Probabilistic Robotics
Graph SLAM

MSc course Artificial Intelligence 2017
https://staff.fnwi.uva.nl/a.visser/education/ProbabilisticRobotics/

Arnoud Visser
Intelligent Robotics Lab
Informatics Institute
Universiteit van Amsterdam
A.Visser@uva.nl

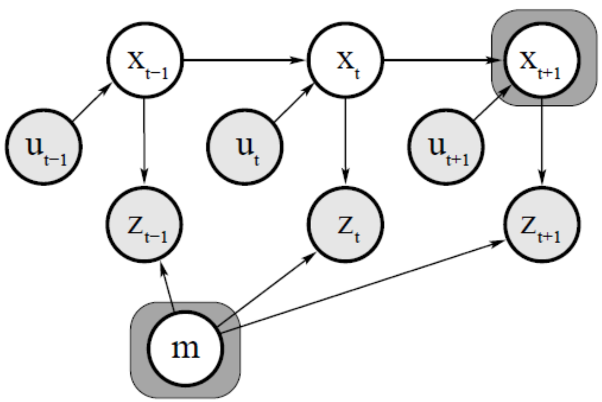
Images courtesy of Sebastian Thrun, Wolfram Burghard, Dieter Fox, Michael Montemerlo, Dick Hähnel, Pieter Abbeel and others.

Simultaneous Localization and Mapping

A robot acquires a map while localizing itself relative to this map.

Online SLAM problem

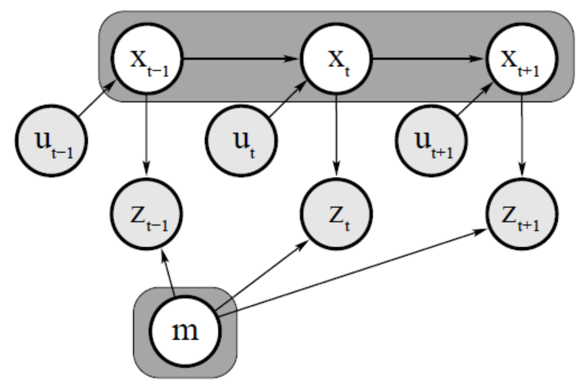
$$p(x_t, m \mid z_{1:t}, u_{1:t})$$



Estimate map *m* and current position x_t

Full SLAM problem

$$p(x_{1:t}, m \mid z_{1:t}, u_{1:t})$$



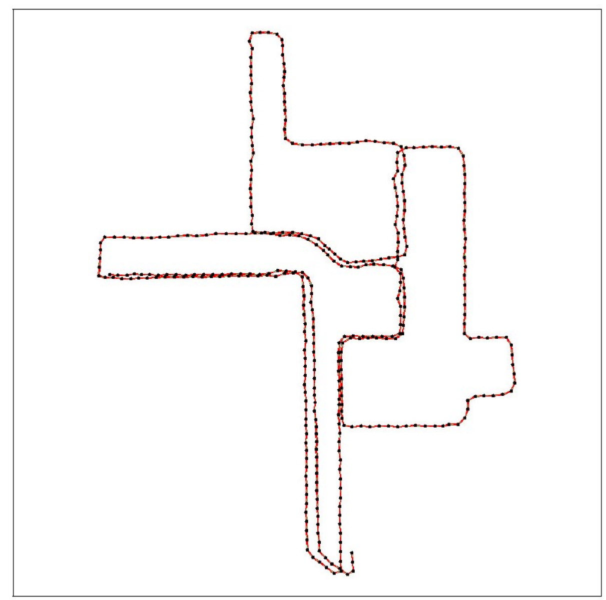
Estimate map *m* and driven path $x_{1:t}$

Graph SLAM

GraphSLAM extends the state vector *y* with the path $x_{0:t}$

$$y_{0:t} = \left(x_0 x_1 \cdots x_t m_{1,x} m_{1,y} s_1 \cdots m_{N,x} m_{N,y} s_N\right)^T$$

Example: Groundhog in abandoned mine:
every 5 meters a local map



State estimate

GraphSLAM requires inference to estimate the state

$$\tilde{\mu}_{0:t} = \tilde{\Omega}^{-1}\tilde{\xi}$$

The state is estimated from the *information matrix Ω* and *vector ξ*, the canonical representation of the *covariance* and *mean.*

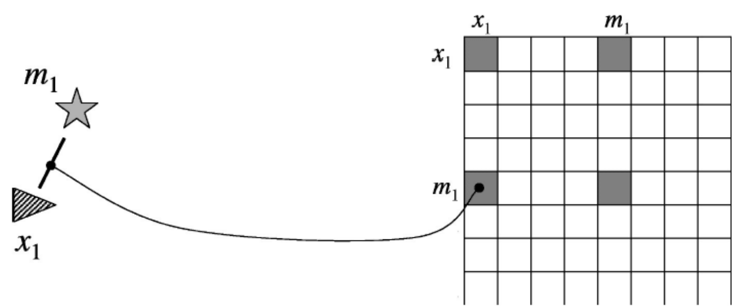
Benefits:

- Uncertainty is easy represented (*Ω=0)*
- Information can be integrated by addition, without direct inference

The state estimated μ_t requires inversion of the *information matrix Ω,* which is done off-line

Acquisition of the *information matrix*

The observation of a landmark m_1 introduces an constraint:



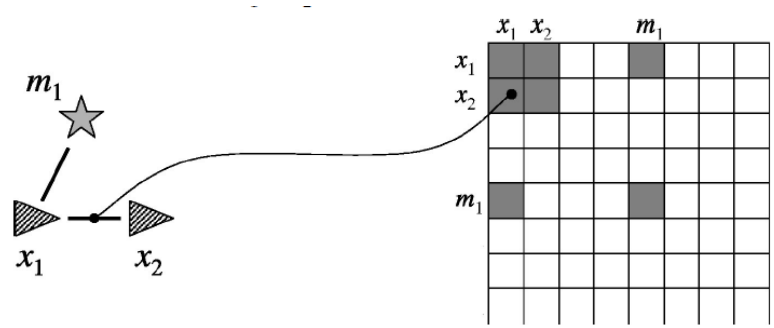
The constraint is of the type:

$$(z_t^i - h(x_t, m_j))^T Q_t^{-1} (z_t^i - h(x_t, m_j))$$

Where $h(x_t,m_j)$ is the measurement model and Q_t the covariance of the measurement noise.

Acquisition of the *information matrix*

The movement of the robot from x_1 to x_2 also introduces an constraint:



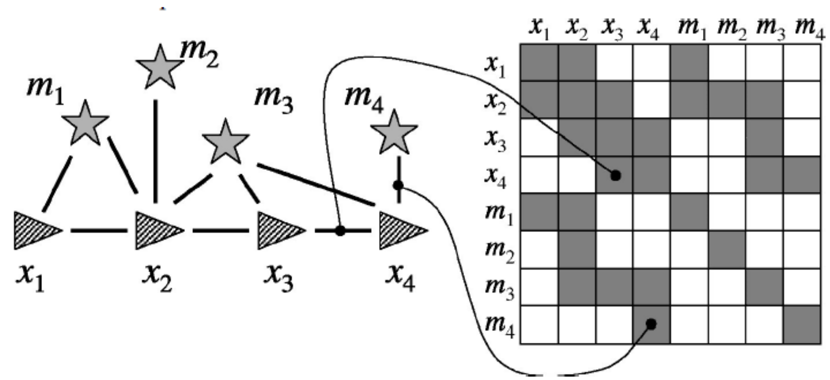
The constraint is of the type:

$$(x_t - g(u_t, x_{t-1}))^T R_t^{-1} (x_t - g(u_t, x_{t-1}))$$

Where $g(u_t,x_{j-1})$ is the motion model and R_t the covariance of the motion noise.

Acquisition of the *information matrix*

After several steps, a dependence graph appears with several constraints:



The resulting *information matrix* is quite sparse.

The sum of all constraints in the graph has the form:

$$J_{\text{GraphSLAM}} = x_0^T \, \Omega_0 \, x_0 + \sum_t (x_t - g(u_t, x_{t-1}))^T \, R_t^{-1} \, (x_t - g(u_t, x_{t-1})) + \sum_t \sum_i (z_t^i - h(y_t, c_t^i, i))^T \, Q_t^{-1} \, (z_t^i - h(y_t, c_t^i, i))$$

Simplifying acquisition

- By a Taylor expansion of the motion and measurement model, the equations can be approximated:

$$\Omega \longleftarrow \Omega + \begin{pmatrix} 1 \\ -G_t \end{pmatrix} R_t^{-1} \,(1 \quad -G_t)$$

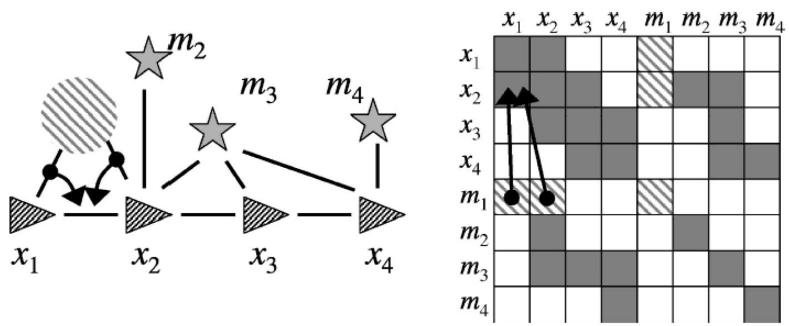
$$\xi \longleftarrow \xi + \begin{pmatrix} 1 \\ -G_t \end{pmatrix} R_t^{-1} \,[g(u_t, \mu_{t-1}) + G_t\, \mu_{t-1}]$$

$$\Omega \longleftarrow \Omega + H_t^{iT}\; Q_t^{-1}\; H_t^i$$

$$\xi \longleftarrow \xi + H_t^{iT}\; Q_t^{-1}\, [z_t^i - h(\mu_t, c_t^i, i) - H_t^i \mu_t]$$

Reducing the dependence graph

Removal of the observation of a landmark m_1 changes the constraint between x_1 to x_2 :



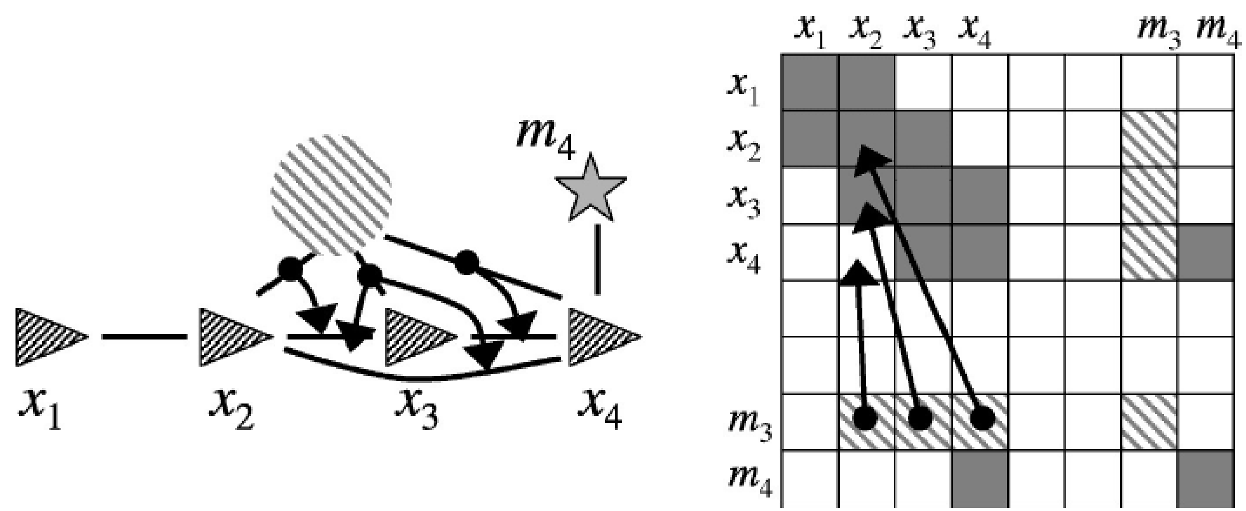
The constraint is changed by the following subtraction:

$$\tilde{\Omega} = \Omega_{x_{0:t},x_{0:t}} - \sum_j \Omega_{x_{0:t},j} \Omega_{j,j}{}^{-1} \sum_j \Omega_{j,x_{0:t}}$$

This is a form of *variable elimination algorithm* for matrix inversion

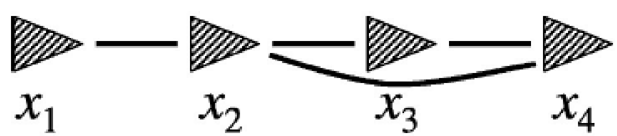
Reducing the dependence graph

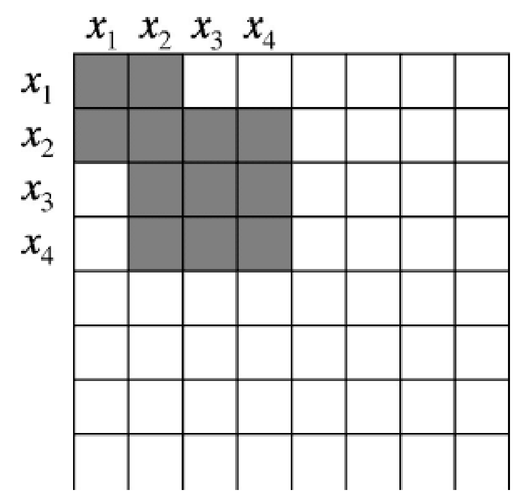
Removal of the observation of a landmark m_2 introduces a new constraint between x_2 to x_4:



Reducing the dependence graph

The final result:





The resulting *information matrix* is much smaller.

This reduction can be done in time linear in size *N*

Updating the full state estimate from the path

There is now an estimate of the path robot

$$\tilde{\mu}_{0:t} = \tilde{\Omega}_{0:t}^{-1} \tilde{\xi}$$

This requires to solve a system of linear equations, which is not linear in size *t* due to cycles (loop closures!).

When found, the map can be recovered. For each landmark m_j:

$$\mu_j = \Omega_{j,j}^{-1} (\xi_j + \Omega_{j,0:t} \tilde{\mu}_{0:t})$$

In addition, an estimate of the covariance $\Sigma_{0:t}$ over the robot path is known (but not over the full state *y*)

Full Algorithm

The previous steps should be iterated to get a reliable state estimate μ:

Algorithm GraphSLAM_known_correspondence($u_{1:t}, z_{1:t}, c_{1:t}$)**:**

$\mu_{0:t} =$ **GraphSLAM_initialize**($u_{1:t}$)

repeat

$\Omega, \xi =$ **GraphSLAM_linearize**($u_{1:t}, z_{1:t}, c_{1:t}, \mu_{0:t}$)

$\tilde{\Omega}, \tilde{\xi} =$ **GraphSLAM_reduce**(Ω, ξ)

$\mu, \Sigma_{0:t} =$ **GraphSLAM_solve**($\tilde{\Omega}, \tilde{\xi}, \Omega, \xi$)

until convergence

return μ

Full Algorithm

The algorithm can be extended for unknown correspondences:

Algorithm GraphSLAM($u_{1:t}, z_{1:t}$):

initialize all c_t^i *with a unique value*

$\mu_{0:t} =$ **GraphSLAM_initialize**$(u_{1:t})$

$\Omega, \xi =$ **GraphSLAM_linearize**$(u_{1:t}, z_{1:t}, c_{1:t}, \mu_{0:t})$

$\tilde{\Omega}, \tilde{\xi} =$ **GraphSLAM_reduce**(Ω, ξ)

$\mu, \Sigma_{0:t} =$ **GraphSLAM_solve**$(\tilde{\Omega}, \tilde{\xi}, \Omega, \xi)$

repeat

  for each pair of non-corresponding features m_j, m_k *do*

    $\pi_{j=k} =$ **GraphSLAM_correspondence_test** $(\Omega, \xi, \mu, \Sigma_{0:t}, j, k)$

    if $\pi_{j=k} > \chi$ *then*

      for all $c_t^i = k$ *set* $c_t^i = j$

      $\Omega, \xi =$ **GraphSLAM_linearize**$(u_{1:t}, z_{1:t}, c_{1:t}, \mu_{0:t})$

      $\tilde{\Omega}, \tilde{\xi} =$ **GraphSLAM_reduce**(Ω, ξ)

      $\mu, \Sigma_{0:t} =$ **GraphSLAM_solve**$(\tilde{\Omega}, \tilde{\xi}, \Omega, \xi)$

    endif

  endfor

until no more pair m_j, m_k *found with* $\pi_{j=k} < \chi$

return μ

Correspondence test

Based on the probability that m_j corresponds to m_k:

Algorithm GraphSLAM_correspondence_test($\Omega,\ \xi,\ \mu,\ \Sigma_{0:t},\ j,\ k$**):**

$$\Omega_{[j,k]} = \Omega_{jk,jk} - \Omega_{jk,\tau(j,k)}\ \Sigma_{\tau(j,k),\tau(j,k)}\ \Omega_{\tau(j,k),jk}$$

$$\xi_{[j,k]} = \Omega_{[j,k]}\ \mu_{j,k}$$

$$\Omega_{\Delta j,k} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}^T \Omega_{[j,k]} \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

$$\xi_{\Delta j,k} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}^T \xi_{[j,k]}$$

$$\mu_{\Delta j,k} = \Omega_{\Delta j,k}^{-1}\ \xi_{\Delta j,k}$$

$$\textit{return}\ |2\pi\ \Omega_{\Delta j,k}^{-1}|^{-\frac{1}{2}}\ \exp\left\{-\tfrac{1}{2}\ \mu_{\Delta j,k}^T\ \Omega_{\Delta j,k}^{-1}\ \mu_{\Delta j,k}\right\}$$

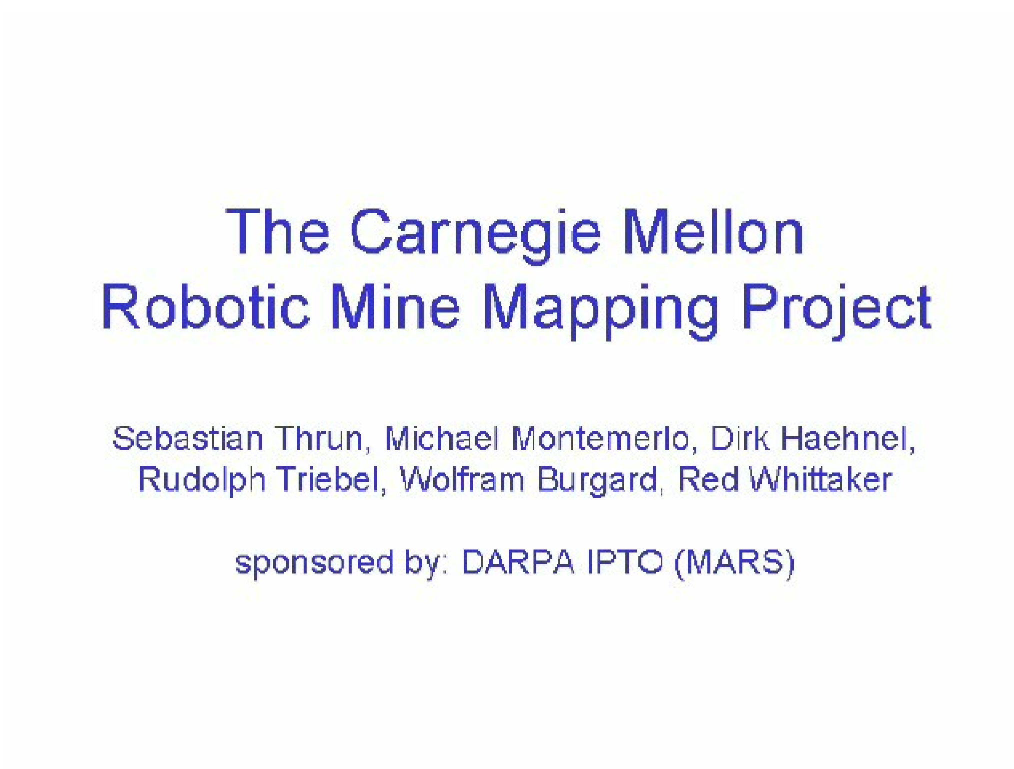
GroundHog in abandoned mine

A robot deployed in a previous flooded coal mine:

Sebastian Thrun *et al.*, Autonomous Exploration and Mapping of Abandoned Mines, IEEE Robotics and Automation Magazine 11(4), 2005.

GroundHog in abandoned mine

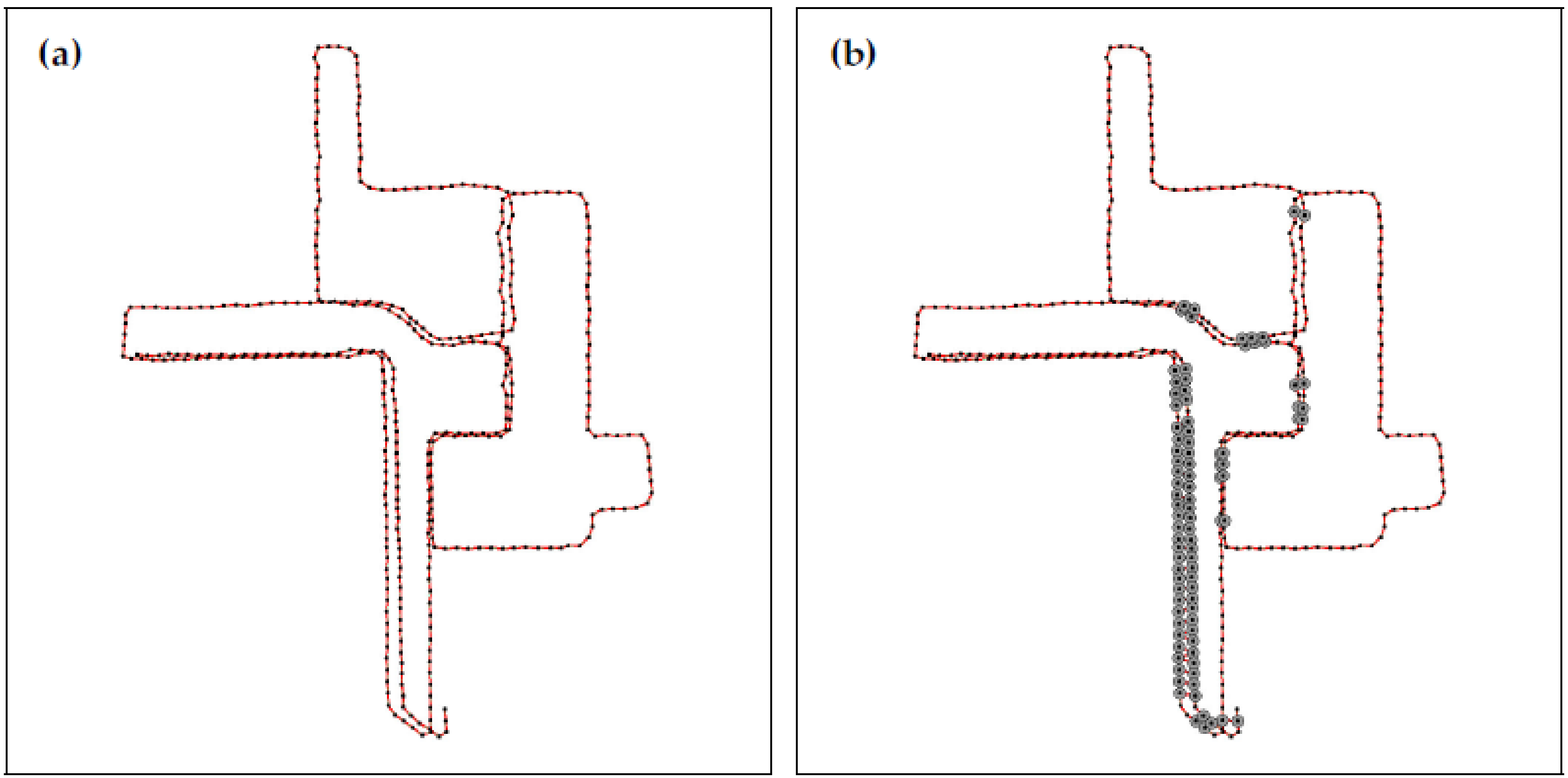
A robot created a 3D model of the coal mine:



Sebastian Thrun *et al.*, Autonomous Exploration and Mapping of Abandoned Mines, IEEE Robotics and Automation Magazine 11(4), 2005.

GroundHog in abandoned mine

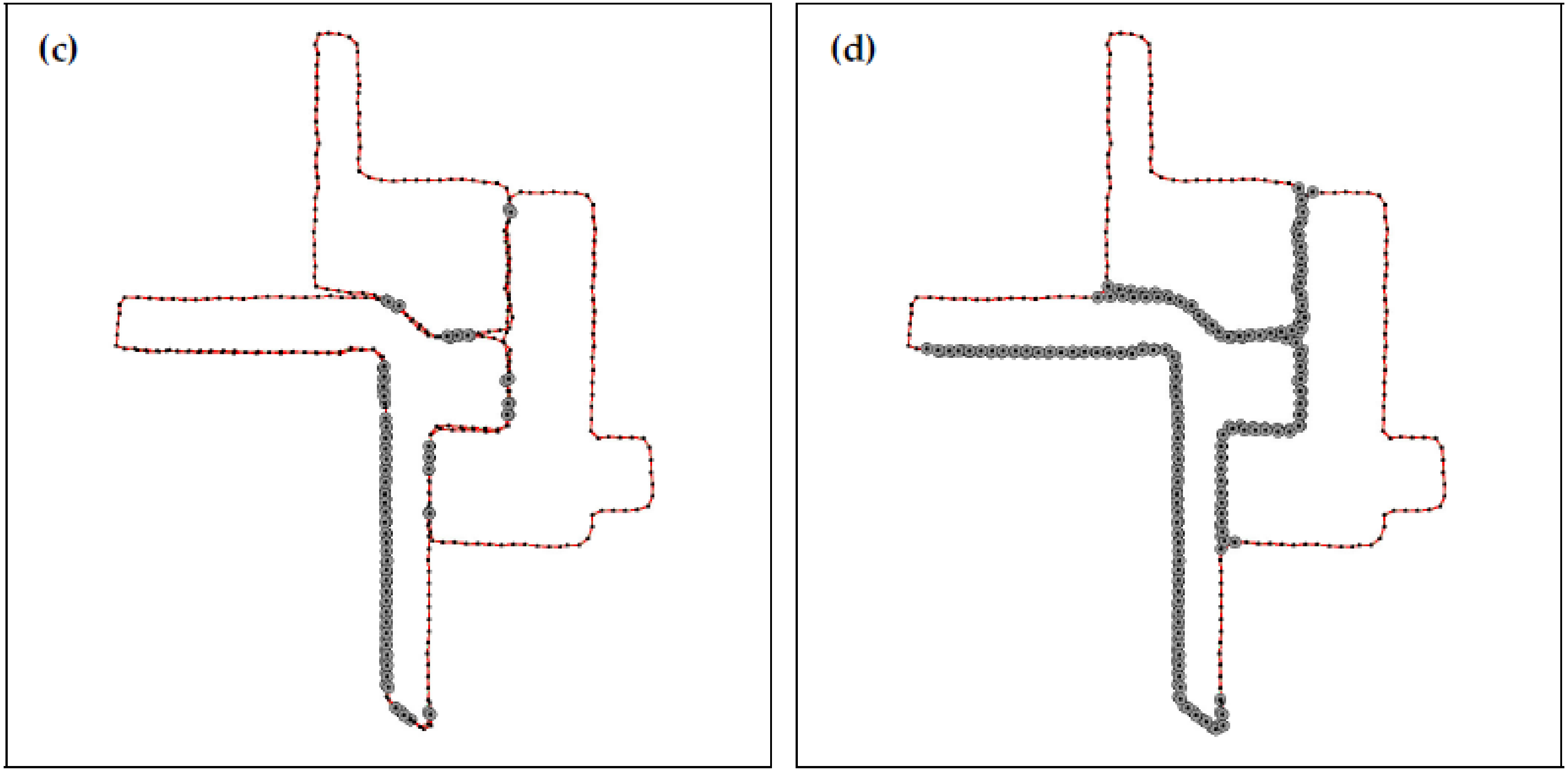
Correspondences are discovered:



Sebastian Thrun *et al.*, Autonomous Exploration and Mapping of Abandoned Mines, IEEE Robotics and Automation Magazine 11(4), 2005.

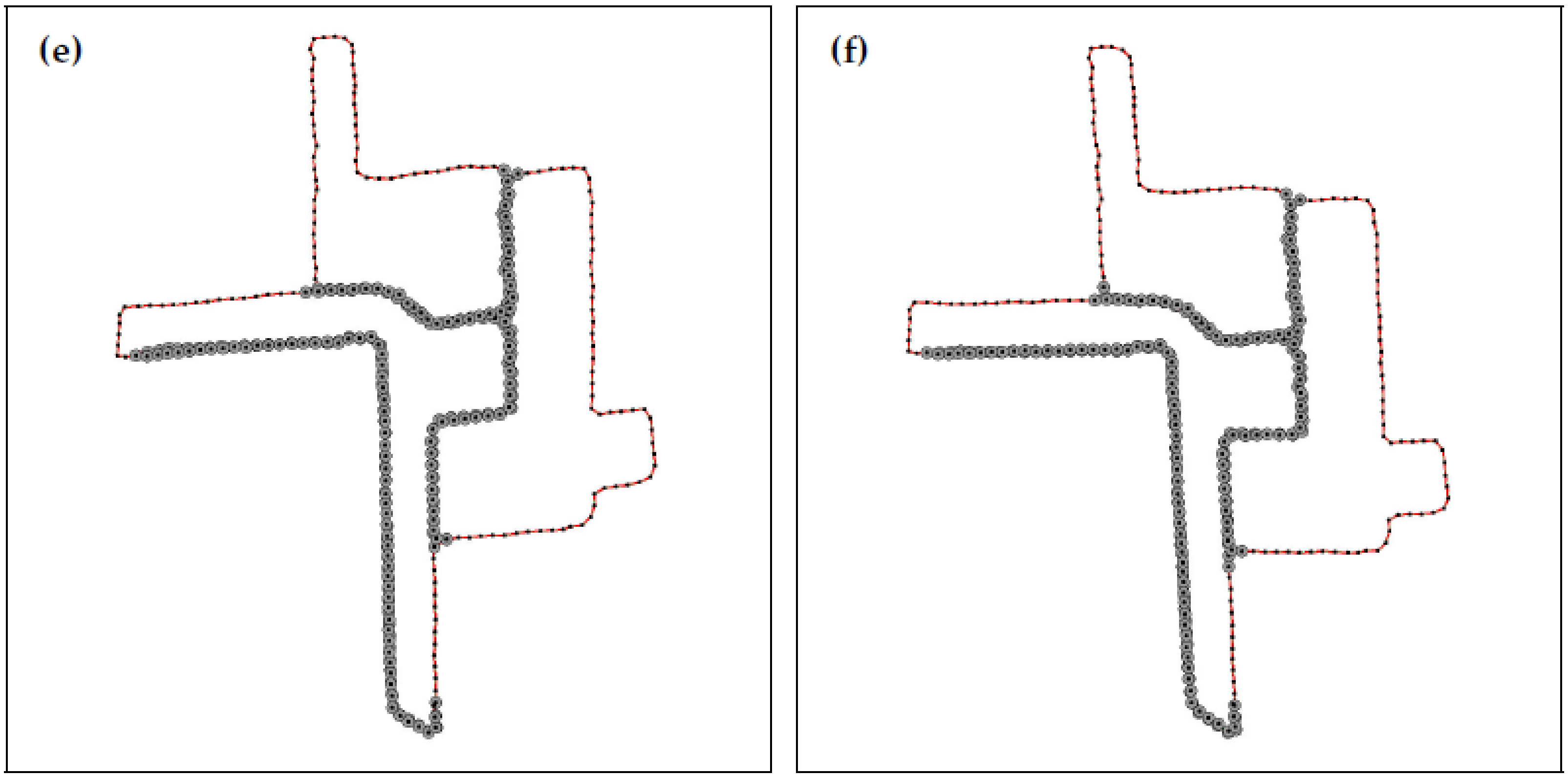
GroundHog in abandoned mine

Correspondences are propagated and dissolved:



GroundHog in abandoned mine

Iterations stops when data associations induce no further changes:



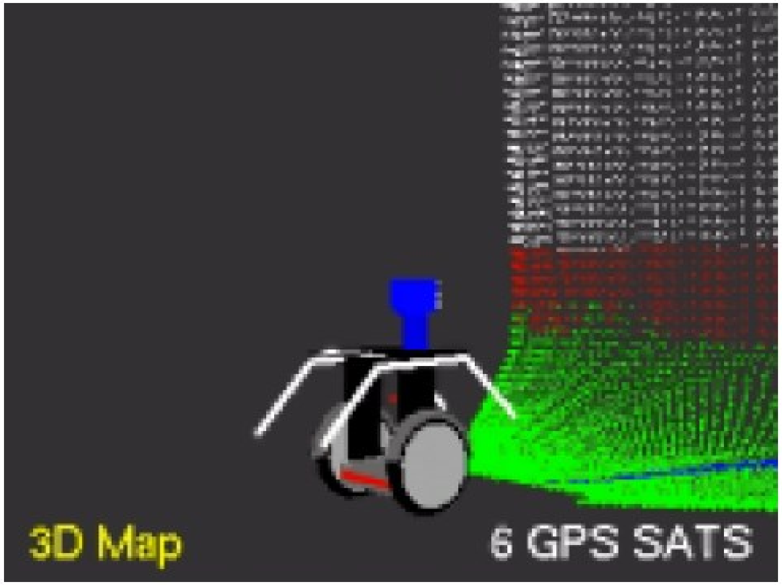
Segway RMP at Stanford

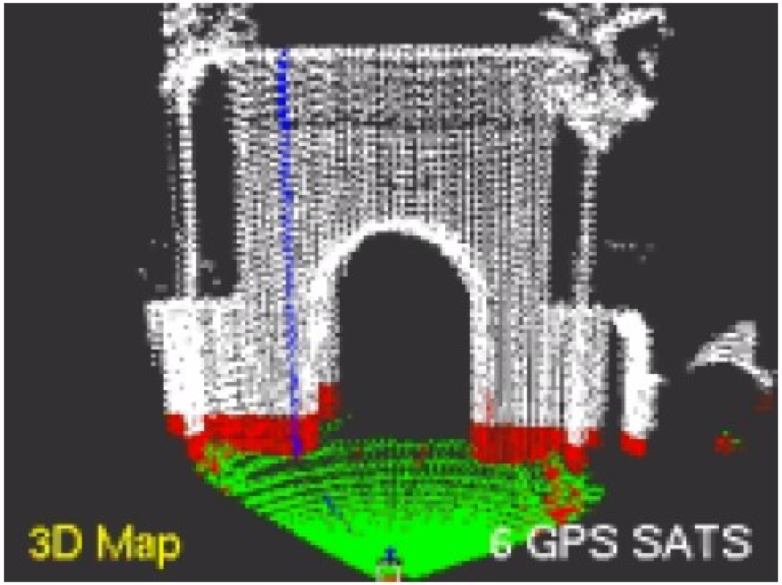
Segway exploring outdoors:

Sebastian Thrun and Micheal Montemerlo, The Graph SLAM Algorithm with Applications to Large-Scale Mapping of Urban Structures, International Journal on Robotics Research 25(5/6), p. 403-430, 2005

Segway RMP at Stanford

Segway with vertically mounted laserscanner:



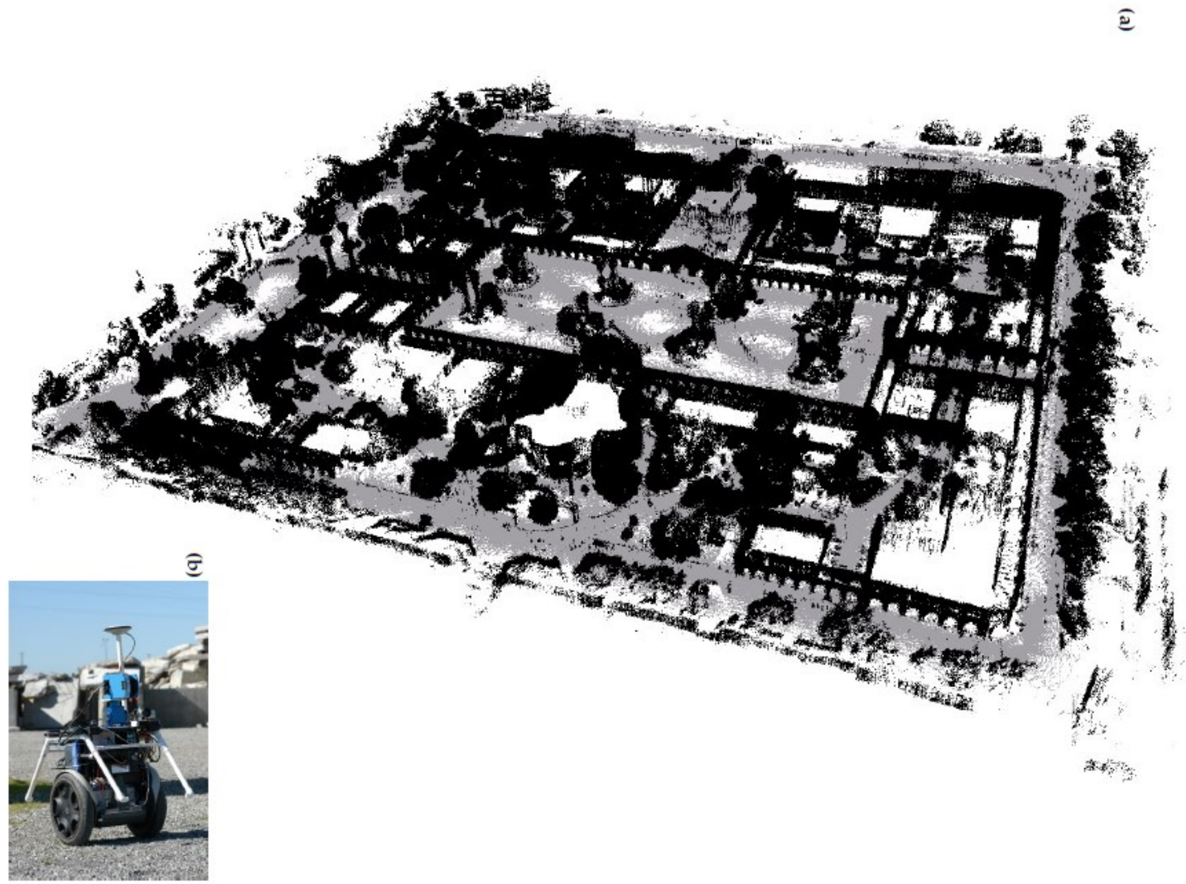


Green is ground, red obstacles, white structures above the robot

Sebastian Thrun and Micheal Montemerlo, The Graph SLAM Algorithm with Applications to Large-Scale Mapping of Urban Structures, International Journal on Robotics Research 25(5/6), p. 403-430, 2005

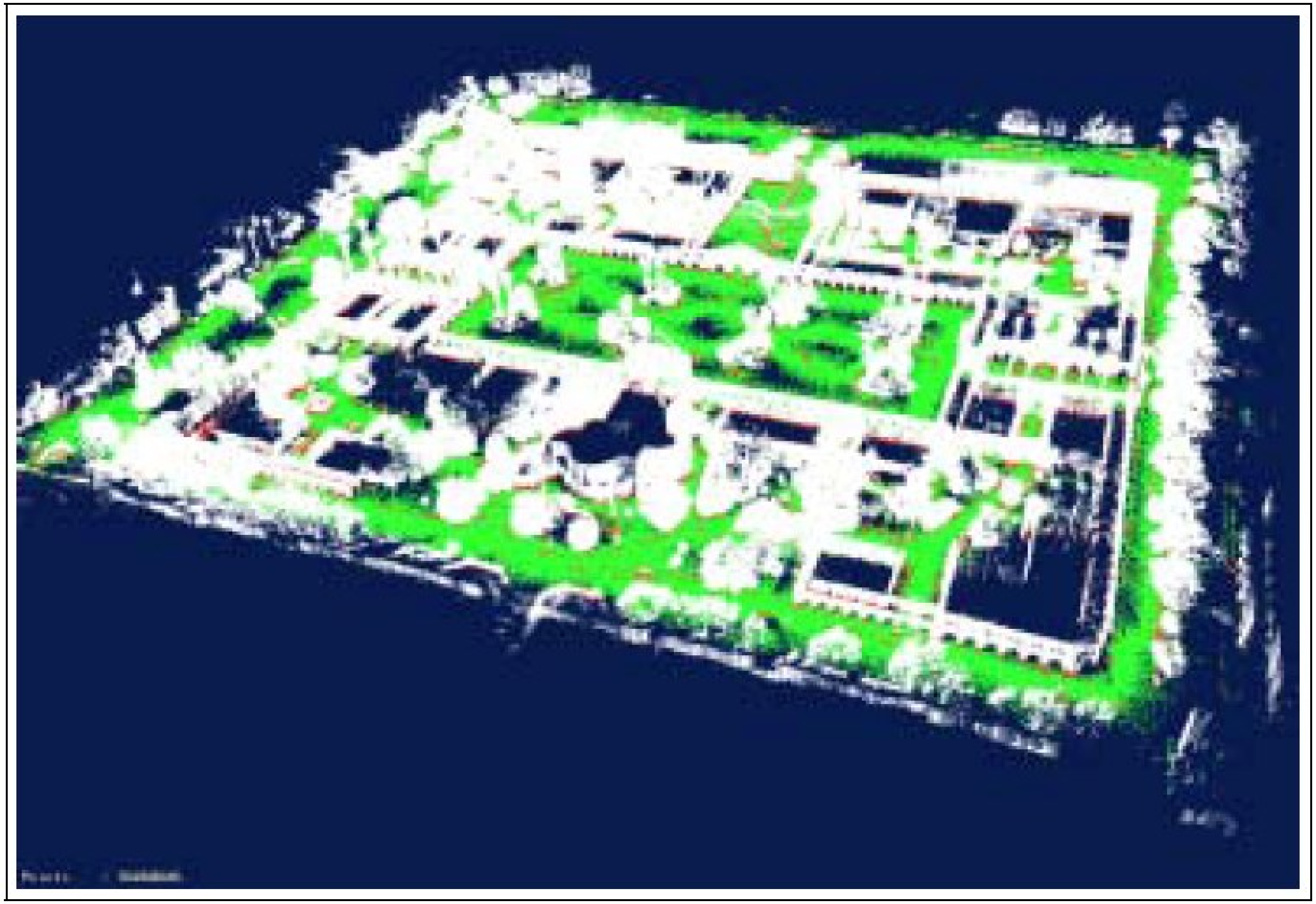
Segway RMP at Stanford

3D map of the Stanford campus:



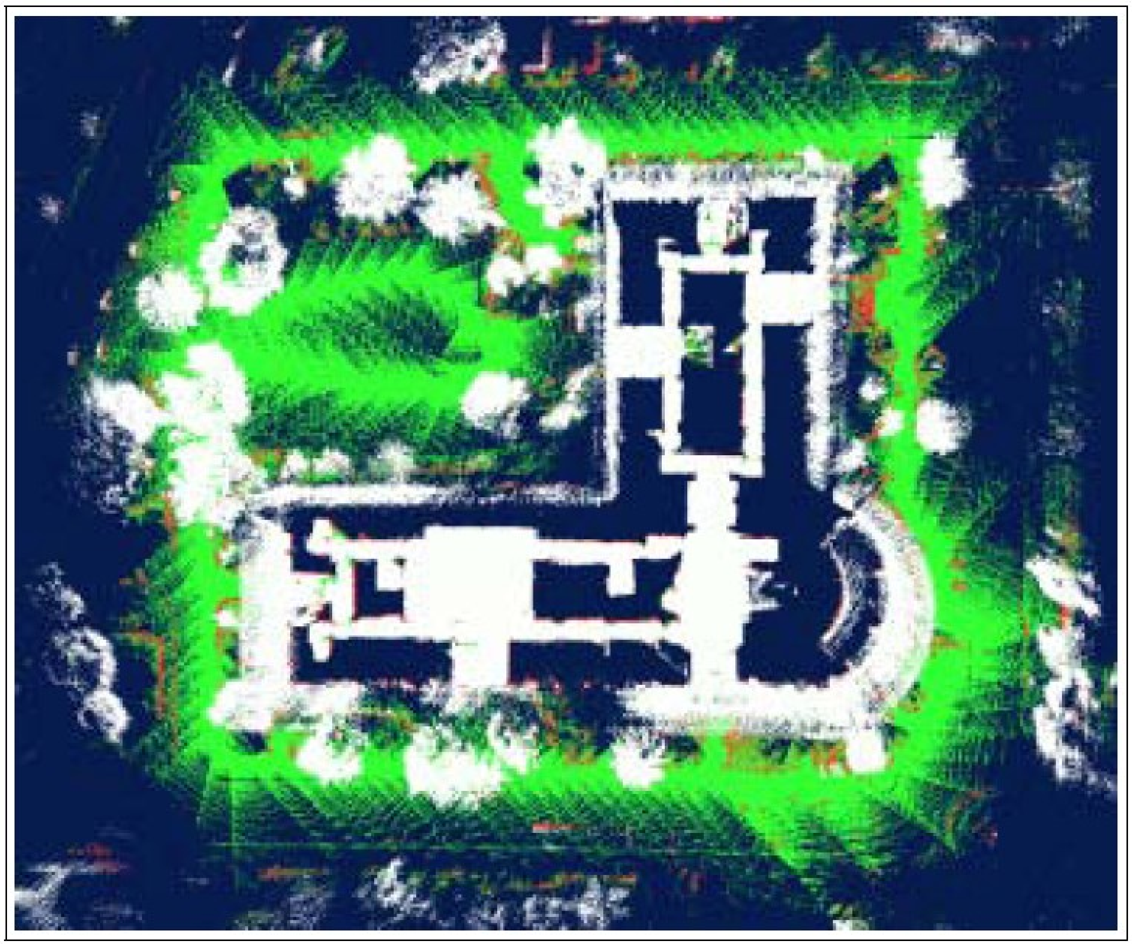
Segway RMP at Stanford

Color coded 3D map of the Stanford campus:



Segway RMP at Stanford

Top view of 3D map of the Stanford campus:



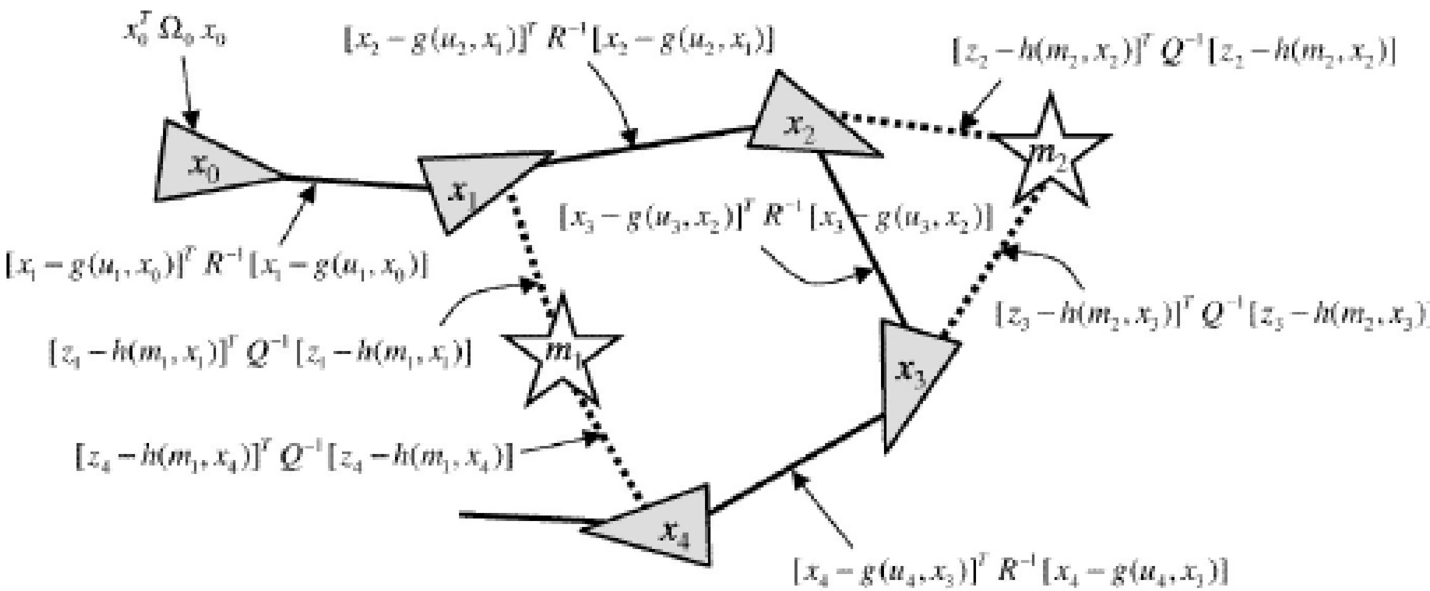
Segway RMP at Stanford

Effect of GPS on indoor mapping:

Resumé

Use a **graph** to represent the problem:

- **Every node** in the graph **corresponds to a pose** or **an observation** of the robot during mapping
- **Every edge** between two nodes **corresponds to the spatial constraints** between them



Conclusion

GraphSLAM:

- Solves the Full SLAM problem as post-processing step
- Creates a graph of soft constraints from the data-set
- By minimizing the sum of all constraints the maximum likelihood estimate of both the map and the robot path is found
- The algorithm works in iterating three steps: construction, reduction, solving remaining equations

$$p(x_{1:t}, m \mid z_{1:t}, u_{1:t})$$