Using the Kalman Filter for SLAM

Paul Newman

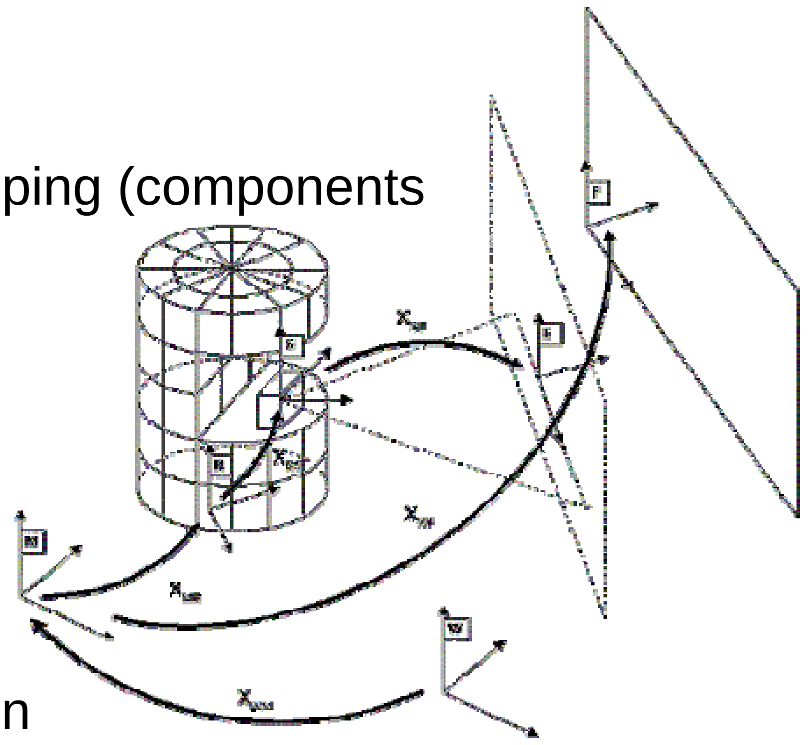
SLAM Summer School 2006

SSS 06

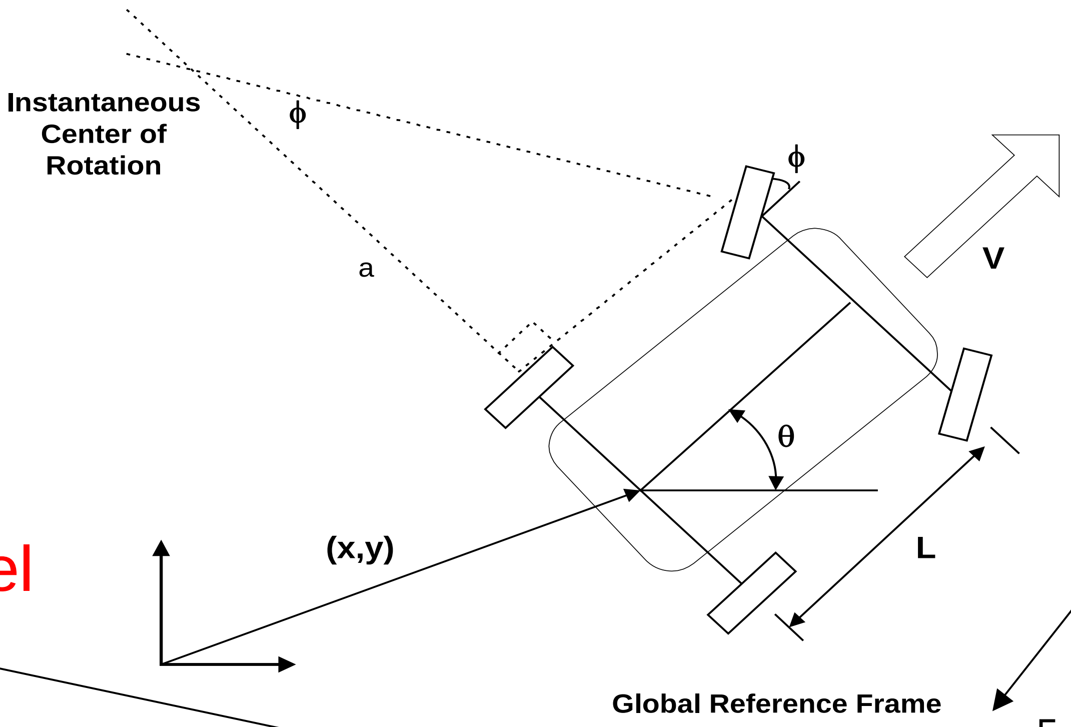
$$p(\mathrm{x}_k, \mathrm{m}|\mathrm{Z}^{k-1}) = \int_{-\infty}^{\infty} p(\mathrm{x}_k|\mathrm{x}_{k-1})p(\mathrm{x}_{k-1}, \mathrm{m}|\mathrm{Z}^{k-1})d\mathrm{x}_{k-1}$$

Contents

- Trivial Kinematics
- Rapid sweep over localisation and mapping (components of SLAM)
- Basic EKF Feature Based SLAM
- Feature types and representations
- Implementation
- Convergence Properties
- Pose Based SLAM
- Sparse Inverse Pose Based Formulation



Vehicle Models - Prediction



Truth model

control

$$\mathbf{x}_v(k+1) = \mathbf{f}(\mathbf{x}_v(k), \mathbf{u}(k)); \quad \mathbf{u}(k) = \begin{bmatrix} V(k) \\ \phi(k) \end{bmatrix}$$

$$\begin{bmatrix} x_v(k+1) \\ y_v(k+1) \\ \theta_v(k+1) \end{bmatrix} = \begin{bmatrix} x_v(k+1) + \delta T V(k) \cos(\theta_v(k)) \\ y_v(k+1) + \delta T V(k) \sin(\theta_v(k)) \\ \theta_v(k) + \frac{\delta T V(k) \tan(\phi(k))}{L} \end{bmatrix}$$

Noise is in control….

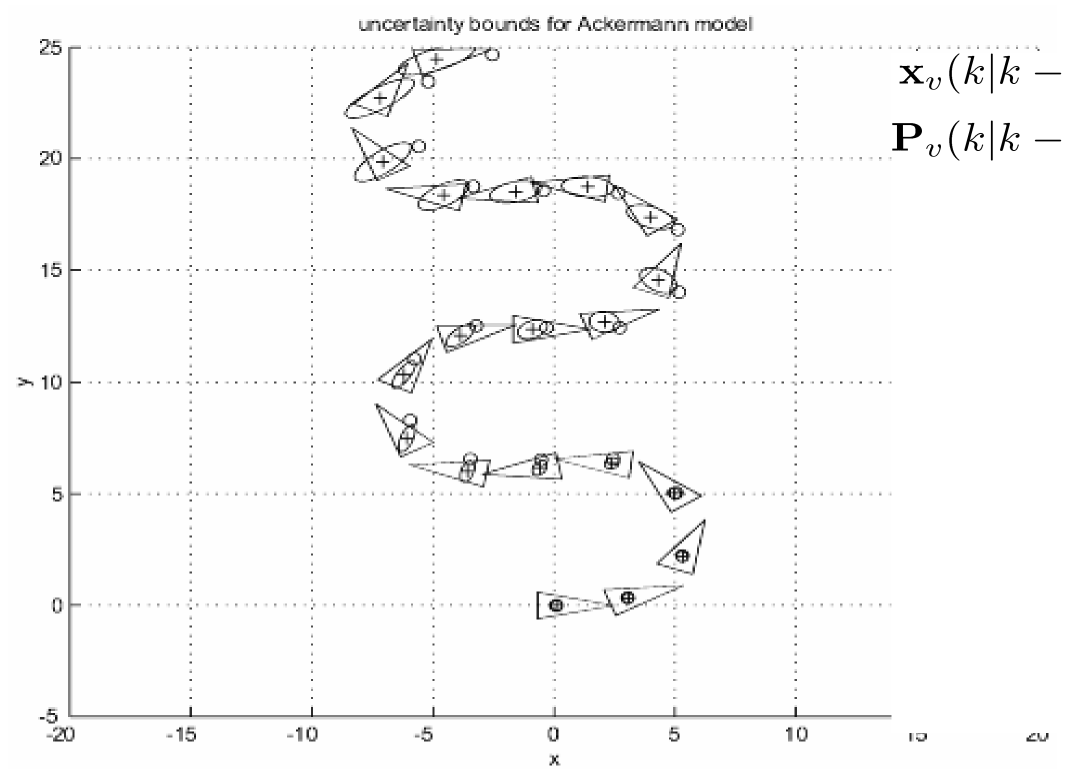
$$\mathbf{u}(k) = \mathbf{u}_n(k) + \mathbf{v}(k)$$

where $\mathbf{u}_n(k)$ is a nominal (intended) control signal and $\mathbf{v}(k)$ is a zero mean gaussian distributed noise vector:

$$\mathbf{v}(k) \sim \mathcal{N}(0, \begin{bmatrix} \sigma_V^2 & 0 \\ 0 & \sigma_\phi^2 \end{bmatrix})$$

$$\mathbf{u}(k) \sim \mathcal{N}(\mathbf{u}_n(k), \begin{bmatrix} \sigma_V^2 & 0 \\ 0 & \sigma_\phi^2 \end{bmatrix})$$

Effect of control noise on uncertainty:



$$\mathbf{x}_v(k|k-1) = \mathbf{f}(\hat{\mathbf{x}}(k-1|k-1), \mathbf{u}(k), k)$$

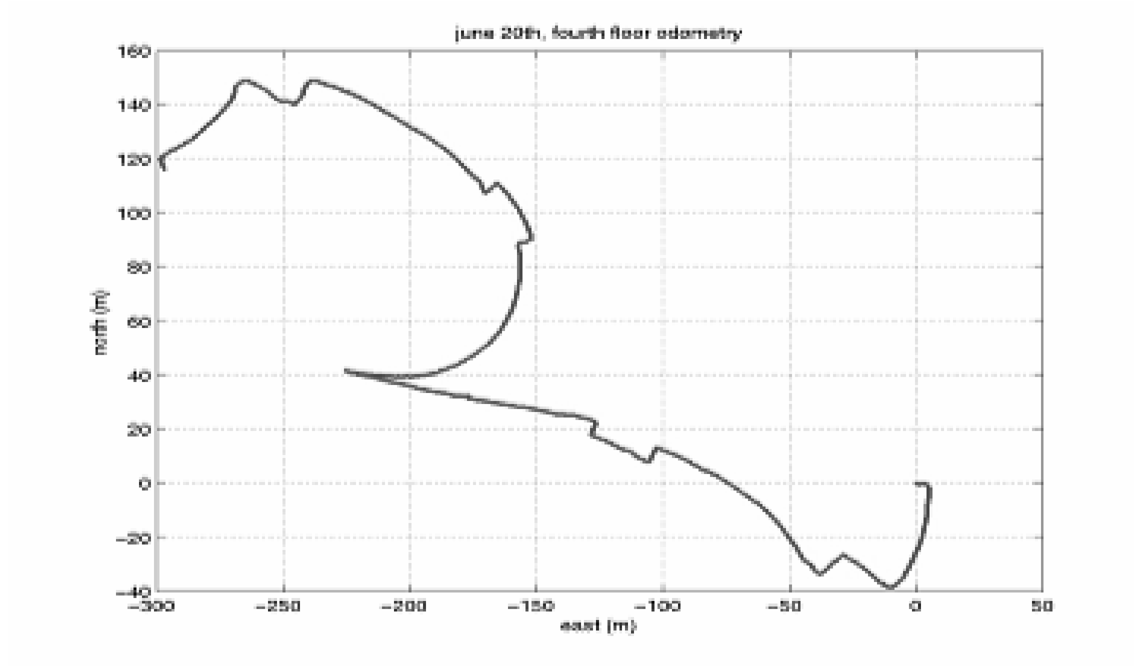
$$\mathbf{P}_v(k|k-1) = \nabla\mathbf{F_x}\mathbf{P}_v(k-1|k-1)\nabla\mathbf{F_x}^T + \nabla\mathbf{F_v}\mathbf{Q}\nabla\mathbf{F_v}^T$$

$$\mathbf{Q} = \begin{bmatrix} \sigma_V^2 & 0 \\ 0 & \sigma_\phi^2 \end{bmatrix}$$

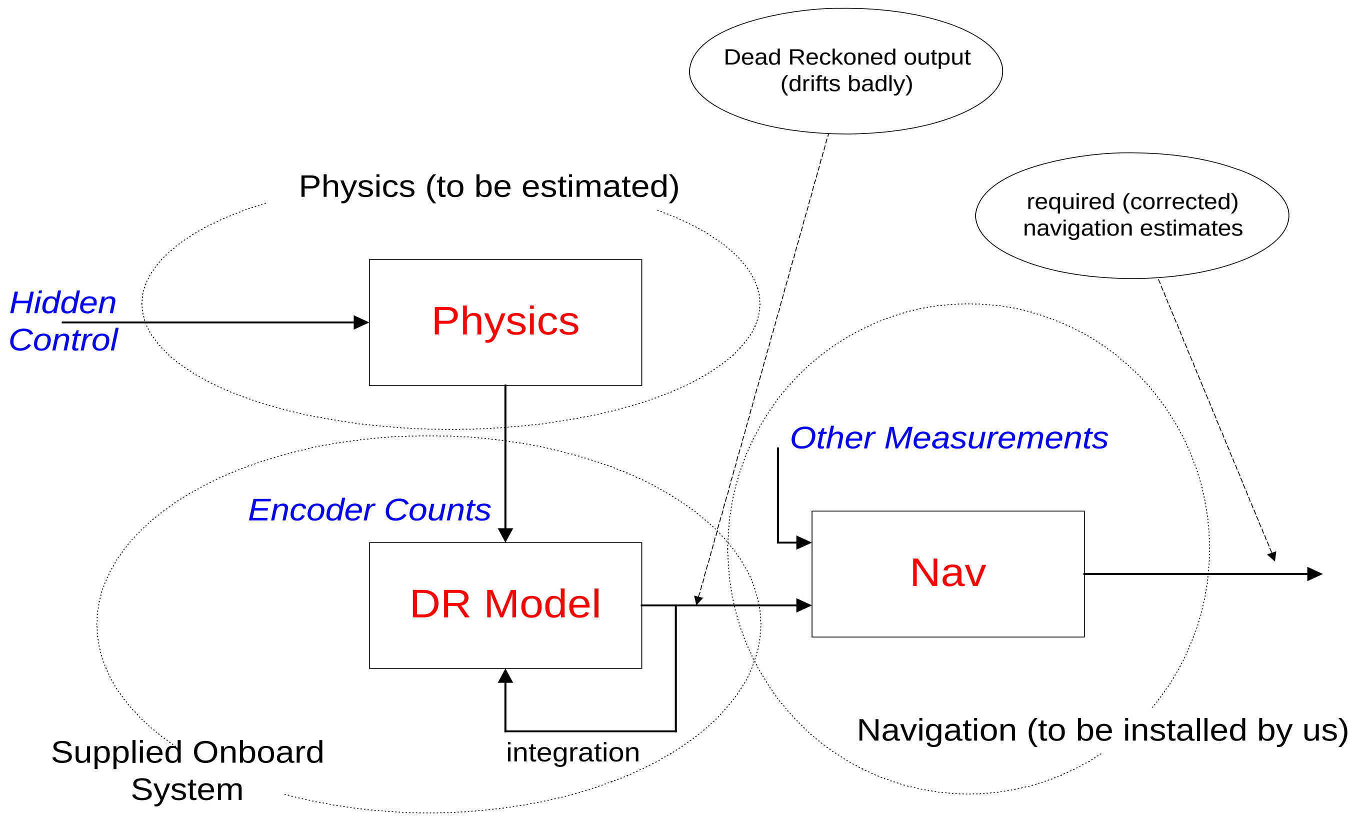
$$\nabla\mathbf{F_x} = \begin{bmatrix} 1 & 0 & -\delta T V \sin(\theta_v) \\ 0 & 1 & \delta T V \cos(\theta_v) \\ 0 & 0 & 1 \end{bmatrix}$$

$$\nabla\mathbf{F_u} = \begin{bmatrix} \delta T \cos(\theta_v) & 0 \\ \delta T \sin(\theta_v) & 0 \\ \frac{\tan(\phi)}{L} & \frac{\delta T V \sec^2(\phi)}{L} \end{bmatrix}$$

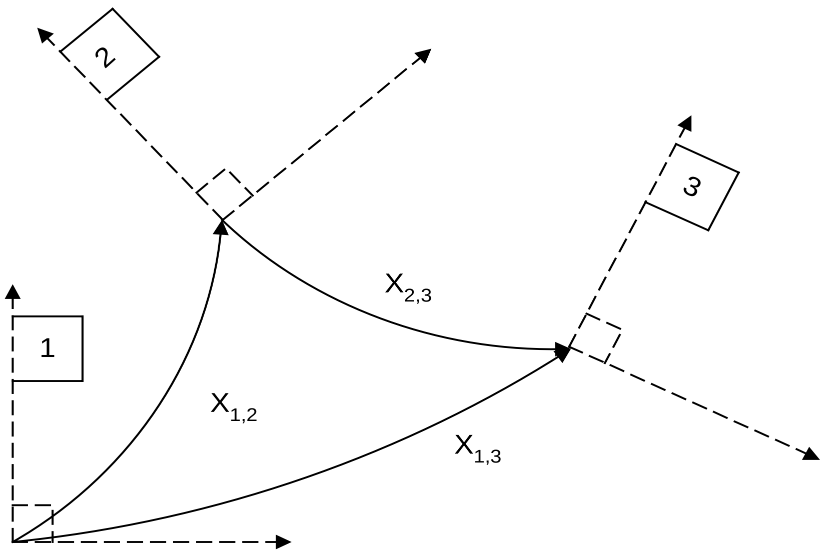
Using Dead-Reckoned Data



Navigation Architecture


Dead Reckoned output (drifts badly)
Physics (to be estimated)
required (corrected) navigation estimates
Hidden Control
Physics
Other Measurements
Encoder Counts
DR Model
Nav
Navigation (to be installed by us)
Supplied Onboard System
integration

Background T-Composition



$$\mathbf{x}_{i,k} = \mathbf{x}_{i,j} \oplus \mathbf{x}_{j,k}$$

$$\mathbf{x}_{j,i} = \ominus \mathbf{x}_{i,j}$$

Compounding transformations

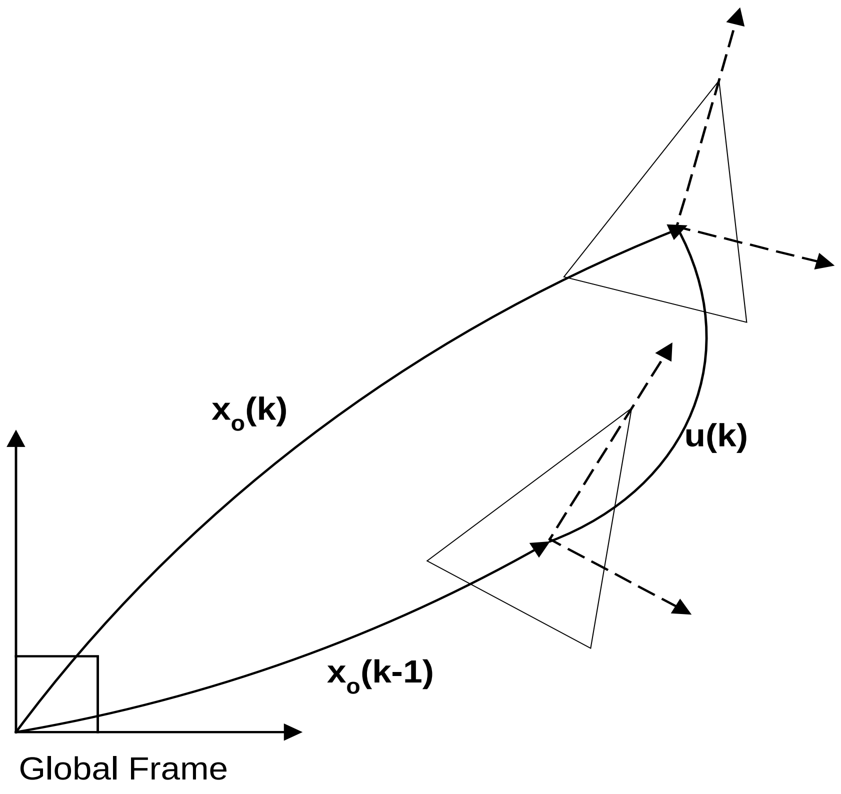
Just functions!

$$\mathbf{x}_1 \oplus \mathbf{x}_2 = \begin{bmatrix} x_1 + x_2 \cos\theta_1 - y_2 \sin\theta_1 \\ y_1 + x_2 \sin\theta_1 + y_2 \cos\theta_1 \end{bmatrix}$$

$$\ominus \mathbf{x}_1 = \begin{bmatrix} -x_1 \cos\theta_1 - y_1 \sin\theta_1 \\ x_1 \sin\theta_1 - y_1 \cos\theta_1 \\ -\theta_1 \end{bmatrix}$$

$$\begin{aligned} \mathbf{J}_1(\mathbf{x}_1, \mathbf{x}_2) &\triangleq \frac{\partial(\mathbf{x}_1 \oplus \mathbf{x}_2)}{\partial \mathbf{x}1} \\ &= \begin{bmatrix} 1 & 0 & -x_2 \sin\theta_1 - y_2 \cos\theta_1 \\ 0 & 1 & x_2 \cos\theta_1 - y_2 \sin\theta_1 \\ 0 & 0 & 1 \end{bmatrix} \\ \mathbf{J}_2(\mathbf{x}_1, \mathbf{x}_2) &\triangleq \frac{\partial(\mathbf{x}_1 \oplus \mathbf{x}_2)}{\partial \mathbf{x}2} \\ &= \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

Deduce an Incremental Move



These can be in massive error

$$\mathbf{u}(k) = \ominus \mathbf{x}_o(k-1) \oplus \mathbf{x}_o(k)$$

But the common error is subtracted out here

Use this “move” as a control

$$\begin{aligned}\mathbf{x}_v(k+1) &= \mathbf{f}(\mathbf{x}_v(k), \mathbf{u}(k)) \\ &= \mathbf{x}_v(k) \oplus \underbrace{(\ominus \mathbf{x}_o(k-1) \oplus \mathbf{x}_o(k))}_{dr-control} \\ &= \mathbf{x}_v(k) \oplus \mathbf{u}_o(k)\end{aligned}$$

Substitution into Prediction equation (using J1 and J2 as Jacobians):

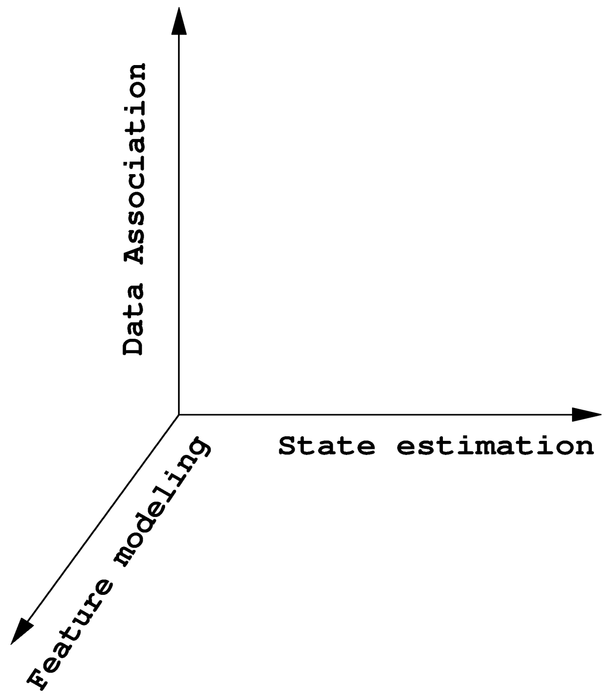
$$\begin{aligned}\mathbf{P}(k|k-1) &= \nabla\mathbf{F_x}\mathbf{P}_v(k-1|k-1)\nabla\mathbf{F_x}^T + \nabla\mathbf{F_v}\mathbf{Q}\nabla\mathbf{F_v}^T \\ &= \mathbf{J}_1(\mathbf{x}_v,\mathbf{u}_o)\mathbf{P}_v(k-1|k-1)\mathbf{J}_1(\mathbf{x}_v,\mathbf{u}_o)^T + \mathbf{J}_2(\mathbf{x}_v,\mathbf{u}_o)\mathbf{U}_o\mathbf{J}_2(\mathbf{x}_v,\mathbf{u}_o)^T\end{aligned}$$

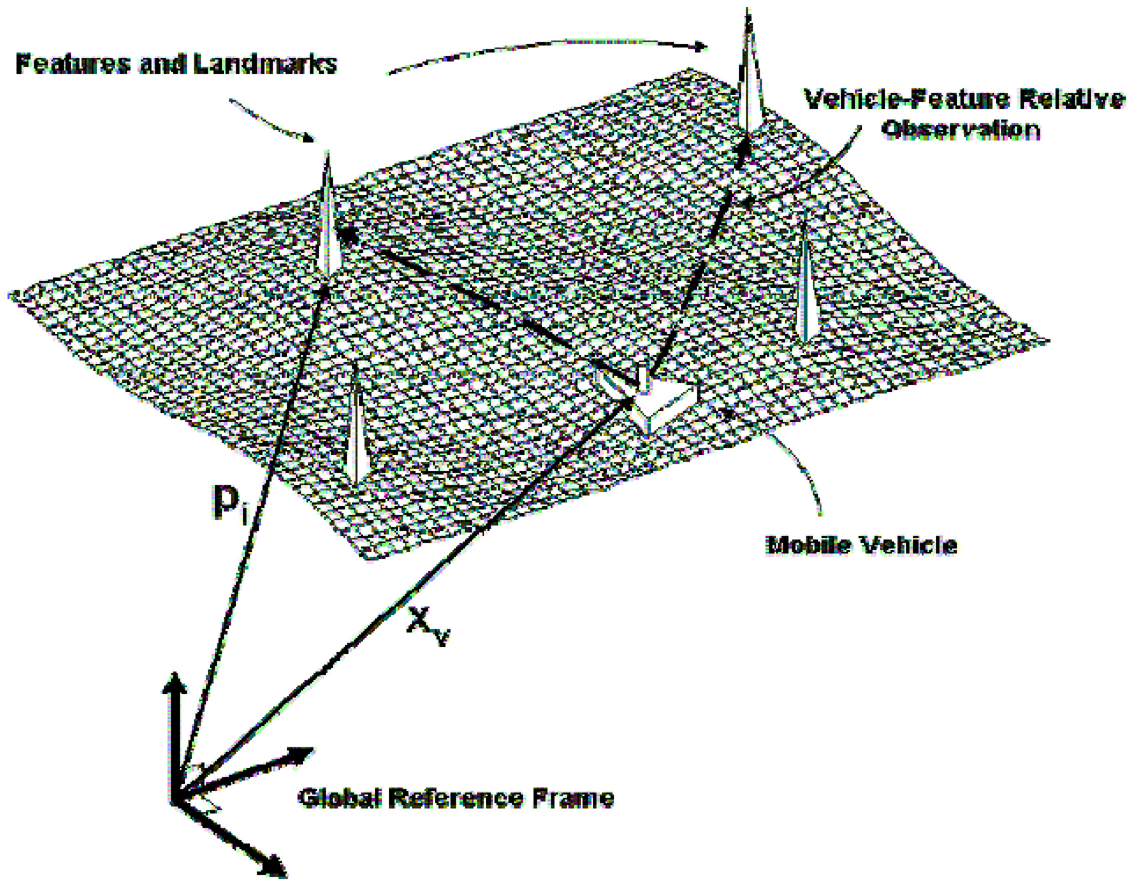
Diagonal covariance matrix (3x3) of error in u_o

Feature Based Mapping and Navigation

Look at the code!!

Problem Space





Landmarks / Features

Things that standout to a sensor:

Corners, windows, walls, bright patches, texture…

$$\mathbf{M} = \begin{bmatrix} \mathbf{x}_{\mathbf{f},1} \\ \mathbf{x}_{\mathbf{f},2} \\ \vdots \\ \mathbf{x}_{\mathbf{f},n} \end{bmatrix} \qquad \mathbf{x}_{\mathbf{f},i} = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

Map

Point Feature called “i”

•We shall initially proceed by considering only point features
•We’ll discuss other feature types later

Three Inference Problem

Input is measurements conditioned on map and vehicle

Data: $p(\mathbf{Z^k}|\mathbf{M}, \mathbf{x}_v)$

We want to use Bayes' rule to "invert" this and get maps and vehicles given measurements.

Problem 1 – inferring location given a map (easy)
Problem 2 – inferring a map given a location (easy)
Problem 3 – SLAM – learning a map and locating simultaneously

We can use a KF for all the above

Simple Motion Model

Plant Model

$$\hat{\mathbf{x}}(k|k-1) = \hat{\mathbf{x}}(k-1|k-1) \oplus (\ominus \mathbf{x}_o(k-1) \oplus \mathbf{x}_o(k))$$

$$\hat{\mathbf{x}}(k|k-1) = \hat{\mathbf{x}}(k-1|k-1) \oplus \mathbf{u}_o(k)$$

$$\begin{aligned}\mathbf{P}(k|k-1) &= \nabla\mathbf{F_x}\mathbf{P}(k-1|k-1)\nabla\mathbf{F_x}^T + \nabla\mathbf{F_v}\mathbf{Q}\nabla\mathbf{F_v}^T \\ &= \mathbf{J}_1(\mathbf{x}_v, \mathbf{u}_o)\mathbf{P}_v(k-1|k-1)\mathbf{J}_1(\mathbf{x}_v, \mathbf{u}_o)^T + \mathbf{J}_2(\mathbf{x}_v, \mathbf{u}_o)\mathbf{U}_o\mathbf{J}_2(\mathbf{x}_v, \mathbf{u}_o)^T\end{aligned}$$

Remember:
u is control, J’s are a fancy way of writing jacobians (composition operator). U is strength of noise in plant model.

Observation Model

$$\mathbf{z}(k) \triangleq \begin{bmatrix} r \\ \theta \end{bmatrix}$$

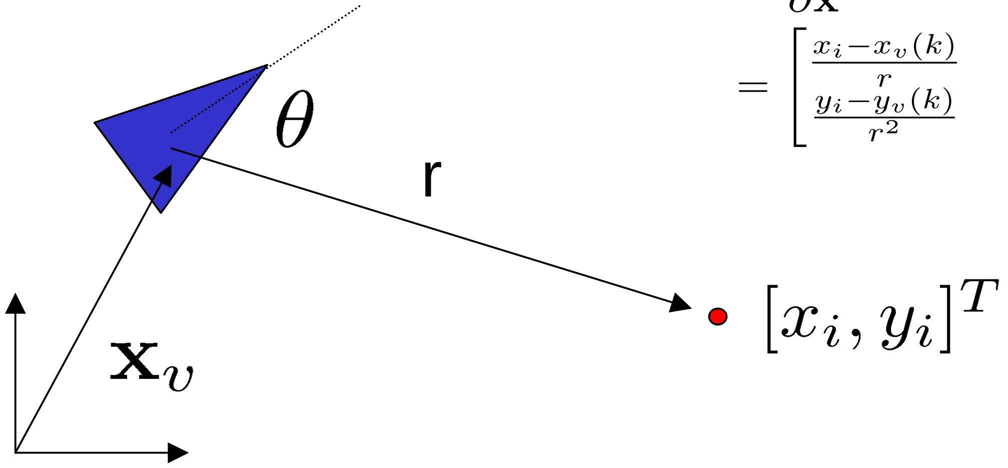
$$= \mathbf{h}(\mathbf{x}(k), \mathbf{w}(k))$$

$$= \begin{bmatrix} \sqrt{(x_i - x_v(k))^2 + (y_i - y_v(k))^2} \\ atan2(\frac{y_i - y_v(k)}{x_i - x_v(k)}) - \theta_v \end{bmatrix}$$

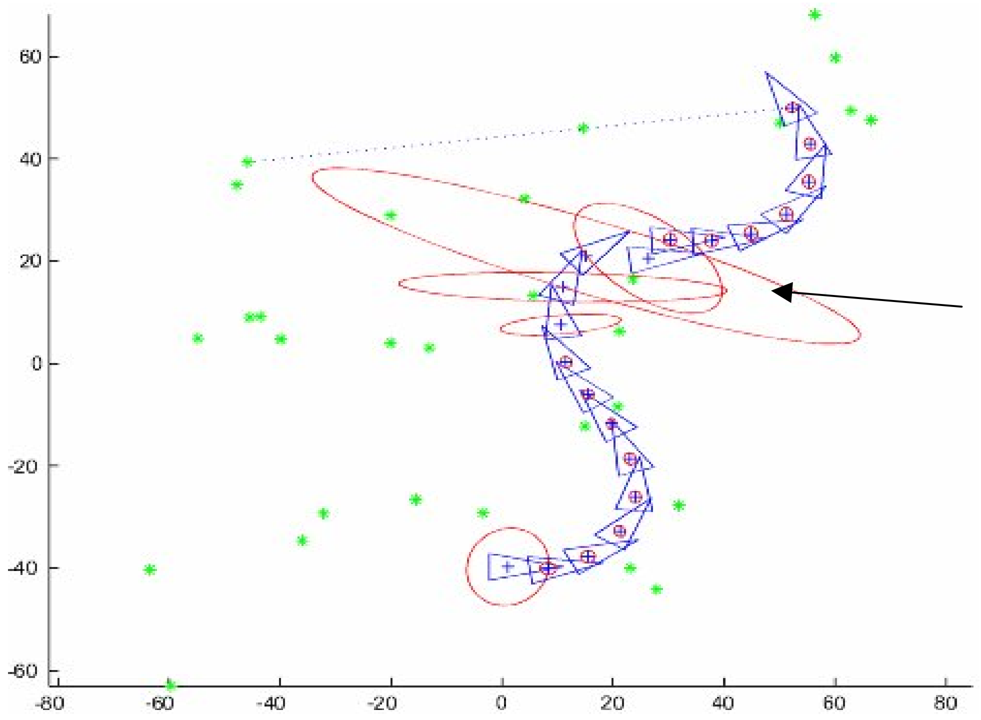
We differentiate w.r.t $\mathbf{x}_v$ to arrive at the observation model jacobian:

$$\nabla \mathbf{H_x} \triangleq \frac{\partial \mathbf{h}}{\partial \mathbf{x}}$$

$$= \begin{bmatrix} \frac{x_i - x_v(k)}{r} & \frac{y_i - y_v(k)}{r} & 0 \\ \frac{y_i - y_v(k)}{r^2} & -\frac{x_i - x_v(k)}{r^2} & -1 \end{bmatrix}$$



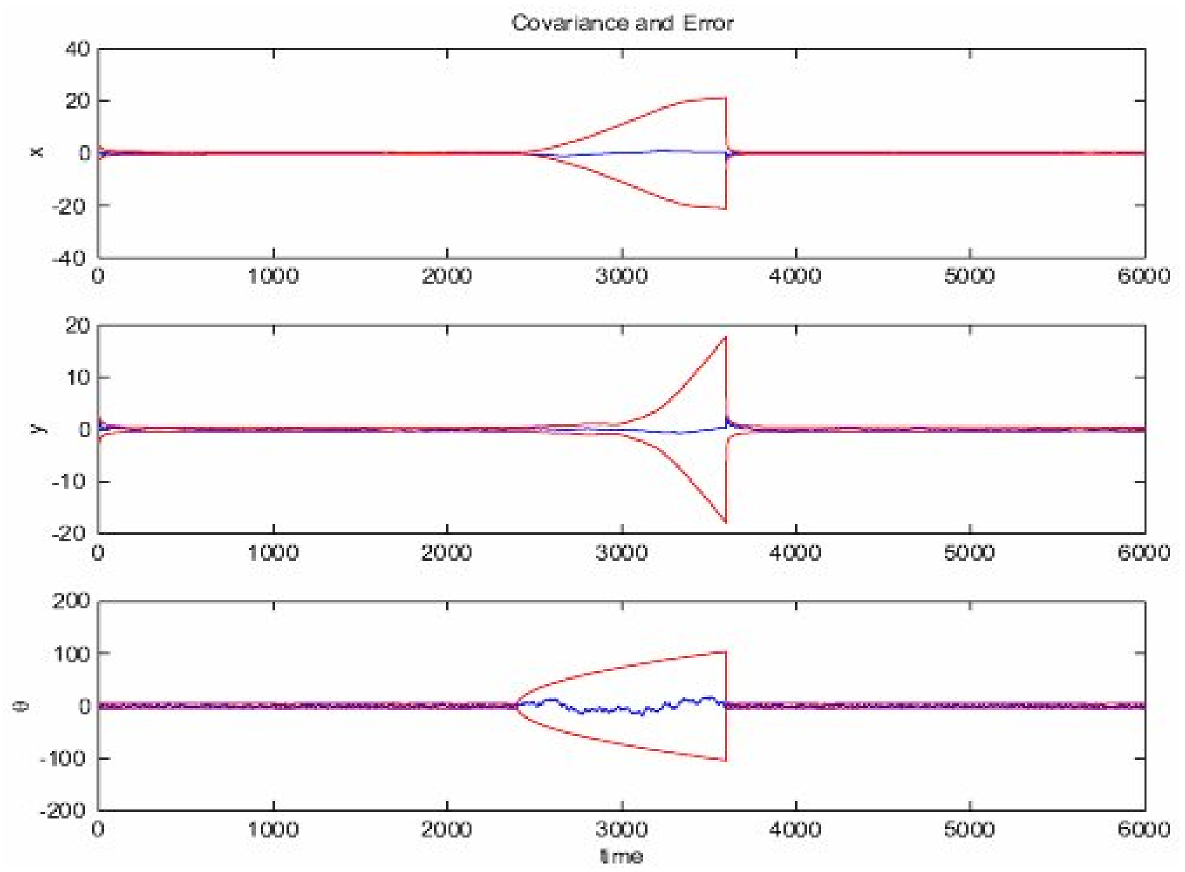
Example



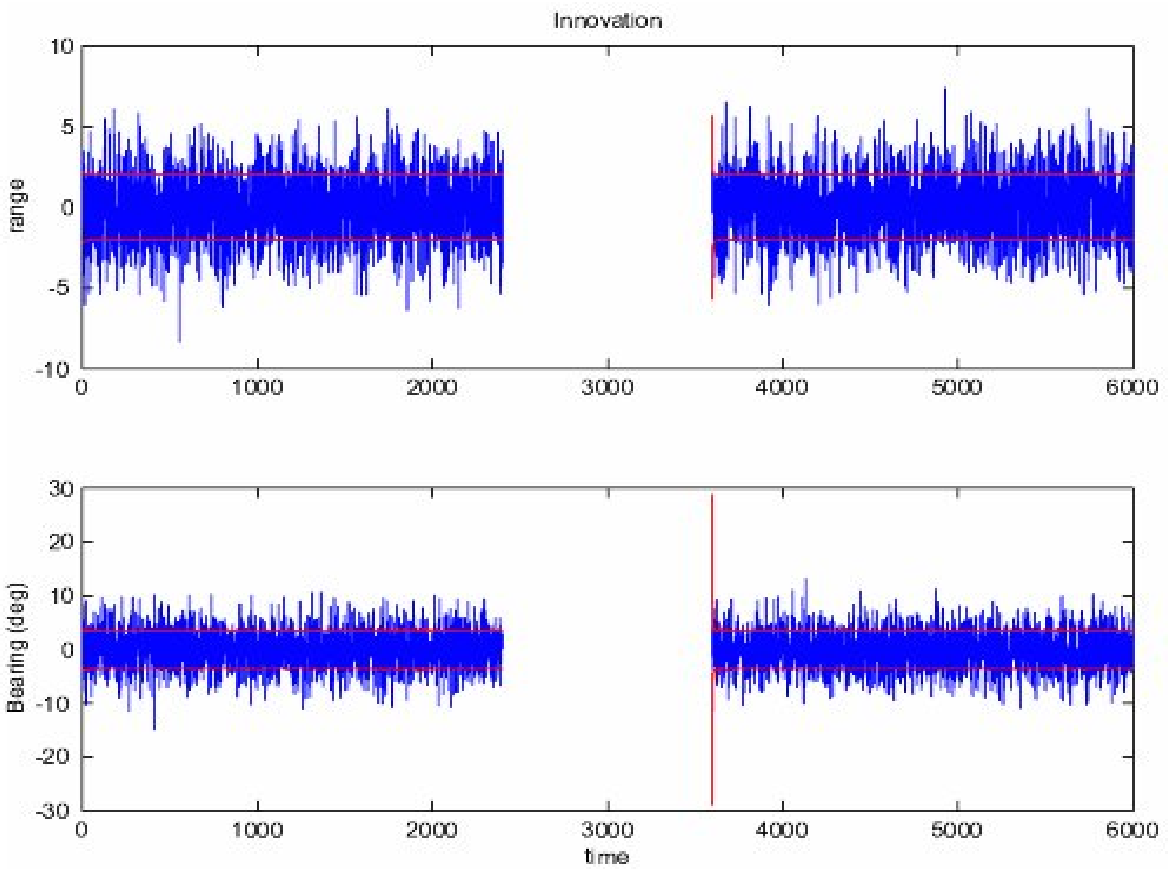
No features seen here

Example Code Provided

Location Covariance



Location Innovation



Problem II - Mapping

Key Point: State Vector GROWS!

New, bigger map

Old map

Obs of new feature

$$\begin{aligned}\mathbf{x}(k|k)^* &= \mathbf{y}(\mathbf{x}(k|k), \mathbf{z}(k), \mathbf{x}_v(k|k)) \\ &= \begin{bmatrix} \mathbf{x}(k|k) \\ \mathbf{g}(\mathbf{x}(k), \mathbf{z}(k), \mathbf{x}_v(k|k)) \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{x}(k|k) \\ x_v + r\cos(\theta + \theta_v) \\ y_v + r\sin(\theta + \theta_v) \end{bmatrix}\end{aligned}$$

"State augmentation"

where the coordinates of the new feature are given by the function **g**:

$$\begin{aligned}\mathbf{x}_{fnew} &= \mathbf{g}(\mathbf{x}(k), \mathbf{z}(k), \mathbf{x}_v(k|k)) \\ &= (\begin{bmatrix} x_v + r\cos(\theta + \theta_v) \\ y_v + r\sin(\theta + \theta_v) \end{bmatrix}\end{aligned}$$

The state vector is the map

How is P augmented?

Simple! Use the transformation of covariance rule..

$$\mathbf{b} = f(\mathbf{a})$$

$$\mathcal{E}\{\tilde{\mathbf{b}}\tilde{\mathbf{b}}^T\} = \nabla\mathbf{Y}\mathcal{E}\{\tilde{\mathbf{a}}\tilde{\mathbf{a}}^T\}\nabla\mathbf{Y}^T$$

$$\begin{aligned}\mathbf{x}(k|k)^* &= \mathbf{y}(\mathbf{x}(k|k), \mathbf{z}(k), \mathbf{x}_v(k|k)) \\ &= \begin{bmatrix} \mathbf{x}(k|k) \\ \mathbf{g}(\mathbf{x}(k), \mathbf{z}(k), \mathbf{x}_v(k|k)) \end{bmatrix}\end{aligned}$$

G is the feature initialisation function

$$\mathbf{P}(k|k)^* = \nabla\mathbf{Y}_{\mathbf{x},\mathbf{z}} \begin{bmatrix} \mathbf{P}(k|k) & 0 \\ 0 & \mathbf{R} \end{bmatrix} \nabla\mathbf{Y}_{\mathbf{x},\mathbf{z}}{}^T$$

$$\nabla\mathbf{Y}_{\mathbf{x},\mathbf{z}} = \begin{bmatrix} \mathbf{I}_{n\times n} & \mathbf{0}_{\mathbf{n}\times\mathbf{2}} \\ \underbrace{\nabla\mathbf{G}_{\mathbf{x}}}_{0 \text{ for mapping case}} & \nabla\mathbf{G}_{\mathbf{z}} \end{bmatrix}$$

Leading to :

$$\mathbf{x}(k|k)^* = \begin{bmatrix} \mathbf{x}(k|k) \\ x_v + r\cos(\theta + \theta_v) \\ y_v + r\sin(\theta + \theta_v) \end{bmatrix}$$

Angle from Veh to feature

Vehicle orientation

$$\mathbf{P}(k|k)^* = \begin{bmatrix} \mathbf{P}(k|k) & 0 \\ 0 & \nabla\mathbf{G_z}\mathbf{R}\nabla\mathbf{G_z}^T \end{bmatrix}$$

So what are models h and f?

h is a function of the feature being observed:

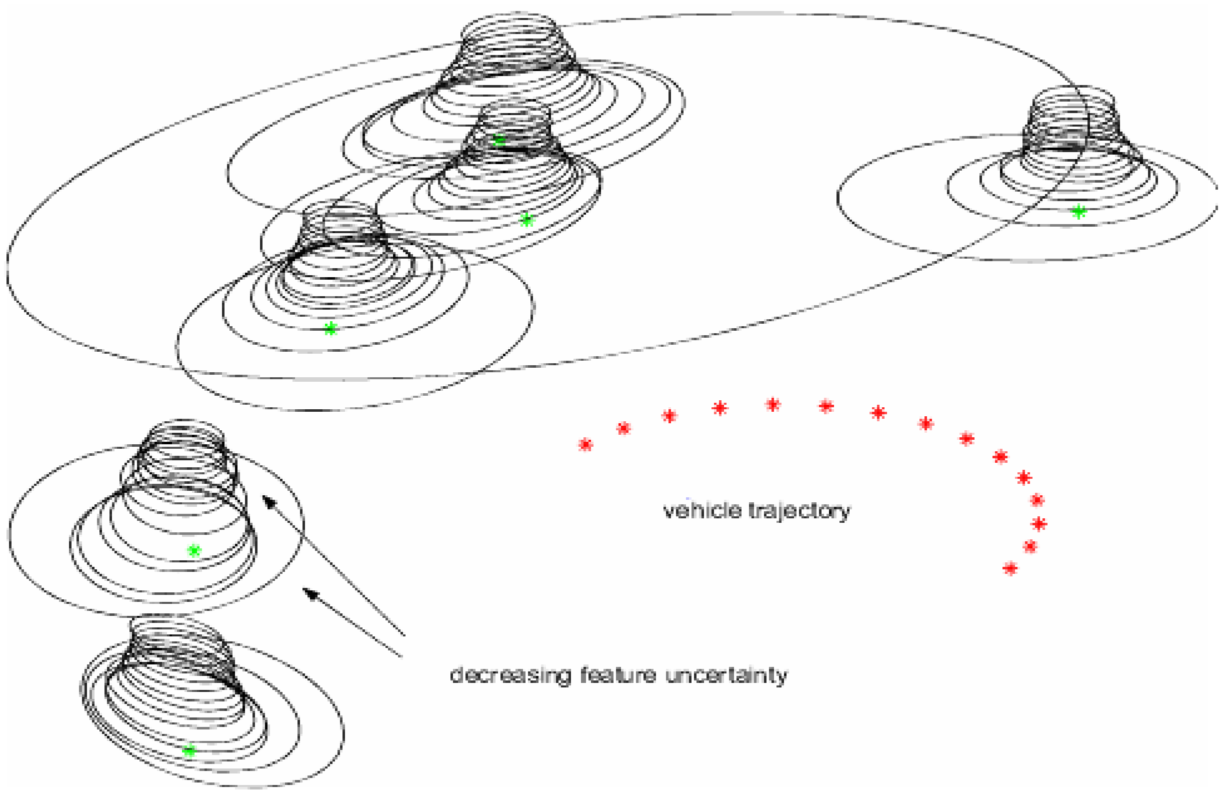
$$\nabla \mathbf{H_x} = \left[\underbrace{\cdots \mathbf{0} \cdots}_{other features} \quad \underbrace{\nabla \mathbf{H_{x_{fi}}}}_{observed feature} \quad \underbrace{\cdots \mathbf{0} \cdots}_{other features} \right]$$

f is simply the identity transformation :

$$\mathbf{x}(k+1|k)_{map} = \mathbf{x}(k|k)_{map}$$

$$\nabla \mathbf{F_x} = \mathbf{I}_{n,n}$$

Turn the handle on the EKF:



All hail the Oracle ! How do we know what feature we are observing?

Problem III SLAM

“Build a map and use it at the same time”

“This a cornerstone of autonomy”

Basic S.S.C SLAM

A union of Localisation and Mapping

$$\mathbf{x} \triangleq \begin{bmatrix} \mathbf{x}_v \\ \mathbf{x}_{\mathbf{f},1} \\ \vdots \\ \mathbf{x}_{\mathbf{f},n} \end{bmatrix}$$

Vehicle parameterisation e.g x,y,theta

Feature parameterisations e.g x,y for points

State vector has vehicle AND map

Why naïve? Computation!

Prediction:

$$\begin{bmatrix} \mathbf{x}_v(k+1) \\ \mathbf{x}_{\mathbf{f},1}(k) \\ \vdots \\ \mathbf{x}_{\mathbf{f},n}(k) \end{bmatrix} = \begin{bmatrix} \mathbf{x}_v(k) \oplus \mathbf{u}(k) \\ \mathbf{x}_{\mathbf{f},1}(k) \\ \vdots \\ \mathbf{x}_{\mathbf{f},n}(k) \end{bmatrix}$$

Vehicle Moves

Map remains unchanged

$$\nabla\mathbf{F_x} = \begin{bmatrix} \nabla\mathbf{F_{x_v}} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{2n\times 2n} \end{bmatrix} = \begin{bmatrix} \mathbf{J}1(\mathbf{x}_v, \mathbf{u}) & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{2n\times 2n} \end{bmatrix}$$

$$\nabla\mathbf{F_u} = \begin{bmatrix} \mathbf{J}2(\mathbf{x}_v, \mathbf{u}) & \mathbf{0} \\ \mathbf{0} & \mathbf{0_{2n\times 2n}} \end{bmatrix}$$

We have all the terms needed for prediction step

$$\mathbf{x}(k|k-1) = f(\mathbf{x}(k|k), (\overbrace{\mathbf{u}(k)}^{\text{control}} + \overbrace{\mathbf{v}(k)}^{\text{noise}}))$$

$$\mathbf{P}(k|k-1) = \nabla\mathbf{F_x}\mathbf{P}(k-1|k-1)\nabla\mathbf{F_x}^T + \nabla\mathbf{F_v}\mathbf{U}\nabla\mathbf{F_v}^T$$

Feature Initialisation:

$$\mathbf{x}(k|k)^* = \begin{bmatrix} \mathbf{x}(k|k) \\ x_v + r\cos(\theta + \theta_v) \\ y_v + r\sin(\theta + \theta_v) \end{bmatrix}$$

This is y(.)

These two lines are g()

$$\mathbf{P}(k|k)^* = \nabla\mathbf{Y}_{\mathbf{x},\mathbf{z}} \begin{bmatrix} \mathbf{P}(k|k) & 0 \\ 0 & \mathbf{R} \end{bmatrix} \nabla\mathbf{Y}_{\mathbf{x},\mathbf{z}}{}^T$$

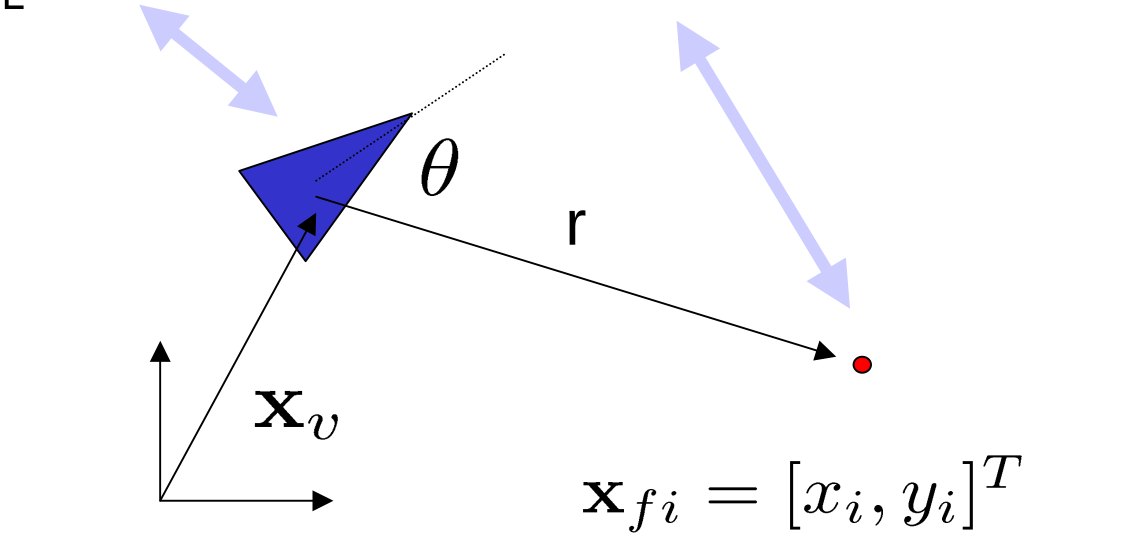
$$\nabla\mathbf{Y}_{\mathbf{x},\mathbf{z}} = \begin{bmatrix} \mathbf{I}_{n\times n} & \mathbf{0}_{\mathbf{n}\times\mathbf{2}} \\ \underbrace{\nabla\mathbf{G}_{\mathbf{x}}}_{\text{now non-zero}} & \nabla\mathbf{G}_{\mathbf{z}} \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{I}_{n\times n} & \mathbf{0}_{\mathbf{n}\times\mathbf{2}} \\ [\nabla\mathbf{G}_{\mathbf{x_v}} \cdots \mathbf{0} \cdots] & \nabla\mathbf{G}_{\mathbf{z}} \end{bmatrix}$$

This a step only runs if a new feature is seen

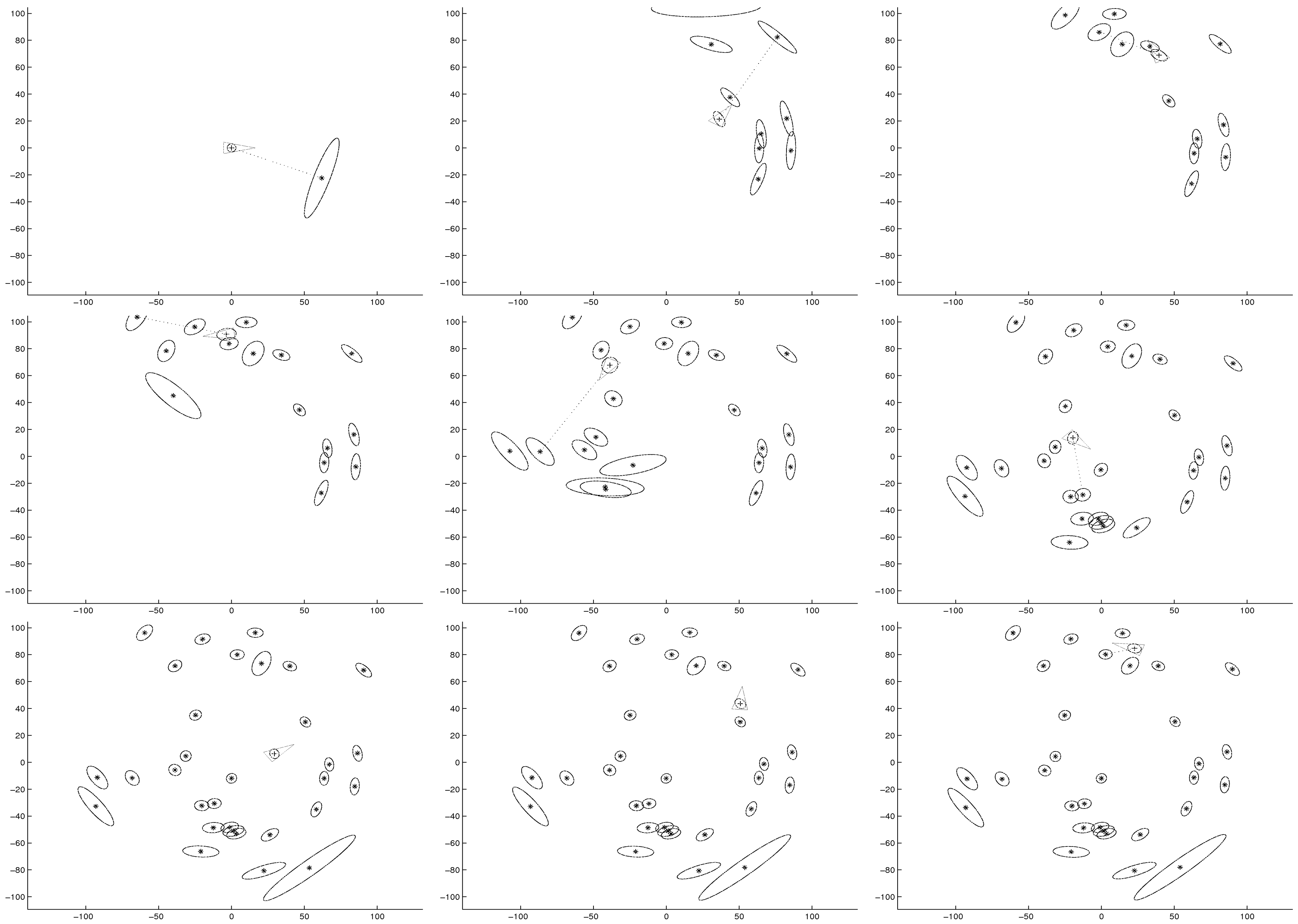
Observation Model

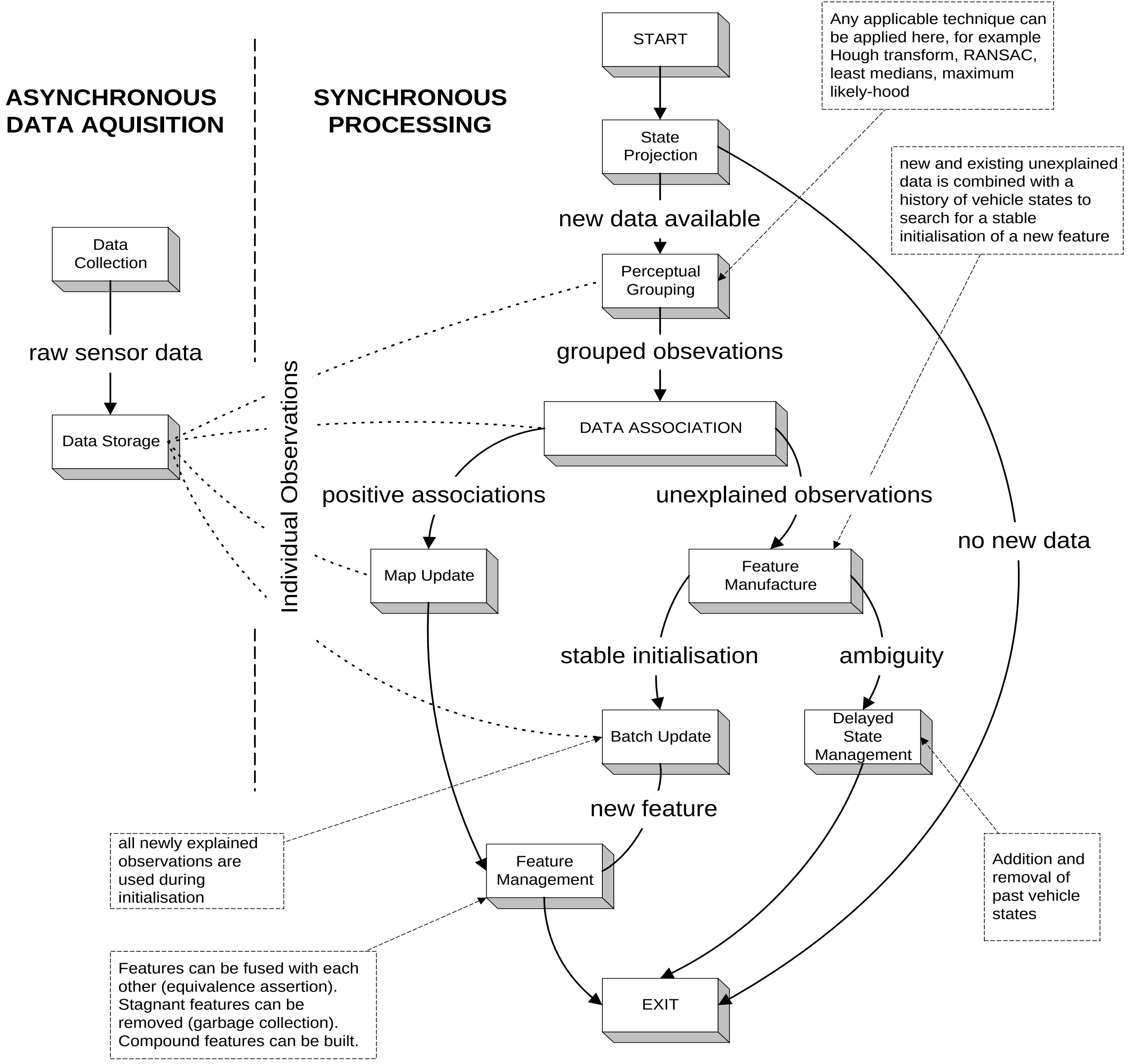
$$\nabla \mathbf{H}_{\mathbf{x}} = \begin{bmatrix} \underbrace{\nabla \mathbf{H}_{\mathbf{x_v}}}_{vehicle} \cdots \mathbf{0} \cdots & \underbrace{\nabla \mathbf{H}_{\mathbf{x_{fi}}}}_{observedfeature} & \underbrace{\cdots \mathbf{0} \cdots}_{otherfeatures} \end{bmatrix}$$



$$\mathbf{x}_{fi} = [x_i, y_i]^T$$

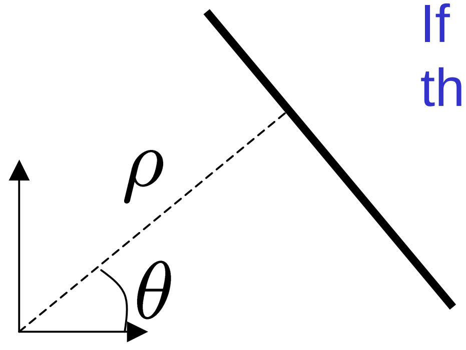
We have all the terms needed for update step


100
80
60
40
20
0
-20
-40
-60
-80
-100
-100
-50
0
50
100

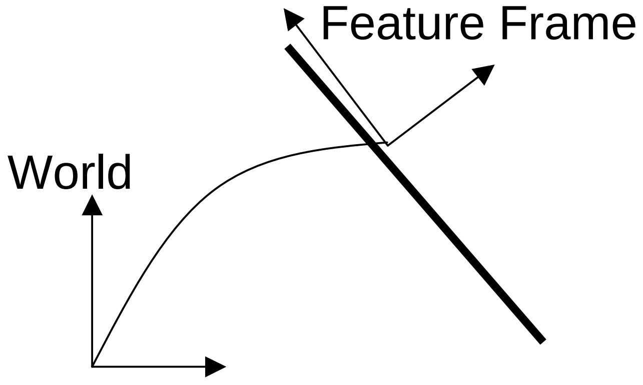

ASYNCHRONOUS DATA AQUISITION
SYNCHRONOUS PROCESSING
START
Any applicable technique can be applied here, for example Hough transform, RANSAC, least medians, maximum likely-hood
State Projection
new and existing unexplained data is combined with a history of vehicle states to search for a stable initialisation of a new feature
Data Collection
new data available
Perceptual Grouping
raw sensor data
grouped obsevations
Data Storage
DATA ASSOCIATION
Individual Observations
positive associations
unexplained observations
no new data
Map Update
Feature Manufacture
stable initialisation
ambiguity
Batch Update
Delayed State Management
new feature
all newly explained observations are used during initialisation
Feature Management
Addition and removal of past vehicle states
Features can be fused with each other (equivalence assertion). Stagnant features can be removed (garbage collection). Compound features can be built.
EXIT

Feature Types

- In theory we can use any geometric parameterisation
- In practice you need to be very careful about a choice of parameterisation.



If ρ is large small errors in θ do crazy things!



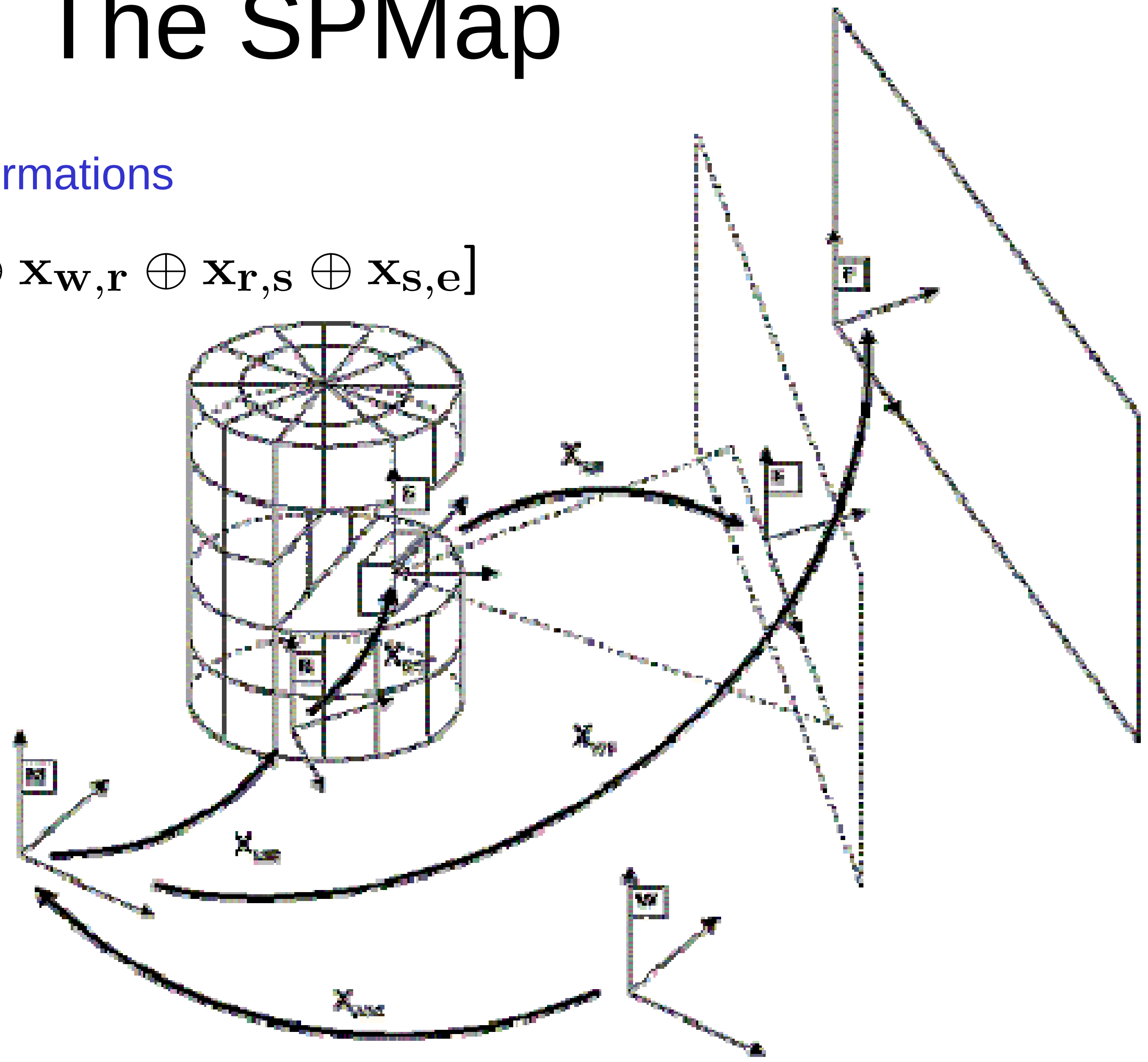
Much smarter to use SPMap (Tardos) which represents geometry and uncertainty in frames centred on a feature. Also see Kai Arras's PhD as an excellent SPMap resource

The SPMap

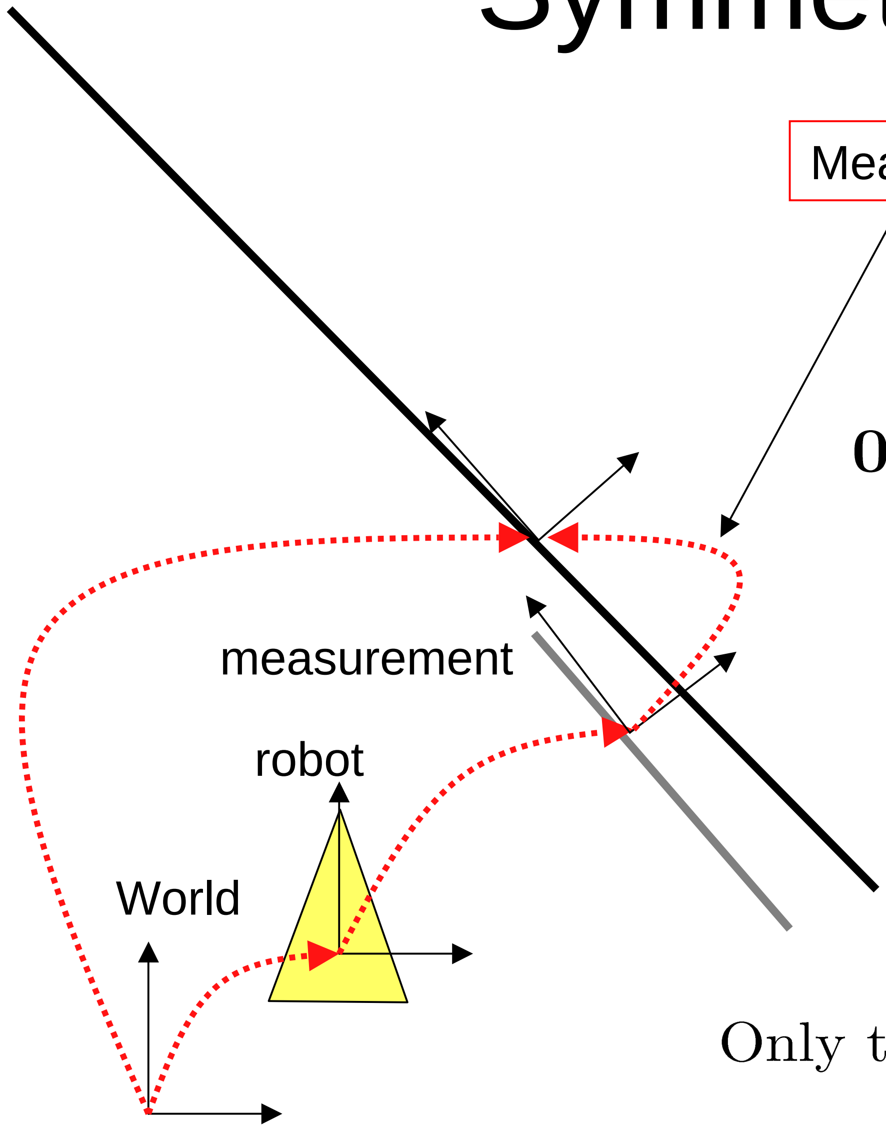
Closed Loop of Transformations

$$\mathbf{0} = \mathbf{B_{f,e}}[\ominus \mathbf{x_{w,f}} \oplus \mathbf{x_{w,r}} \oplus \mathbf{x_{r,s}} \oplus \mathbf{x_{s,e}}]$$

This is the observation model for every feature type!



Symmetry Selection



$$\mathbf{0} = \mathbf{B}_{\mathbf{f},\mathbf{e}}[\ominus\mathbf{x}_{\mathbf{w},\mathbf{f}} \oplus \mathbf{x}_{\mathbf{w},\mathbf{r}} \oplus \mathbf{x}_{\mathbf{r},\mathbf{s}} \oplus \mathbf{x}_{\mathbf{s},\mathbf{e}}]$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{x}_{\mathbf{f},\mathbf{e}}$$

Only the Δx and $\Delta\theta$ components of $\mathbf{x}_{\mathbf{f},\mathbf{e}}$ matter

Calculating EKF Jacobians

Differentiating with respect to feature location

$$\begin{aligned}
\frac{\partial\Psi(\cdot)}{\partial\mathbf{p_f}} &= \mathbf{B_{f,e}} \times \frac{\partial\Psi(\cdot)}{\partial[\ominus\mathbf{B^T_{f,f}p_f}]} \times \frac{\partial[\ominus\mathbf{B^T_{f,f}p_f}]}{\partial[\mathbf{B^T_{f,f}p_f}]} \times \frac{\partial[\mathbf{B^T_{f,f}p_f}]}{\partial\mathbf{p_f}} \\
&= \mathbf{B_{f,e}} \times \mathbf{J_1}\{\ominus\mathbf{B^T_{f,f}p_f}, \mathbf{x_{f,e}}\} \times \mathbf{J}_\ominus\{\mathbf{B^T_{f,f}p_f}\} \times \mathbf{B^T_{f,f}} \\
&= -\mathbf{B_{f,e}} \times \mathbf{J_1}\{\mathbf{0}, \mathbf{x_{f,e}}\} \times \mathbf{B^T_{f,f}}
\end{aligned}$$

Common term X_{fe}: how does Observation appear in feature frame?

similarly with respect to robot center

$$\begin{aligned}
\frac{\partial\Psi(\cdot)}{\partial\mathbf{p_r}} &= \mathbf{B_{f,e}} \times \frac{\partial\Psi(\cdot)}{\partial[\mathbf{J_{e,r}B^T_{r,r}p_r}]} \times \frac{\partial[\mathbf{J_{e,r}B^T_{r,r}p_r}]}{\partial[\mathbf{B^T_{r,r}p_r}]} \times \frac{\partial[\mathbf{B^T_{r,r}p_r}]}{\partial\mathbf{p_r}} \\
&\ \ \vdots \\
&= \mathbf{B_{f,e}} \times \mathbf{J_2}\{\mathbf{x_{f,e}}, \mathbf{0}\} \times \mathbf{J_{e,r}} \times \mathbf{B^T_{r,r}}
\end{aligned}$$

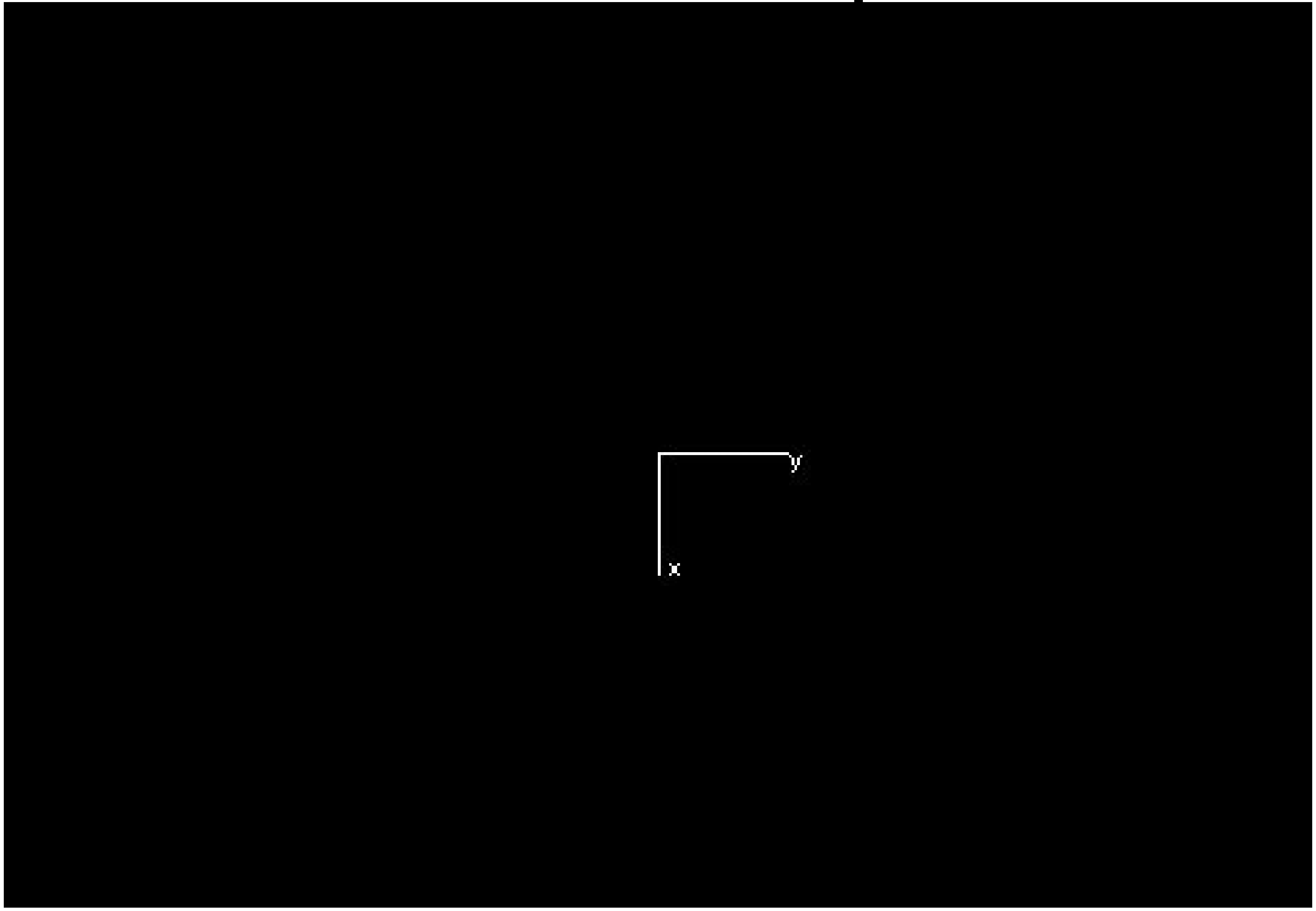
and finally with respect to observation vector

$$\begin{aligned}
\frac{\partial\Psi(\cdot)}{\partial\mathbf{p_e}} &= \mathbf{B_{f,e}} \times \frac{\partial\Psi(\cdot)}{\partial[\mathbf{B^T_{e,e}p_e}]} \times \frac{\partial[\mathbf{B^T_{e,e}p_e}]}{\partial[\mathbf{B^T_{e,e}p_e}]} \\
&\ \ \vdots \\
&= \mathbf{B_{f,e}} \times \mathbf{J_2}\{\mathbf{x_{f,e}}, \mathbf{0}\} \times \mathbf{B^T_{e,e}}
\end{aligned}$$

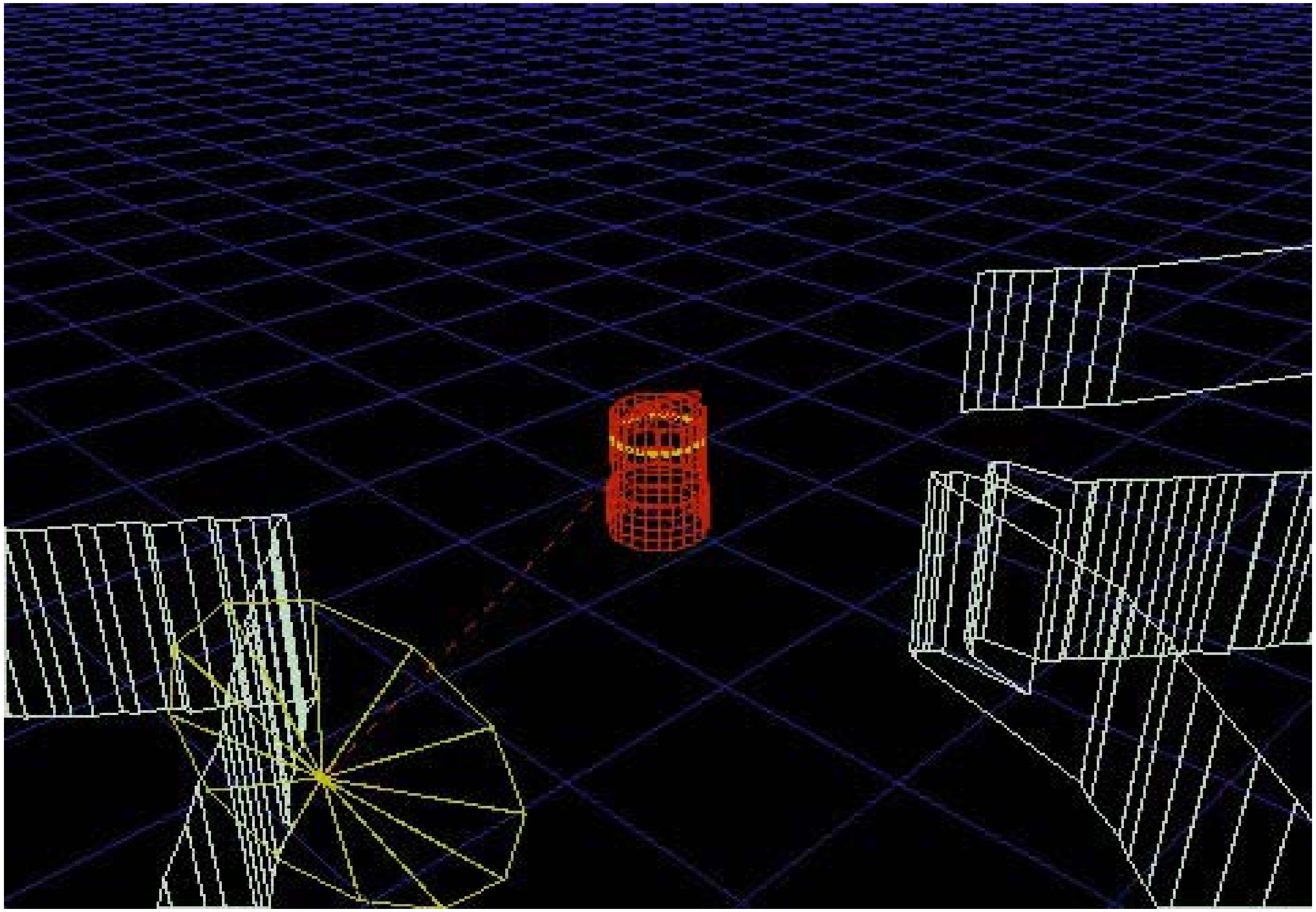
SLAM in action

SPMap Formulation

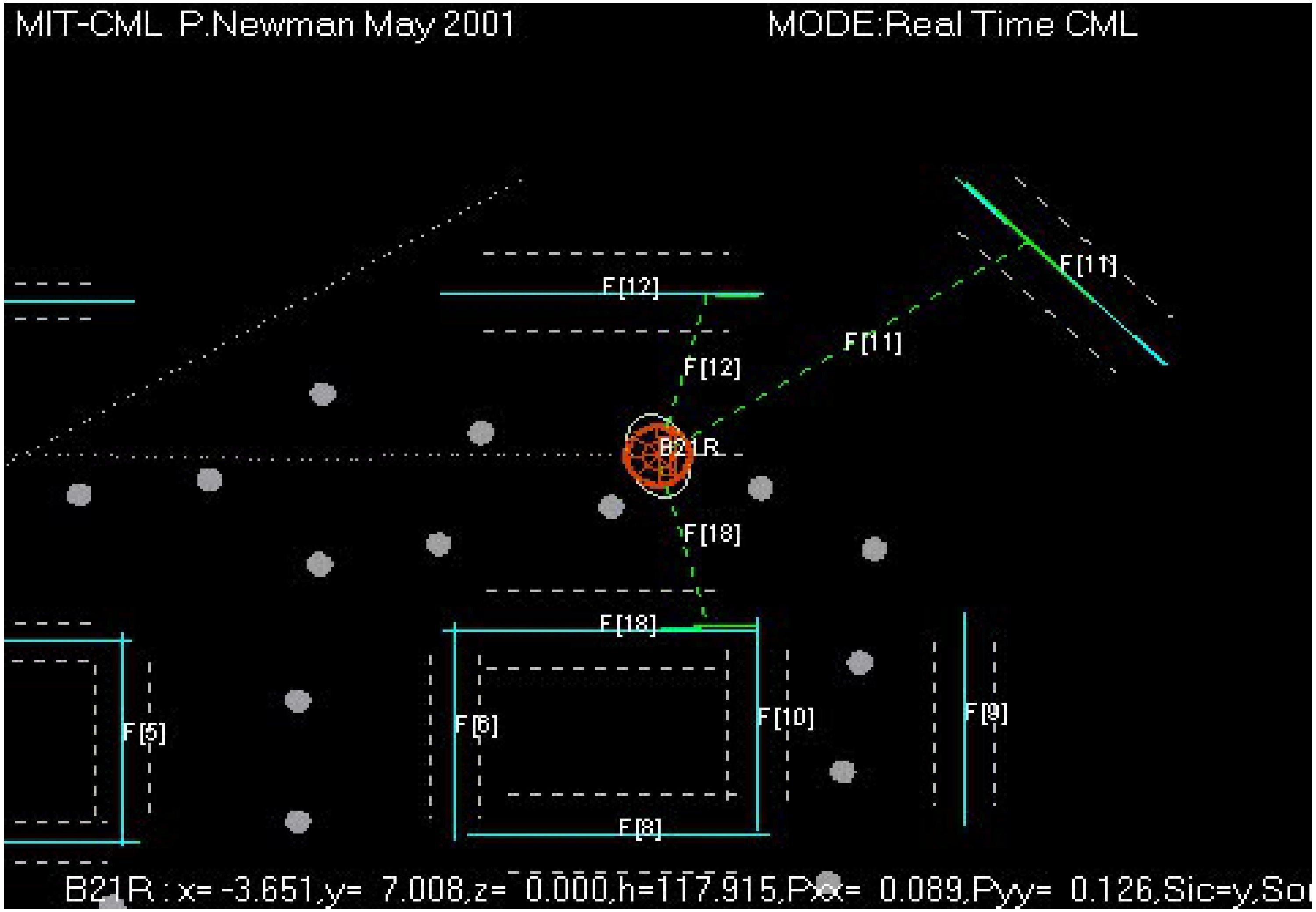
Human Driven Exploration



Navigating



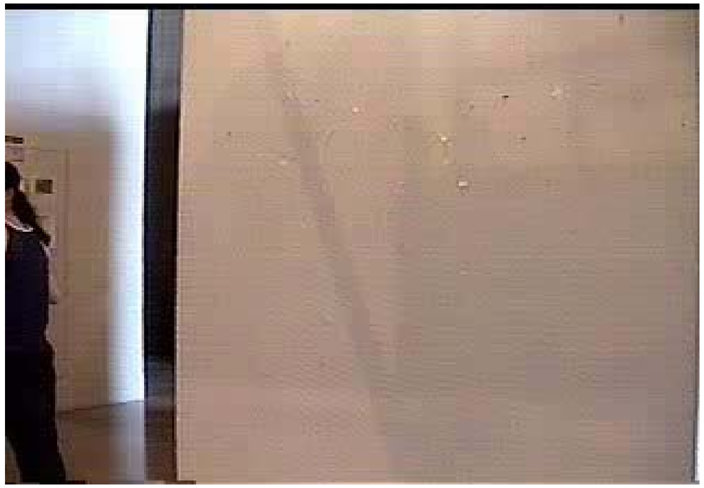
Autonomous Homing


MIT-CML P.Newman May 2001
MODE:Real Time CML
F[12]
F[12]
F[11]
F[11]
B21R
F[18]
F[18]
F[5]
F[6]
F[10]
F[9]
F[8]
B21R : x= -3.651,y= 7.008,z= 0.000,h=117.915,Pxx= 0.089,Pyy= 0.126,Sic=y,So

Using the Map

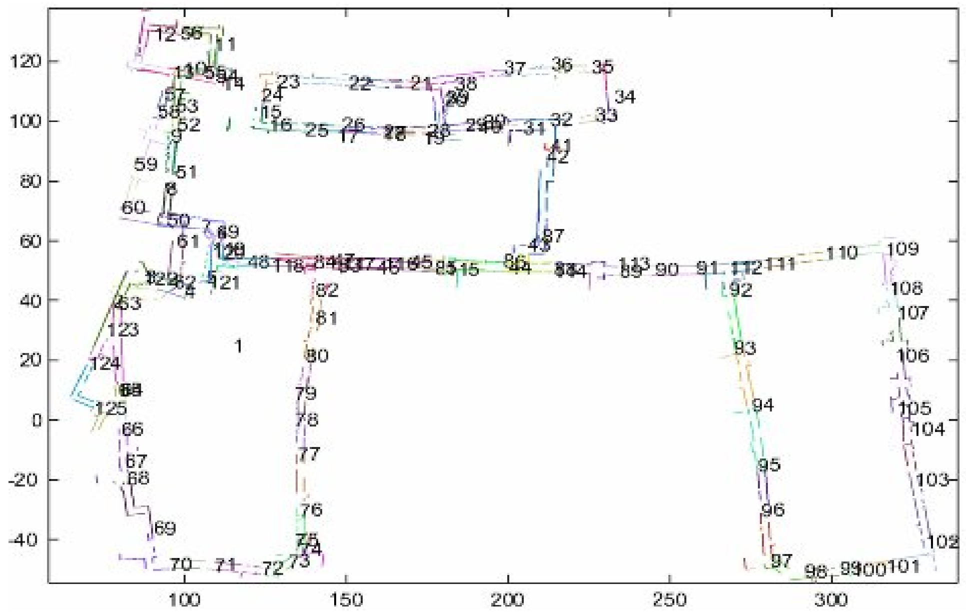
Homing…

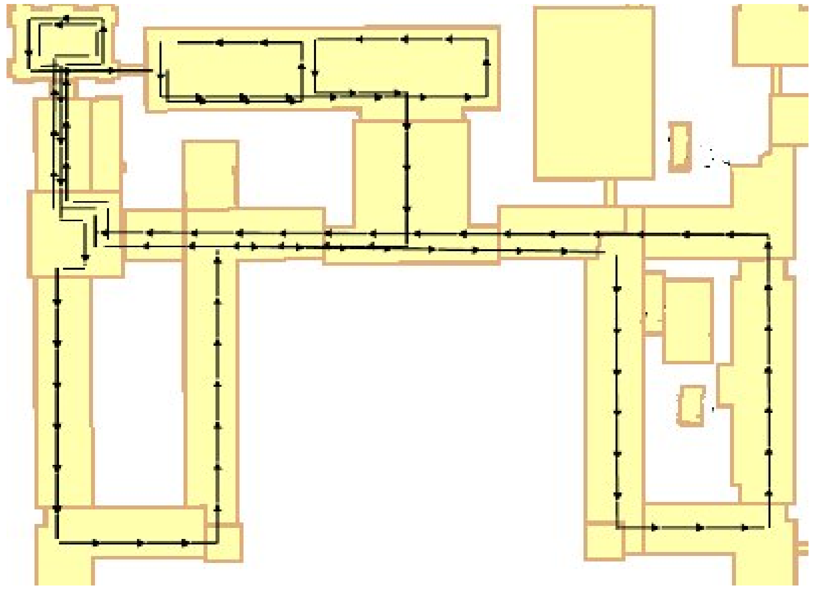
Homing…Final Adjustment



Get Insurance….

Atlas – Extending the EKF Using Multiple Frame





•Mapping MIT's "Infinite Corridor"
•2.2 km driven path

In collaboration with M. Bosse (PhD Thesis) Prof. J. Leonard and Prof. S. Teller (MIT)

The Convergence and Stability of SLAM

By analysing the behaviour of the LG-KF we can learn about the governing properties of the SLAM problem – which are actually completely intuitive….

We can show that:

•The determinant of any submatrix of the map covariance matrix decreases monotonically as observations are successively made.

•In the limit as the number of observations increases, the landmark estimates become fully correlated.

•In the limit, the covariance associated with any single landmark location estimate is determined only by the initial covariance in the vehicle location estimate.

$$\begin{aligned}\det \mathbf{P}(k+1|k+1) &= \det[\mathbf{P}(k+1|k) - \mathbf{W}_i(k+1)\mathbf{S}_i(k+1)\mathbf{W}^T(k+1)] \\ &\leq \det \mathbf{P}(k+1|k) \qquad (1)\end{aligned}$$

Any principal submatrix of a *psd* matrix is also *psd* Thus,

$$\det \mathbf{P}_{mm}(k+1|k+1) \leq \det \mathbf{P}_{mm}(k+1|k) \qquad (2)$$

From the prediction equations (features don't change)

$$\mathbf{P}(k+1|k) = \begin{bmatrix} \mathbf{F}_v \mathbf{P}_{vv}(k|k)\mathbf{F}_v{}^T + \mathbf{Q}_{vv} & \mathbf{F}_v \mathbf{P}_{vm}(k|k) \\ \mathbf{P}_{vm}^T(k|k)\mathbf{F}_v{}^T & \mathbf{P}_{mm}(k|k) \end{bmatrix}.$$

$$\det \mathbf{P}_{mm}(k+1|k+1) \leq \det \mathbf{P}_{mm}(k|k). \qquad (3)$$

$$\boxed{\sigma_{ii}^2(k+1|k+1) \leq \sigma_{ii}^2(k|k).}$$

where σ_{ii}^2 is a diagnoal element of $\mathbf{P}_{mm}$

As the number of observations taken tends to infinity a lower limit on the map covariance limit will be reached such that

$$\lim_{k \to \infty} [\mathbf{P}_{mm}(k+1|k+1)] = \mathbf{P}_{mm}(k|k) \tag{1}$$

Writing $\mathbf{P}(k+1|k)$ as $\mathbf{P}^{\ominus}$ and $\mathbf{P}(k+1|k+1)$ as $\mathbf{P}^{\oplus}$ for notational clarity the update for the map can be written as

$$\begin{aligned}\mathbf{P}_{mm}(k+1|k+1) &= \mathbf{P}_{mm}(k+1|k) - \mathbf{M}_2\mathbf{S}_i^{-1}{\mathbf{M}_2}^T \\ &= \mathbf{P}_{mm}(k|k) - \mathbf{M}_2\mathbf{S}_i^{-1}{\mathbf{M}_2}^T \end{aligned} \tag{2}$$

where

$$\mathbf{M}_2 = -\mathbf{P}_{vm}^{\ominus}{\mathbf{H}_v}^T + \mathbf{P}_{mm}^{\ominus}{\mathbf{H}_{pi}}^T \tag{3}$$

In the limit then $\mathbf{M}_2\mathbf{S}_i^{-1}{\mathbf{M}_2}^T = \mathbf{0}$ which implies that $\mathbf{M}_2 = \mathbf{0}$

$$\mathbf{P}_{mm}(k|k){\mathbf{H}_{pi}}^T = \mathbf{P}_{vm}(k|k){\mathbf{H}_v}^T \quad \forall i \tag{1}$$

this holds for all i and therefore the block columns of $\mathbf{P}_{mm}$ are linearly dependent and therefore

$$\lim_{k\to\infty} [\det \mathbf{P}_{mm}(k|k)] = \mathbf{0} \tag{2}$$

Consider now the relative position between any two landmarks $\mathbf{p}_i$ and $\mathbf{p}_j$ of the same type.

$$\begin{aligned} \mathbf{d} &= \hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j \\ &= \mathbf{G}_{ij}\mathbf{x} \end{aligned}$$

The covariance $\mathbf{P}_d$ of $\mathbf{d}$ is given by

$$\begin{aligned} \mathbf{P}_d &= \mathbf{G}_{ij}\mathbf{P}{\mathbf{G}_{ij}}^T \\ &= \mathbf{P}_{ii} + \mathbf{P}_{jj} - \mathbf{P}_{ij} - {\mathbf{P}_{ij}}^T \end{aligned}$$

With similar landmarks types, $\mathbf{H}_{pi} = \mathbf{H}_{pj}$ and so Equation ?? implies that the block columns of $\mathbf{P}_{mm}$ are identical. Furthermore, because $\mathbf{P}_{mm}$ is symmetric it follows that

$$\mathbf{P}_{ii} = \mathbf{P}_{jj} = \mathbf{P}_{ij} = {\mathbf{P}_{ij}}^T \quad \forall i, j \tag{3}$$

Therefore, in the limit, $\mathbf{P}_d = \mathbf{0}$ and the relationship between the landmarks is known with complete certainty.

To be Tattooed on Inside of Eyelids

•The entire structure of the SLAM problem critically depends on maintaining complete knowledge of the cross correlation between landmark estimates. Minimizing or ignoring cross correlations is precisely contrary to the structure of the problem.

•As the vehicle progresses through the environment the errors in the estimates of any pair of landmarks become more and more correlated, and indeed never become less correlated.

•In the limit, the errors in the estimates of any pair of landmarks becomes fully correlated. This means that given the exact location of any one landmark, the location of any other landmark in the map can also be determined with absolute certainty.

•As the vehicle moves through the environment taking observations of individual landmarks, the error in the estimates of the relative location between different landmarks reduces monotonically to the point where the map of relative locations is known with absolute precision.

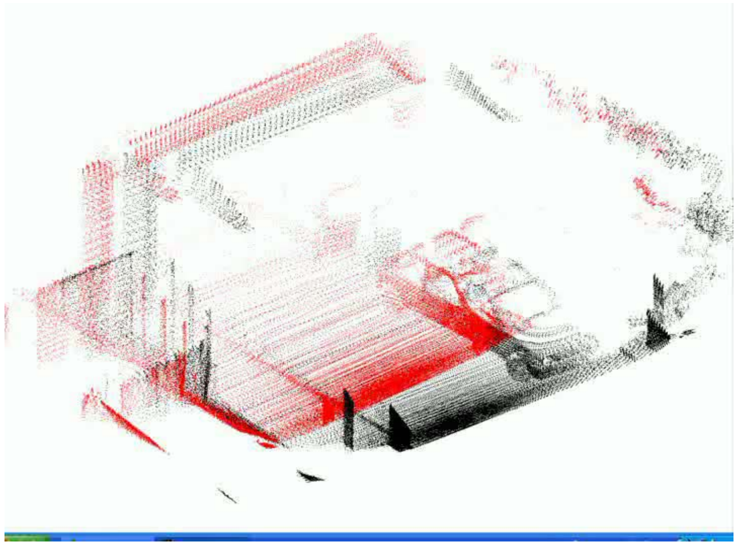
•As the map converges in the above manner, the error in the absolute location of every landmark (and thus the whole map) reaches a lower bound determined only by the error that existed when the first observation was made.

(We didn't prove this here. However it is an **excellent** test for consistency in new SLAM algorithms)

This is all under the assumption that we observe all features equally often…

– for other cases see Kim Sun-Joon PhD MIT 2004

Pose Based SLAM



Scan matching between past poses produces observation with which to update state-vector

$$\mathbf{x}_v(k+1|k) = \mathbf{x}_v(k|k) \oplus \mathbf{u}(k+1)$$

Control

$$\mathbf{x}(k+1|k) = \begin{bmatrix} \mathbf{x}(k|k) \\ \mathbf{x}_v(k|k) \oplus \mathbf{u}(k+1) \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{x}_v(0|0) \\ \vdots \\ \mathbf{x}_v(k|k) \\ \mathbf{x}_v(k+1|k) \end{bmatrix}$$

Past poses

Growth with time

- •State vector contains only past vehicle poses.
- •Each pose has a scan rigidly attached to it

Pose Based Prediction Step

$$\mathbf{x}(k+1|k) = \begin{bmatrix} \mathbf{x}_v[0] \\ \vdots \\ \mathbf{x}_v[k] \\ \mathbf{x}_v[k+1] \end{bmatrix}$$

Past poses

$$= \begin{bmatrix} \mathbf{x}(k|k) \\ \mathbf{x}_v[k] \oplus \mathbf{u}(k+1) \end{bmatrix}$$

Writing this as a transformation allows us to calculate the augmented state covariance

$$\mathbf{x}(k+1|k) = f(\mathbf{x}(k|k), \mathbf{u}(k+1))$$

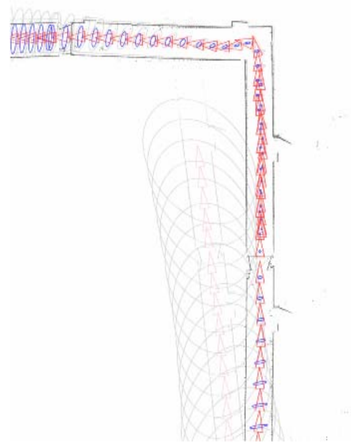
$$\nabla \mathbf{F_x} = \begin{bmatrix} & \mathbf{I}_{n\times n} \\ \mathbf{0}_{3\times n-3} \quad | & \mathbf{J}_1(\mathbf{x}_v[k], \mathbf{u}(k+1)) \end{bmatrix}$$

$$\nabla \mathbf{F_u} = \begin{bmatrix} & \mathbf{0}_{n\times n} \\ \mathbf{0}_{3\times n-3} \quad | & \mathbf{J}_2(\mathbf{x}_v[k], \mathbf{u}(k+1)) \end{bmatrix}$$

Note f() changes the size of the state vector!

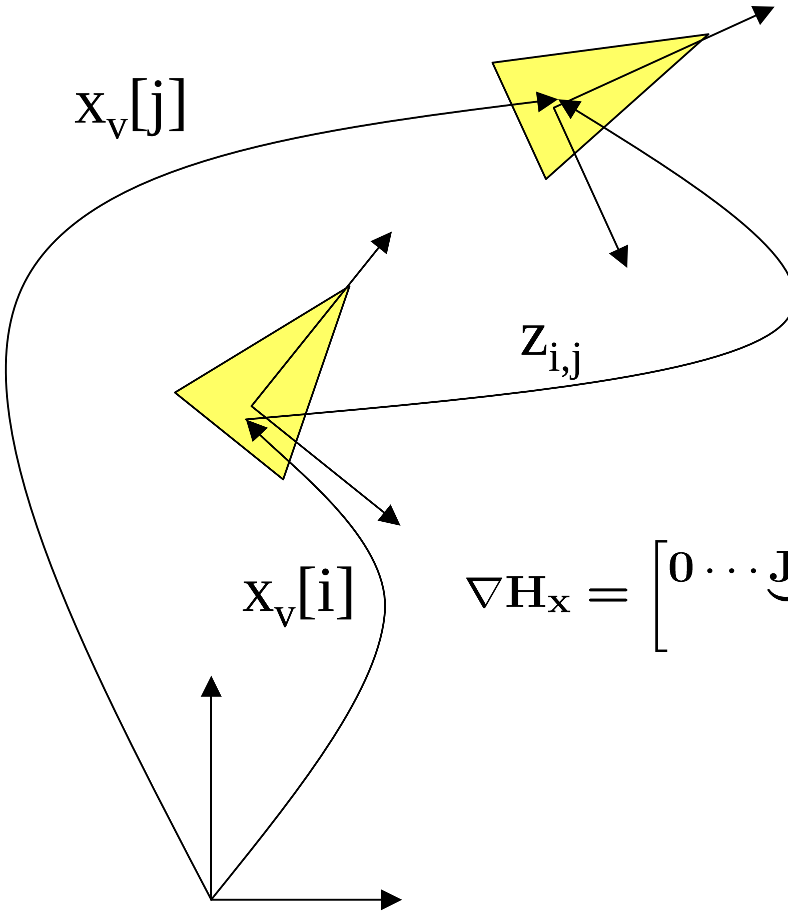
and so

$$\mathbf{P}(k+1|k) = \nabla \mathbf{F_x} \mathbf{P}(k|k) \nabla \mathbf{F_x}^T + \nabla \mathbf{F_u} \mathbf{U} \nabla \mathbf{F_u}^T$$



Plant model (f) and Jacobian are all we need to plug into EKF prediction equations

Pose Based Update Step



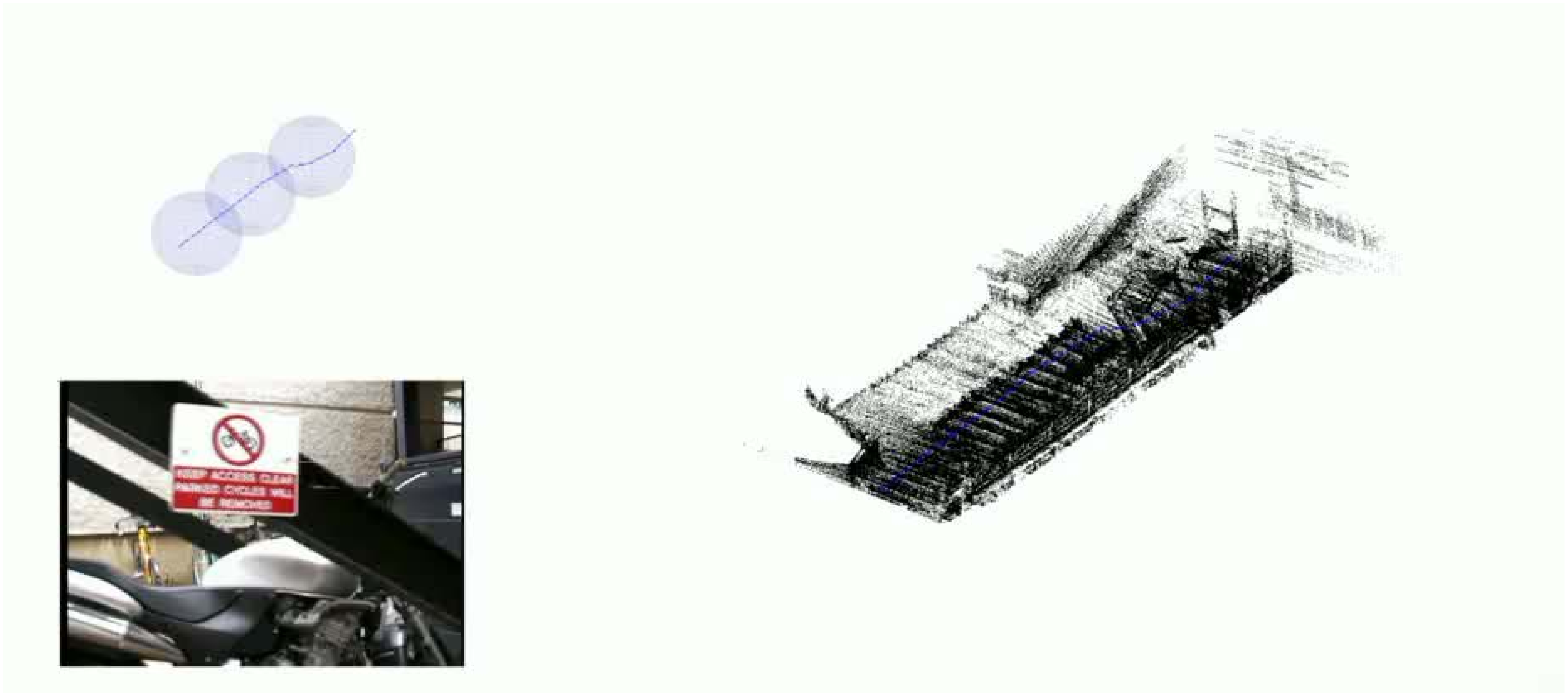
$$\mathbf{z}_{i,j}(k) = \ominus \mathbf{x}_v[i] \oplus \mathbf{x}_v[j] + \overbrace{\mathbf{w}(k)}^{\text{white noise}}$$
$$= h(\mathbf{x}(k)) + \mathbf{w}(k)$$

$$\nabla \mathbf{H_x} = \left[\mathbf{0} \cdots \underbrace{\mathbf{J}_1(\ominus \mathbf{x}_v[i], \mathbf{x}_v[j]) \mathbf{J}_m(\mathbf{x}_v[i])}_{\text{state i}} \cdots \mathbf{0} \cdots \underbrace{\mathbf{J}_2(\ominus \mathbf{x}_v[i], \mathbf{x}_v[j])}_{\text{state j}} \cdots \mathbf{0} \cdots \right]$$

Observation model (h) and Jacobian are all we need to plug into EKF update equations

RUN LIVE DEMO HERE

Demonstration 3D Pose Based SLAM



Issues with Single Frame SLAM

• It is uni-modal. It cannot cope with ambiguous situations
• It is inconsistent - the linearisations lead to errors which underestimate the covariance of the underlying pdf
• It is fragile - if the estimated is in error the linearisation is very poor – disaster.

• But the biggest problem is (was) scaling

Much progress has been made on scaling – see upcoming talks. In particular
- •FastSLAM Montemerlo et. al
- •Atlas(Bosse et al)
- •Postponement (Davison, Nebot)
- •CTS (Leonard Newman)
- •Exactly Sparse Delayed State Filter (Eustice et. Al (see third practical))

And Finally: turn it all on its Head – the information form

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \hat{\mathbf{x}}, \mathbf{P})$$

$$\alpha \exp\{-\frac{1}{2}(\mathbf{x} - \hat{\mathbf{x}})^T \mathbf{P}^{-1} (\mathbf{x} - \hat{\mathbf{x}})\}$$

$$\alpha \exp\{-\frac{1}{2}\mathbf{x}^T \mathbf{P}^{-1} \mathbf{x} + \hat{\mathbf{x}}^T \mathbf{P}^{-1} \mathbf{x}\}$$

$$= \exp\{-\frac{1}{2}\mathbf{x}^T \Lambda \mathbf{x} + \eta^T \mathbf{x}\}$$

where

Information Matrix is inverse of covariance (makes sense!)

$$\Lambda = \mathbf{P}^{-1}$$

$$\eta = \Lambda \hat{\mathbf{x}}$$

We can propagate information matrix an information vector instead of state and covariance

Look Again at Augmentation

$$\hat{\mathbf{x}} = \begin{bmatrix} \overbrace{\mathbf{x}_v[0:k-1]}^{\mathbf{x}_m} \\ \vdots \\ \mathbf{x}_v[k] \\ f(\mathbf{x}_v[k],\mathbf{u}) \end{bmatrix} \quad \mathbf{P} = \begin{bmatrix} \mathbf{P}_{m,m} & \mathbf{P}_{m,v} & \mathbf{P}_{m,v}\nabla\mathbf{F_x}^T \\ \mathbf{P}_{m,v}^T & \mathbf{P}_{v,v} & \mathbf{P}_{v,v}\nabla\mathbf{F_x}^T \\ \nabla\mathbf{F_x}\mathbf{P}_{m,v} & \nabla\mathbf{F_x}\mathbf{P}_{v,v} & \nabla\mathbf{F_x}\mathbf{P}_{v,v}\nabla\mathbf{F_x}^T + \underbrace{\nabla\mathbf{F_u}\mathbf{U}\nabla\mathbf{F_u}^T}_{\mathbf{Q}} \end{bmatrix}$$

$$\Lambda = \mathbf{P}^{-1} \quad \eta = \Lambda\hat{\mathbf{x}}$$

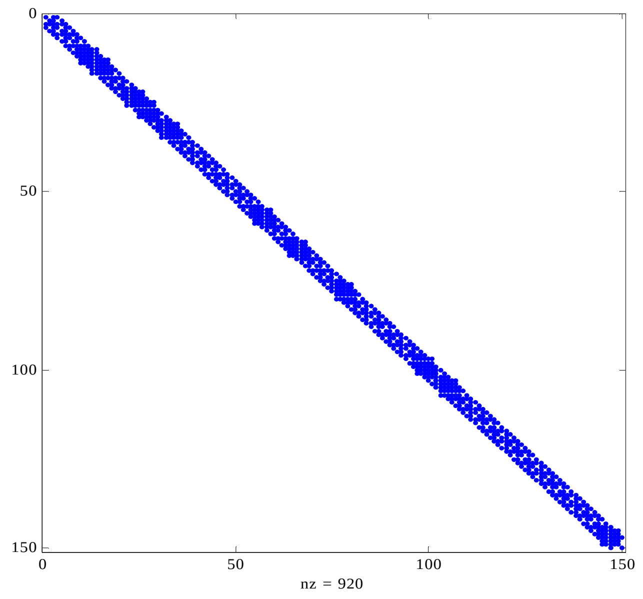
Matrix inversion lemma

$$\eta = \begin{bmatrix} \eta_m \\ \eta_v - \nabla\mathbf{F_x}^T\mathbf{Q}^{-1}\left(f(\mathbf{x}_v[k],\mathbf{u}) - \nabla\mathbf{F_x}\mathbf{x}_v[k]\right) \\ \mathbf{Q}^{-1}\left(f(\mathbf{x}_v[k],\mathbf{u}) - \nabla\mathbf{F_x}\mathbf{x}_v[k]\right) \end{bmatrix}$$

$$\Lambda = \begin{bmatrix} \Lambda_{m,m} & \Lambda_{m,v} & \mathbf{0} \\ \Lambda_{v,m} & \Lambda_{v,v} + \nabla\mathbf{F_x}^T\mathbf{Q}^{-1}\nabla\mathbf{F_x} & -\nabla\mathbf{F_x}^T\mathbf{Q}^{-1} \\ \mathbf{0} & -\mathbf{Q}^{-1}\nabla\mathbf{F_x}^T & \mathbf{Q}^{-1} \end{bmatrix}$$

Note the zeros in the information matrix that don’t appear in the EKF version…..

So What?



The information matrix remains band-diagonal when augmenting! -> constant time. (compare with EKF which is linear with state size)

•Why? Because of the markov property of the plant model – we only need the previous state to predict a new pose.
•There is much that can be said about the structure of the information matrix and the underlying pd.f but time is short….

And there's more…

Information update equations are:

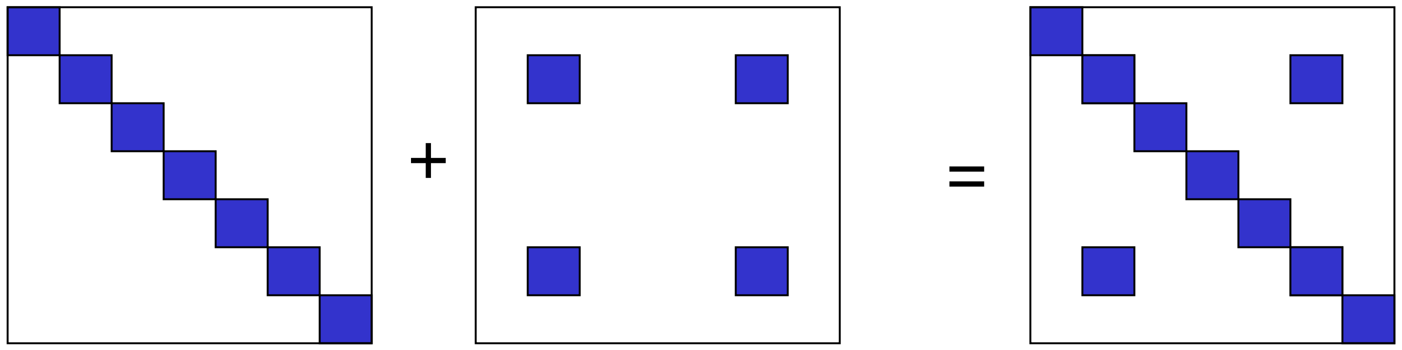
$$\eta^{+} = \eta^{-} + \nabla\mathbf{H_x}^{T}\mathbf{R}^{-1}(\mathbf{z} - h(\mathbf{x}) + \nabla\mathbf{H_x}\mathbf{x})$$
$$\Lambda^{+} = \Lambda^{-} + \nabla\mathbf{H_x}^{T}\mathbf{R}^{-1}\nabla\mathbf{H_x}$$

where

$$\nabla\mathbf{H_x} = \left[\mathbf{0}\cdots\nabla\mathbf{H_{xi}}\cdots\mathbf{0}\cdots\nabla\mathbf{H_{xj}}\cdots\mathbf{0}\cdots\right]$$

so Information update is additive and very very sparse:

$$\Lambda^{-} \quad + \quad \nabla\mathbf{H_x}^{T}\mathbf{R}^{-1}\nabla\mathbf{H_x} \quad = \quad \Lambda^{+}$$



•It appears then that the update is constant time…..

But There’s No Free Lunch

The augmentation and update equations need knowledge of $\mathbf{x}$ which is also needed to evaluate $\nabla\mathbf{H}_\mathbf{x}$ and $\nabla\mathbf{F}_\mathbf{x}$. Naively we could solve as

$$\mathbf{x} = \Lambda^{-1}\eta$$

but that would be stupid:

- Cubic cost!
- We don’t want all of $\Lambda^{-1} = \mathbf{P}$ at run time

Better plan is to solve for $\mathbf{x}$ using Cholesky decomposition as Λ is P.S.D.

- note that because $\nabla\mathbf{H}_\mathbf{x}$ and $\nabla\mathbf{F}_\mathbf{x}$ are so sparse we only ever need part of the state vector.
- this allows us to effectively side step the cost of solving for all $\mathbf{x}$ if we keep a broadly correct $\mathbf{x}$ around (problem when loop closing though)

•More details in Eustice et al’s ICRA 2005 paper
•Relates strongly to contemporary graphical-SLAM methods (see upcoming talks)

RUN LIVE DEMO HERE

What remains under the carpet?

- Data Association (what am I looking at)
- Loop Closing (was I really that lost?)
- Consistency and Robustness
- Scaling
- Relationship to large scale optimisation

- These are meaty topics and will be the focus of other summer school lectures….

Many thanks for your time