2021 Clean Space Industrial Days

AI-aided Guidance and Navigation for Dynamics Reconstruction of Uncooperative Spacecraft

Dr. Stefano Silvestrini, Prof. Michèle Lavagna

POLITECNICO MILANO 1863

DIPARTIMENTO DI SCIENZE E TECNOLOGIE AEROSPAZIALI

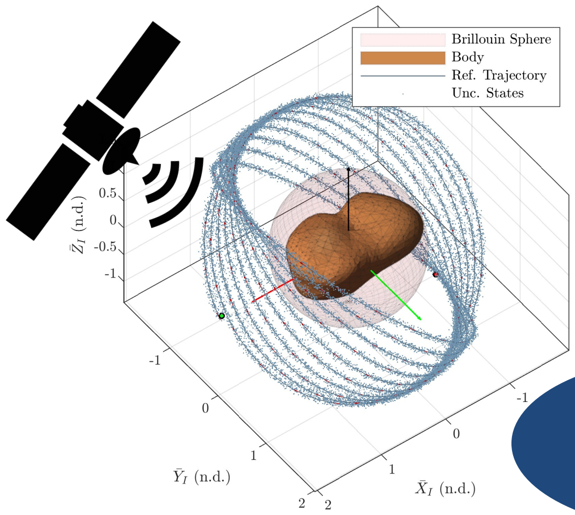
Advanced Space Technologies for Robotics and Astrodynamics

Outline

- Research Context & Motivation
- Research Objective and Methodology
- Neural Dynamics Learning & Navigation
- Neural-aided Guidance & Control
- Environment and External Agents Prediction
- Final remarks

Research Context & Motivation:
Proximity Operations and Relative Dynamics

Proximity Operations for debris removal



Exploration missions

- **Proximity Operations** allows daring mission objectives (close proximity with asteroids, planets, etc.)

↓

Challenging flight conditions due to **distances involved** and **uncertain environment**

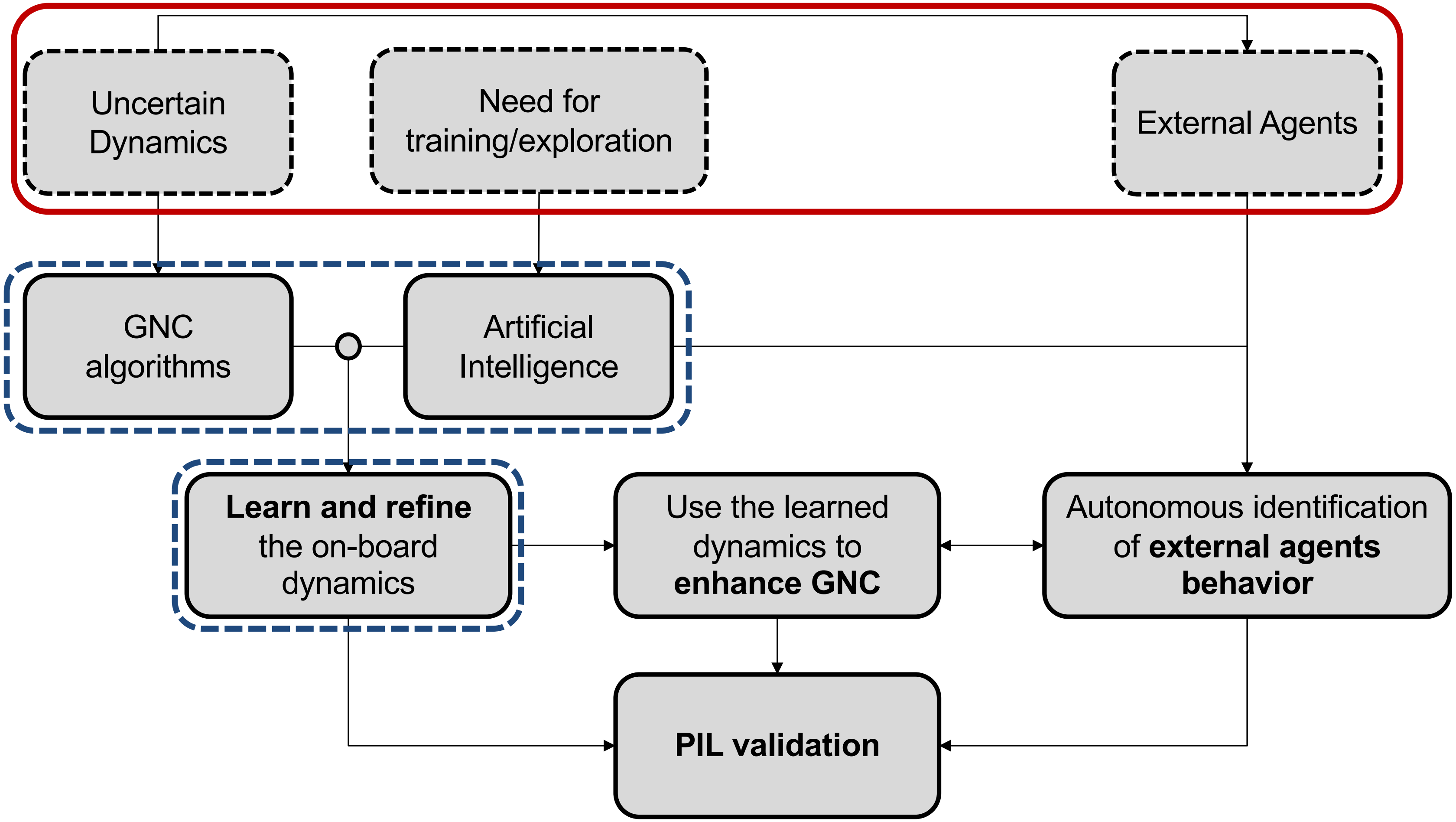
↓

Need for accurate, flexible, adaptive and light GNC algorithm

Research Objectives

Research methods:

framework and drivers



Neural Dynamics Learning & Navigation:
Dynamics Reconstruction

Learn and refine the on-board dynamics

$$\dot{\mathbf{x}} = \boxed{\mathcal{N}(\mathbf{x}, \mathbf{u})} \qquad \dot{\mathbf{x}} = A\mathbf{x} + \boxed{\gamma(\mathbf{x}, \mathbf{u})} \qquad \mathbf{y} = A(\mathbf{x}) \cdot \boxed{\mathbf{C}}$$

The dynamics reconstruction can be performed in three ways:

1. **Neural Dynamics**: Fully encapsulated in a NN → very powerful when RNN are used, need rough initialization

2. **Neural Disturbance Reconstruction**: Analytical models refined with disturbance approximation output of the NN → robust due to Lyapunov convergence RBFNN when dealing with uncertain disturbance

3. **Neural Parameter Identification**: Parameter estimation of given analytical models → suitable when only parameters are uncertain

Neural Dynamics Learning & Navigation:
Dynamics Reconstruction

Learn and refine the on-board dynamics

$$\boxed{\dot{\mathbf{x}} = \mathcal{N}(\mathbf{x}, \mathbf{u})} \qquad \dot{\mathbf{x}} = A\mathbf{x} + \gamma(\mathbf{x}, \mathbf{u}) \qquad \mathbf{y} = A(\mathbf{x}) \cdot \mathbf{C}$$

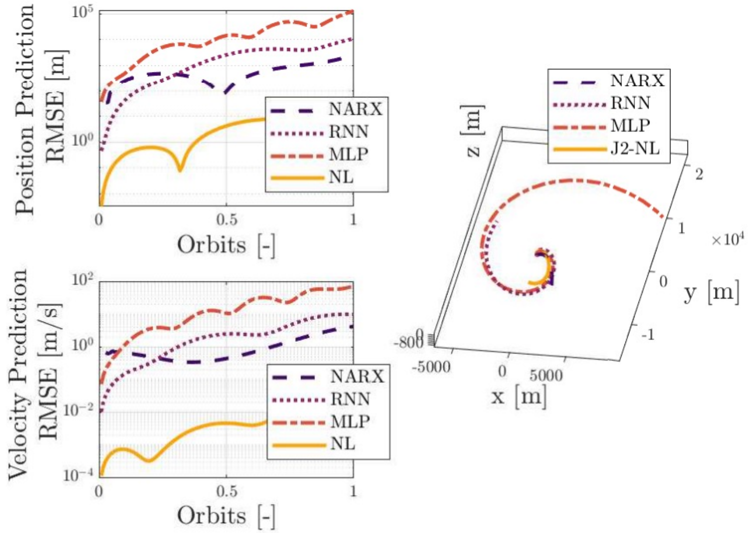
The dynamics reconstruction can be performed in three ways:

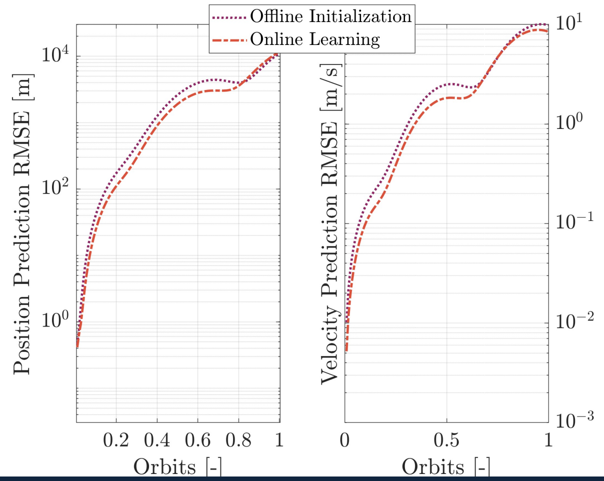
1. **Neural Dynamics**: Fully encapsulated in a NN → very powerful when RNN are used, need rough initialization

2. **Neural Disturbance Reconstruction**: Analytical models refined with disturbance approximation output of the NN → robust due to Lyapunov convergence RBFNN when dealing with uncertain disturbance

3. **Neural Parameter Identification**: Parameter estimation of given analytical models → suitable when only parameters are uncertain

Neural Dynamics Learning & Navigation: Fully-Neural Dynamics



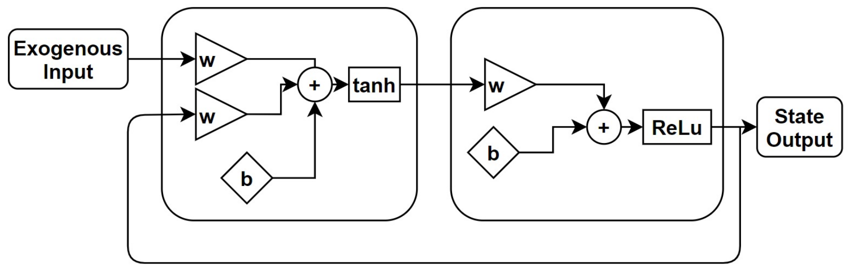


Neural Network is trained with supervised learning using **measurements** as training data:

$$\min_{\mathbf{w}} \sum_i ||\tilde{\mathcal{N}}_{T_s}(\mathbf{x},\mathbf{u},\mathbf{w}) - \mathbf{y}_{k+1}||^2$$

Online learning refines and **incrementally train** the network

Recurrent Neural Networks catches **temporal and secular behavior** → more suitable to fully encapsulate the dynamics



It explicitly recalls the **forced** dynamics

Neural Dynamics Learning & Navigation:
Dynamics Reconstruction

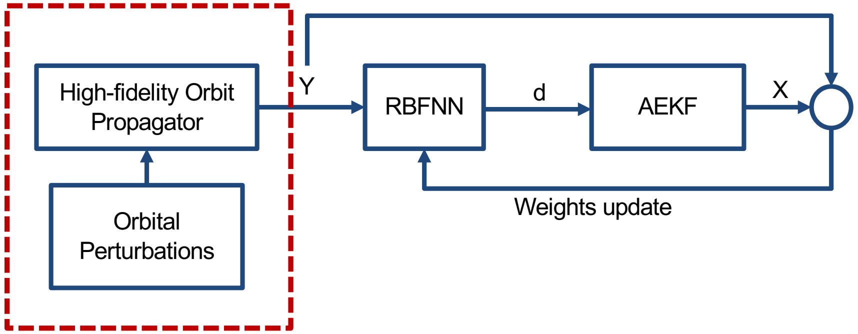
Learn and refine the on-board dynamics

$$\dot{\mathbf{x}} = \mathcal{N}(\mathbf{x}, \mathbf{u}) \qquad \boxed{\dot{\mathbf{x}} = A\mathbf{x} + \gamma(\mathbf{x}, \mathbf{u})} \qquad \mathbf{y} = A(\mathbf{x}) \cdot \mathbf{C}$$

The dynamics reconstruction can be performed in three ways:

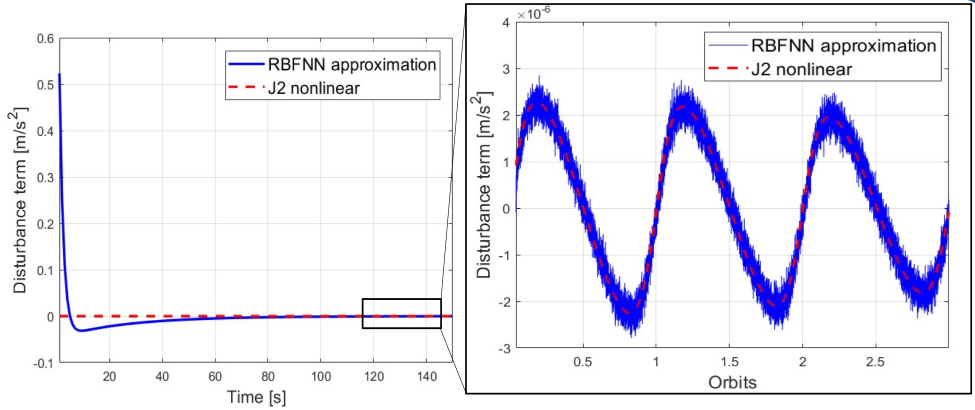
1. **Neural Dynamics**: Fully encapsulated in a NN → very powerful when RNN are used, need rough initialization
2. **Neural Disturbance Reconstruction**: Analytical models refined with disturbance approximation output of the NN → robust due to Lyapunov convergence RBFNN when dealing with uncertain disturbance
3. **Neural Parameter Identification**: Parameter estimation of given analytical models → suitable when only parameters are uncertain

Neural Dynamics Learning & Navigation:
Disturbance Reconstruction

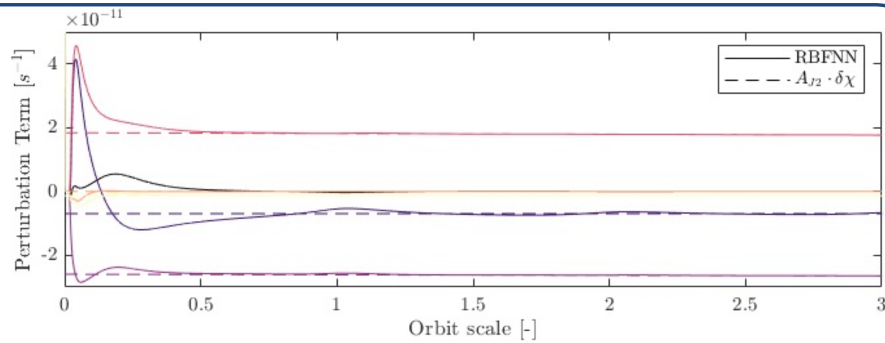


V. Pesce, **S. Silvestrini**, M. Lavagna, Radial Basis Function Neural Network aided Adaptive Extended Kalman Filter for Spacecraft Relative Navigation, Aerospace Science and Technology, vol. 96, 2020

Cartesian dynamics

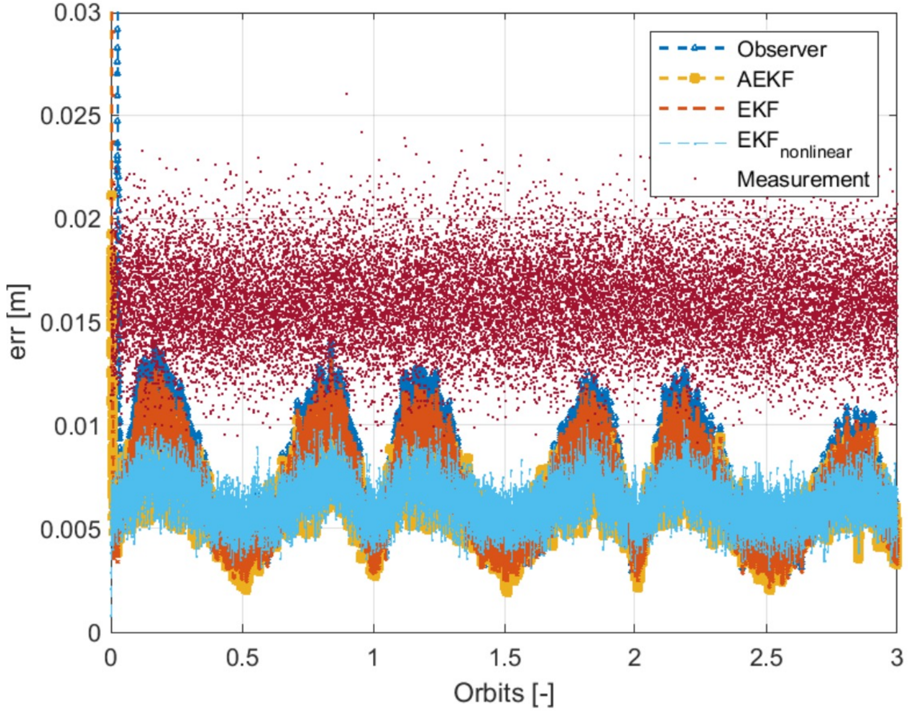


ROE dynamics



***ROE: Relative Orbital Elements**

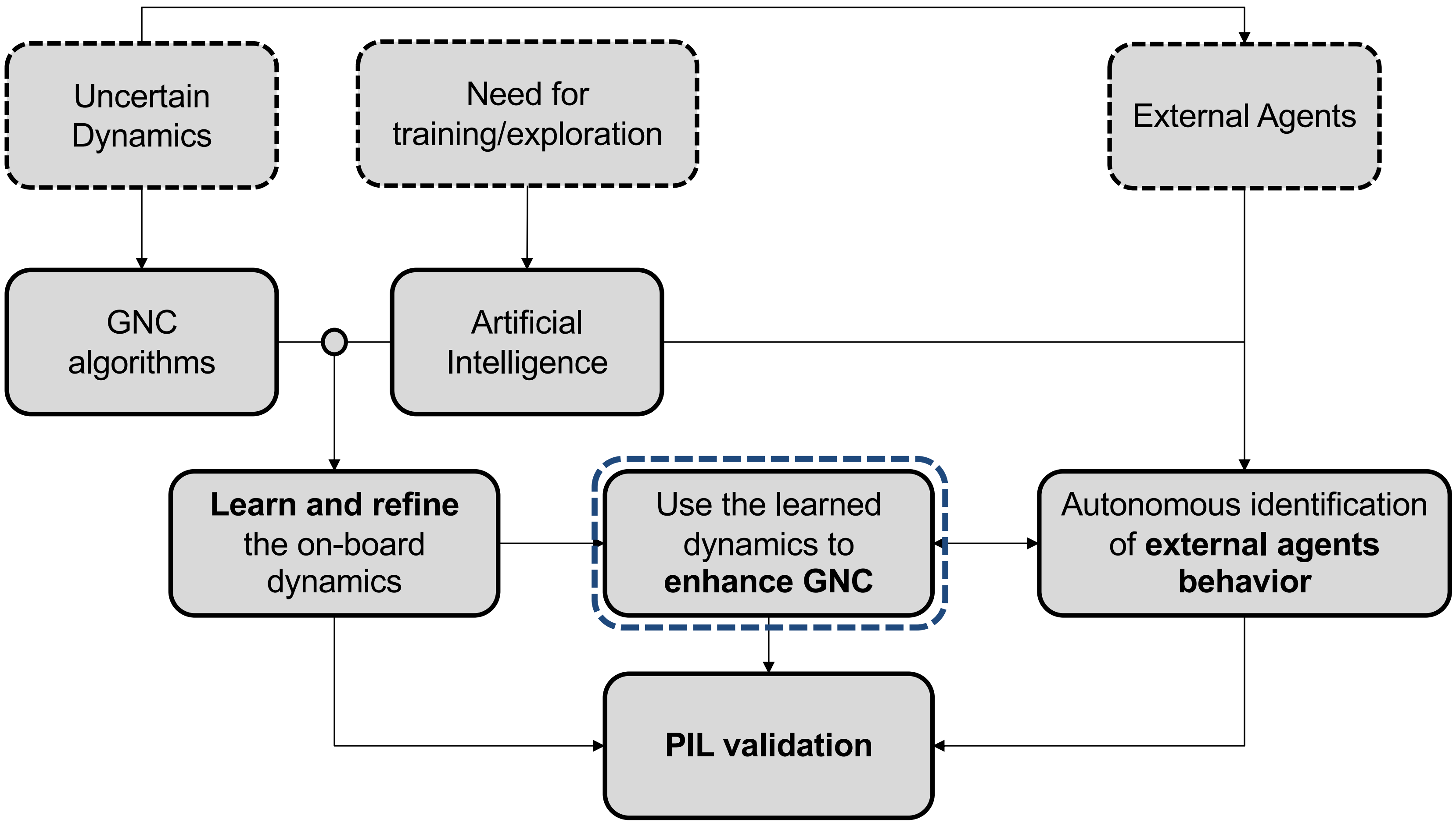
Neural Dynamics Learning & Navigation:
Disturbance Reconstruction



Pros Adaptive EKF-RBFNN	Cons Adaptive EKF-RBFNN
Higher robustness when filter not tuned	Initial main learning process
Reconstructed perturbations term	Stability guaranteed under certain bounds
Better average accuracy	

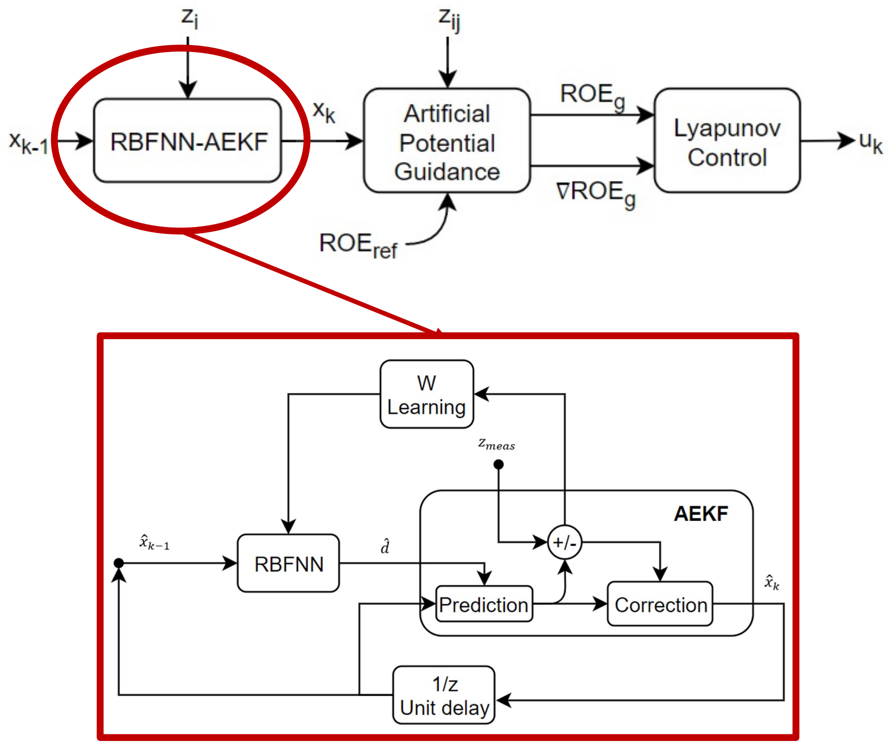
V. Pesce, **S. Silvestrini**, M. Lavagna, Radial Basis Function Neural Network aided Adaptive Extended Kalman Filter for Spacecraft Relative Navigation, Aerospace Science and Technology, vol. 96, 2020

Research Objectives



Neural-aided Guidance & Control:
Radial Basis Function Neural Network - Artificial Potential Field

APF reconfiguration coupled with a **neural controller** based on **reconstructed relative dynamics**



The set of ROE to be achieved are called **reference state** and indicated as $\delta\chi_r$:

$$\Phi_a(\delta\chi) = \frac{1}{2}\xi_a\left\|\delta\chi_g - \delta\chi_r\right\|^2$$

Calculating the gradient:

$$\nabla_{\delta\chi_g} = \xi_a(\delta\chi_g - \delta\chi_r)$$

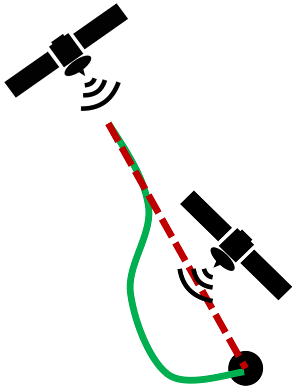
The repulsive potential is useful to calculate the trajectory in presence of other satellites, **avoiding collision between agents**.

$$\Phi_{r_{ij}} = \begin{cases} \frac{1}{2}\xi_r e^{-\frac{d_{ij}^2}{\eta}} = \frac{1}{2}\xi_r e^{-\frac{\left\|\boldsymbol{x_i}-\boldsymbol{x_j}\right\|^2}{\eta}} & \text{if } d_{ij} < d_{lim}, \\ 0 & \text{if } d_{ij} > d_{lim} \end{cases}$$

The gradient of the potential is calculated using the chain-rule, which involves the coordinate transformation from Cartesian state X to ROE:

$$\nabla_{\delta\chi_g}\Phi_{r_{ij}} = -\frac{\xi_r}{\eta}e^{-\frac{d_{ij}^2}{\eta}} \cdot (\boldsymbol{X_i} - \boldsymbol{X_j}) \cdot J_{\delta\chi}^{X}$$

Neural-aided Guidance & Control:
Radial Basis Function Neural Network - Artificial Potential Field



The output of the artificial potential guidance is a set of ROE, which may differ from the target reference ones. **Feedback control law** to guarantee that the **forced guidance dynamics** is followed.

$$\delta\dot{\chi}_{\mathbf{g}} = -\nabla\Phi_{glb} + (\mathbf{A_k} + \mathbf{A_{J2}})\cdot\delta\chi$$

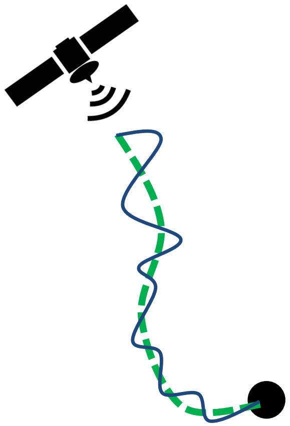
Control Lyapunov function (CLF):

$$\dot{V} = \left(\delta\chi_g - \delta\chi\right)^T \cdot \left[-\left(\nabla\Phi_a + \nabla\Phi_r + \gamma(\delta\chi) + B\mathbf{u}\right)\right]$$

The following control law is derived:

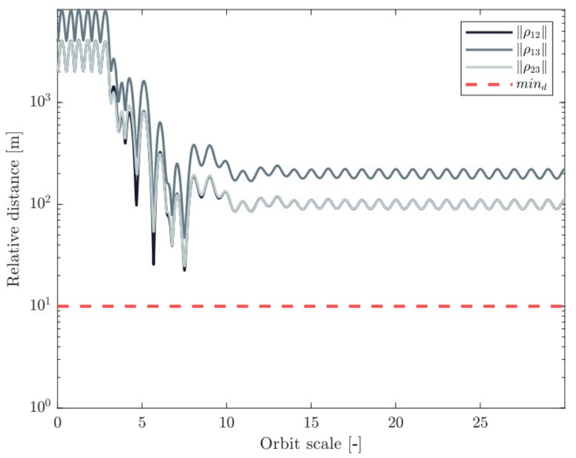
Neural Approximation

$$\mathbf{u} = \mathbf{B}^{-1}\left[\left(\delta\chi_g - \delta\chi\right) - \left(\nabla\Phi_a + \nabla\Phi_r\right) - \gamma(\delta\chi)\right]$$

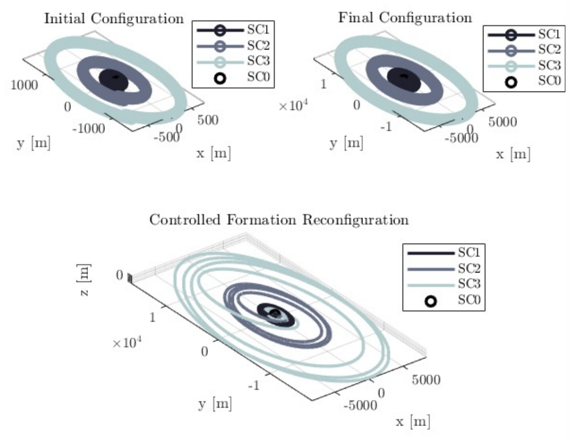


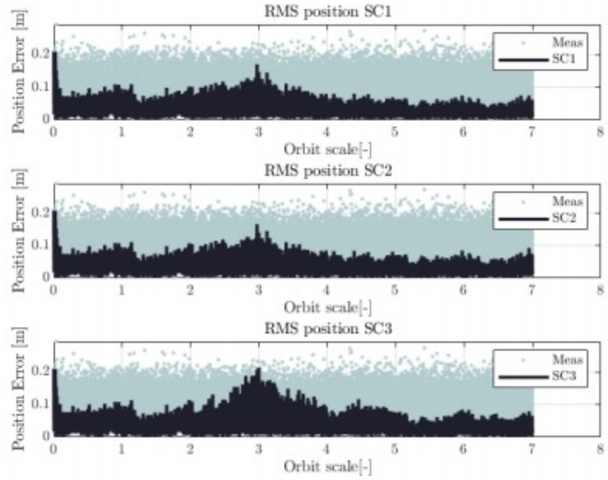
The repulsive potential relies on **mutual relative position**, hence a distributed architecture.

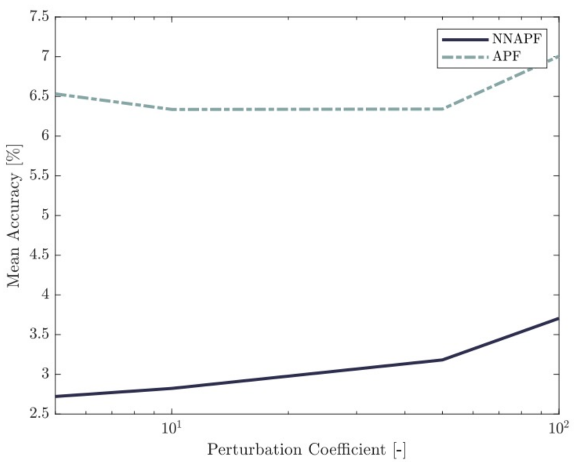
Neural-aided Guidance & Control:
Radial Basis Function Neural Network - Artificial Potential Field Numerical Results



Relative distance **safe** thanks to collision avoidance repulsive contribution







The **higher the perturbation** the better RBFNN-APF performs with respect to APF

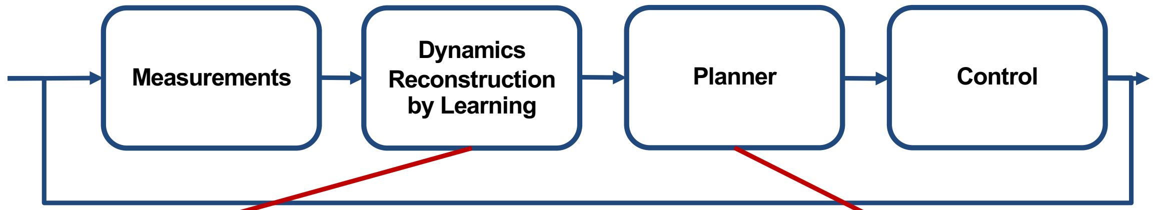
PROS RBFNN-APF	CONS RBFNN-APF
Better control accuracy	Δv < 10 m/s Slightly higher control wrt APF (~ 5%)
Better estimation accuracy	

S. Silvestrini, M. Lavagna, Neural-aided GNC Reconfiguration Algorithm for Distributed Space System: Development and PIL test, Advances in Space Research, vol. 67, 5, 1490-1505, 2021

Neural-aided Guidance & Control

Model-based Reinforcement Learning for Maneuver Planning: Architecture

Use the learned dynamics to **enhance GNC**



Dynamics Reconstruction

- $\dot{x} = N(x, u, w)$
- $\dot{x} = f(x, u) + N(x, u, w)$
- $\dot{x} = N(x, u, w) \cdot C$

Model predictive control (MPC) - like → **optimization** + **closed-loop**

Plan through $\dot{x} = f(x, u)$ for N_s steps → Execute first control action

Cost function: quadratic in **target state distance + control effort**

$$J(x_k, u_k) = (x_{k+N} - x_k^*)^T \hat{S} (x_{k+N} - x_k^*) + \sum_{i=1}^{N-1} (x_{k+i} - x_k^*)^T S (x_{k+i} - x_k^*) + \sum_{i=0}^{N-1} u_{k+1}^T R u_{k+1}$$

Subject to **dynamics** and **thrust constraint**.

Neural-aided Guidance & Control

Model-based Reinforcement Learning Algorithm

Use the learned dynamics to **enhance GNC**

Initialize NN model $\tilde{N}(x_k, u_k, w_0)$

Collect Measurements
$D = \{(x_k, u_k, x_{k+1})\}$

Train by supervised learning
$x_{k+1} = \tilde{N}(x_k, u_k, w_k)$

Constrained Optimization

for N_s time steps

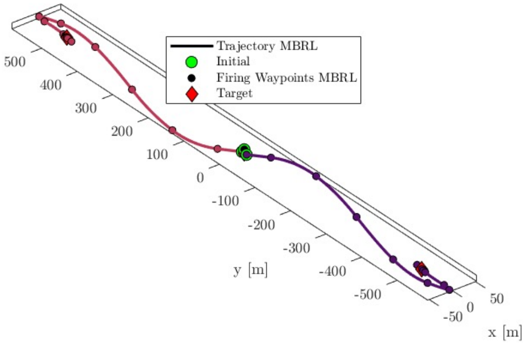
Execution of first control action u_k

Actuators

S. Silvestrini, M. Lavagna, Neural-based Predictive Control and Relative Trajectory Identification Algorithms for Relative Spacecraft Maneuvers, Journal of Guidance, Control and Dynamics, 2020

Neural-aided Guidance & Control

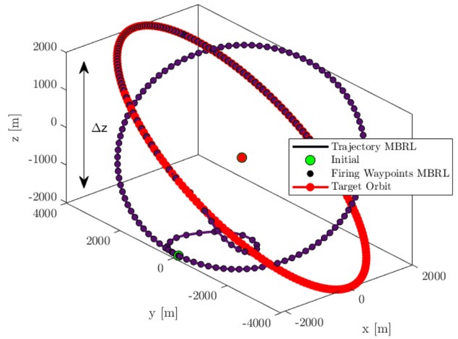
Model-based Reinforcement Learning Numerical Results

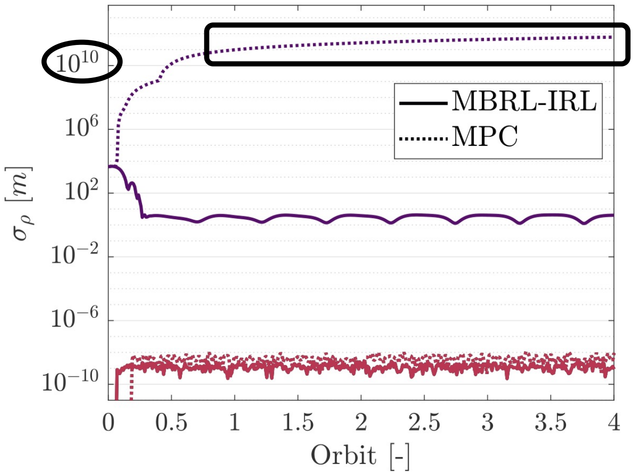


Planar:

Lower Δv ~ 10% for MBRL

Lower TOF for MBRL





Large formation Out-of-Plane:

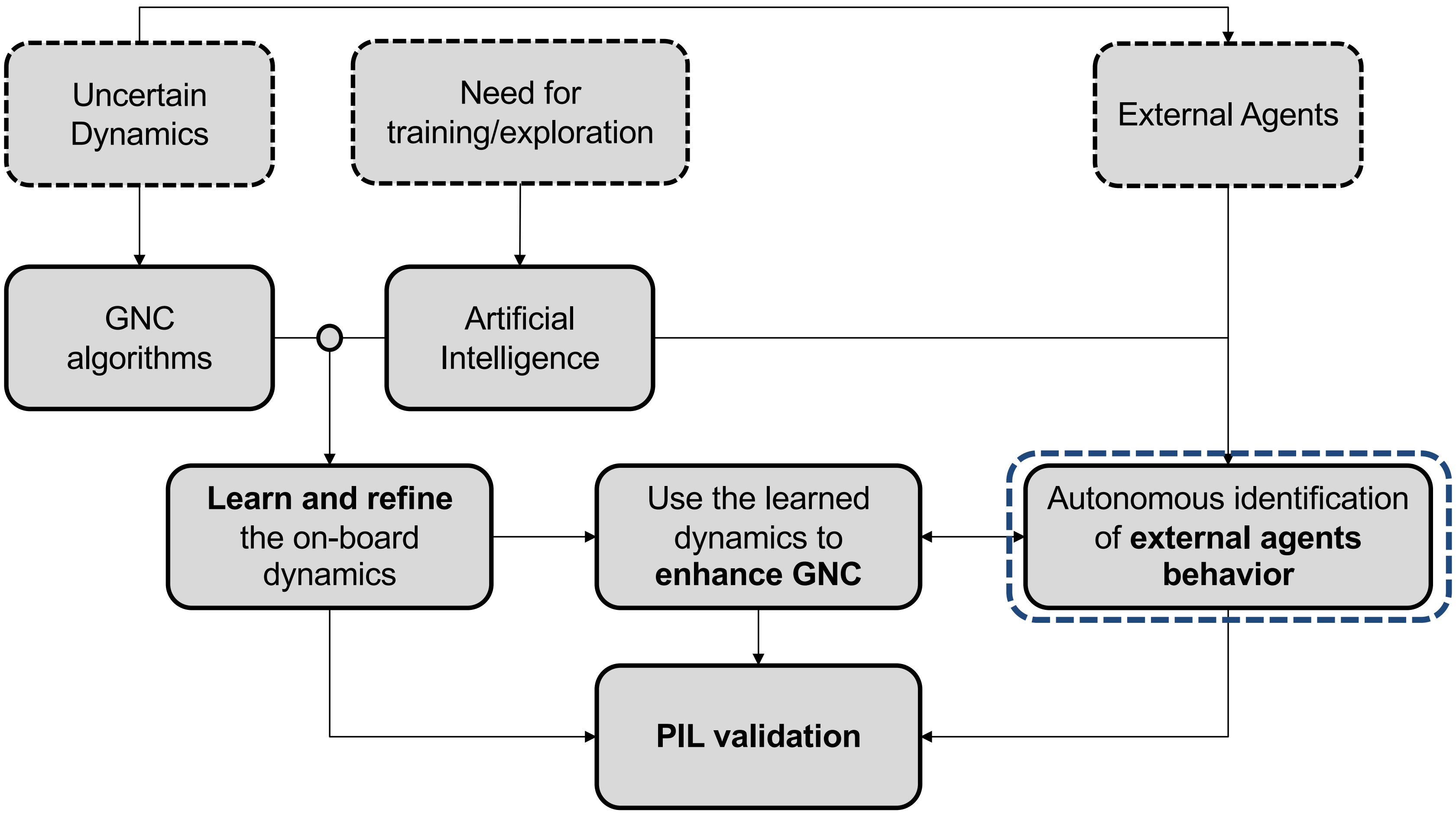
MBRL successful/ MPC fails

PROS MBRL	CONS MBRL
Lower Δv	Initial learning process
Lower TOF (free variable)	Excellent performance with offline initialization
It manages non-modelled environment	Recurrent Neural Networks lacks implementation support

S. Silvestrini, M. Lavagna, Neural-based Predictive Control and Relative Trajectory Identification Algorithms for Relative Spacecraft Maneuvers, Journal of Guidance, Control and Dynamics, 2020

Research Objectives

Advanced Space Technologies for Robotics and Astrodynamics
DIPARTIMENTO DI SCIENZE E TECNOLOGIE AEROSPAZIALI



Environment and External Agents Prediction: Collision Avoidance

Assuming only one agent is maneuvering and the rest following natural motion

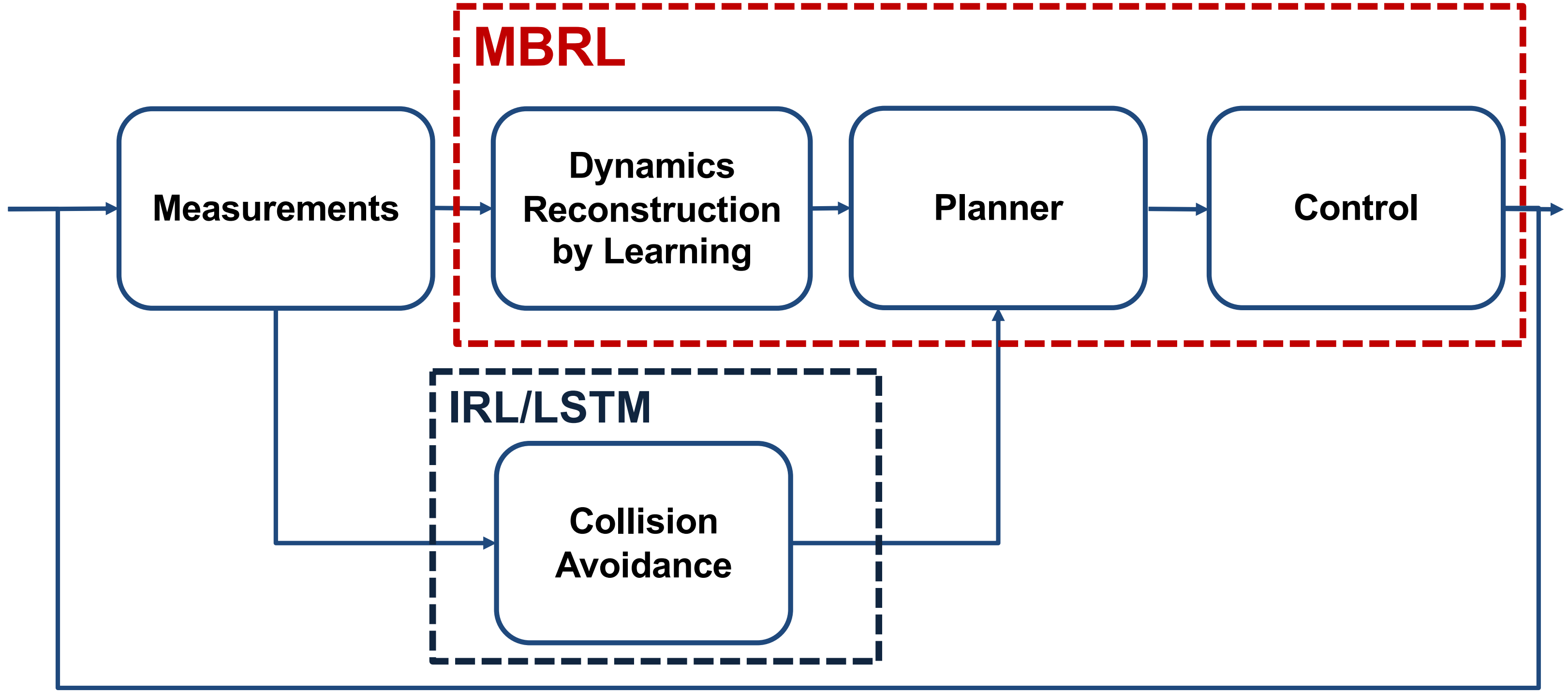
Reconfiguring distributed formation → each agent **plans maneuvers autonomously**
→ need to know what the other agents are doing

One critical task for distributed operation is to safely maneuver **avoiding collision between agents.**

Assuming evolution following natural motion → **too restrictive**

We need a technique to **predict neighboring agents future trajectory based only on past observations of relative positions**

Environment and External Agents Prediction:
Inverse Reinforcement Learning for Collision Avoidance: Algorithm



Autonomous identification of **external agents behavior**

MBRL: Model-based Reinforcement Learning
IRL: Inverse Reinforcement Learning
LSTM: Long Short-Term Memory

Environment and External Agents Prediction:
Inverse Reinforcement Learning for Collision Avoidance: Algorithm

Advanced Space Technologies for Robotics and Astrodynamics
DIPARTIMENTO DI SCIENZE E TECNOLOGIE AEROSPAZIALI

The concept of Inverse Reinforcement Learning is to **estimate a cost function** that delivers an **optimal trajectory compatible with an expert demonstrated trajectory.**

In this research, parametrize the cost function:

$$\mathbf{w} = [w_1, w_2, ..., w_f, w_T]^T, \quad \mu(\pi) = \begin{bmatrix} \sum_t^{T-1} \phi_1(\mathbf{x_t}, \mathbf{u_t}) \\ \sum_t^{T-1} \phi_2(\mathbf{x_t}, \mathbf{u_t}) \\ \dots \\ \sum_t^{T-1} \phi_f(\mathbf{x_t}, \mathbf{u_t}) \\ \phi_T(\mathbf{x_T}) \end{bmatrix}$$

Cumulative feature cost

$$\text{cost: } \mathcal{J} = \mathbf{w}^T \cdot \mu(\pi)$$

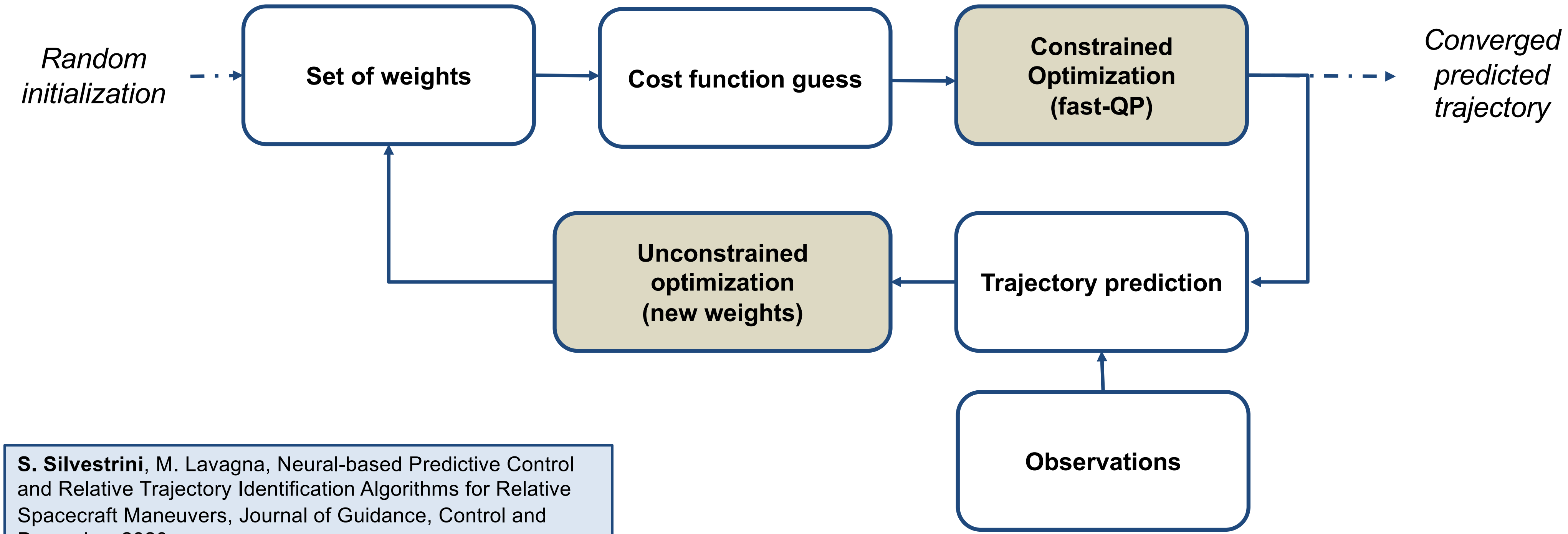
Cumulative **estimated** feature cost

Cumulative **observed** feature cost

Environment and External Agents Prediction:
Inverse Reinforcement Learning for Collision Avoidance: Algorithm

Autonomous identification of **external agents behavior**

In this research a nested optimization is developed:



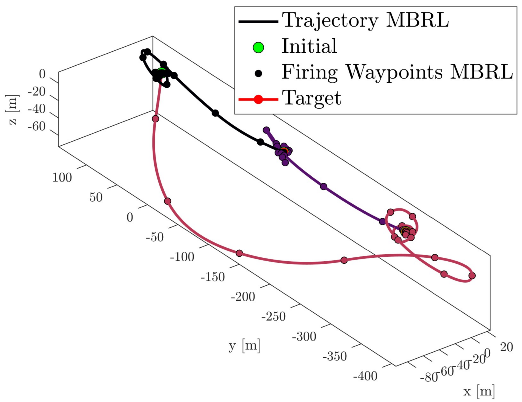
S. Silvestrini, M. Lavagna, Neural-based Predictive Control and Relative Trajectory Identification Algorithms for Relative Spacecraft Maneuvers, Journal of Guidance, Control and Dynamics, 2020

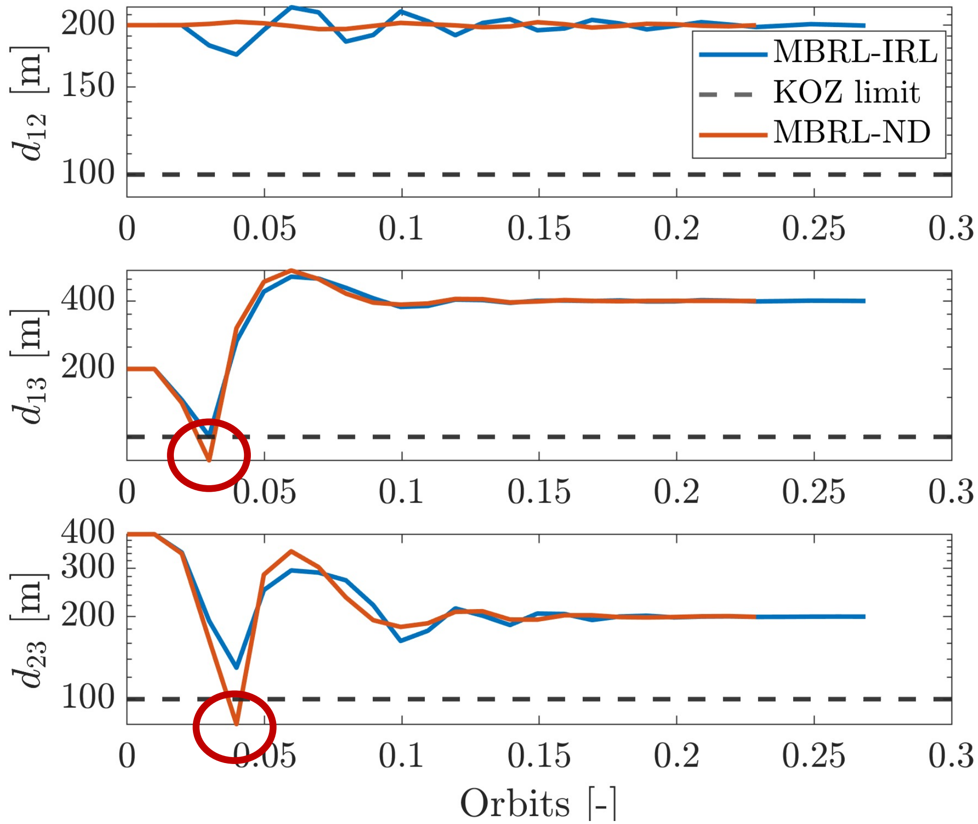
Environment and External Agents Prediction:
Numerical Results

Challenging test case:
Swap the along track positions separated by 200 m.

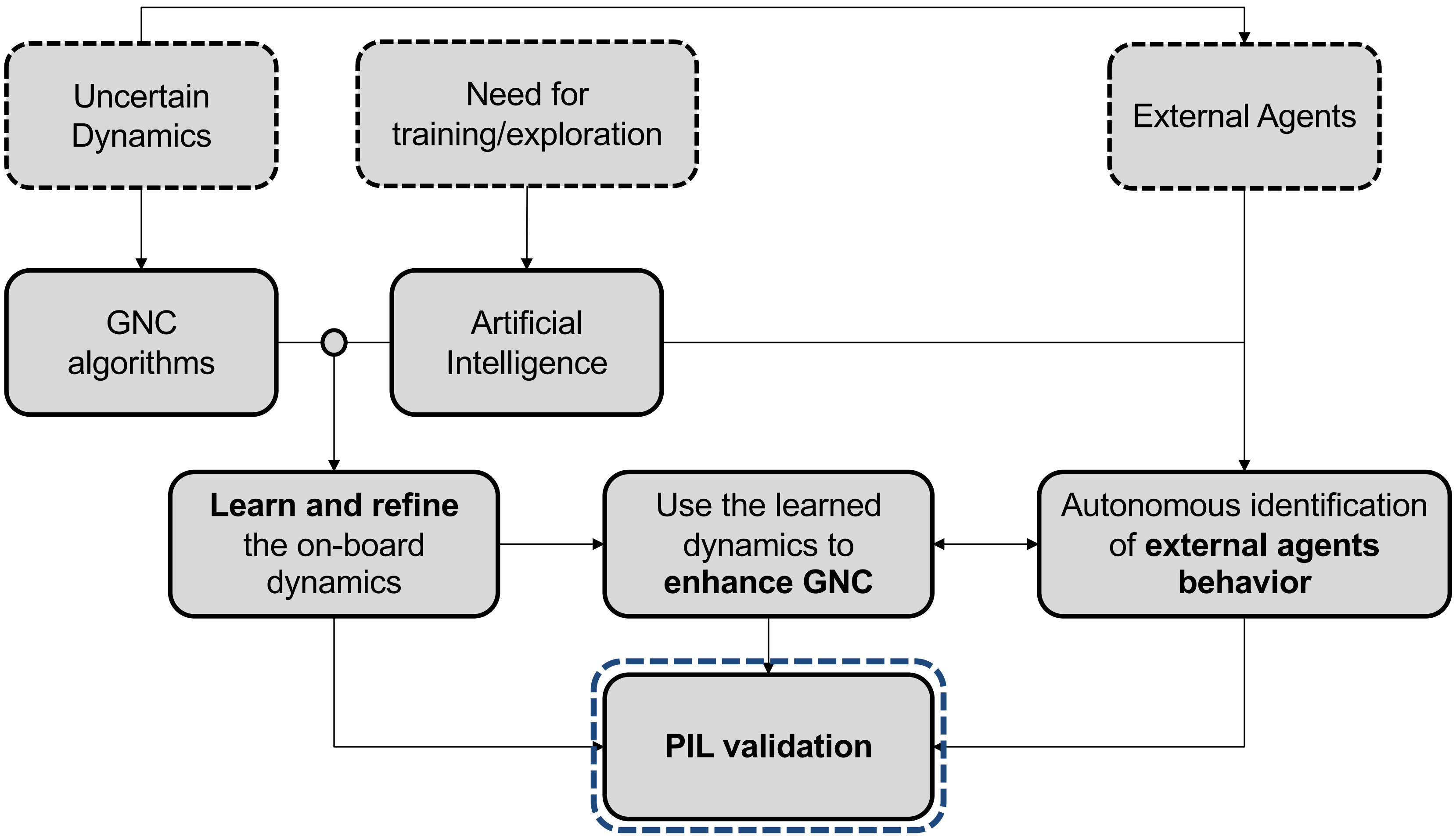
The relative distance between the satellites **falls below the Keep-Out-Zone limit (100 m)**, when predicting neighboring trajectories using natural dynamics.

MBRL planner with an impulsive trajectory identification algorithm, such as IRL, allows a **safe reconfiguration.**



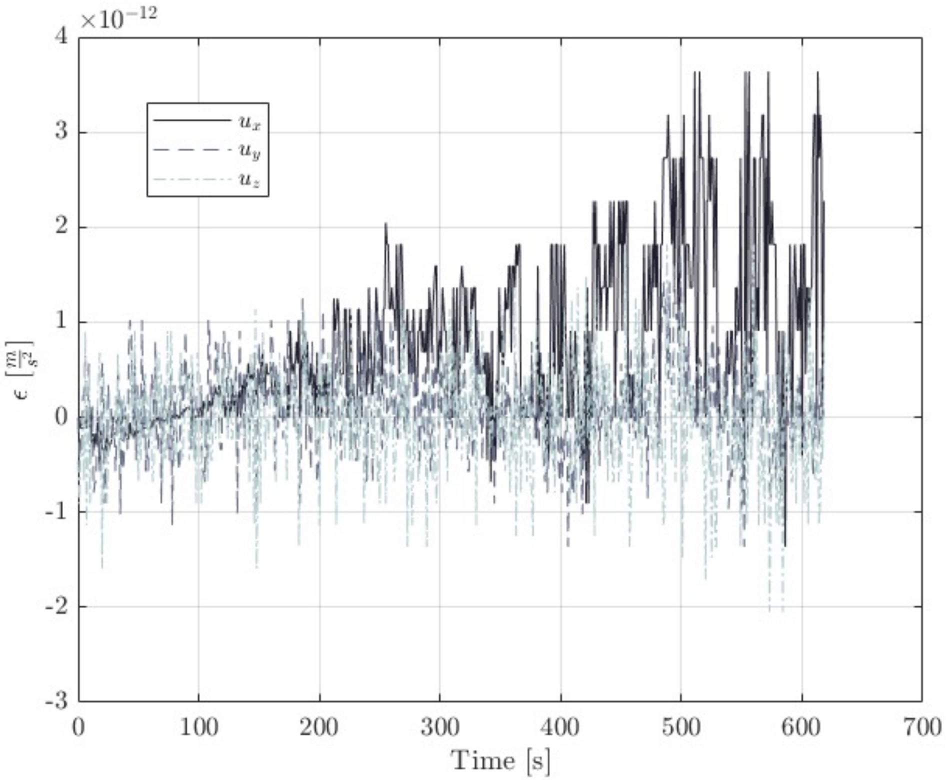


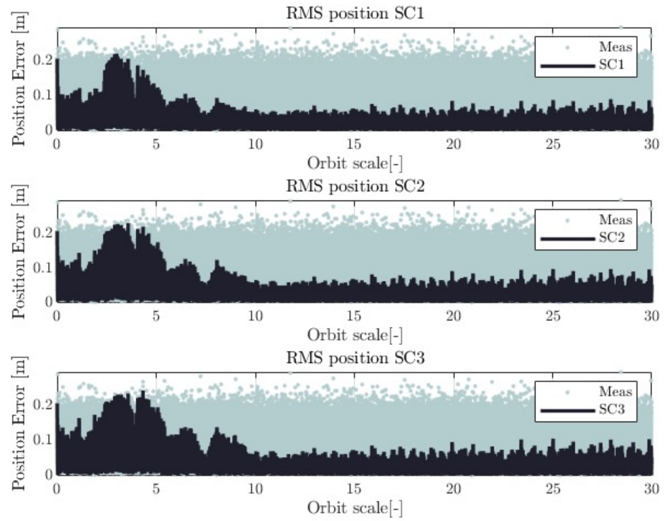
Research Objectives



PIL Validation:
Test Results TI LAUNCHXL-F28379D

Discrepancy between CPU output and PIL module output in the order of 10^{-12} m/s^2





DIPARTIMENTO DI SCIENZE E TECNOLOGIE AEROSPAZIALI

PIL validation

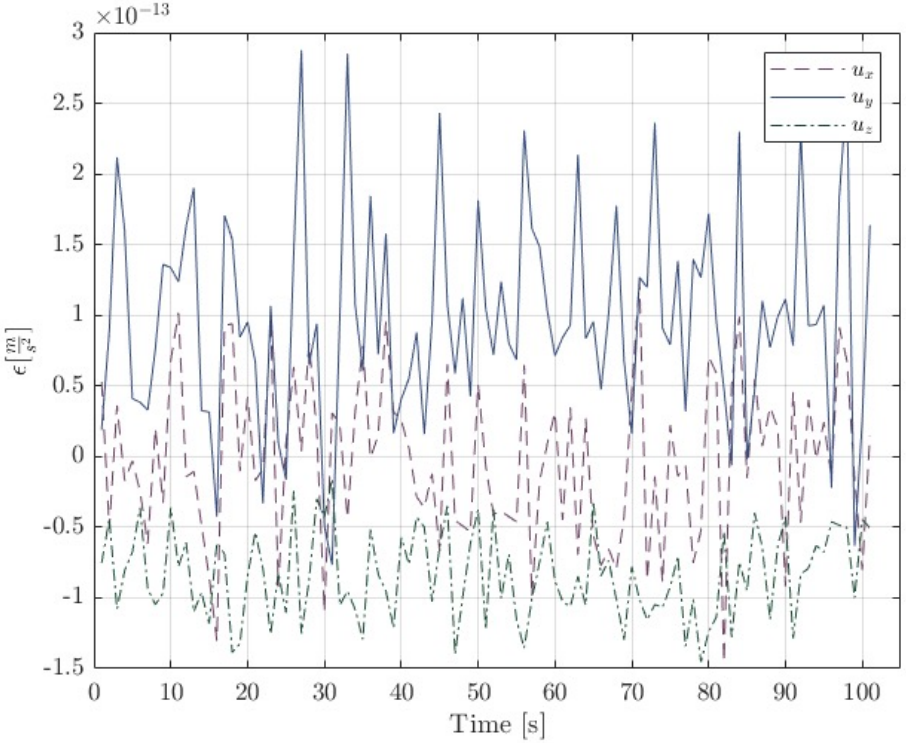
Elapsed times of execution: Runner Profiler

Routine	Avg [ms]	Max [ms]
ANN-learning	0.89	0.90
Navigation	0.81	0.82
Guidance	0.32	0.32
Control	0.28	0.33

PIL Validation:
Test Results BeagleBoneBlack

PIL validation

MBRL and MPC for IRL requires **double** precision. Discrepancy between CPU output and PIL module output in the order of 10^{-13} m/s^2



On-board **resource utilization**:

qp: ~ 0.3 % for MBRL running 1/60 Hz
sqp: ~ 0.5 % for MBRL running 1/60 Hz

Percentage of CPU time assigned to a task. Computed by dividing task execution time by sample time.

Elapsed times of execution: Runner Profiler
MBRL sample time is 60 s.

Routine	Avg [ms]	Max [ms]
ANN-learning	0.89	0.90
MBRL (qp)	191.10	272.20
MBRL (sqp)	290.10	337.10

Thank you for your attention.
Questions?