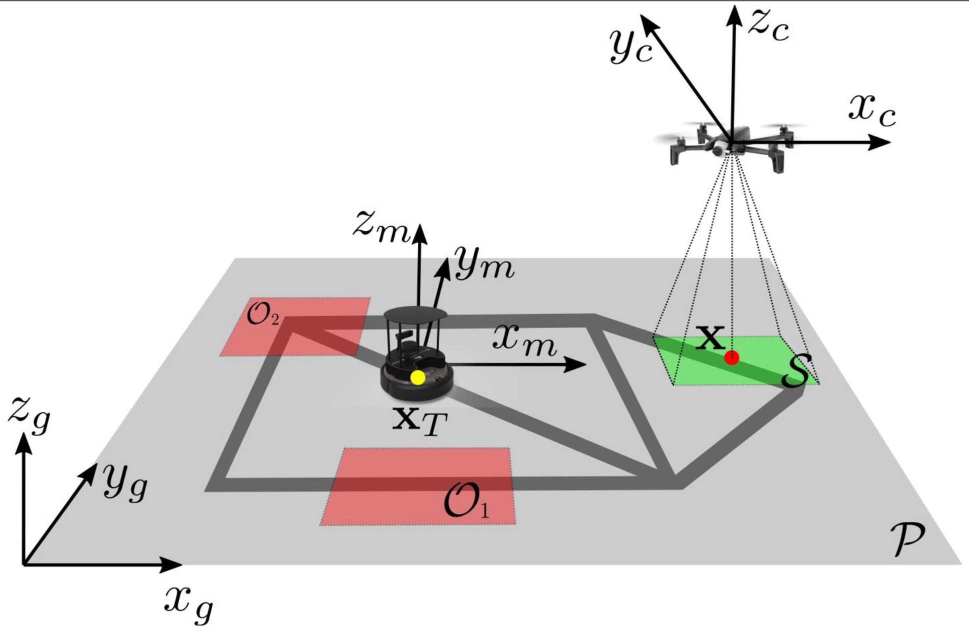
Uncertainty-Aware Guidance for Multi-Agent Target Tracking subject to Intermittent Measurements using Motion Model Learning

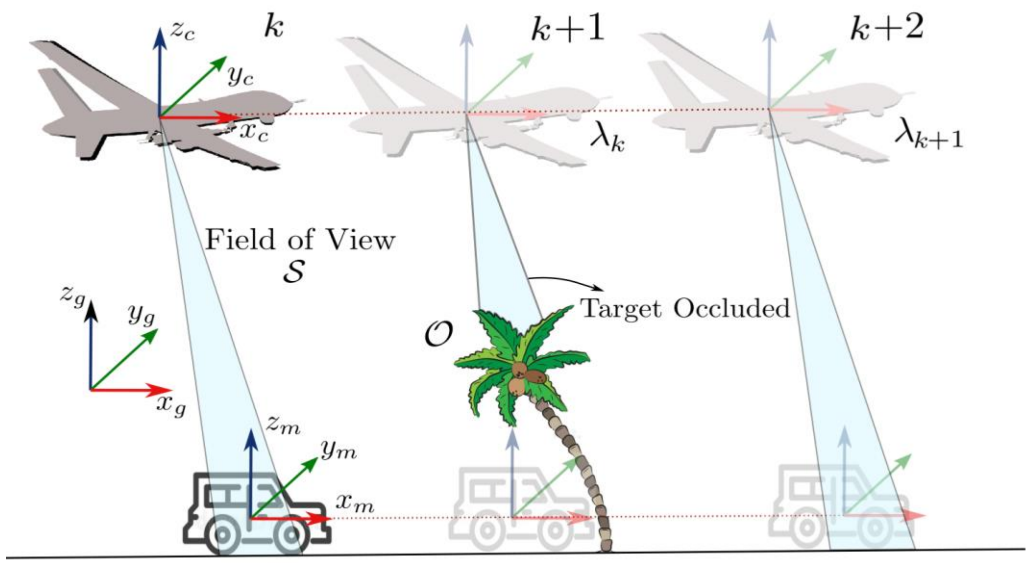
Andres Pulido (UF), Dr. Jane Shin (UF), Dr. Kyle Volle (Torch), Prashant Ganesh(EpiSci), **Dr. Zachary Bell (AFRL)**

August 30, 2024

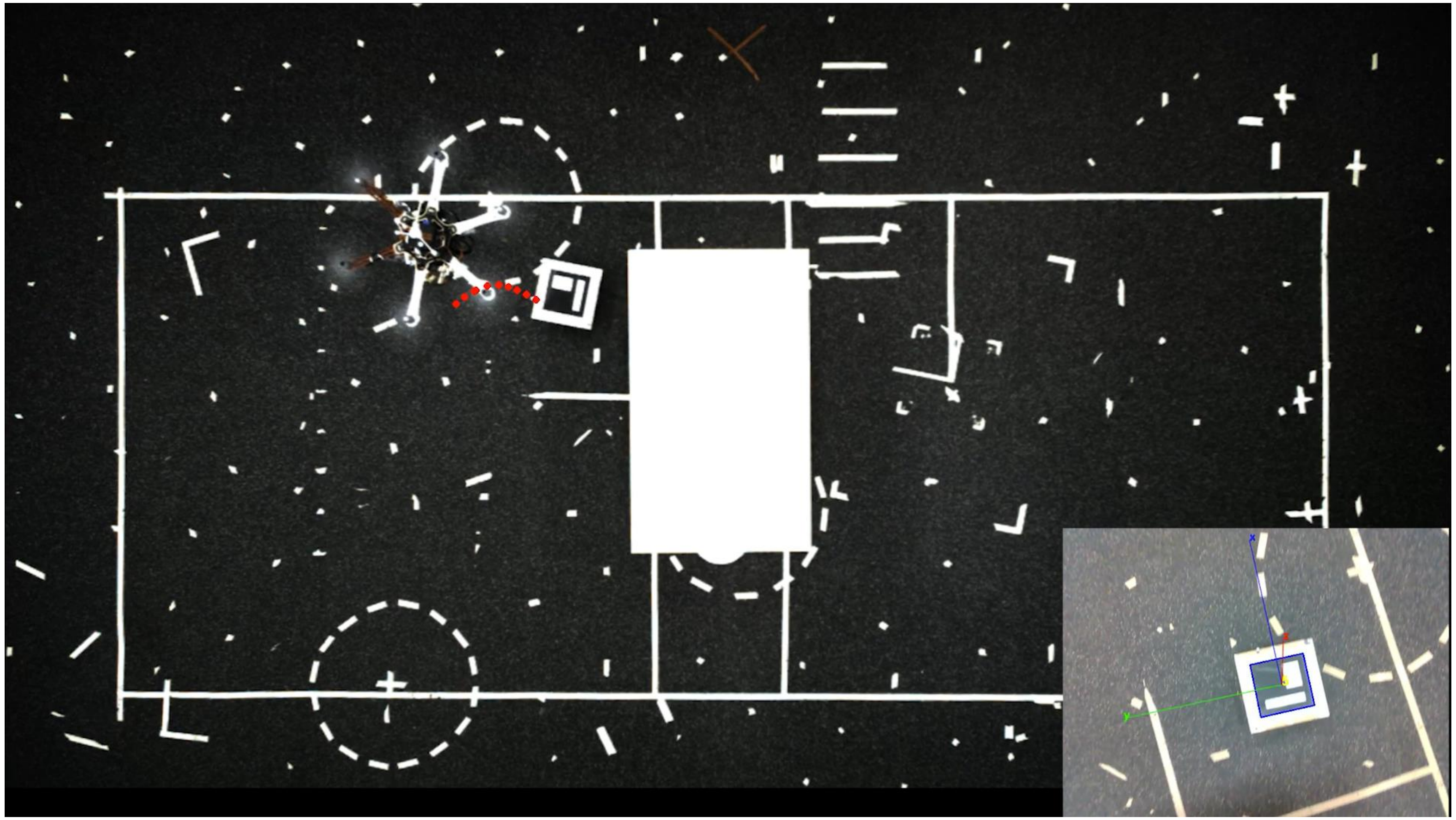
Problem Description

- **General Objective: Maintain an estimate of a specified target's position and velocity, develop motion model to predict target future position, quantify the uncertainty of target estimates.**
- Assumptions:
  - Agent's own position is always known
  - Relative target position measured accurately when target is in agent's field of view (FOV)
  - Occlusions cause intermittent measurements
  - Target follows a stochastic road network path
  - Agent has position history of previous targets on road network
- **Guidance Objective: Determine how to command the agent's next position based on predicted future information gain.**





Prior Work

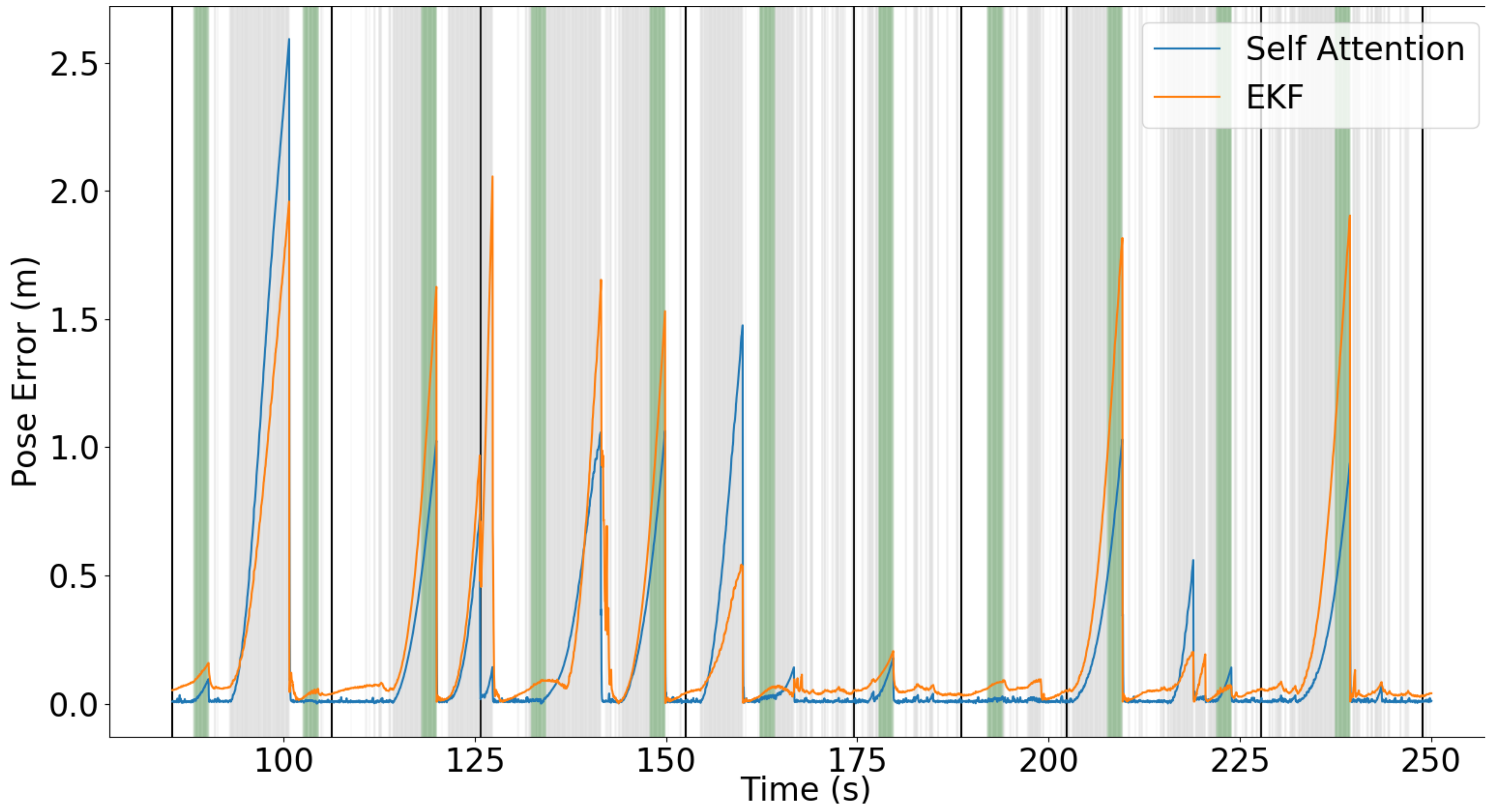


Z. I. Bell, R. Sun, K. Volle, P. Ganesh, S. A. Nivison, W. E. Dixon, “Target Tracking Subject to Intermittent Measurements using Attention Deep Neural Networks," *IEEE Control Systems Letters*, Vol. 7, pp. 379-384 (2023).

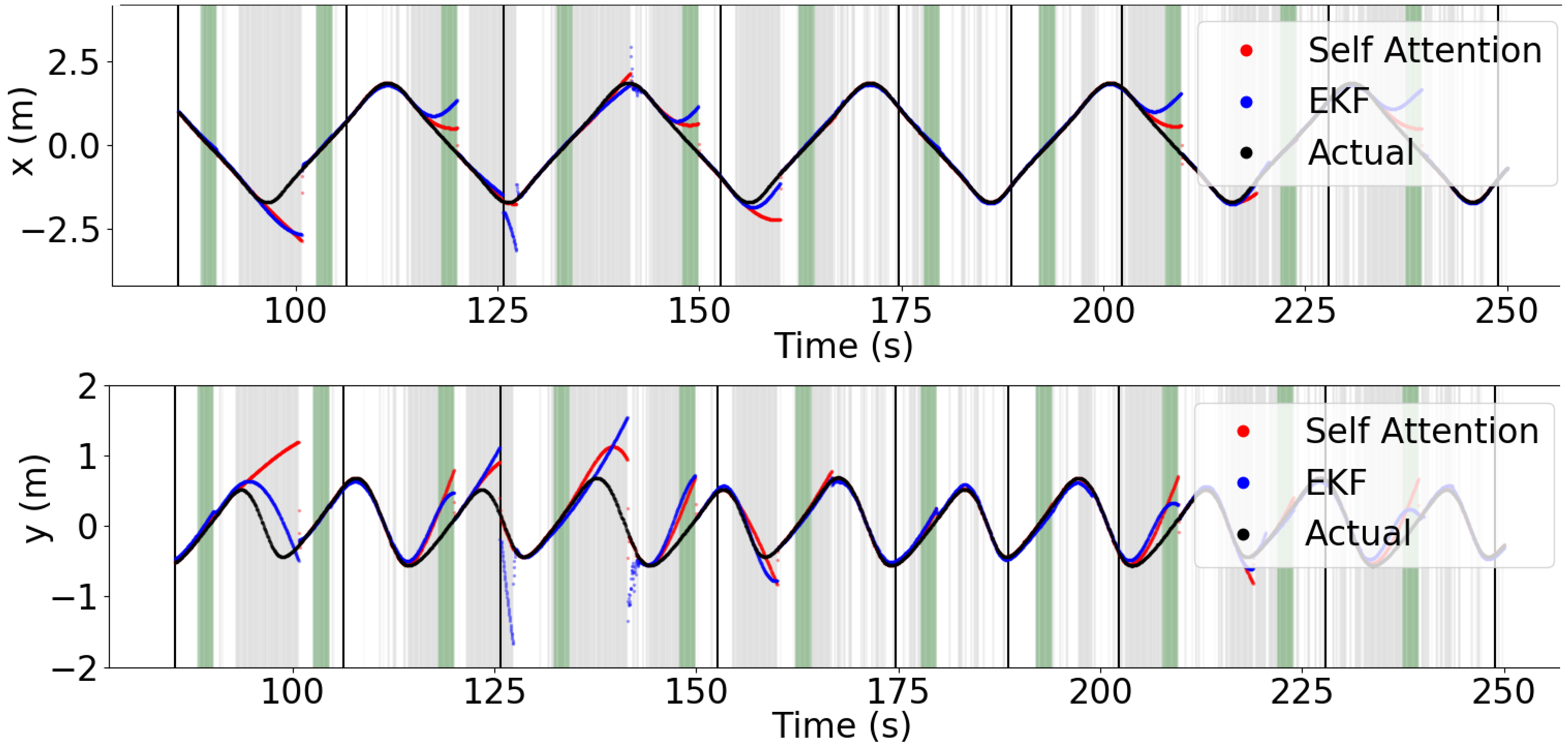
USSF

AFRL

Prior Work – Pose Error



Prior Work – Pose Error (by axis)



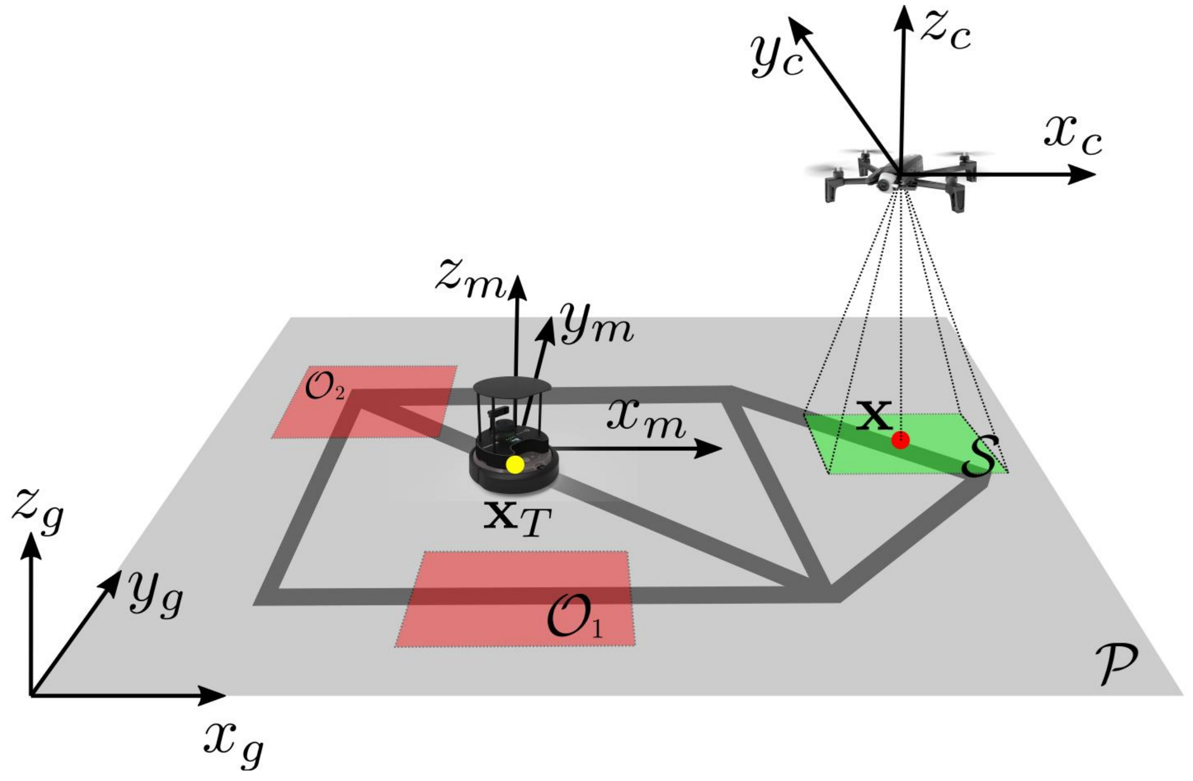
Project Overview

Given:

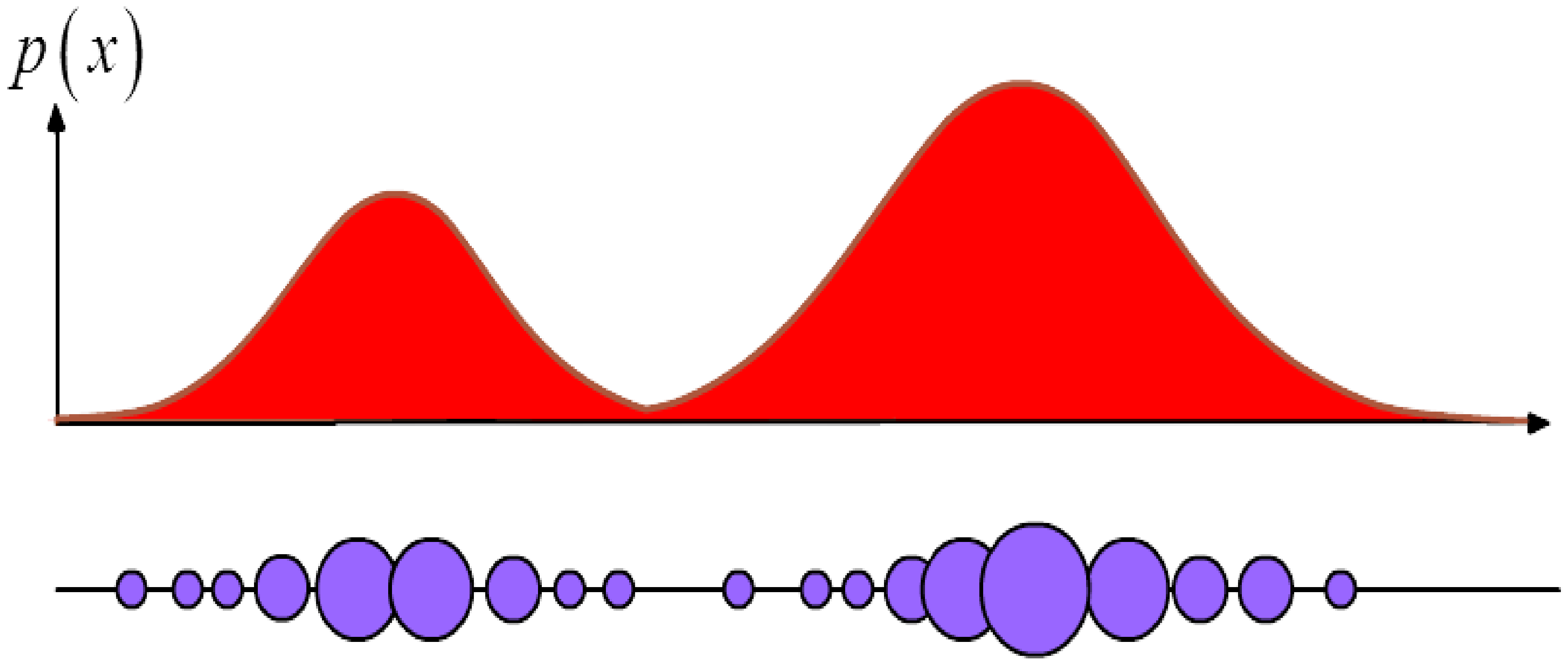
- Target following an **unknown** (to the agent) **stochastic road network path** that we treat as some general nonlinear function of the state f(x)
- Occlusions that cause **intermittent and uncertain measurements**
- **Only measure the position** of the target

We want to answer:

- How to **command the next agent position** to maximize the future information of target?
- Particle Filter for target state estimation
- Transformer-based Neural Network (NN) for motion prediction
- Expected Entropy Reduction (EER) estimation for guidance



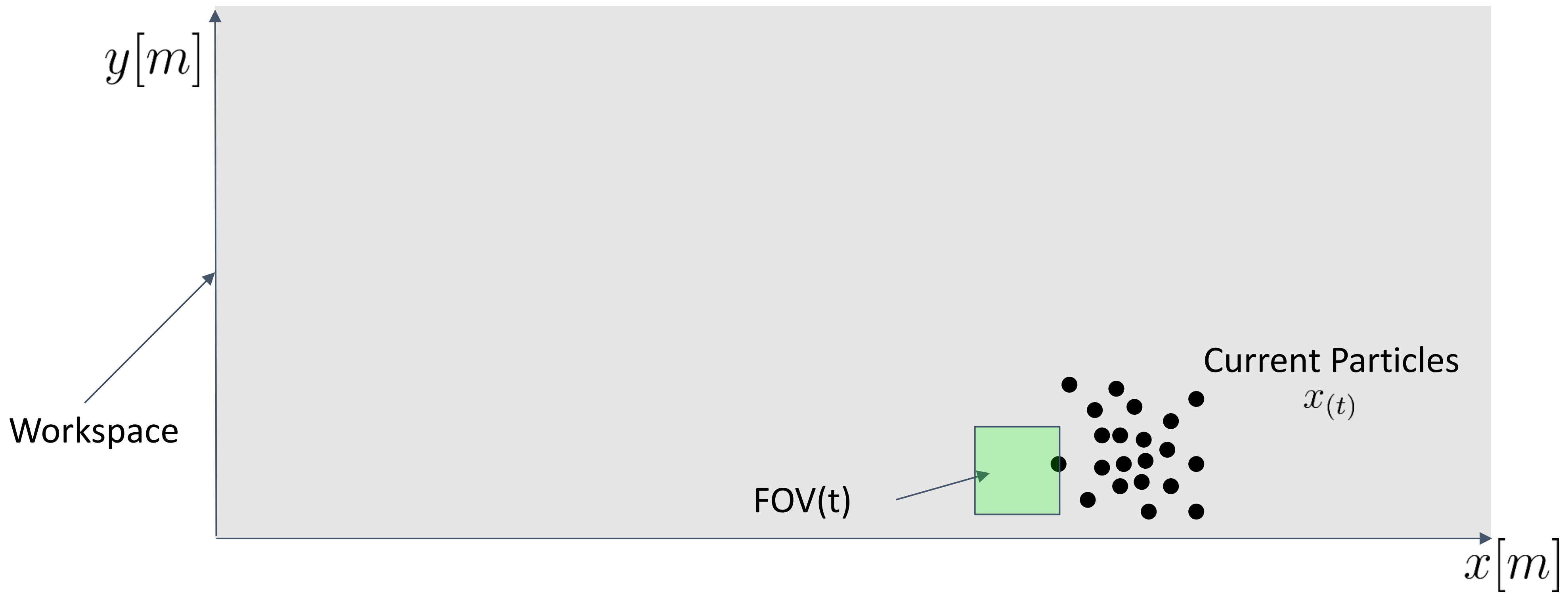
Particle Filter Concept



We approximate the belief distribution of the target position using a set of particles (guesses) with an associated “weight” for each particle.

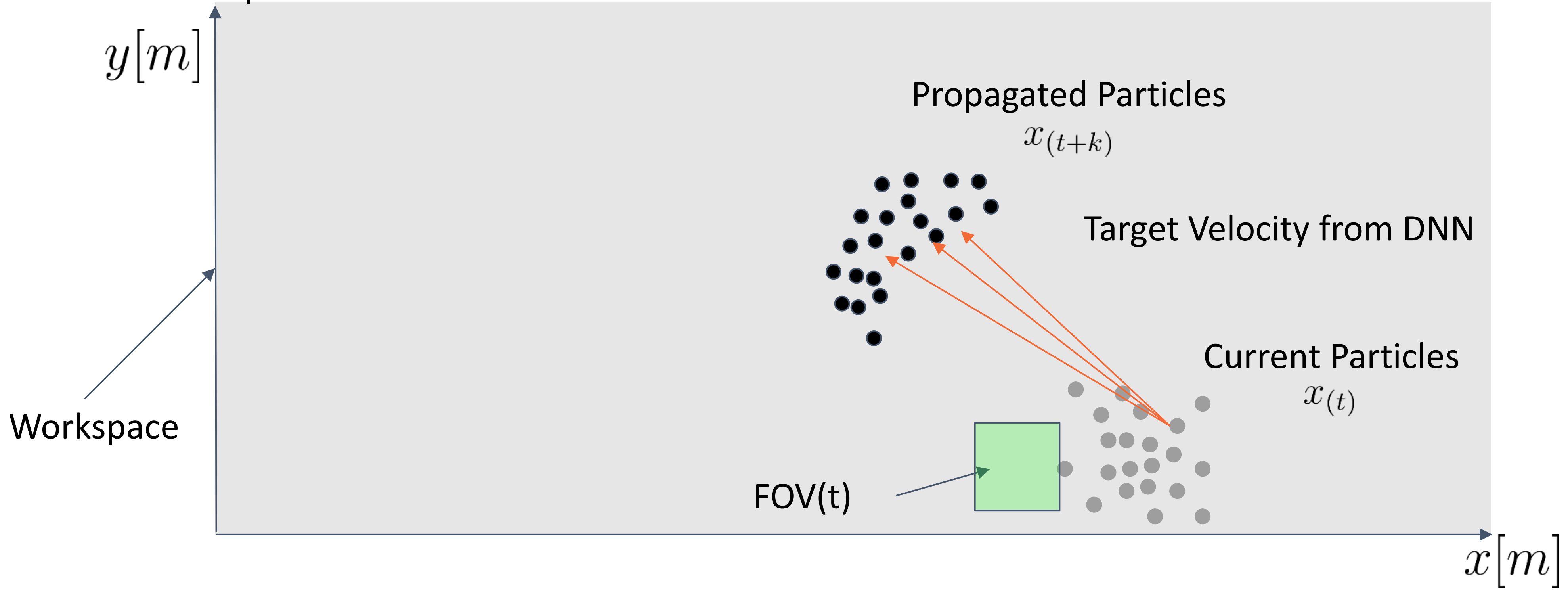
System Overview

Approximate the state belief distribution of the target state: only 2D position $x_i = \begin{bmatrix} x & y \end{bmatrix}^T$



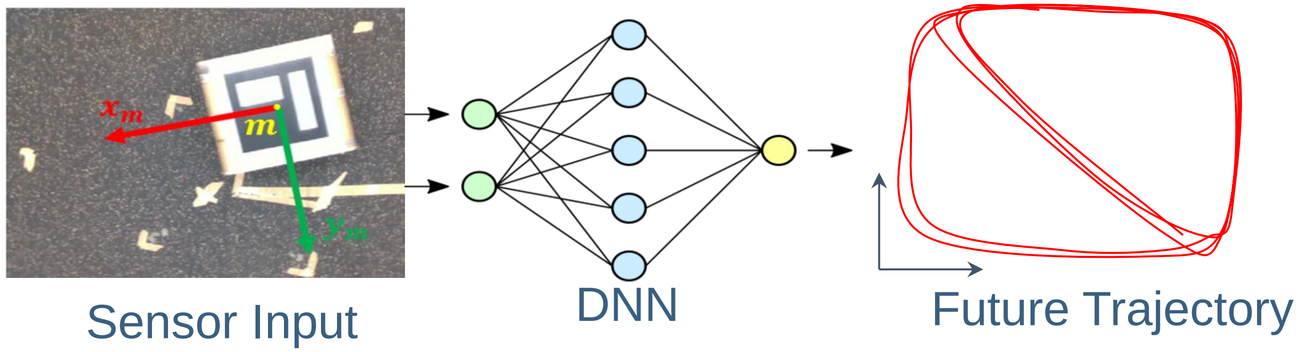
Propagation of Particles with DNN Motion Model

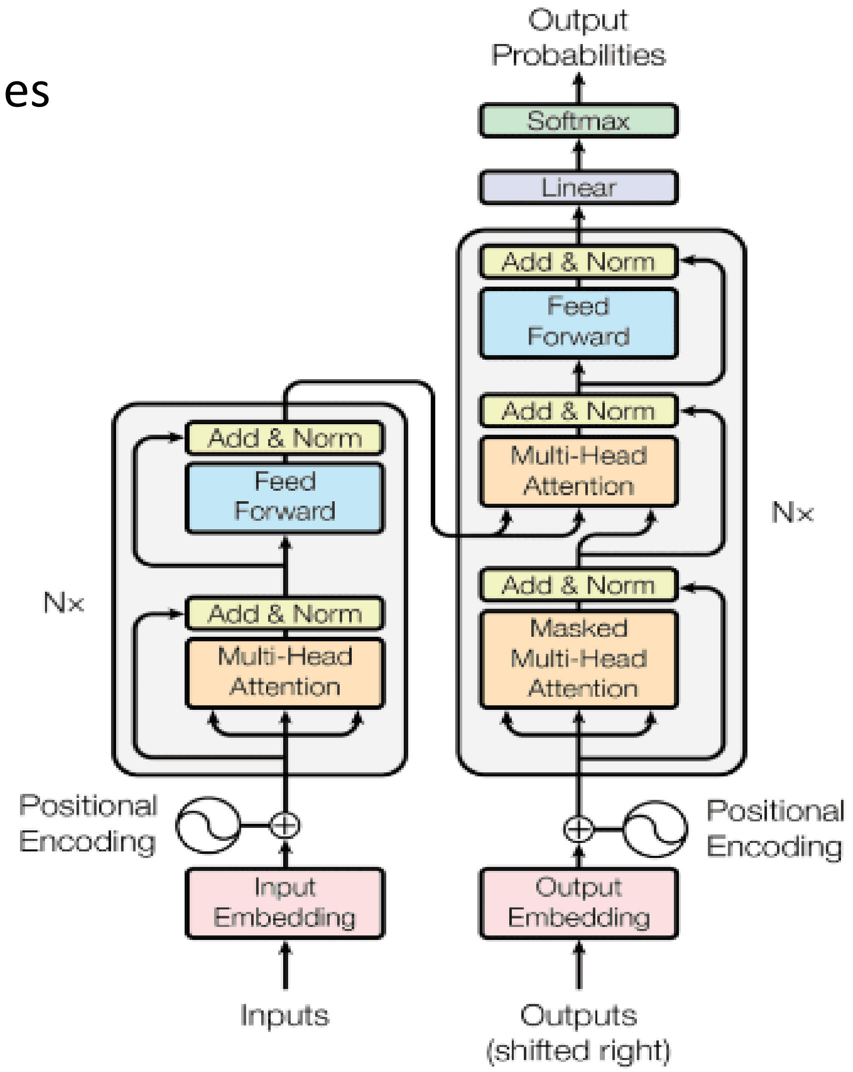
Propagate the current distribution based on the state history with DNN motion model
k time-steps into the future



DNN Motion Model Learning

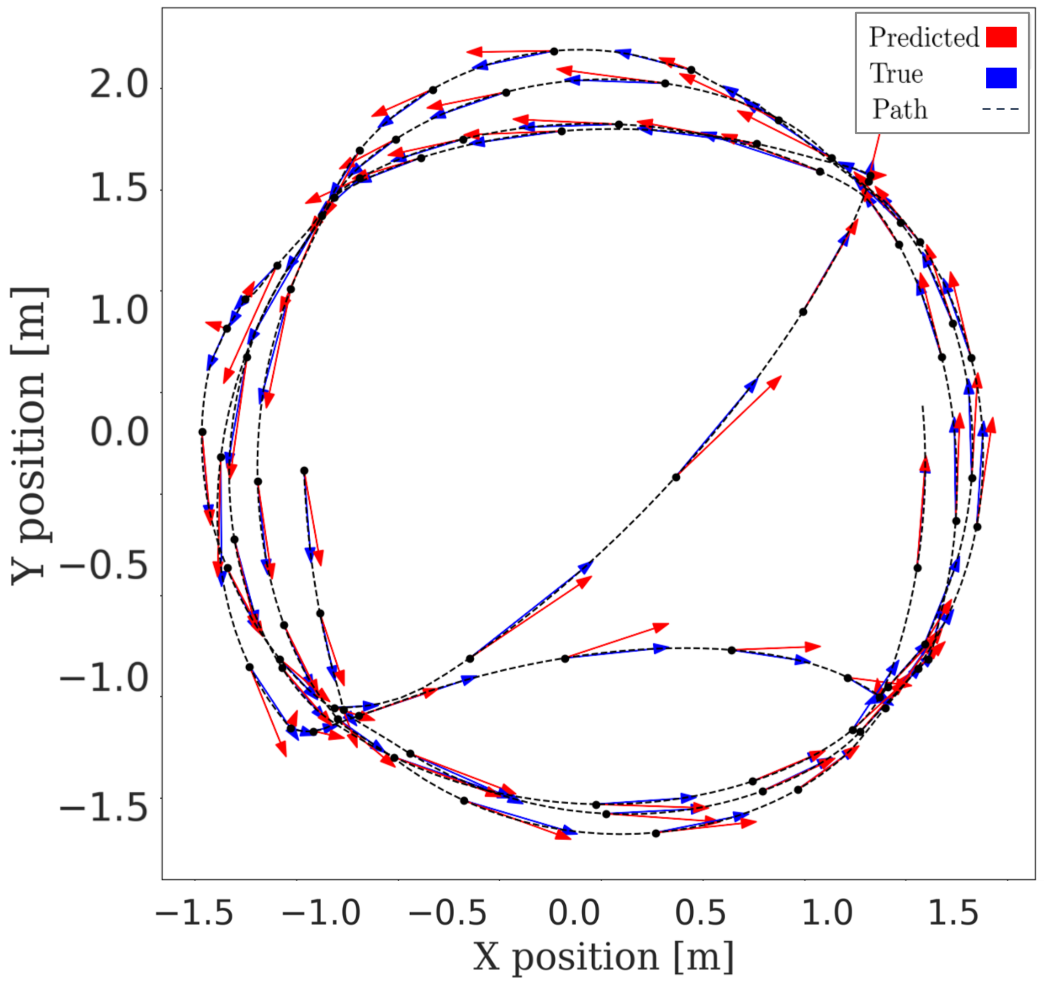
- Transformer-based NN trained in advance on position histories
- Prior work looked at online training, neglected here
- NN takes as input the immediate position history
- NN recursively outputs future positions
- Used for particle motion prediction





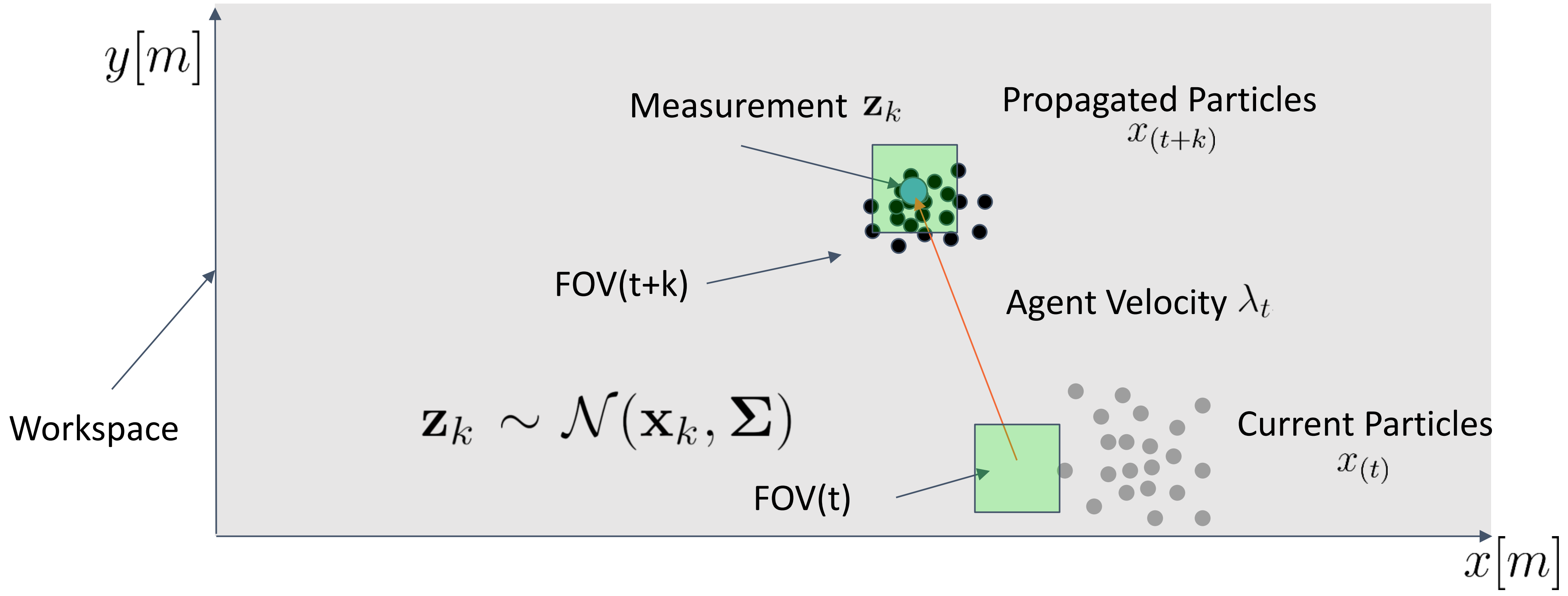
Transformer DNN Motion Model Results

- NN is fed 1 second of position history and projects 1 second forward
- Trained on an hour of data from observation



Particle Filter Measurement Update and Guidance

Update (and resample) when a measurement appears in the sensor field of view (FOV) or update particles inside FOV when there are no measurements (negative measurements). How do we incorporate the information of the updated distribution into the agent's guidance law to determine where future FOV should be? We look at maximizing the **expected entropy reduction**.


$y[m]$
Measurement $\mathbf{z}_k$
Propagated Particles $x_{(t+k)}$
FOV(t+k)
Agent Velocity λ_t
Workspace
$\mathbf{z}_k \sim \mathcal{N}(\mathbf{x}_k, \mathbf{\Sigma})$
Current Particles $x_{(t)}$
FOV(t)
$x[m]$

Entropy

Average level of "information", "surprise", or "uncertainty" inherent to the variable's possible outcomes.

For discrete random variables:

$$H(X) = \mathbb{E}_x[-\log p(x)] = -\sum_{x\in} p(x)\log p(x)$$

For Continuous random variables:

$$H(p(x_t|z_{1:t},\lambda_{1:t})) = \mathbb{E}_{x_t}[-\log p(x_t|z_{1:t},\lambda_{1:t})]$$

$$= -\int p(x_t|z_{1:t},\lambda_{1:t})\log p(x_t|z_{1:t},\lambda_{1:t})dx_t$$

Measurements

Agent Action

Particle-based

$$H(p(x_t|z_{1:t},\lambda_{1:t})) \approx \log(\sum_{i=1}^{N} p(z_t|x_t^{(i)})w_{t-1}^{(i)}) -$$

$$\sum_{i=1}^{N}\log(p(z_t|x_t^{(i)})(\sum_{j=1}^{N} p(x_t^{(i)}|x_{t-1}^{j})w_{t-1}^{j}))w_t^{(i)}$$

Weights

Y. Boers, H. Driessen, A. Bagchi, and P. Mandal, “Particle filter based entropy,” in 2010 13th International Conference on Information Fusion, July 2010, pp. 1–8

Information Gain and Expected Entropy Reduction

Reduction of entropy between two different distributions at next step

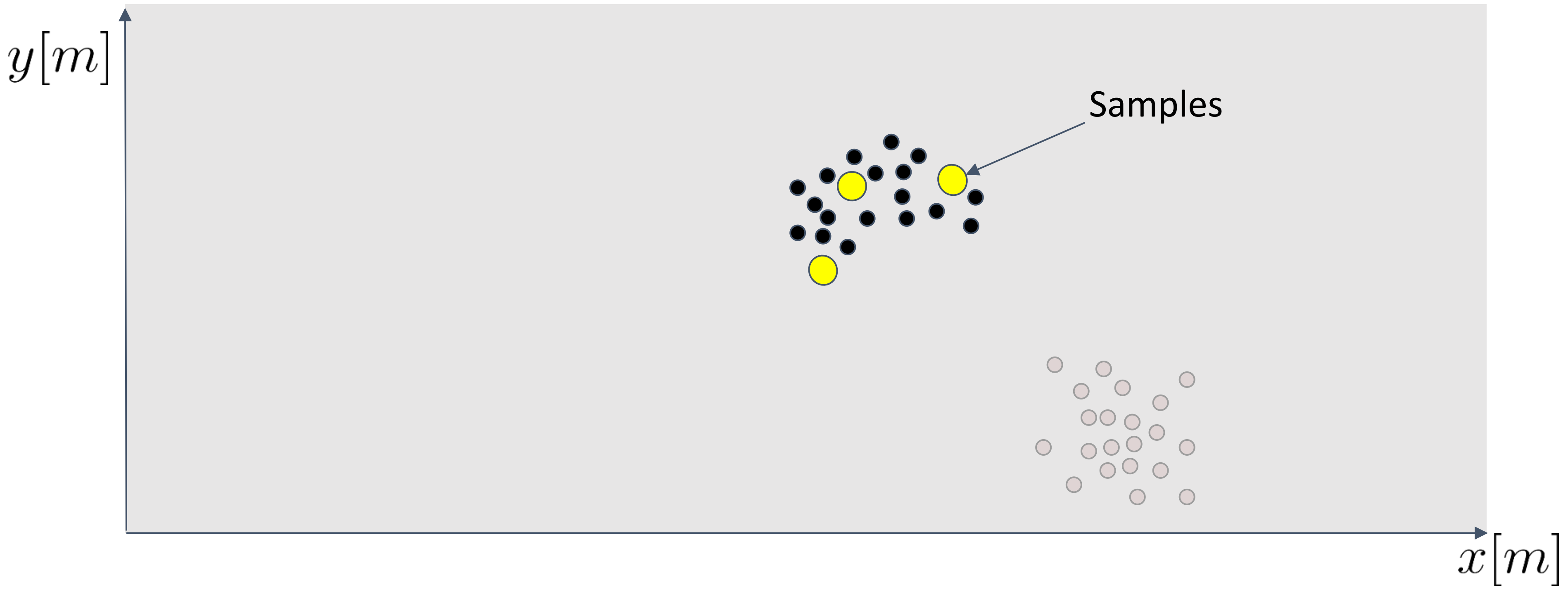
$$I(\hat{z}_{t+1}, \lambda_{t+1}) = H(p(x_t|z_{1:t}, \lambda_{1:t})) - H(p(\hat{x}_{t+1}|z_{1:t}, \hat{z}_{t+1}, \lambda_{1:t}, \lambda_{t+1}))$$

Expected entropy reduction over all possible future measurements at next step

$$\begin{aligned} EER(\lambda_{t+1}) &= \mathbb{E}_{\hat{z}_{t+1}}[I(\hat{z}_{t+1}, \lambda_{t+1})] \\ &= \int p(\hat{z}_{t+1}|\hat{x}_{t+1}, \lambda_{t+1}) I(\hat{z}_{t+1}, \lambda_{t+1}) d\hat{z}_{t+1} \end{aligned}$$

Sample Particles for Entropy Calculation

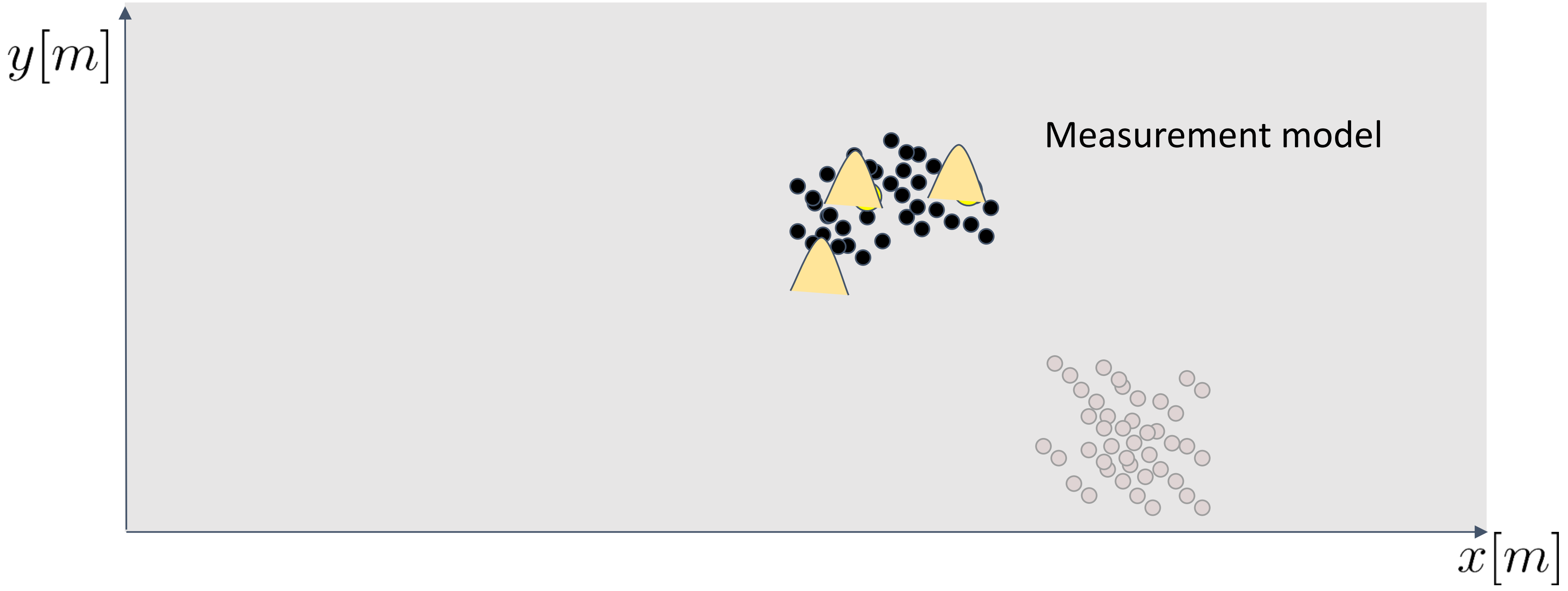
Sample N_s particles from the particle states distribution at **k** time-steps to calculate expected entropy



Sample Particles for Entropy Calculation

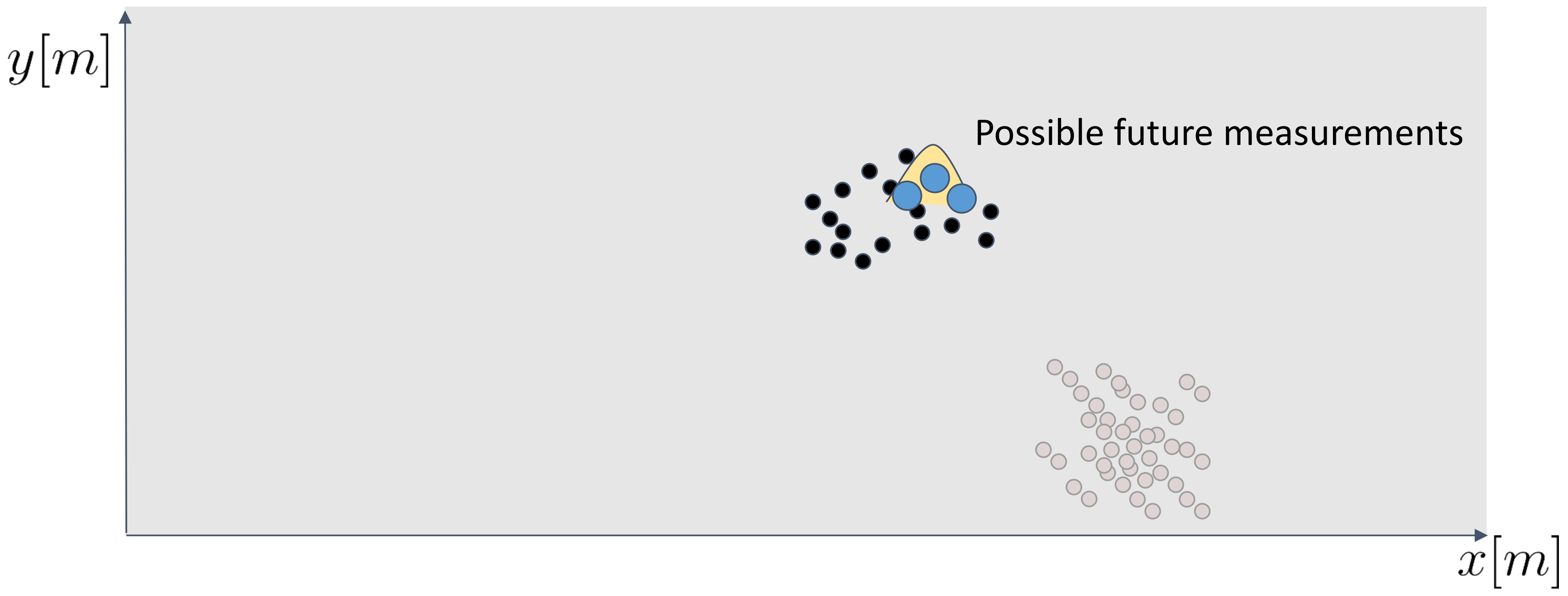
Each of the sampled particles has a pdf that indicates how likely it is to be measured, this is the **measurement model**. In our case, it is Gaussian.

The measurement model generates the measurement likelihood, $p(z(t+k))$



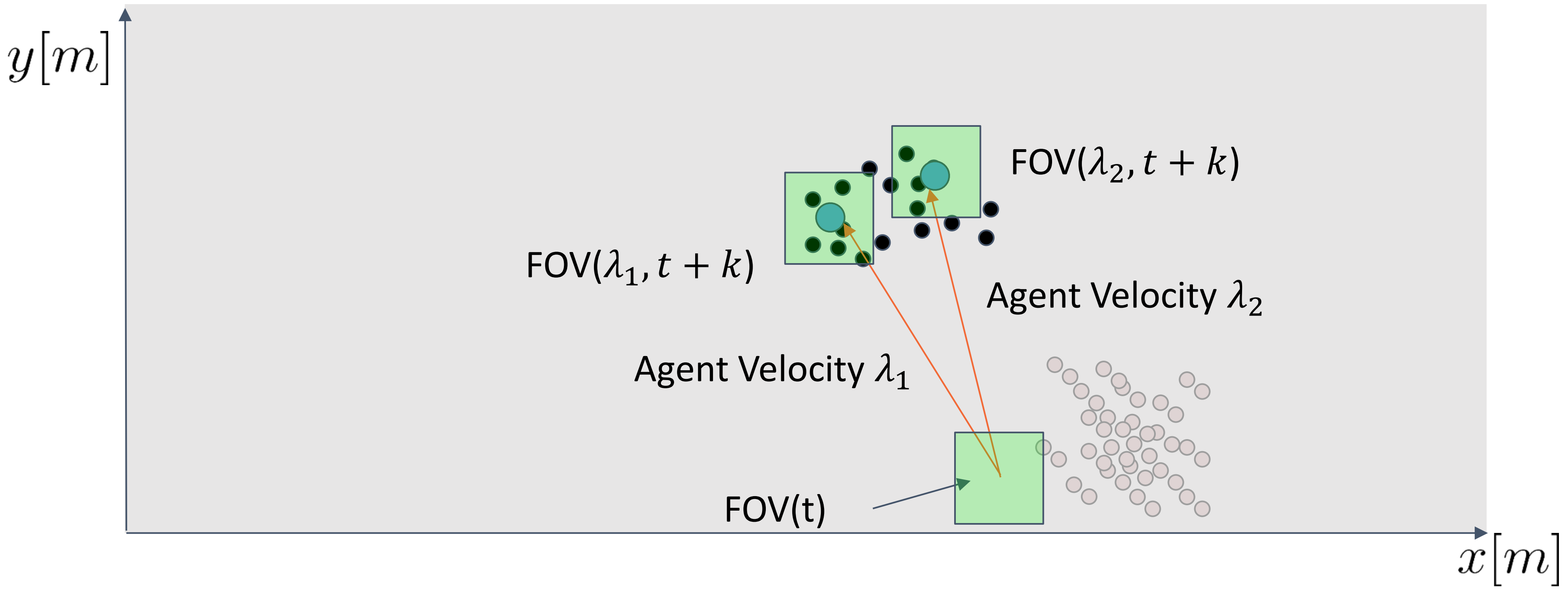
Sample Particles for Entropy Calculation

N_m measurements are sampled from the PDF to get the possible future measurement. We take the entropy with these N_m possible measurements



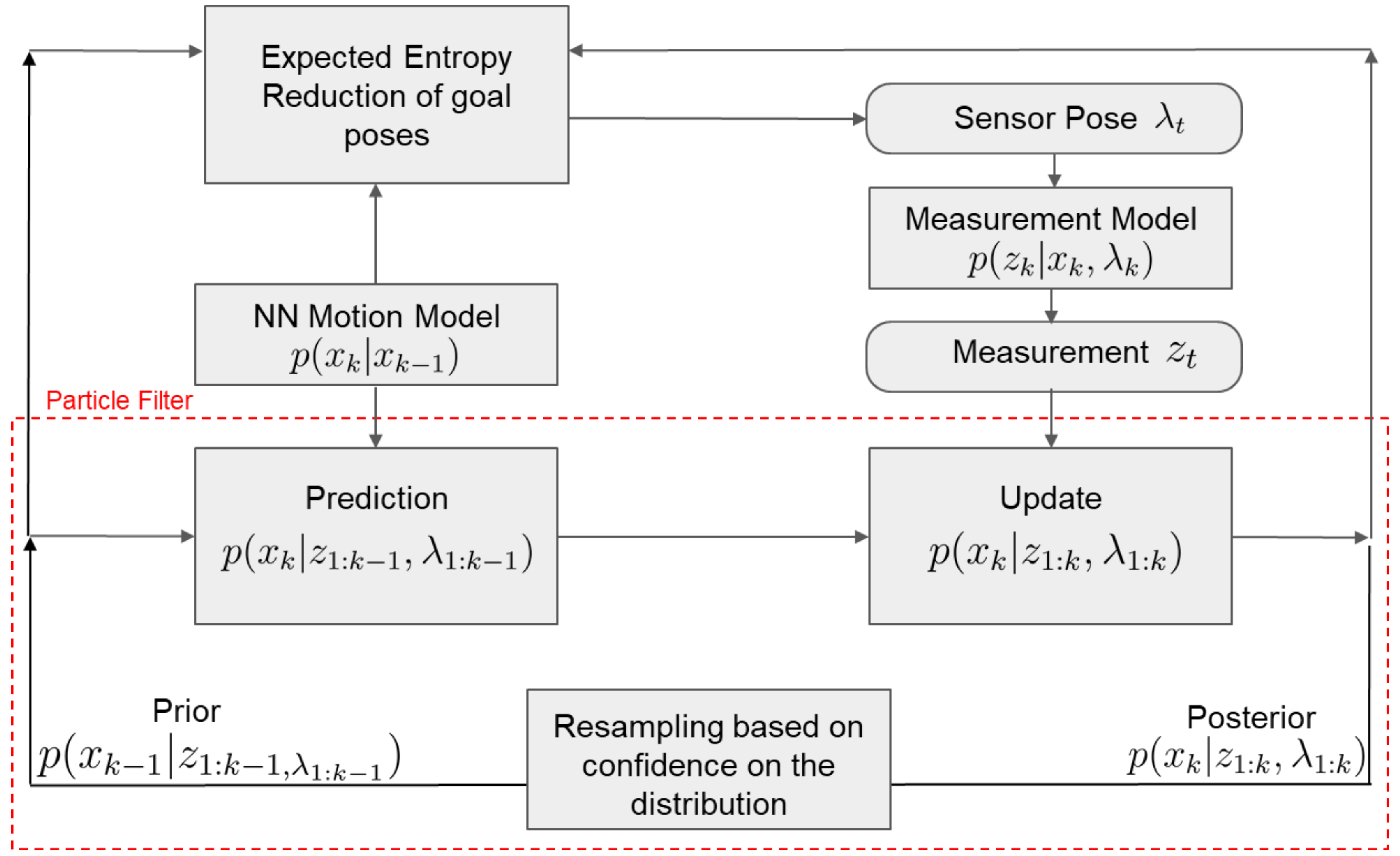
Actions

The sampled particles positions are the set of possible goal positions to go. Actions are the velocities that will take the drone to those positions. Select the action for time *t* that maximizes the expected entropy reduction.



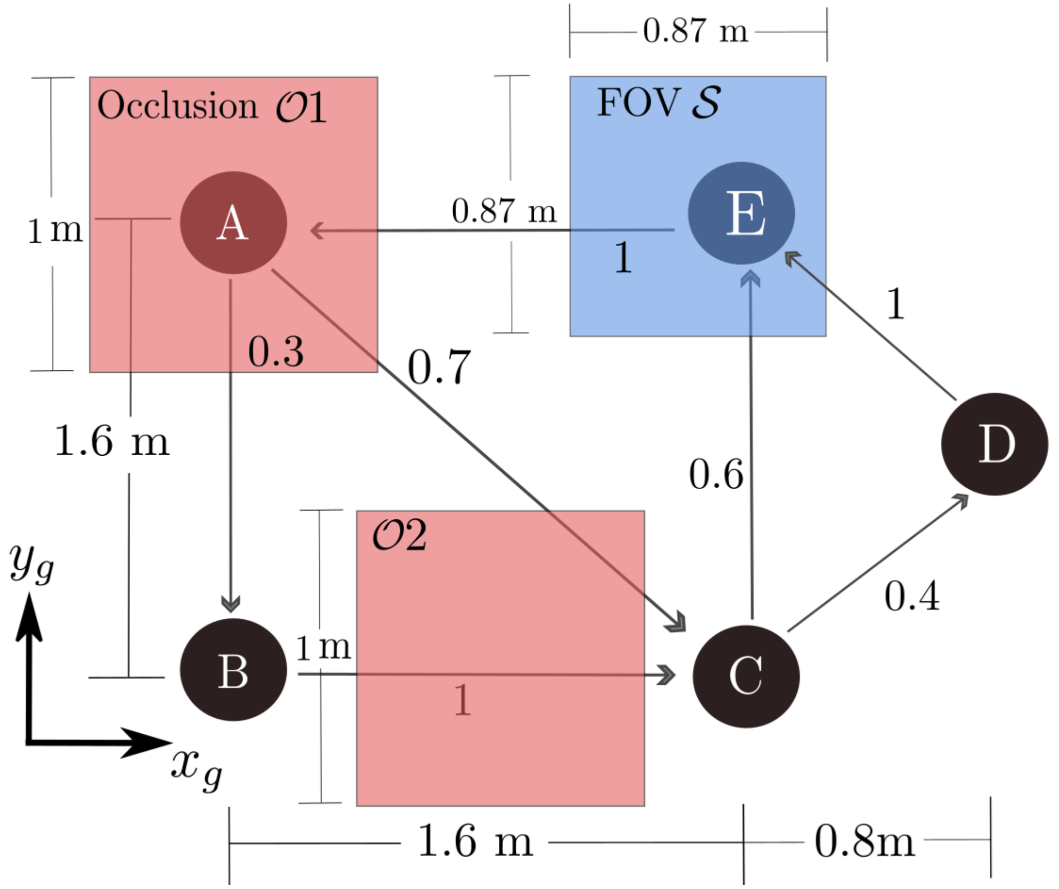
$$\lambda_t = argmax\ EER(\lambda_m)$$

DNN Particle Filter Guidance Architecture


Expected Entropy Reduction of goal poses
Sensor Pose λ_t
Measurement Model $p(z_k|x_k, \lambda_k)$
NN Motion Model $p(x_k|x_{k-1})$
Measurement z_t
Particle Filter
Prediction $p(x_k|z_{1:k-1}, \lambda_{1:k-1})$
Update $p(x_k|z_{1:k}, \lambda_{1:k})$
Prior $p(x_{k-1}|z_{1:k-1},\lambda_{1:k-1})$
Resampling based on confidence on the distribution
Posterior $p(x_k|z_{1:k}, \lambda_{1:k})$

Test Scenario

- Two occlusions (one in a stochastic node)
- Agent’s FOV with smaller size than the occlusions
- Five nodes in the Markov road network
- Two stochastic nodes (A and C)



Guidance Methods

Particles: go to the position of the weighted mean of the particle filter

$$p_{goal} = \mu_{particles} = \sum_{i=1}^{n} w_i p_i$$

Lawnmower: go to the position of the last target measurement, and if a measurement is not available perform a lawnmower path in the working area of the target

$$p_{goal} = z_{target}$$

Information (to compare): go to the position of the maximum Expected Entropy Reduction (EER) at the future k time step

$$p_{goal} = p^{k}_{max\,EER}$$

KF: go to the mean estimated position from a Kalman Filter which also estimates velocities from sensor data

$$p_{goal} = \mu$$

Guidance Comparison Metrics

- Tracking error is the difference between the positions of the agent (s_k) and target (x_k)

$$e = \|s_k - x_k\|$$

- This captures the intuition that the agent should stay above the target
- Estimation error is the error between the position of the weighted mean of the particle filter and the position of the target

$$\tilde{e} = \|\mu_k - x_k\|$$

$$\mu_k = \sum_{i=1}^{N} w_k^{(i)} x_k^{(i)}$$

The determinant of the particle position covariance matrix is a measure of estimate uncertainty.

$$\Sigma_k = \left(\sum_{i=1}^{N} w_k^{(i)} (\mu_k - x_k^{(i)})\right)\left(\sum_{i=1}^{N} w_k^{(i)} (\mu_k - x_k^{(i)})\right)^{\mathrm{T}}$$

USSF AFRL

ACTIVE PERCEPTION AND ROBOT INTELLIGENCE LAB

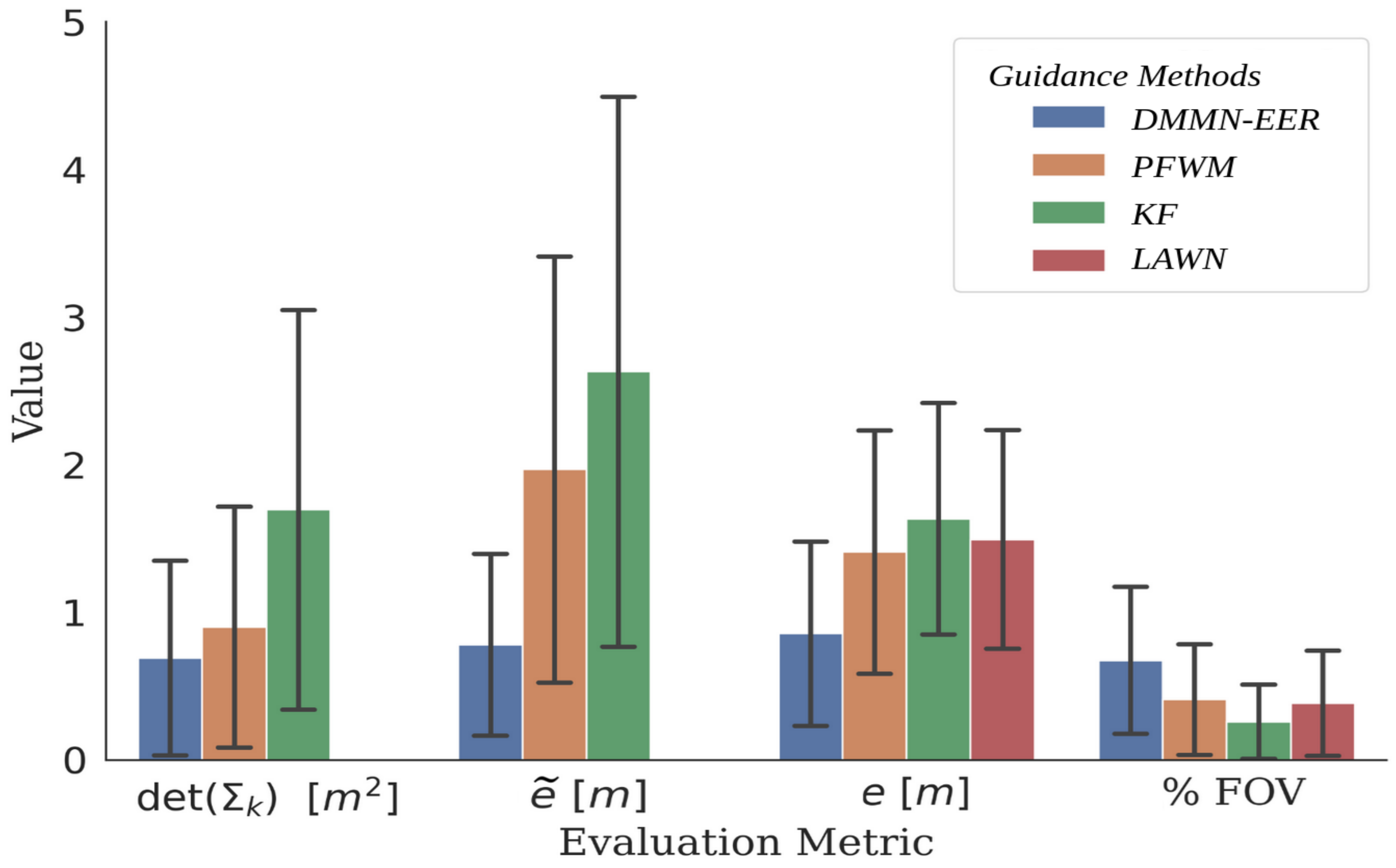
THE AIR FORCE RESEARCH LABORATORY

Research & Engineering Education Facility UNIVERSITY of FLORIDA

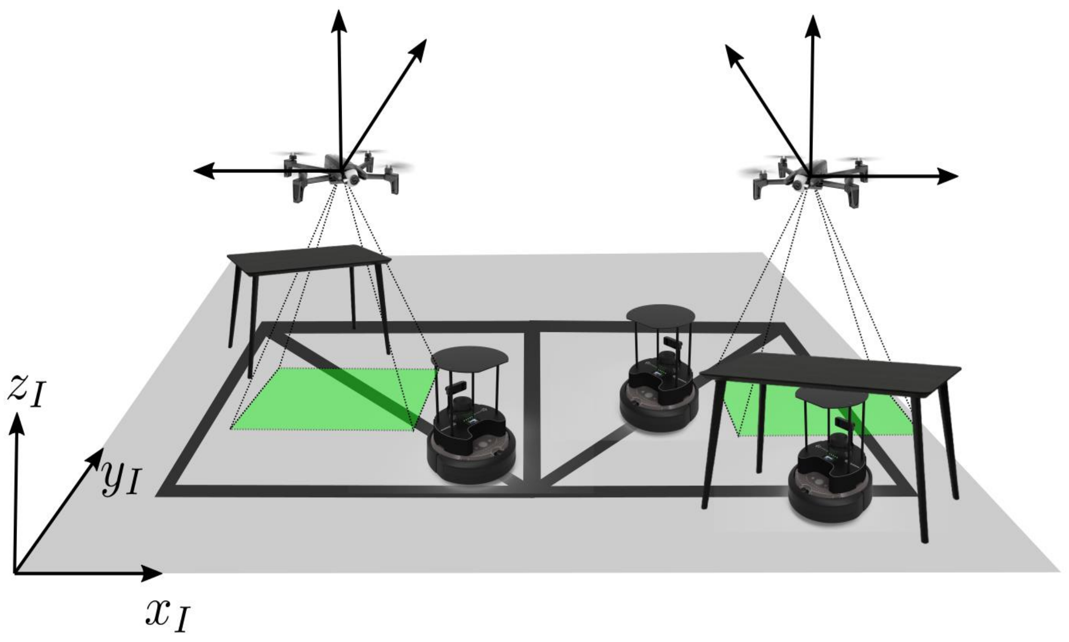
Uncertainty-Aware Guidance for Target Tracking subject to Intermittent Measurements using Motion Model Learning

Andres Pulido, Kyle Volle, Prashant Ganesh, Zach Bell, Jane Shin

Comparison of Results



Next steps: Multi-agent and Multi-target Tracking



- Extend to multi-vehicles: 2 vs 3 target tracking
- Incorporate policy approximation for every agent with recent deep approximate dynamic programming results
- Leverage hierarchical reinforcement learning for resource allocation to determine what target to track and which agent to use
- Decision maker using entropy for where to get most information

Max L. Greene, Zachary I. Bell, Scott Nivison, Warren E. Dixon, "Deep Neural Network-based Approximate Optimal Tracking for Unknown Nonlinear Systems", *IEEE Transactions on Automatic Control,* (2023).
Wanjiku A. Makumi, Zachary I. Bell, Warren E. Dixon, "Approximate Optimal Indirect Regulation of an Uncertain Agent with a Lyapunov-Based Deep Neural Network", *IEEE Control Systems Letters,* (2023).
Wanjiku A. Makumi, Max L. Greene, Zachary I. Bell, Scott Nivison, Rushikesh Kamalapurkar, Warren E. Dixon, "Heiarchical Reinforcment Learning-based Supervisory Control of Unknown Nonlinear Systems", IFAC World Congress, July 2023.
Wanjiku A. Makumi, Zachary I. Bell, Warren E. Dixon, "Lyapunov-based Deep Reinforcement Learning for Approximate Optimal Control", *IEEE Transactions on Automatic Control, submitted* (2024).

Next Steps: Multi-Target Selection via Deep Approximate Dynamic Programming

- The deep ADP approach consists of two DNNs working together per subsystem in a model-based actor-critic architecture
  - First, we use separate DNN motion models to estimate the dynamics of each target, just as we are currently doing to estimate the motion model previously described.
  - We pair that with separate DNNs to estimate the value function associated with each pairing of an agent tracking each potential target. With ADP, the control policy is derived from the gradient of the value function as below with the DNN estimating the value.
  - Currently investigating how to incorporate the entropy from the particle filter into the cost for each agent to improve exploration in the policy and better cost structure in general for the multi-agent problem.
  - Also, must determine how the switching strategy/minimum dwell-time must change from a previous HRL result which selected the minimizing value subsystem and computed a minimum dwell-time condition to remain in that subsystem.

- Optimal value function (cost-to-go)

$$V^*(x) = \min_{\mu(\tau)\epsilon U} \int_t^\infty Q(\zeta(\tau)) + \mu(\tau)^T R \mu(\tau)\, d\tau$$

- Optimal control policy is based on gradient of value

$$\mu^*(\zeta) = -\frac{1}{2} R^{-1} G(\zeta)^T \left(\nabla_\zeta V^*(\zeta)\right)^T$$

DNN Optimal Value Function and DNN Optimal Control Policy

$$V^*(\zeta) = W^T \psi(\Psi^*(\zeta)) + \epsilon_v(\zeta) \qquad \mu^*(\zeta) = -\frac{1}{2} R^{-1} g(\zeta)^T \left(W^T \nabla_\zeta \psi(\Psi^*(\zeta)) + \nabla_\zeta \epsilon_v(\zeta)^T\right)$$

$\widehat{W}_c$ = Critic weight estimate
$\widehat{W}_a$ = Actor weight estimate

Optimal Value Function and Optimal Control Policy Approximation

$$\hat{V}(\zeta, \widehat{W}_c) = \widehat{\boldsymbol{W}}_{\boldsymbol{c}}{}^T \psi\left(\widehat{\Psi}_i(\zeta)\right) \qquad \hat{\mu}(\zeta, \widehat{W}_a) = -\frac{1}{2} R^{-1} g(\zeta)^T \left(\widehat{\boldsymbol{W}}_{\boldsymbol{a}}{}^T \nabla_\zeta \psi\left(\widehat{\Psi}_i(\zeta)\right)^T\right)$$

Max L. Greene, Zachary I. Bell, Scott Nivison, Warren E. Dixon, "Deep Neural Network-based Approximate Optimal Tracking for Unknown Nonlinear Systems", *IEEE Transactions on Automatic Control,* (2023).
Wanjiku A. Makumi, Zachary I. Bell, Warren E. Dixon, "Approximate Optimal Indirect Regulation of an Uncertain Agent with a Lyapunov-Based Deep Neural Network", *IEEE Control Systems Letters,* (2023).
Wanjiku A. Makumi, Max L. Greene, Zachary I. Bell, Scott Nivison, Rushikesh Kamalapurkar, Warren E. Dixon, "Hierarchical Reinforcement Learning-based Supervisory Control of Unknown Nonlinear Systems", IFAC World Congress, July 2023.
Wanjiku A. Makumi, Zachary I. Bell, Warren E. Dixon, "Lyapunov-based Deep Reinforcement Learning for Approximate Optimal Control", *IEEE Transactions on Automatic Control, submitted* (2024).

Questions?