In class `Particle`:

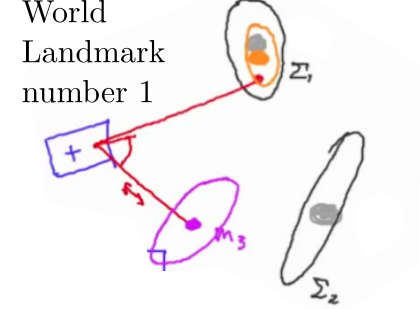
`update_particle(` $\overbrace{(D_t,\ \phi_t)}^{\vec{z}_t\ \text{(measurement)}}$ `):`

A) Compute likelihood of correspondence and compare it to a threshold.
Depending on the result:

B) Initialize a new landmark

or

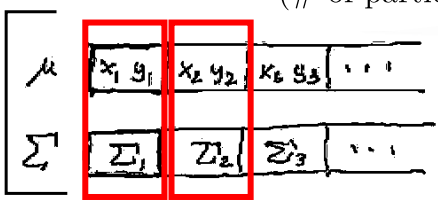
C) Update landmark j (EKF)




$\vec{P}_{kt}$ | $x_{kt}$ | $y_{kt}$ | $\theta_{kt}$ |

$k = 1, \ldots, M$
(# of particles)

For each particle




$\text{EKF}_1$ $\text{EKF}_2$

In the situation 1 the algorithm computes the likelihood between the measurement $\vec{z}_t$ and the landmark number 1.
In the situation 2 the algorithm computes the likelihood between the measurement $\vec{z}_t$ and the landmark number 2.

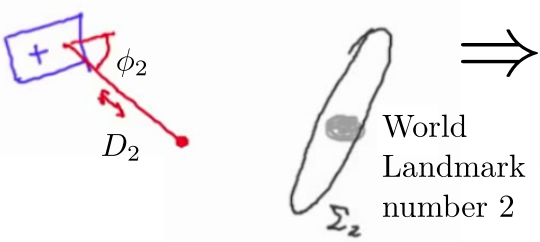
One EKF for each world landmark that a particular particle has registered.
Each EKF consists on two elements:
1º. Vector with the estimated coordinates for a registered world landmark.
2º. Covariance matrix with uncertainties.

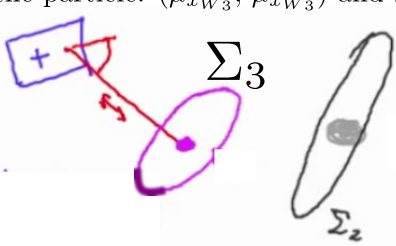
⇓

Original situation 1

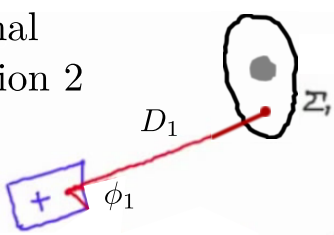

B stage

Register a new world landmark with the number 3.
Add to the particle: $(\mu_{x_{W3}},\ \mu_{x_{W3}})$ and $\Sigma_3$

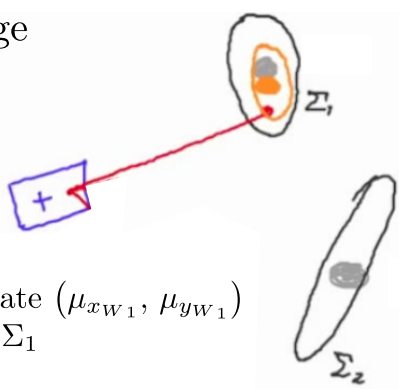



Original situation 2




C stage



Update $(\mu_{x_{W1}},\ \mu_{y_{W1}})$ and $\Sigma_1$