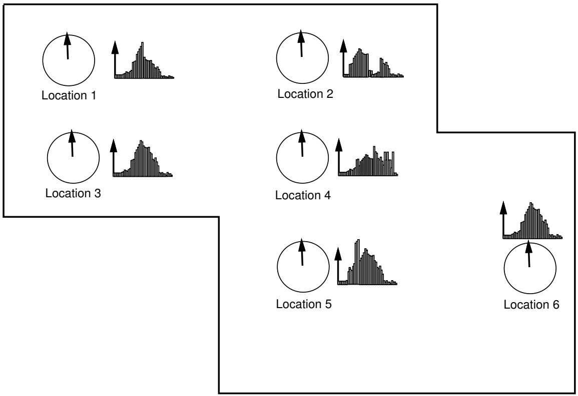
Robotics

Lecture 7: Simultaneous Localisation and Mapping (SLAM)

See course website `http://www.doc.ic.ac.uk/~ajd/Robotics/` for up to date information.

Andrew Davison
Department of Computing
Imperial College London

Review: Practical 6



- Need repeatable spin and measurement — but probably not a measurement at every degree. Use velocity control to spin the motor continuously with a loop checking the encoder readings?
- Recognising orientation too will be computationally costly without invariant descriptors.

Simultaneous Localisation and Mapping

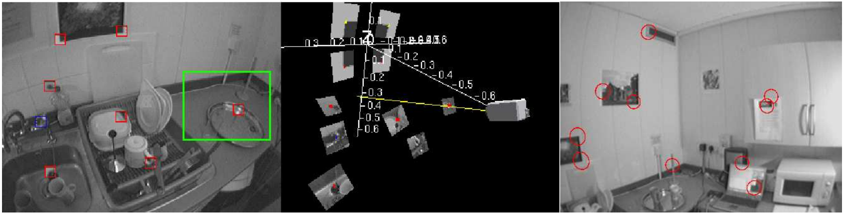
- A fundamental problem in mobile robotics, and providing some solutions is one of the main successes of probabilistic robotics.

A body with quantitative sensors
moves through a previously unknown,
static environment, mapping it
and calculating its egomotion.

- When do we need SLAM?
  - When a robot must be truly autonomous (no human input).
  - When little or nothing is known in advance about the environment (no prior map).
  - When we can't or don't want to place artificial beacons, or use GPS.
  - And when the robot actually needs to know where it is.
- In SLAM we build a map incrementally, and localise with respect to that map as it grows and is gradually refined.

Features for SLAM

- Most SLAM algorithms make maps of natural scene *features*.
- Laser/sonar: wall segments, planes, corners, etc.
- Vision: salient point features, lines, textured surfaces.

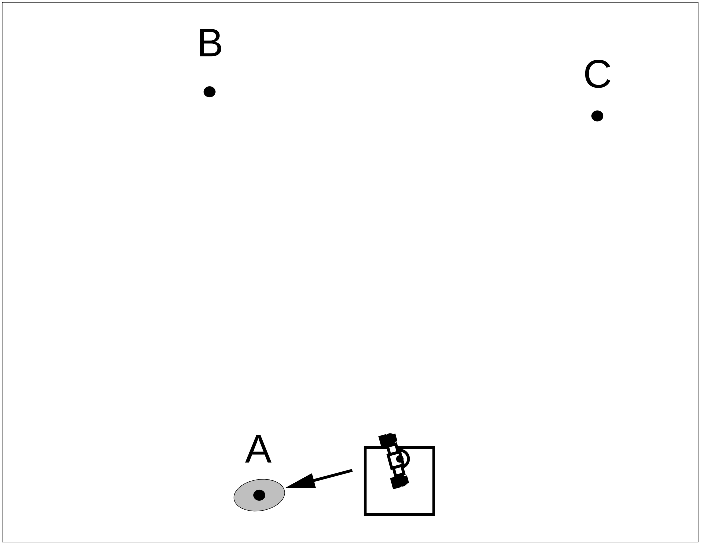


- Features should be distinctive and easily recognisable from different viewpoints to enable reliable matching (also called correspondence or data association).

Propagating Uncertainty

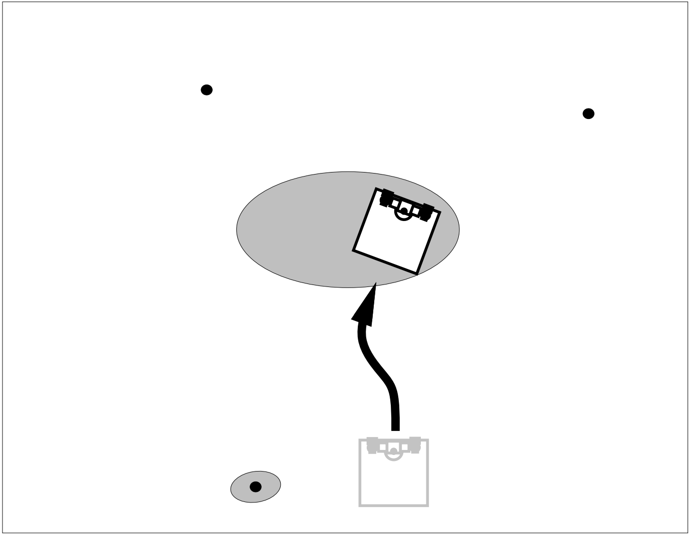
- Because we must both map and localise at the same time SLAM seems like a chicken and egg problem — but we can make progress if we assume the robot is the only thing that moves.
- Main assumption in most SLAM systems: the world, (or at least a large fraction of the mappable things in it) is static.
- With this assumption, we just go ahead and extend probabilistic estimation (from just the robot state as in MCL) to the features of the map as well. In SLAM we store and update a joint distribution over the states of both the robot and the mapped world... and if the data is good enough it just works.
- New features are gradually discovered as the robot explores so the dimension of this joint estimation problem will grow.

Simultaneous Localisation and Mapping



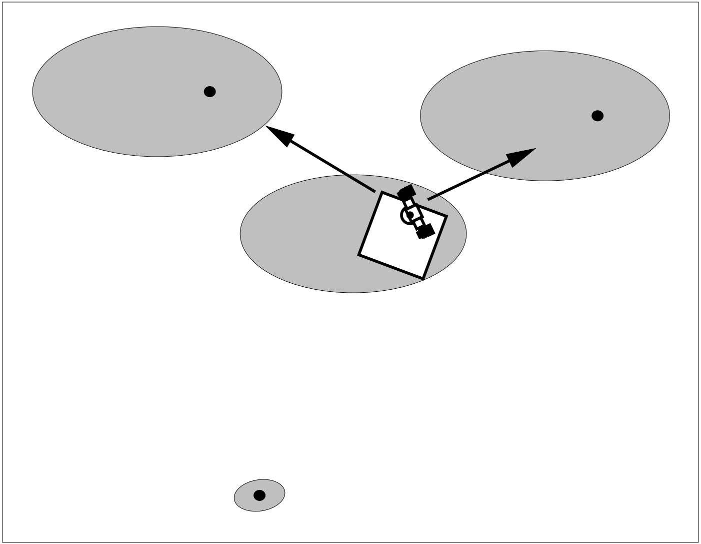
(a) Robot start (zero uncertainty); first measurement of feature A.

Simultaneous Localisation and Mapping



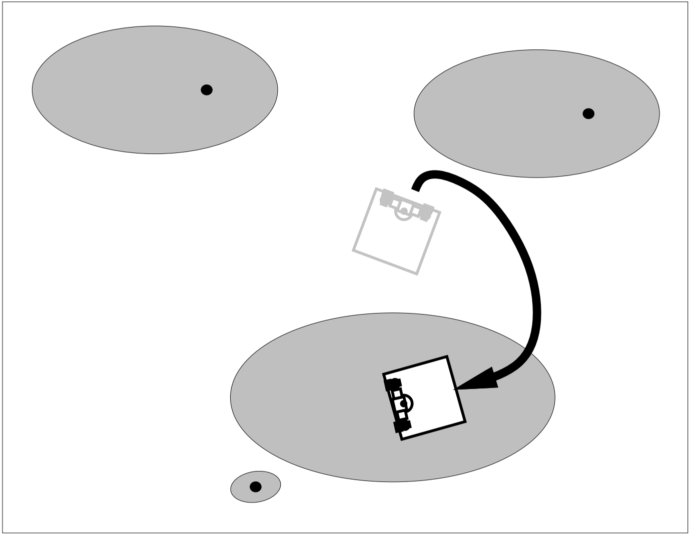
(b) Robot drives forwards (uncertainty grows).

Simultaneous Localisation and Mapping



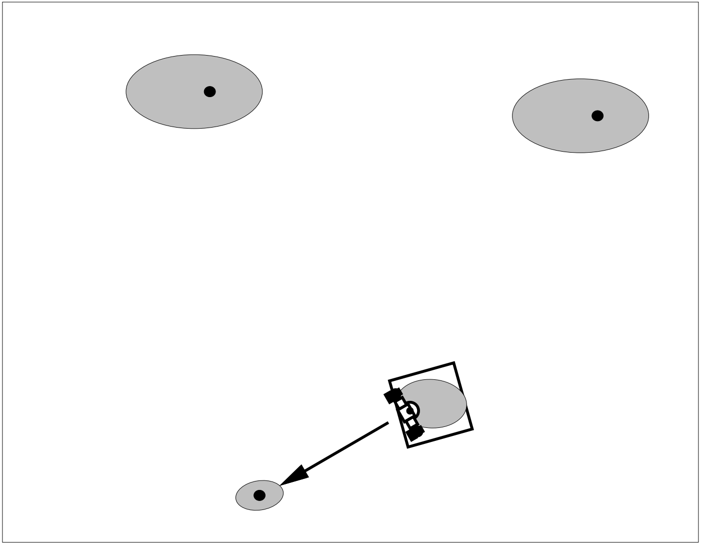
(c) Robot initialises B and C: they inherit its uncertainty + a little more.

Simultaneous Localisation and Mapping



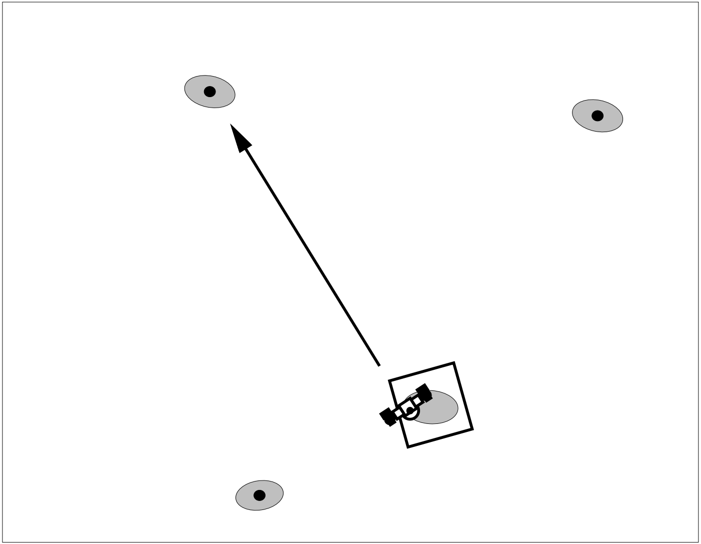
(d) Robot drives back towards starting position (uncertainty grows more).

Simultaneous Localisation and Mapping



(e) Robot re-measures A; a mini *loop closure*! Uncertainty shrinks.

Simultaneous Localisation and Mapping



(f) Robot re-measures B; note that uncertainty of C also shrinks.

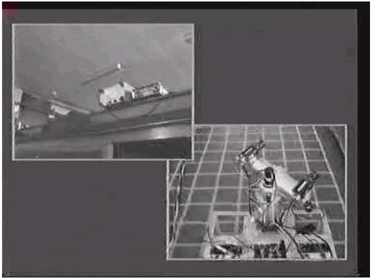
SLAM with Joint Gaussian Uncertainty

- The most common and efficient way to represent the high-dimensional probability distributions we need to propagate in SLAM is as a joint Gaussian distribution. Updates can be made via the Extended Kalman Filter.
- PDF represented with state vector and covariance matrix.

$$\hat{\mathbf{x}} = \begin{pmatrix} \hat{\mathbf{x}}_v \\ \hat{\mathbf{y}}_1 \\ \hat{\mathbf{y}}_2 \\ \vdots \end{pmatrix} \quad , \quad \mathrm{P} = \begin{bmatrix} \mathrm{P}_{xx} & \mathrm{P}_{xy_1} & \mathrm{P}_{xy_2} & \dots \\ \mathrm{P}_{y_1x} & \mathrm{P}_{y_1y_1} & \mathrm{P}_{y_1y_2} & \dots \\ \mathrm{P}_{y_2x} & \mathrm{P}_{y_2y_1} & \mathrm{P}_{y_2y_2} & \dots \\ \vdots & \vdots & \vdots & \end{bmatrix}$$

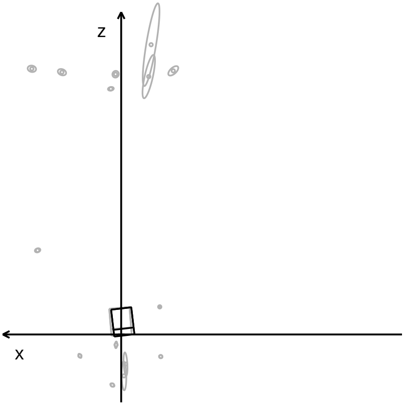
- The state vector contains the robot state and all the feature states. $\mathbf{x}_v$ is robot state, e.g. (x, y, θ) in 2D; y_i is feature state, e.g. (X, Y) in 2D.

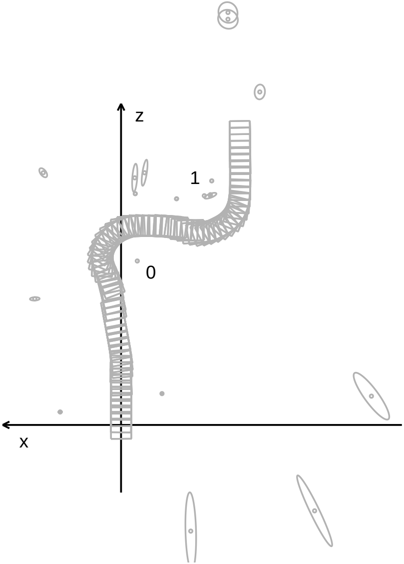
SLAM Using Active Vision

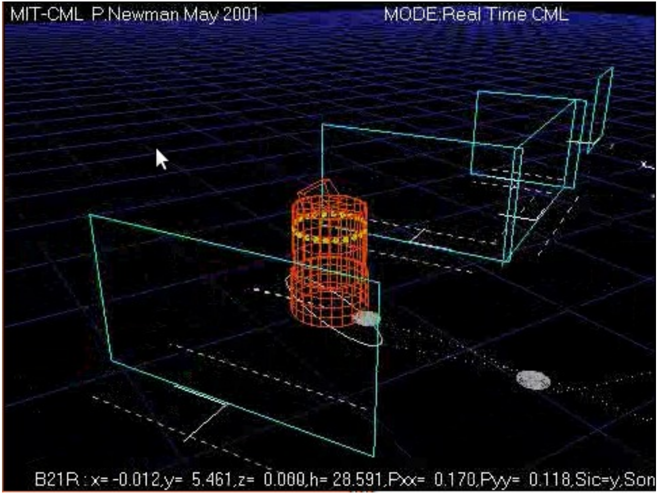
- Stereo active vision; 3-wheel robot base.
- Automatic fixated active mapping and measurement of arbitrary scene features.
- Sparse mapping.

SLAM Using Active Stereo Vision



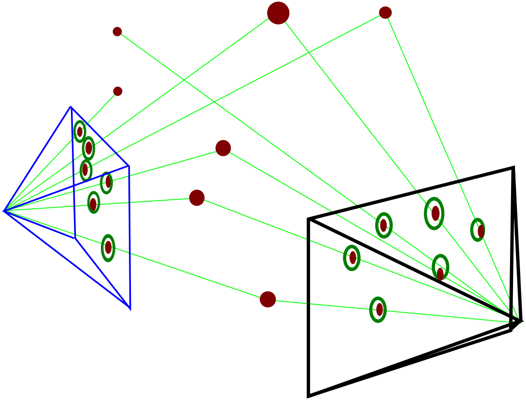


SLAM with Ring of Sonars



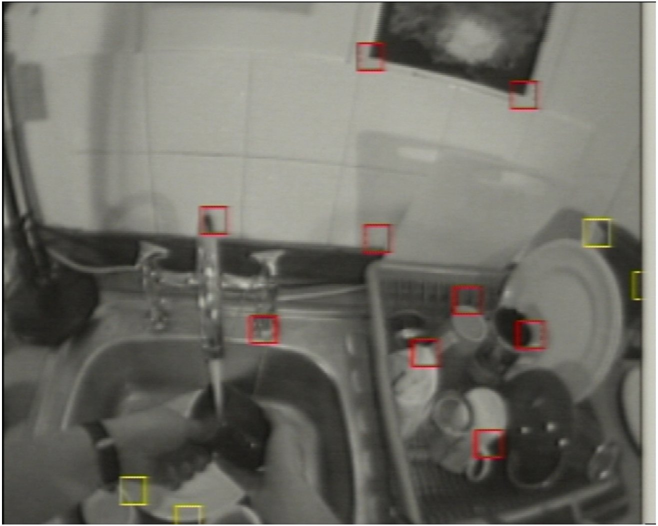
Newman, Leonard, Neira and Tardós, ICRA 2002

SLAM with a Single Camera – the Principle



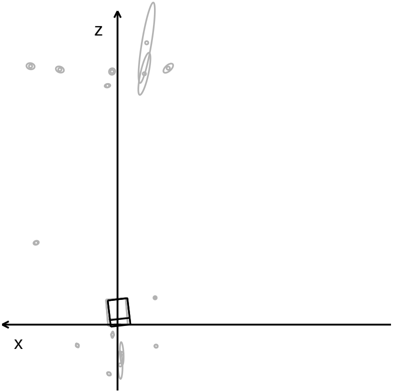
- *Frontend*: keypoints are detected in successive images, and associated with a 3D point in the world.
- *Backend*: the pose of the camera(s) and 3D points that best explain these keypoint measurements are estimated.

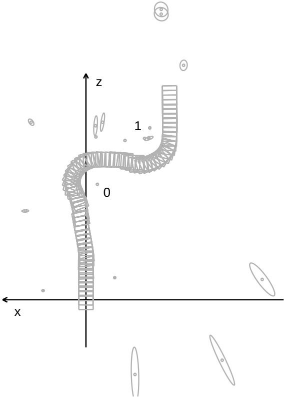
SLAM with a Single Camera: MonoSLAM



Davison, ICCV 2003; Davison, Molton, Reid, Stasse, PAMI 2007.

Limits of Metric SLAM

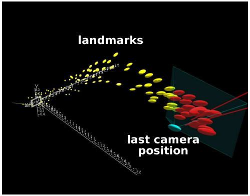




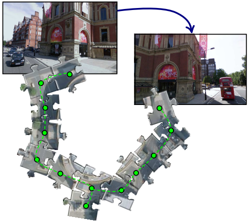
Purely metric probabilistic SLAM is limited to small domains due to:

- Poor computational scaling of probabilistic filters.
- Growth in uncertainty at large distances from map origin makes representation of uncertainty inaccurate.
- *Data Association* (matching features) gets hard at high uncertainty.

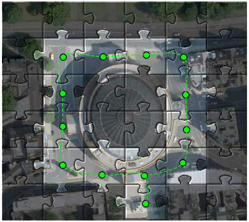
Large Scale Localisation and Mapping



Local Metric



Place Recognition



Global Optimisation

Practical modern solutions to large scale mapping follow a *metric/topological* approach which approximates full metric SLAM. They need the following elements:

- Local metric mapping to estimate trajectory and make local maps.
- Place recognition, to perform 'loop closure' or relocalise the robot when lost.
- Map optimisation/relaxation to optimise a map when loops are closed.

Global Topological: ‘Loop Closure Detection’

- One very effective way to detect when an ‘old’ place is revisited is to save images at regular intervals and use an image retrieval approach (where each image is represented using a Visual Bag of Words which has very much the same character as our invariant sonar descriptors).
- Angeli *et al.*, IEEE Transactions on Robotics 2008.

Pure Topological SLAM

- In fact we can make an interesting SLAM system using *only* place recognition. Topological SLAM with a graph-based representation.
- We simply keep a record of places we have visited and how they connect together, without any explicit geometry information.
- Adapted to symbolic planning and navigation.

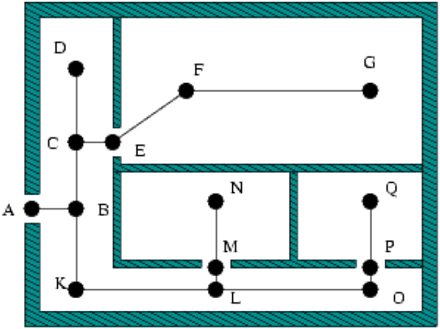
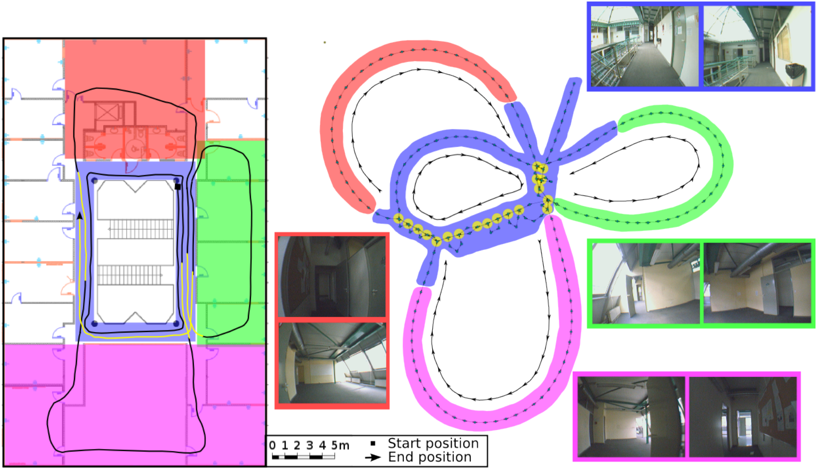


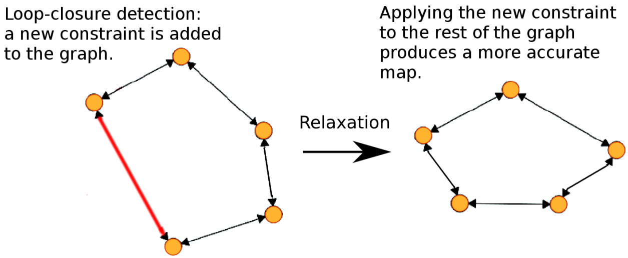
Figure : Topological representation

Indoor Topological Map

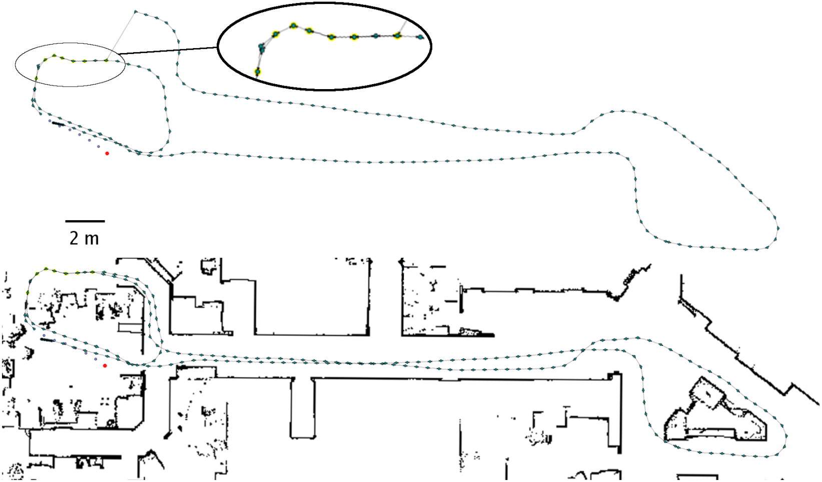


Adding Metric Information to the Graph Edges

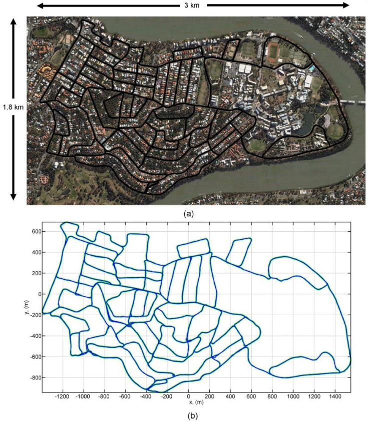
- The edges between linked nodes are annotated with relative motion information; could be from local mapping or purely incremental information like odometry or visual odometry.
- Apply **pose graph optimisation (relaxation)** algorithm, which computes the set of node positions which is maximally probable given both the metric and topological constraints.
- Pose graph optimisation only has an effect when there are loops in the graph.



Map Relaxation: Good Odometry, One Loop Closure



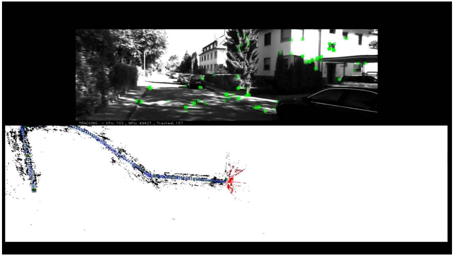
Simple Large-Scale SLAM: RATSLAM



Milford and Wyeth, 2007.
`http://www.youtube.com/watch?v=-0XSUi69Yvs`

- Very simple ‘visual odometry’ gives rough trajectory.
- Simple visual place recognition provides *many* loop closures.
- Map relaxation/optimisation to build global map.

More Accurate Large-Scale Monocular SLAM: ORB-SLAM



ORB-SLAM: Tracking and Mapping Recognizable Features.
Raul Mur-Artal and Juan D. Tardos, arXiv:1502.00956 (2015)
`https://www.youtube.com/watch?v=8DISRmsO2YQ`

- Very accurate 'visual odometry' trajectory.
- Visual place recognition based on 2D image features with binary descriptors for very fast matching.
- Pose graph/map optimisation for global consistency.

More Information about SLAM

If you want to find out more about SLAM there is plenty of good information and open source software available online; e.g.:

- Visual/monocular SLAM: ORB-SLAM, LSD-SLAM, OKVIS, SceneLib2, PTAM.
- Pose Graph Optimisation: g2o, Ceres, iSAM.
- Place recognition: FAB-MAP.