$$M = \left\{ \overbrace{(x_{ek}, y_{ek})}^{m_k}, k = 1 \dots N \right\} = \left\{ \overbrace{(x_{e1}, y_{e1})}^{m_1}, \overbrace{(x_{e2}, y_{e2})}^{m_2} \dots \overbrace{(x_{eN}, y_{eN})}^{m_N} \right\}$$

↳ Set of all landmarks' coordinates

$m_k = (x_{ek}, y_{ek})$ → Coordinates of the **landmark** stored at **index k** in the **landmark list** for a particular particle

$$p(x_{1:t}, M \mid z_{1:t}, u_{1:t}, c_{1:t}) = p(M \mid x_{1:t}, z_{1:t}, u_{1:t}, c_{1:t}) \cdot p(x_{1:t} \mid z_{1:t}, u_{1:t}, c_{1:t})$$

(correspondences)

$$= \left[\prod_{k=1}^{N^k} p(m_k \mid x_{1:t}, z_{1:t}, c_{1:t}) \right] \cdot p(x_{1:t} \mid z_{1:t}, u_{1:t}, c_{1:t})$$

(N^k = # of landmarks for a particular particle; i.i.d)

One EKF for each landmark

Represented with particles

If you don't have an observation at a time step, you don't need the pose at that time step, unless we want to set the center of the reference frame. ← x_0 is ignore because there is not any observation at this time step

I think that: $p(x_{0:t}, M \mid z_{1:t}, u_{1:t}) = p(M \mid \cancel{x_0}, x_{1:t}, z_{1:t}, \cancel{u_t}) \cdot p(x_{0:t} \mid z_{1:t}, u_{1:t})$

FastSLAM - Blackwellization for SLAM:

(From Cyrill Stachniss' class)

$$p(x_{0:t}, M \mid z_{1:t}, u_{1:t}) = p(x_{0:t} \mid z_{1:t}, u_{1:t}) \cdot p(M \mid x_{1:t}, z_{1:t})$$

- $x_{0:t}$: Poses
- M: Map
- $z_{1:t}$: Measurements (Observations)
- $u_{1:t}$: Movements
- $x_{0:t}$: Path posterior, Particle filter
- M: Map posterior (Given the path)

x_0 is used to set the initial uncertainty → Eg: if the initial uncertainty is 0, all the particles start at one location. Or you can distribute them according to a initial belief

- Once we know the poses of the robot, the mapping problem is easy to solve

In the SLAM-EKF lecture (lecture F)

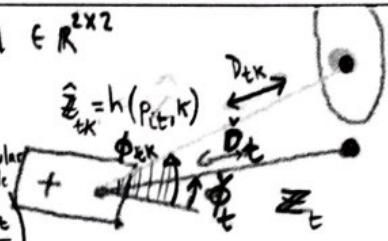
$H \in \mathbb{R}^{2 \times (3+2N)}$, here, in this lecture $H \in \mathbb{R}^{2 \times 2}$

(A) Compute likelihoods.

$i = 1 \dots M$ → Number of particles

$K = 1 \dots N$ → Number of landmarks for a particular particle

$m_K = (x_{ek}, y_{ek})$



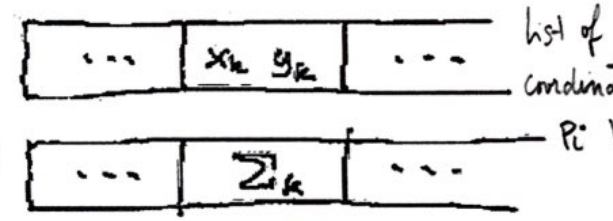
Expected measurement: $\begin{bmatrix} \check{D} \\ \check{\Phi} \end{bmatrix} = \check{z} = h(\overbrace{(x, y, \theta)}^{\text{particle } p_{it}}, j)$ Fuction already programmed

$$H = \frac{\partial h}{\partial \text{landmark}} \Big|_{(x_{it}, y_{it}, \theta_{it}, K)} \quad (2 \times 2) \quad \begin{bmatrix} \frac{\partial D_{tK}}{\partial x_{ek}} & \frac{\partial D_{tK}}{\partial y_{ek}} \\ \frac{\partial \Phi_{tK}}{\partial x_{ek}} & \frac{\partial \Phi_{tK}}{\partial y_{ek}} \end{bmatrix}$$

$\left(\frac{x_{ek} - x_{st}}{\sqrt{q_{tK}}}\right) \left(\frac{y_{ek} - y_{st}}{\sqrt{q_{tK}}}\right)$

$\left(\frac{-(y_{ek} - y_{st})}{q_{tK}}\right) \left(\frac{x_{ek} - x_{st}}{q_{tK}}\right)$

$x_{it}, y_{it}, \theta_{it}$ = Particle i, p_{it}, $i = 1 \dots M$ at time t



Covariance of the expected measurement $\left[h_{tK}(p_{it}, K)\right]$

$$Q_t = H \Sigma_t H^T + Q$$

a.k.a. (Q_t)

In the SLAM-EKF lecture we used $H_{tK} \in \mathbb{R}^{2 \times (3+2N)}$ and $\Sigma_t \in \mathbb{R}^{(3+2N) \times (3+2N)}$

and now, in this lecture, we use $H_{tK} \in \mathbb{R}^{2 \times 2}$ and $\Sigma_K \in \mathbb{R}^{2 \times 2}$

Variance of landmark k (in x, y)

Measurement variance due to the landmark variance

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

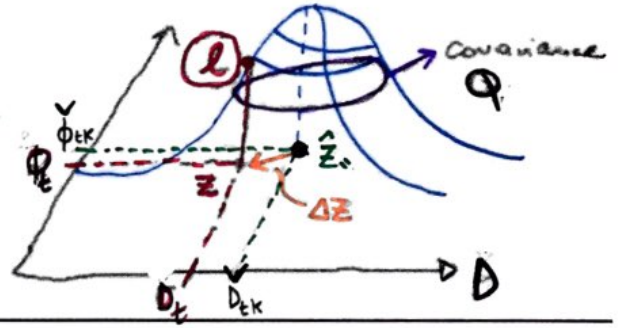
Variance due to actual measurement

$\Delta z_{tK} = z_t - \hat{z}_{tK}$

Likelihood that the measurement z_t is due to observing the landmark $m_K = (x_{ek}, y_{ek})$

$$\mathcal{L}_{it} = \frac{1}{2\pi \sqrt{\det(Q_{tK})}} \cdot e^{-\frac{1}{2} \Delta z_{tK}^T Q_{tK}^{-1} \Delta z_{tK}}$$

z_t to a detected cylinder.



z_t is $\begin{bmatrix} \check{D}_t \\ \check{\Phi}_t \end{bmatrix}$, z_{tK} is $\begin{bmatrix} D_{tK} \\ \Phi_{tK} \end{bmatrix} = h(p_{it}, K)$

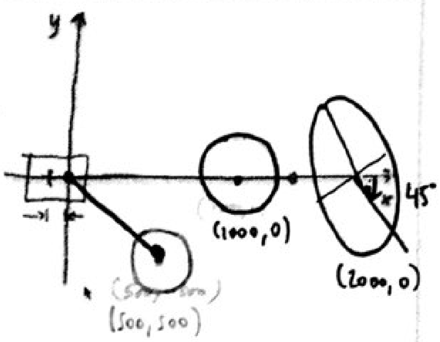
Q_{tK} (explanation) ⇒ The uncertainty in the value $h_{tK}(p_{it}, K)$, i.e, Q_{tK}, is due to the uncertainty of the landmark "k", i.e, Σ_K, which then translates into uncertainty in a distance and bearing angle, i.e, $H_{tK} \Sigma_K H_{tK}^T$, plus the uncertainty of the actual measurement of the sensor, i.e, Q_z

$$\Sigma_K = \begin{bmatrix} \sigma_{x_{ek}}^2 & \sigma_{x_{ek} y_{ek}} \\ \sigma_{x_{ek} y_{ek}} & \sigma_{y_{ek}}^2 \end{bmatrix}$$

"Not"

```
        [-100.0**2,  200.0**2 ]]))

if __name__ == '__main__':
    # Robot constants.
    scanner_displacement = 30.0

    # Filter constants.
    measurement_distance_stddev = 200.0  # Distance measurement error of cylinders.
    measurement_angle_stddev = 15.0 / 180.0 * pi  # Angle measurement error.
    Qt_measurement_covariance = \
        np.diag([measurement_distance_stddev**2,
                 measurement_angle_stddev**2])

    # Define a particle: position (x, y) and orientation.
    p = Particle(np.array([-scanner_displacement, 0.0, 0.0]))
    add_landmarks(p)
    N = p.number_of_landmarks()

    # Compute expected measurements.
    for i in xrange(N):
        print 'Landmark', i, '----------'
        em = p.h_expected_measurement_for_landmark(i, scanner_displacement)
        print ' Expected range:', em[0], 'bearing [deg]:', em[1]/pi*180
        H, Ql = p.H_Ql_jacobian_and_measurement_covariance_for_landmark(
            i, Qt_measurement_covariance, scanner_displacement)
        print ' Covariance of measurement:\n', Ql

    # Compute correspondence likelihoods.
    # Define a set of measurements: (range, bearing).
    print 'Measurement likelihoods ----------'
    measurements = [
        ('close to landmark 0', np.array([500*sqrt(2), -pi/4])),
        ('exactly between landmark 1 and 2', np.array([1500.0, 0.0]))
        ]
    for (text, z) in measurements:
        print 'Likelihoods for measurement', text
        likelihoods = p.compute_correspondence_likelihoods(
            z, N, Qt_measurement_covariance, scanner_displacement)
        print likelihoods
```

Ln 140 Col 20




```
Python 2.7.3 |EPD_free 7.3-2 (32-bit)| (default, Apr 12 2012, 11:28:34)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
Landmark 0 ----------
 Expected range: 707.106781187 bearing [deg] -45.0
 Covariance of measurement:
[[  5.00000000e-04   0.00000000e+00]
 [  0.00000000e+00   4.85389195e-02]]
Landmark 1 ----------
 Expected range: 1000.0 bearing [deg] 0.0
 Covariance of measurement:
[[  5.00000000e-04   0.00000000e+00]
 [  0.00000000e+00   7.85389195e-02]]
Landmark 2 ----------
 Expected range: 2000.0 bearing [deg] 0.0
 Covariance of measurement:
[[  4.00000000e-04  -5.00000000e+00]
 [ -5.00000000e+00   7.85389195e-02]]
Measurement likelihoods ----------
Likelihoods for measurement close to landmark 0
[0.0023926377034520276, 2.1215[illegible]e-05, 4.826[illegible]e-10]
Likelihoods for measurement exactly between landmark 1 and 2
[1.3664163924358227e-07, 0.00020847619108492658, 0.00041908533374704956]
>>>
```

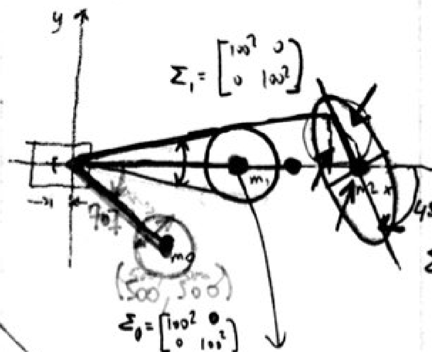
Ln 26 Col 4

Q_{t0} ←

Q_{t1} —

Q_{t2} —

Correlation between range (ρ_{t2}) and bearing angle (ϕ_{t2})



$$\Sigma_K = \begin{bmatrix} \sigma^2_{x_{ek}} & 0 \\ 0 & \sigma^2_{y_{ek}} \end{bmatrix}$$

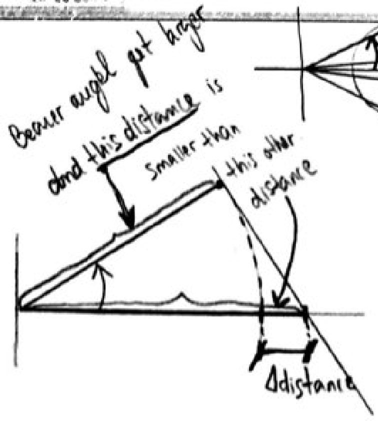
$$\Sigma_2 = \begin{bmatrix} 200^2 & -(100)^2 \\ -(100)^2 & 200^2 \end{bmatrix}$$

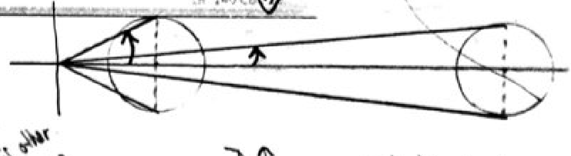
The further the point is the lesser the uncertainty in the bearing angle

Ln 140 Col

If the bearing angle gets larger the distance gets smaller → thats why the - in the term $_\phi$ -5'0000 ⇒ ↑ → ↓

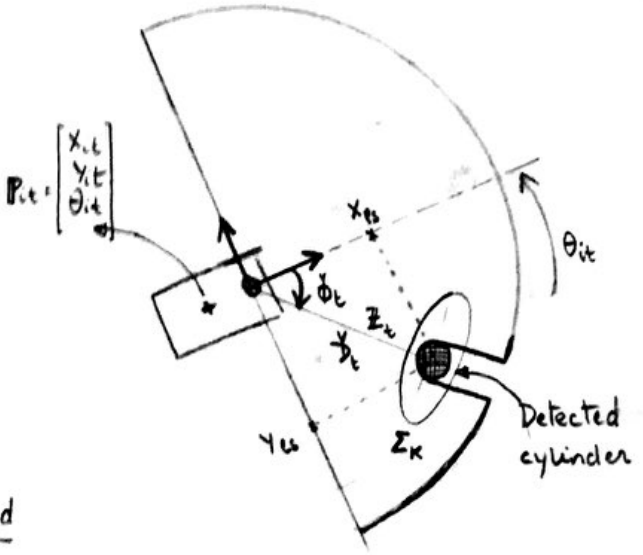
⟹





$\rho_{tk}\uparrow$ → uncertainty $\phi_{tk}\downarrow$

(3) Incorporate a new Landmark



After computing the likelihood for the landmark in the picture the algorithm determines that this landmark is a new landmark and it should be incorporated into the list of landmarks for the particle the algorithm is working with.

$$\underbrace{(x_{lk}, y_{lk})}_{m_k} = \underbrace{h^{-1}(P_{it}, sd, z_t)}$$

- With P_{it} and sd compute the scanner's pose
- With z_t compute (x_{sk}, y_{sk}), i.e, the landmark's coordinates in the scanner's (sensor) reference frame.
- Invoke LegoLogfile.scanner_to_world (scanner_pose, (x_{sk}, y_{sk}))

$$H_{tk} = \frac{\partial h}{\partial m_k} = \begin{bmatrix} \frac{\partial d_{tk}}{\partial x_{lk}} & \frac{\partial d_{tk}}{\partial y_{lk}} \\ \frac{\partial \phi_{tk}}{\partial x_{lk}} & \frac{\partial \phi_{tk}}{\partial y_{lk}} \end{bmatrix}$$

(x_{lk}, y_{lk})

translates the uncertainty in the landmark's position, Σ_k, into an uncertainty in the (propagates)

expected measurement, Q_{tk}.

Now, the situation is different. The robot takes a scan and the algorithm makes an observation in the scan (detects a cylinder), i.e, $z_t = \begin{pmatrix} \check{d}_t \\ \check{\phi}_t \end{pmatrix}$. The uncertainty in this observation, Q_z, translates into an uncertainty in the landmark's position, Σ_k. (error)

So we need the jacobian of $h_t^{-1}(.)$, i.e, the inverse of $H_{tk} \Rightarrow H_{tk}^{-1}$

$$\Sigma_k = \underbrace{H_{tk}^{-1} \cdot Q_z \cdot \left(H_{tk}^{-1}\right)^T}_{\begin{bmatrix} \sigma^2_{\check{d}_t} & 0 \\ 0 & \sigma^2_{\check{\phi}_t} \end{bmatrix}} = \left[\begin{pmatrix} V_{||} & V_{\perp} \end{pmatrix} \cdot \begin{bmatrix} \lambda_{||}^2 & 0 \\ 0 & \lambda_{\perp}^2 \end{bmatrix} \cdot \begin{pmatrix} V_{||} & V_{\perp} \end{pmatrix}^T \right]$$

$\alpha_{||} = \operatorname{atan}\left(\frac{V_{||y}}{V_{||x}}\right)$

$\alpha_{\perp} = \alpha_{||} \pm 90^{\circ}$

$= \operatorname{atan}^{rr}\left(\frac{V_{\perp y}}{V_{\perp x}}\right)$

$V_{||}$: Eigenvector showing the direction of the ellipse axis parallel to the scan ray

$V_{\perp}$: Eigenvector showing the direction of the ellipse axis perpendicular to the scan ray

$\lambda_{||}$: Semilength of the ellipse axis parallel to the scan ray.

$\lambda_{\perp}$: Semilength of the ellipse axis perpendicular to the scan ray.

Previously we had the covariance of the observation Q_z and the covariance of the landmark, Σ_k. We computed H_{tk} and with those terms we computed the covariance of the expected measurement, Q_{tk}

$$Q_{tk} = H_{tk} \Sigma_k H_{tk}^T + Q_z$$

Now, we have only the covariance of the observation Q_z. We compute H_{tk}^{-1} and then:

$$\Sigma_k = H_{tk}^{-1} Q_z \left(H_{tk}^{-1}\right)^T$$

That's the only term we have in this situation.

Exercises of adding new landmarks

$z_t^{(1)} = \begin{bmatrix} \check{D}_t \\ \check{\Phi}_t \end{bmatrix} = \begin{bmatrix} 1000 \\ 0° \end{bmatrix} \xrightarrow{h^{-1}(\cdot)}$ Add a new landmark at $(1000, 0)$

$$\Sigma_1 = \begin{bmatrix} 200^2 & 0 \\ 0 & 261.8^2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 200^2 & 0 \\ 0 & 261.8^2 \end{bmatrix} \left(V_{\parallel}\, V_{\perp}\right)^T$$

($V_{\parallel}$, $V_{\perp}$, $\lambda_{\parallel}$, $\lambda_{\perp}$)

$\alpha_{\parallel} = \text{atan}\left(\frac{V_{\parallel y}}{V_{\parallel x}}\right) = \text{atan}\left(\frac{0}{1}\right) = 0° \rightarrow \lambda_{\parallel} = 200$

$\alpha_{\perp} = \text{atan}\left(\frac{V_{\perp y}}{V_{\perp x}}\right) = \text{atan}\left(\frac{1}{0}\right) = 90° \rightarrow \lambda_{\perp} = 261.8$

$z_t^{(2)} = \begin{bmatrix} \check{D}_t \\ \check{\Phi}_t \end{bmatrix} = \begin{bmatrix} 2000 \\ 0 \end{bmatrix} \xrightarrow{h^{-1}(\cdot)}$ Add a new landmark at $(2000, 0)$

$$\Sigma_2 = \begin{bmatrix} 200^2 & 0 \\ 0 & 523.6^2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 200^2 & 0 \\ 0 & 523.6^2 \end{bmatrix} \left(V_{\parallel}\, V_{\perp}\right)^T$$

($V_{\parallel}$, $V_{\perp}$, $\lambda_{\parallel}$, $\lambda_{\perp}$)

$\alpha_{\parallel} = \text{atan}\left(\frac{V_{\parallel y}}{V_{\parallel x}}\right) = \text{atan}\left(\frac{0}{1}\right) = 0° \rightarrow \lambda_{\parallel} = 200$

$\alpha_{\perp} = \text{atan}\left(\frac{V_{\perp y}}{V_{\perp x}}\right) = \text{atan}\left(\frac{1}{0}\right) = 90° \rightarrow \lambda_{\perp} = 523.6$

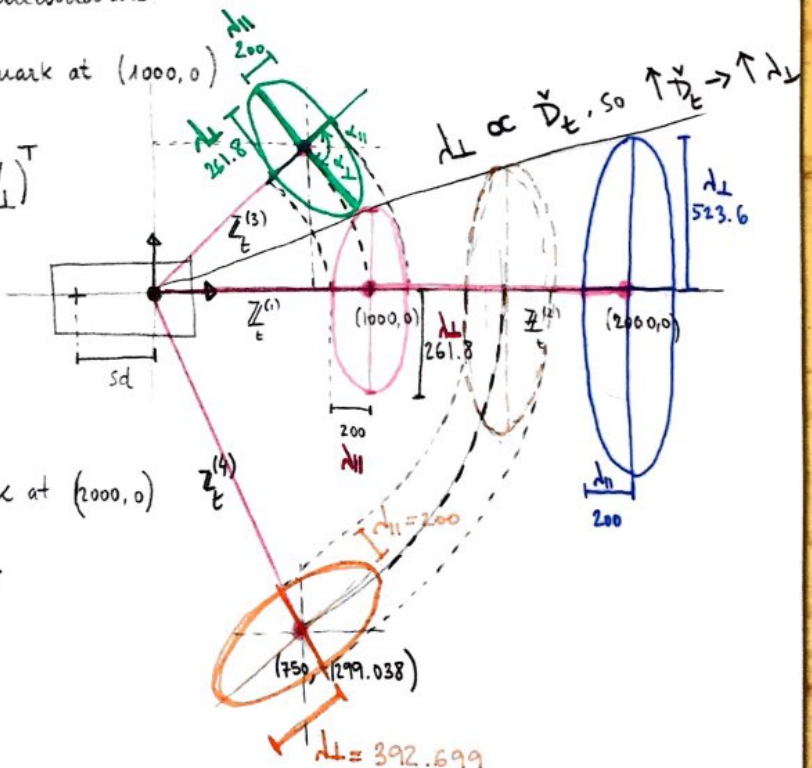
$z_t = \begin{bmatrix} \check{D}_t \\ \check{\Phi}_t \end{bmatrix} = \begin{bmatrix} 2000 \\ 45° \end{bmatrix} \xrightarrow{h^{-1}(\cdot)}$ Add a new landmark at $(1000\sqrt{2}, 1000\sqrt{2})$ $\quad 707.107 \quad 707.107$

$$\Sigma_3 = \begin{bmatrix} 54269.46 & -14269.46 \\ -14269.46 & 54269.46 \end{bmatrix} = \begin{bmatrix} 0.707 & 0.707 \\ 0.707 & -0.707 \end{bmatrix} \begin{bmatrix} (200)^2 & 0 \\ 0 & (261.8)^2 \end{bmatrix} \left(V_{\parallel}\, V_{\perp}\right)^T$$

($V_{\parallel}$, $V_{\perp}$)

$\alpha_{\parallel} = \text{atan}\left(\frac{0.707}{0.707}\right) = 45° \rightarrow \lambda_{\parallel} = 200$

$\alpha_{\perp} = \text{atan}\left(\frac{V_{\perp y}}{V_{\perp x}}\right) = \text{atan}\left(\frac{-0.707}{0.707}\right) = -45° \rightarrow \lambda_{\perp} = 261.8$



$$\underset{t}{\not{z}}^{(4)} = \begin{bmatrix} \overset{v}{D}_t \\ \overset{v}{\Phi}_t \end{bmatrix} = \begin{bmatrix} 1500 \\ 60 \end{bmatrix} \xrightarrow{h^{-1}(\cdot)} \text{Add a new landmark at } (760, -1299.038)$$

$$\Sigma_4 = \begin{bmatrix} 125659.427 & +49455.493 \\ +49455.493 & 68553.142 \end{bmatrix} = \underbrace{\begin{bmatrix} -0.5 & 0.866 \\ 0.866 & 0.5 \end{bmatrix}}_{V_{\parallel} \quad V_{\perp}} \begin{bmatrix} (200)^2 & 0 \\ 0 & (392.699)^2 \end{bmatrix} (V_{\parallel} \; V_{\perp})^T$$

$$\alpha_{\parallel} = \text{atan}\left(\frac{V_{\parallel y}}{V_{\parallel x}}\right) = \text{atan}\left(\frac{0.866}{-0.5}\right) = -60 \longrightarrow \lambda_{\parallel} = 200$$

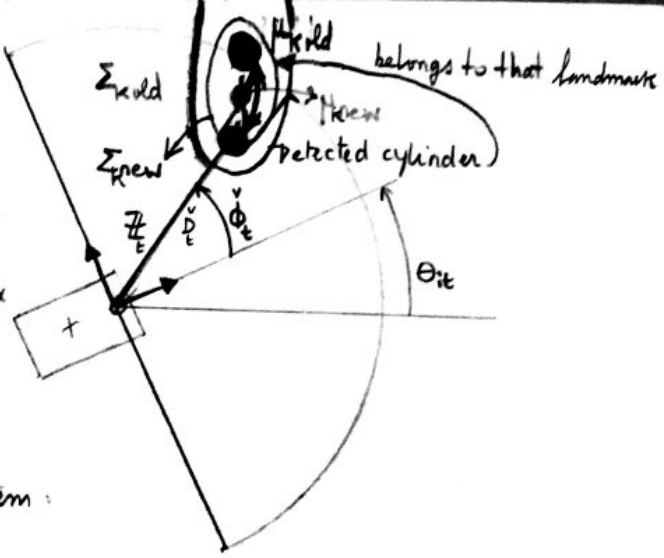
$$\alpha_{\perp} = \text{atan}\left(\frac{V_{\perp y}}{V_{\perp x}}\right) = \text{atan}\left(\frac{0.5}{0.866}\right) = 30 \longrightarrow \lambda_{\perp} = 392,699$$

$\lambda_{\perp} \propto \overset{v}{D}_t$

Therefore,

$\overset{v}{D}_t \uparrow \Rightarrow \lambda_{\perp} \uparrow$

(c) Landmark Update



As a result of this update process, the landmark's position moves a bit and the covariance matrix get smaller.

Each landmark has an EKF to keep track them:

$$K_t = \Sigma_{k\,old} \cdot H_{tk}^T \cdot \underbrace{\left(\underbrace{H_{tk}\, \Sigma_{k\,old}\, H_{tk}^T}_{\text{Uncertainty of our landmark propagated through our observation}} + \underbrace{Q_{zt}}_{\text{Observation Uncertainty}} \right)^{-1}}_{Q_{tk}}$$

$$K_t = \Sigma_{k\,old} \cdot H_{tk}^T \cdot Q_{tk}^{-1}$$

$$\mu_{k\,new} = \mu_{k\,old} + K_t \underbrace{\left(z_t - h(P_{it}, k) \right)}_{\text{Innovation}}$$

We are going to use the coordinates of the old landmark stored at index k in the list of landmarks for particle i.

$$\Sigma_{k\,new} = (I - K_t \cdot H_{tk})\, \Sigma_{k\,old}$$

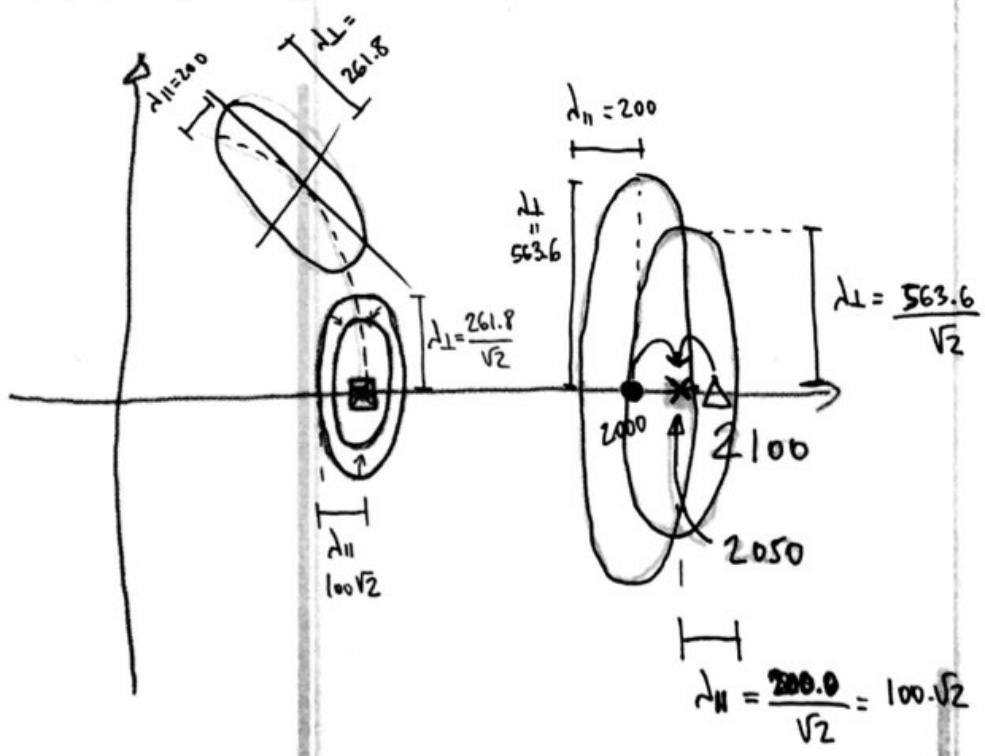
```
>>>
Landmarks - before update:
Landmark 0 ----------
 Position: [ 1000.      0.]
 Error ellipse:
  Angle [deg]: 0.0
  Axis 1: 200.0
  Axis 2: 261.799387799
Landmark 1 ----------
 Position: [ 2000.      0.]
 Error ellipse:
  Angle [deg]: 0.0
  Axis 1: 200.0
  Axis 2: 523.598775598
Landmark 2 ----------
 Position: [ 707.10678119  707.10678119]
 Error ellipse:
  Angle [deg]: -45.0
  Axis 1: 261.799387799
  Axis 2: 200.0

Landmarks - after update:
Landmark 0 ----------
 Position: [ 1000.      0.]
 Error ellipse:
  Angle [deg]: 0.0
  Axis 1: 141.421356237
  Axis 2: 185.120122423
Landmark 1 ----------
 Position: [ 2050.      0.]
 Error ellipse:
  Angle [deg]: 0.0
  Axis 1: 141.421356237
  Axis 2: 370.240244847
Landmark 2 ----------
 Position: [ 707.10678119  707.10678119]
 Error ellipse:
  Angle [deg]: -45.0
  Axis 1: 261.799387799
  Axis 2: 200.0
```

Ln 45 Col 4

200/√2 = 100·√2

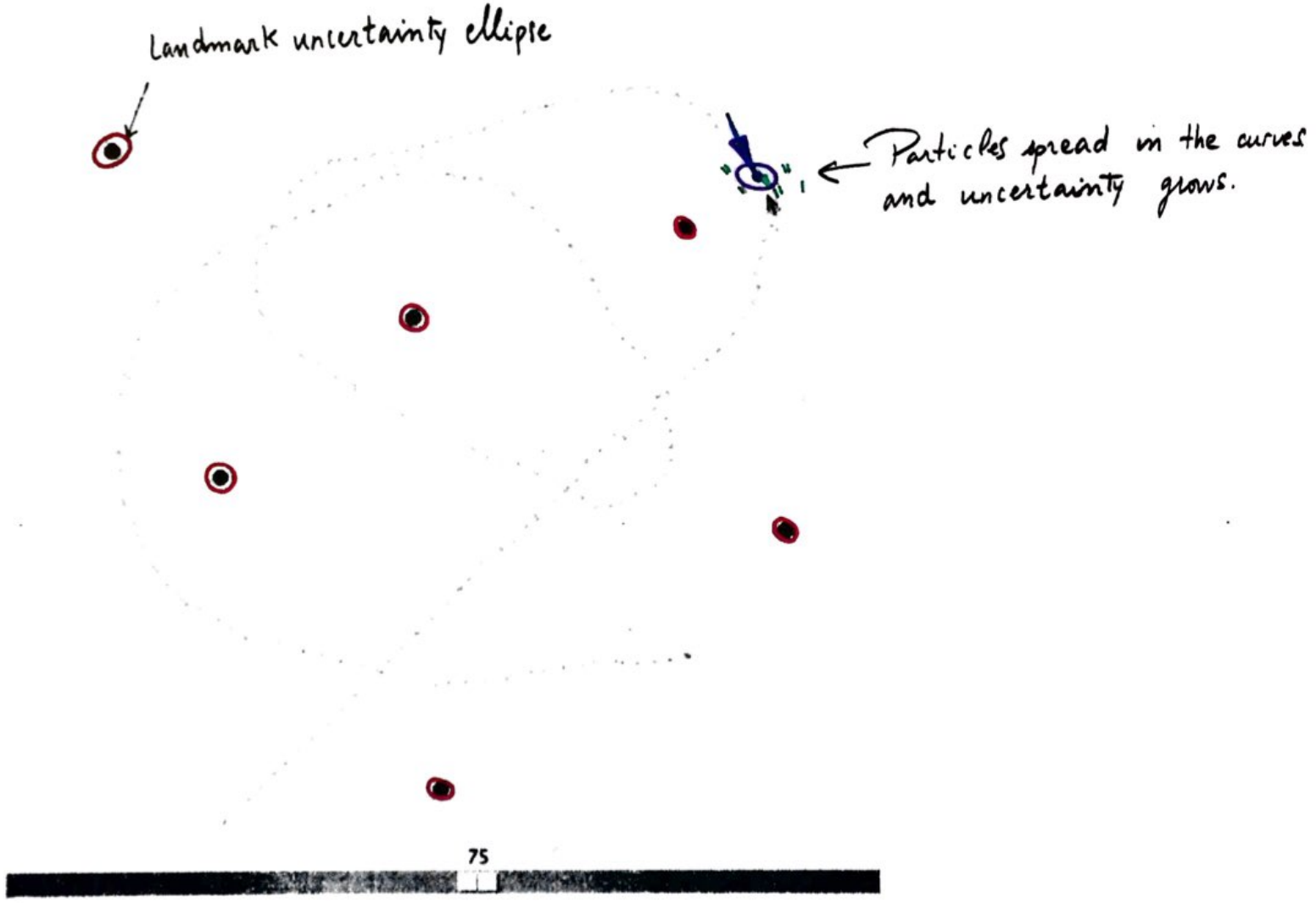
261.8/√2

... /√2

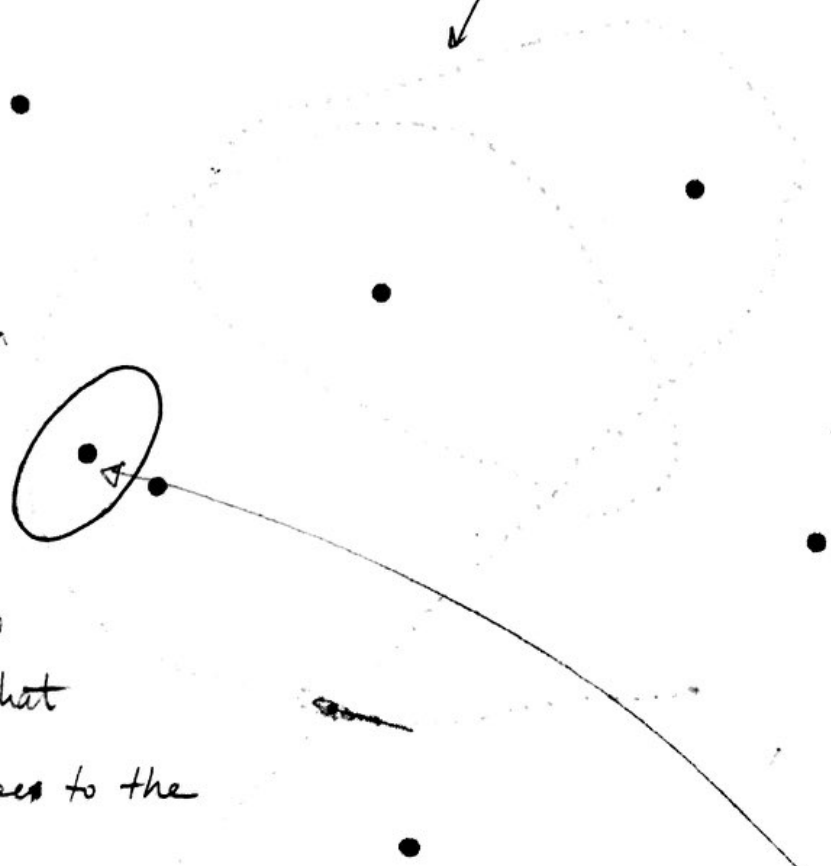


Ln 146 Col 40

S-704/2


Landmark uncertainty ellipse
Particles spread in the curves and uncertainty grows.
75

The trayectory looks a bit jagged with 25 particles.



Additional landmark which will stay until the end.

This landmark appeared at some point in some particle and finally it's obviously consistently present as a landmark in those particles that are picked because they are close to the mean state.

This Landmark remains unmodified after being inserted for all those positions of the robot. None of the observations of the robot is assigned to that landmarks anymore

We must find a way to get rid of these ghosts landmarks that appear at some point in time.

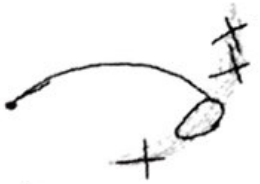
5-405/2

P final

Efficiency of FastSLAM

- Proposal distribution

"FastSLAM 1.0" → "FastSLAM 2.0"
("Probabilistic Robotics book") modifies proposal distribution (The observations are taken into account)



- Map management

M particles
N map features (landmarks)

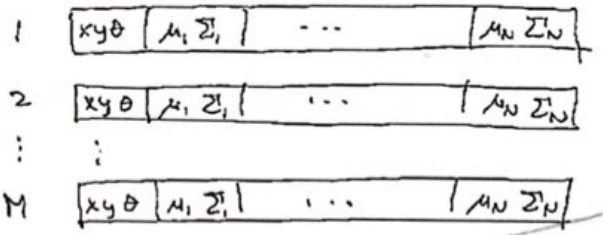
$O(MN)$

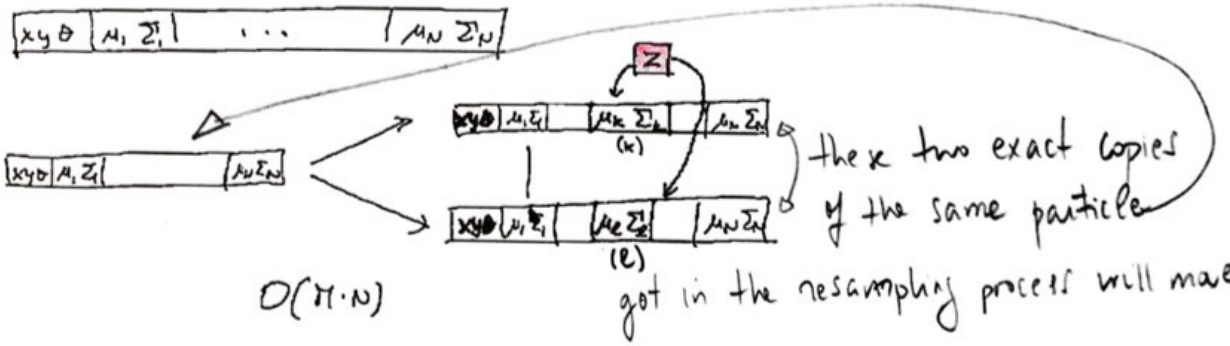
• copy

The problem of the presented fast slam approach is that our proposal is only made based on the given control and this may lead to inefficiencies because if the control is very noisy this may lead to the widespread of the particles and only a few will survive in the end, because all the others do not fit very well into the observations of our robot.

Each $z_t = \begin{bmatrix} \check{D}_t \\ \check{\Phi}_t \end{bmatrix}$ got in the scan

Implementation: $O(MN)$

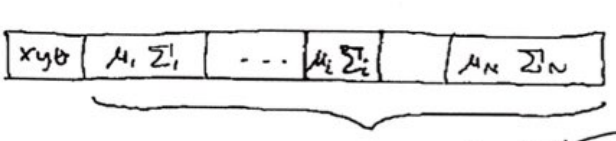




$O(M \cdot N)$

got in the resampling process will move differently in the prediction step, which won't modify the map. When the observation occurs they will be updated differently. This measurement z relates to an existing landmark in one particle, let's say landmark k, and another landmark, landmark ℓ, in the other particle. If this particles are close, these indices, probably, are the same $\Rightarrow k = \ell$. So we modify the list of landmarks only in one place, where all the other entries stay the same

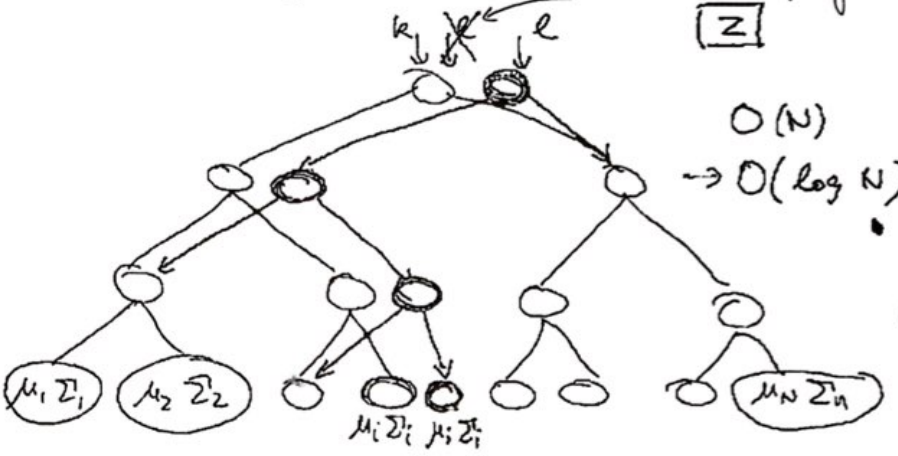
Let's think about putting all the landmarks into a balanced tree



Particle K
k ←

In the resampling make a second pointer stored in the new particle

Z



O(N)
→ O(log N)

Then, if there is at least one change, we need another root, a new root, that the pointer l will point.

Whenever we need some element with a certain index we would now have a logarithmic time complexity for accesing it

---

Efficiency

$O(MN) \rightarrow O(M \log N)$

↑ #particles ↑ #map features