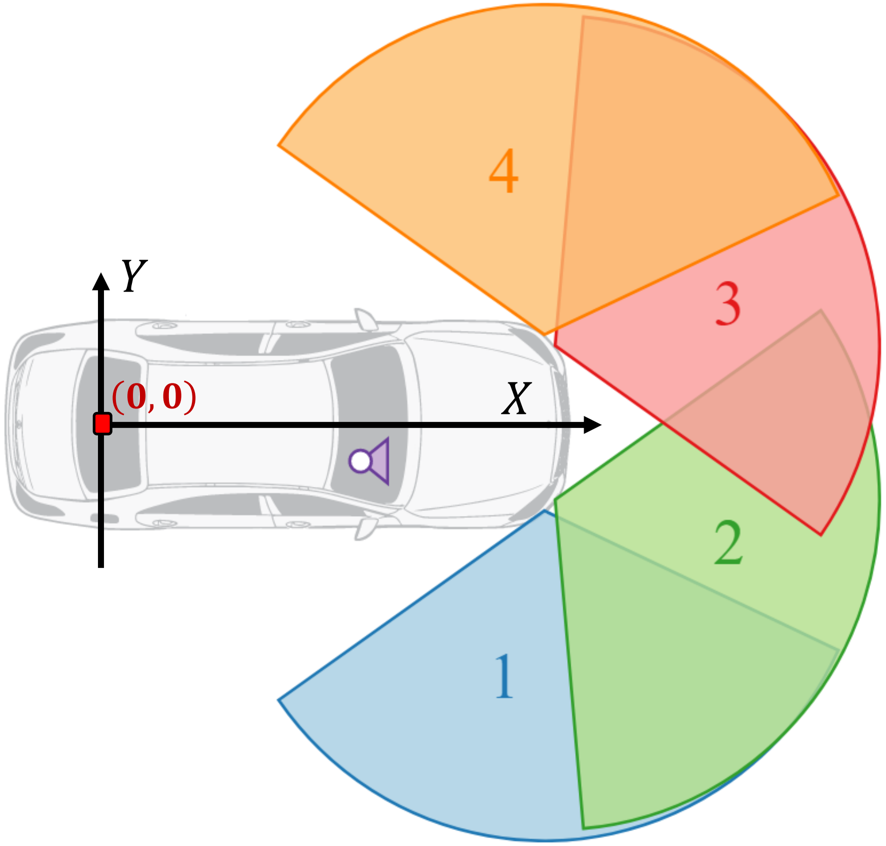
Static Environment Representation from Radar Measurements

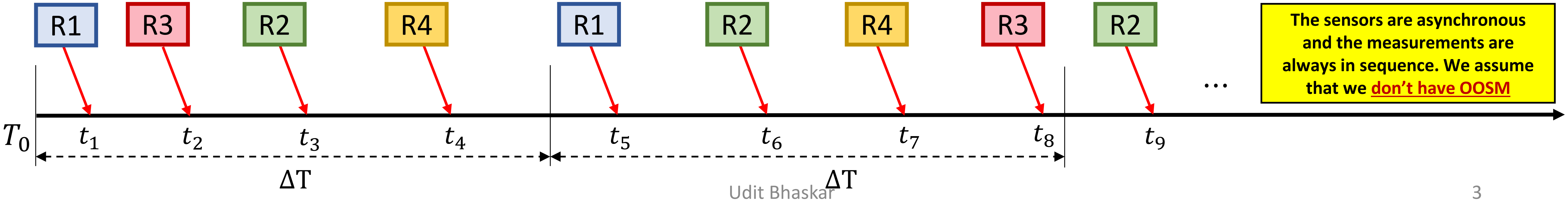
Contents

Sensor Setup

Source: *https://radar-scenes.com/dataset/sensors/*

Parameters / Sensor	Radar 1	Radar 2	Radar 3	Radar 4
Mount x coordinate	$+3.663$	$+3.86$	$+3.86$	$+3.663$
Mount y coordinate	-0.873	-0.7	$+0.7$	$+0.873$
Mount angle	$-85°$	$-25°$	$+25°$	$+85°$
Range resolution	0.15 meters			
Azimuth resolution	At the boresight direction, the resolution is about 0.5° and degrades to 2° at the outer parts of the field of view			
Range rate resolution	0.1 km/hr			
Maximum range	100 meters			
Maximum azimuth	$\pm 60°$			
Approximate measurement cycle	60 millisecond (approx. 17 Hz)			





The sensors are asynchronous and the measurements are always in sequence. We assume that we don't have OOSM

Inputs Considered

<u>Measuremets from radar *i* at time t in sensor frame</u>

$Z_t^{radar_i} = \{z_1 \quad z_2 \quad \cdots \quad z_{m_k}\}$

$z_i = [r \quad \alpha \quad v_r]^T$

$r \rightarrow$ range

$\alpha \rightarrow$ azimuth

$v_r \rightarrow$ range rate

<u>Radar *i* mount info</u>

installation coordinates $\rightarrow \left(x_m^{radar_i} \quad y_m^{radar_i}\right)$

mounting angle $\rightarrow \theta_m^{radar_i}$

<u>Ego vehicle odometry at time t</u>

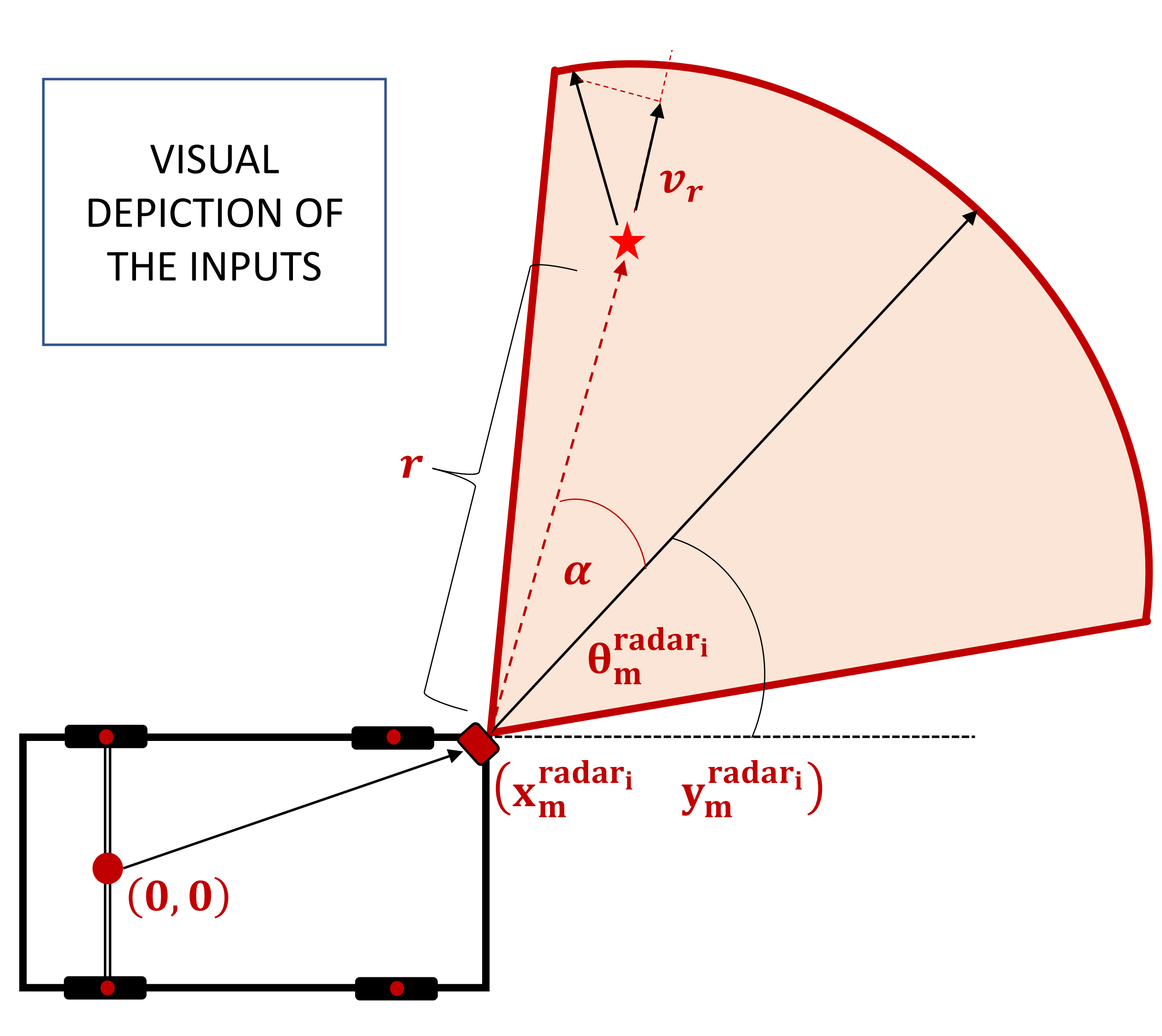
$v_t^x \rightarrow$ lateral velocity

$\dot{\omega}_t \rightarrow$ yaw rate

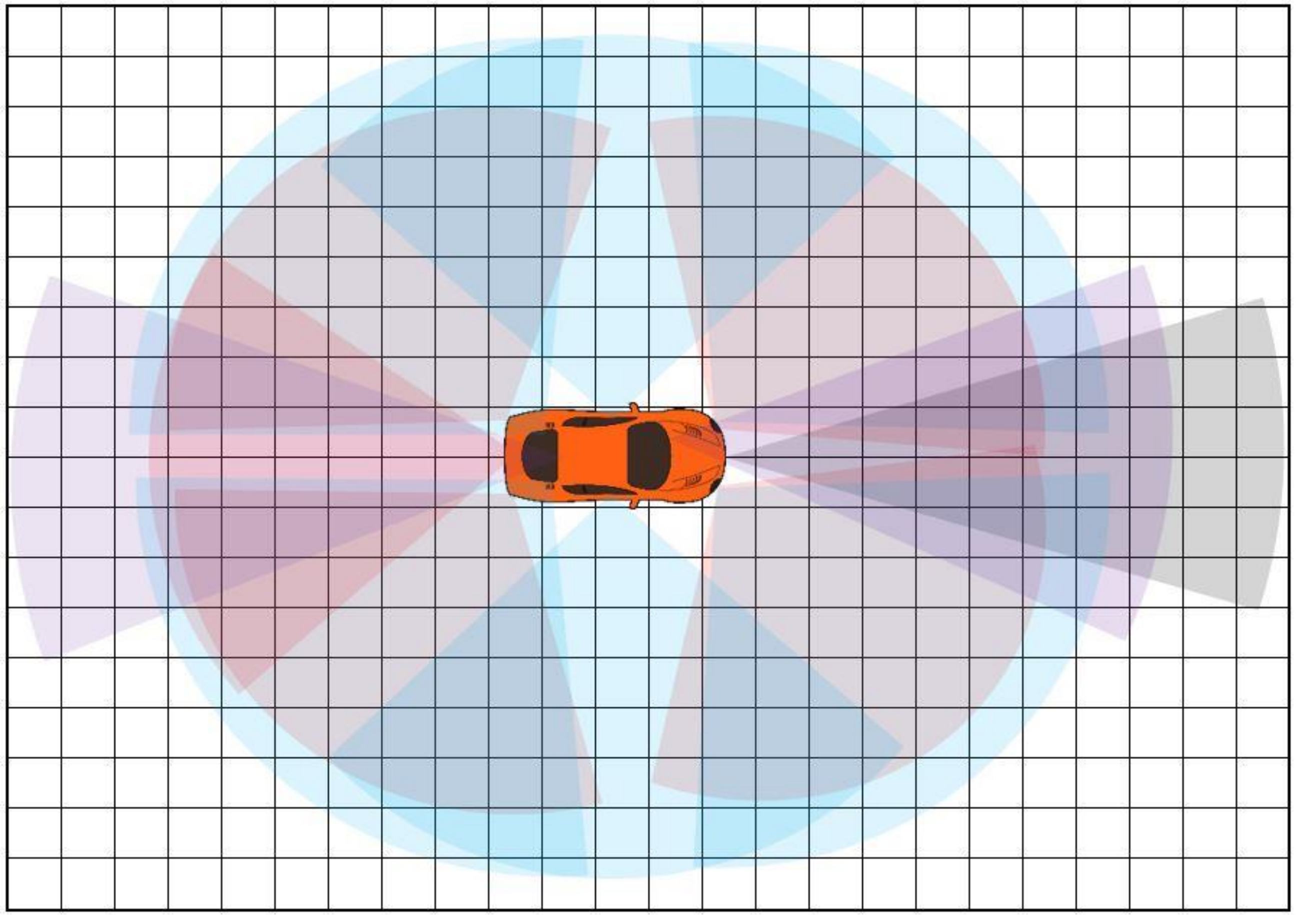
<u>Ego vehicle localization information at time t (arbitrary frame)</u>

position $\rightarrow (px_t^{loc} \quad py_t^{loc})$

orientation $\rightarrow \theta_t^{loc}$



Grid Representation



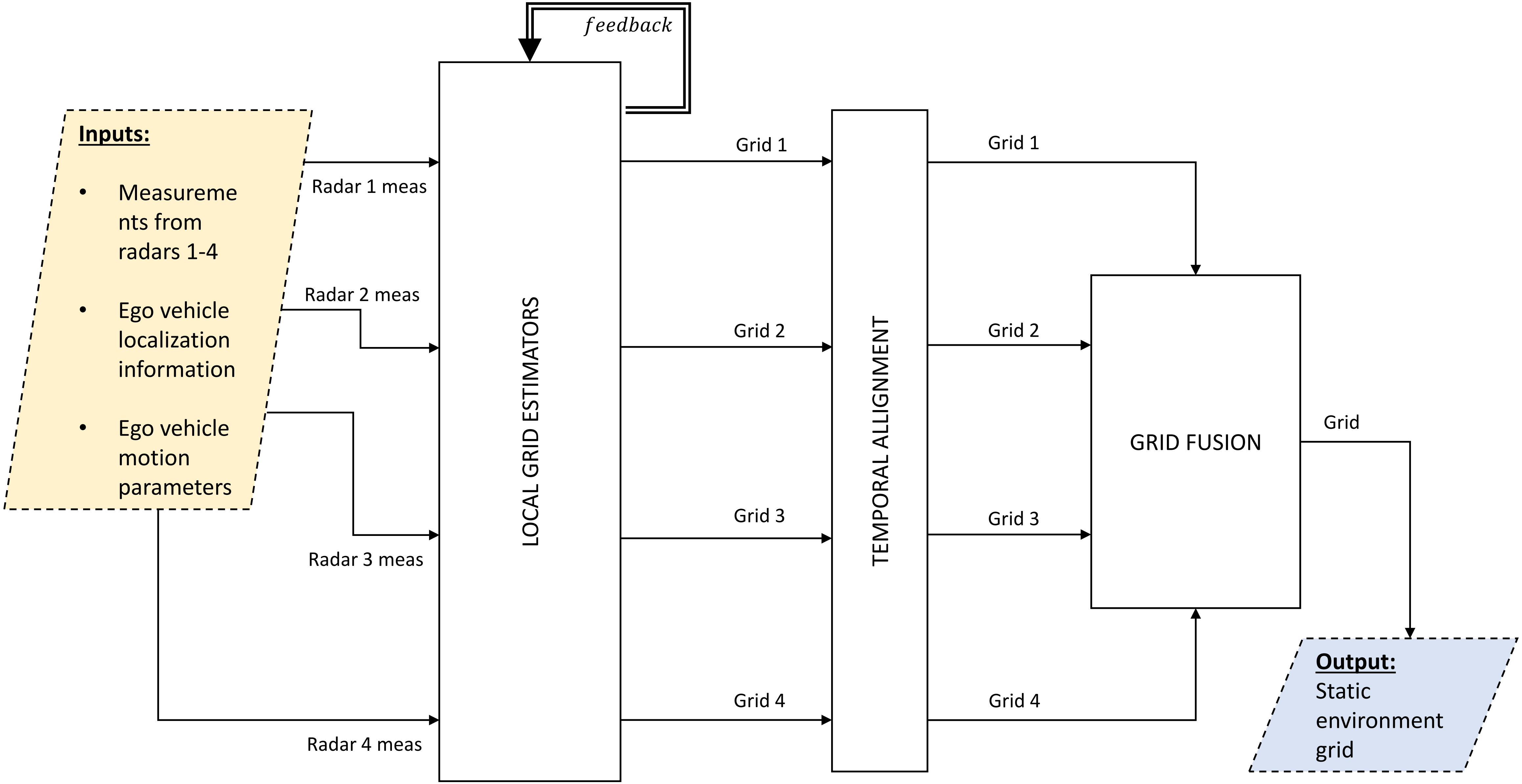
It is assumed that **only a single measurement can occupy a grid cell**. If a measurement is in the grid cell then the corresponding cell state is $(\boldsymbol{px}, \boldsymbol{py}, \boldsymbol{logodds})$

At a time a subset of grid cells are maintained which holds valid measurements/states

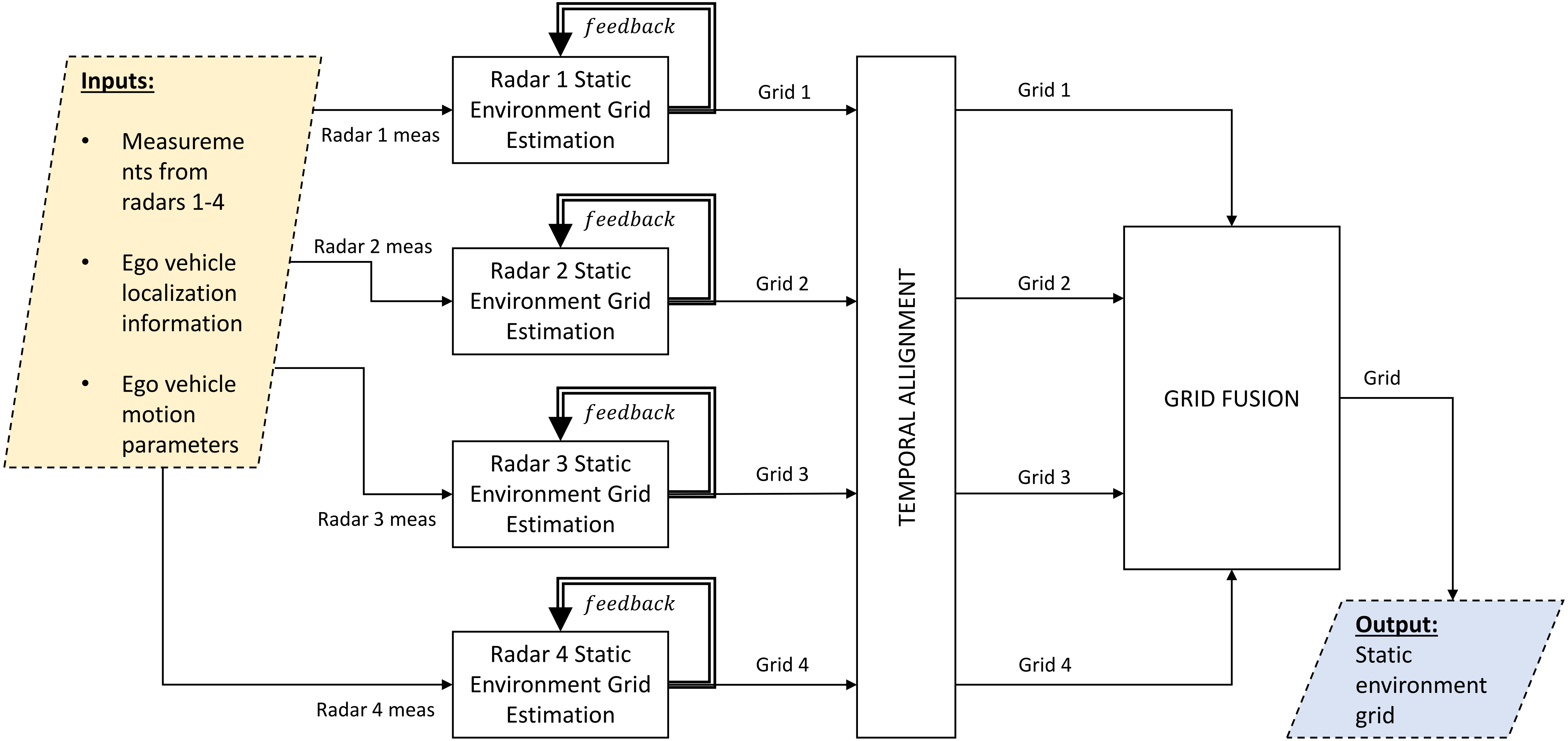
Since the grid is rectangular with uniformly sized cells. Each grid cell can be indexed like an image leading to efficient gating and state updates

Separate grids are maintained w.r.t the ego vehicle for each radars which are then combined to get a fused grid. This scheme ensures that the architecture can be made modular and more robust against sensor failures

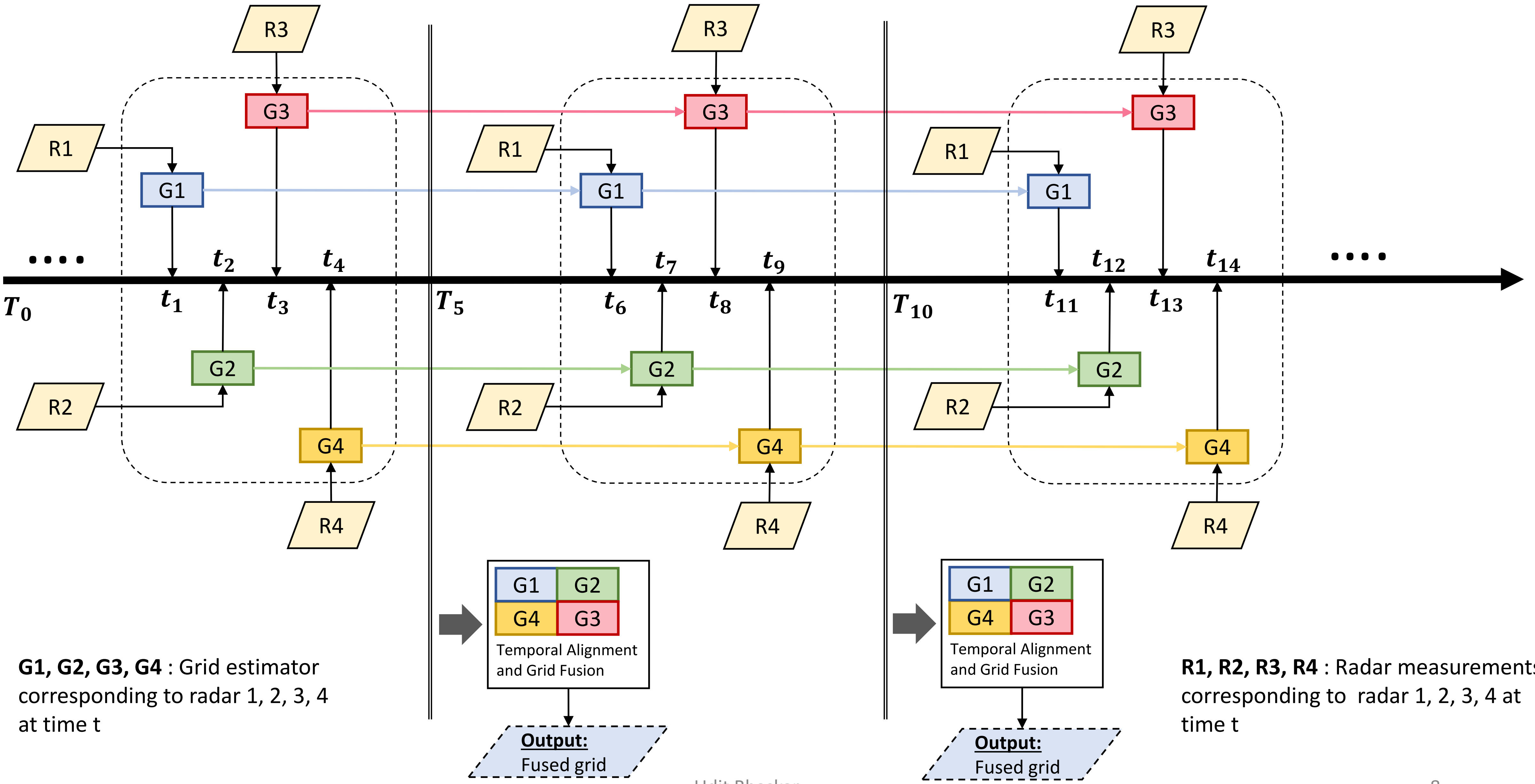
High-Level Architecture 1



High-Level Architecture 2



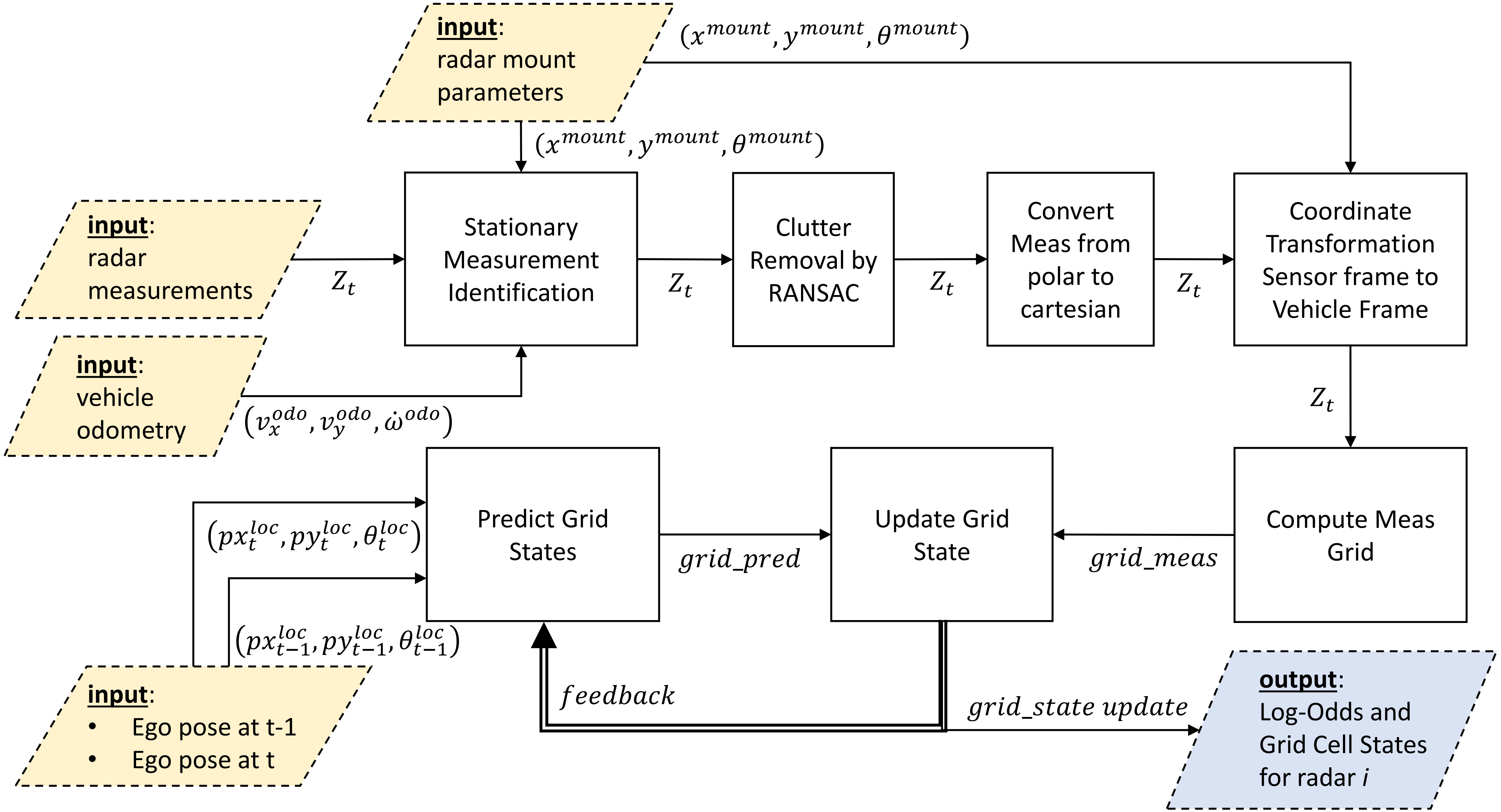
Sequence Diagram



G1, G2, G3, G4 : Grid estimator corresponding to radar 1, 2, 3, 4 at time t

R1, R2, R3, R4 : Radar measurements corresponding to radar 1, 2, 3, 4 at time t

High-Level Architecture : Grid State Estimation for each Radar



Stationary Measurement Identification

input: Radar mount parameters

$\left(r_x^{mount}, r_y^{mount}, \theta^{mount}\right)$

input: Predicted ego motion

$\left(v_x^{ego\_pred}, v_y^{ego\_pred}, \dot{\omega}^{ego\_pred}\right)$

compute ego motion at the radar location in the radar frame

$$\mathrm{R} = \begin{bmatrix} \cos(\theta^{\mathrm{mount}}) & -\sin(\theta^{\mathrm{mount}}) \\ \sin(\theta^{\mathrm{mount}}) & \cos(\theta^{\mathrm{mount}}) \end{bmatrix}$$

$$\begin{bmatrix} \mathrm{v}_{\mathrm{x}}^{\mathrm{rad\_pred}} \\ \mathrm{v}_{\mathrm{y}}^{\mathrm{rad\_pred}} \end{bmatrix} = \mathrm{R}^{-1} \begin{bmatrix} 1 & 0 & -\mathrm{r}_{\mathrm{y}}^{\mathrm{mount}} \\ 0 & 1 & \mathrm{r}_{\mathrm{x}}^{\mathrm{mount}} \end{bmatrix} \begin{bmatrix} \mathrm{v}_{\mathrm{x}}^{\mathrm{ego\_pred}} \\ \mathrm{v}_{\mathrm{y}}^{\mathrm{ego\_pred}} \\ \dot{\omega}^{\mathrm{ego\_pred}} \end{bmatrix}$$

$\left(v_x^{rad\_pred}, v_y^{rad\_pred}\right)$

input: azimuth angles of the Radar measurements

θ^i

for each of the locations corresponding to radar measurements compute the predicted range rates

$$\mathrm{v}_{\mathrm{r\_pred}}^{\mathrm{i}} = -\left(\mathrm{v}_{\mathrm{x}}^{\mathrm{rad\_pred}} \cos\left(\theta^{\mathrm{i}}\right) + \mathrm{v}_{\mathrm{y}}^{\mathrm{rad\_pred}} \sin\left(\theta^{\mathrm{i}}\right)\right)$$

$v_{r\_pred}^i$

input: range rates of the Radar measurements

v_r^i

Compare the predicted range rate with the measurement range rate

$$\mathrm{error}^{\mathrm{i}} = \left|\mathrm{v}_{\mathrm{r\_pred}}^{\mathrm{i}} - \mathrm{v}_{\mathrm{r}}^{\mathrm{i}}\right|$$

$error^i$

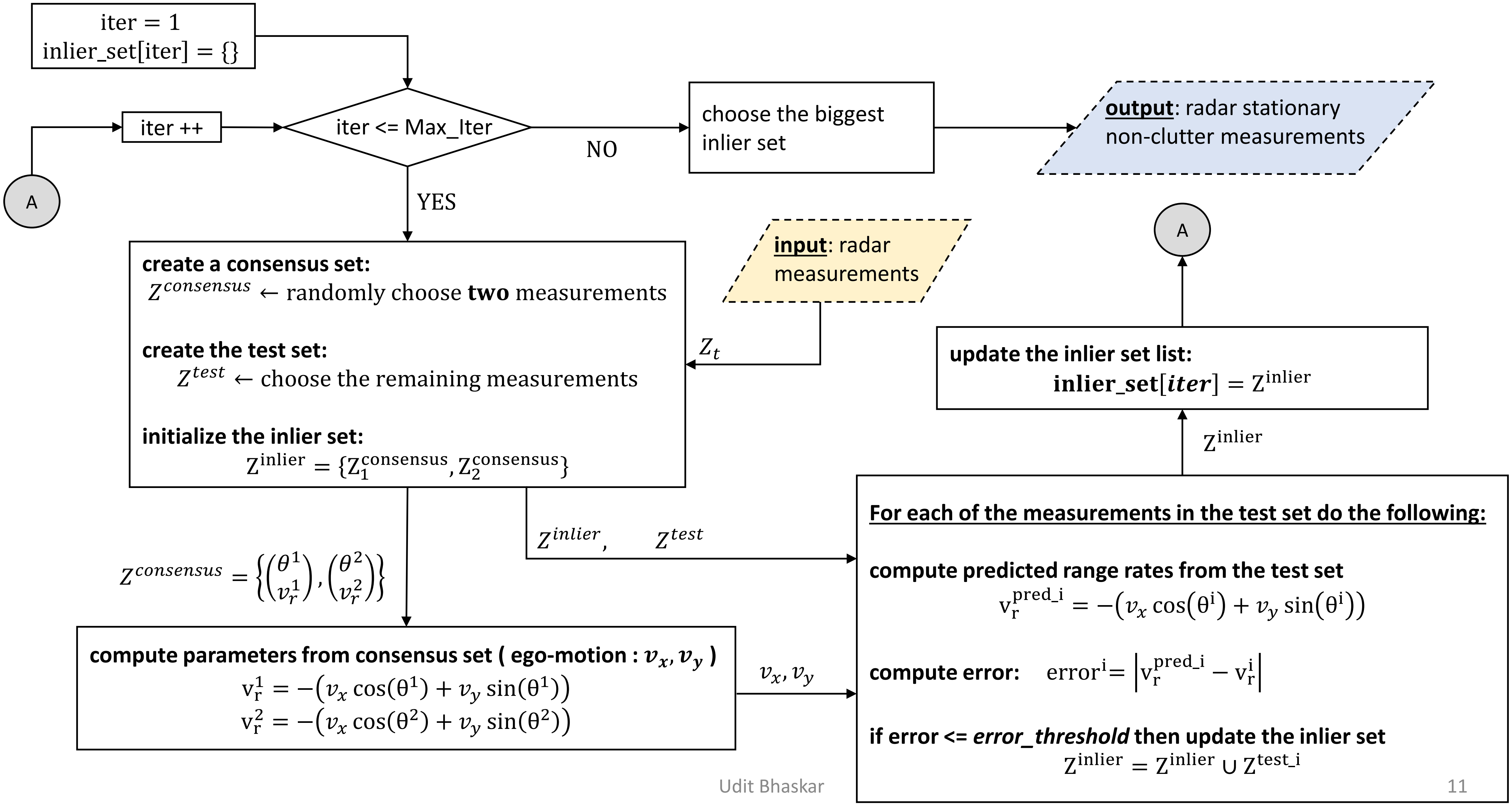
Stationary measurement selection

if $\mathrm{error}^{\mathrm{i}} \leq threshold$,
then Z^{i} is considered stationary

$Z_{stationary}^i$

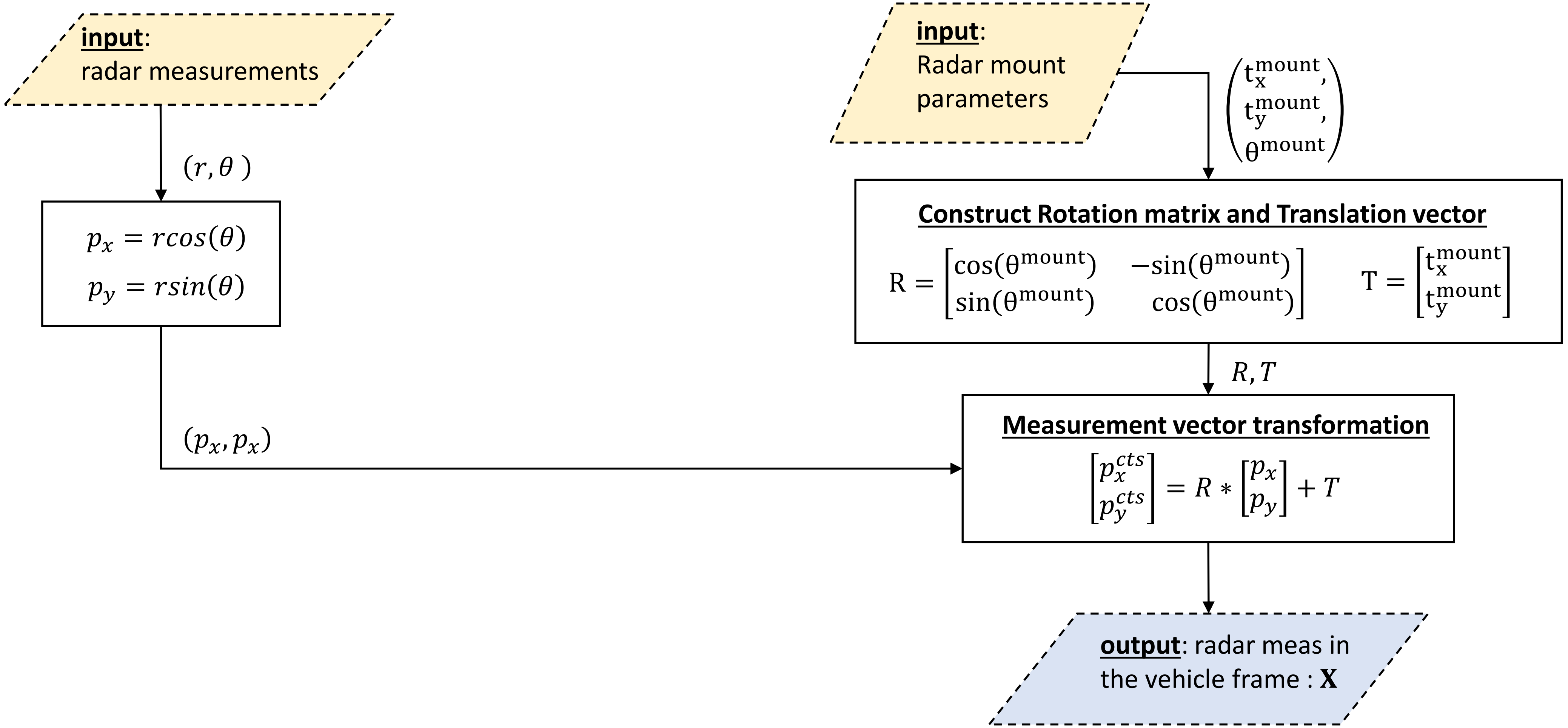
output: preliminary list of stationary measurements

Clutter Removal by RANSAC

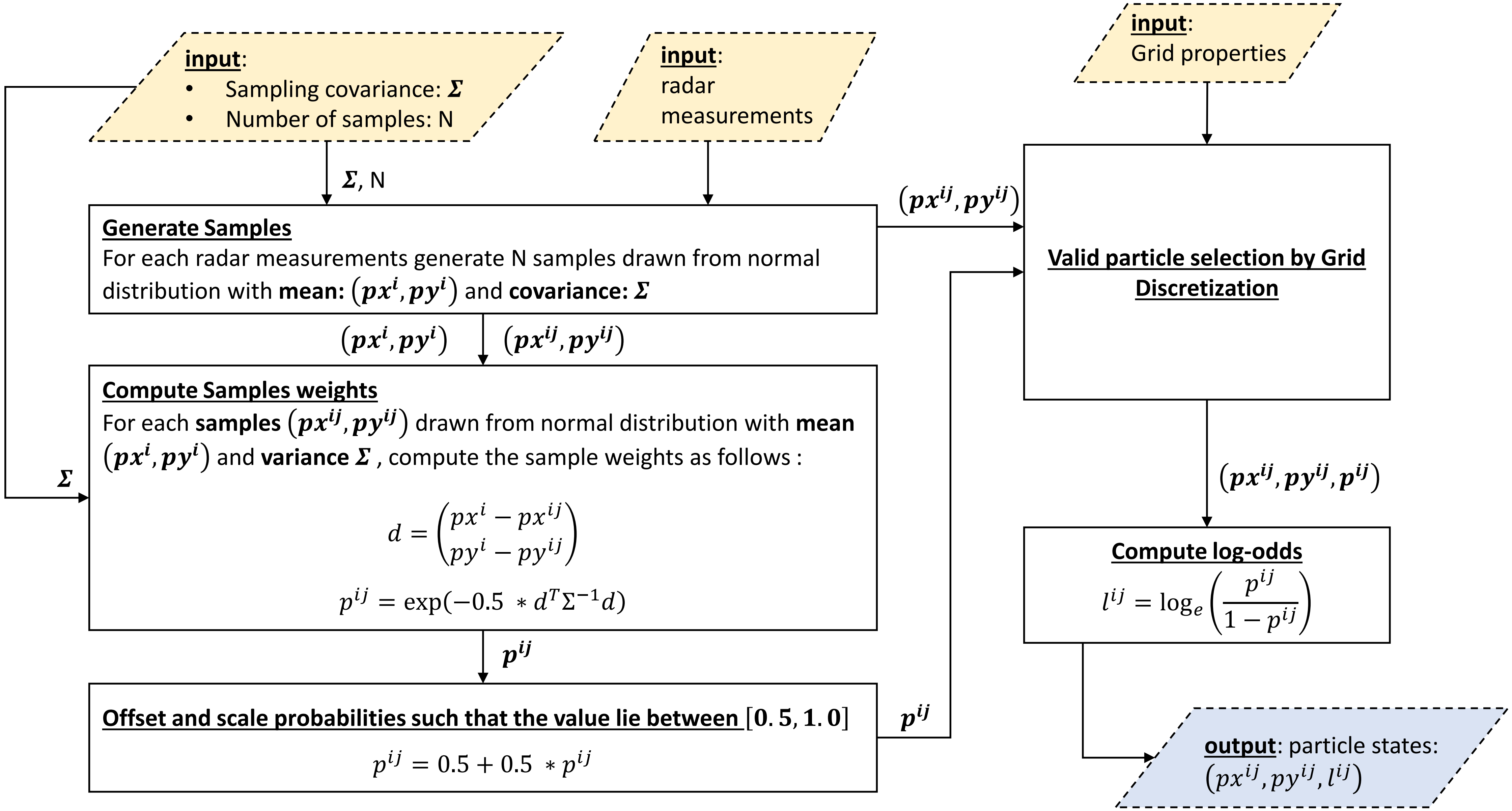


Polar to Cartesian measurement conversion

Coordinate Transformation Sensor to Vehicle Frame



Measurement Grid Computation



Grid State Prediction

input:
- Ego pose at t-1
- Ego pose at t

$\left(px_t^{loc}, py_t^{loc}, \theta_t^{loc}\right)$ $\left(px_{t-1}^{loc}, py_{t-1}^{loc}, \theta_{t-1}^{loc}\right)$

Compute pose transformation matrix that can transform vectors from vehicle frame at t-1 to vehicle frame at t

$$T_{prev} = \begin{bmatrix} \cos\left(\theta_{t-1}^{loc}\right) & -\sin\left(\theta_{t-1}^{loc}\right) & px_{t-1}^{loc} \\ \sin\left(\theta_{t-1}^{loc}\right) & \cos\left(\theta_{t-1}^{loc}\right) & py_{t-1}^{loc} \\ 0 & 0 & 1 \end{bmatrix}$$

$$T_{curr} = \begin{bmatrix} \cos\left(\theta_{t}^{loc}\right) & -\sin\left(\theta_{t}^{loc}\right) & px_{t}^{loc} \\ \sin\left(\theta_{t}^{loc}\right) & \cos\left(\theta_{t}^{loc}\right) & py_{t}^{loc} \\ 0 & 0 & 1 \end{bmatrix}$$

$$T = T_{curr}^{-1} T_{prev} = \begin{bmatrix} R_{2\times2} & t_{2\times1} \\ 0_{1\times2} & 1 \end{bmatrix}$$

$R_{2\times2}$

$t_{2\times1}$

input: particle updated states at time t-1: $\left(x_{t-1}^k, y_{t-1}^k, l_{t-1}^k\right)$

Particle state prediction

$$\begin{pmatrix} x_{pred}^k \\ x_{pred}^k \end{pmatrix} = R_{2\times2} \begin{pmatrix} x_{t-1}^k \\ x_{t-1}^k \end{pmatrix} + t_{2\times1}$$

$$l_{pred}^k = l_{t-1}^k$$

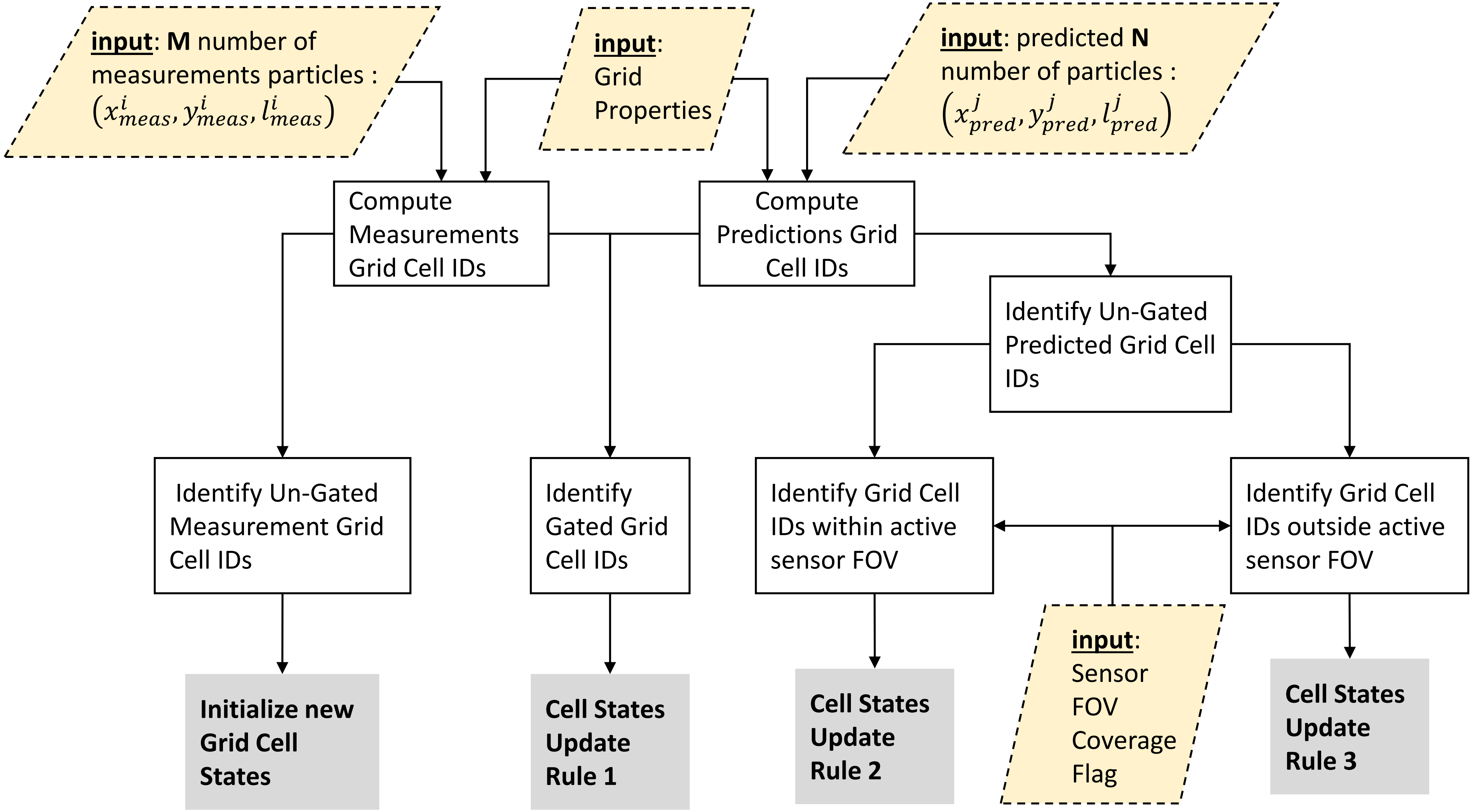
$\left(\boldsymbol{x_{pred}^k}, \boldsymbol{y_{pred}^k}, \boldsymbol{l_{pred}^k}\right)$

input:
Grid properties

Valid particle selection by Grid Discretization

output: particle predicted states: $\left(\boldsymbol{x_{pred}^k}, \boldsymbol{y_{pred}^k}, \boldsymbol{l_{pred}^k}\right)$

Grid State Update : procedure



Grid State Update : equations

Initialize new Cell States

$$x_{upd}^{i} = x_{meas}^{i}$$

$$y_{upd}^{i} = y_{meas}^{i}$$

$$l_{upd}^{i} = \alpha_0 * l_{meas}^{i}$$

Cell States Update Rule 1

$$x_{upd}^{i} = w_x * x_{meas}^{i} + (1 - w_x) * x_{pred}^{i}$$

$$y_{upd}^{i} = w_y * y_{meas}^{i} + \left(1 - w_y\right) * y_{pred}^{i}$$

$$l_{upd}^{i} = l_{meas}^{i} + \alpha_1 * l_{pred}^{i}$$

Cell States Update Rule 2

$$x_{upd}^{i} = x_{pred}^{i}$$

$$y_{upd}^{i} = y_{pred}^{i}$$

$$l_{upd}^{i} = \alpha_2 * l_{pred}^{i}$$

Cell States Update Rule 3

$$x_{upd}^{i} = x_{pred}^{i}$$

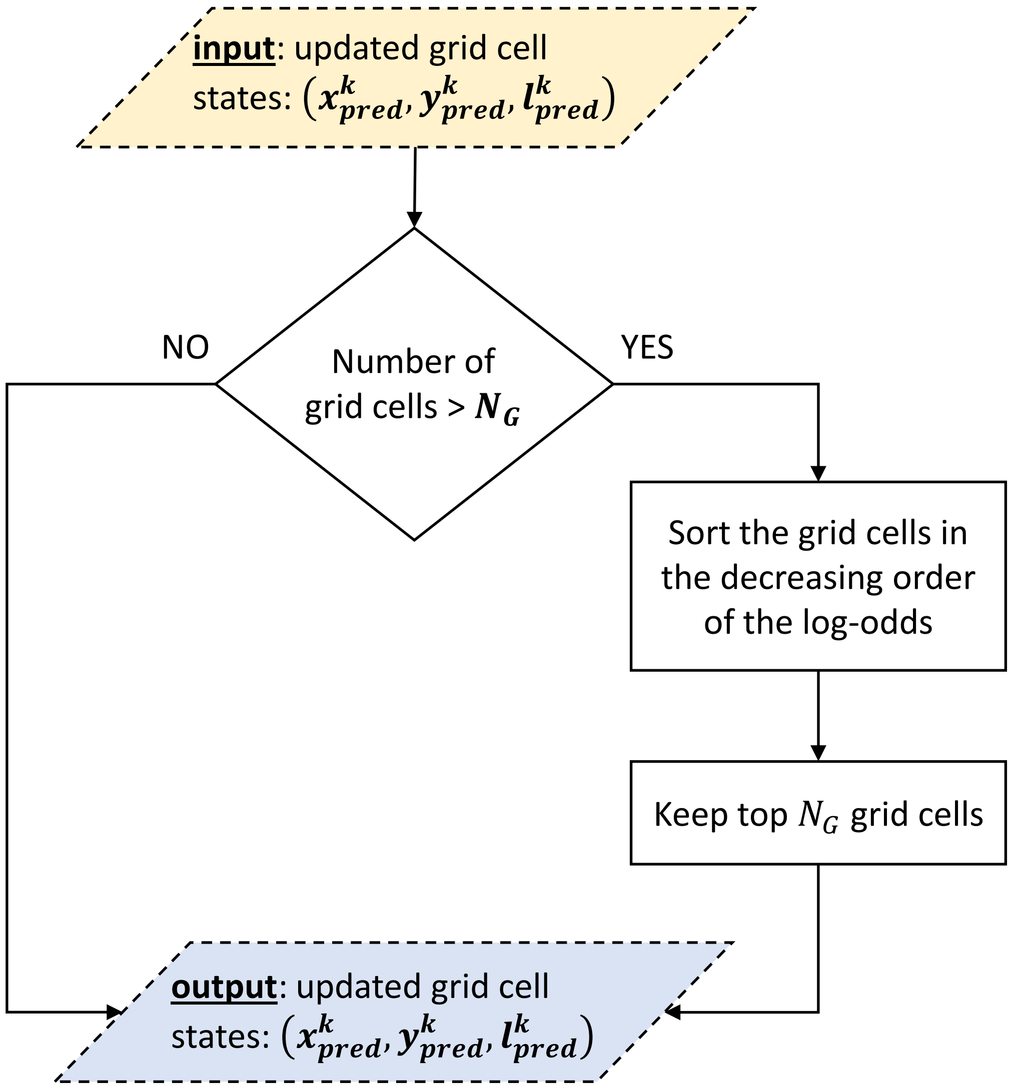
$$y_{upd}^{i} = y_{pred}^{i}$$

$$l_{upd}^{i} = \alpha_3 * l_{pred}^{i}$$

Note:

$$0 < w_x, w_y, \alpha_0, \alpha_1, \alpha_2, \alpha_3 \leq 1$$

Grid State Update : prune grid cells



Grid Fusion

$$x_{fus}^{i} = w_{sensor1} * x_{sensor1}^{i} + w_{sensor2} * x_{sensor2}^{i} + w_{sensor3} * x_{sensor3}^{i} + w_{sensor4} * x_{sensor4}^{i}$$

$$y_{fus}^{i} = w_{sensor1} * y_{sensor1}^{i} + w_{sensor2} * y_{sensor2}^{i} + w_{sensor3} * y_{sensor3}^{i} + w_{sensor4} * y_{sensor4}^{i}$$

$$l_{fus}^{i} = l_{sensor1}^{i} + l_{sensor2}^{i} + l_{sensor3}^{i} + l_{sensor4}^{i}$$

Note:

$$w_{sensor1} + w_{sensor2} + w_{sensor3} + w_{sensor4} = 1$$

Use-cases

- Occupancy grid mapping
- Road profile computation
- Input Image to a deep Neural Network for various perception tasks
- Radar only perception for AD/ADAS

etc …

The End