Drones Demystified!

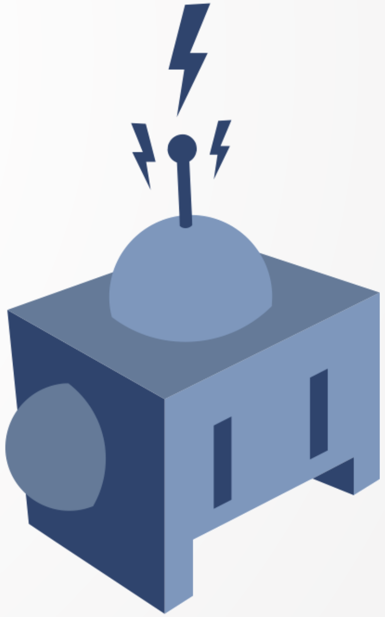
K. Alexis, C. Papachristos, Autonomous Robots Lab, University of Nevada, Reno

A. Tzes, Autonomous Robots & Intelligent Systems Lab, NYU Abu Dhabi

Drones Demystified!

Topic: Introduction to Path Planning

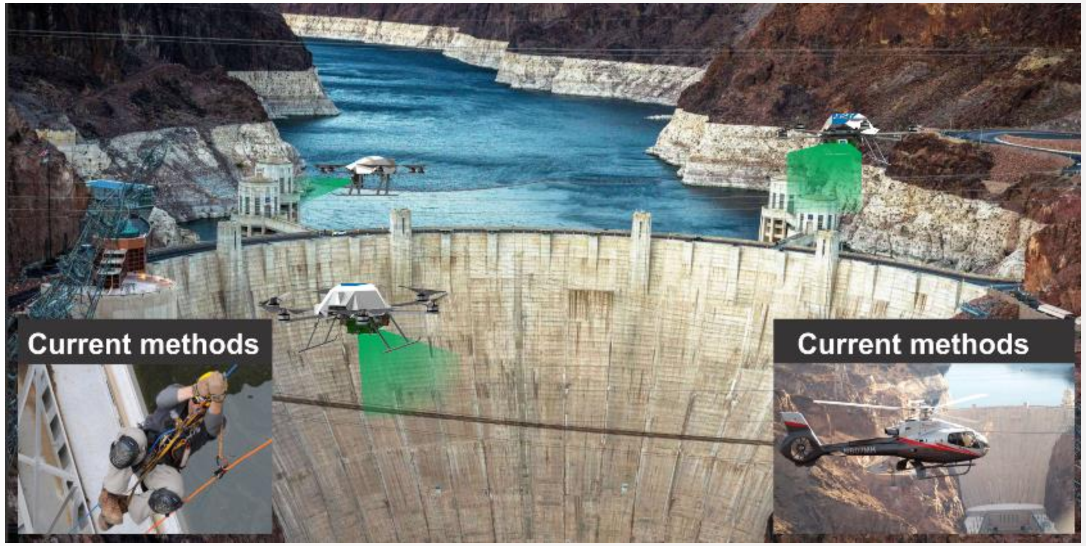
How do I plan exploration paths?



Motivation

- Autonomous Exploration and Mapping is a fundamental ability for major application domains:
  - High-resolution 3D Modeling
  - 3D Modeling and change detection
  - Semantic Classification
  - Structural health monitoring
  - …
- Three necessities:
  - Robust control
  - Accurate, robust mapping
  - Autonomous path planning

ZOOM
Current methods



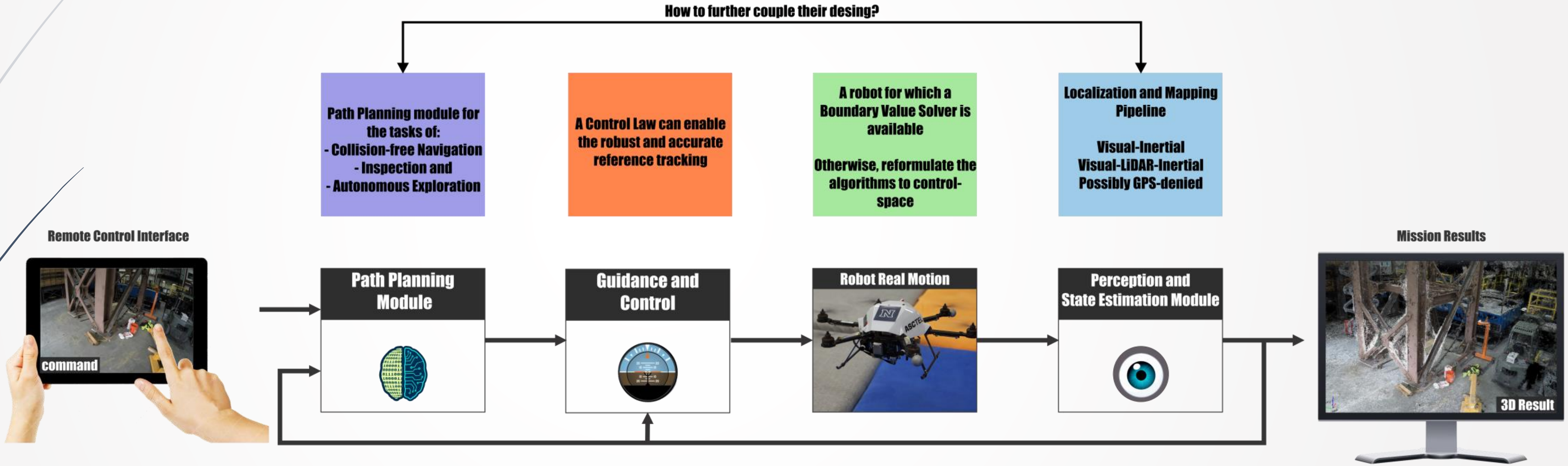
Deal with all the challenges involved

- Environment
- Robot configuration
- Sensing modalities
- Computational capabilities
- …

Multirotors (Firefly)

Fixed-Wing (AtlantikSolar)

Assumed Robot Configuration



How to further couple their desing?

Path Planning module for the tasks of:
- Collision-free Navigation
- Inspection and
- Autonomous Exploration

A Control Law can enable the robust and accurate reference tracking

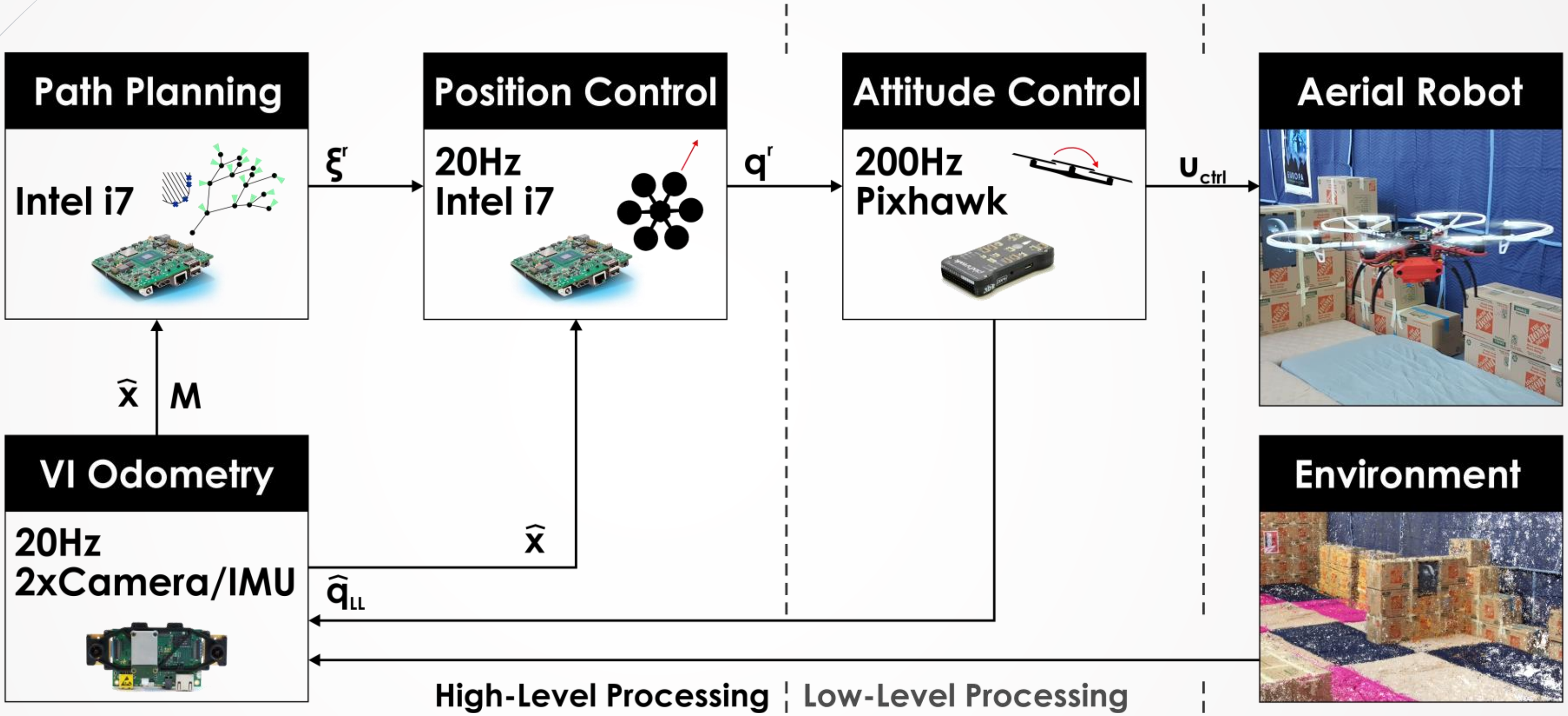
A robot for which a Boundary Value Solver is available

Otherwise, reformulate the algorithms to control-space

Localization and Mapping Pipeline

Visual-Inertial
Visual-LiDAR-Inertial
Possibly GPS-denied

Robot Configuration Specifics



Path Planning

Intel i7

ξ^r

Position Control

20Hz
Intel i7

q^r

Attitude Control

200Hz
Pixhawk

u_{ctrl}

Aerial Robot

$\hat{x}$ M

VI Odometry

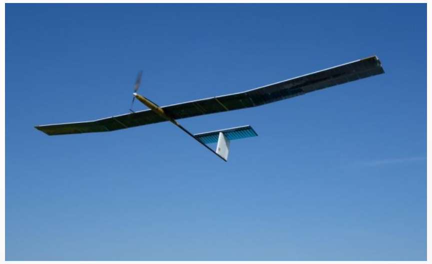
20Hz
2xCamera/IMU

$\hat{x}$

$\hat{q}_{LL}$

Environment

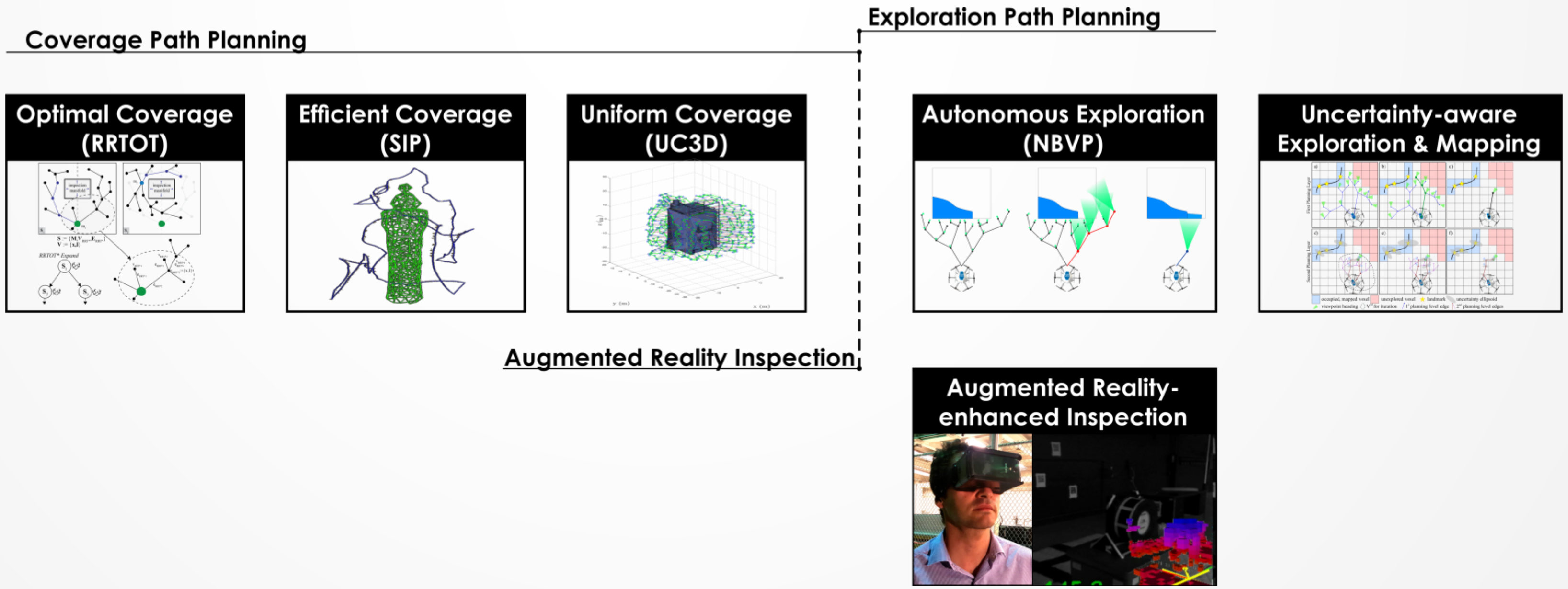
High-Level Processing | Low-Level Processing



Proposed Planning Ensemble

- **Classification:**
  - **<u>NO</u> Prior environmental knowledge**
  - **Active perception and belief-space planning?**
  - Possible human co-working?
  - Applicable to “any” robot configuration?

- **Robot evaluation:**
  - Multi-rotor UAV systems
  - Fixed-wing UAV systems


Coverage Path Planning
Exploration Path Planning
Augmented Reality Inspection
Optimal Coverage (RRTOT)
Efficient Coverage (SIP)
Uniform Coverage (UC3D)
Autonomous Exploration (NBVP)
Uncertainty-aware Exploration & Mapping
Augmented Reality-enhanced Inspection

The Exploration path planning problem

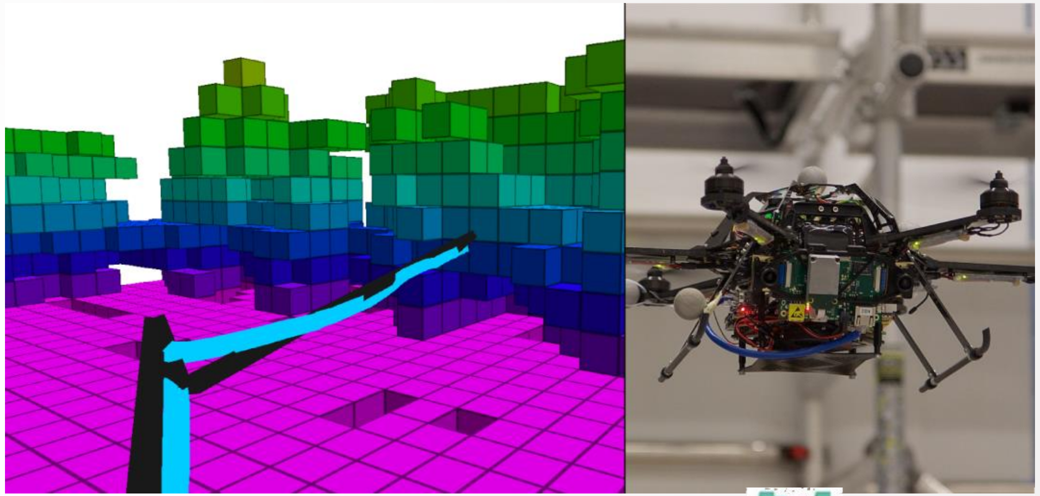
Problem Definition: Volumetric Exploration

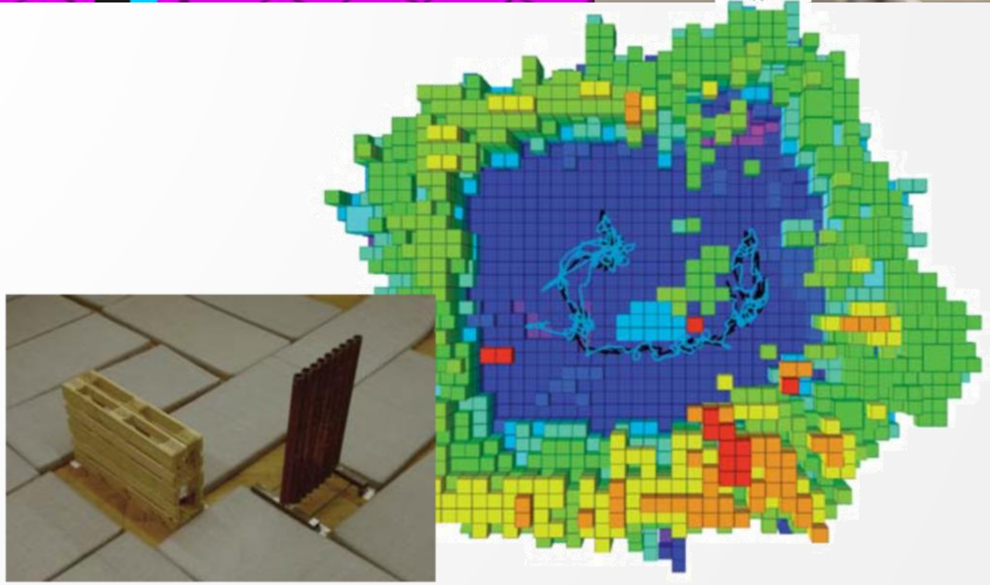
The exploration path planning problem consists in **exploring a previously unknown bounded 3D space** $V \subset \mathbb{R}^3$. This is to determine which parts of the initially unmapped space $V_{unm} = V$ are free $V_{free} \subset V$ or occupied $V_{occ} \subset V$. The operation is subject to vehicle kinematic and dynamic constraints, localization uncertainty and limitations of the employed sensor system with which the space is explored.

- As for most sensors the perception stops at surfaces, hollow spaces or narrow pockets can sometimes not be explored with a given setup. This residual space is denoted as V_{res}. The problem is considered to be fully solved when $V_{free} \cup V_{occ} = V \backslash V_{res}$.
- Due to the nature of the problem, a suitable path has to be computed online and in real-time, as free space to navigate is not known prior to its exploration.

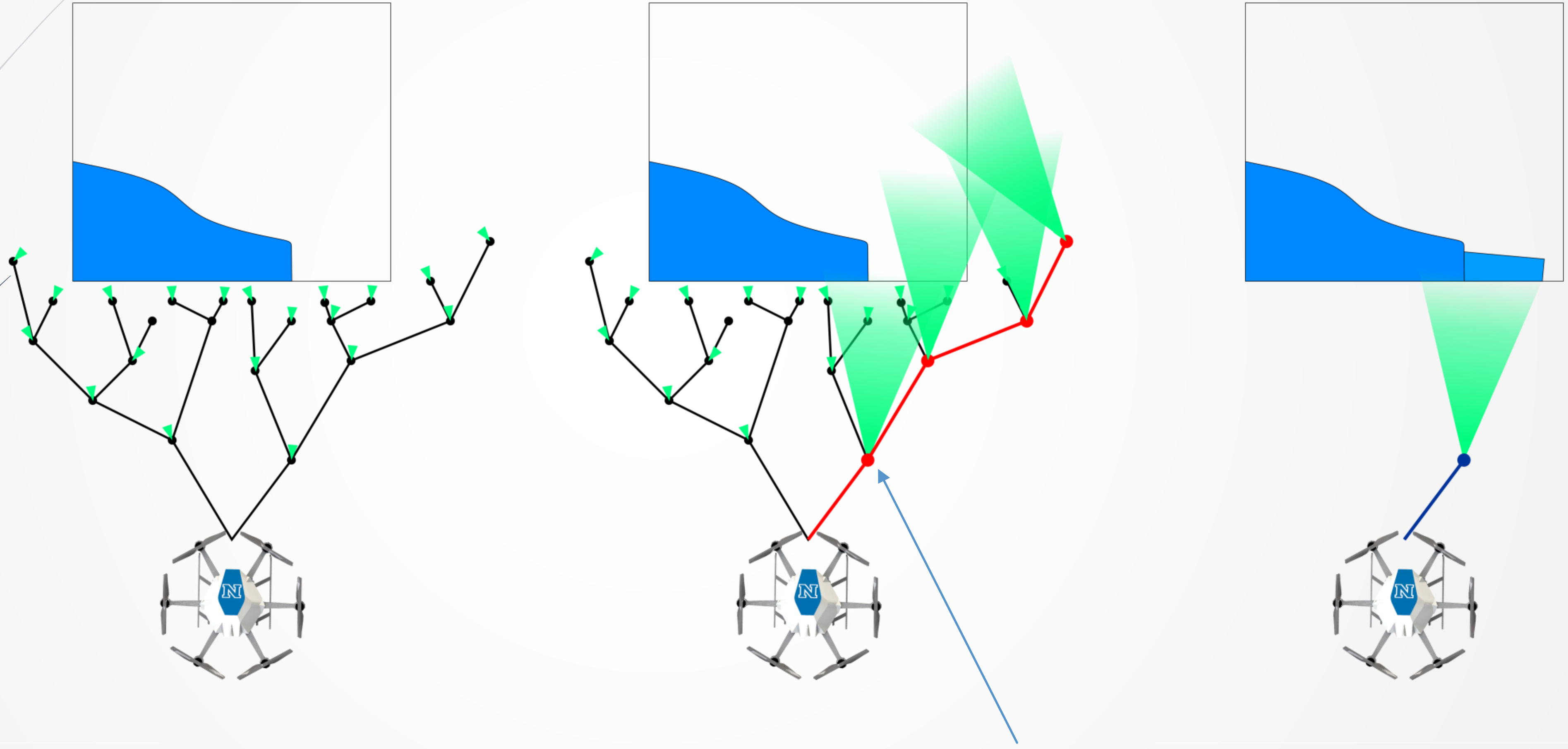
Receding Horizon Next-Best-View Exploration

- **Goal:** Fast and complete exploration of unknown environments.
- Define **sequences of viewpoints based on vertices sampled using random trees**.
- Select the path with the best sequence of best views.
- Execute only the first step of this best exploration path.
- Update the map after each iteration.
- Repeat the whole process in a receding horizon fashion.





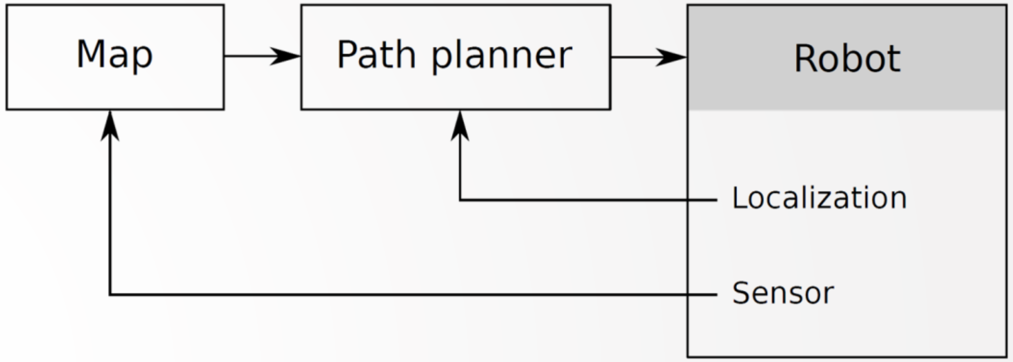
Exploration Planning (**`nbvplanner`**)



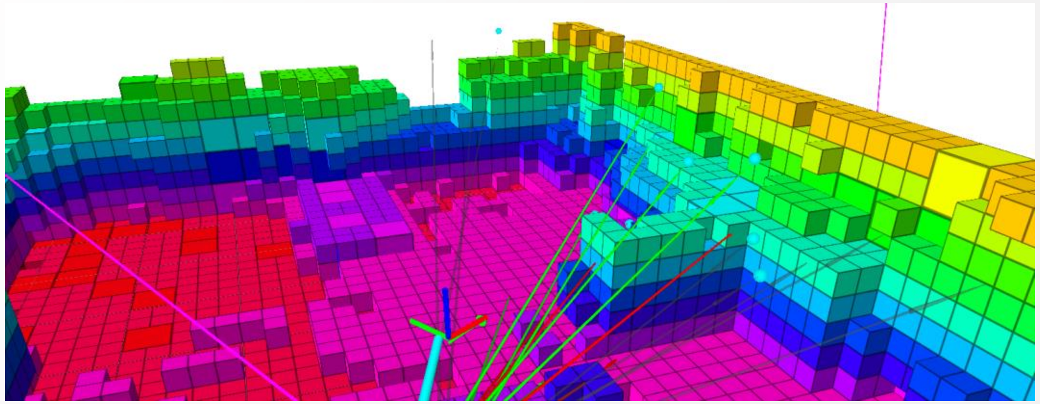
$$\mathbf{Gain}(n_k) = \mathbf{Gain}(n_{k-1}) + \mathbf{Visible}(\mathcal{M}, \xi_k) e^{-\lambda c(\sigma_{k-1}^{k})}$$

Exploration Planning (**`nbvplanner`**)

- **Environment representation:** Occupancy Map dividing space V into $m \in M$ cubical volumes (voxels) that can be marked either as free, occupied or unmapped.
- Use of the `octomap` representation to enable computationally efficient access and search.
- Paths are planned only within the free space V_{free} and collision free point-to-point navigation is inherently supported.
- At each viewpoint/configuration of the environment ξ, the amount of space that is visible is computed as $Visible(M,\xi)$



The Receding Horizon Next-Best-View Exploration Planner relies on the real-time update of the 3D map of the environment.

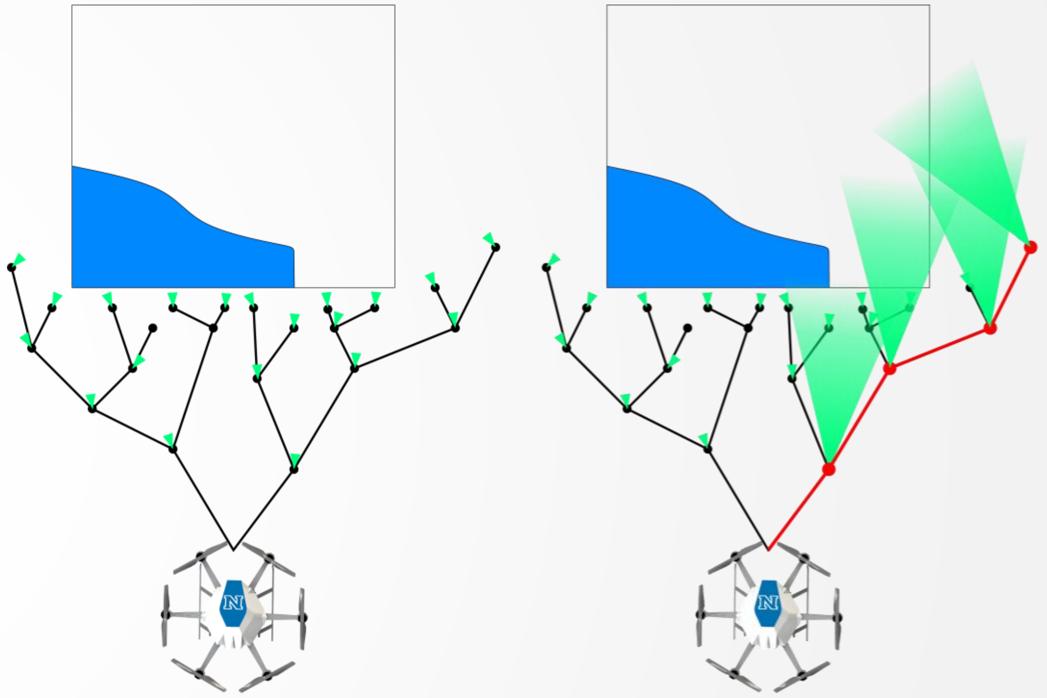


Exploration Planning (**`nbvplanner`**)

- **Tree-based exploration:** At every iteration, `nbvplanner` spans a random tree of finite depth. Each vertex of the tree is annotated regarding the collected Information Gain – a metric of how much new space is going to be explored.

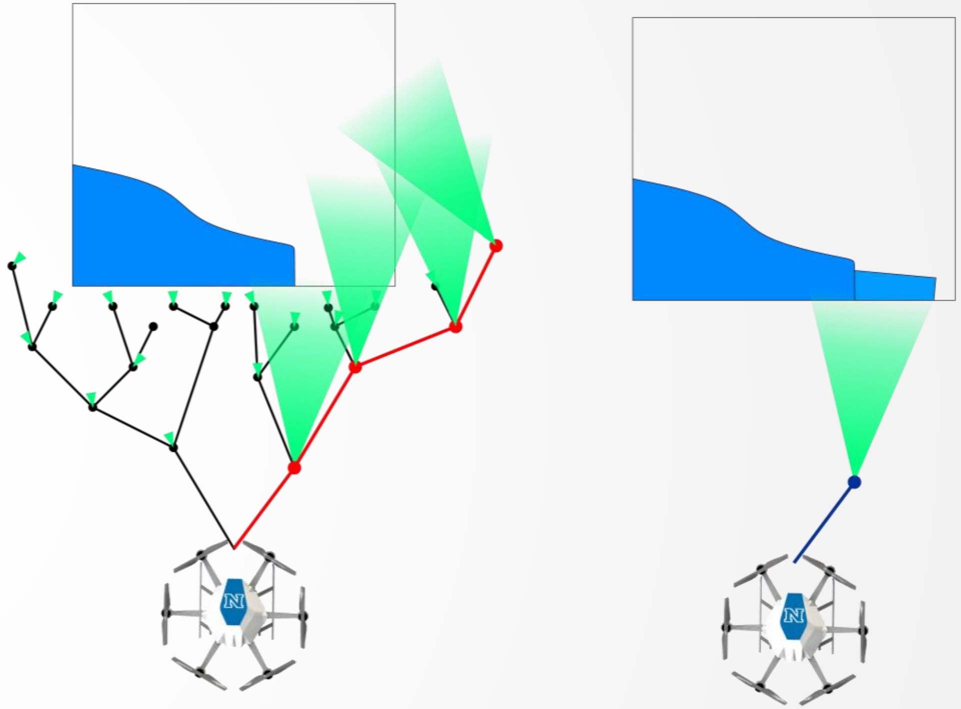
$$\mathbf{Gain}(n_k) = \mathbf{Gain}(n_{k-1}) + \mathbf{Visible}(\mathcal{M}, \xi_k) e^{-\lambda c(\sigma^k_{k-1})}$$

- Within the sampled tree, evaluation regarding the path that overall leads to the highest information gain is conducted. This corresponds to the **best path** for the given iteration. It is a sequence of next-best-views as sampled based on the vertices of the spanned random tree.



Exploration Planning (`nbvplanner`)

- **Receding Horizon:** For the extracted best path of viewpoints, only the first viewpoint is actually executed.
- The system moves to the first viewpoint of the path of best viewpoints.
- The map is subsequently updated.
- Subsequently, the whole process is repeated within the next iteration. This gives rise to a receding horizon operation.



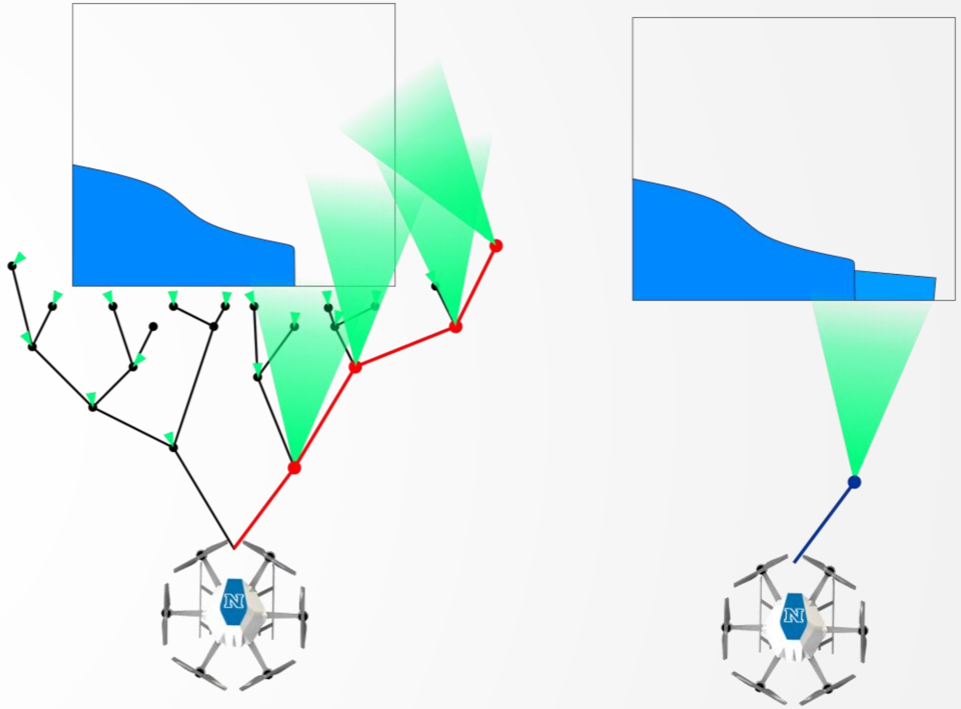
Exploration Planning (**nbvplanner**) Algorithm

nbvplanner Iterative Step

```
$\xi_0$ ←current vehicle configuration
Initialize $\boldsymbol{T}$ with $\xi_0$ and, unless first planner call, also previous best branch
$g_{best} \leftarrow 0$                          // Set best gain to zero
$n_{best} \leftarrow n_0(\xi_0)$                 // Set best node to root
$N_T$ ←Number of nodes in $\boldsymbol{T}$
while $N_T < N_{max}$ or $g_{best} == 0$ do
    Incrementally build $\boldsymbol{T}$ by adding $n_{new}(\xi_{new})$
    $N_T \leftarrow N_T + 1$
    if $Gain(n_{new}) > g_{best}$ then
        $n_{best} \leftarrow n_{new}$
        $g_{best} \leftarrow Gain(n_{new})$
    if $N_T > N_{TOT}$ then
        Terminate exploration
$\sigma \leftarrow \boldsymbol{ExtractBestPathSegment}(n_{best})$
Delete $\boldsymbol{T}$
return $\sigma$
```

Exploration Planning (**`nbvplanner`**) Remarks

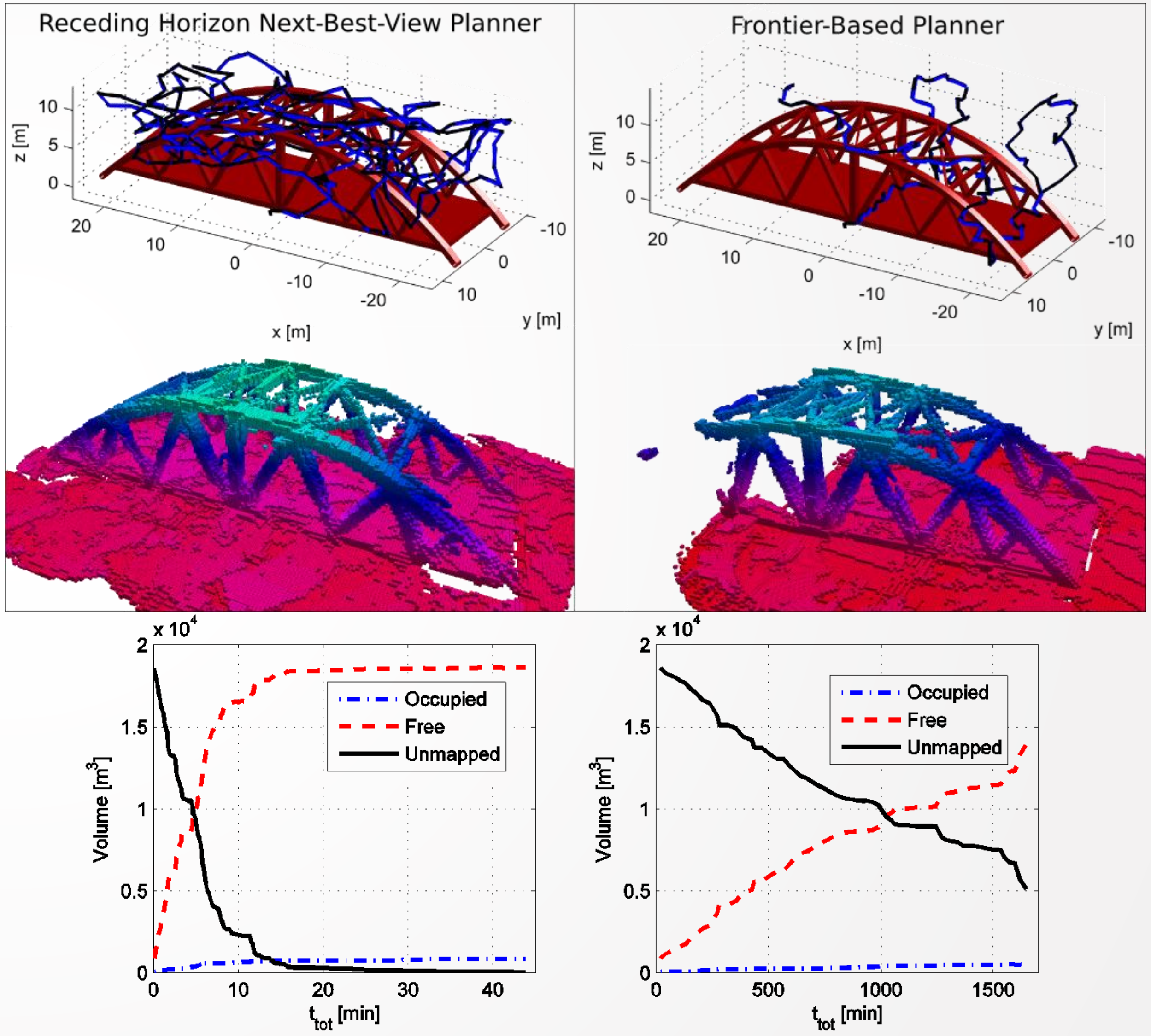
- **Inherently Collision-free:** As all paths of `nbvplanner` are selected along branches within RRT-based spanned trees, all paths are inherently collision-free.
- **Computational Cost:** nbvplanner has a thin structure and most of the computational cost is related with collision-checking functionalities. The formula that expresses the complexity of the algorithm takes the form:



$$\mathcal{O}(N_{\mathbb{T}}\log(N_{\mathbb{T}}) + N_{\mathbb{T}}/r^3\log(V/r^3) + N_{\mathbb{T}}(d_{\max}^{\text{planner}}/r)^4\log(V/r^3))$$

`nbvplanner` Evaluation (Simulation)

- **Simulation-based evaluation:** Explore a bridge.
- Comparison with Frontier-based exploration.



Extension to Surface Inspection

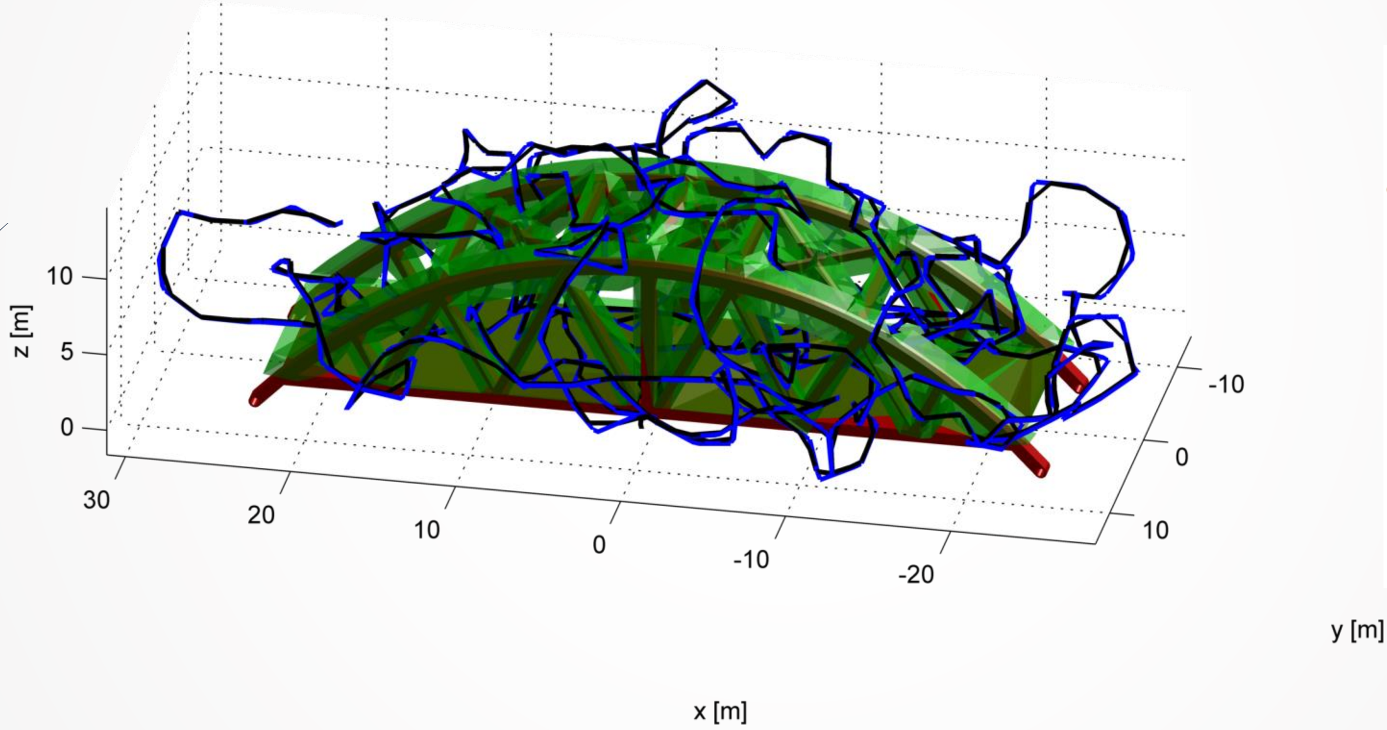
Problem Definition: Surface Inspection

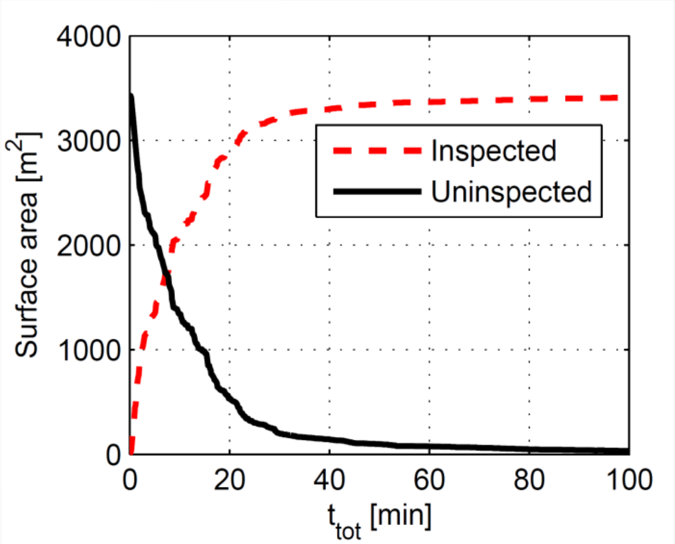
Given a surface S, find a collision free path σ starting at an initial configuration $\xi_{init} \in \Xi$ that leads to the inspection of the part S_{insp}, when being executed, such that there does not exist any collision free configuration from which any piece of $S \backslash S_{insp}$ could be inspected. Thus, $S_{insp} = S \backslash S_{res}$.

- Let $\overline{V}_s \subseteq \Xi$ be the set of all configurations from which the surface piece $s \subseteq S$ can be inspected. Then the residual surface is given as $S_{res} = \cup_{s \in S} (s | \overline{V}_s = 0)$

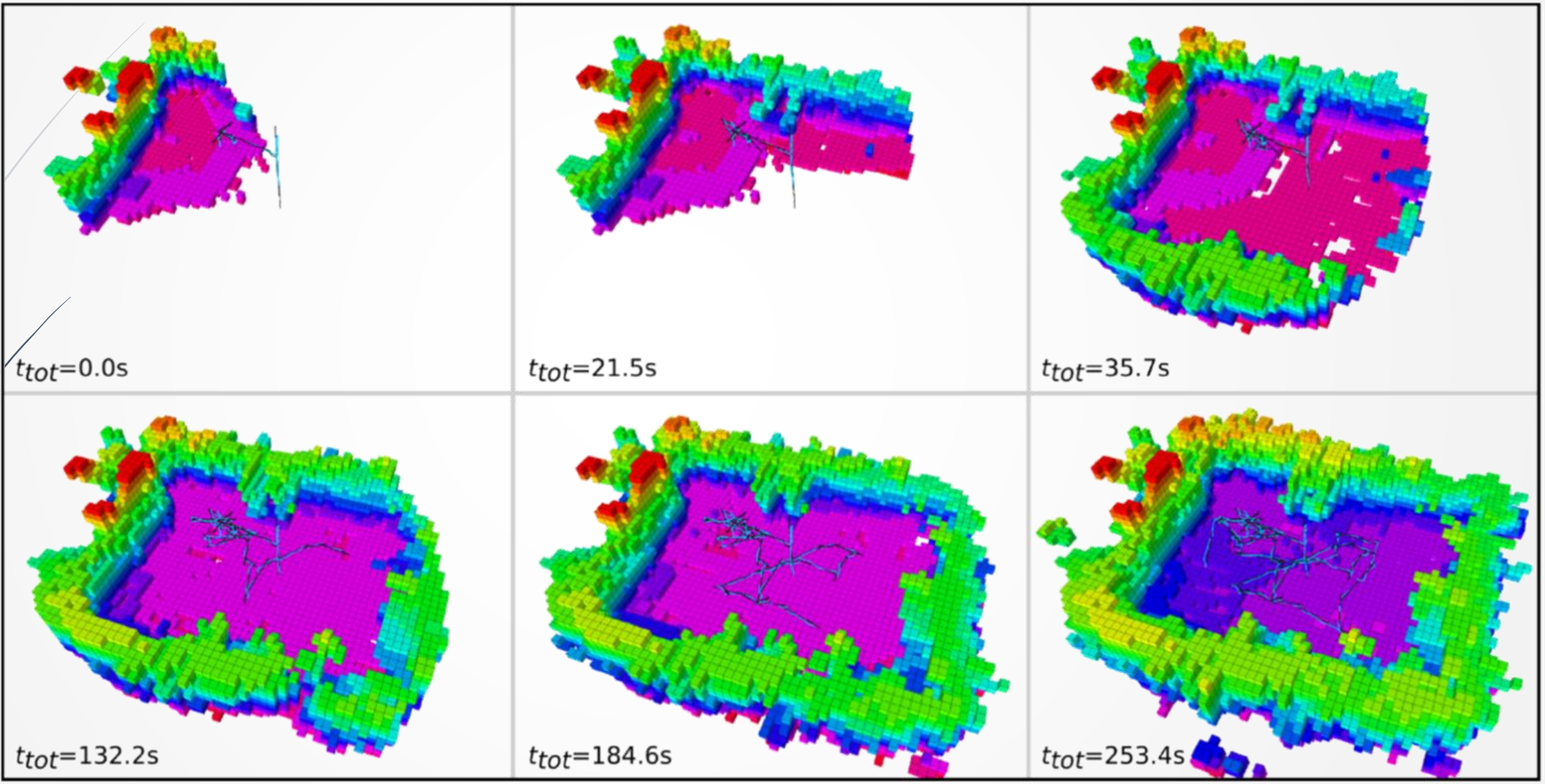
`nbvplanner` Evaluation (Simulation)

- **Extension to surface inspection:** The robot identifies trajectories that locally ensure maximum information gain regarding surface coverage.



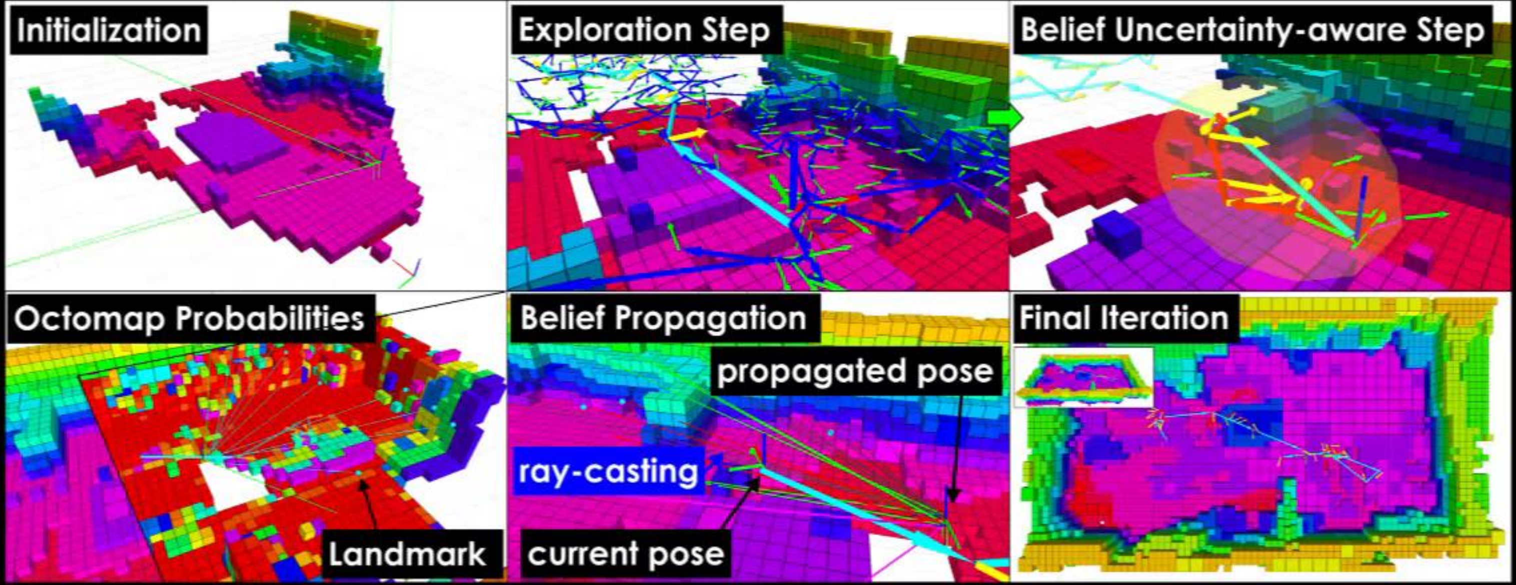


`nbvplanner` Evaluation (Experiment)



Uncertainty-aware Receding Horizon Exploration and Mapping using Aerial Robots

Christos Papachristos, Shehryar Khattak, Kostas Alexis

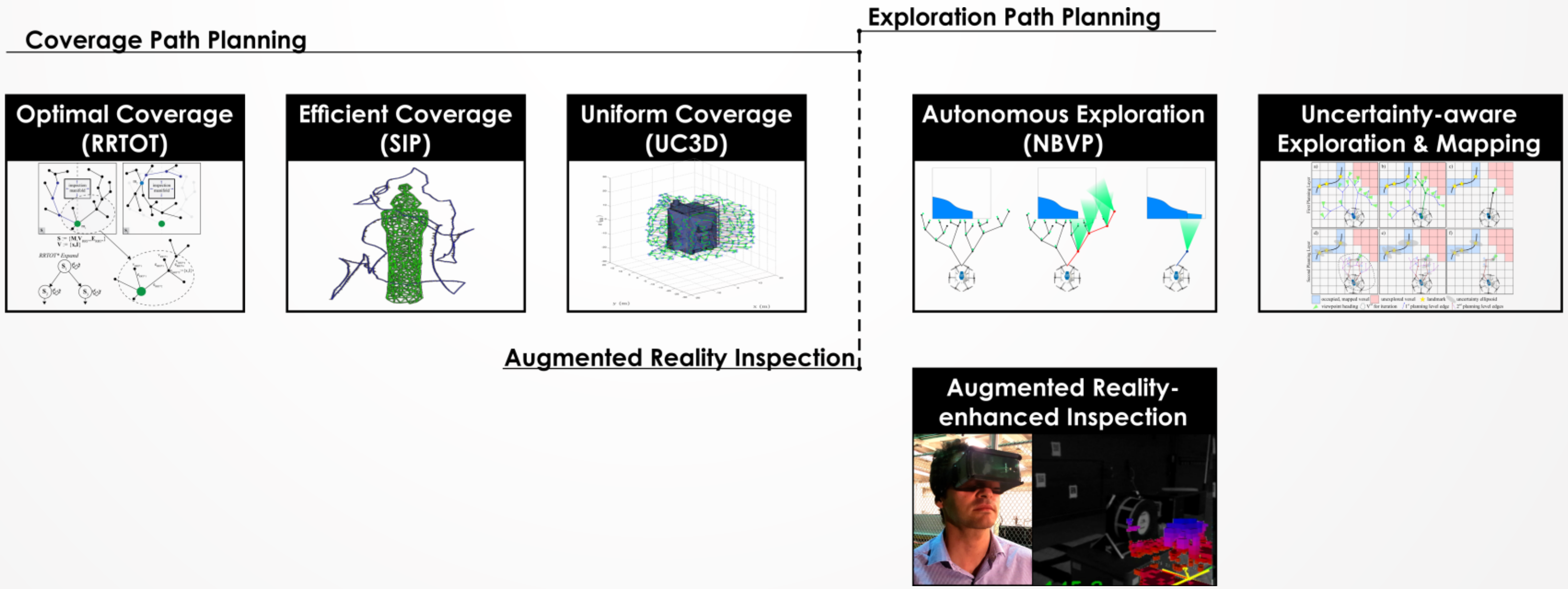


AUTONOMOUS
ROBOTS
LAB

Proposed Planning Ensemble

- **Classification:**
  - **<u>WITH</u> Prior environmental knowledge**
  - **Active perception and belief-space planning**
  - **Possible human co-working?**
  - **Applicable to "any" robot configuration?**

- **Robot evaluation:**
  - **Multi-rotor UAV systems**
  - **Fixed-wing UAV systems**


Coverage Path Planning
Exploration Path Planning
Optimal Coverage (RRTOT)
Efficient Coverage (SIP)
Uniform Coverage (UC3D)
Autonomous Exploration (NBVP)
Uncertainty-aware Exploration & Mapping
Augmented Reality Inspection
Augmented Reality-enhanced Inspection

The inspection path planning problem

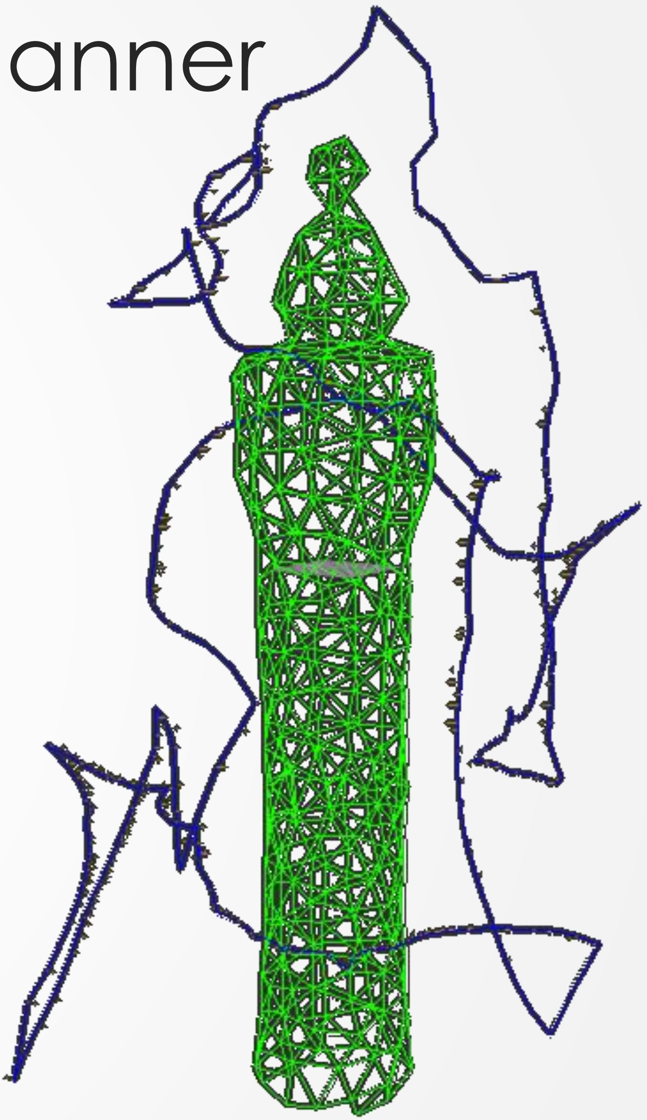
Problem Definition: Full Coverage Surface Inspection

$$\frac{dx}{dt} = f(x,u), x(0) = x_{init}$$

- Where is x the state, u is the control. As well as a sensor model of field of view $FOV = [F_H, F_V]$ and maximum range d.
- Given an obstacle set X_{obs}, and a inspection manifold S_I, the objective of the motion planning problem is to find, if it exists, a path $\boldsymbol{r}$ that provides the viewpoints to the sensor such that the whole surface of S_I is perceived, the vehicle dynamics are respected and the cost of the path (distance, time, etc) is minimized.

Motivation of the Proposed Planner

- **Can we find a "good enough" solution but compute very fast?**

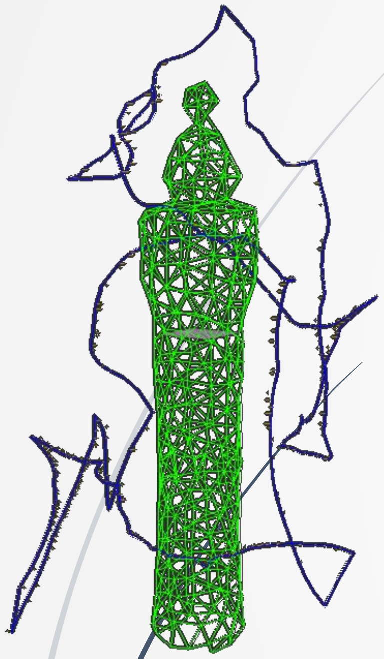


Basic Concepts of the Inspection Planner

- **Main classes of existing 3D methods:**
  - Separated Approach (AGP + TSP or Control)
    - Prone to be suboptimal
    - In specific cases lead to infeasible paths (nonholonomic vehicles)
  - First attempts for optimal solutions via a unified cycle
    - In specific cases can lead to the optimal solution
    - Very high CPU and Memory Requirements & Time

- **Structural Inspection Planner (`SIP`):**
  - Driven by the idea that *with a continuously sensing sensor, the number of viewpoints is not necessarily important but **mostly their configuration in space***.
  - Not a minimal set of viewpoints but a set of full coverage viewpoints positioned such that the overall path gets minimized.
  - 2-step paradigm with viewpoint alternation
  - Guaranteed feasible paths for both holonomic and nonholonomic vehicles

Structural Inspection Planner (**SIP**)



First solution

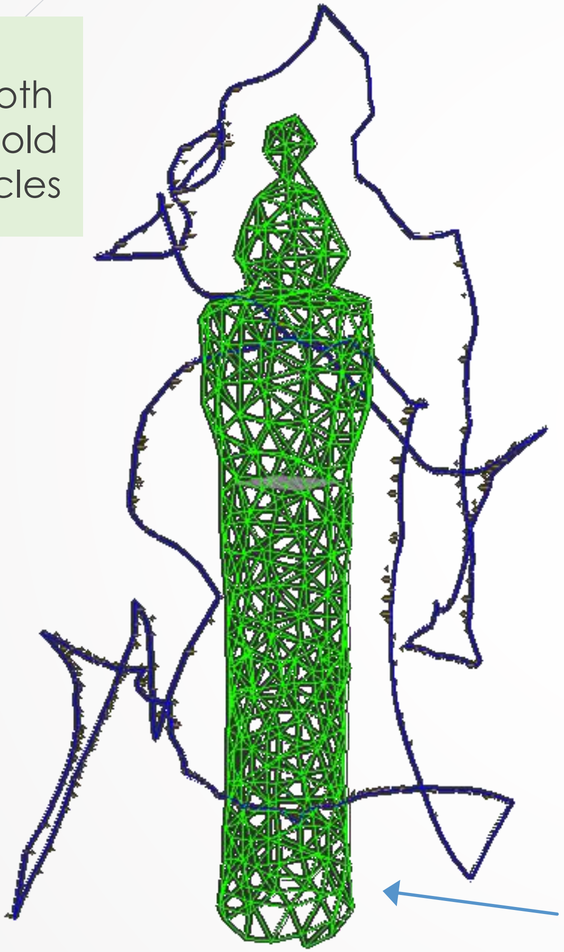
```
Load the mesh model
$k = 0$
Sample Initial Viewpoint Configurations (Viewpoint Sampler)
Find Collision-free paths for all viewpoint combinations (BVS, RRT*)
Populate the Cost Matrix and Solve the Traveling Salesman Problem (LKH)
while running
    Re-sample Viewpoint Configurations (Viewpoint Sampler)
    Re-compute the Collision-free paths (BVS, RRT*)
    Re-populate the Cost Matrix and solve the new Traveling Salesman Problem to update the best inspection tour (LKH)
    $k = k + 1$
Return $BestTour, CostBestTour$
```

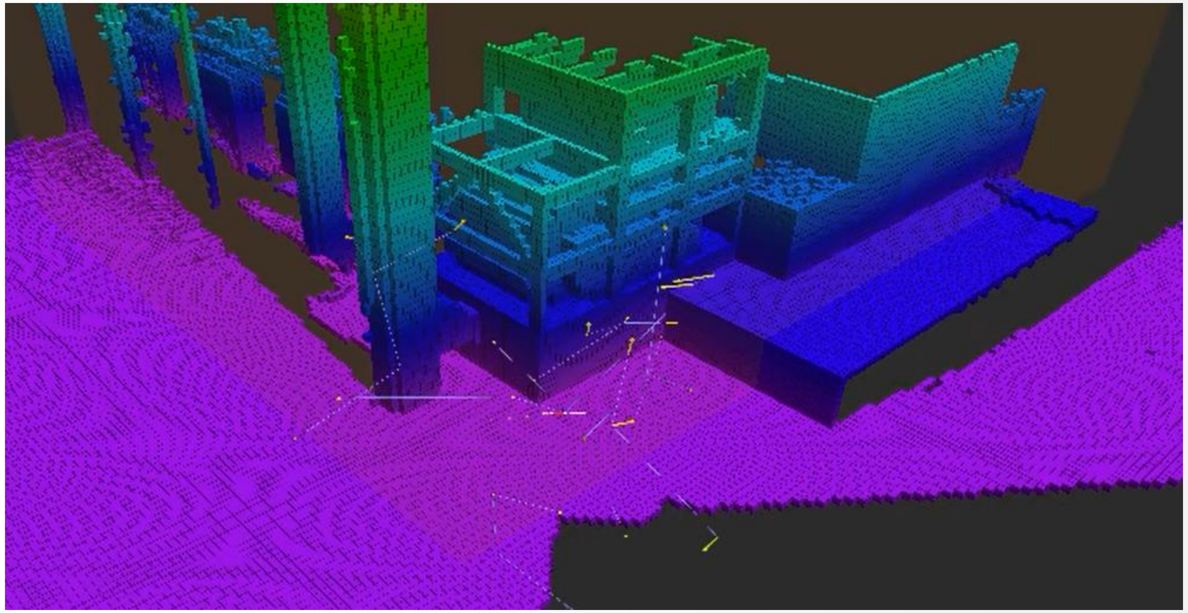
Available Time

Optimized solutions

SIP Supported World Representations

Same type of representation for both the inspection manifold as well as any obstacles





Octomap [possibly enlarged voxels]

Meshes [possibly downsampled]

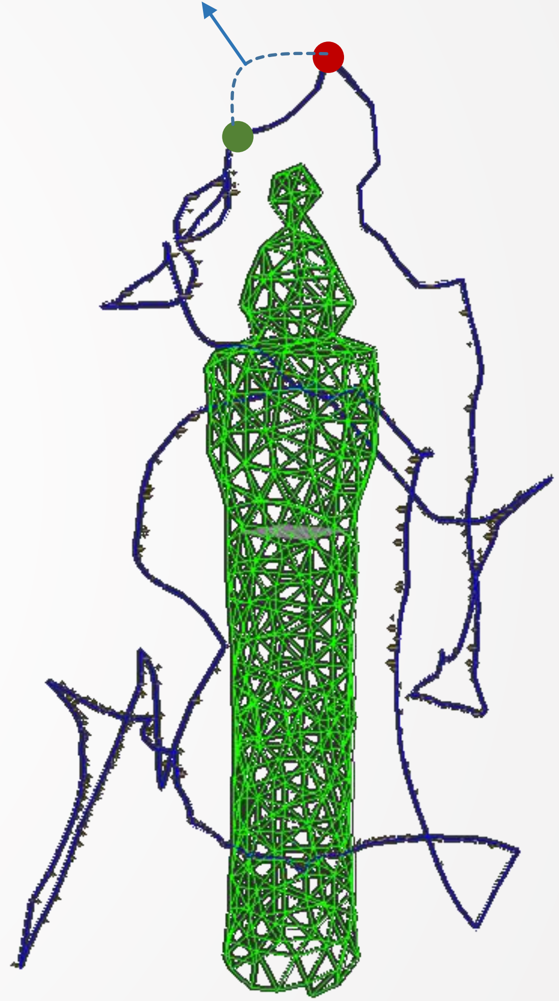
Supported in the open-sourced **SIP**

Sampling-based and Collision-checking implemented

SIP Point-to-Point Paths

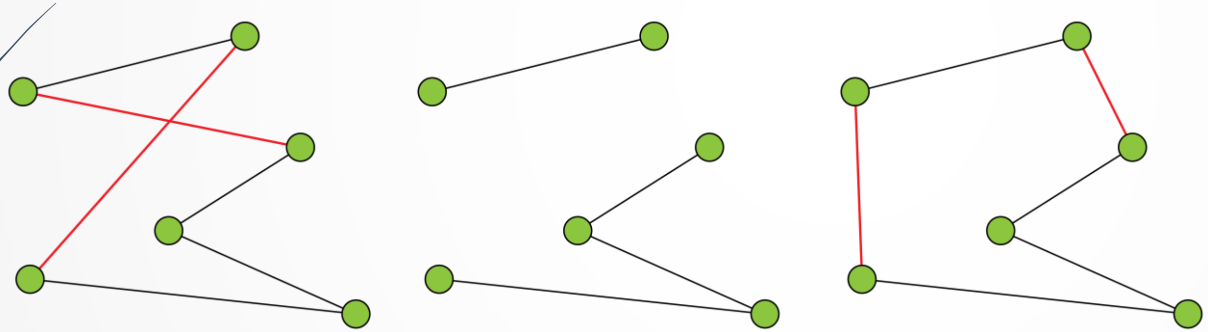
- State-Space Sampling – *extension to Control-Space sampling possible*
- Employ **B**oundary **V**alue **S**olvers for
  - Holonomic (with Yaw-rate constraints) or
  - Nonholonomic Aerial Robots *(fixed-wing UAVs – 2.5D approx., Dubins Airplane approx.)*
- Derive Collision-free paths that connect all viewpoint configurations by invoking ***RRT****
- Assemble the Traveling Salesman Problem Cost Matrix using the path execution times t_{ex}

- **Compute RRT* Path**
- **Extract the t_{ex} of the RRT* Path**
- **Populate the Cost Matrix**



SIP TSP Solution

- Solve the (possibly asymmetric) TSP problem using the Lin-Kernighan-Helsgaun heuristic
- ***Extract the Optimized Inspection Tour***

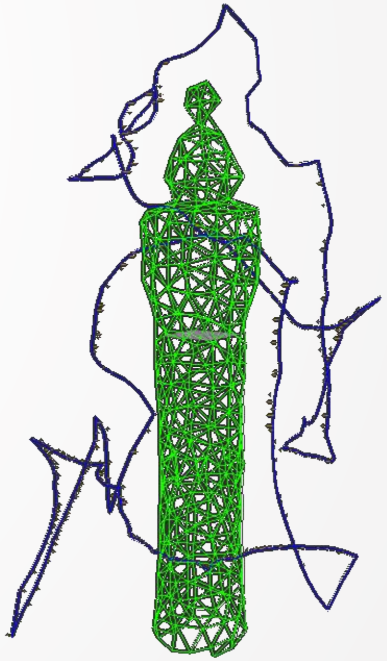


Identify 2 edges

Remove them

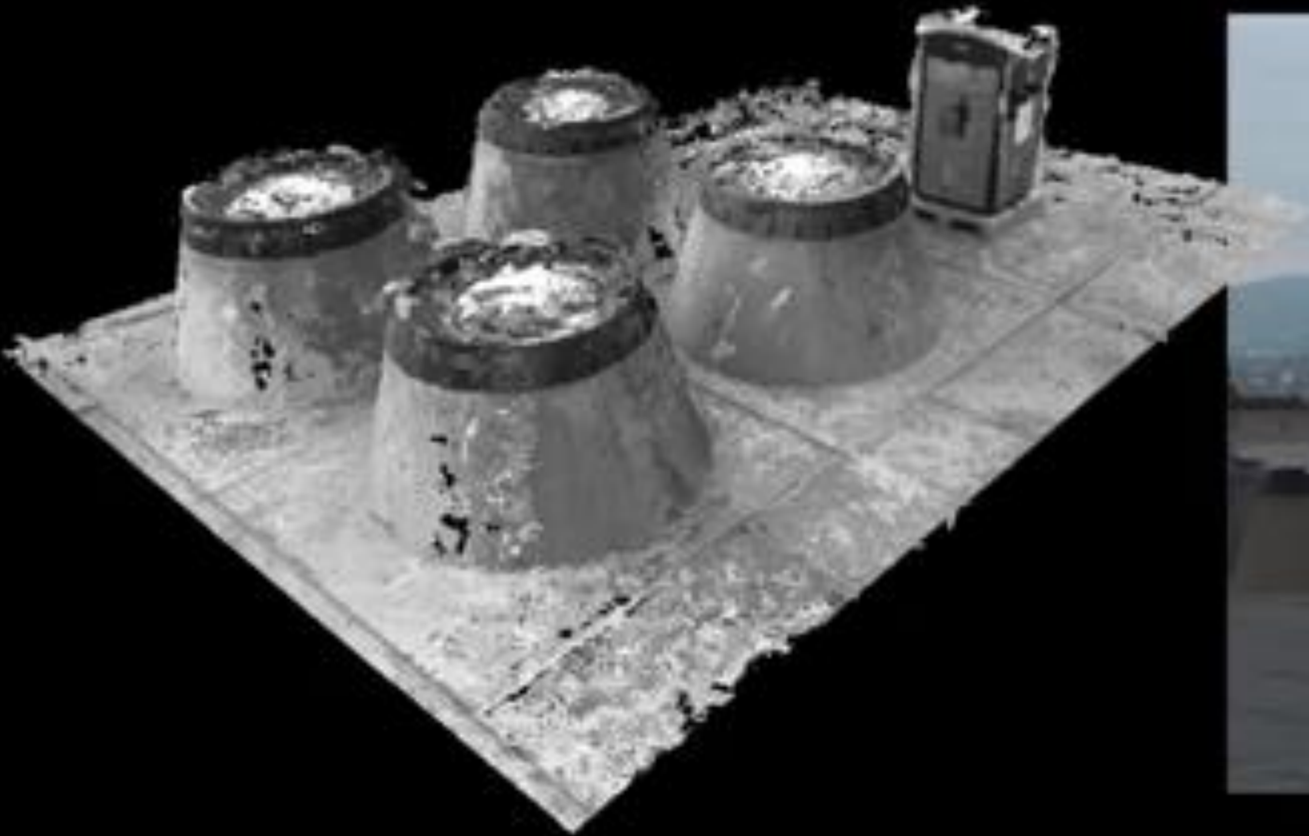
Insert two new tour-completing edges

$O(N^{2.2})$, N the number of viewpoints



Three-dimensional Coverage Path Planning via Viewpoint Resampling and Tour Optimization using Aerial Robots

A. Bircher, K. Alexis, M. Kamel, M. Burri, P. Oettershagen, S. Omari, T. Mantel, R. Siegwart



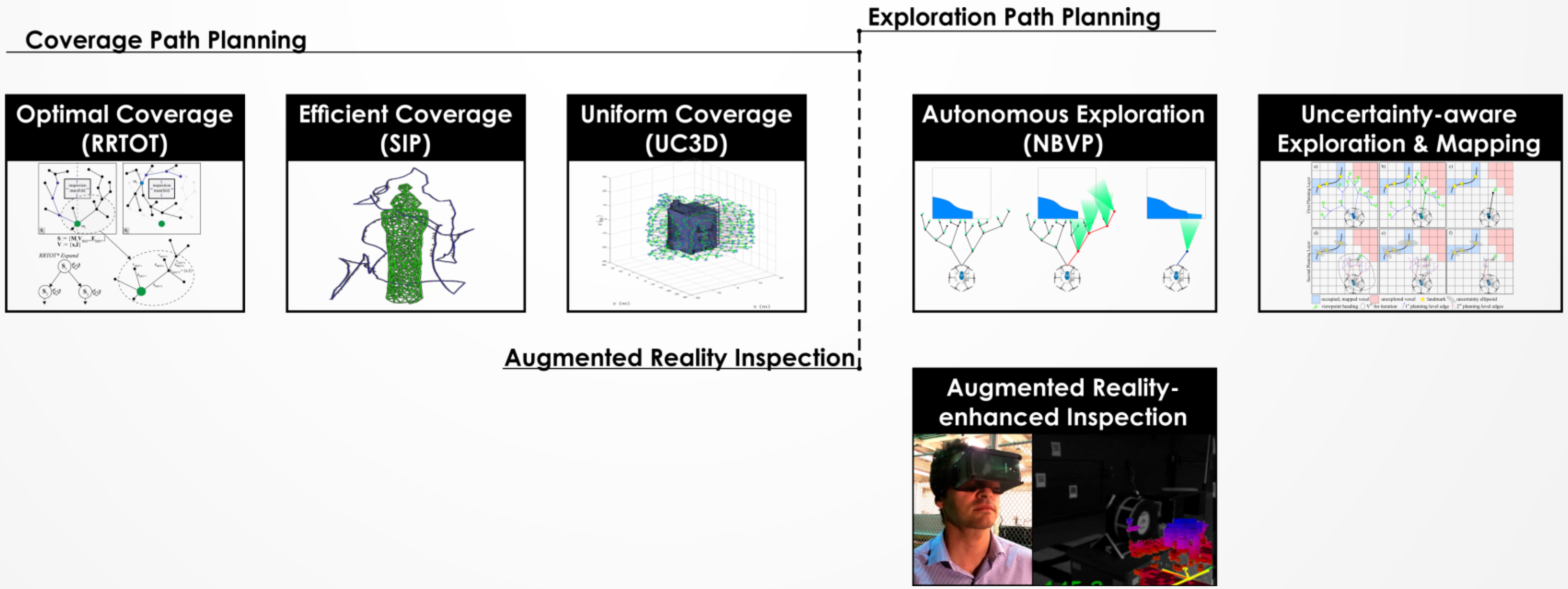
Structural Inspection Path Planning via Iterative Viewpoint Resampling with Application to Aerial Robotics

Andreas Bircher, Kostas Alexis, Michael Burri, Phlipp Oettershagen, Sammy Omari, Thomas Mantel and Roland Siegwart

Proposed Planning Ensemble

- **Classification:**
  - **WITH Partial Prior environmental knowledge**
  - **Active perception and belief-space planning**
  - **Possible human co-working?**
  - **Applicable to "any" robot configuration?**

- **Robot evaluation:**
  - **Multi-rotor UAV systems**
  - **Fixed-wing UAV systems**



Coverage Path Planning

Exploration Path Planning

Augmented Reality Inspection

Optimal Coverage (RRTOT)

Efficient Coverage (SIP)

Uniform Coverage (UC3D)

Autonomous Exploration (NBVP)

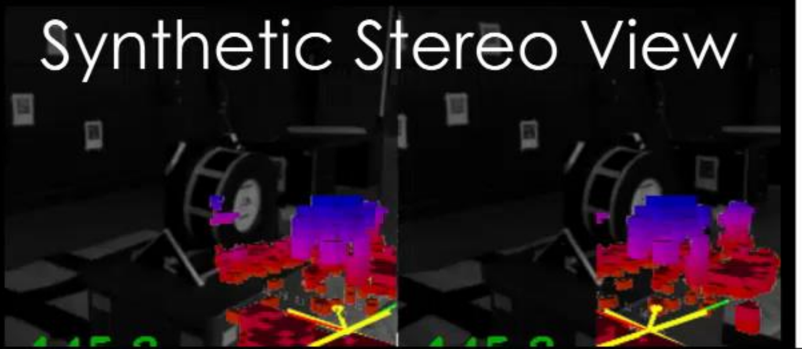
Uncertainty-aware Exploration & Mapping

Augmented Reality-enhanced Inspection

Augmented Reality-enhanced Structural Inspection using Aerial Robots

Christos Papachristos, Kostas Alexis

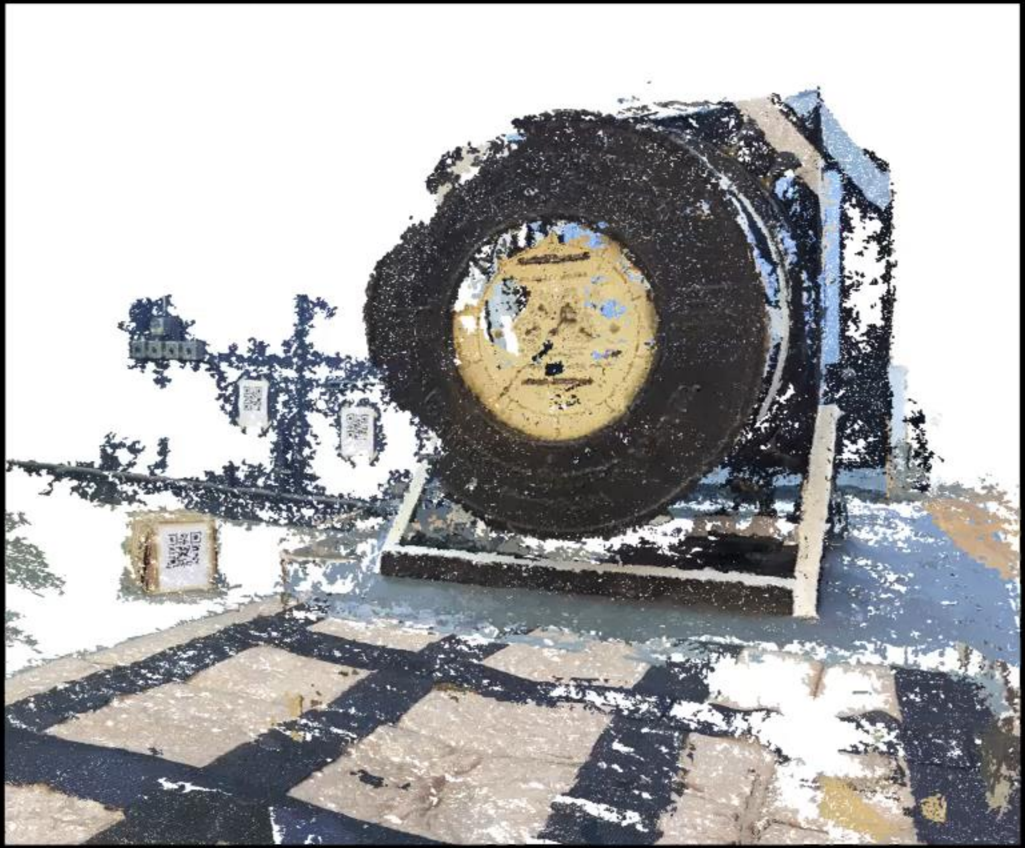
Augmented Reality



Synthetic Stereo View

Autonomous Flight

Accurate 3D Reconstruction



Autonomous Robots Lab, University of Nevada, Reno

Open Source contributions

- **Open Source Code:**
  - Structural Inspection Planner:
    - https://github.com/ethz-asl/StructuralInspectionPlanner
  - Next-Best-View Planner:
    - https://github.com/ethz-asl/nbvplanner
  - Receding Horizon Exploration and Mapping Planner:
    - https://github.com/unr-arl/rhem_planner
- **Associated Datasets:**
  - Structural Inspection Planner:
    - https://github.com/ethz-asl/StructuralInspectionPlanner/wiki/Example-Results
    - http://changedetectiondataset.wikispaces.com/
  - Next-Best-View Planner:
    - https://github.com/ethz-asl/nbvplanner/wiki/Example-Results
  - Receding Horizon Exploration and Mapping Planner:
    - https://github.com/unr-arl/icra-2017-datasets
  - Solar-powered UAV Sensing & Mapping:
    - http://projects.asl.ethz.ch/datasets/doku.php?id=fsr2015
  - Change Detection Dataset
    - http://changedetectiondataset.wikispaces.com/

Code Examples and Tasks

- https://github.com/unr-arl/drones_demystified/tree/master/matlab/path-planning/rrt
- https://github.com/unr-arl/drones_demystified/tree/master/ROS/path-planning/structural-inspection
- https://github.com/unr-arl/drones_demystified/tree/master/ROS/path-planning/autonomous-exploration

Find out more

- http://www.autonomousrobotslab.com/holonomic-vehicle-bvs.html
- http://www.autonomousrobotslab.com/dubins-airplane.html
- http://www.autonomousrobotslab.com/collision-free-navigation.html
- http://www.autonomousrobotslab.com/structural-inspection-path-planning.html

- http://ompl.kavrakilab.org/
- http://moveit.ros.org/
- http://planning.cs.uiuc.edu/

- http://www.autonomousrobotslab.com/literature-and-links1.html

Thank you!

Please ask your question!