# ROBÓTICA


## *Grupo de Sistemas y Comunicaciones*

***jmplaza@gsyc.es***


***Curso 2007-2008***

# Localización

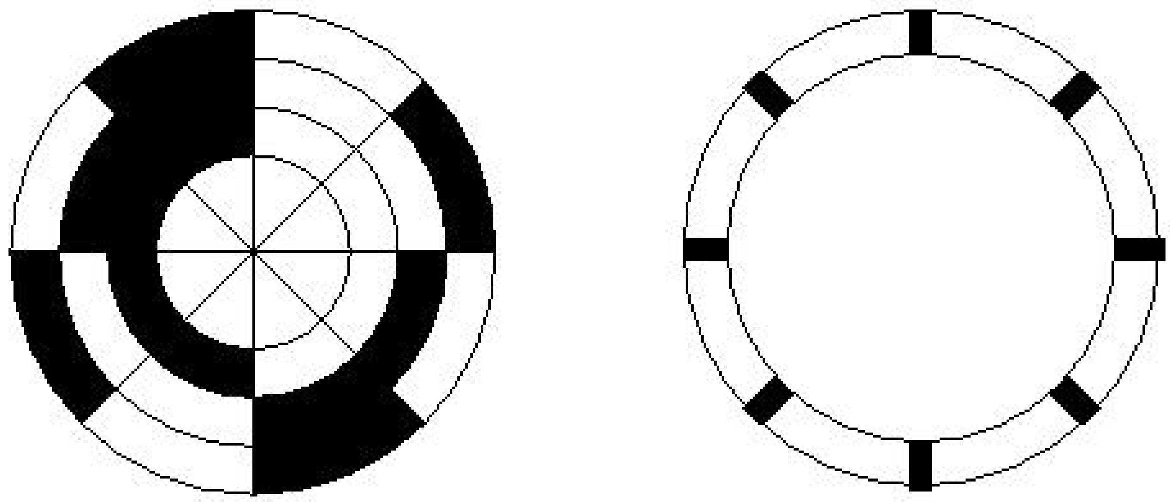
# Introducción localización

- Ya hemos visto algunos mecanismos de creación mapas
- Para poder usarlos necesitamos saber donde estamos
- Mecanismos usuales:
  - Odometría: movimiento de los actuadores
  - Sensores especializados: GPS
  - Otros sensores: cámara cenital
- Localización usando los propios mapas (Monte Carlo, etc.)

©2007 GSyC

# Odometría

- Suposición: Las vueltas de las ruedas pueden traducirse en desplazamientos lineales sobre el suelo
- *Encoders* absolutos y relativos

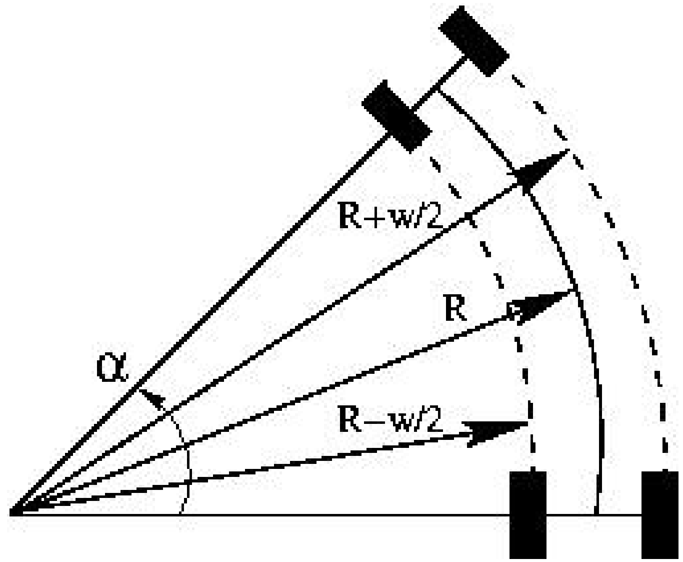


## Transformación de las coordenadas

- Nos interesa la posición $(x, y, \theta)$ del robot
- Los *encoders* nos devuelven unicamente pulsos o posición de las ruedas
- Para intervalos pequeños supondremos:
  - La posición inicial conocida $(x_0, y_0, \theta_0)$
  - Conocidos los pulsos por metro (p.e. 3240 en el caso del EyeBot)
  - La velocidad constante (tanto lineal como rotacional)
  - Los $k$ pulsos recorridos por cada *encoder* que se pueden traducir en distancia recorrida por cada rueda:

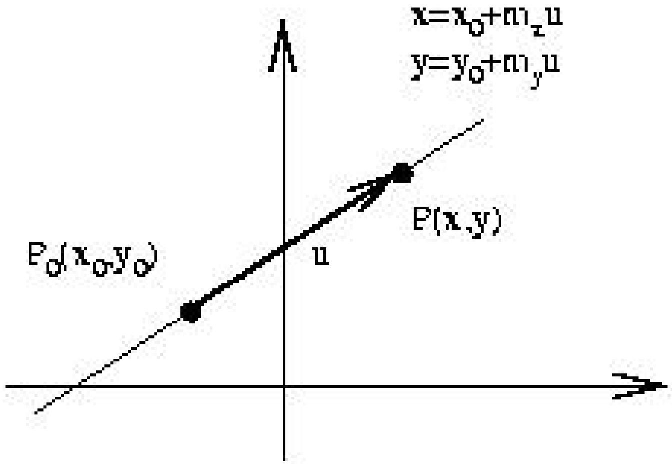
    $\Delta S_x = k/3240$ (donde $x$ será cada uno de los encoders)
  - La distancia entre los encoders $w$

## Transformación de las coordenadas




- Longitud de los arcos: $\Delta S_i = \Delta\alpha(R - \frac{w}{2})$, $\Delta S_d = \Delta\alpha(R + \frac{w}{2})$
- Si $|\Delta S_i| < |\Delta S_d|$ la rueda derecha se ha movido más, el centro de giro está a la izquierda del robot (como en la figura)
- Si $|\Delta S_i| > |\Delta S_d|$ el centro de giro estará a la derecha

## Transformación de las coordenadas



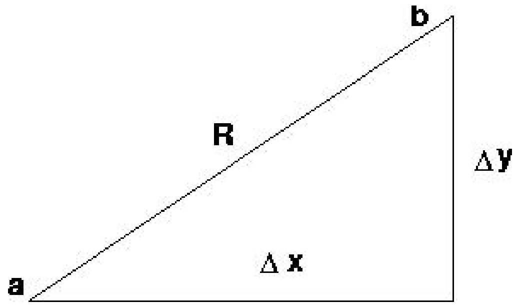

- Ecuación paramétrica de la recta que pasa por el centro ($a$) del robot:

$$y = y_a + m_y\lambda \qquad\qquad x = x_a + m_x\lambda$$

- Como la distancia del centro a una rueda (punto $b$) es conocida, es $\lambda = \frac{w}{2}$, la recta que pasa por la rueda y el centro nos permite despejar:

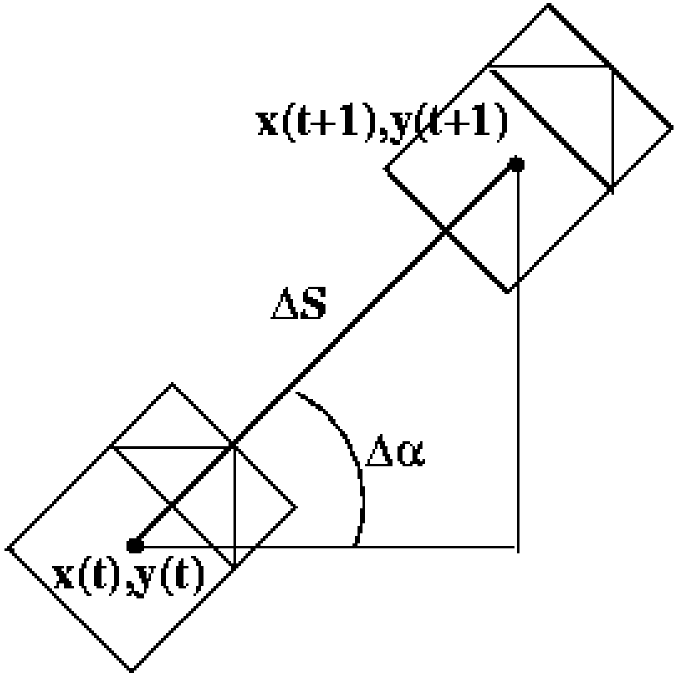
$$m_y = (y_b - y_a)\frac{2}{w} \qquad\qquad m_x = (x_b - x_a)\frac{2}{w}$$

## Transformación de las coordenadas




- La distancia del centro del robot al centro de giro es $R = \sqrt{\Delta y^2 + \Delta x^2}$
- $\lambda = \frac{R^2}{(m_y \lambda)^2 + (m_x \lambda)^2}$
- La nueva orientación: $\theta(t+1) = \theta(t) + \Delta\alpha$
- En función de la nueva orientación y la posición del centro de giro $(x_{cdg}, y_{cdg})$:
  - $x_r(t+1) = Rcos\theta(t+1) + x_{cdg}$
  - $y_r(t+1) = Rsin\theta(t+1) + x_{cdg}$

## Transformación de las coordenadas(recta)



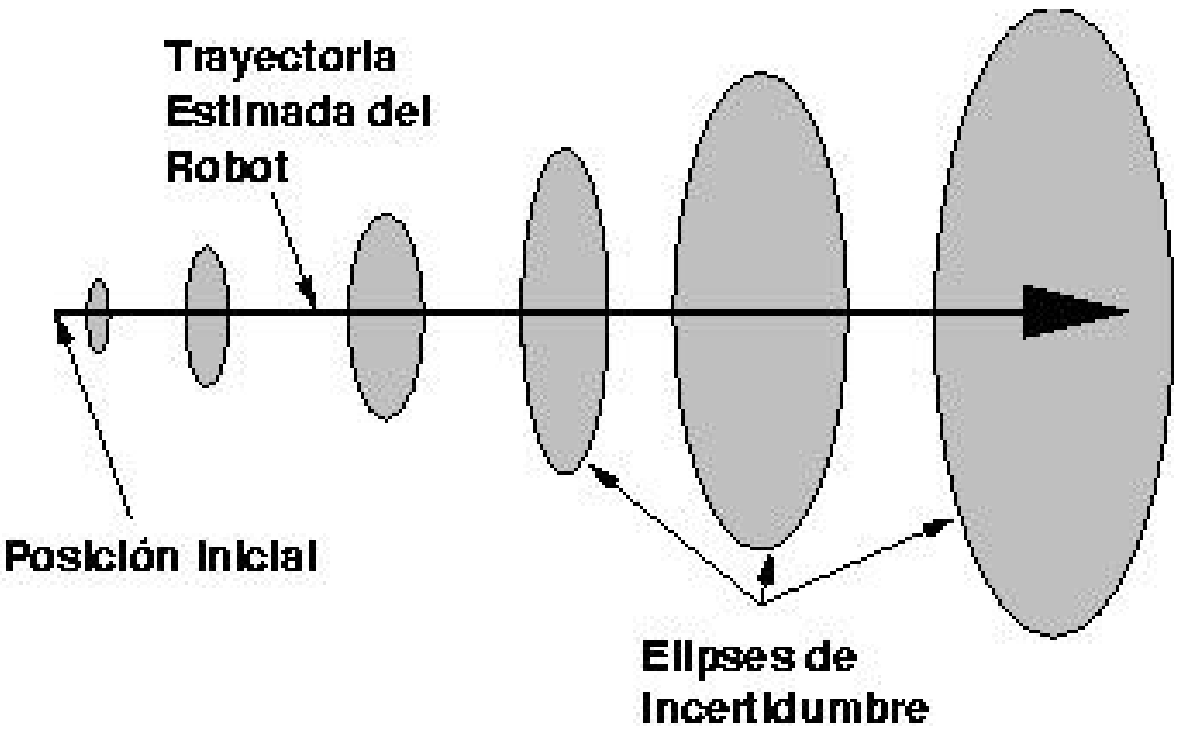

Si se mueve en línea recta ($|\Delta S_i| = |\Delta S_d|$) es más fácil (trigonometría):

- $x_r(t+1) = x_r(t) + \Delta S cos\beta$
- $y_r(t+1) = y_r(t) + \Delta S sin\beta$

## Problemas con la odometría

- Suposición débil: errores por deslizamiento, holguras, precisión, etc.
- Errores sistemáticos:
  - Diametros de rueda diferentes
  - Alineamiento de las ruedas
  - Resolución y muestreo de los *encoders*
- Errores no sistemáticos:
  - Patinaje de las ruedas
  - Rugosidad del suelo
  - Objetos en el suelo

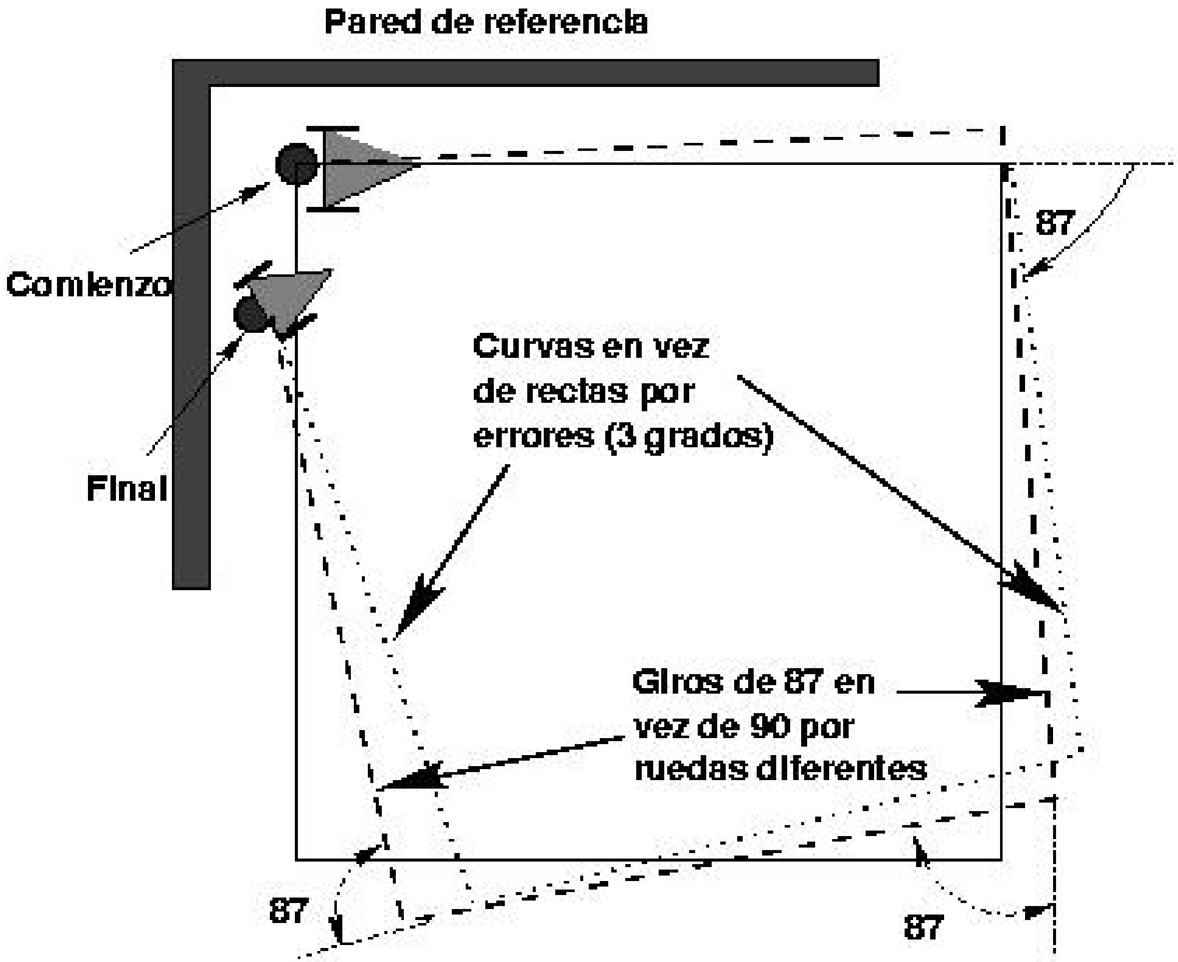
## Elipse de error estimado (Tonouchi 1994)



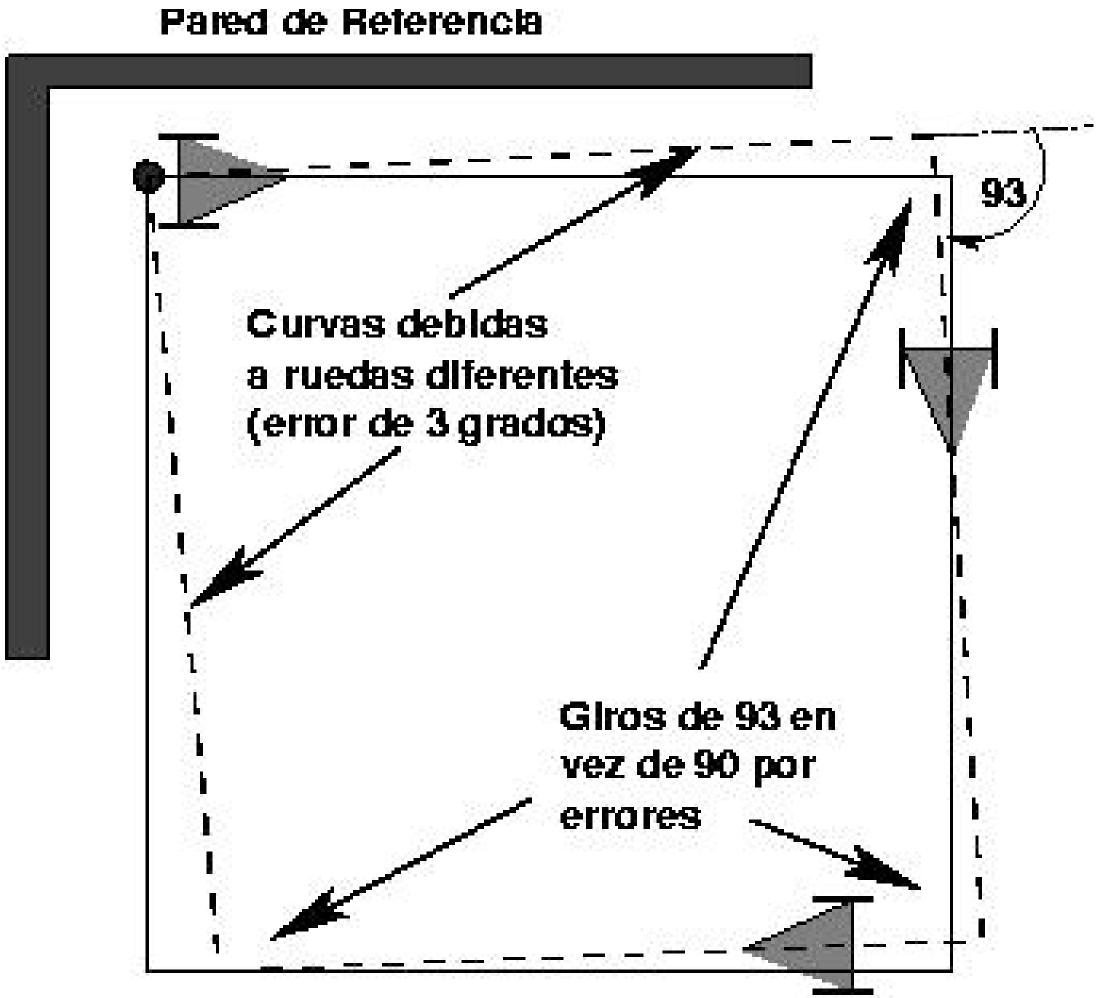
## Medida: El test del cuadrado unidireccional

- Comienzo en $(x_0, y_0, \theta_0)$, junto a una pared
- El robot debe recorrer un cuadrado (p.e. 4x4 metros)
- Al final se miden las distancias de tres puntos del robot a las paredes
- Se obtiene un error $E_x, E_y, E_\theta$
- El error puede deberse:
  - Diferentes diámetros (curva)
  - Error de giro
- Otra solución es "la figura del 8" (Tsumra, 1981)

## El test del cuadrado unidireccional



## Problemas con el cuadrado unidireccional



## Mejoras al cuadrado unidireccional

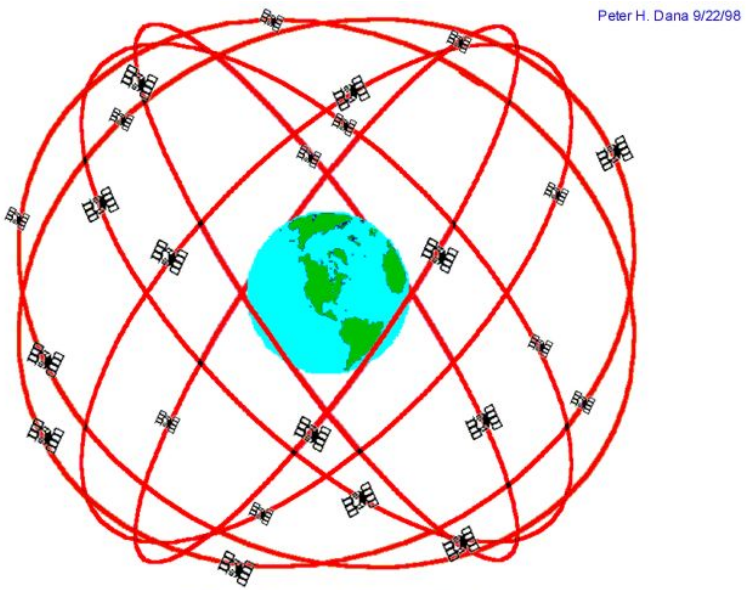
Borenstein: El cuadrado bidireccional (UMBark):

- Evita que ciertos errores se compensen
- Ahora hay que recorrer el cuadrado en los dos sentidos
- Es conveniente repertirlo varias veces
- Cuidado con las medias (mejor centro de gravedad)
- El error en orientación influye

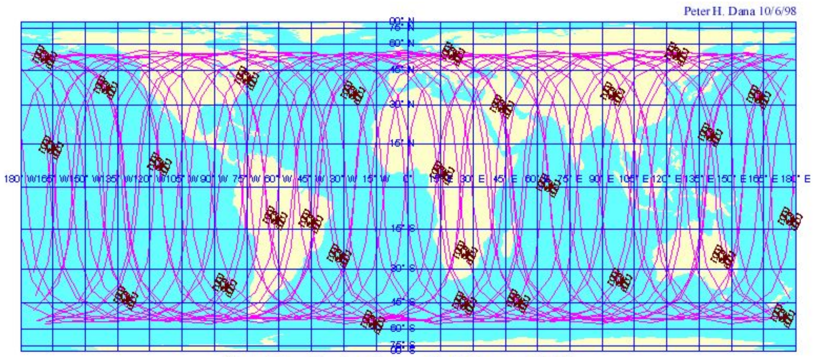
## Corrección del error odométrico

- Error sistemático:
  - Ruedas auxiliares (sin peso)
  - *encoders* adicionales (carrito)
  - Calibración sistemática: usando el UMBark p.e.
- Error no sistemático:
  - Referencias mutuas: dos robots, uno parado
  - Corrección interna: dos robots en movimiento
  - Navegación inercial: medir aceleración en los tres ejes e integrar (0.1 % de la distancia viajada) Muy caros

# El sensor GPS (Global Positioning System)




**GPS Nominal Constellation**
**24 Satellites in 6 Orbital Planes**
**4 Satellites in each Plane**
**20,200 km Altitudes, 55 Degree Inclination**




Global Positioning System Satellites and Orbits
for 27 Operational Satellites on September 29, 1998
Satellite Positions at 00:00:00 9/29/98 with 24 hours (2 orbits) of Ground Tracks to 00:00:00 9/30/98

- Se necesitan 4 satélites para obtener posición (X,Y,Z) y tiempo
- `http://www.colorado.edu/geography/gcraft/notes/gps/gps_f.html`

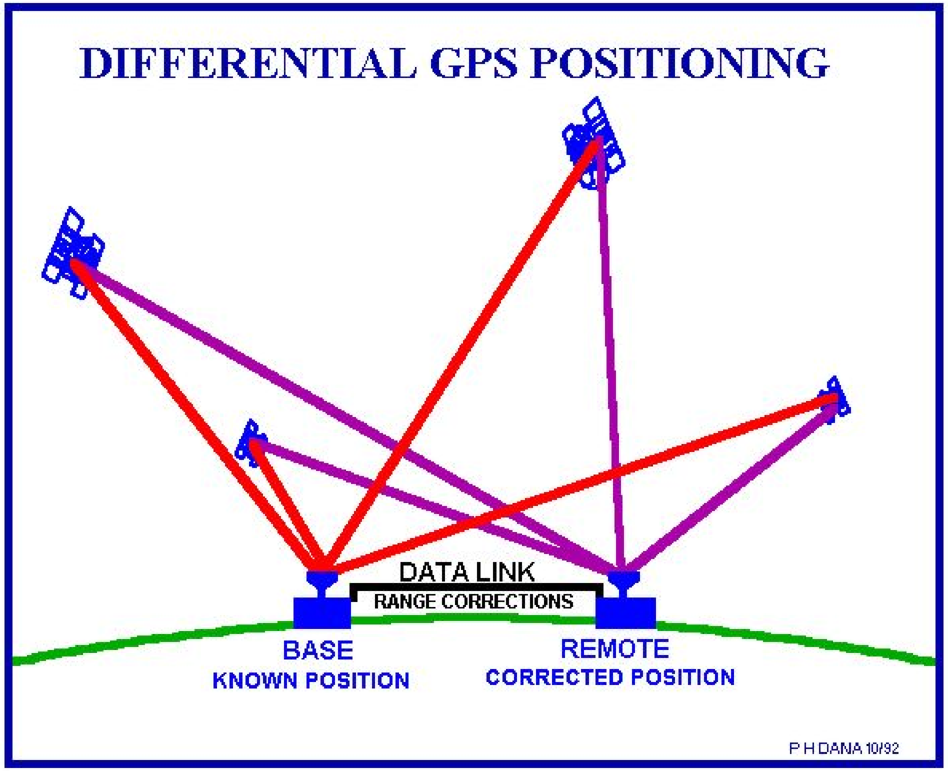
## El sensor GPS (Global Positioning System)

- Las estaciones en tierra actualizan la señal de los satélites (hora y órbita)
- *Precise Positioning Service* (PPS) cifrado: 22 metros horizontal, 27.7 vertical y 200 nanosegundos tiempo (UTC)
- *Standard Positioning Service* (PPS): 100 metros horizontal, 156 vertical y 340 nanosegundos tiempo (UTC)
- Los satélites envían *frames* de 1500 bytes (5 *subframes* de 300)
- Los satélites envían un *frame* cada 30 segundos.
- Los *frames* llevan diferentes informaciones (id. del satélite, la hora, etc.)

## Fuentes de error del GPS

- Degradación intencional del DoD (diferente para cada satélite)
- Errores del reloj (1 metro como máximo)
- Errores en la órbita (1 metro como máximo)
- Errores en la modelización de la propagación en la troposfera (8-13km): presión, temperatura, etc. (1 metro)
- Errores en la modelización de la propagación en la ionosfera (50-500km): (10 metros)
- Errores debidos a rebotes de la señal (0.5 metro)
- Errores de usuario, de 1m a cientos de kilómetros (más usuales):
  - Errores en el receptor
  - Errores en la configuración (zona horaria, fecha, etc.)

## GPS Diferencial




Corregir la desviación del robot en función de la desviación recibida en una posición conocida. Esa información se transmite al robot (FM en Madrid)

## GPS en robótica

- Útil para exteriores
- Más interesante el GPS diferencial
- Se suele usar como un sensor más
- Se conecta generalmente al puerto serie
- También GPS en tarjetas PCMCIA, de mano,...
- Suele fusionarse con otros sensores
- Foto: icady (*Cady* de golf):

Anillo sonar

GPS (antena)

## Galileo

`http://www.esa.int`

- Proyecto europeo equivalente al GPS
- No existe todavía (operativo en el 2008)
- 30 satélites
- Uso civil: no tendrá la señal degradada, no lo “apagan”
- Cubrirá todo el planeta (p.e. los polos)
- El uso combinado con GPS permitirá mucha mayor precisión
- GLONASS es el equivalente ruso

©2007 GSyC

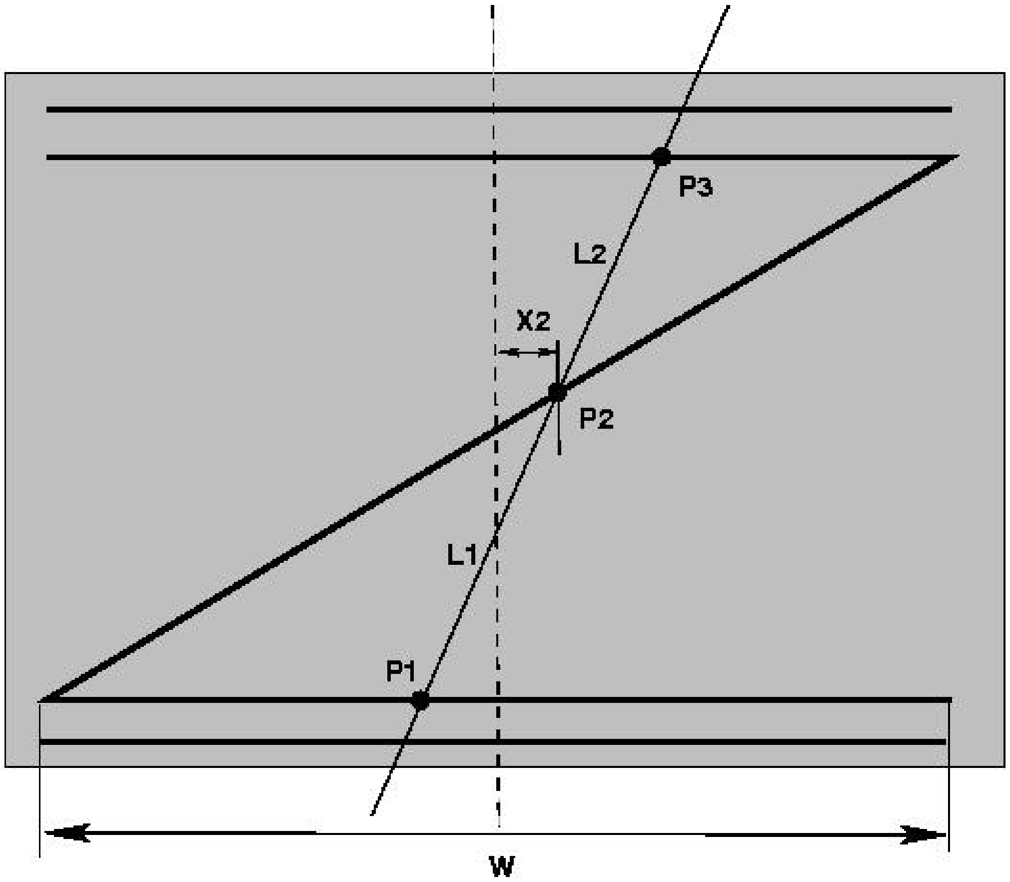
# Otros sensores

- En determinados entornos se pueden usar sensores específicos
- Ejemplo: El campo de la RoboCup y una cámara cenital:

# Balizamiento

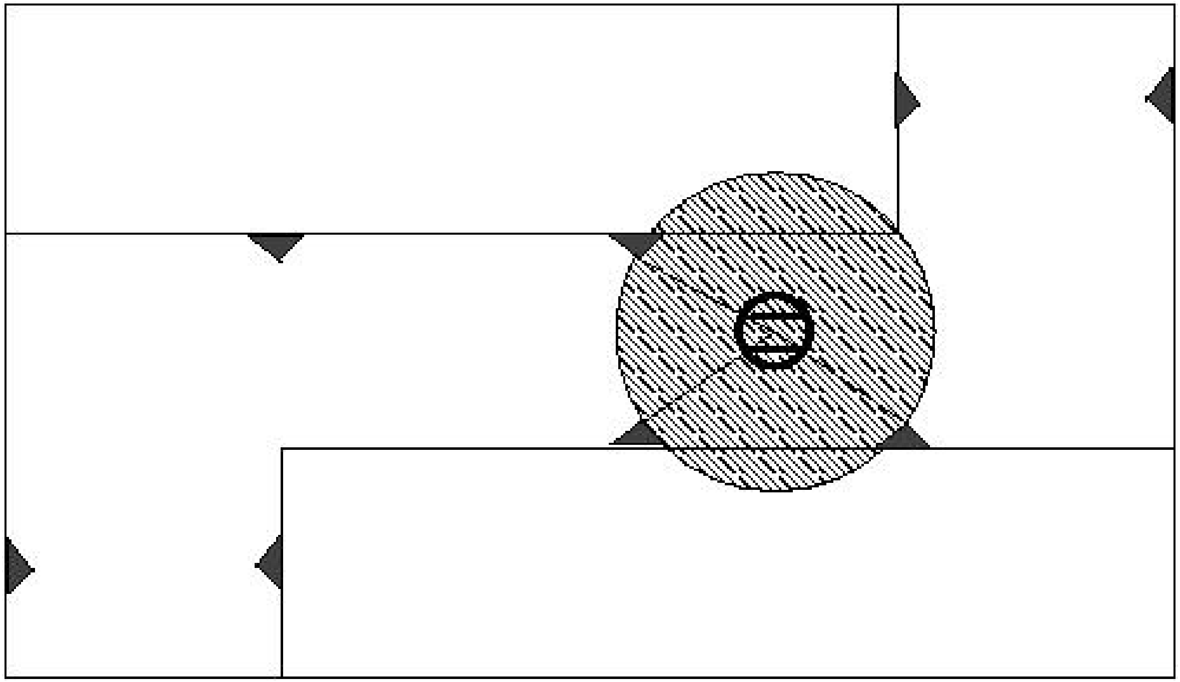
- Triangulación: Cálculo de $(x, y, \theta)$ basado en el ángulo con el que se “ven” las balizas
- Trilateración: Cálculo de $(x, y, \theta)$ basado en las distancias a las balizas (El GPS es de este tipo)
- ¿Qué elemento es activo?
  - Emisor en el robot: pocos robots
  - Balizas emisoras: incontrolable por el robot
- Marcas visuales (códigos de barras, colores, etc.) en paredes o en el suelo
- Múltiples sistemas: infrarrojos, ultrasonidos, láser, etc.

## Ejemplo de marca en el suelo




$$X_2 = W\left(\frac{L_1}{L_1+L_2} - \frac{1}{2}\right)$$

## Ejemplo de localización con balizas

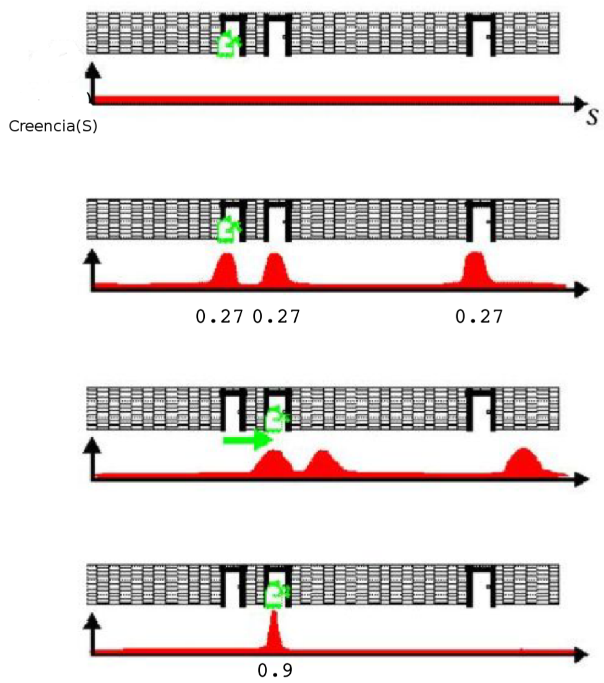


# Localización probabilística

- Problema de la localización global (error sin acotar)
- Problema del secuestro (recuperación de errores)
- Diferentes aproximaciones a lo localización probabilística:
  - Markov (Simmons y Koening, 1995)
  - Filtros de partículas (Doucet, 1998)
  - Condensación (Isard, 1998)
  - Montecarlo (Thurn, 1999)

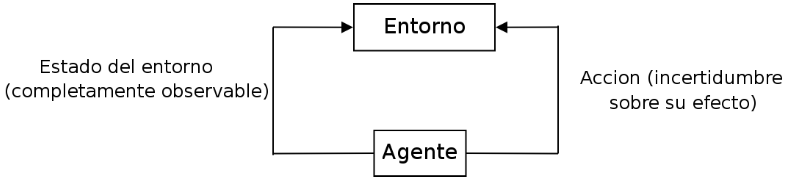
©2007 GSyC

# Introducción




- Se mantiene la probabilidad $p(s_i)$ de estar en alguno de los estados $s_i \in S$
- Después de cada acción se modifica esta probabilidad.

# Procesos de Decisión de Markov




- Tenemos incertidumbre de que una acción produzca el cambio deseado en el entorno.
- dependiendo de la información que tengamos del entorno tenemos

  **Procesos de Decisión de Markov Observables** Después de realizar cada acción podemos saber el estado del entorno.

  **Procesos de Decisión de Markov Parcialmente Observables** No sabemos el estado real del entorno, sino indicios (observaciones).

## Elementos de un Proceso de Decisión de Markov Observable

- $S$ es el conjunto de estados.

$$\mathbf{S} = \{s_0, s_1, s_2, ..., s_n\}$$

- $A$ es el conjunto de acciones que puede ejecutar una agente

$$\mathbf{A} = \{a_0, a_1, a_2, \ldots, s_k\}$$

- $T$ es la función de transición

$$T : S \times A \longrightarrow p(S)$$

para cada estado final $s'$, un estado inicial $s$ y una acción $a$, determina $p(s'|s, a)$

## Procesos de Decisión de Markov Parcialmente Observables

- Es parcialmente observable porque el agente no puede determinar con exactitud el estado actual del entorno.
- Recibe observaciones que le dan una idea de cual es el estado actual, que son modeladas mediante una *función de observación* $\vartheta = \{o_1, o_2, \cdot\}$.
- Puesto que varias estados pueden producir la misma observación, es necesario conocer la distribución de probabilidad $Creencia(S)$ para reducir la ambigüedad de las observaciones.

$$\mathbf{Creencia(S)_t} = \{p(s_0)_t, p(s_1)_t, \ldots, p(s_n)_t\}$$

$$\sum_{i=0}^{n} Creencia(s_i) = 1$$

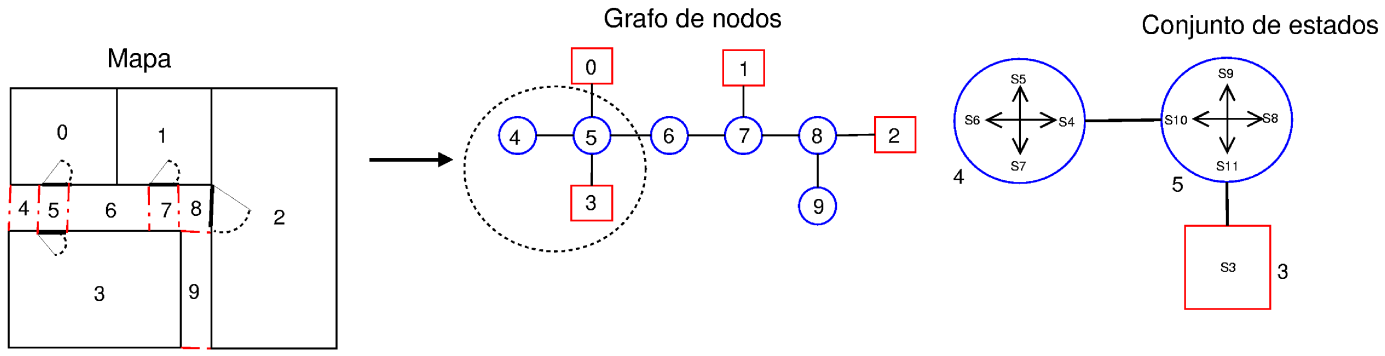
# Un Ejemplo: localización en un pasillo

Vamos a definirlo como un POMDP. Necesitamos los siguientes elementos:

- Estados $S$, que representa el estado del agente con respecto al entorno.
- Acciones $A$ que podemos realizar sobre el entorno para cambiar su estado.
- Función de transición $T$ que modela la incertidumbre del cambio de estado.
- Conjunto de observaciones $\vartheta$ que nos da indicios de cual es estado del entorno.

©2007 GSyC 

## Conjunto de estados $S$




- Usamos un modelo topológico del entorno.
- Representamos las habitaciones y zonas del pasillo mediante nodos.

  **Nodo de habitacion** Corresponde a un estado.

  **Nodo de pasillo** Corresponde a 4 estados claramente separados, según la orientación del robot.
- $S = \{s_0, s_1, s_2, ..., s_{29}\}$

# Conjunto de acciones$A$

Conjunto de acciones $A = \{a_r, a_l, a_f, a_o, a_e\}$

**Acción** $a_r$ *Girar 90º a la derecha*. En un estado en un pasillo, transita idealmente al nodo de pasillo situado a la derecha, en el estado con la misma orientación.

**Acción** $a_l$ *Girar 90º a la izquierda*. En un estado en un pasillo, transita idealmente al nodo de pasillo situado a la izquierda, en el estado con la misma orientación.

**Acción** $a_f$ *Seguir pasillo*. En un estado en un pasillo, transita idealmente al siguiente nodo del pasillo cuya orientación coincide con la del estado inicial.

**Acción** $a_o$ *Salir de habitación*. En estados de habitación, realiza el conjunto de acciones a menor nivel destinadas a salir de una habitación.

**Acción** $a_e$ *Entrar en habitación*. En estados de nodos de pasillos con orientación hacia un nodo de habitación, realiza el conjunto de acciones a menor nivel destinadas a entrar de una habitación.

## Función de transición $T$

- Partimos de un modelo general de incertidumbre por acciones:

  **Acción** $a_r$ p(No hace nada)=0.05, p(gira a la dcha 90°)=0.90, p(gira a la derecha más de 90°)=0.05

  **Acción** $a_l$ p(No hace nada)=0.05, p(gira a la izda 90°)=0.90, p(gira a la izquierda más de 90°)=0.05

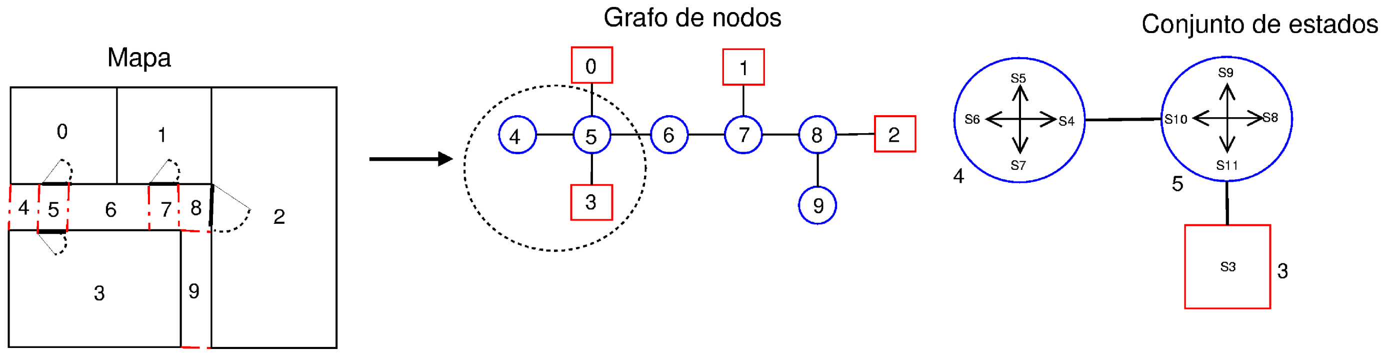
  **Acción** $a_f$ p(No hace nada)=0.10, p(Avanza lo adecuado para llegar al siguiente estado)=0.70, p(Avanza más de lo deseado)=0.15, p(Avanza mucho más)=0.05

  **Acción** $a_o$ p(No hace nada)=0.05, p(sale de la habitacion)=0.85, p(Sale de la habitaciñon y avanza más de lo deseado)=0.10

  **Acción** $a_e$ p(No hace nada)=0.10, p(Entra en la habitación)=0.90

- A partir de ahí creamos la función de transición por estados, que modelamos como una tabla por cada acción, que indica la probabilidad de llegar del estado $s$ a $s'$ con esa acción

Tabla de transiciones para la acción de avanzar pasillo:



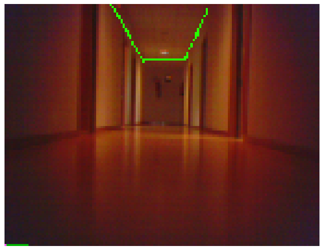

| $p$ | $\cdots$ | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | $\cdots$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $origen\mathbf{4}$ | $\cdots$ | 10 | 0 | 0 | 0 | 70 | 0 | 0 | 0 | 15 | 0 | $\cdots$ |
| $origen\mathbf{5}$ | $\cdots$ | 0 | 100 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $\cdots$ |
| $origen\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $origen\mathbf{8}$ | $\cdots$ | 0 | 0 | 0 | 0 | 10 | 0 | 0 | 0 | 70 | 0 | $\cdots$ |

## Función de observación $\vartheta$

- Modelamos las probabilidades de obtener una observación estando en un estado determinado, es decir, $p(\mathbf{o}|s)$
- Tendremos varios tipos de observaciones

**OVPuertas** Detectaremos el nº de puertas que hay en una imagen.

**OVProfundidad** Usando el trapecio que visualmente se percibe al detectar el techo, detectaremos la cercanía con el final del pasillo.

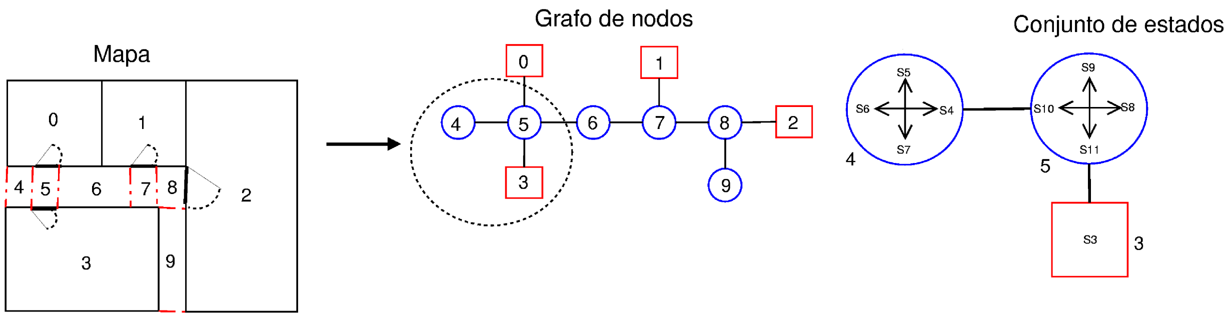


- A partir del vector que nos proporcionan las observaciones y del conocimiento del entorno, podemos construir una tabla de probabilidades para cada observación relacionando el estado con cada uno de los posibles valores de cada tipo de observación.

- Al ser las observaciones independientes, tenemos que podemos descomponer:

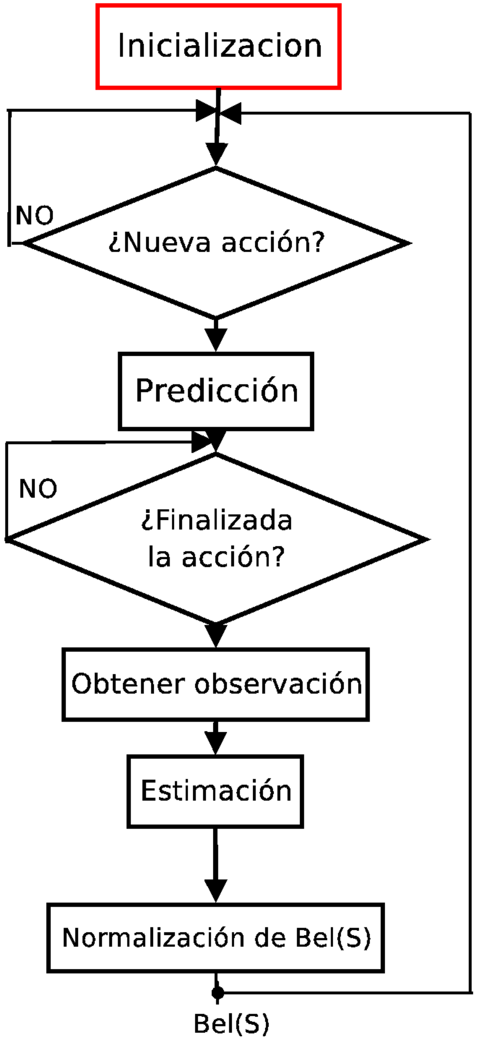
$$p(\mathbf{o}|s) = p(o_{OV\,Puertas}, o_{OV\,Profundidad}|s) = p(o_{OV\,Puertas}|s) \cdot$$

$$p(o_{OV\,Profundidad}|s)$$




Ejemplo de matriz de $p(o_{OVProfundidad}|s)$

| $p$ | $\mathbf{o_{ovp0}}$ | $\mathbf{o_{ovp1}}$ | $\mathbf{o_{ovp2}}$ |
|---|---|---|---|
| ... | ... | ... | ... |
| $\mathbf{s_4}$ | 3 | 7 | 90 |
| $\mathbf{s_5}$ | 95 | 4 | 1 |
| $\mathbf{s_8}$ | 5 | 15 | 80 |
| $\mathbf{s_{10}}$ | 85 | 10 | 5 |

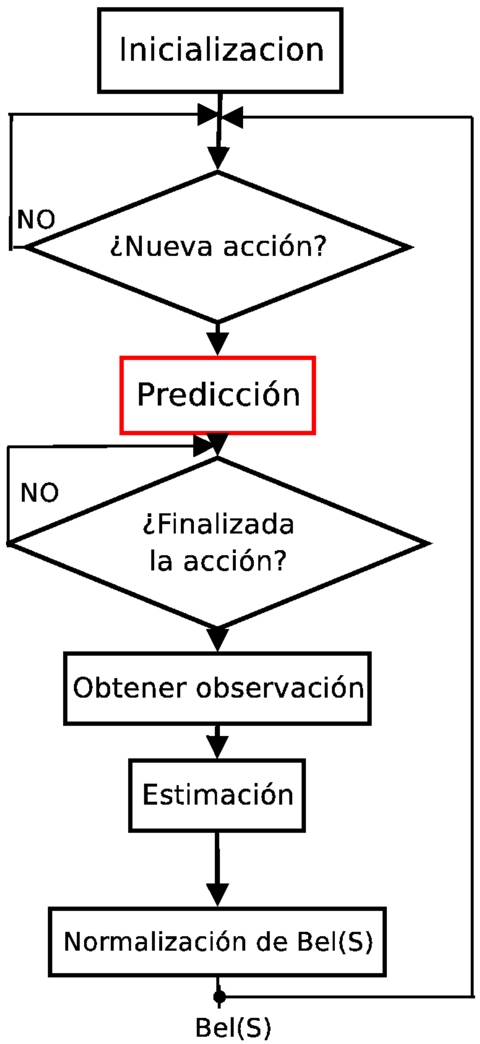



- Si desconocemos donde estamos,

$$Creencia(s) = \frac{1}{n^o deestados}, \forall s \in S$$

- Si sabemos donde estamos,

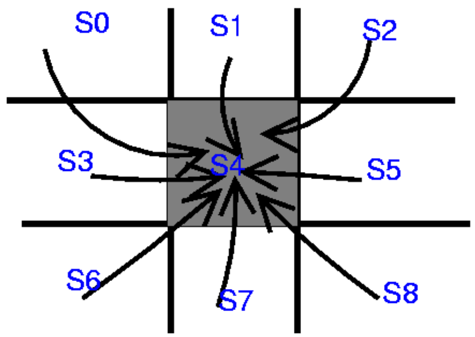
$$Creencia(s) = \begin{cases} 1 & s = s_{ini} \\ 0 & s \neq s_{ini} \end{cases}$$

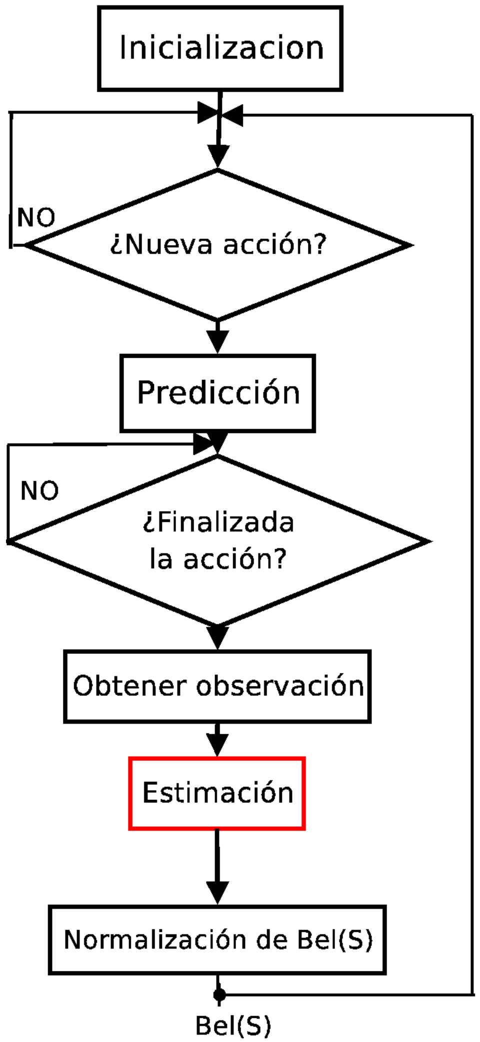



- Actualizamos la creencia $Creencia(S')$ después de haber realizado una acción

$$Creencia_t(S') = \sum_{s \in S} p(s'|s,a) \cdot Creencia_{t-1}(s), \forall s' \in S$$

- $p(s'|s,a)$ lo tenemos calculado en forma de tabla
- $Creencia_{t-1}(s)$ es la creencia en el instante de tiempo anterior.
- La fórmula indica el cálculo de la creencia para cada estado $s$, que depende del resto de estados.






- Corregimos $Creencia(S')$ con los "indicios", es decir, las observaciones, que tenemos del entorno- después de haber realizado una acción

$$\mathbf{Creencia}_{\mathbf{posterior}}(\mathbf{s}) = p(\mathbf{o}|s) \cdot Creencia_{anterior}(s), \forall s \in S$$

$$\mathbf{Creencia}_{\mathbf{posterior}}(\mathbf{s}) = p(o_{OVPuertas}, o_{OVProfundidad}|s) \cdot$$
$$\cdot Creencia_{anterior}(s), \forall s \in S$$

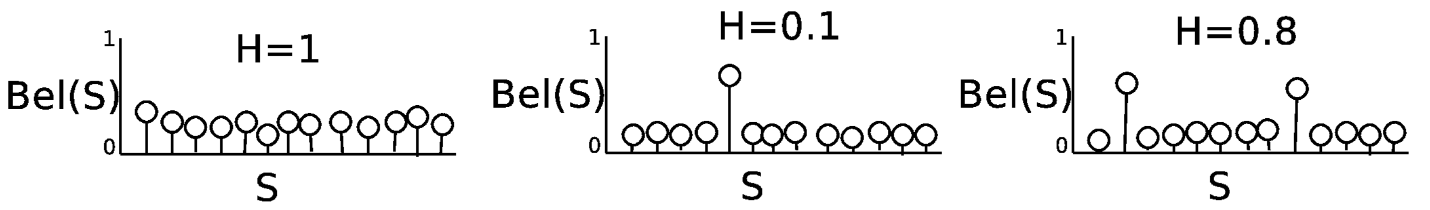
$$\mathbf{Creencia}_{\mathbf{posterior}}(\mathbf{s}) = p(o_{OVPuertas}|s) \cdot p(o_{OVProfundidad}|s) \cdot$$
$$\cdot Creencia_{anterior}(s), \forall s \in S$$

# Interpretacion de $Creencia(S)$

- Podemos calcular la incertidumbre sobre la posición del robot mediante el cálculo de la *entropía normalizada*:

$$\widetilde{H} = -\frac{\sum_{Creencia(s)\neq 0} Creencia(s) \cdot log(Creencia(s))}{log(m)}$$

- Si $\widetilde{H}$ es $0$, no hay incertidumbre sobre la posición del robot, y el valor mayor de $Creencia(s_i), 0 \leq i < n$ indicará que el robot está en el estado $s_i$.
- Si $\widetilde{H}$ es $1$, los valores de $Creencia(s_i), 0 \leq i < n$ son iguales, con lo que no tenemos ninguna informacion de posición.




©2007 GSyC 