Enhancing Autonomy of Unmanned Surface Vehicles through Integrated Perception and Control

Collision Avoidance System for Sonobot

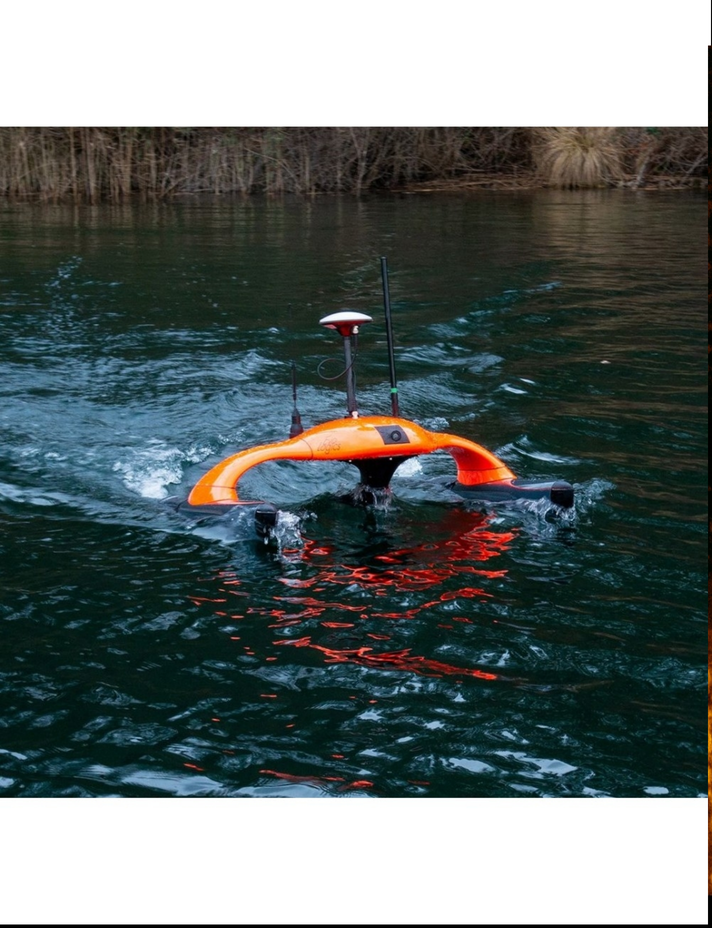
University of Salzburg
Juan M. Montoya
Simon Hirländer
Christian Borgelt

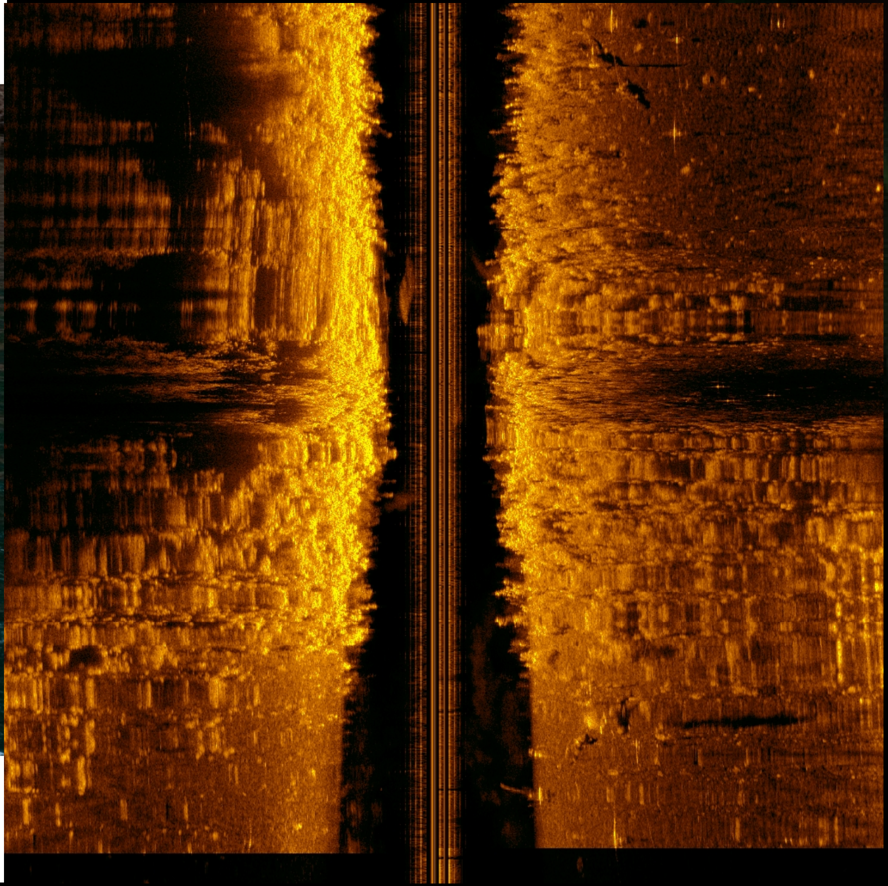
Table of Contents

EvoLogics®

PARIS LODRON UNIVERSITÄT SALZBURG

1. EvoLogics' Unmanned Surface Vehicle: Sonobot





Research Motivation

Integrated GPS for auto pilot-system

YET don't have any collision avoidance system requiring supervision

EvoLogics®

PARIS LODRON UNIVERSITÄT SALZBURG

2. MPC vs RL

- Model Predictive Controller (MPC) is explicitly model-based, utilizing a known model of the system to make predictions and control decisions.
- RL is data-driven, implicitly creating a model from the data it interacts with.
- MPC operates under the assumption of known system dynamics, making it suitable for systems with predictable behaviors.
- RL is capable of handling complex, non-linear systems with unknown dynamics, making it versatile in dealing with unpredictable environments.
- MPC provides stability and performance guarantees
- Both MPC and RL are integral components of control systems

2. Model Predictive Control

- MPC uses a dynamic model to predict system behavior and optimize control decisions.
- Physical models are essential in MPC.
- Past measurements are utilized to predict the system’s most likely next state.
- Both regulation and estimation in MPC require dynamic models and optimization.

2. Model Predictive Control

```
class ModelPredictiveControl:
    def initModelPredictiveControl(t):
        # Horizon steps t for planning
        horizon = t

    def dynamic_model(u, current_state):
        # u is the input (e.g actions)
        # Internal dynamic model of the process for planning
        planed_state= current_state + change_in_time
        return planed state

    def cost_function(u):
        # Cost function J over the receding horizon
        return cost
```

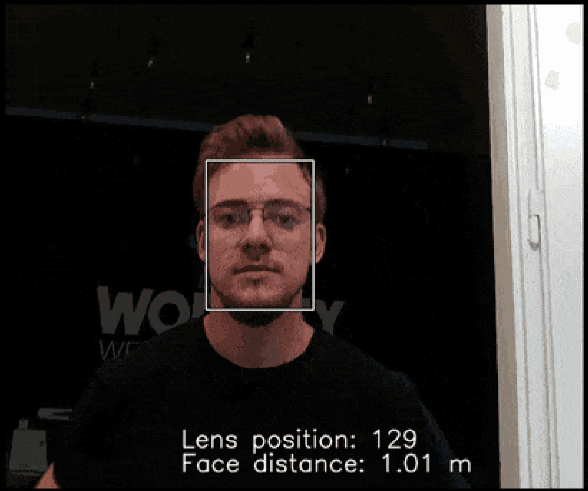
- Control loop minimize cost function by planning for t horizon using the dynamic model

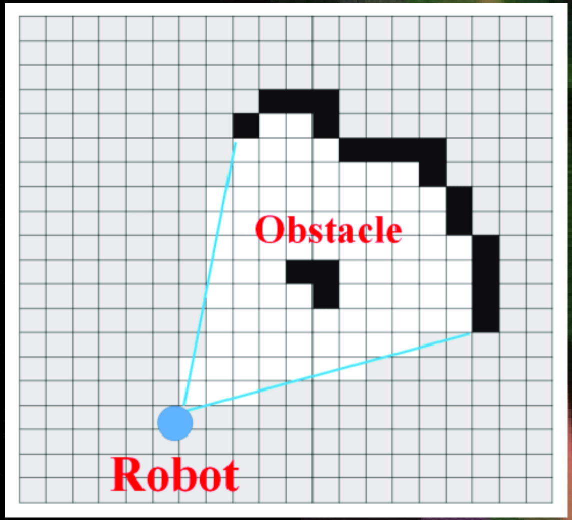
EvoLogics®

PARIS LODRON UNIVERSITÄT SALZBURG

3. Agent Setup: Input & Output

- **Perception Layer**: Stereo Camera and Forward Looking Sonar
- **Agent's input**: Posprocessed Camera Data and sensor data of Sonobot (GPS Position, start position and end position)
- **Agent's output**: GPS Position
- **Path Controller:** Proportional–integral–derivative controller (PID)
  - **No need of dynamic model in system for static obstacles!**

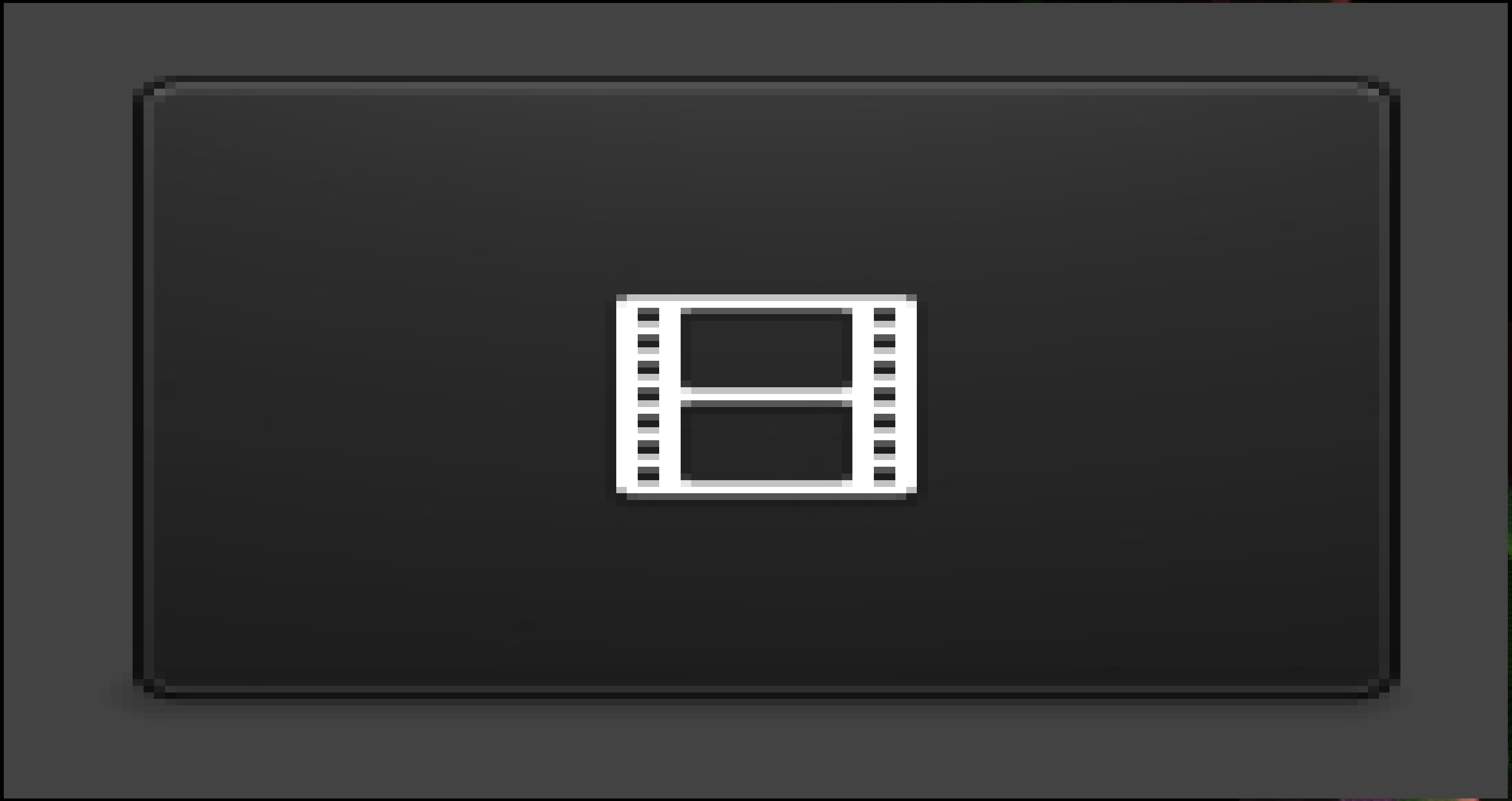




EvoLogics®

PARIS LODRON UNIVERSITÄT SALZBURG

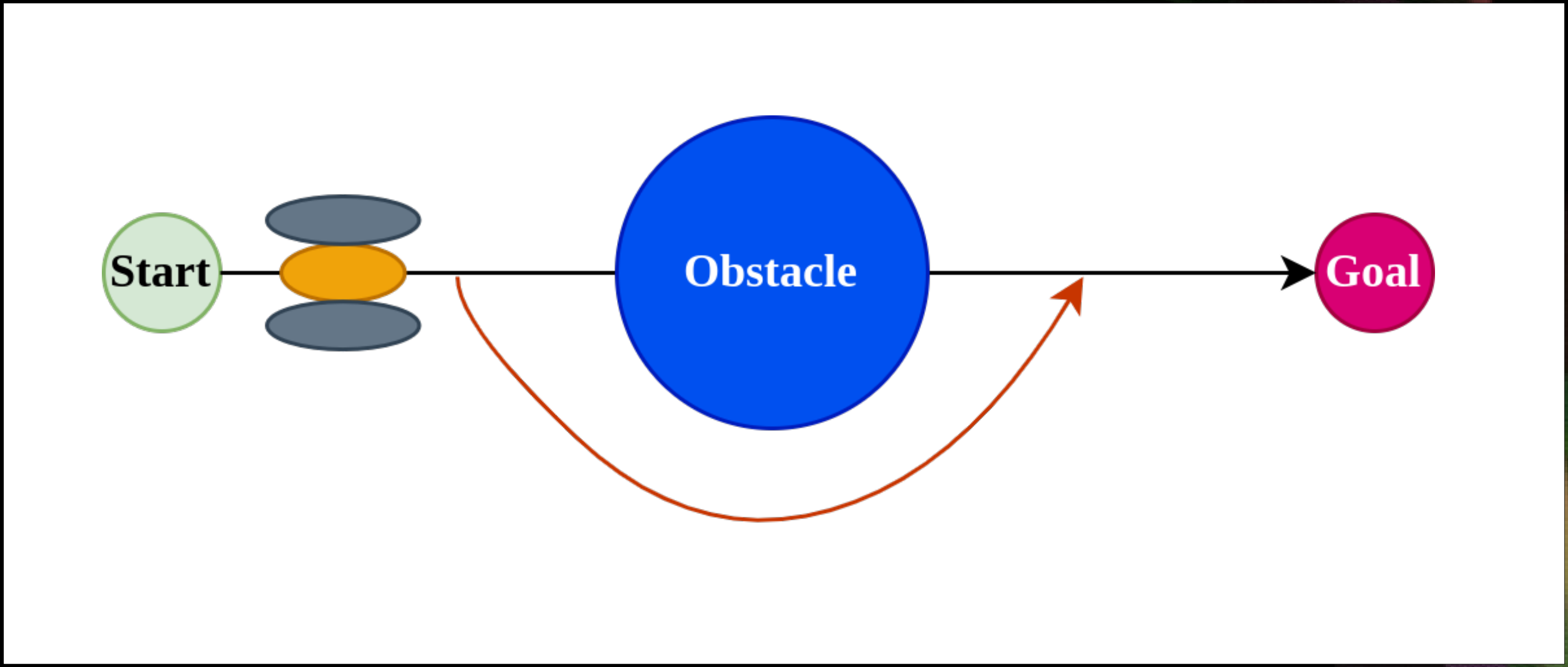
3. Perception Layer Example



EvoLogics®

PARIS LODRON UNIVERSITÄT SALZBURG

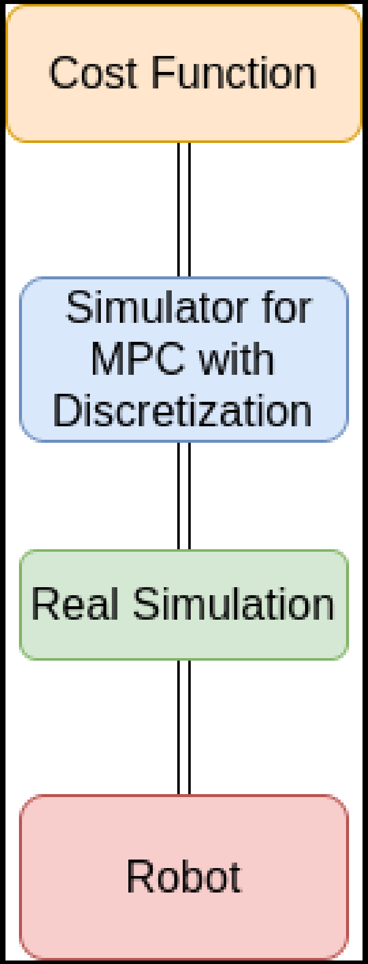
3. Building block: Goto-Maneuver for static objects



EvoLogics®

PARIS LODRON UNIVERSITÄT SALZBURG

4. Research and Development Flow



I. **Reward Engineering:** Optimization using Gradient Descent in a continuous space.

II. **Discretization:** Representing the path as discrete points and testing the MPC algorithm.

III. **Software Engineering:** Integration of new libraries and testing with existing components.

IV. **Unanticipated Problems:** All other issues not yet considered...

EvoLogics®

PARIS LODRON UNIVERSITÄT SALZBURG

4. Cost Function

- The attraction and repulsion terms of the cost function:

$$Waypoint(x_{current}, y_{current}, x_{goal}, y_{goal}) = \frac{1}{(c_1((x_{current} - x_{goal})^2 + (y_{current} - y_{goal})^2) + 1)}$$

  - where $x_{current}$ and $y_{current}$ are, for example, the current GPS coordinates of the Sonobot and x_{goal} and y_{goal} the desired point of the goto maneuver

$$Collision(x_{current}, y_{current}, x_{obj}, y_{obj}) = \frac{1}{(c_2(\sqrt{(x_{current} - x_{obj})^2 + (y_{current} - y_{obj})^2} - r) + 1)}$$

  where x_{obj} and y_{obj} are, for example, the GPS coordinates of the dangerous object to avoid for the Sonobot.

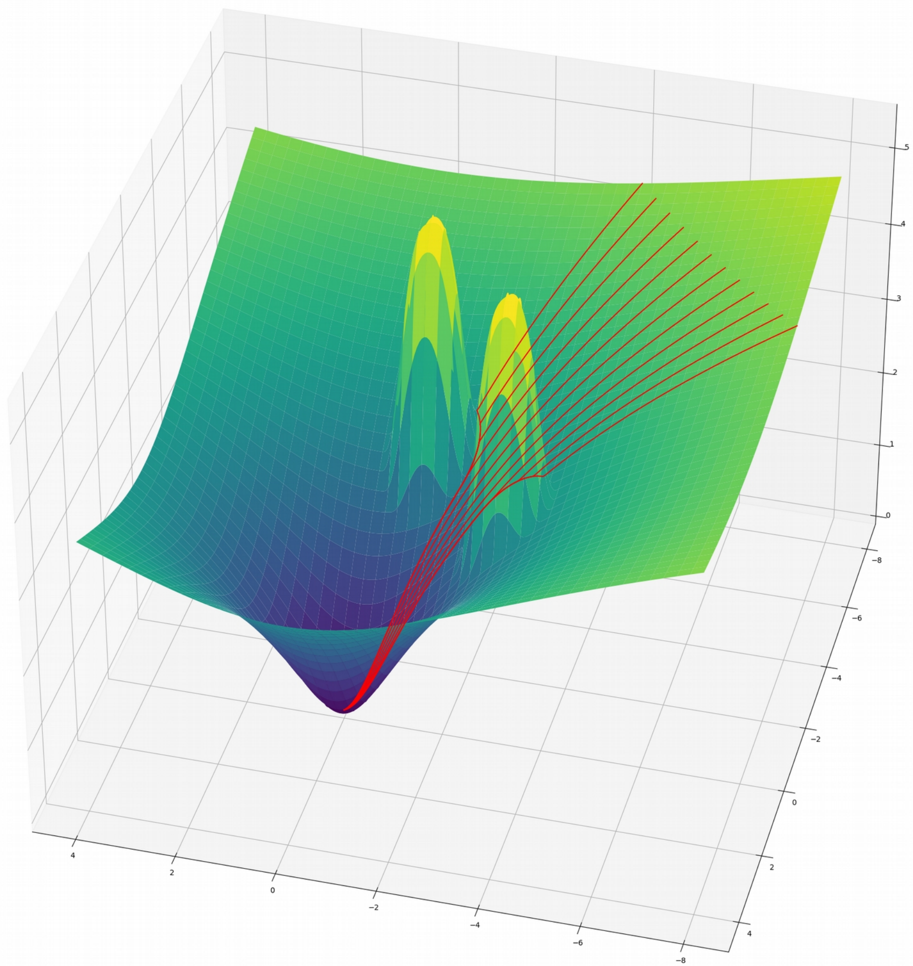
$$Cost = Collision(x_{current}, y_{current}, x_{obj}, y_{obj}) - Waypoint(x_{current}, y_{current}, x_{goal}, y_{goal})$$

- CA system activates if an object is less than a given threshold (e.g. 20 meters)

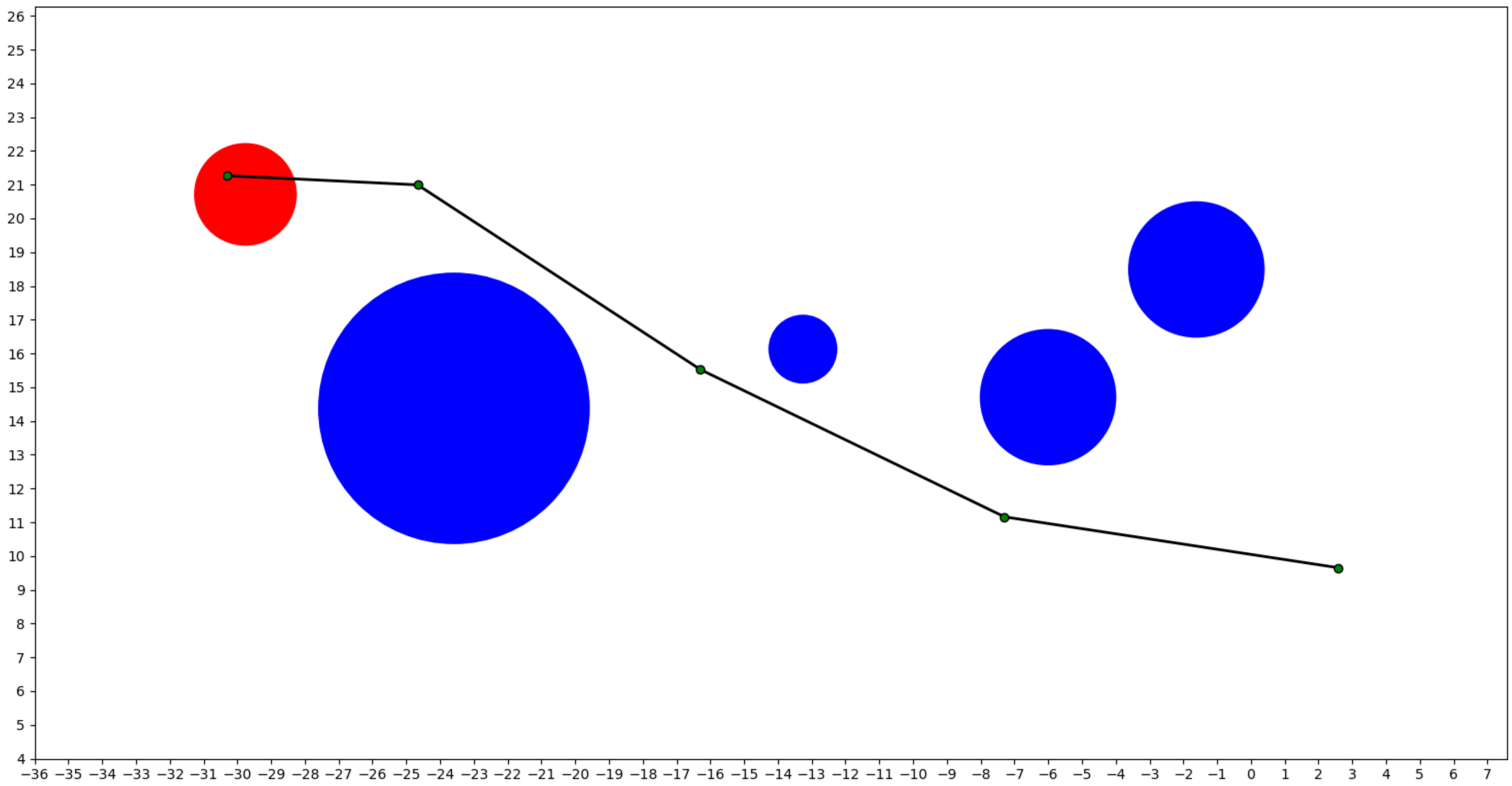
EvoLogics®

PARIS LODRON UNIVERSITÄT SALZBURG

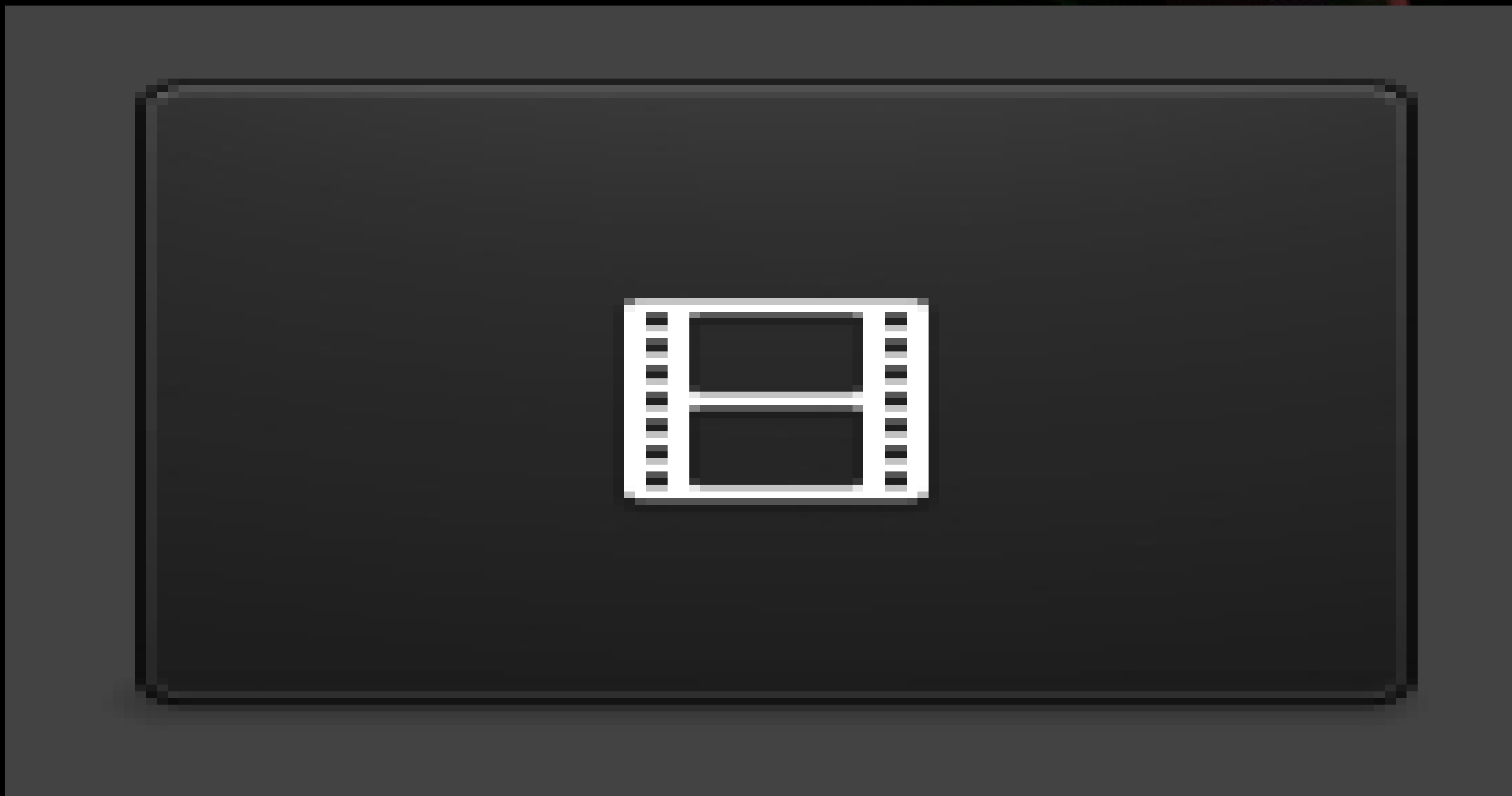
4. Cost Function



4. Discretization for static obstacles



4. Real Simulation



EvoLogics®

PARIS
LODRON
UNIVERSITÄT
SALZBURG

4. Robot

5. Conclusion and Future Research

- **A have still a lot to do!**
- Development of a dynamic model to calculate the trajectories of moving obstacles.
- Enhancement of the perception layer through the integration of error measurement methodologies and additional sensors such as LiDAR.
- Incorporation of data-driven approaches, such as Reinforcement Learning, into the MPC to manage noise.
- Integration of a robust gradient descent approach for multi-step planning.

EvoLogics®
PARIS LODRON UNIVERSITÄT SALZBURG