Localización

Dr José Martinez
Carranza
carranza@inaoep.mx

Coordinación de Ciencias Computacionales,
INAOE

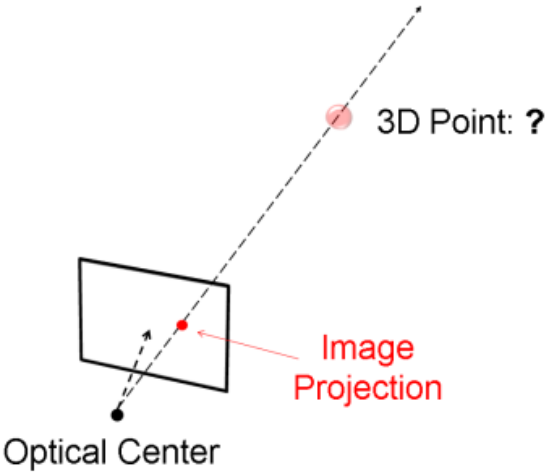
Algoritmos

- Essential/Fundamental Matrix
- 3 Point Pose Estimation
  https://www.uio.no/studier/emner/matnat/its/nedlagte-emner/UNIK4690/v16/forelesninger/lecture_7_3-pose-from-epipolar-geometry.pdf

- Neural Localisation (PoseNet)
- Iterative Closest Point (ICP)
  https://cs.gmu.edu/~kosecka/cs685/cs685-icp.pdf


Fire-i

(a)


3D Point: ?
Image
Projection
Optical Center

(b)

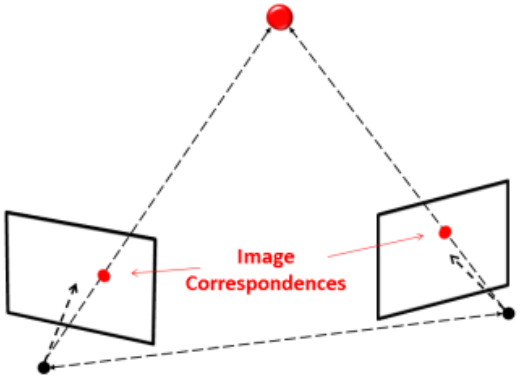
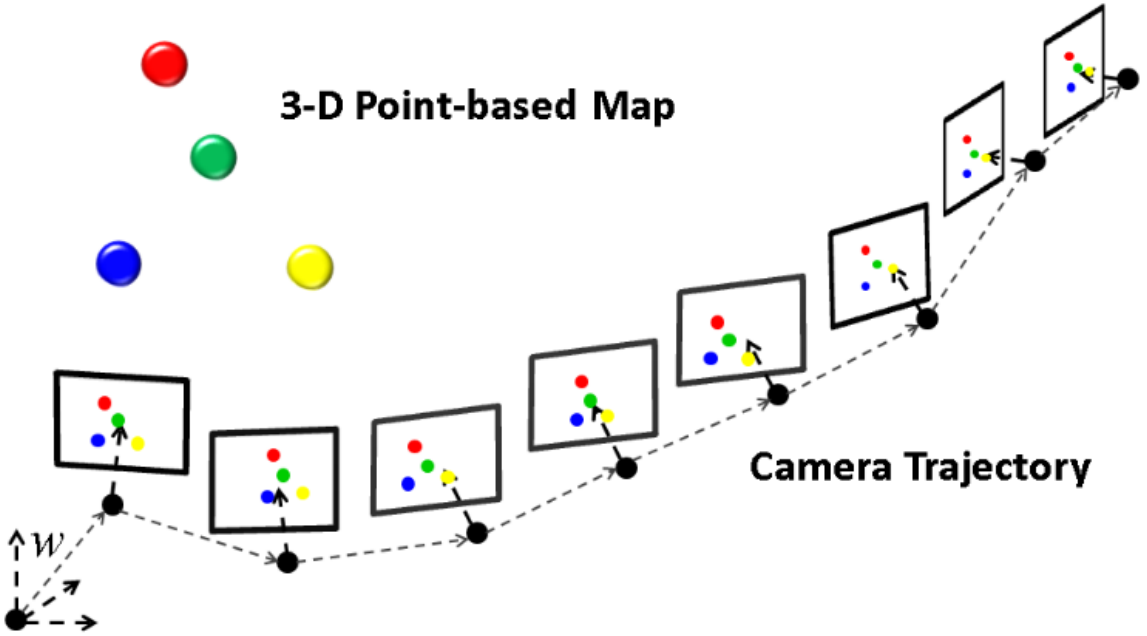
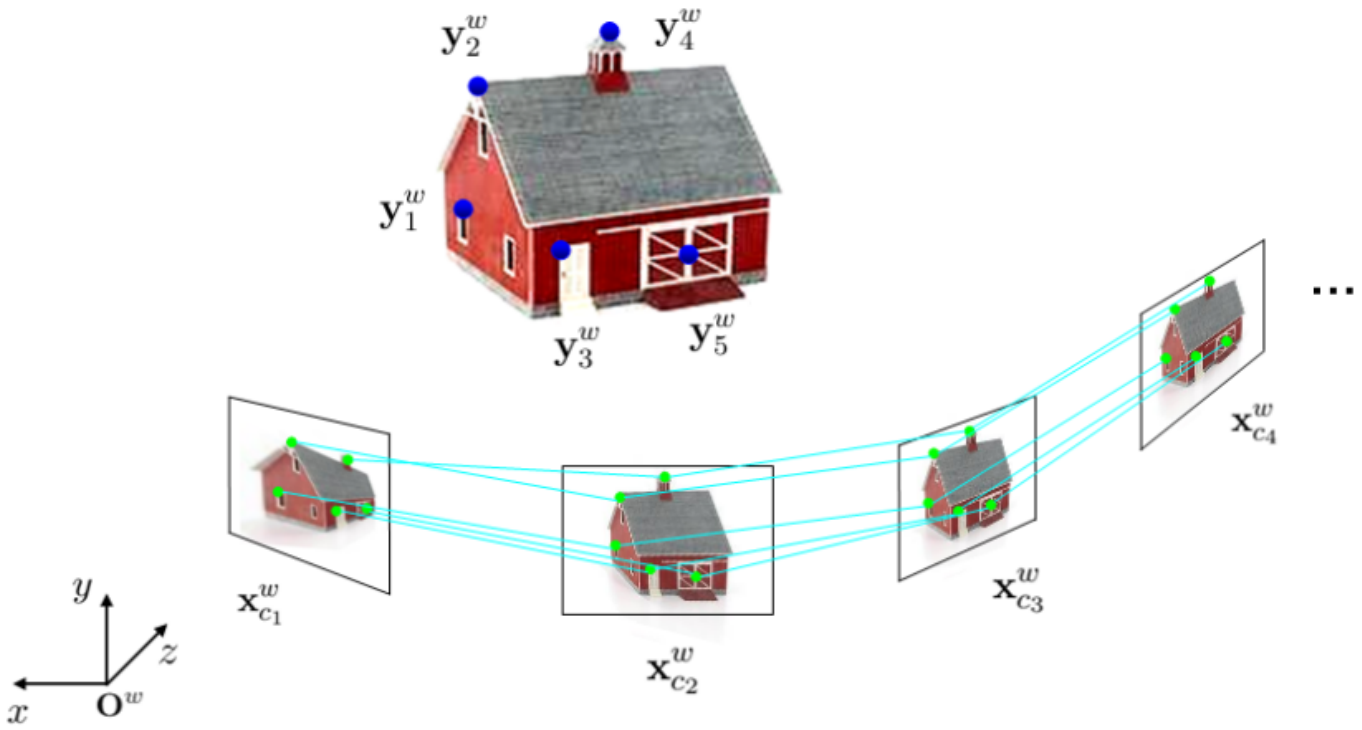


Image
Correspondences

(c)

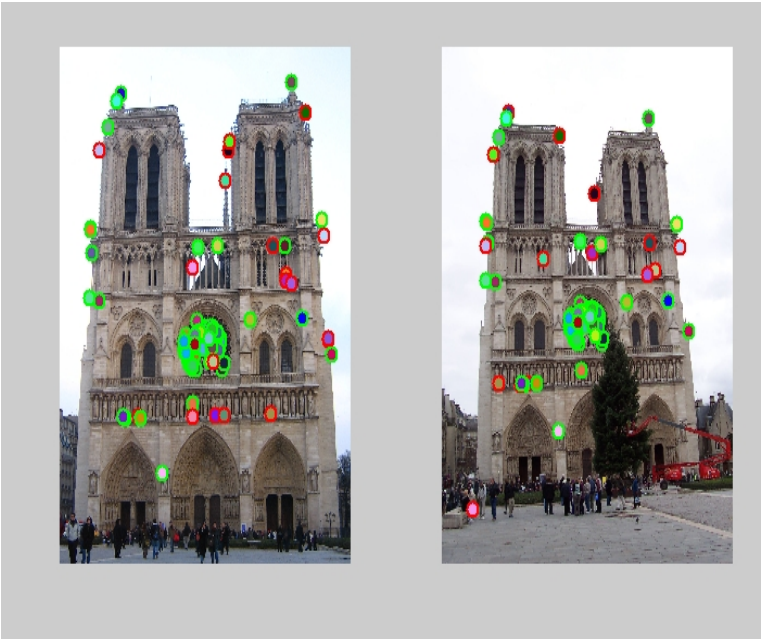

DO NOT RIDE
ON CONVEYOR

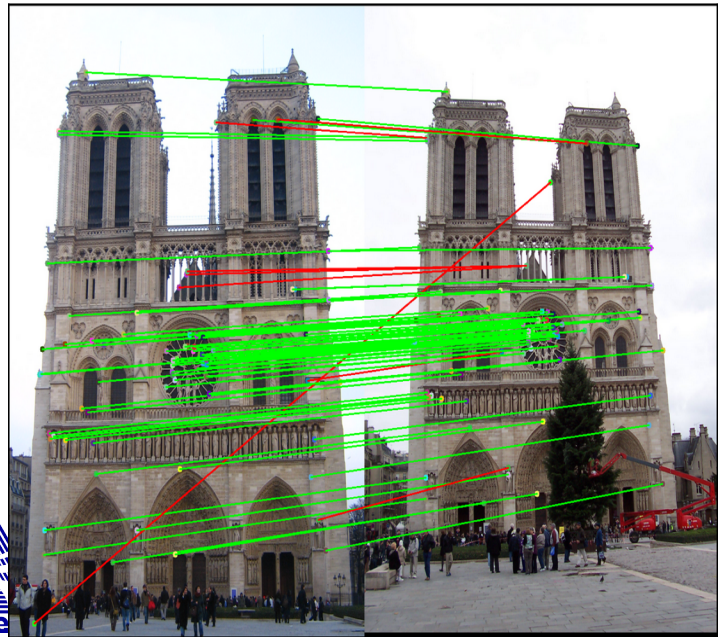

3-D Point-based Map
Camera Trajectory
W

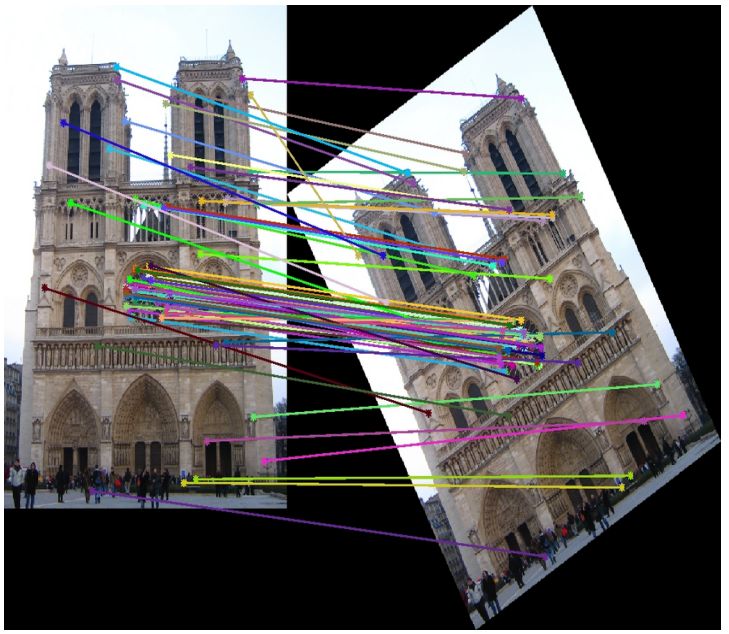

$\mathbf{y}_2^w$
$\mathbf{y}_4^w$
$\mathbf{y}_1^w$
$\mathbf{y}_3^w$
$\mathbf{y}_5^w$
...
$\mathbf{x}_{c_1}^w$
$\mathbf{x}_{c_2}^w$
$\mathbf{x}_{c_3}^w$
$\mathbf{x}_{c_4}^w$
y
z
x
$\mathbf{O}^w$

Descriptor Visual

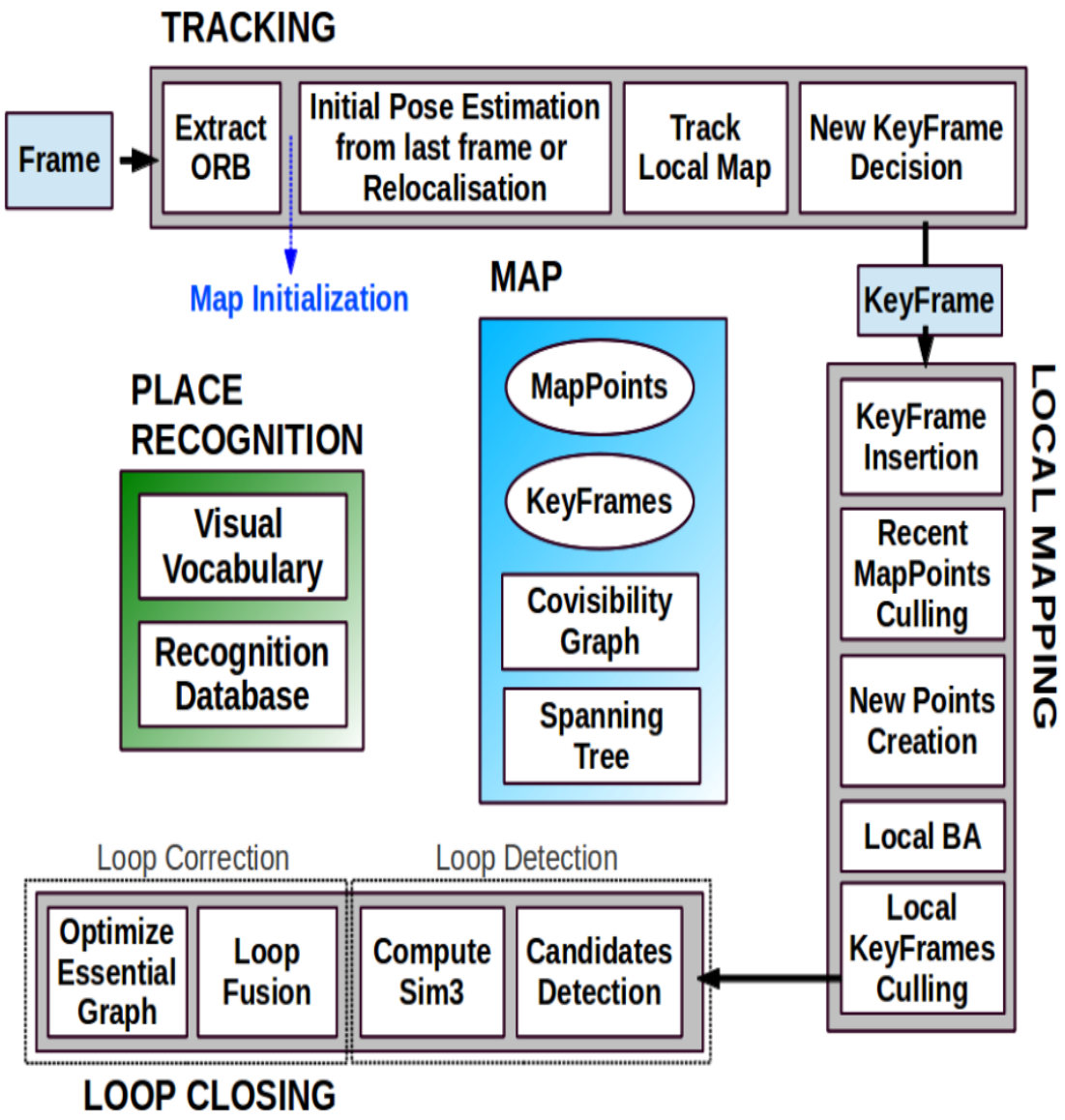
- SIFT
- Histograma de gradientes
- Descriptores binarios
  - BRIEF
  - BRISK
  - ORB







Reconocimiento de lugares previamente visitados = Relocalización



Estimación de Posición / Relocalización utilizando Redes Neuronales

POSENET [1]

Posenet: A **convolutional network** for **real-time 6-dof camera relocalization**

A Kendall, M Grimes, R Cipolla - Proceedings of the IEEE ..., 2015 - openaccess.thecvf.com

... **real-time** monocular six degree of freedom **relocalization** system. Our system trains a **convolutional neural network** to regress the **6-DOF cam**... and outdoors in **real time**, taking 5ms per ...

☆ Save 99 Cite Cited by 2138 Related articles All 22 versions »

Posenet: A convolutional network for real-time 6-dof camera relocalization

A Kendall, M Grimes, R Cipolla - Proceedings of the IEEE ..., 2015 - openaccess.thecvf.com

We present a robust and real-time monocular six degree of freedom relocalization system. Our system trains a convolutional neural network to regress the 6-DOF camera pose from a ...

☆ Save 99 Cite Cited by 2599 Related articles All 23 versions »

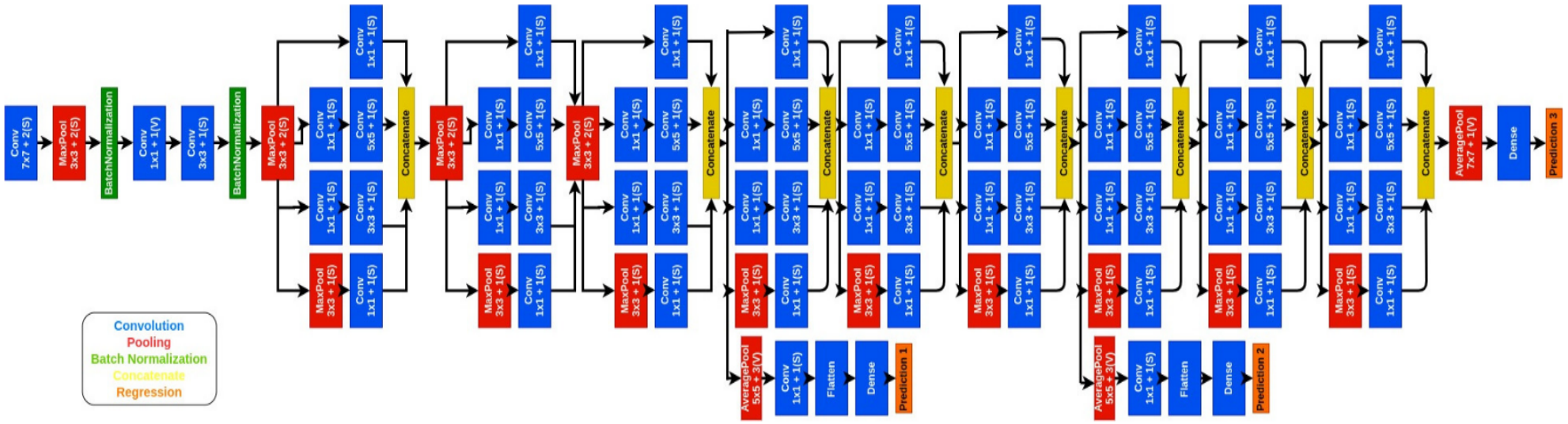
Regresión sobre la pose de la cámara

$$\mathbf{p} = [\mathbf{x}, \mathbf{q}]$$

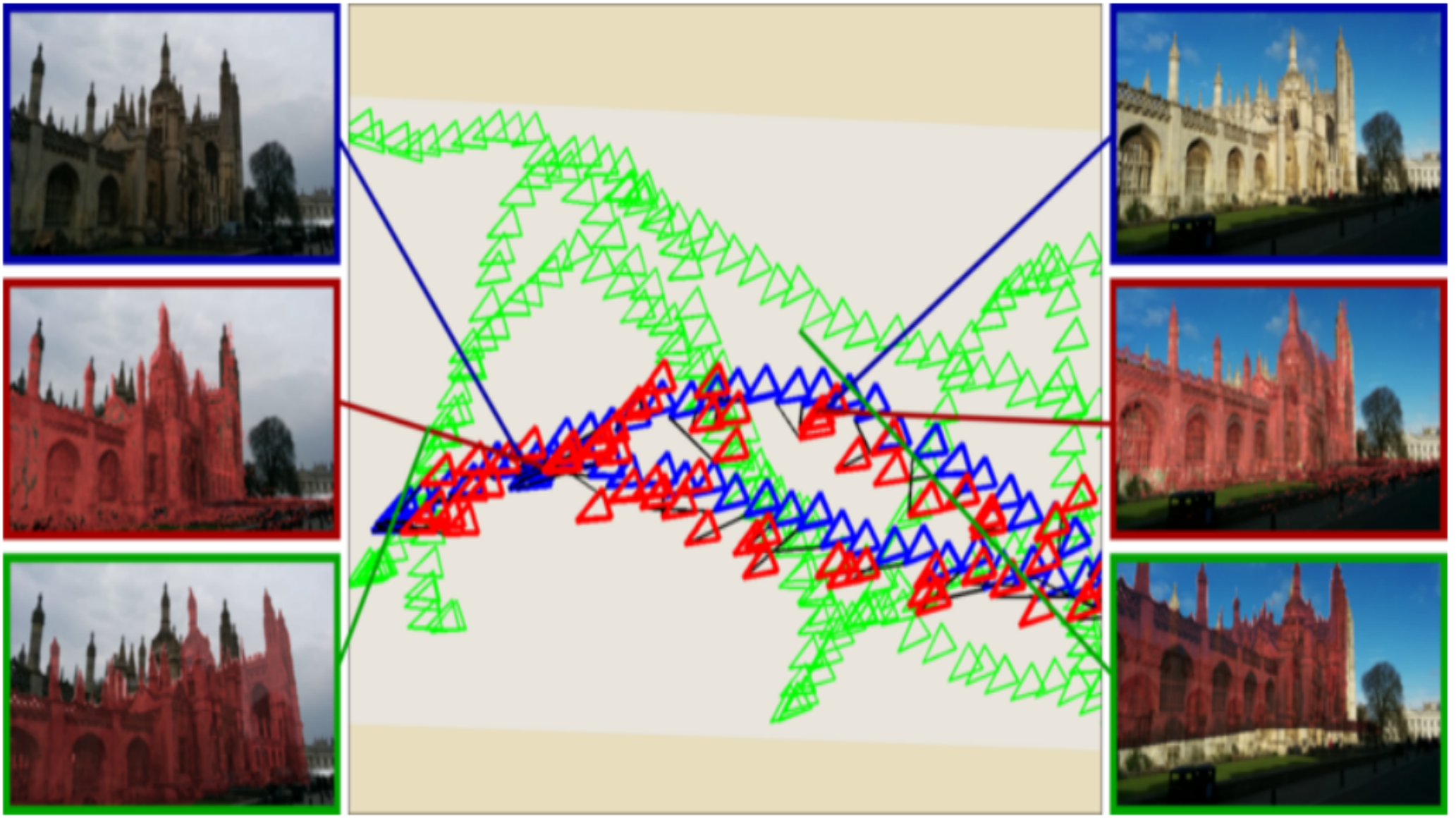
Función de pérdida

$$loss(I) = \|\hat{\mathbf{x}} - \mathbf{x}\|_2 + \beta \left\| \hat{\mathbf{q}} - \frac{\mathbf{q}}{\|\mathbf{q}\|} \right\|_2$$

Red basada en GoogLeNet [2]



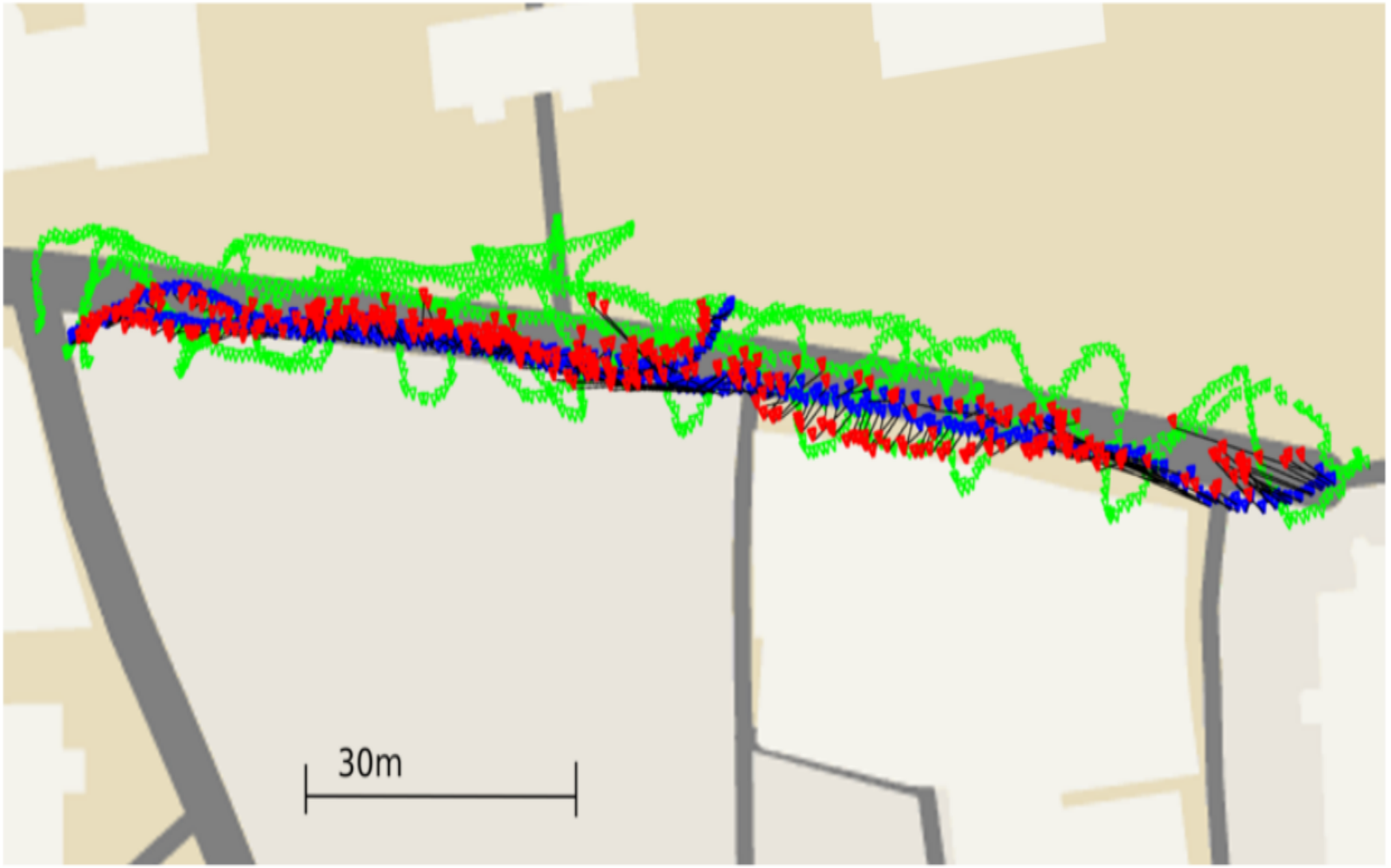
Red basada en GoogLeNet [2]



Tiempo de ejecución (GPU)

- **Normal: Imagen de entrada es reducida a 224x244.**
  - 5 ms => 200 fps

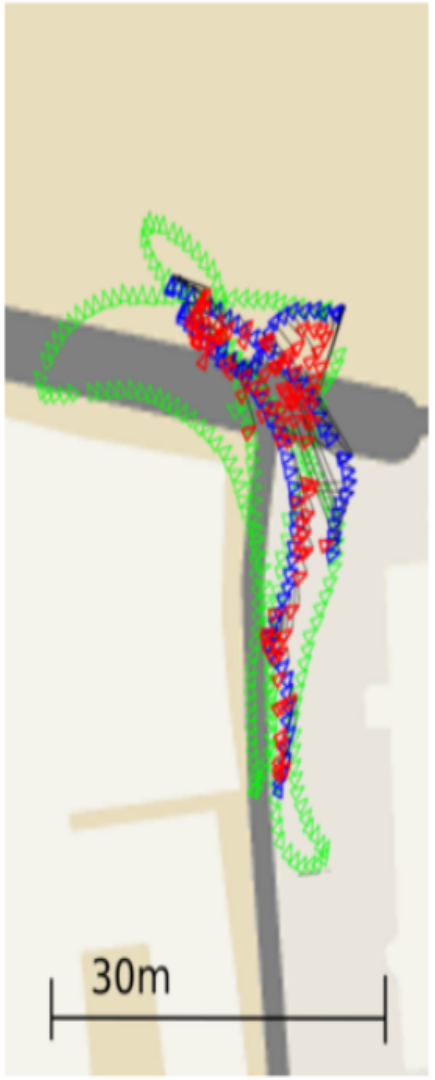
- **Densa: 128 recortes de 224x224 muestreados uniformemente sobre la imagen original.**
  - 95 ms = > 10.5 fps



King's College

Street

Old Hospital



Shop Façade

St Mary's Church

7 Scenes Dataset (Microsoft [3])

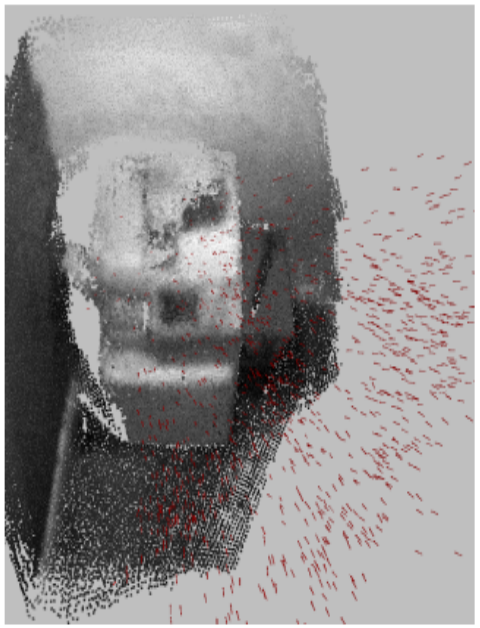
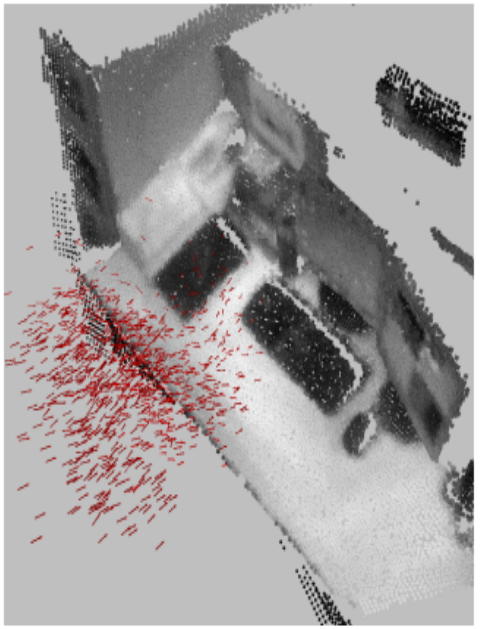
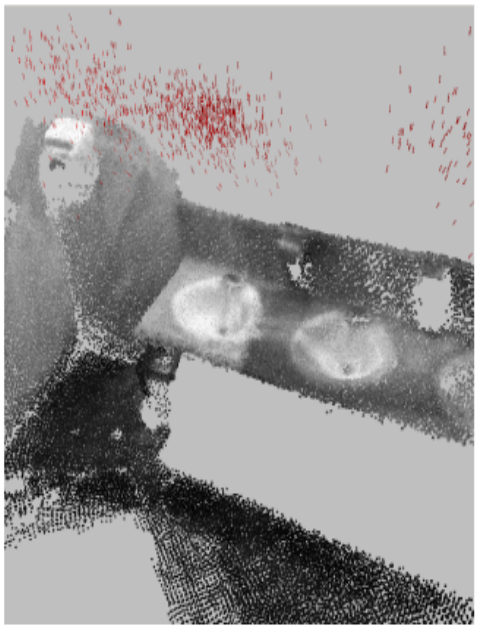
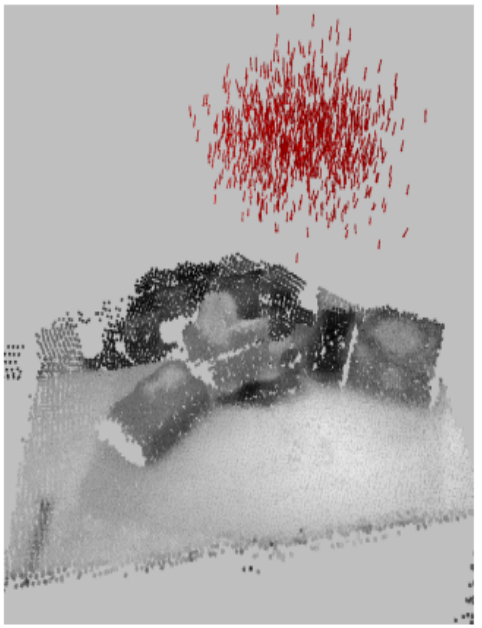
Scene	# Frames Train	# Frames Test	Spatial Extent (m)	SCoRe Forest (Uses RGB-D)	Dist. to Conv. Nearest Neighbour	PoseNet	Dense PoseNet
King's College	1220	343	140 x 40m	N/A	3.34m, 5.92°	1.92m, 5.40°	1.66m, 4.86°
Street	3015	2923	500 x 100m	N/A	1.95m, 9.02°	3.67m, 6.50°	2.96m, 6.00°
Old Hospital	895	182	50 x 40m	N/A	5.38m, 9.02°	2.31m, 5.38°	2.62m, 4.90°
Shop Façade	231	103	35 x 25m	N/A	2.10m, 10.4°	1.46m, 8.08°	1.41m, 7.18°
St Mary's Church	1487	530	80 x 60m	N/A	4.48m, 11.3°	2.65m, 8.48°	2.45m, 7.96°
Chess	4000	2000	3 x 2 x 1m	0.03m, 0.66°	0.41m, 11.2°	0.32m, 8.12°	0.32m, 6.60°
Fire	2000	2000	2.5 x 1 x 1m	0.05m, 1.50°	0.54m, 15.5°	0.47m, 14.4°	0.47m, 14.0°
Heads	1000	1000	2 x 0.5 x 1m	0.06m, 5.50°	0.28m, 14.0°	0.29m, 12.0°	0.30m, 12.2°
Office	6000	4000	2.5 x 2 x 1.5m	0.04m, 0.78°	0.49m, 12.0°	0.48m, 7.68°	0.48m, 7.24°
Pumpkin	4000	2000	2.5 x 2 x 1m	0.04m, 0.68°	0.58m, 12.1°	0.47m, 8.42°	0.49m, 8.12°
Red Kitchen	7000	5000	4 x 3 x 1.5m	0.04m, 0.76°	0.58m, 11.3°	0.59m, 8.64°	0.58m, 8.34°
Stairs	2000	1000	2.5 x 2 x 1.5m	0.32m, 1.32°	0.56m, 15.4°	0.47m, 13.8°	0.48m, 13.1°

6. Conclusions

We present, to our knowledge, the first application of deep convolutional neural networks to end-to-end 6-DOF camera pose localization. We have demonstrated that one can sidestep the need for millions of training images by use of transfer learning from networks trained as classifiers. We showed that such networks preserve ample pose information in their feature vectors, despite being trained to produce pose-invariant outputs. Our method tolerates large baselines that cause SIFT-based localizers to fail sharply.

In future work, we aim to pursue further uses of multiview geometry as a source of training data for deep pose regressors, and explore probabilistic extensions to this algorithm [12]. It is obvious that a finite neural network has an upper bound on the physical area that it can learn to localize within. We leave finding this limit to future work.

6D Relocalisation for RGBD Cameras Using Synthetic View Regression [4]

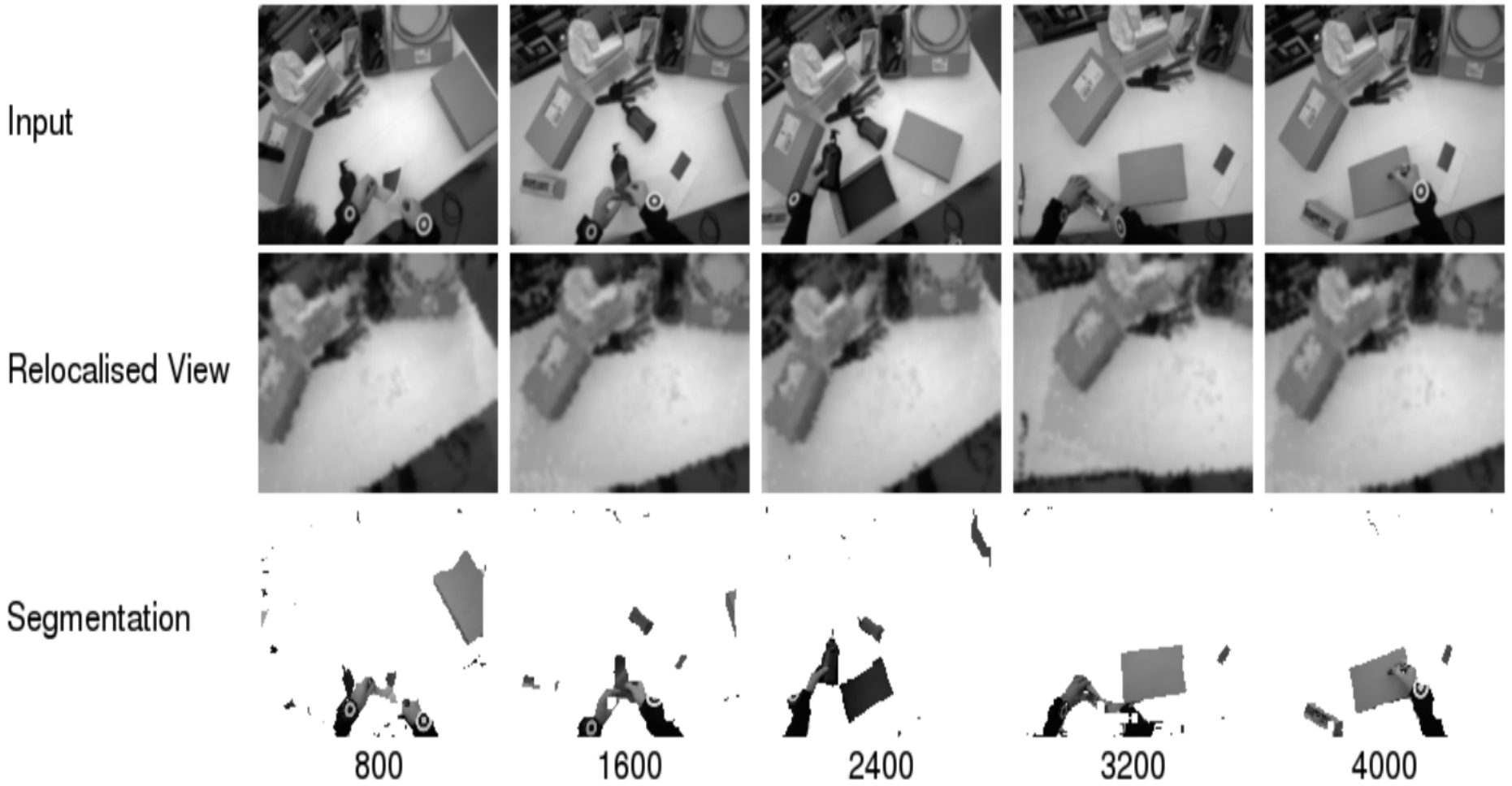





6D Relocalisation for RGBD Cameras Using Synthetic View Regression [4]

6D Relocalisation for RGBD Cameras Using Synthetic View Regression [4]

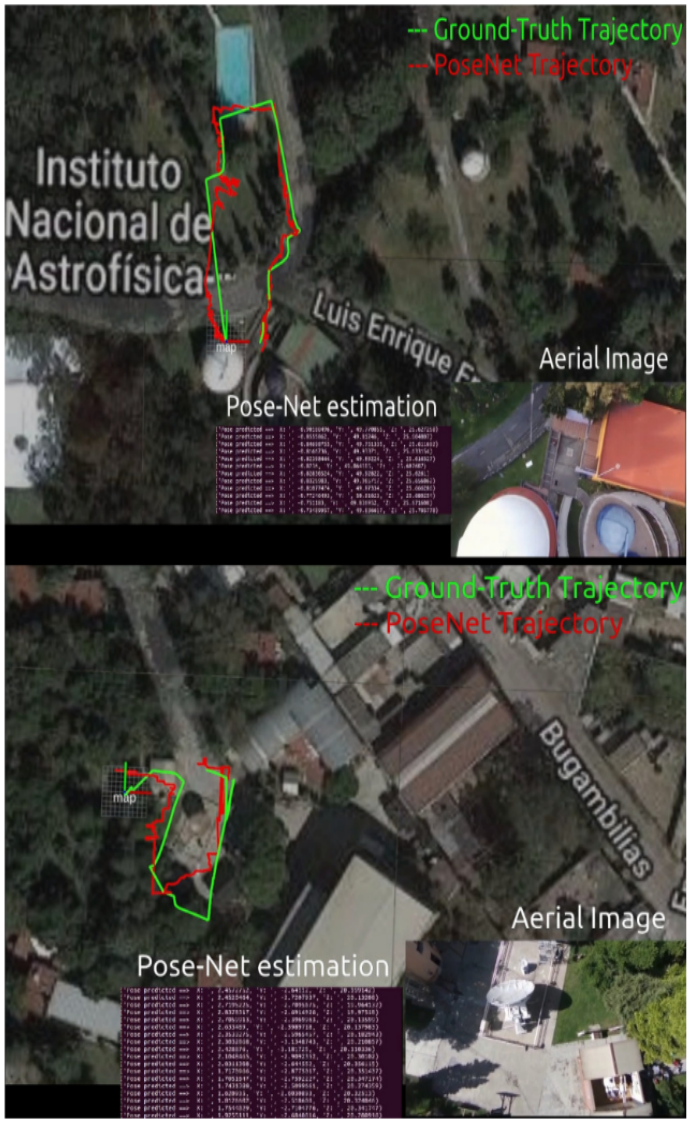
Resolution and Computational Cost Figure 3 (top row) shows the effect of resolution on relocalisation error. Although we use a resolution of 80×60 for our other results, there appears to be considerable scope to reduce resolution without significantly affecting accuracy. At a resolution of 80×60, each distance measurement between the current view and a sampled synthetic view takes $\sim 60\mu$s, enabling regression over 1000 samples in 60ms on a 2.66GHz Intel Core2 CPU. Reducing the resolution to 20×15 reduces these timings an order of magnitude to $\sim 4\mu$s and 4ms respectively, and also reduces the memory footprint for their storage substantially by about 94%. This potentially enables more samples to be stored and tested or the algorithm to be run on mobile platforms with lower computing power.

6D Relocalisation for RGBD Cameras Using Synthetic View Regression [4]



The speed and quality of the pose estimation using the synthetic views relocalisation method enables the possibility of using constant relocalisation as an alternative to conventional tracking of the camera. Wearable cameras often experience occlusions and erratic motion that cause tracking systems to fail, and a constant relocalisation approach avoids continuous switching between relocalisation and tracking.

Aerial Geo-Localisation for MAVs using PoseNet [5]

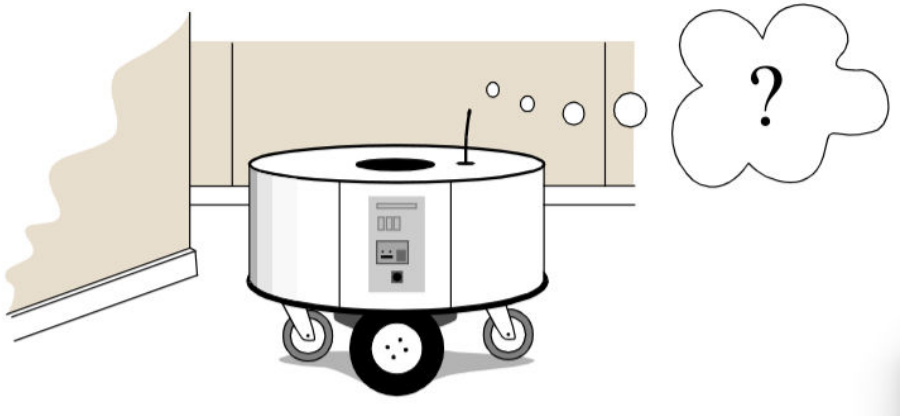


Localización y su papel en la navegación

- La navegación es una capacidad de los robots móviles que involucran mayores retos.
- La navegación le permite a un robot móvil el recorrer su ambiente de acuerdo a un plan.
- Esta capacidad involucra 4 habilidades importantes:
  - Percepción: interpretar información de los sensores.
  - **Localización**: determinar su posición en el ambiente.
  - Razonamiento: decidir qué hacer para cumplir sus metas.
  - Control: realizar las acciones.

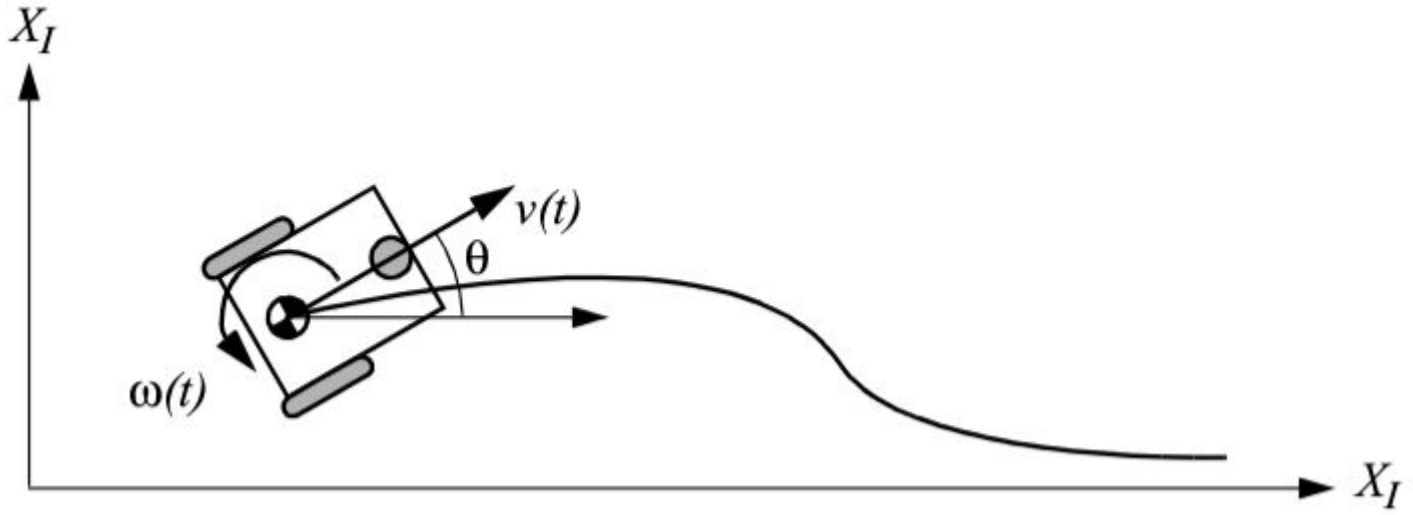
Localización

- La localización es la habilidad de un robot móvil para situarse en un ambiente .
- Saber “donde está” le sirve para decidir que hacer y construir una representación de lo que se encuentra a su alrededor (mapa).
- La localización puede realizarse de diversas maneras:
  - Utilizando mapas
  - A través de marcas en el ambiente
  - Estimando su posición a través de los datos obtenidos con los sensores a bordo (sonar, cámara, laser, odómetro, brújula, GPS, etc.).



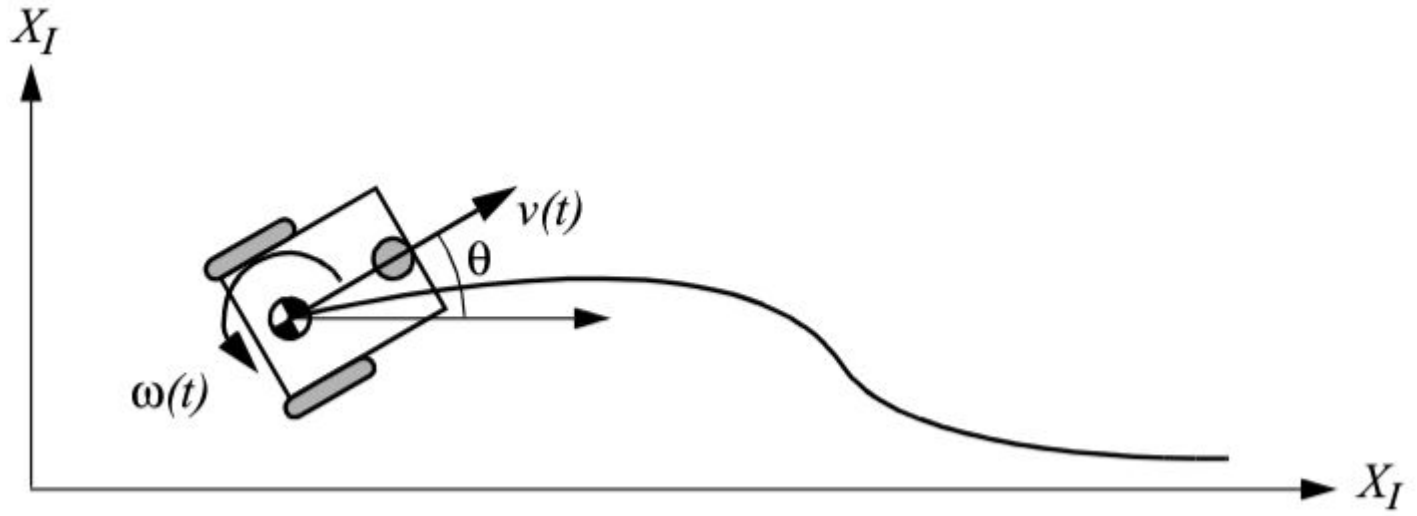
Localización

- La localización puede llevarse a cabo de dos maneras:
  - De manera continua (para cada moviento del robot).

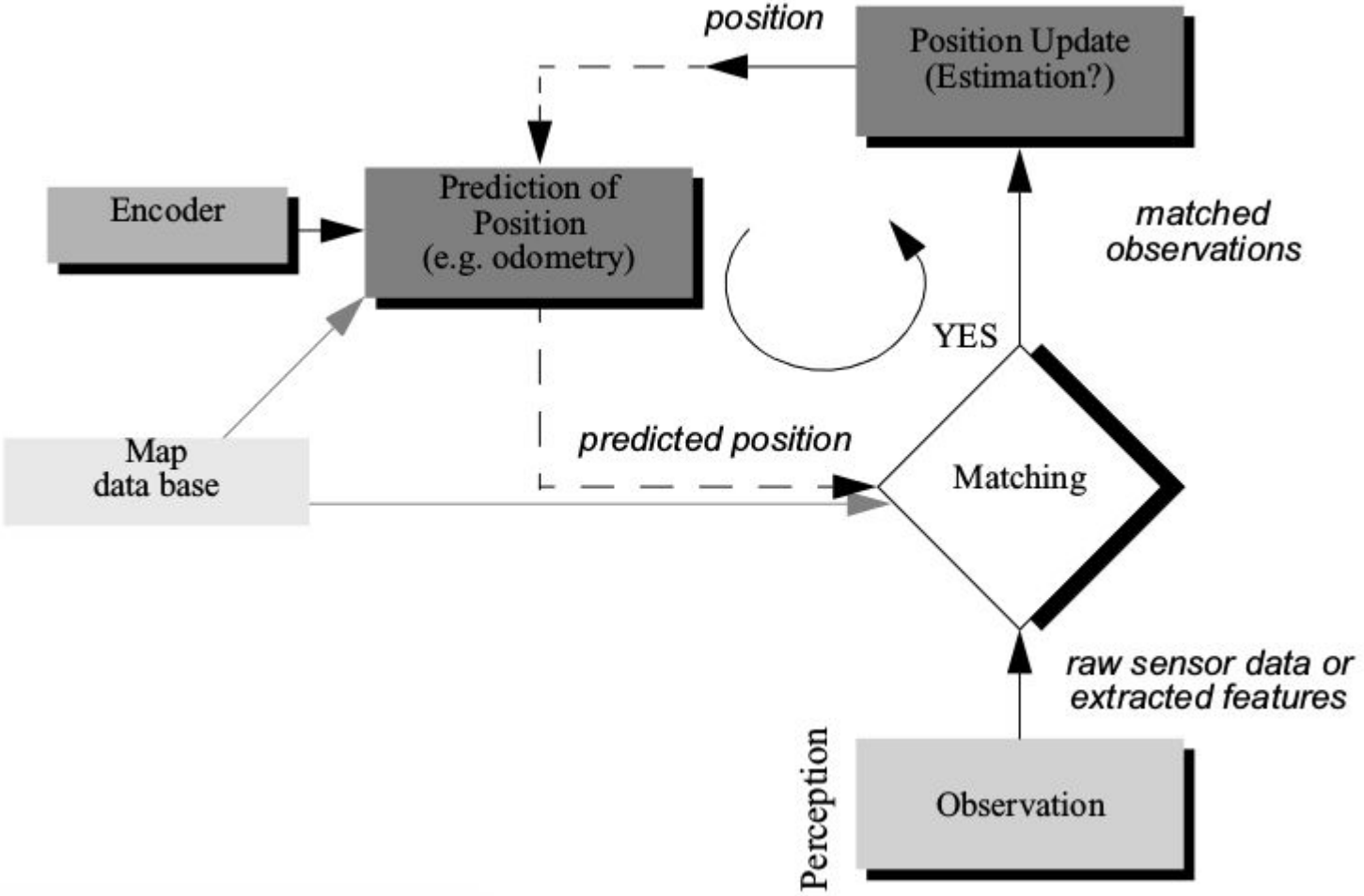


Localización

- La localización puede llevarse a cabo de dos maneras:
  - De manera continua (para cada moviento del robot).
  - De manera puntual (sólo cuando se reconocen u observan marcas conocidas en el ambiente).

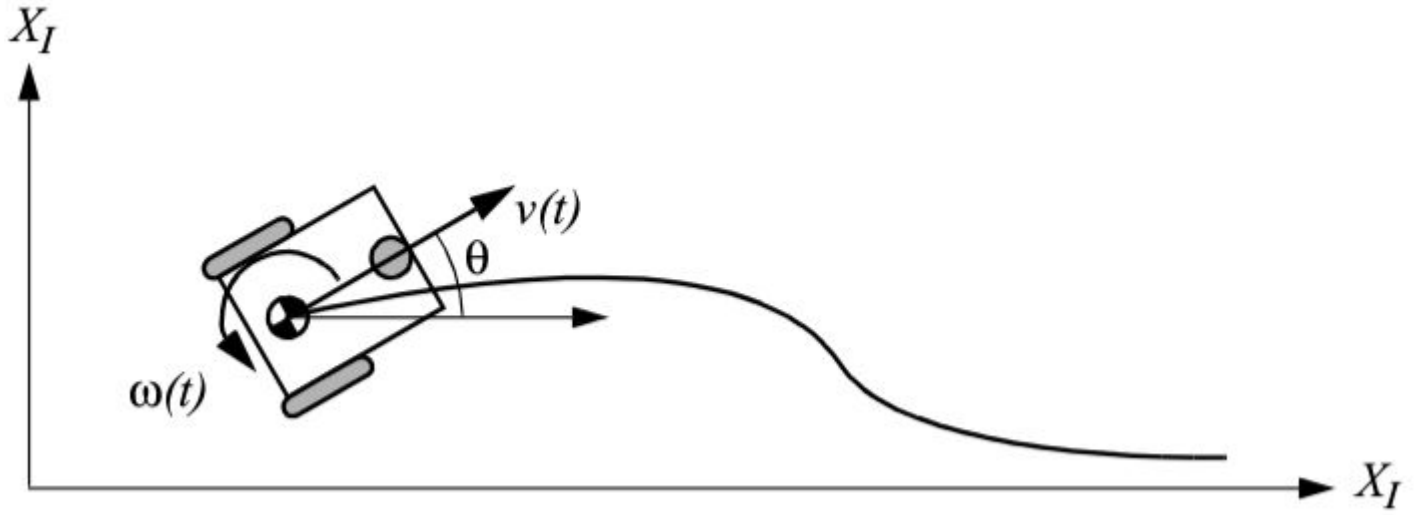


Esquema general de localización



Localización

- Retos en el proceso de localización:
  - Ruido en los sensores.
  - Aliasing.
  - Ruido en los efectores.



Ruidos en sensores

- Sonares y laser (dependen del material y ángulo de incidencia).
- Sensores inerciales (acelerómetros, inclinómetros).
- Se recomienda tomar múltiples lecturas y hacen fusión temporal/multi-sensorial o filtrado.

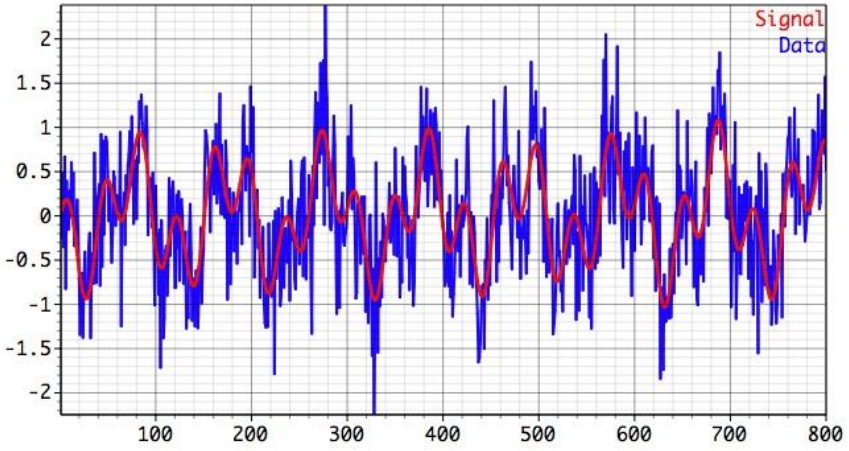


Imagen tomada de http://www.spectraworks.com/Help/prediction.html

Aliasing

- Mismas lecturas para diferentes estados.
- Ejemplo: las lecturas de un laser en dos cuartos diferentes pero de mismas dimensiones.

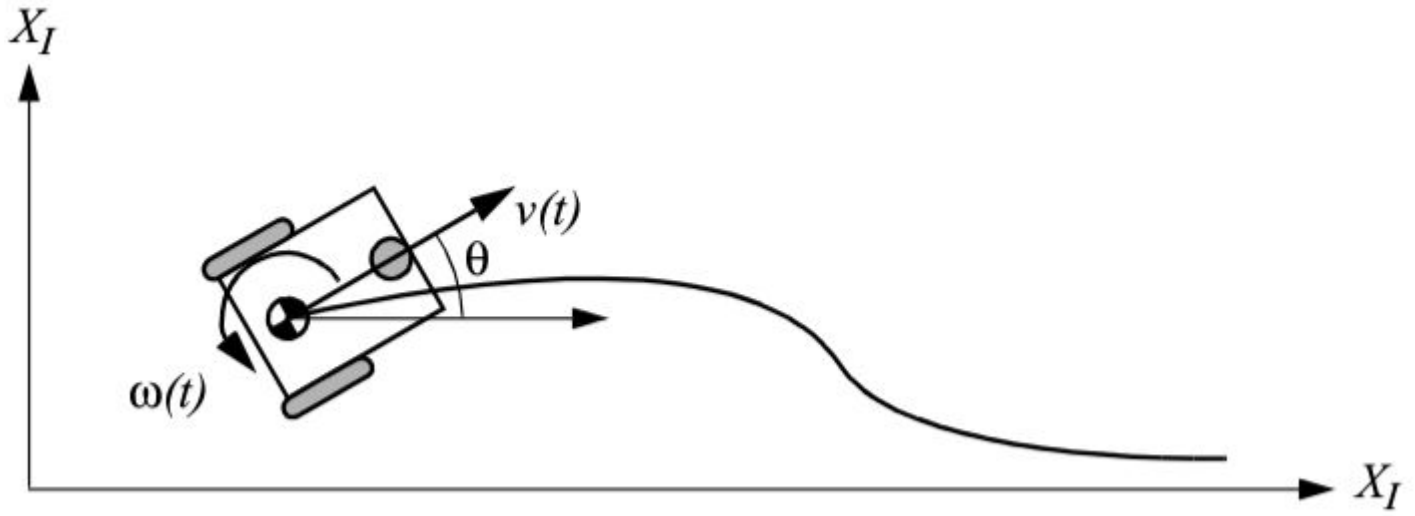
Ruido en los efectores

- Errores odométricos: poca resolución, no alineación de llantas, diámetro desigual, variaciones de contacto.
- Introducen errores en los estados futuros.
- El error de posición es acumulativo, por lo que hay que actualizar la posición contínuamente.
- Algunos errores son sintemáticos – determinístas y otros aleatorios – no determinístas.

Dead reckoning

- Estimación de la posición utilizando odometría.

$$p = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$$



Dead reckoning

- Estimación de la posición utilizando odometría.

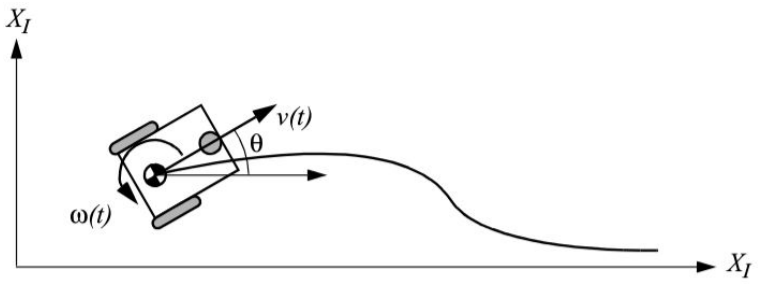
$$p = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$$

$$\Delta x = \Delta s \cos(\theta + \Delta\theta / 2)$$

$$\Delta y = \Delta s \sin(\theta + \Delta\theta / 2)$$

$$\Delta\theta = \frac{\Delta s_r - \Delta s_l}{b}$$

$$\Delta s = \frac{\Delta s_r + \Delta s_l}{2}$$



$(\Delta x;\Delta y;\Delta\theta)$ = path traveled in the last sampling interval;

$\Delta s_r;\Delta s_l$ = traveled distances for the right and left wheel respectively;

b = distance between the two wheels of differential-drive robot.

Dead reckoning

- Estimación de la posición utilizando odometría.

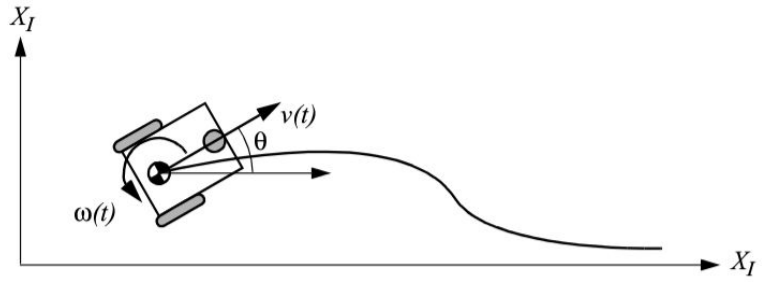
$$p = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$$

$$\Delta x = \Delta s \cos(\theta + \Delta\theta/2)$$

$$\Delta y = \Delta s \sin(\theta + \Delta\theta/2)$$

$$\Delta\theta = \frac{\Delta s_r - \Delta s_l}{b}$$

$$\Delta s = \frac{\Delta s_r + \Delta s_l}{2}$$



$(\Delta x;\Delta y;\Delta\theta)$ = path traveled in the last sampling interval;

$\Delta s_r;\Delta s_l$ = traveled distances for the right and left wheel respectively;

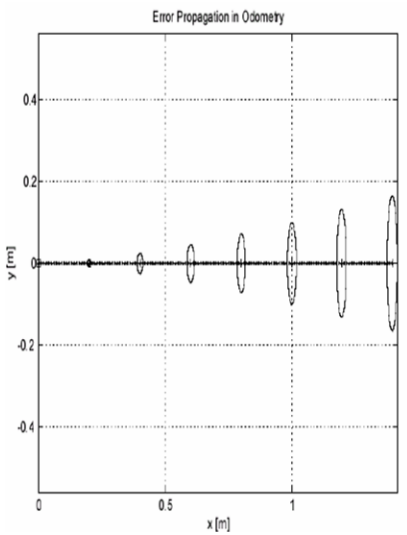
b = distance between the two wheels of differential-drive robot.

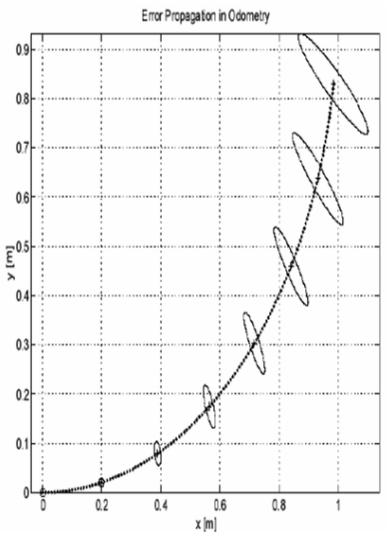
$$p' = \begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = p + \begin{bmatrix} \Delta s \cos(\theta + \Delta\theta/2) \\ \Delta s \sin(\theta + \Delta\theta/2) \\ \Delta\theta \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} \Delta s \cos(\theta + \Delta\theta/2) \\ \Delta s \sin(\theta + \Delta\theta/2) \\ \Delta\theta \end{bmatrix}$$

Dead reckoning (errores odométricos)

- Los errores de orientación son mayores que los de distancia.

$$p' = f(x, y, \theta, \Delta s_r, \Delta s_l) = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} \frac{\Delta s_r + \Delta s_l}{2} \cos\left(\theta + \frac{\Delta s_r - \Delta s_l}{2b}\right) \\ \frac{\Delta s_r + \Delta s_l}{2} \sin\left(\theta + \frac{\Delta s_r - \Delta s_l}{2b}\right) \\ \frac{\Delta s_r - \Delta s_l}{b} \end{bmatrix}$$





Localizarse o no localizarse

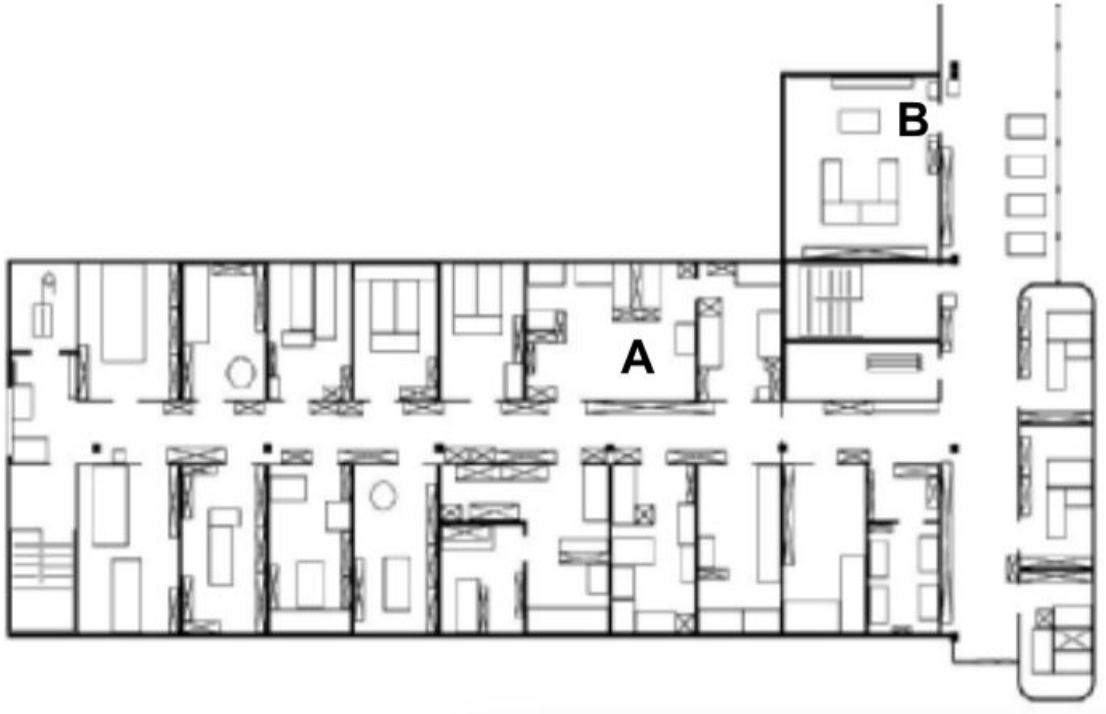
- La navegación no tiene que ser basada en localización necesariamente, esto es, no tiene por que conocerse la posición/orientación del robot con respecto a un mapa.
- Localizarse con respecto a un mapa ayuda a concer cuando un robot ha llegado a la meta.
- Sin embargo, conocer la ubicación explícita en un mapa no es la única solución.

Navegación basada en comportamiento

- Si los sensores y los actuadores son ruidosos de resolución limitada, entonces debe evitarse la creación de un mapa geométrico (métrico).
- **Alternativa**: diseñar un conjunto de comportamientos que en conjunto resulten en el movimento deseado del robot.
  - Bajo este esquema se evita el razonamiento explícito acerca de la posición/localización del robot, así como de la planificación.

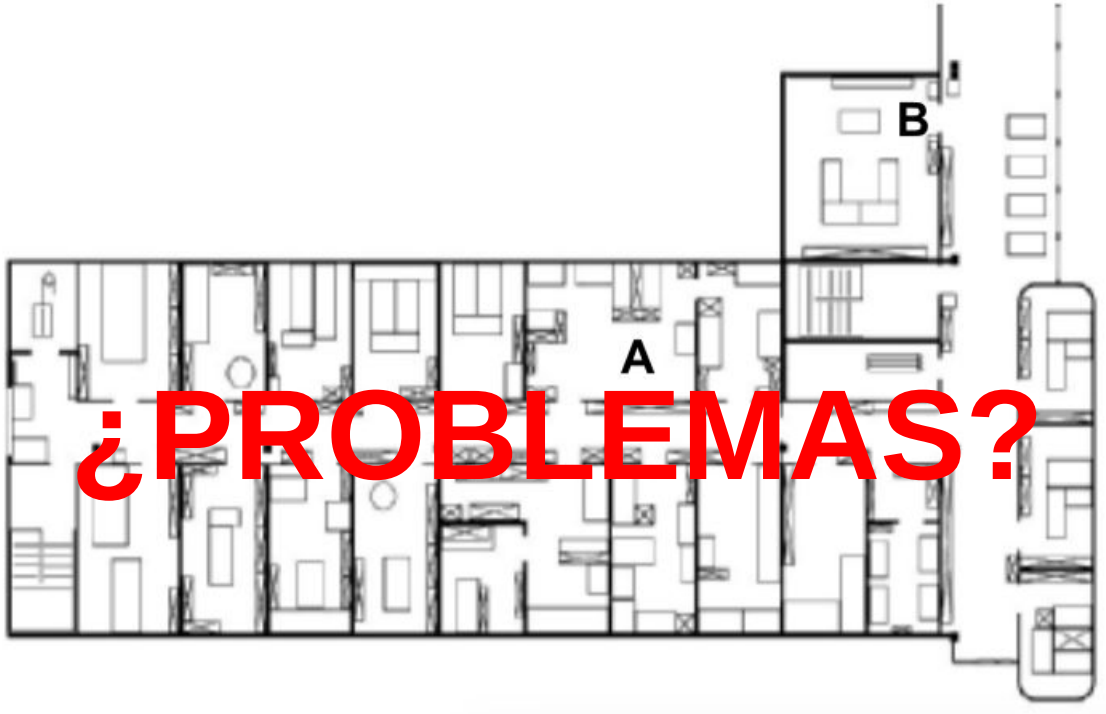
Navegación basada en comportamiento

- Debe existir una solución basada en un procedimiento más que en un cálculo geométrico.
  - **Ejemplo**: navegación siguiendo el muro izquierdo combinado con un reconocedor de la meta (la meta incluye algún razgo distinctivo único).

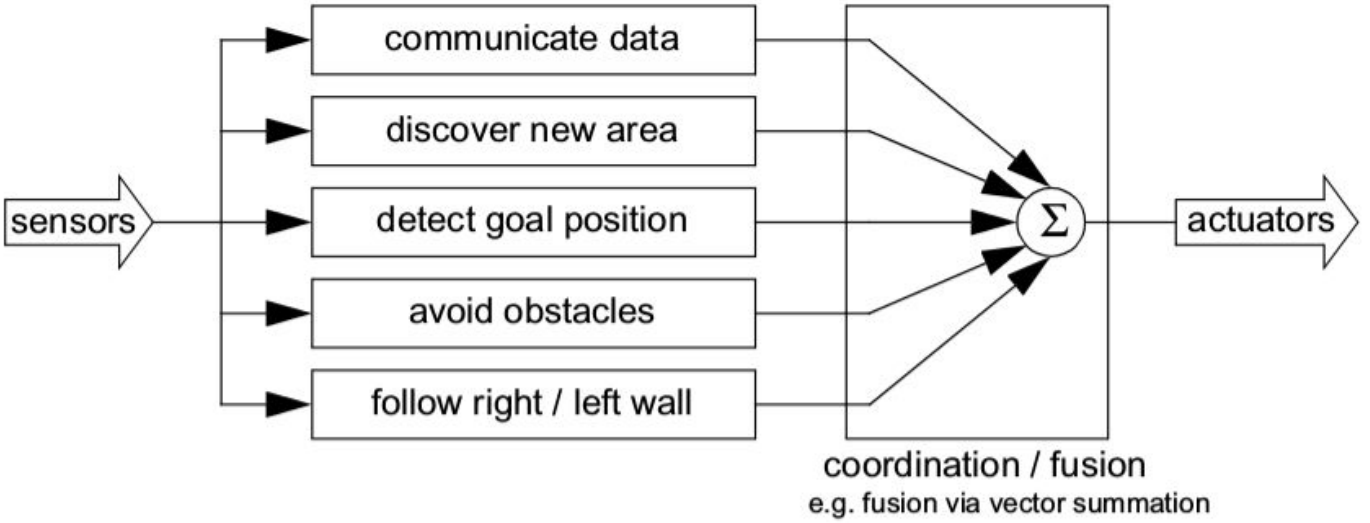


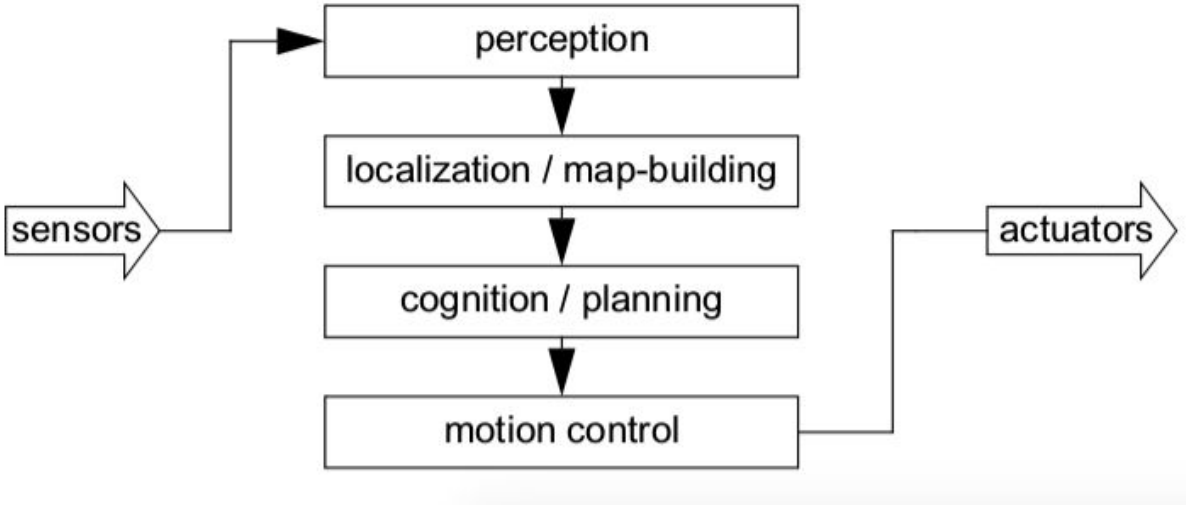
Navegación basada en comportamiento

- Debe existir una solución basada en un procedimiento más que en un cálculo geométrico.
  - **Ejemplo**: navegación siguiendo el muro izquierdo combinado con un reconocedor de la meta (la meta incluye algún razgo distinctivo único).



Localización vs Comportamiento



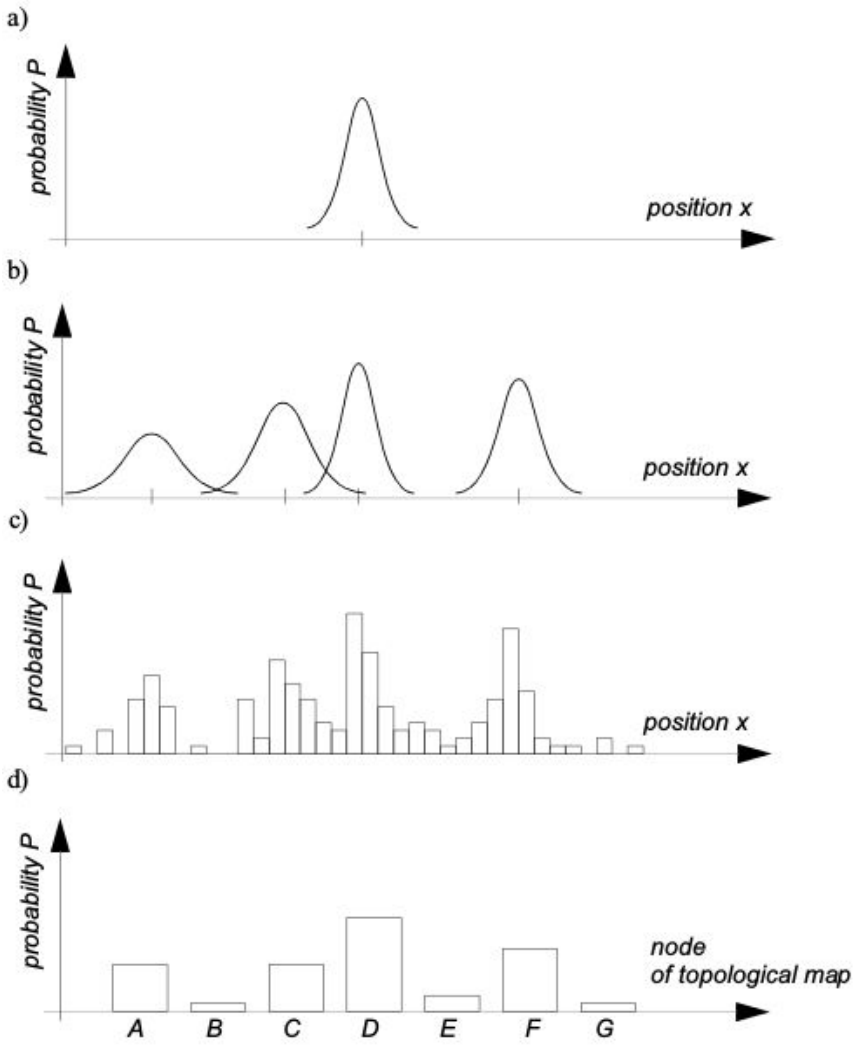


Representación de creencias

- Existen dos tipos de conceptos que deben ser representados para que el robot pueda llevar a cabo la navegación:
  - El modelo del ambiente a través de un mapa.
  - Su *creencia* acerca de su posición en el ambiente (mapa).
    - ¿Se cree en una sóla posición (hipótesis única)?
    - ¿Qué pasa si existen varias posibilidades (múltiples hipótesis)?

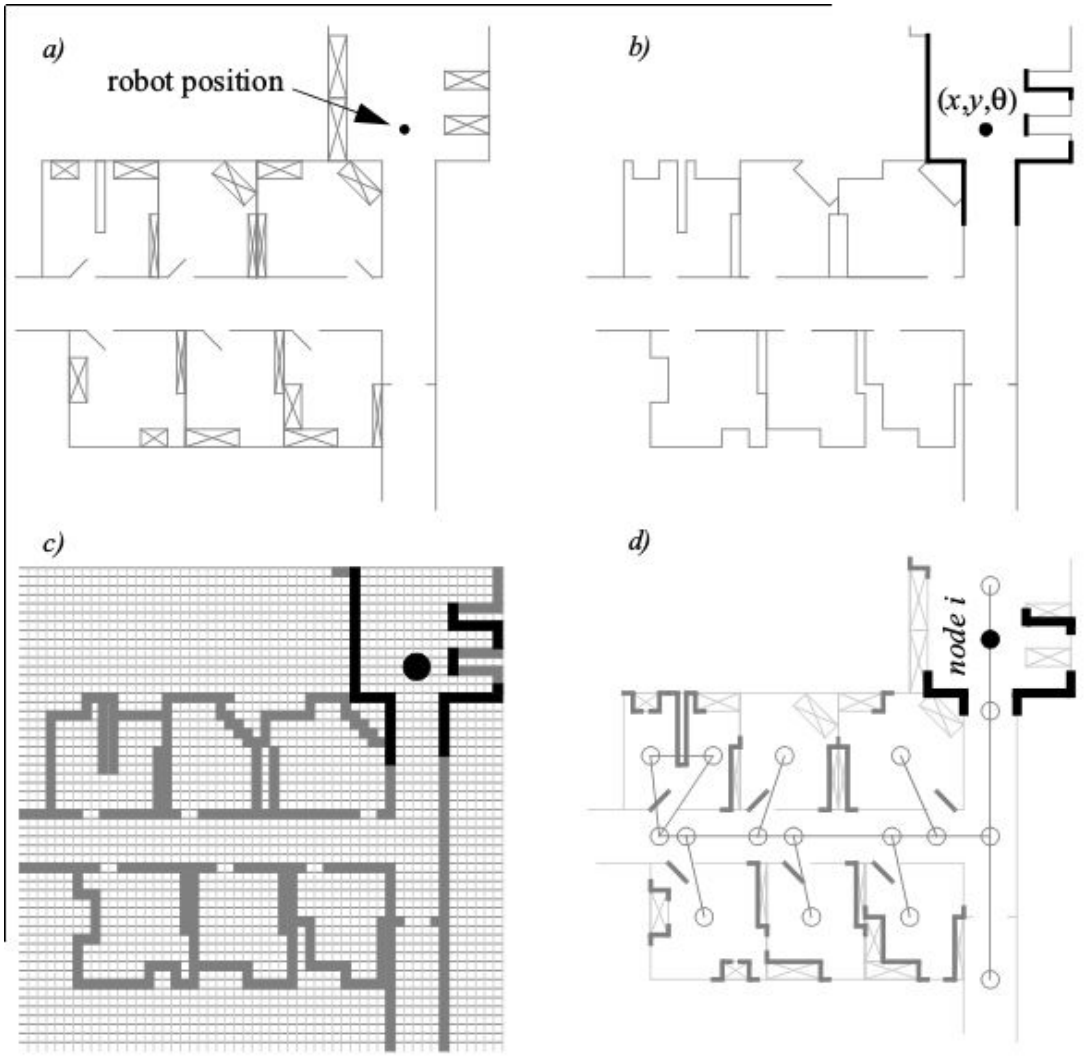
Representación de creencias

- Mapa contínuo una hipótesis.
- Mapa contínuo muchas hipótesis.
- Mapa discreto de rejillas con todas las posiciones.
- Mapa discreto de grafos con todos los nodos.



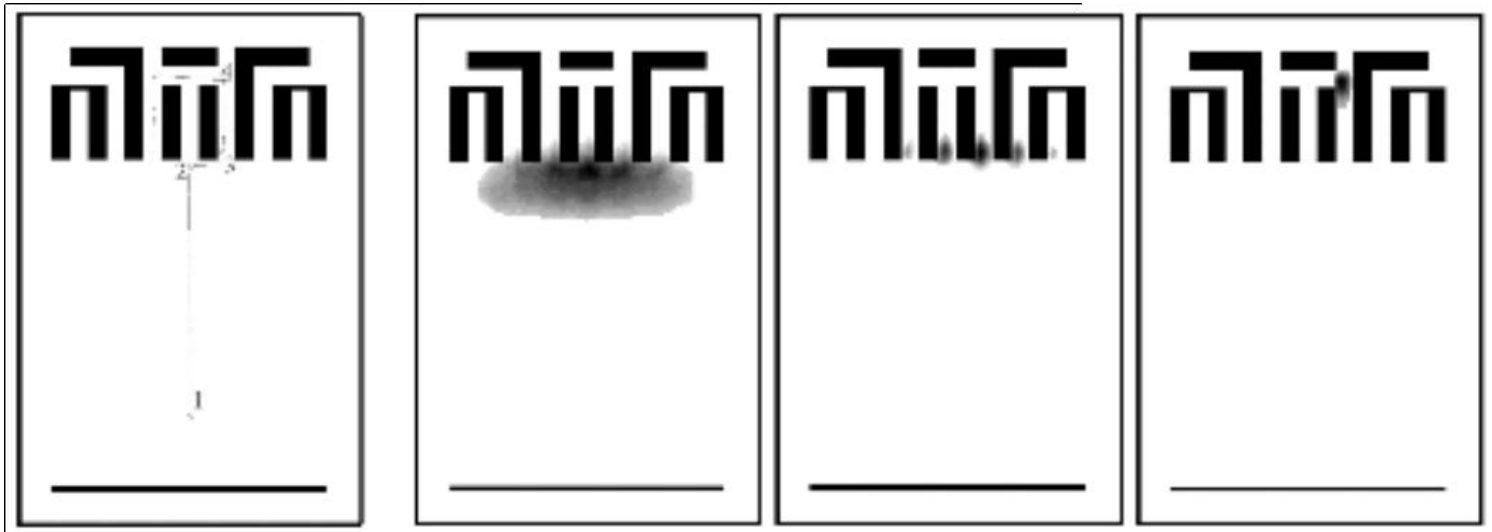
Representación de creencias

(Hipótesis única)



Representación de creencias
(Múltiples hipótesis)

- Se puede mantener un modelo de la incertidumbre con respecto a la posición del robot.
- **Problemas:**
  - De toma de decisión: ¿qué hipótesis es la mejor?
  - Asignación de probabilidad a cada hipótesis.
  - El incremento en el número de posibles hipótesis incrementa rápida el tiempo computacional (se vuelve intratable rapidamente en el espacio 3D.

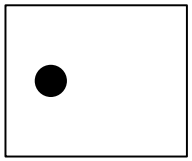


Path of the robot *Belief states at positions 2, 3, and 4*

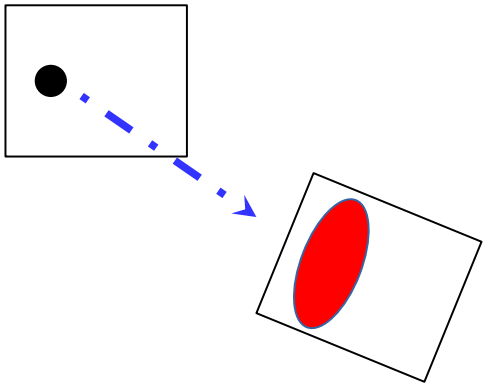
Localización basada en mapas probabilísticos

- Similar a la representación de creencias con múltiples hipótesis.
- Asociación de probabilidades a las posibles posiciones (configuraciones) del robot:
  - Localización de Markov.
    - Utiliza una distribución de probabilidad explícita.
    - Cada configuración se asume independiente de otra.
  - Localización basada en el Filtro Kalman (popular en SLAM).
    - Utiliza una distribución de probabilidad Gaussiana.
    - Las configuraciónes no son independientes.
    - Desarrollado originalmente para sistemas lineales.
    - EKF, UKF, para sistemas no lineales.

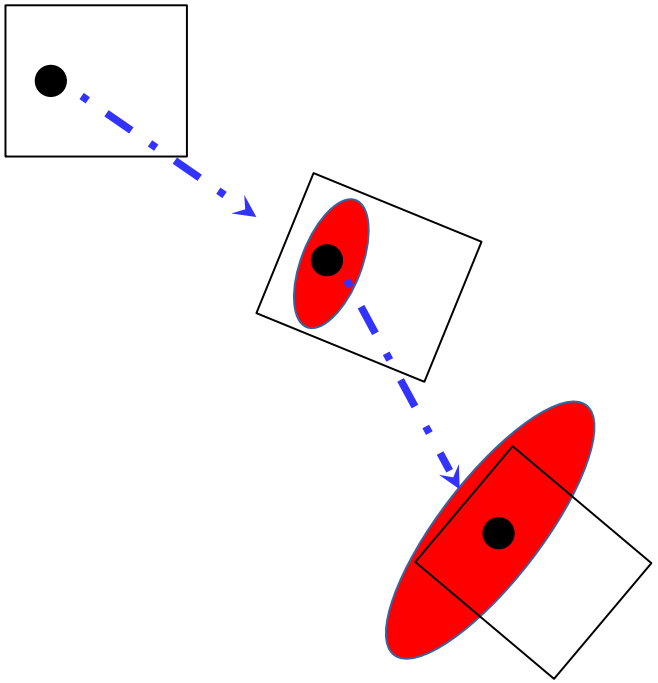
Localización basada en mapas probabilísticos



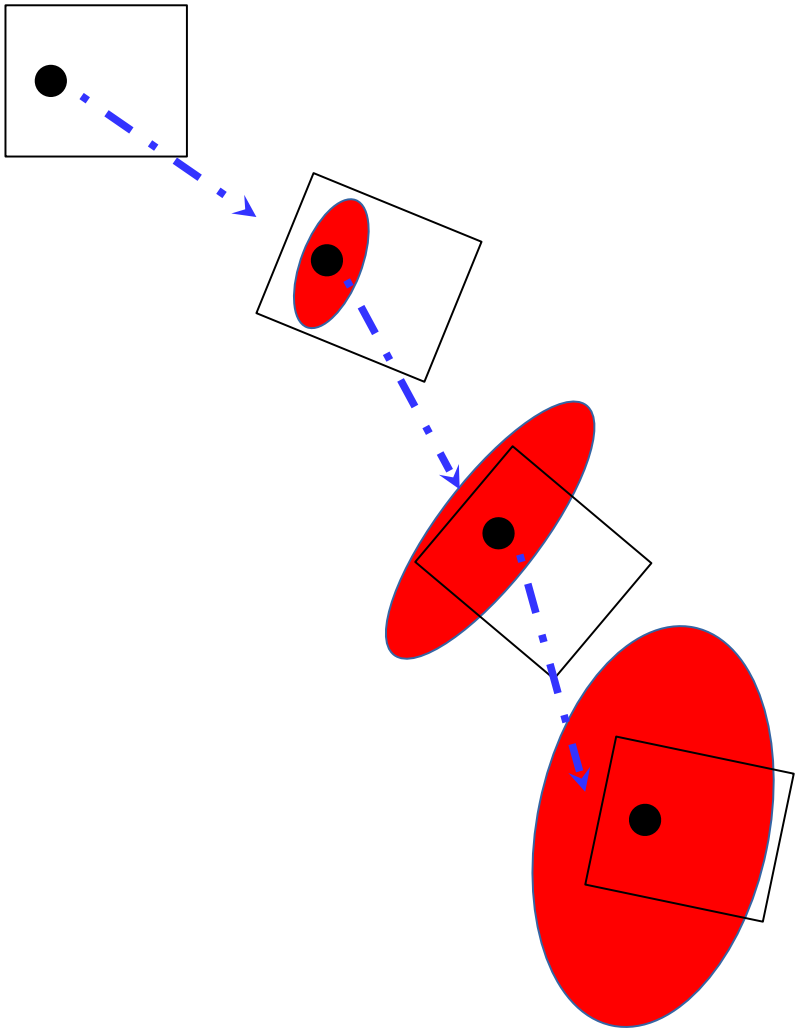
Localización basada en mapas probabilísticos



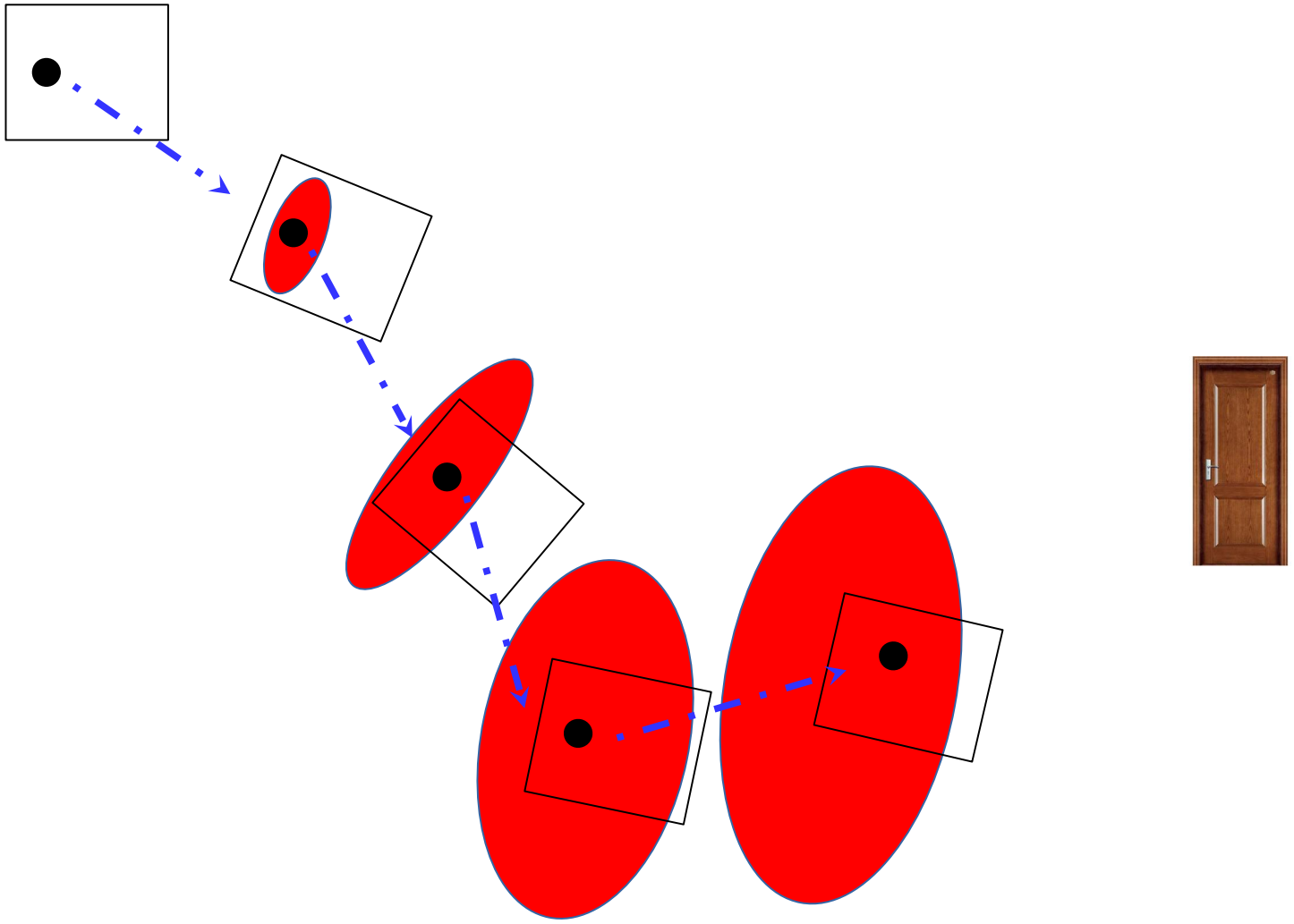
Localización basada en mapas probabilísticos



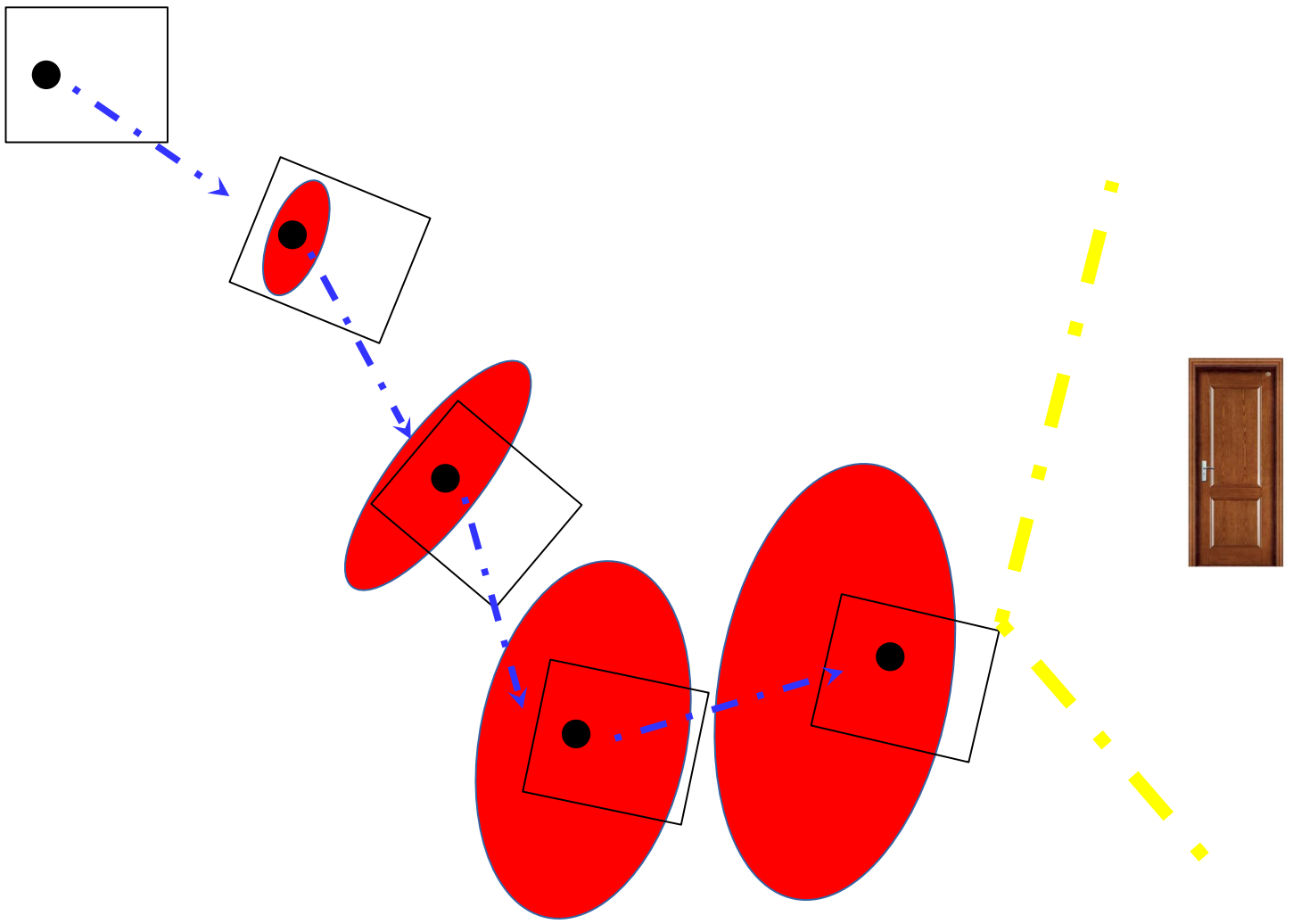
Localización basada en mapas probabilísticos



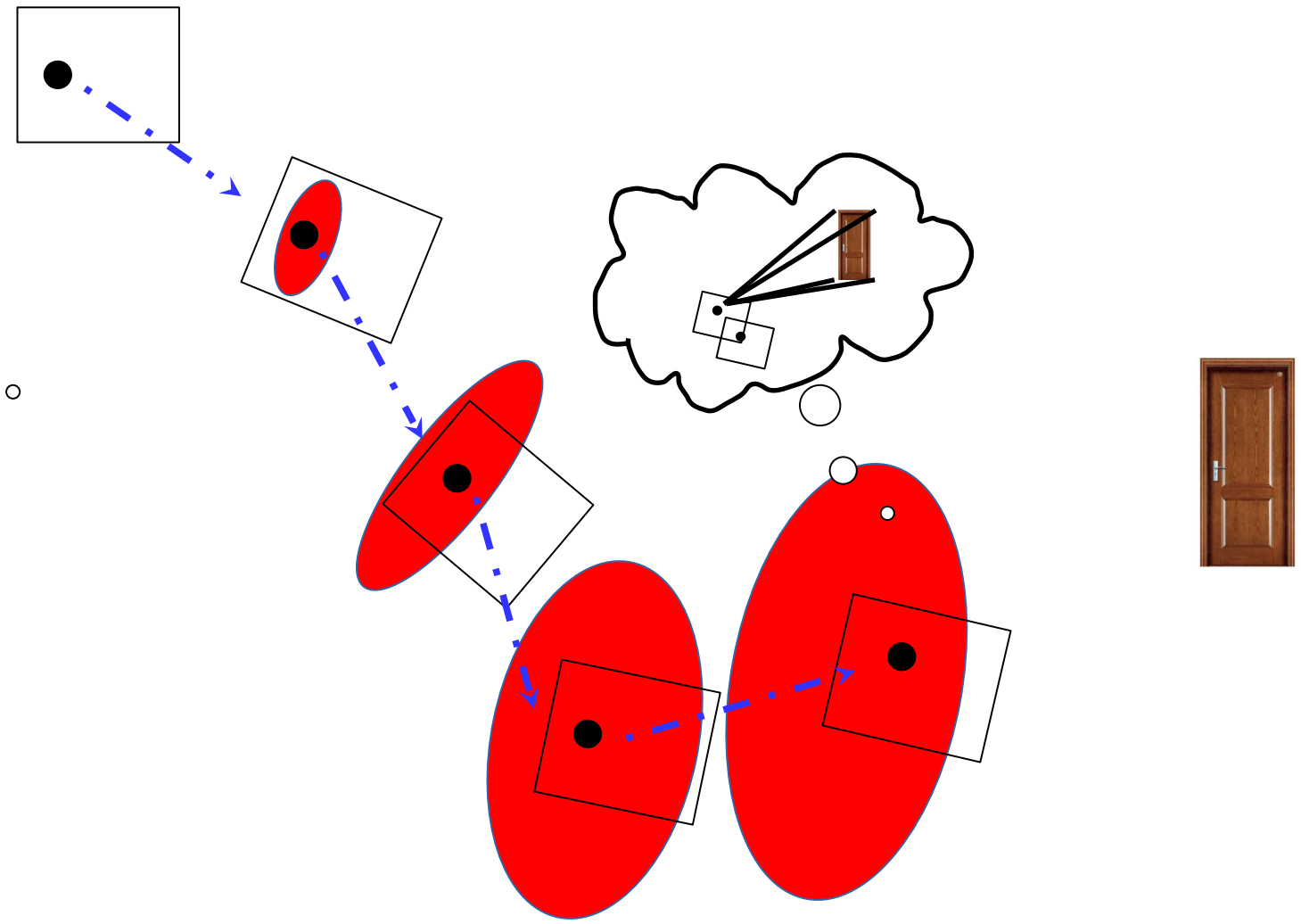
Localización basada en mapas probabilísticos



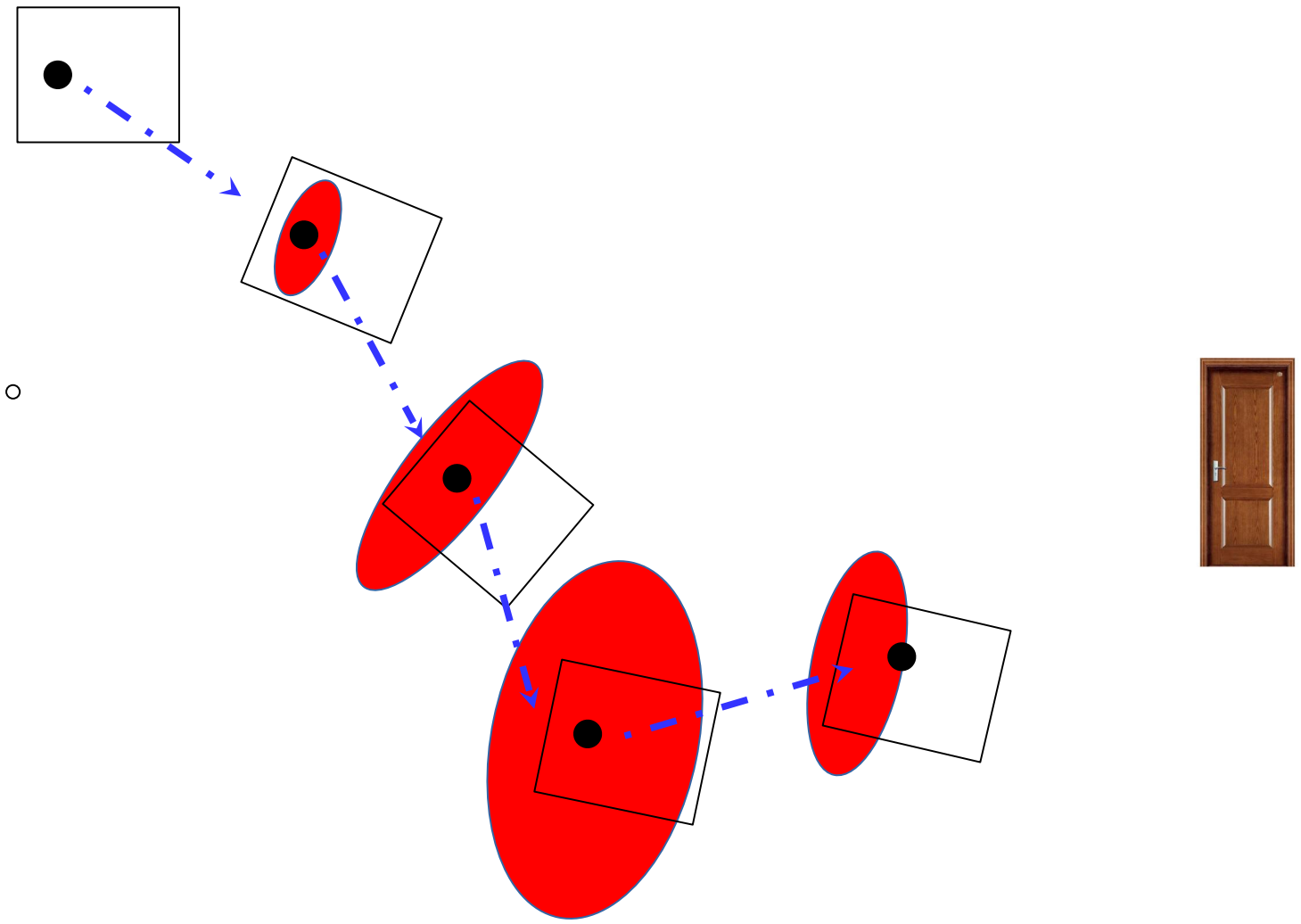
Localización basada en mapas probabilísticos



Localización basada en mapas probabilísticos



Localización basada en mapas probabilísticos



Localización de Markov

- **Idea**: asignar una probabilidad a cada posible posición del robot.
- Dada una distribución de probabilidad inicial o a priori P(A), queremos calcular la probabilidad a porterior dado el movimiento y las medidas de los sensores P(A|B).
- Con la información de los sensores (s), podemos calcular la probabilidad del lugar (l) usando Bayes.

Localización de Markov

$$P(l|s) = \frac{p(s|l)p(l)}{p(s)}$$

- La parte clave es el cálculo de p(s|l) o modelo del sensor.
- La probabilidad del lugar (l) dada una acción (a) la podemos calcular como:

$$p(l_t|a_t) = \int p(l_t|l'_{t-1}, a_t)P(l'_{t-1})dl'_{t-1}$$

- Tenemos que ver todos los posibles lugares que pueden alcanzar el mismo lugar con esa acción.

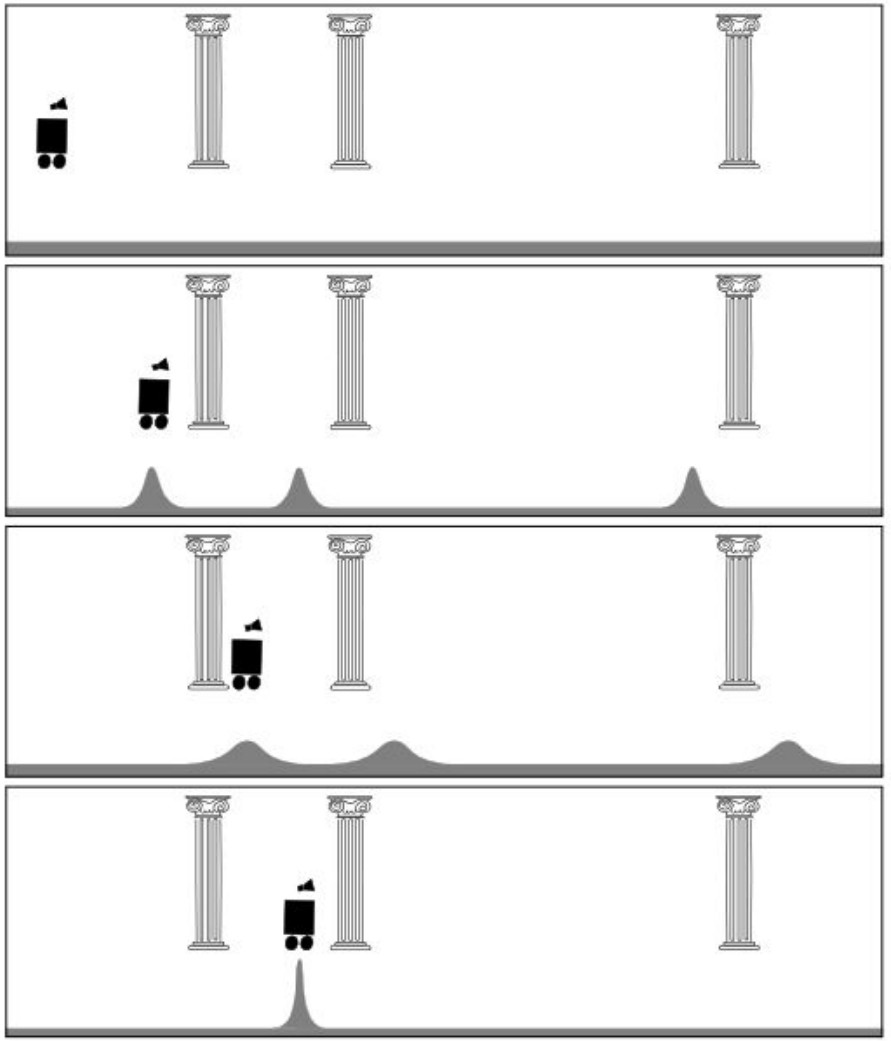
Localización de Markov

- En el caso de un mapa de rejillas de ocupación, esto sería:

$$p(l_t|a_t) = \sum p(l_t|l'_{t-1}, a_t)P(l'_{t-1})$$

- Como los cálculos se tienen que hacer para todos los posibles lugares, se vuelve computacionalmente caro para espacios grandes y/o celdas de ocupación pequeñas.

Localización de Markov

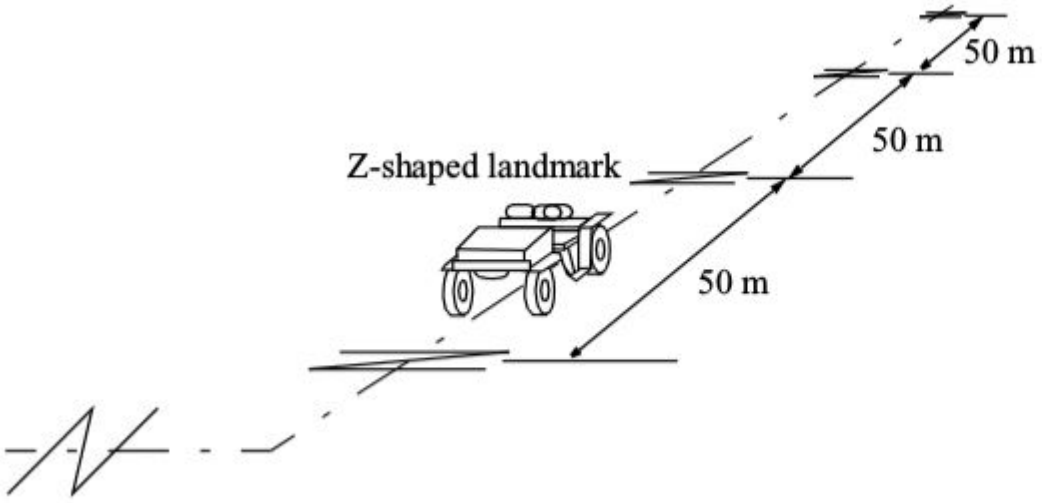


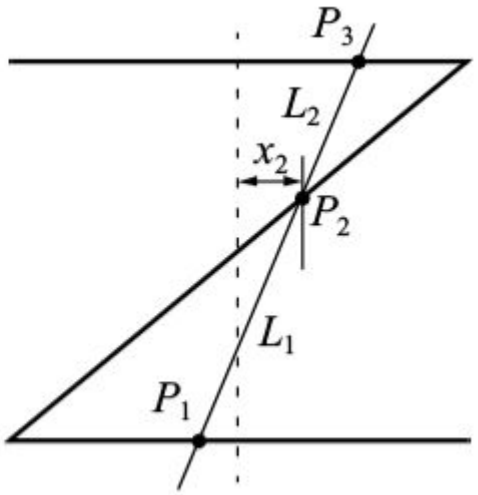
Localización de Markov

- Una posibilidad es hacer la actualización solo sobre un muestreo sesgado (randomized sampling, particle filter, Monte Carlo).
- La actualización se hace sólo sobre un subconjunto y normalmente se incluyen algunos otros lugares no tan probables.
- Se gana eficiencia pero se pierde representación de hipótesis.
- El filtro Kalman se puede utilizar como parte atómica de éste método.

Localización y navegación basada en marcas

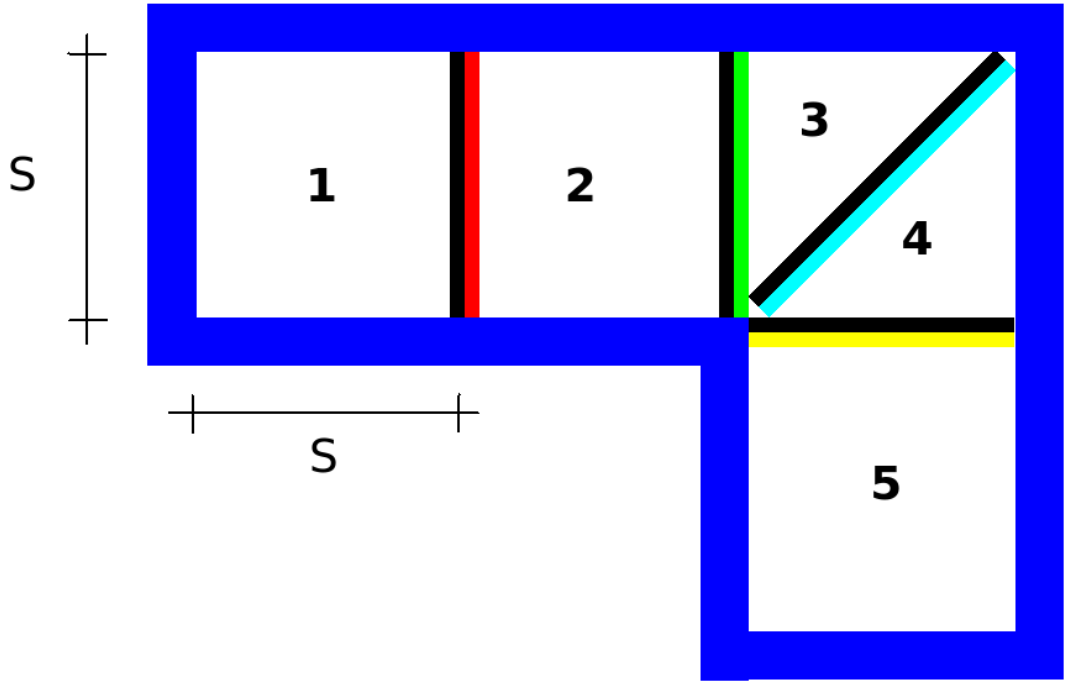
- El ambiente se señala con marcas que puedan ser identificadas por el robot para así localizar su posición.
- Las marcas definen objetos pasivos en el ambiente que permitan realizar una buena localización.
- Cuando el robot identifica la marca se localiza, pero sin las marcas viaja a ciegas usando sólo la odometría hasta que vuelve a encontrar otra marca.
- Esto requiere colocar marcas seguidas y alterar el ambiente o utilizar marcas naturales, que pueden ser de diferente tipo.





Localización y navegación basada en marcas

- El ambiente se señala con marcas que puedan ser identificadas por el robot para así localizar su posición.



Localización global

- Los métodos anteriores son métodos que usalmente utilizan la posición anterior del robot de algún modo para estimar la posición actual, estos métodos se conocen como locales o incrementales.
- La localización global determina la posición del robot sin tener una referencia de su posición anterior.
- Dadas las lecturas de los sensores, normalmente hay varias posibles localizaciones, por lo que se tiene que desplazar para encontrar su posición real.

Referencias

- Siegwart, R., & Nourbakhsh, I. R. (2004). Autonomous mobile robots. Massachusetts Institute of Technology.
- Kendall, A., Grimes, M., & Cipolla, R. (2015). Posenet: A convolutional network for real-time 6-dof camera relocalization. In Proceedings of the IEEE international conference on computer vision (pp. 2938-2946).
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).
- Glocker, B., Izadi, S., Shotton, J., & Criminisi, A. (2013, October). Real-time RGB-D camera relocalization. In 2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR) (pp. 173-179). IEEE.
- Gee, A. P., & Mayol-Cuevas, W. W. (2012, September). 6D Relocalisation for RGBD Cameras Using Synthetic View Regression. In BMVC (Vol. 1, p. 2).
- Cabrera-Ponce, A. A., & Martinez-Carranza, J. (2019, November). Aerial geo-localisation for MAVs using PoseNet. In 2019 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED UAS) (pp. 192-198). IEEE.
- Mur-Artal, R., Montiel, J. M. M., & Tardos, J. D. (2015). ORB-SLAM: a versatile and accurate monocular SLAM system. IEEE transactions on robotics, 31(5), 1147-1163.