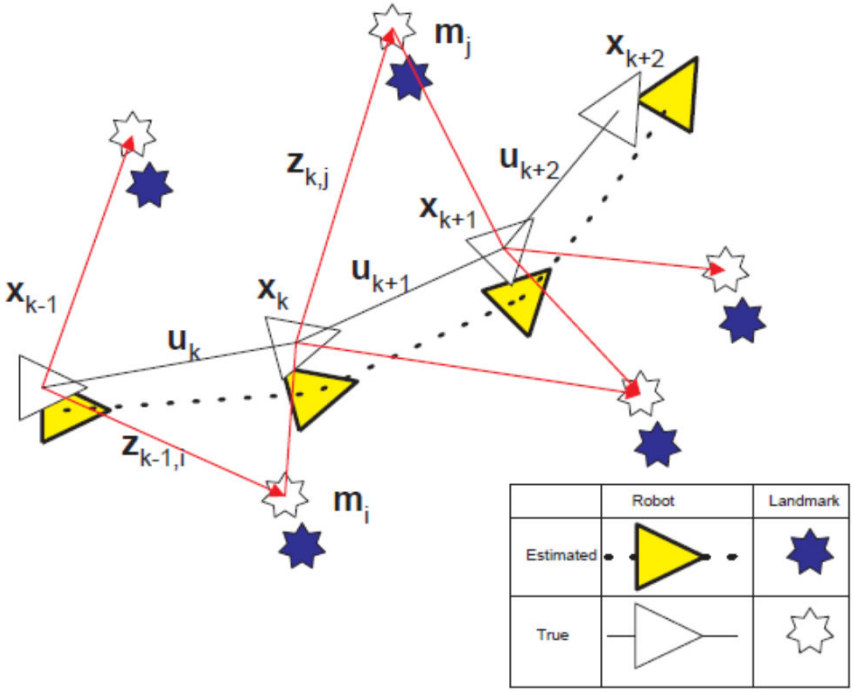
SLAM in practice

IAR Lecture 11

Barbara Webb

In principle, SLAM methods can use the sequence of actions **U** and measurements **Z** to infer both the map **M** and the robot positions **X**.



In practice there are a number of remaining issues:

- Computational complexity
- Data association (correspondence of landmarks)
- Dynamic environments

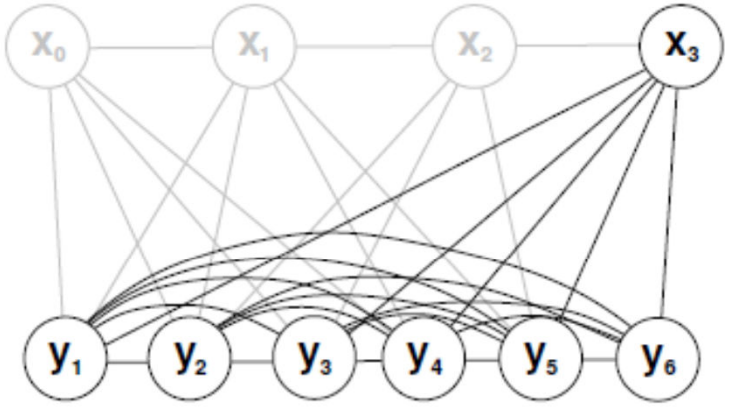
Reducing computational load

- Recall that *prediction* step only changes *robot pose* uncertainty
- Don't need to calculate landmark-landmark entries in covariance matrix
- There are several other methods to keep the matrix sparse.

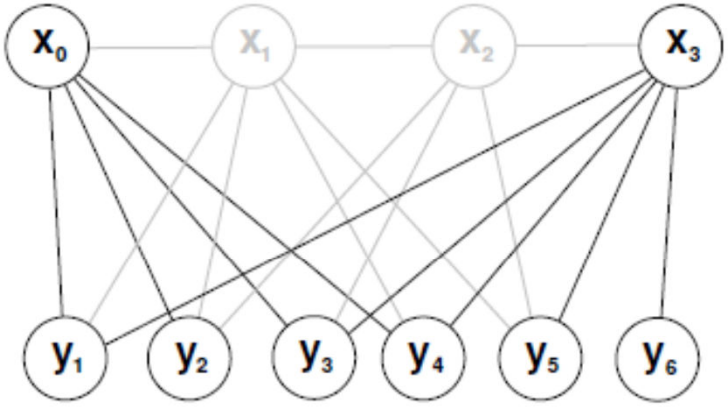
$$Bel(x_t, m_t) = \left\langle \begin{pmatrix} x \\ y \\ \theta \\ l_1 \\ l_2 \\ \vdots \\ l_N \end{pmatrix}, \begin{pmatrix} \sigma_x^2 & \sigma_{xy} & \sigma_{x\theta} & \sigma_{xl_1} & \sigma_{xl_2} & \cdots & \sigma_{xl_N} \\ \sigma_{xy} & \sigma_y^2 & \sigma_{y\theta} & \sigma_{yl_1} & \sigma_{yl_2} & \cdots & \sigma_{yl_N} \\ \sigma_{x\theta} & \sigma_{y\theta} & \sigma_\theta^2 & \sigma_{\theta l_1} & \sigma_{\theta l_2} & \cdots & \sigma_{\theta l_N} \\ \sigma_{xl_1} & \sigma_{yl_1} & \sigma_{\theta l_1} & \sigma_{l_1}^2 & \sigma_{l_1 l_2} & \cdots & \sigma_{l_1 l_N} \\ \sigma_{xl_2} & \sigma_{yl_2} & \sigma_{\theta l_2} & \sigma_{l_1 l_2} & \sigma_{l_2}^2 & \cdots & \sigma_{l_2 l_N} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \sigma_{xl_N} & \sigma_{yl_N} & \sigma_{\theta l_N} & \sigma_{l_1 l_N} & \sigma_{l_2 l_N} & \cdots & \sigma_{l_N}^2 \end{pmatrix} \right\rangle$$

Reducing computational load

- SLAM is formally equivalent to the computer vision problem of ‘structure from motion’
- An efficient solution is to use Keyframe bundle adjustment to optimise over the graph
- See Strasdat et al (2010)



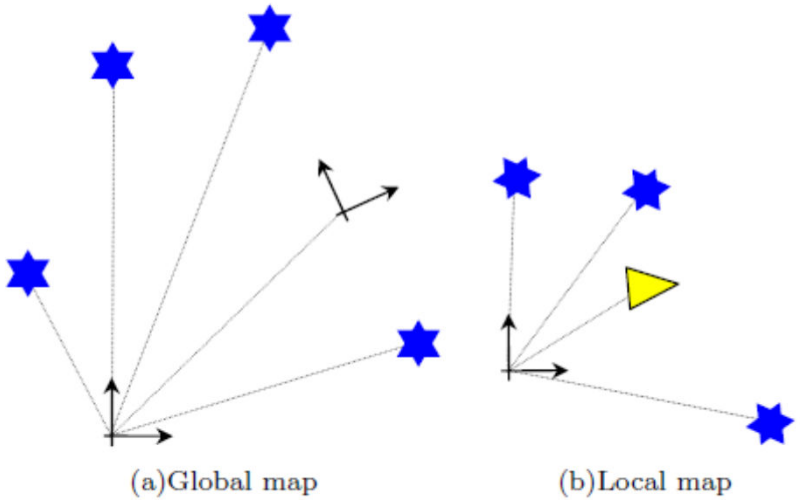
(c) Filter

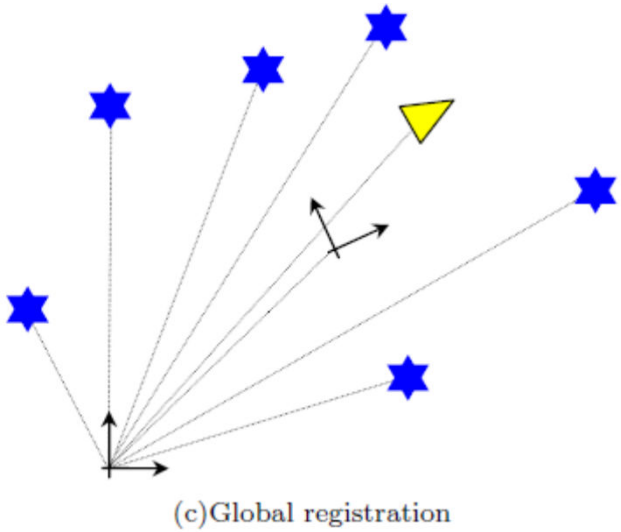


(d) Keyframe BA

Reducing computational load

- Observation update only needs to apply to locally visible landmarks
- Use occasional global updates to synchronise whole map





Data association problem

- EKF SLAM assumes that observed landmarks can be correctly matched to those in the map
- A single incorrect association can lead to catastrophic increase in uncertainty
- Standard solution ("statistical validation gating") for each observed landmark:
  - Hypothesise it is a new landmark
  - Update all landmark estimates
  - Measure how close the new landmark is to each existing landmark
  - If it exceeds a threshold distance from all existing landmarks, expand the map by adding it to the landmark list
  - Else associate it with whichever landmark is closest
- Works best for few, well separated landmarks, but fewer landmarks results in less accurate localisation = trade-off.

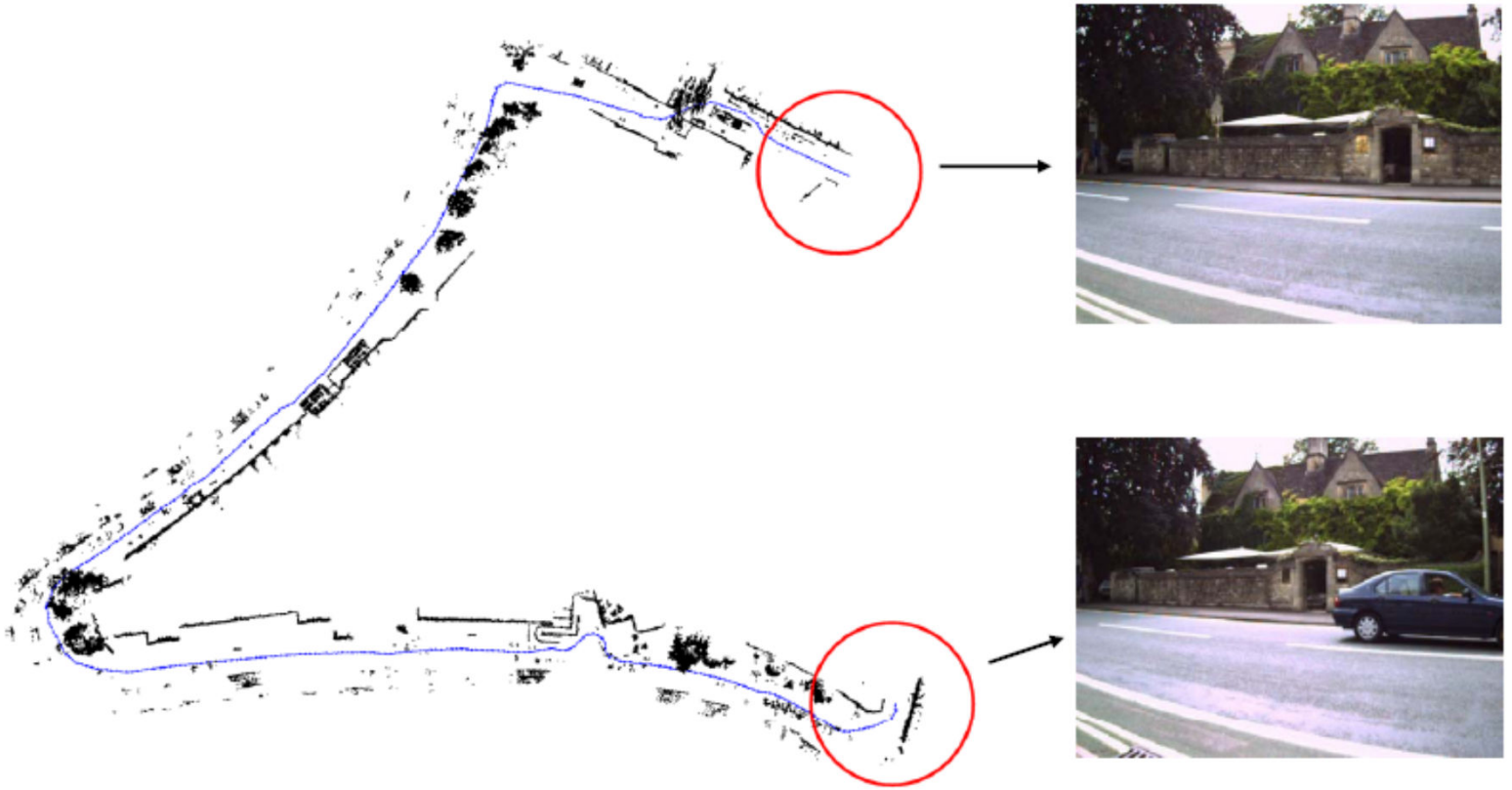
Data association problem

Improving the standard approach:

- ‘Batch-gating’: look for simultaneous best possible match across all landmarks
- ‘Map management’:
  - Add new landmarks to provisional list and don’t use in pose estimate until observed in several successive steps
  - Keep existence probability estimate for each landmark and remove if repeatedly not observed when expected
- Determine association by appearance as well as position – ‘distance’ includes distance in feature space

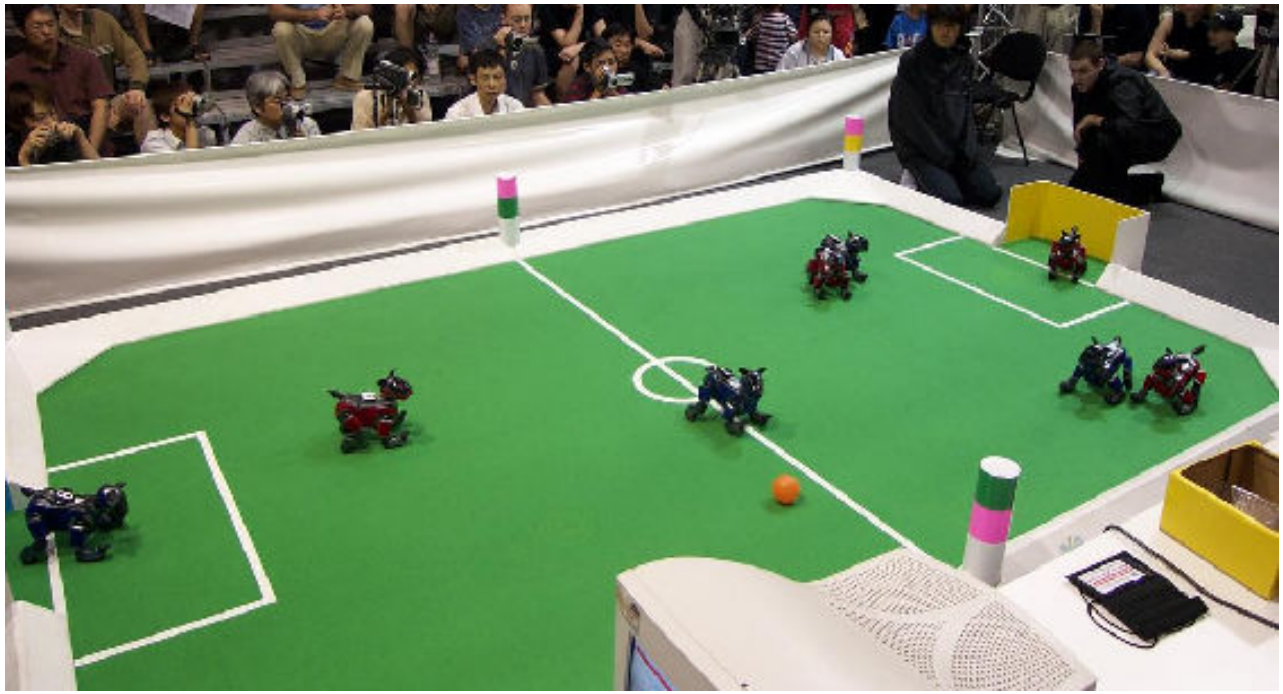
Appearance matching

- Particularly helpful for loop closing (old landmark *position* estimates are likely to be very noisy)



Appearance matching

- There are several approaches to make appearance matching more robust:
  - Using distinctive environmental cues



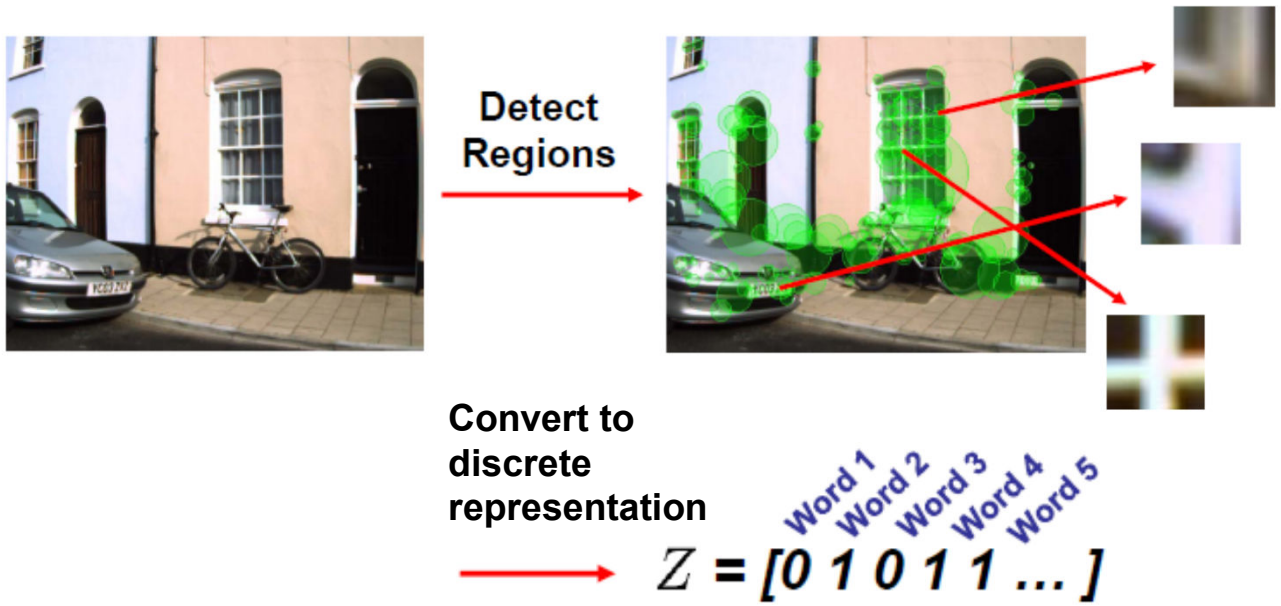
Appearance matching

- Can use computer vision methods to match from different viewpoints, under different lighting etc.

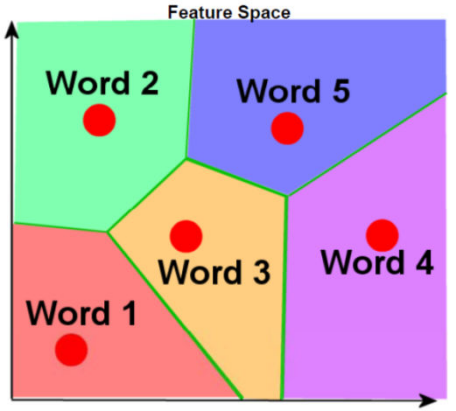
Appearance matching

- Using more complex feature descriptions

  E.g. ‘bag of words’ (Cummins & Newman)



Learn words by assigning to clusters in feature space from many example regions



Data association problem

- The methods mentioned so far use statistics to find the best match, but then assume this match was correct for all future updates, i.e., they do not maintain estimate of ‘association probability’
- Particle filter method inherently maintains multiple hypotheses about the match, one for each particle
- Particles with poor matches will be poor estimators and will tend to disappear

Dynamic environments

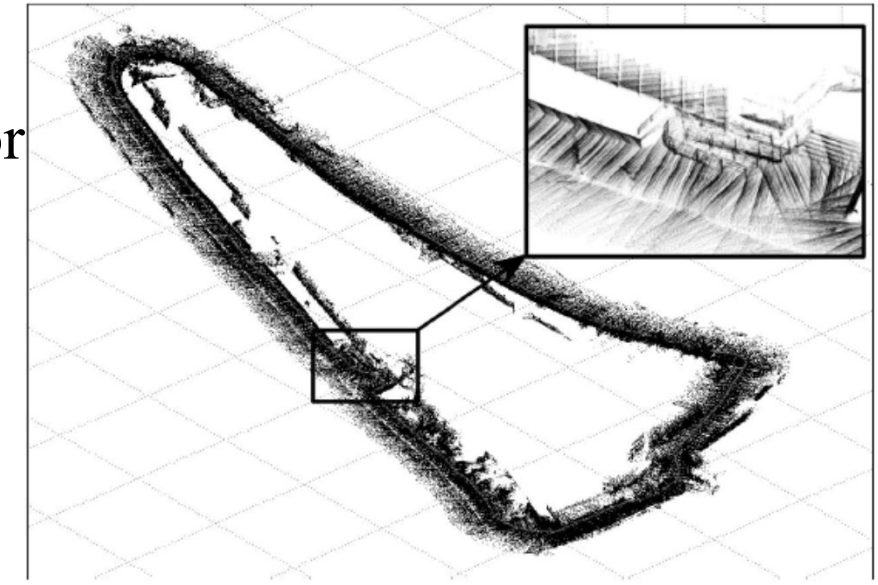
- Previous map management techniques can help:
  - Only include landmarks that have some permanence
  - Remove landmarks that appear obsolete (no longer observed where predicted)
- For some sensor systems might be able to detect and exclude moving landmarks
- SLAM is highly redundant so can still often work in changing environments

Some additional SLAM varieties

Pose-based SLAM

- Maintain explicit estimate of the pose but not the map
- Greatly simplifies update complexity
- Store raw sensor data along with trajectory

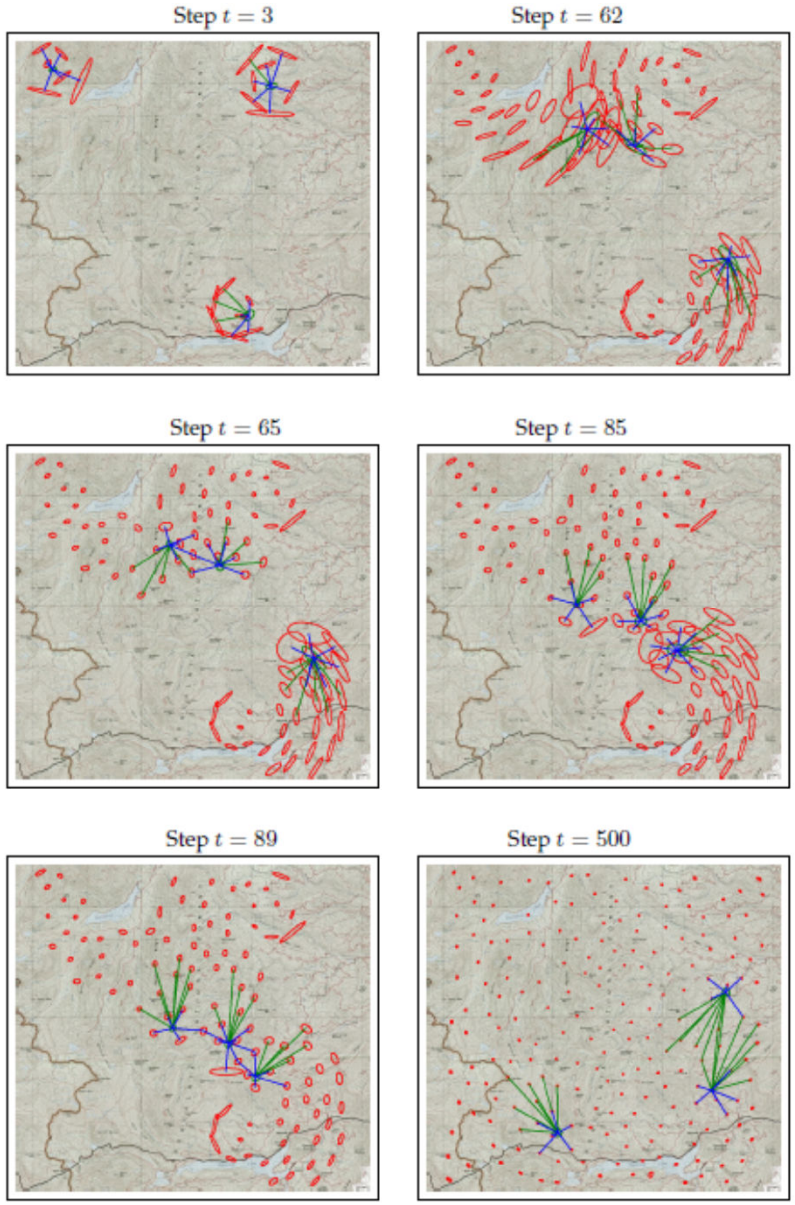
- Can visualise the map by superimposing recorded sensor data on the positions
- Easy to store auxiliary data, i.e., aspects of the measurements not used for estimation, such as colour, texture, events.



Newman et al (2006)

Multi-robot SLAM

- Each robot builds own map: want to merge into joint map
- Data association is now is to establish correspondence between landmarks between the individual robot maps
- Can then find appropriate alignment of co-ordinate frames
- Difficult if no *a priori* knowledge of how much the maps *should* overlap
  - E.g. mapping different floors of same building

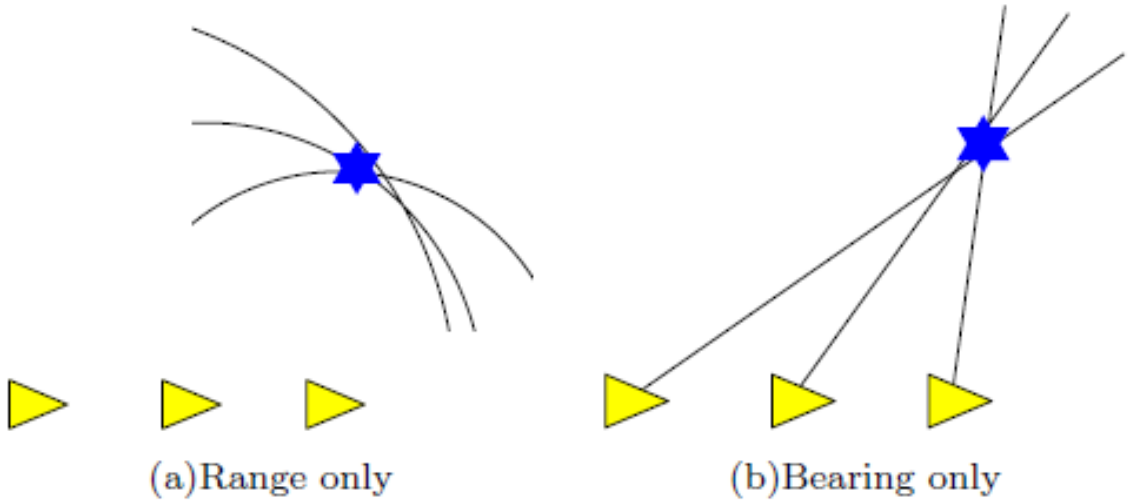


Thrun et al (2005)

Active SLAM

- Can utilise active *sensing* to improve position estimates

  Sonar sensing vs. visual sensing



- Note both can be disambiguated by repeated observations from a moving robot

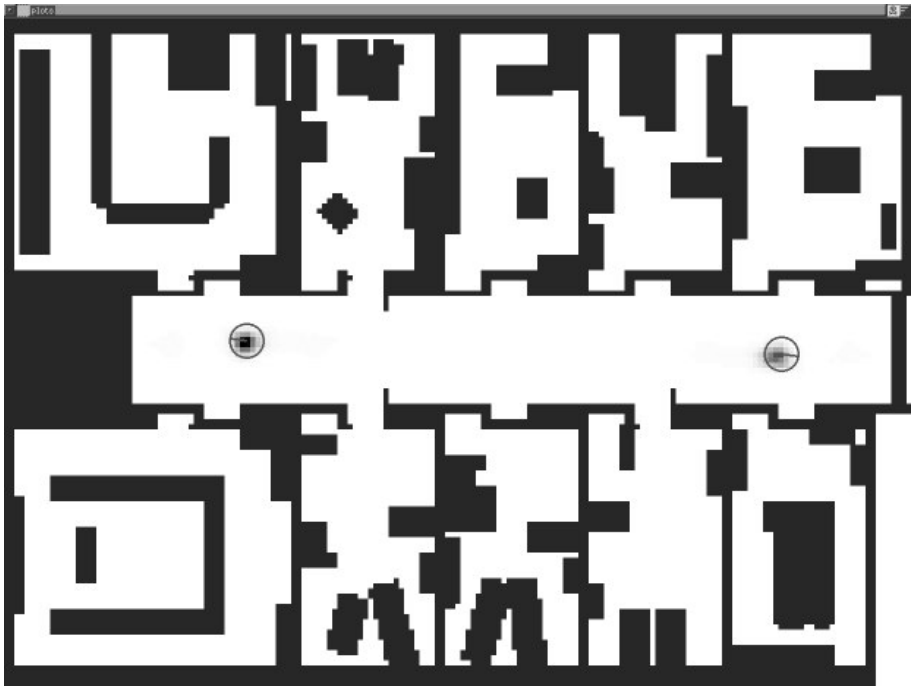
Active Localisation

- Try to actively choose a control action **u** that will improve localisation accuracy
- A common approach is information theoretic control:
  - Quantify the state uncertainty using an information metric *I=f(p(x))*
    - E.g. entropy: $I = H_p(x) = E(-\log_2 p(x)) = -\int p(x) \log_2 p(x) dx$
  - For each possible action, calculate the information gain *I(x,u) = f(p(x|u)) − f(p(x))*
    - E.g. for Kalman Filter this is given by the change in the covariance matrix after the prediction step $\overline{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$
  - Choose the action that maximises the information gain, balanced against the cost of that action

Example from Thrun et al 2005, fig 17.4

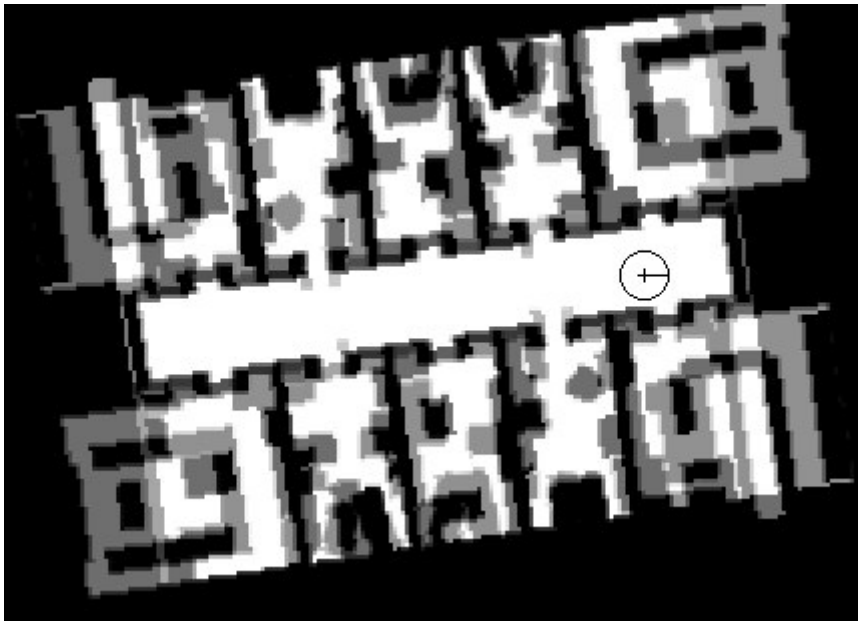
Active Localisation

Example from Thrun et al 2005, fig 17.4



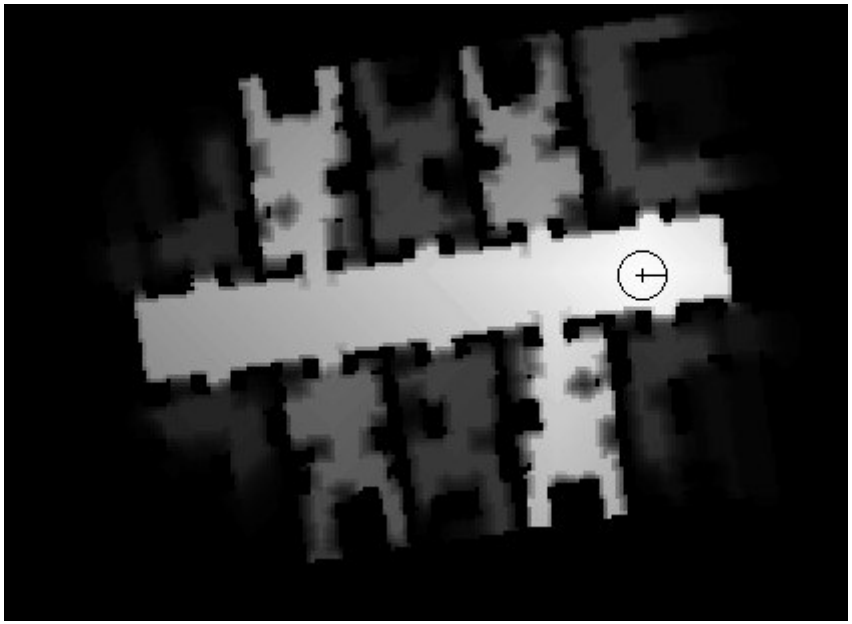
(a) Belief distribution (has two modes)

Active Localisation



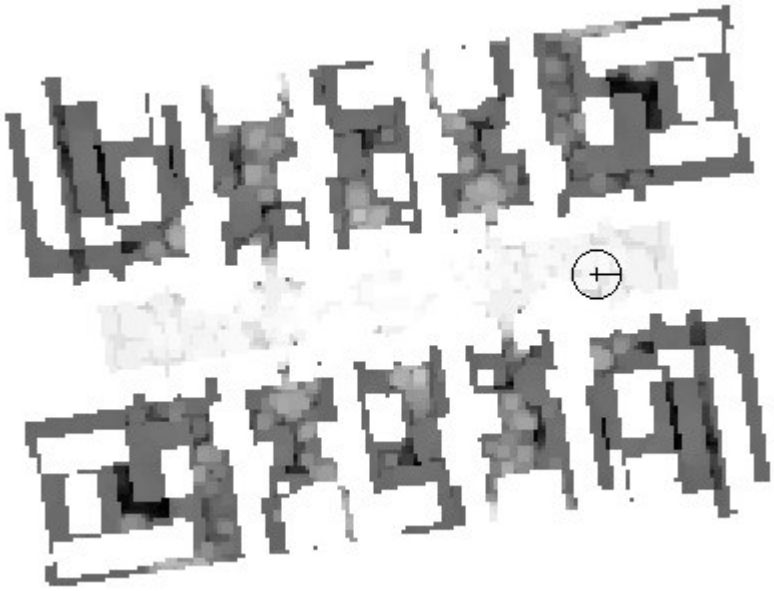
(b) Occupancy probability in robot coordinates – superimpose maps for each possible location, weighted by their respective probability.

Active Localisation



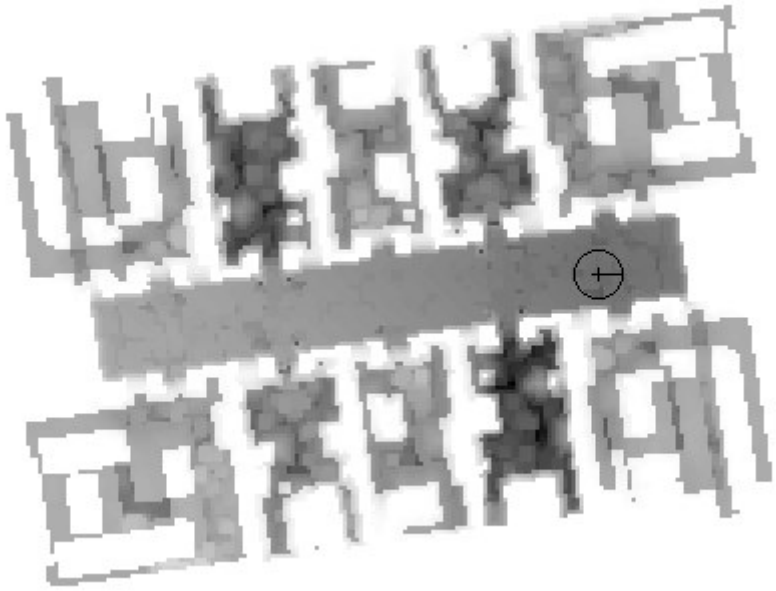
(c) Expected costs of motion – nearby locations cost less, and getting into rooms costs more.

Active Localisation



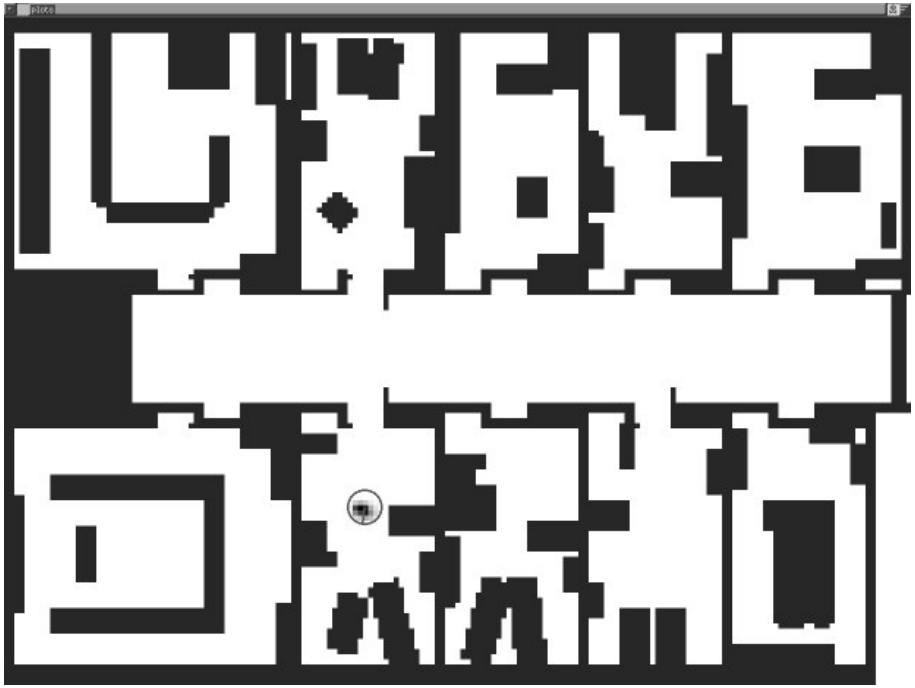
(d) Expected information gain in robot coordinates.
Rooms are much more informative than corridor.

Active Localisation



(e) Gain plus costs (the darker, the better)

Active Localisation



(f) Final belief after active localization

Active Mapping

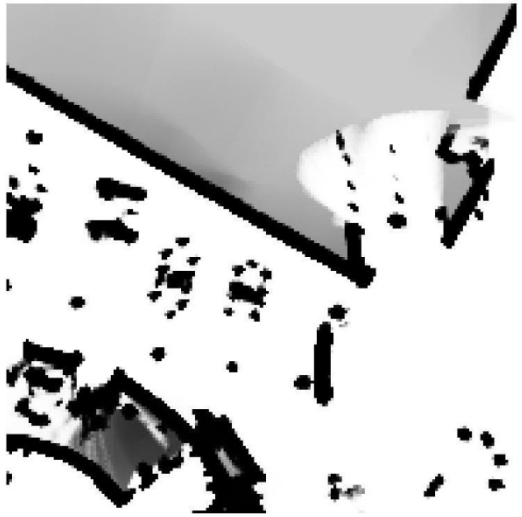
- Try to actively choose a control action **u** that will decrease uncertainty about the map – i.e. optimise exploration
- E.g. for an occupancy grid map, can define the entropy for each cell m_i with occupancy probability p_i:

$$H_p(m_i) = -p_i \log p_i - (1 - p_i) \log(1 - p_i)$$

- Cells with high entropy would be good to explore next.
- Technically, want to know the potential information gain for moving to a cell, which also depends on the sensors; but in practice this is generally very similar to the entropy.
- Simple approach: use a binary information gain function by marking cells as either explored or unexplored; suitable action choice is to move to nearest unexplored frontier.

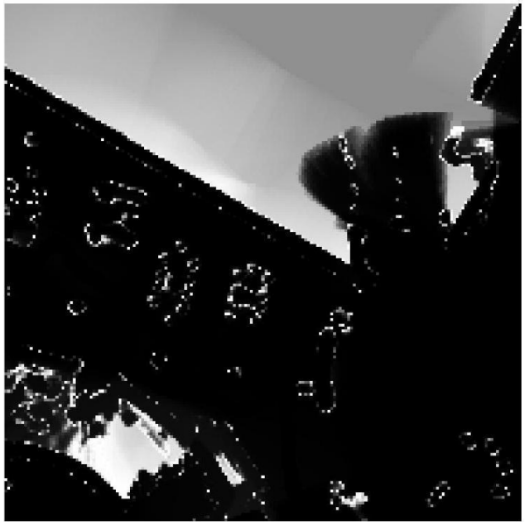
Active Mapping

(a) Map segment



(b) Entropy

(c) Exp. information gain



Active SLAM

- Try to actively choose a control action **u** that will decrease uncertainty about both location *and* map.
- E.g. closing a loop mostly reduces location uncertainty, moving into new terrain mostly reduces map uncertainty.
- Factoring of the SLAM posterior:

$$p(x, m \mid z, u) = p(x \mid z, u)\, p(m \mid x, z, u)$$

results in decomposition of the entropy:

$$H(p(x, m \mid z, u)) = H(p(x \mid z, u)) + E[H(p(m \mid x, z, u))]$$

- SLAM entropy is the sum of the path entropy and the expected entropy of the map, consider both to make action choice.

For further details see Thrun et al. (2005) or:
http://www.acfr.usyd.edu.au/education/summerschool.shtml

References:

T. Bailey and H. Durrant-Whyte Simultaneous Localisation and Mapping (SLAM): Part II State of the Art

Sebastian Thrun, Wolfram Burgard and Dieter Fox, “Probabilistic Robotics”, MIT Press, Cambridge MA, 2005

Hauke Strasdat, J. M. M. Montiel and Andrew J. Davison Real-time Monocular SLAM: Why Filter? ICRA 2010

http://www.doc.ic.ac.uk/~ajd/Publications/strasdat_etal_icra2010.pdf

P. Newman, D. Cole, and K. Ho. “Outdoor SLAM using visual appearance and laser ranging”. In *IEEE International Conference on Robotics and Automation*, 2006

M. Cummins and P. Newman (2008) “FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance” *International Journal of Robotics Research* 27:647-665, 2008