Lección 7: Representación del Medio Ambiente

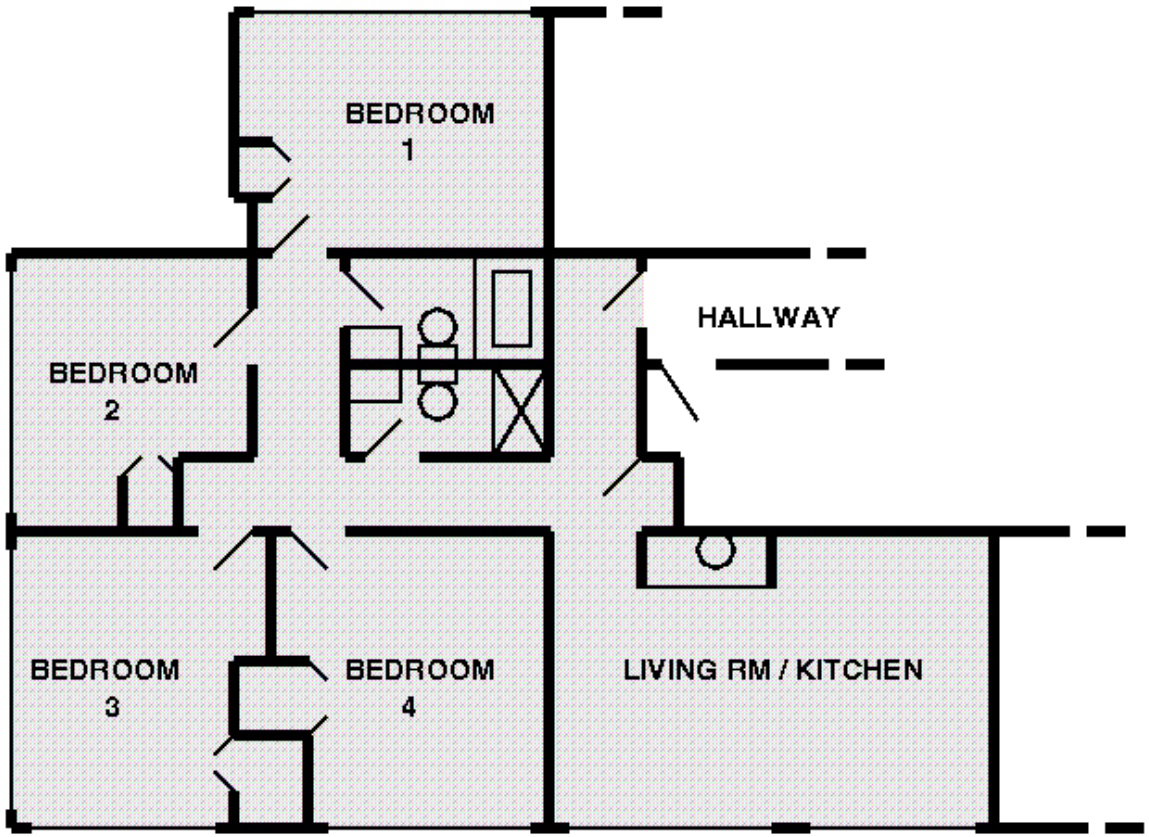
Jesús Savage
Facultad de Ingeniería, UNAM
Trabajo realizado con el apoyo del Programa
UNAM-DGAPA-PAPIME PE100821

19 de junio de 2023

Índice

Introducción

En esta lección se encontrarán cuales son los pasos que se tienen que hacer para crear un representación del medio ambiente.



Introducción

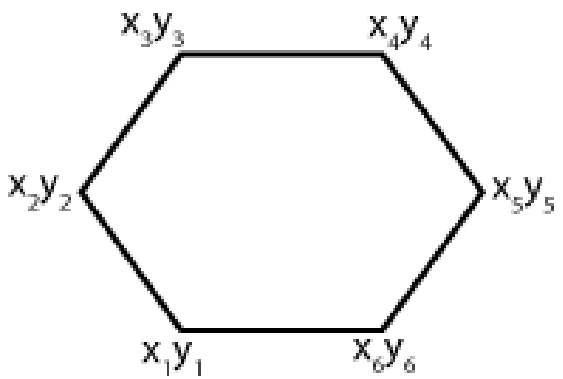
Para representaciones el medio ambiente se requiere una estructura de datos ideal:

- Mapea el medio ambiente lo más cercano posible a la realidad.
- Guarda figuras complejas en una estructura de datos con pocos elementos.
- Facilita la integración de datos provenientes de diferentes tipos de sensores.
- Que pueda manejar mapas locales y globales.
- A la medida para hacer planeación de rutas.
- Facilita el uso de algoritmos para la búsqueda de caminos, localización del robot y navegación.

Introducción

En los modelos tradicionales de la robótica de servicio se cuenta con una representación símbolica del medio ambiente, especialmente de los objetos en éste, es decir, del espacio ocupado en donde el robot no puede navegar. Esta representación simbólica es hecha principalmente por un humano. Se usa principalmente una representación poligonal usando los vértices de los poligonos:

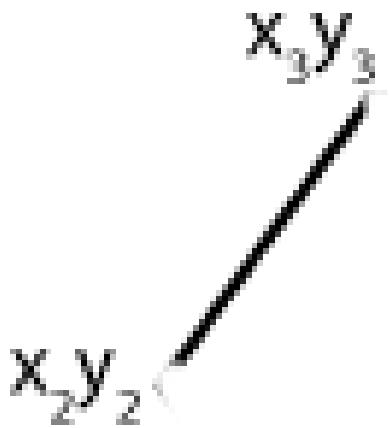
$$(x_1, y_1, x_2, y_2, ..., x_n, y_n)$$



Si las coordenadas están ordenadas con el sentido de las manecillas del reloj, como se muestra en la figura, éstas determinan un espacio ocupado

Representación Poligonal de un Medio Ambiente

Con los vertices de los poligonos se encuentran las líneas que los unen. Por ejemplo la línea que se forma con los vertices $(x_2, y_2), (x_3, y_3)$ se muestra en la siguiente figura:



Representación Poligonal de un Medio Ambiente

La ecuación general de una línea o segmento dada en forma parametrica es:

$$x = x_0 + mt$$

$$y = y_0 + nt$$

Para los vertices $(x_K, y_K), (x_L, y_L)$ Donde t es el parametro y (x_K, y_K) es el punto de la recta para $t = 0$. Adoptando la convención que t toma valores de 0 empezando en la orilla K y con $t = 1$ terminando en la otra orilla L, (x_L, y_L). Entonces:

$$x = x_K + (x_L - x_K)t$$

$$y = y_K + (y_L - y_K)t$$

Representación Poligonal de un Medio Ambiente

Esta ecuación puede ser convertida en forma implicita, en donde los puntos (x, y) estarán sobre la línea si se cumple:

$$f(x, y) = (y_K - y_L)x + (x_L - x_K)y + (x_K y_L - x_L y_K) = 0$$

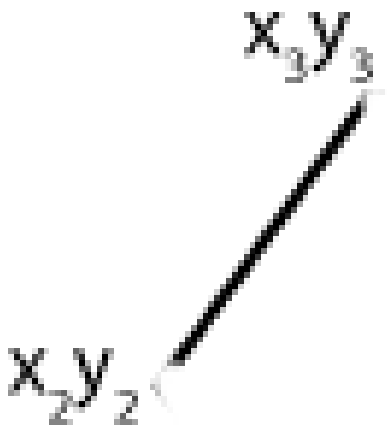
Mientras los puntos (x, y) están en el lado derecho de la línea dirigida si se cumple que:

$$f(x, y) = (y_K - y_L)x + (x_L - x_K)y + (x_K y_L - x_L y_K) < 0$$

y estaran del lado izquierdo si $f(x, y) > 0$
Se necesita checar que el rango de (x, y) este dentro de los limites de (x_K, y_K) y (x_L, y_L).

Representación Poligonal de un Medio Ambiente

$f(x, y) > 0$



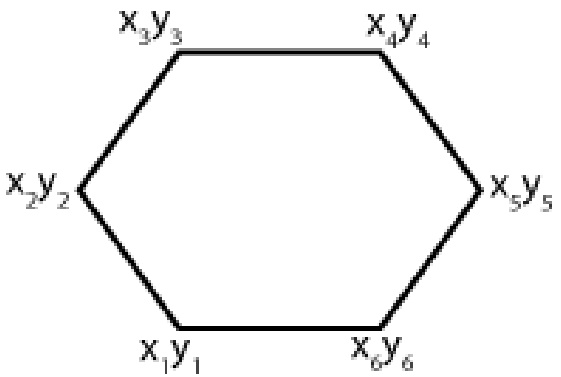
$f(x, y) < 0$

Representación Poligonal de un Medio Ambiente

Por lo tanto para saber si un punto esta dentro del espacio ocupado que define el poligono, con sus vertices ordenados con las manecillas del reloj, se debe cumplir que para todas las líneas formadas por los pares

$$((x_1, y_1), (x_2, y_2)), ((x_2, y_2), (x_3, y_3)), ..., ((x_n, y_n), (x_1, y_1)))$$

se cumpla que $f(x, y) < 0$

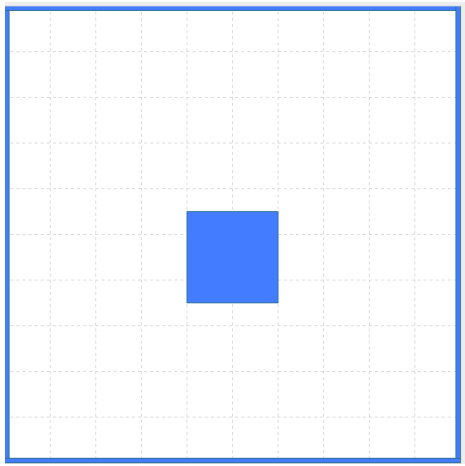


Esto se cumple para obstáculos representados con poligonos concavos.

Representación Poligonal de un Medio Ambiente

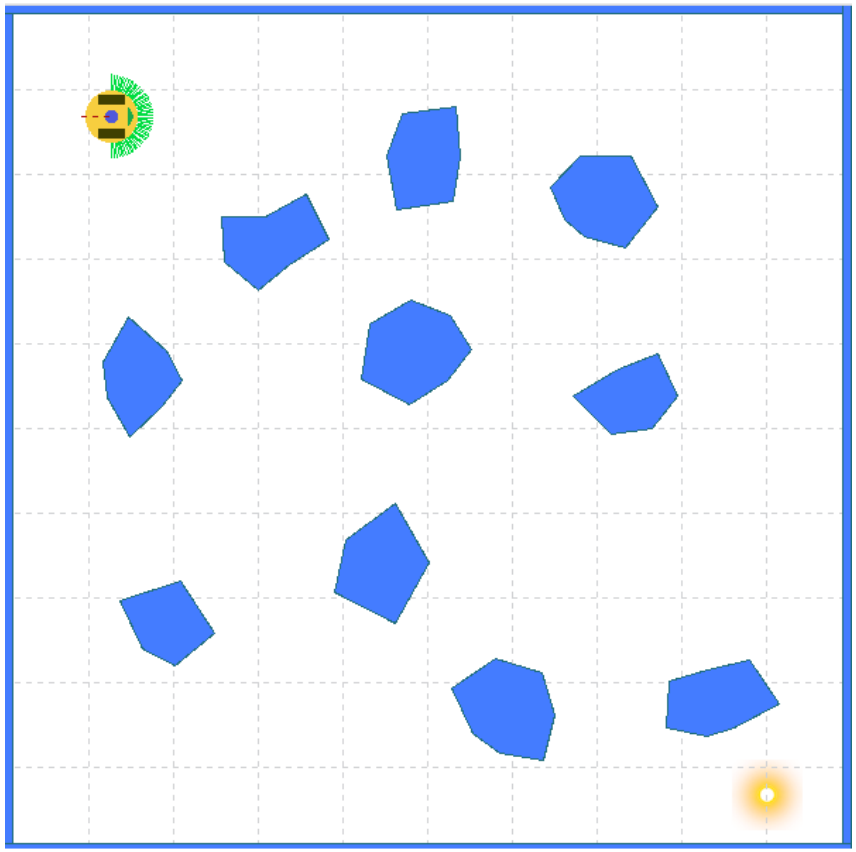
Por ejemplo, la siguiente figura muestra un obstáculo rodeado de paredes, el cual se representa simbólicamente de la siguiente forma, tomando el origen en el lado inferior izquierdo:

(dimensions room 1.000 1.000)
(polygon obstacle obs1 .40 .55 .60 .55 .60 .35 .40 .35)
(polygon wall wall1 0.0 0.0 0.0 1.0 0.01 1.0 0.01 0.0)
(polygon wall wall2 0.0 0.99 0.0 1.0 1.0 1.0 1.0 0.99)
(polygon wall wall2 0.99 1.0 1.0 1.0 1.0 0.0 0.99 0.0)
(polygon wall wall2 1.0 0.01 1.0 0.0 0.0 0.0 0.0 0.01)



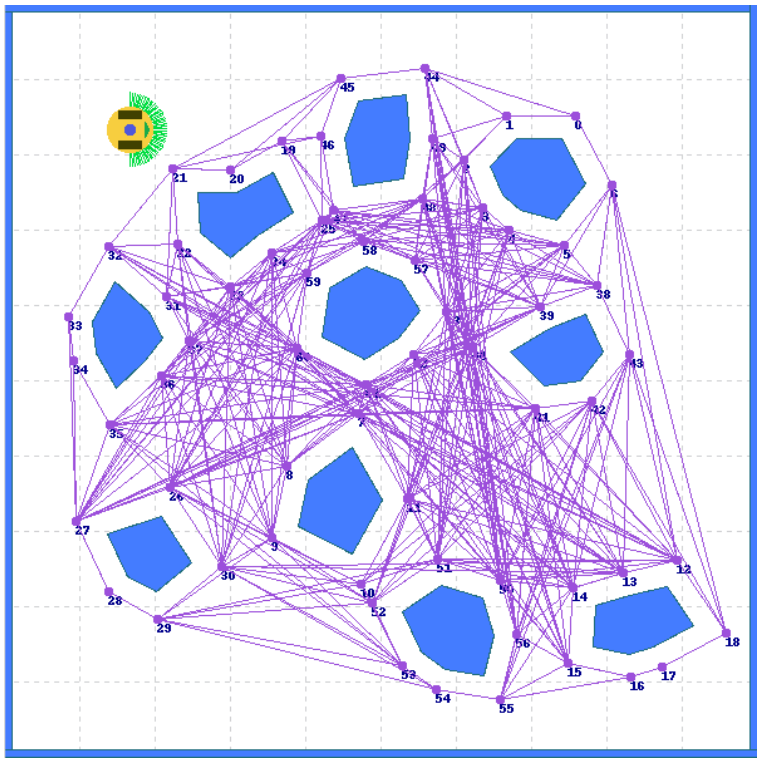
Representación Poligonal de un Medio Ambiente

Entonces todos los objetos que se encuentran en un medio ambiente pueden ser representados usando poligonos, incluidas las paredes como se muestra en la siguiente figura, aquí se incluye un robot y una fuente luminosa como destino.



Mapa Topológico

Expandiendo los poligonos de los objetos, por lo menos por el radio del robot, y uniendo los vertices de éstos, que no se intersecten con las líneas de los poligonos, se puede encontrar un mapa topológico del espacio libre como se muestra en la siguiente figura:



Creación de Mapas Usando Sensores

Para crear mapas utilizando lecturas que hace el robot con sus sensores de rango, éste toma una serie de muestras de los objetos que lo rodean. Los sensores pueden ser: sonar, láser, infrarrojo, visión estereo o RGB-3D. Estos sensores envían una señal la cual hace contacto con un objeto y midiendo el tiempo que tarda la señal en regresar al sensor y conociendo la velocidad en que viaja la señal, se puede calcular la distancia o rango al cual se encuentra el objeto. Con esta distancia y conociendo la posición del robot y el angulo del sensor con respecto al centro del robot se puede encontrar la posición del objeto con el que se esta haciendo contacto.

Creación de Mapas Usando Sensores

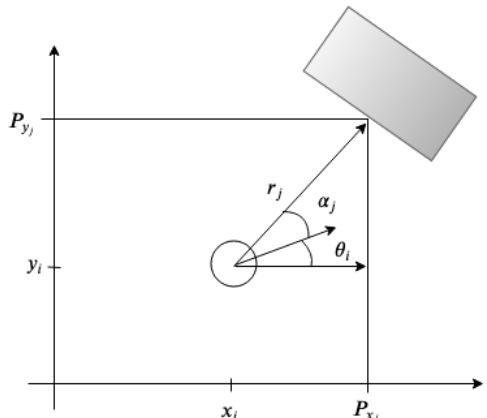
Cada sensor entrega una rango r_j el cual representa la distancia a la cual se encuentra un obstáculo. Conociendo la posición del robot en el tiempo i y el rango r_j se puede encontrar la posición del objeto en donde hizo contacto la señal del transductor:

$$\textit{Pose del Robot} = \{x_i,\ y_i,\ \theta_i\}$$

$$\textit{Pose del Obstaculo } P_j = \{P_{x_j},\ P_{y_j}\}$$

$$P_{x_j} = r_j\ cos(\theta_i + \alpha_j) + x_i$$

$$P_{y_j} = r_j\ sen(\theta_i + \alpha_j) + y_i$$

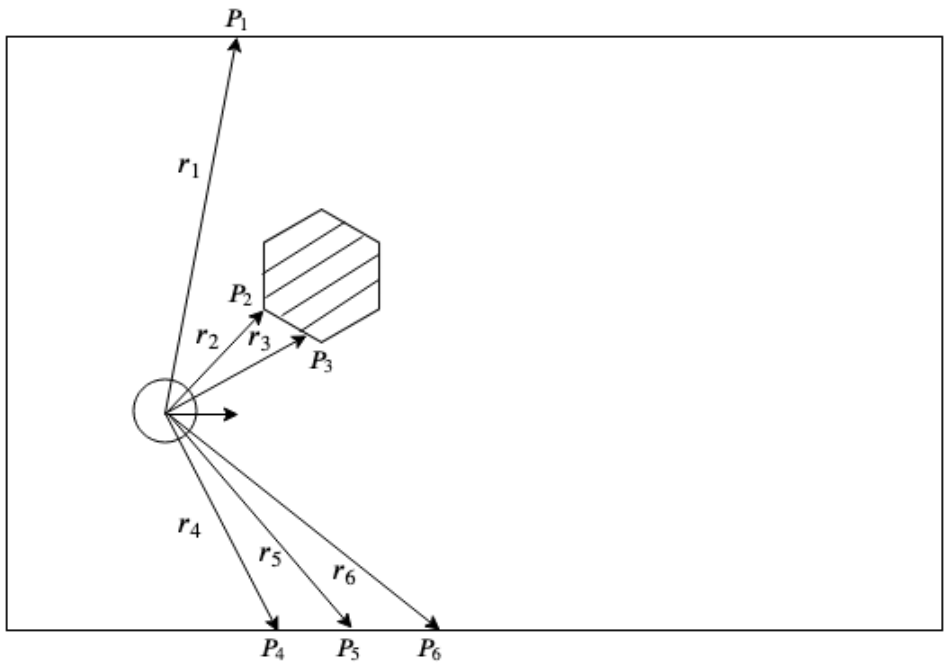


Creación de Mapas Usando Sensores

Se deja al robot que navegue en el cuarto tomando lecturas con sus sensores de rango, formando en el tiempo la nube de puntos del medio ambiente formada por los puntos:

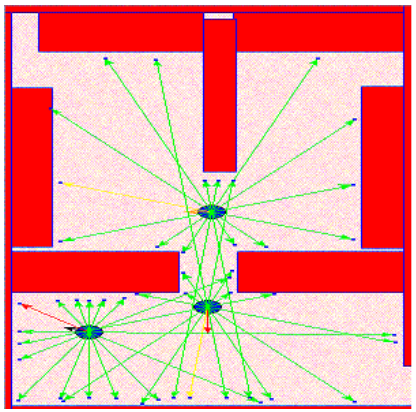
$$\underline{P^t} = (p_1^t, p_2^t, ..., p_N^t)$$

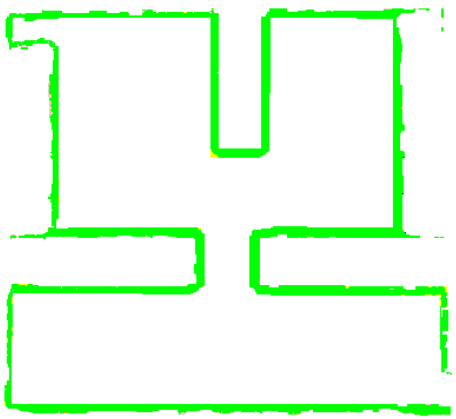
.



Creación de Mapas Usando Sensores

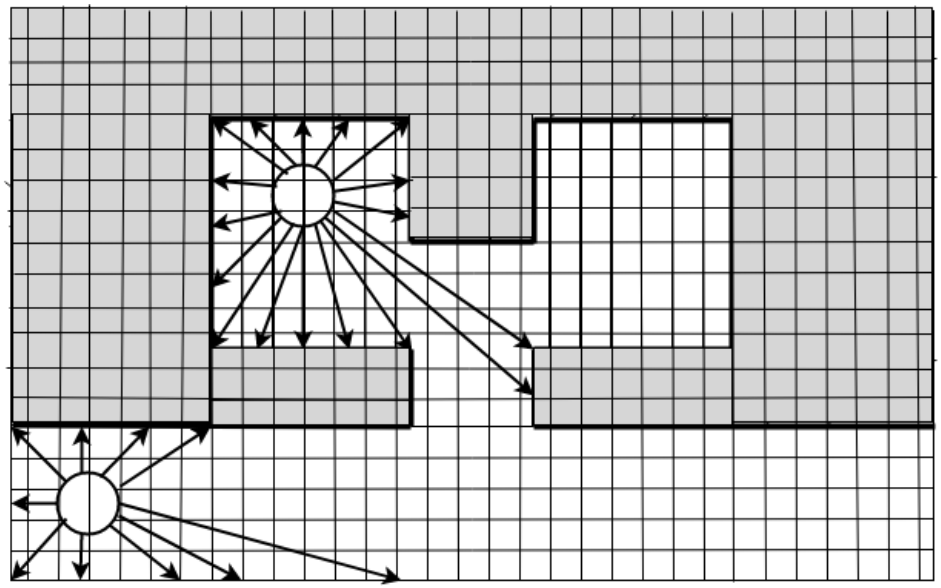
Con los puntos $\underline{P^t} = (p_1^t, p_2^t, ..., p_N^t)$ se van formando las nubes de puntos como se muestra en la siguiente figura: Se coloca al robot en el medio ambiente y esté empieza a navegar tomando muestras con su sensores.





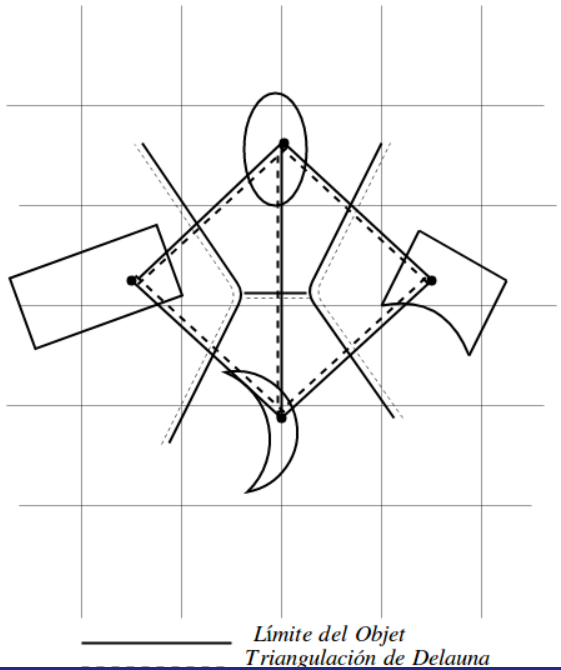
Celdas de Ocupación

El espacio en donde opera el robot se separa en espacio libre y en espacio ocupado utilizando celdas. En donde existan puntos $\underline{P^t} = (p_1^t, p_2^t, ..., p_N^t)$ la celdas se denominan celdas ocupadas, en donde no existan estos puntos se llaman celdas libres, como se muestra en la siguiente figura:



Digramas de Voronoi

Uno de los objetivos en la creación de mapas es el encontrar caminos que sean seguros para la navegación de un robot, para que éste no colisione con los objetos en el medio ambiente. Los diagramas de Voronoi encuentran una separación del espacio tomando como referencia los centroides de los objetos, encontrando líneas a la mitad de la distancia entre éstos. Estas líneas dejan un margen para que el robot pueda circular entre ellos sin chocar.



Digramas de Voronoi

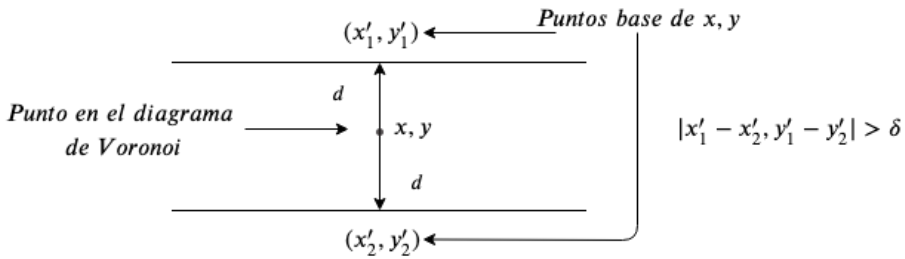
Para encontrar los diagramas de Voronoi cuando se tiene una nube de puntos se procede de la siguiente forma:

1. Encontrar puntos bases

Considérese un punto $< x, y > \in C$ en el espacio libre. Los **puntos base** de $< x, y >$ son los puntos $< x', y' >$ más cercanos en el espacio ocupado $\overline{C}$.

Digramas de Voronoi

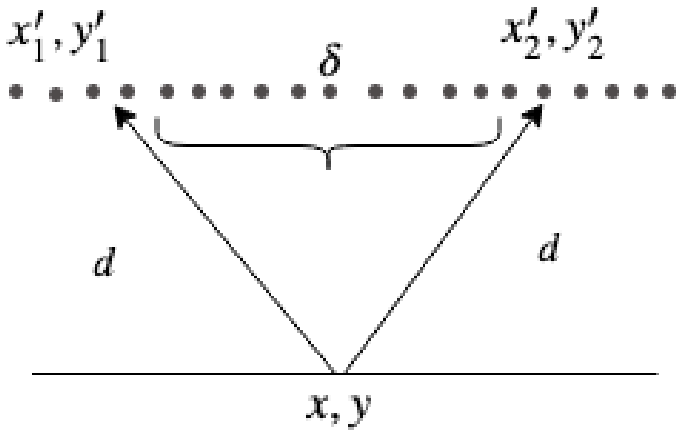
El diagrama de Voronoi es el conjunto de puntos en el espacio libre que tiene cuando menos dos puntos base diferentes a la misma distancia y que se encuentran separados por una δ.



$$|(x'_1, y'_1) - (x, y)| = |(x'_2, y'_2) - (x, y)| = d$$

Diagramas de Voronoi

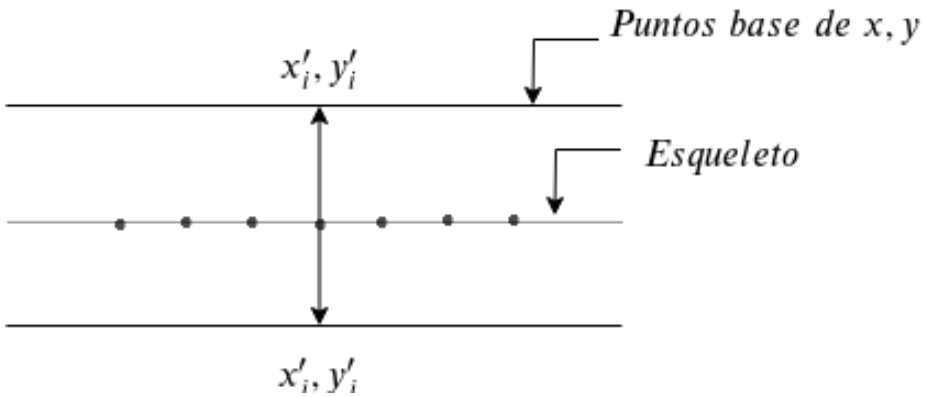
En la siguiente figura a pesar que los puntos (x_1', y_1') y (x_2', y_2') se encuentran a la misma distancia d de (x, y), no que se encuentran separados por una δ y por lo tanto no forman puntos base.



$$|x_1' - x_2', y_1' - y_2'| < \delta$$

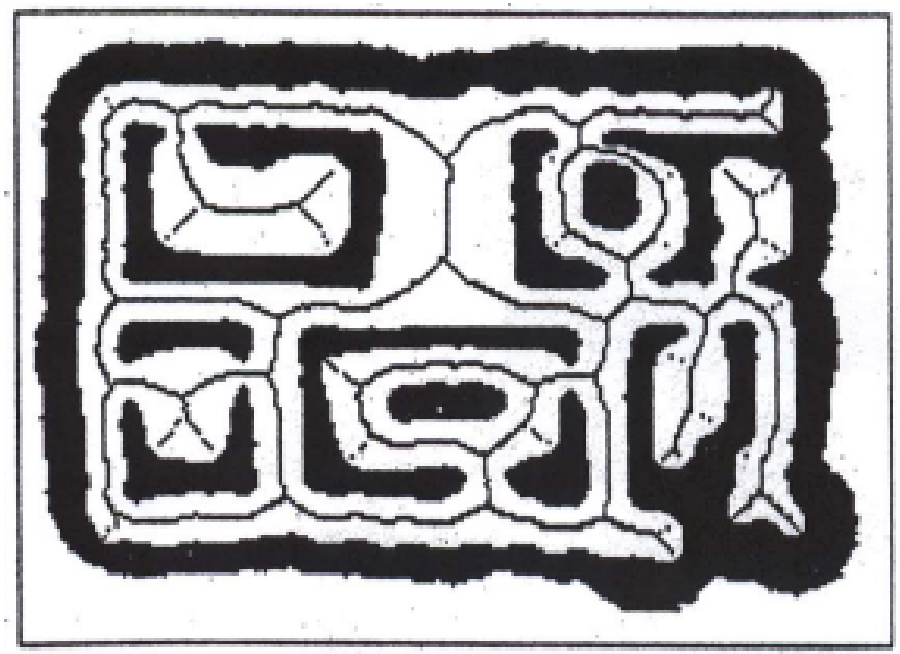
Diagramas de Voronoi

El conjunto de puntos (x_i, y_i) que tienen dos puntos bases diferentes forman el diagrama de Voronoi como se muestra en la siguiente figura:



Diagramas de Voronoi

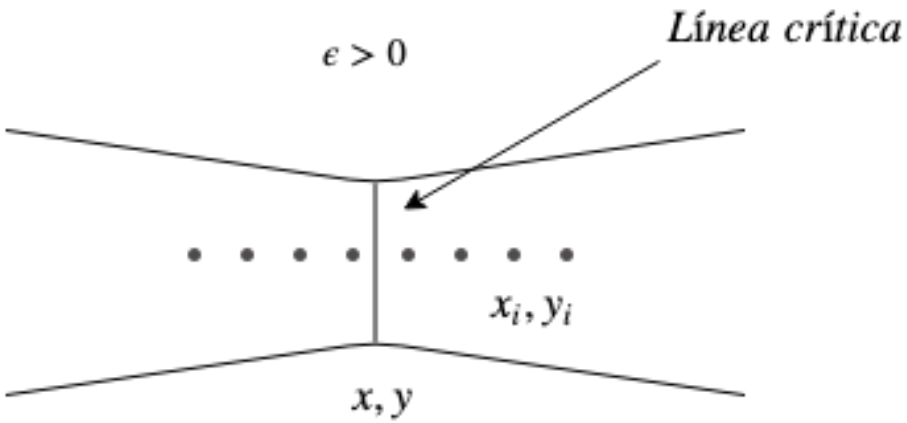
La siguiente figura muestra un diagrama de Voronoi para una nube de puntos de un medio ambiente.



Diagramas de Voronoi

2. Encontrar puntos críticos

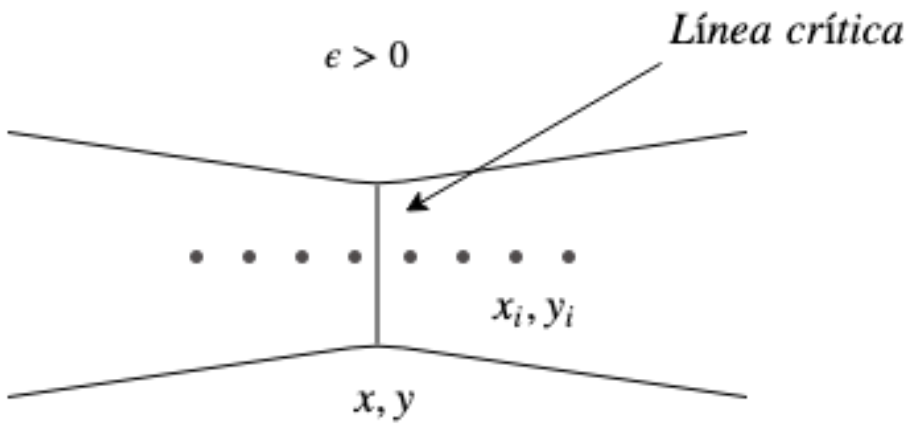
Los **puntos críticos** $< x, y >$ son puntos en el diagrama de Voronoi que minimiza el margen localmente:
Existe una $\epsilon > 0$ para la cual el margen de todos los puntos en el vecindario de $< x\ ,\ y >$ no es menor.



Diagramas de Voronoi

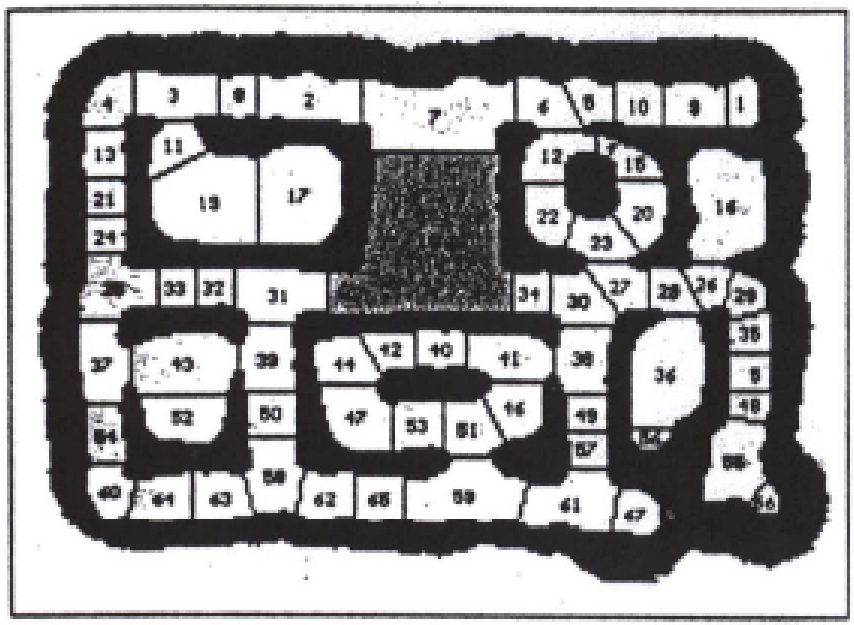
3. Encontrar líneas críticas

Las **líneas críticas** son obtenidas conectando cada punto crítico con sus puntos bases.



Diagramas de Voronoi

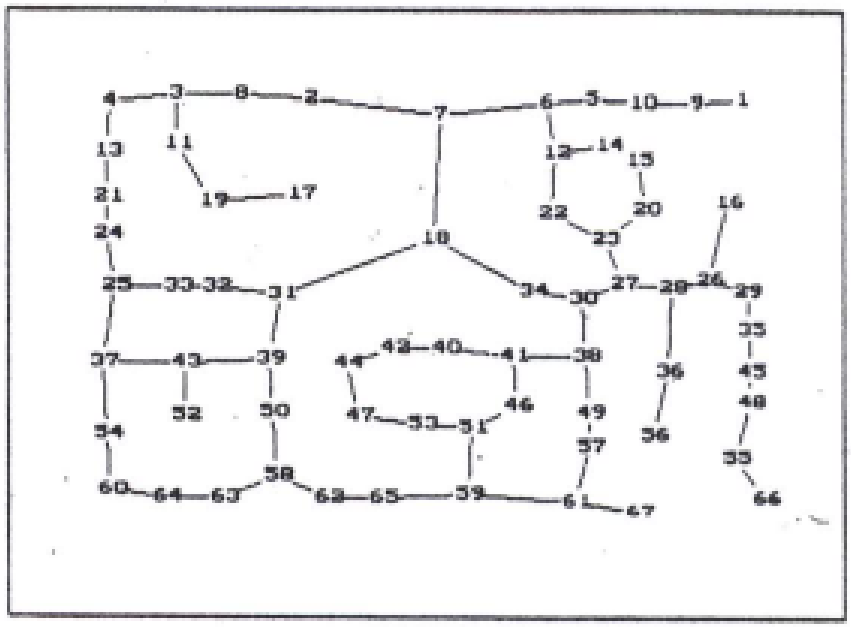
Líneas críticas particionan el espacio libre en regiones disconjuntas.



Diagramas de Voronoi

4. Encontrar las gráficas topológicas

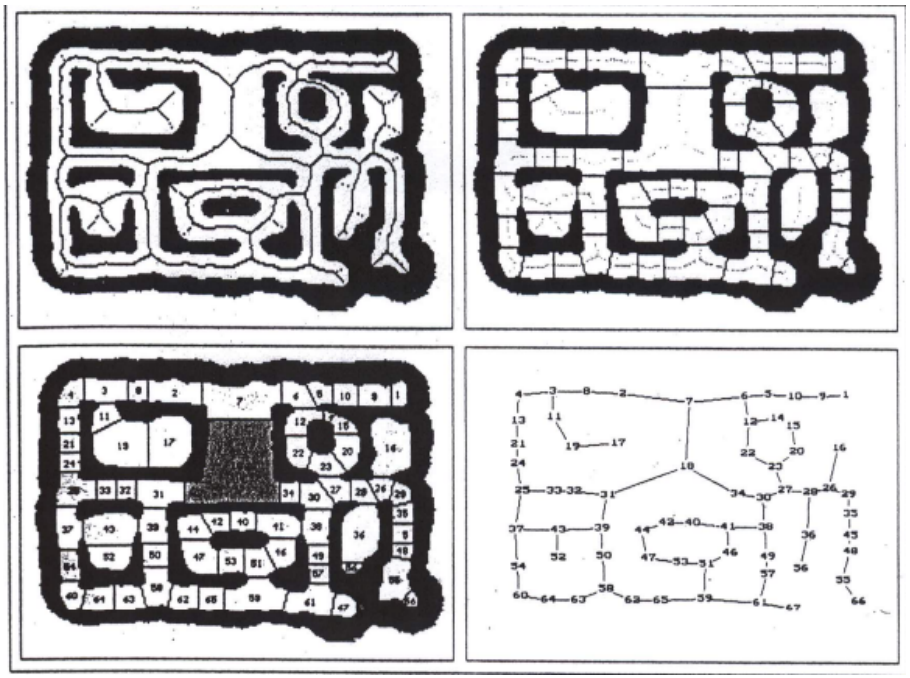
Las **gráficas topológicas** se crean usando los centros (nodos) de cada región separada por las líneas críticas.



Diagramas de Voronoi

Recapitulando:

1. Encontrar puntos bases
2. Formar el diagrama de Voronoi
3. Encontrar puntos críticos
4. Encontrar líneas críticas
5. Encontrar las gráficas topológicas



Agrupamiento (Clustering) Usando Cuantización Vectorial

Dado un conjunto de N_v vectores, $\underline{X_j} = (x_j, y_j), j = 1, \ldots, N_v$ que representan la posición de puntos en el espacio libre, un conjunto de centroides es encontrado los cuales representan los vectores. Una colección de centroides es llamado *libro de codificación* o *CodeBook*.
Para encontrar los centroides óptimos que particionen un espacio se utiliza el algoritmo modificado de Lloyd o mejor conocido como Linde-Buzo-Gray (LBG)

Algoritmo de Linde-Buzo-Gray

1. Calcular el primer centroide:

Dados los puntos en el espacio libre:

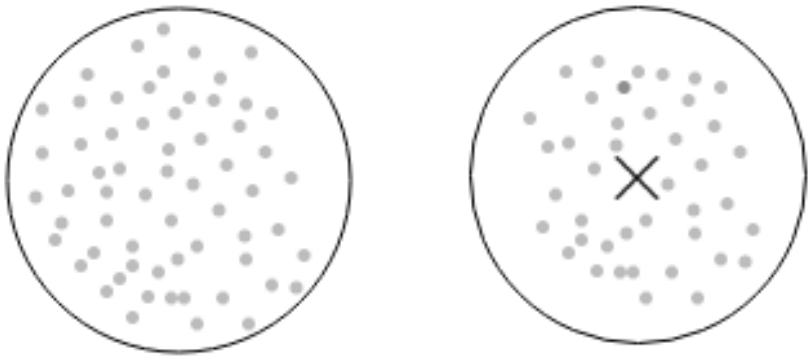
$$\underline{X}_1 = [x_1, y_1]$$
$$\underline{X}_2 = [x_2, y_2]$$
$$\vdots$$
$$\underline{X}_i = [x_i, y_i]$$
$$\vdots$$
$$\underline{X}_{N_v} = [x_{N_v}, y_{N_v}]$$

Algoritmo de Linde-Buzo-Gray

El primer centroide se calcula, con $i = 1$:

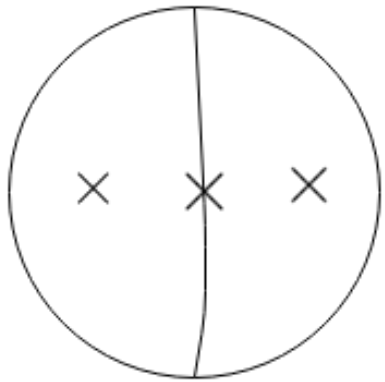
$$\overline{C_i} = \frac{1}{N_v} \sum_{j=1}^{N_v} \underline{X_j}$$

$$\overline{C_i} = [C_{ix}, C_{iy}] = \left[\frac{1}{N_v} \sum_{j=1}^{N_v} x_j \ , \ \frac{1}{N_v} \sum_{j=1}^{N_v} y_j \right]$$



Algoritmo de Linde-Buzo-Gray

2. Generar dos centroides nuevos a partir del centroide $\overline{C_i}$ o centroides anteriores:



$$\overline{C}_{i+1} = \overline{C_i} * \epsilon_1$$

$$\overline{C}_{i+2} = \overline{C_i} * \epsilon_2$$

$$\epsilon_1 = 0 \, . \, 9999$$

$$\epsilon_2 = 1 \, . \, 0001$$

Algoritmo de Linde-Buzo-Gray

3. Agrupe los vectores de entrenamiento en dos grupos de acuerdo al centroide más cercano.

$$d_1 = \textit{distancia}\,(\underline{X_j}\,,\ \overline{C}_{i+1})$$

$$d_2 = \textit{distancia}\,(\underline{X_j}\,,\ \overline{C}_{i+2})$$

Algoritmo de Linde-Buzo-Gray

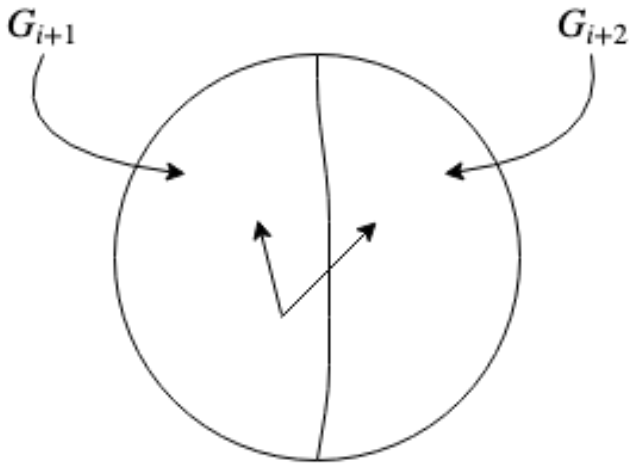
Si $d_1 < d_2$
Se asigna el vector $\underline{X}_j$ a la region G_{i+1}

$$\underline{X}_j \rightarrow G_{i+1}$$

en caso contrario asignarlo a la región G_{i+2}:

$$\underline{X}_j \rightarrow G_{i+2}$$

Esta operación se hace para todos los vectores, $1 \leq j \leq N_v$



Algoritmo de Linde-Buzo-Gray

4. Recalcular los centroides

El proceso de separar los vectores en regiones y encontrar nuevos centroides se debe repetir en varias iteraciones. Para cada iteración t se calcula la distancia global, empezando con:

$$D_g^t = 0$$

Sumar la distancia global con la menor distancia calculada para todos los vectores $\underline{X_j}$, $1 \leq j \leq N_v$

$$d_1 = \ distancia\ (\underline{X_j}\ ,\ \overline{C}_{i+1})$$

$$d_2 = \ distancia\ (\underline{X_j}\ ,\ \overline{C}_{i+2})$$

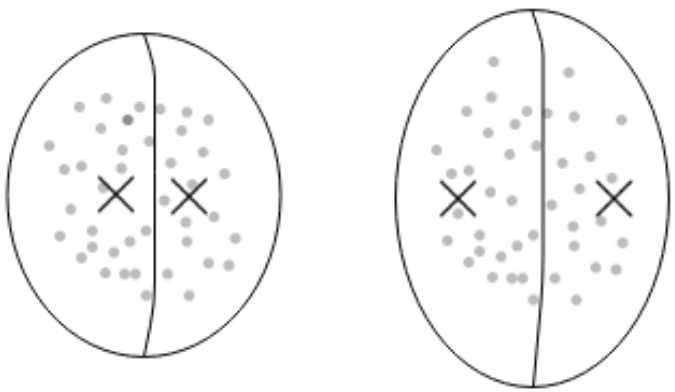
$$D_g^t = D_g^t + min\ (d_1, d_2)$$

Algoritmo de Linde-Buzo-Gray

Con los vectores separados se vuelven a calcular los centroides:

$$\overline{C}_{i+1} = \frac{1}{Num.\ Vect.\ G_{i+1}} \sum_{j=1}^{Num.\ Vect.\ G_{i+1}} \underline{X}_j$$

$$\overline{C}_{i+2} = \frac{1}{Num.\ Vect.\ G_{i+2}} \sum_{j=1}^{Num.\ Vect.\ G_{i+2}} \underline{X}_j$$



Algoritmo de Linde-Buzo-Gray

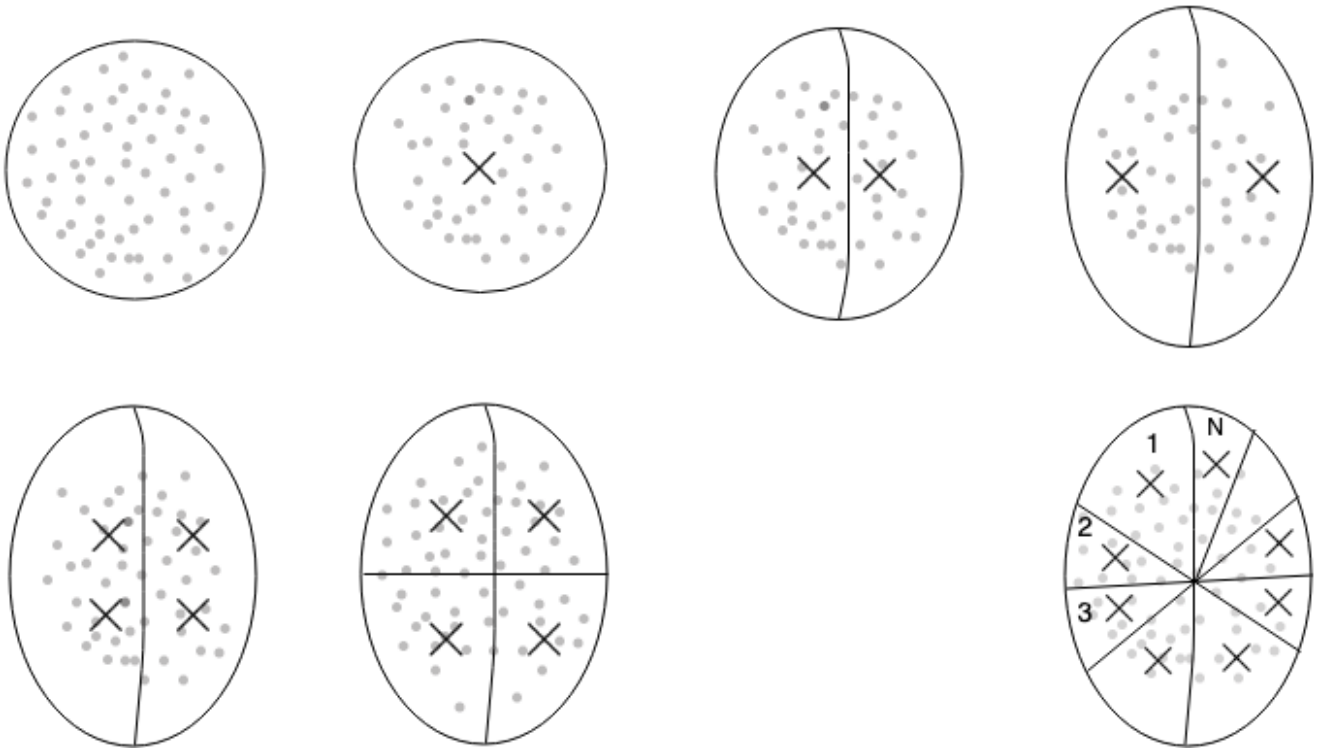
Si la diferencia de distancia entre una iteración actual y la anterior:

$$(D_g^t - D_g^{t_1}) \geq \epsilon$$

ir al punto 3. Este proceso se repite varias veces hasta que se estabilicen los centroides.
En caso contrario ir al punto 2 para modificar los centroides por una pequeña cantidad.

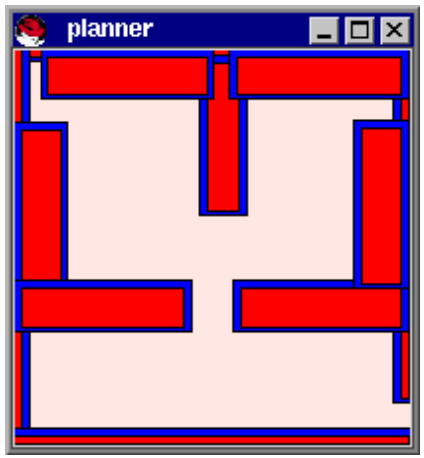
Algoritmo de Linde-Buzo-Gray

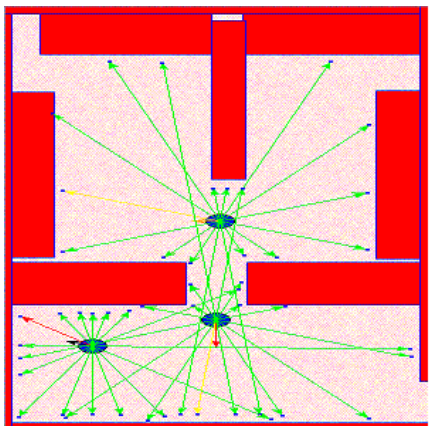
Se termina cuando se han obtenido las particiones necesarias.



Algoritmo de Linde-Buzo-Gray

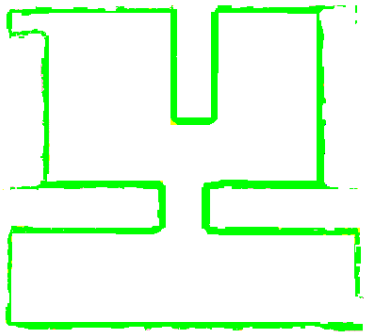
Ejemplo:
Se coloca al robot en el medio ambiente y esté empieza a navegar tomando muestras con su sensores.



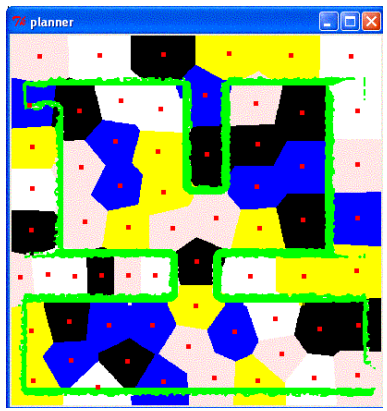


Algoritmo de Linde-Buzo-Gray

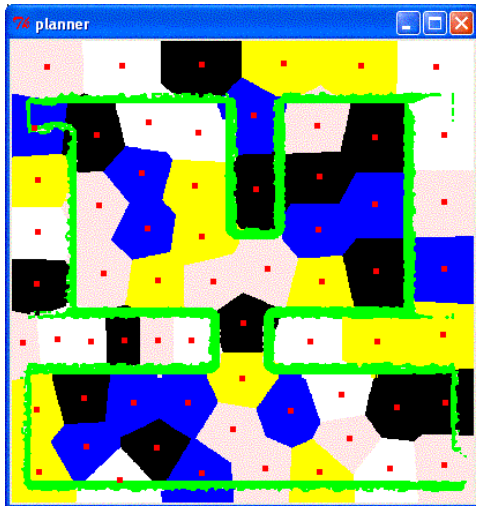
Se genera una nube de puntos del medio ambiente, separando el espacio libre del espacio ocupado.



Se cuantiza el espacio libre, es decir puntos en los cuales no se detectó un obstáculo.



Algoritmo de Linde-Buzo-Gray



Uniendo los centroides se encuentra el mapa topológico del lugar.