podemos decir que el desempeño fue similar en la prueba de seguimiento de la posición[7]. ya que en este determinará la probabilidad de las partículas más cercanas agregados según los datos de visión. Una posible expansión de las funciones del módulo de localización es el cálculo de las distancia de los objetos, lo cual tiene gran importancia, ya que muchas de las decisiones que toma el localizar el objetivo tienen relación con la posición de la partículas, pero estas decisiones podrían ser más acertadas si se contara con un estimador de la posición futura de ella. Es natural que este cálculo sea realizado en el módulo de localización ya que es él quien tiene más información con respecto a las posiciones de los objetos, y es el que maneja las posiciones absolutas de los objetos con respecto al mapa. Finalmente, sería de gran contribución para el desempeño de los algoritmos implementados el mejoramiento de la cantidad y calidad de la información recibida desde visión. Por ejemplo, mejorar el cálculo de distancias, en cada partícula dependiendo la distancia que se fija al punto de origen, si hay mayores distancias y poder detectar las partículas con más rapidez que existentes en todo el mapa. Existen en la literatura varias aproximaciones a las mejoras anteriormente mencionadas. Por ejemplo, Monte Carlo Dual y Mixure - Monte Carlo

VII. Bibliografía

1 Dellaerty, F., Foxy, D., Burgardz, W., Thruny, S., Robust Monte Carlo Localization for Mobile Robots. Artificial Intelligence, vol 128 Issue 1- 2, May 2001

2 F. Dellaert, D. Fox, W. Burgard, S. Thrun. MonteCarlo localization for mobile robots. Proc. IEEE International Conference on Robotics and Automation (ICRA-99), Detroit, MI (1999).

3 Zhang, L., Zapata, R., Lépinay, P. Self-Adaptive Monte Carlo Localization for Mobile Robots Using Range Sensors. IEEE/RSJ International Conference on Intelligent Robots and Systems (2009).

4 Gleason, C.P.; Perez, L.C.; Goddard, S.; , .ºn the Ranging Connectivity in the Cricket Localization System,ÏEEE International Conference on Electro/information Technology, (2006), pp.619-624. doi: 10.1109/EIT.2006.252217

5 A. M. Johansen, L. Evers, Simulation and the Monte Carlo Methods — Lecture Notes, Ed. Nick Whiteley, University of Bristol, 2011.

6 S. Weinzierl, Introduction to Monte Carlo Methods, arXiv: hep-ph/0006269, 2000.

7 http://www.regeusp.com.br/arquivos/c6-Art7.pdf