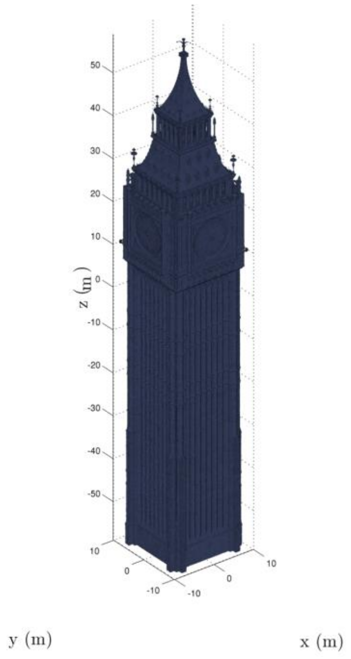
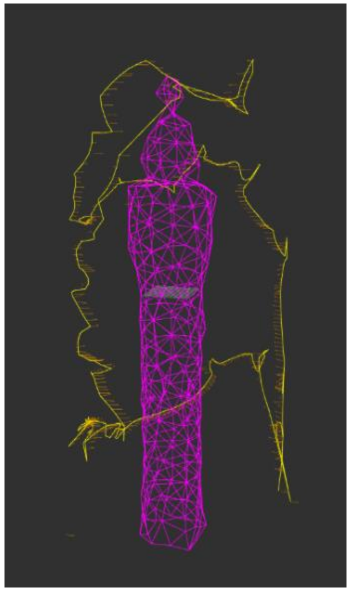
Structural Inspection Path Planning via Iterative Viewpoint Resampling with Application to Aerial Robot



Sungwook Jung (정성욱)

2017.04.27
CS686
Presentation #1



Bircher, Andreas, et al. "Structural inspection path planning via iterative viewpoint resampling with application to aerial robotics," *IEEE International Conference on Robotics and Automation (ICRA),* pp. 6423-6430, Seattle, USA, 2015.

Introduction

- In the fields of inspection operations,
- autonomous complete coverage 3D structural path planning is required.

- A robot needs quick algorithms that result in full coverage of the structure while respecting any sensor limitations and motion constraints.

- A novel fast algorithm that provides efficient solutions to the problem of inspection path planning for complex 3D structures is proposed

Problem description

- **Conventional 3D methods:**
  - **Two-step optimization**
    1. **Compute the minimal set of viewpoints that cover whole structure (solving Art Gallery Problem (AGP[1])).**
    2. **Compute the shortest connecting tour over these viewpoints (Travelling Salesman Problem (TSP[2])).**

  - **Large cost of computing efficiency**
  - **They are prone to be suboptimal due to the two-step separation of the problem.**
  - **In specific cases they can lead to unfeasible solutions/paths (e.g. in the case of non-holonomic vehicles)**

[1] J. O'rourke, Art gallery theorems and algorithms. Oxford University Press Oxford, 1987, vol. 57.

[2] G. Dantzig, R. Fulkerson, and S. Johnson, "Solution of a large- scale traveling-salesman problem," Journal of the operations research society of America, vol. 2, no. 4, pp. 393–410, 1954.

Contributions

- Not minimizing the number of viewpoints, it samples them such that connecting path is short while ensuring full coverage
- A two step optimization paradigm to find good viewpoints that together provide full coverage and a connecting path that has low cost
  - First: In every iteration, each viewpoints is chosen to reduce the cost-to-travel between itself and its neighbors
  - Second: the optimally connecting tour is recomputed

Methodology: psedo-code

Algorithm 1 Inspection path planner

First solution

```
1. Load the mesh model
2. k = 0
3. Sample Initial Viewpoint Configurations (Viewpoint Sampler)
4. Find a Collision-free path for all possible viewpoint combinations (boundary value solver (BVS), RRT*)
5. Populate the Cost Matrix and Solve the Traveling Salesman Problem (Lin-Kernighan-Helsgaun Heuristic (LKH))
```

Optimized solution

```
6. while running
    1. Re-sample Viewpoint Configurations (Viewpoint Sampler)
    2. Re-compute the Collision-free paths (BVS, RRT*)
    3. Re-populate the Cost Matrix and solve the new Traveling Salesman Problem to update the current best inspection tour (LKH)
    4. k = k + 1
7. end while
8. Return BestTour, CostBestTour
```

Methodology: path computation and cost estimation

- To find the best tour among viewpoints, TSP solver requires a cost matrix of all pairs of viewpoints
- Path generation and cost estimation is done by either
  - BVS - directly connect the two viewpoints
  - BVS+RRT* - due to obstacles, connection is not feasible

- The cost of a path segment corresponds to the execution time t_{ex}
  - $t_{ex} = \max\left(d/v_{max}, \|\psi_1 - \psi_0\|/\dot{\psi}_{max}\right)$

Where d is the Euclidean distance, translation speed limit is v_{max}, rotational speed limit is $\dot{\psi}_{max}$, and ψ is yaw angle, respectively

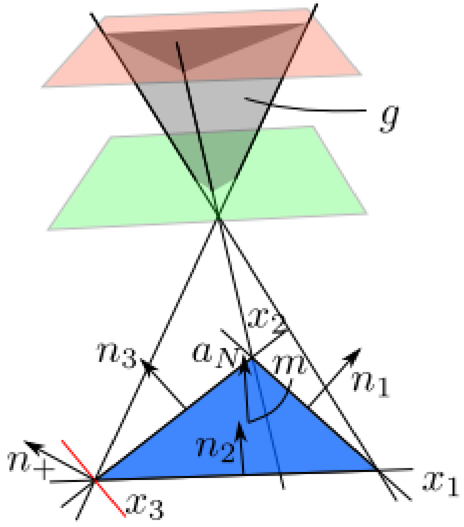
Methodology: viewpoint sampling

- **For every triangle in the mesh, one viewpoint has to be sampled, the position and heading is determined while retaining visibility of the corresponding triangle.**
- **First, the position is optimized for distance to the neighboring viewpoints using a convex problem formulation and then heading is optimized.**
- **To guarantee a good result, the position solution must be constrained such as to allow finding an orientation for which the triangle is visible.**

Methodology: viewpoint sampling

- The constraints on the position $g = [x, y, z]$ consist of the inspection sensor limitation of minimum incidence angle, minimum and maximum range (d_{min}, d_{max}) constraints.
  - $$\begin{bmatrix} (g - x_i)^T n_i \\ (g - x_1)^T a_N \\ -(g - x_1)^T a_N \end{bmatrix} \succeq \begin{bmatrix} 0 \\ d_{min} \\ -d_{max} \end{bmatrix}, i = \{1, 2, 3\}$$
  - Where x_i are the corner of the mesh triangle, a_N is the normalized triangle normal and x_i are the normal of the separating hyperplanes for incidence angle constraints, respectively.

Three main planar angle of incidence constraints on all three sides of the triangle. For a finite number of such constraints the incidence angle is only enforced approximately. The red line (and n_+) demarks a sample orientation for a possible additional planar constraint at a corner. Minimum (green plane) and maximum (red plane) distance constraints are similar planar constraints on the sampling area. These constraints bound the sampling space, where g can be chosen, on all sides (gray area). b)



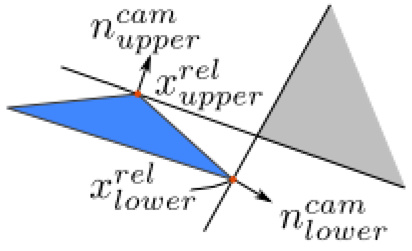
Incidence angle constraints on a triangular facet

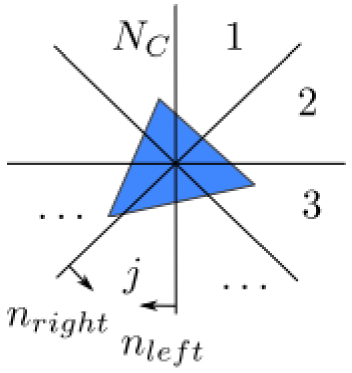
Methodology: viewpoint sampling

- To account for the limited FoV with fixed pitch angle of camera, it imposes a revoluted 2D-cone constraint which is nonconvex problem and then convexified by dividing the space into N_C equal convex pieces.
- The optimum is computed for every slice.

$$\begin{bmatrix} (g - x^{rel}_{lower})^T n^{cam}_{lower} \\ (g - x^{rel}_{upper})^T n^{cam}_{upper} \\ (g - m)^T n_{right} \\ (g - m)^T n_{left} \end{bmatrix} \succeq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

- where x^{rel}_{lower}, x^{rel}_{upper} are the respective relevant corners of the mesh triangle, m the middle of the triangle and n^{cam}_{lower}, n^{cam}_{upper}, n_{right} and n_{left} denote the normal of the respective separating hyperplanes.





Methodology: viewpoint sampling

- **Optimization objective is to minimize the sum of squared distances to the preceding viewpoint g_p^{k-1}, the subsequent viewpoint g_s^{k-1} and the current viewpoint in the old tour g^{k-1}.**

$$\min_{g^k} \quad (g^k - g_p^{k-1})^T (g^k - g_p^{k-1}) + \tag{3}$$
$$(g^k - g_s^{k-1})^T (g^k - g_s^{k-1}) + (g^k - g^{k-1})^T (g^k - g^{k-1})$$

$$\text{s.t.} \quad \begin{bmatrix} n_1^T \\ n_2^T \\ n_3^T \\ a_N^T \\ -a_N^T \\ n_{lower}^{cam\ T} \\ n_{upper}^{cam\ T} \\ n_{right}^T \\ n_{left}^T \end{bmatrix} g^k \succeq \begin{bmatrix} n_1^T x_1 \\ n_2^T x_2 \\ n_3^T x_3 \\ a_N^T x_1 + d_{min} \\ -a_N^T x_1 - d_{max} \\ n_{lower}^{cam\ T} x_{lower}^{rel} \\ n_{upper}^{cam\ T} x_{upper}^{rel} \\ n_{right}^T m \\ n_{left}^T m \end{bmatrix} \tag{4}$$

- **The heading is determined according to**

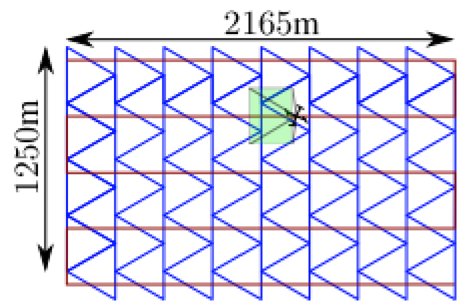
$$\min_{\psi^k} = \left(\psi_p^{k-1} - \psi^k\right)^2 / d_p + \left(\psi_s^{k-1} - \psi^k\right)^2 / d_s, \qquad \text{s.t.} \qquad \text{Visible}(g^k, \psi^k)$$

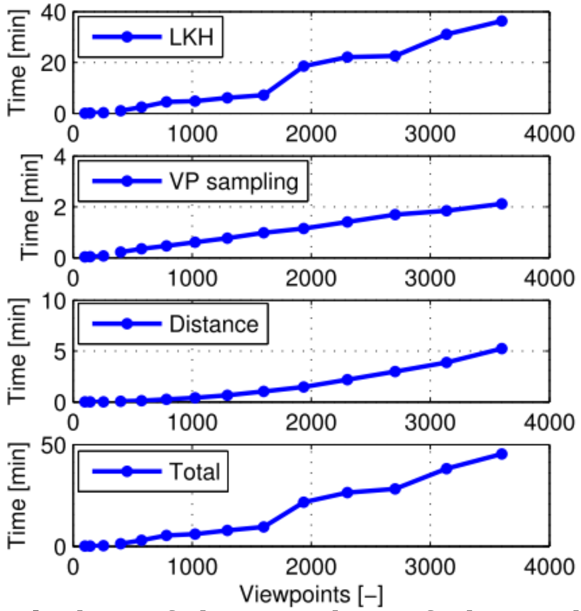
Visible(g^k, ψ^k) means that from the given configuration, g^k and ψ^k, the whole triangle is visible. d_p and d_s are the Euclidean distances from g^k to g_p^{k-1} and g_s^{k-1} respectively.

Computational Analysis

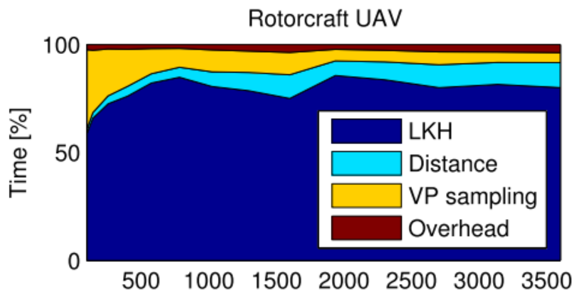
- **To evaluate the capabilities, a simple scenario is used.**

Facets	variable	incidence	30°
FoV	[70,70] °	Mounting pitch	25°
d_{min}	200m	d_{max}	200m
v_{max}	5m/s	ψ_{max}	0.5rad/s

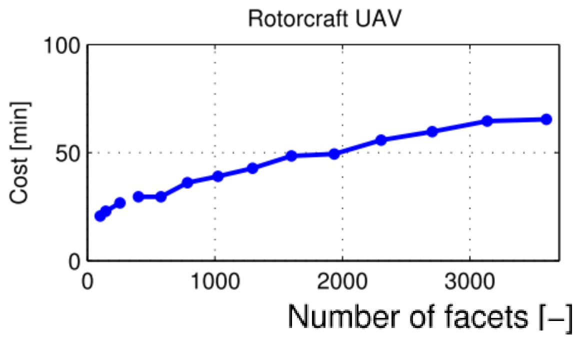




Correlation of the number of viewpoints with the computational time consumption.



(a) Relative time consumption



(b) Resolution dependent cost

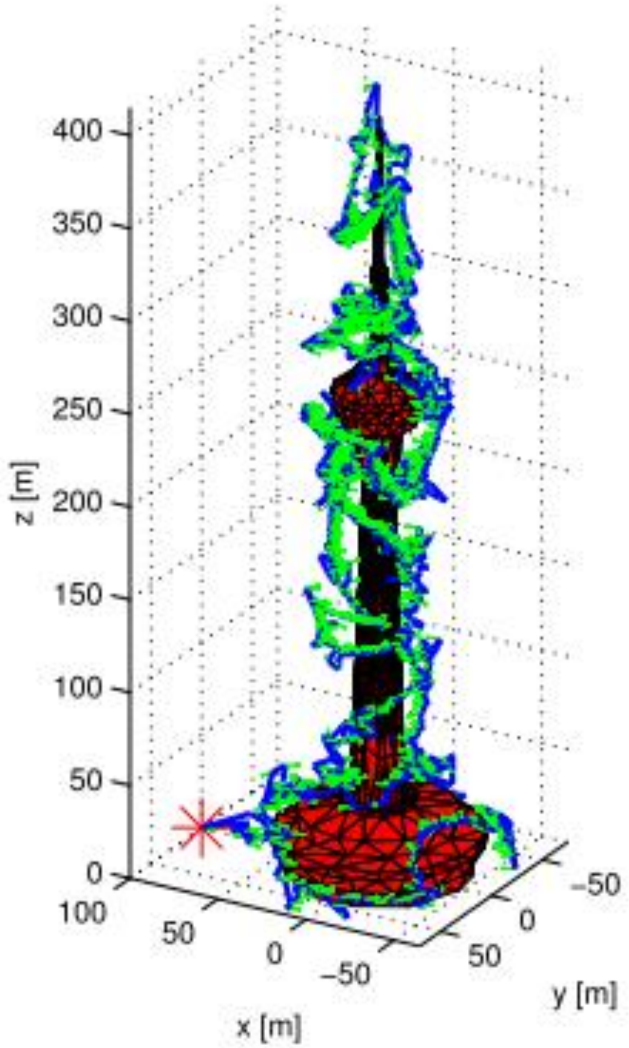
The time complexity:
LKH: $O(N^{2.2})$
VP Sampling: $O(N)$
Distance compute.: $O(N^2)$

Evaluation Test - Simulation

- **405m Tower**

N_{facets}	1701	$\angle$incidence	30°
FoV	[120, 120]°	Mouting pitch	15°
$d_{\min}$	10m	$d_{\max}$	25m
$v_{\max}$	2m/s	$\dot{\psi}_{\max}$	0.5rad/s

Large scale structure to be inspected:
The 405m high Central Radio & TV Tower in Beijing. The mesh used to compute the path contains 1701 triangular facets. After a computation time of 92s the cost for the inspection is 2997.44s The red point denotes, start– and end–point of the inspection.

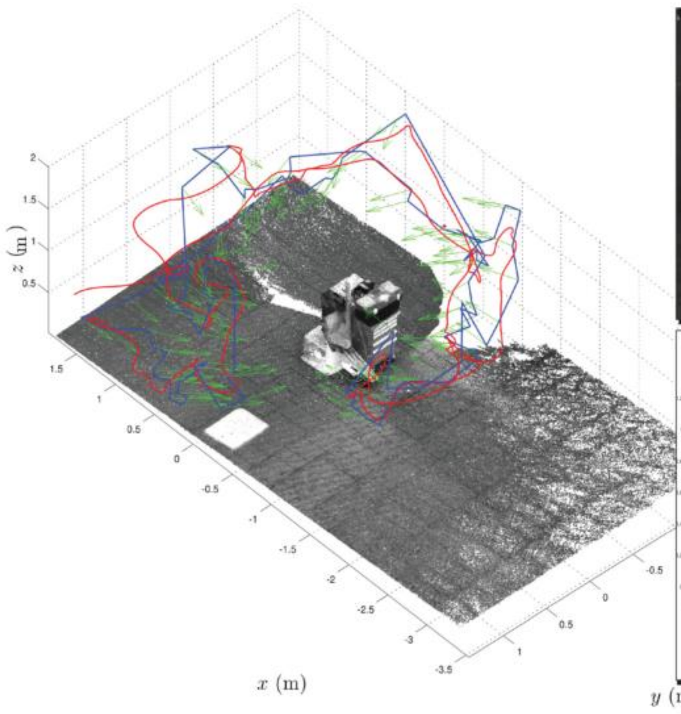


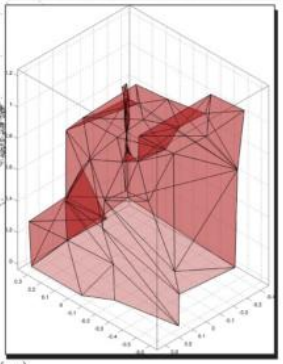
Evaluation Test – VTOL UAV

- **Online: image processing**
- **Offline: 3D reconstruction (Image->Pix4D)**

  Path planner
- **Cost for the inspection: 151.44s**

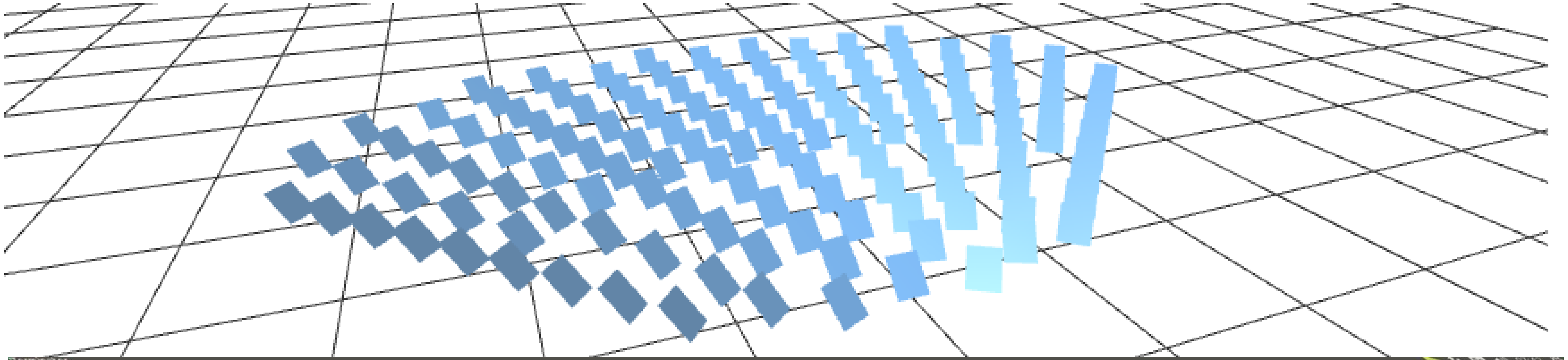
Visual-Inertial Sensor, ATOM CPU (Linux)

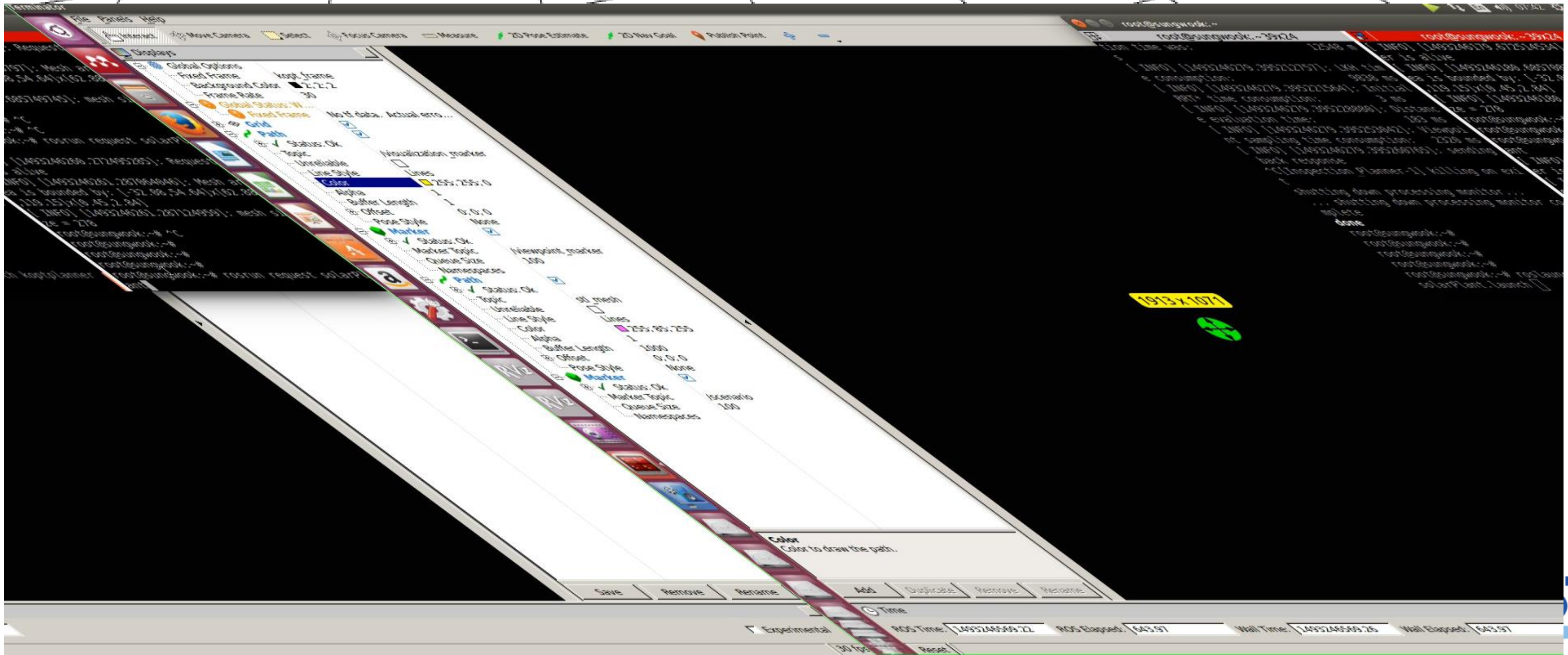




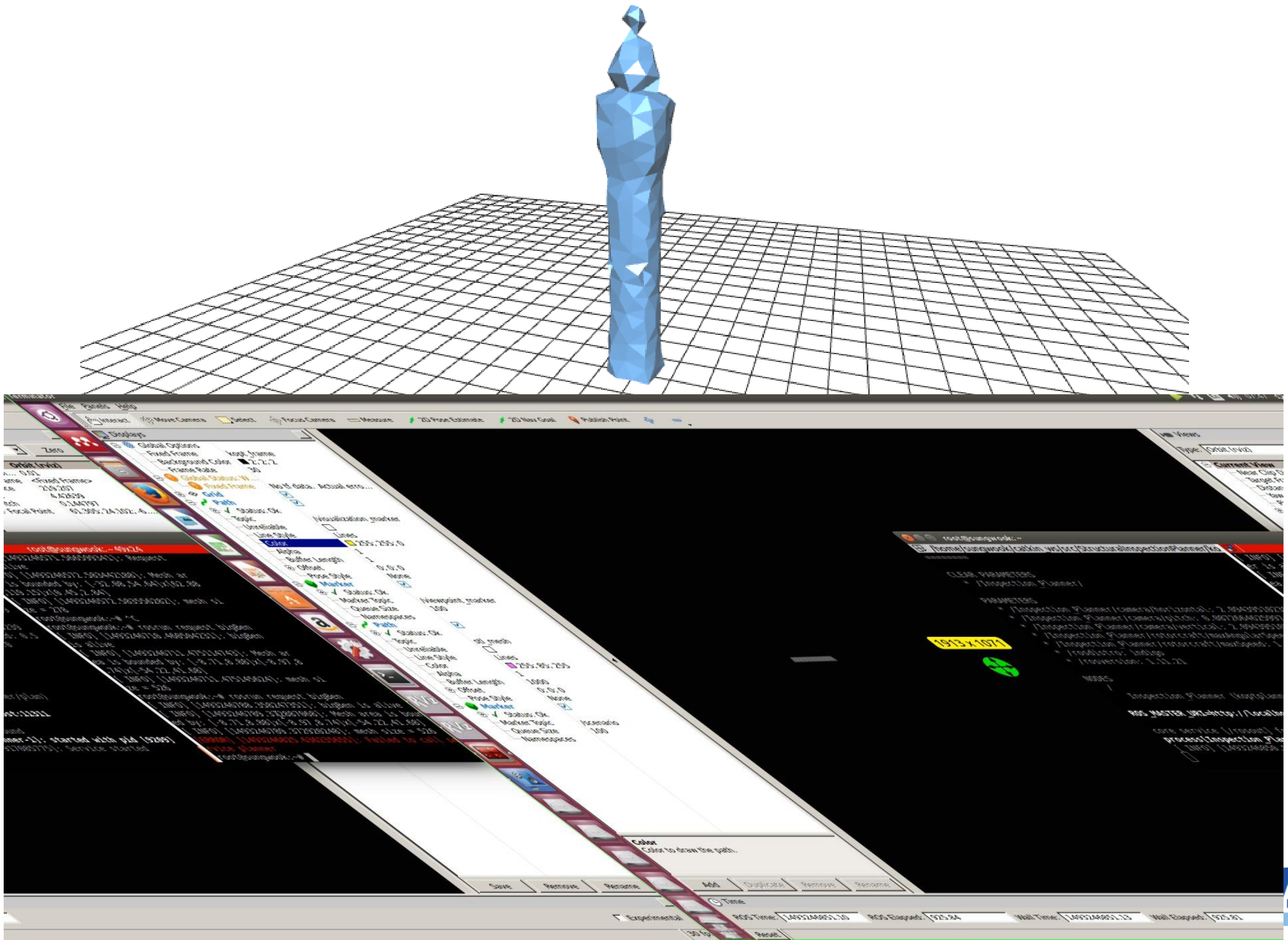
N_{facets}	106		
$\angle$incidence	30°	Bounding box	3x3x2.75m
FoV	$[60, 90]°$	Mouting pitch	15°
$d_{\min}$	1m	$d_{\max}$	3m
$v_{\max}$	0.25m/s	$\dot{\psi}_{\max}$	0.5rad/s

Self-test: solar plant





Self-test: Bigben



```
Global Options
  Fixed Frame          kopt_frame
  Background Color     2;2;2
  Frame Rate           30
Global Status: W...
  Fixed Frame          No tf data. Actual erro...
Grid
Path
  Status: Ok
  Topic                /visualization_marker
  Unreliable
  Line Style           Lines
  Color                255;255;0
  Alpha                1
  Buffer Length        1
  Offset               0;0;0
  Pose Style           None
Marker
  Status: Ok
  Marker Topic         /viewpoint_marker
  Queue Size           100
  Namespaces
Path
  Status: Ok
  Topic                /stl_mesh
  Unreliable
  Line Style           Lines
  Color                255;85;255
  Alpha                1
  Buffer Length        1000
  Offset               0;0;0
  Pose Style           None
Marker
  Status: Ok
  Marker Topic         /scenario
  Queue Size           100
  Namespaces
```

```
Orbit (rviz)
... 0.01
ame   <Fixed Frame>
ce    219.207
      4.42639
tch   0.144797
Focal Point   61.305;24.102;-6....
```

```
root@sungwook: ~ 49x24
1493246571.566959341]: Request
alive
O] [1493246572.583442188]: Mesh ar
is bounded by: [-32.88,54.64]x[62.88
119.15]x[0.45,2.84]
[ INFO] [1493246572.585556282]: mesh si
ze = 278
root@sungwook:~# ^C
root@sungwook:~# rosrun request bigBen
[ INFO] [1493246710.468584231]: bigBen
is alive
[ INFO] [1493246711.475114743]: Mesh ar
ea is bounded by: [-8.71,8.88]x[-8.97,8
.74]x[-54.22,41.68]
[ INFO] [1493246711.475145624]: mesh si
ze = 526
root@sungwook:~# rosrun request bigBen
[ INFO] [1493246788.359247351]: bigBen is alive
[ INFO] [1493246789.372887868]: Mesh area is boun
ded by: [-8.71,8.88]x[-8.97,8.74]x[-54.22,41.68]
[ INFO] [1493246789.372928248]: mesh size = 526
[ERROR] [1493248025.658235655]: Failed to call s
ervice planner
root@sungwook:~#
```

```
er/plan)
st:11311
und
ner-1]: started with pid [9209]
17865775]: Service started
```

```
root@sungwook: ~
/home/sungwook/catkin_ws/src/StructuralInspectionPlanner/ko

CLEAR PARAMETERS
 * /Inspection_Planner/

PARAMETERS
 * /Inspection_Planner/camera/horizontal: 2.98439510
 * /Inspection_Planner/camera/pitch: 0.98726646259
 * /Inspection_Planner/camera/vertical: 2.98439510
 * /Inspection_Planner/rotorcraft/maxAngularSpee
 * /Inspection_Planner/rotorcraft/maxSpeed: 1
 * /rosdistro: indigo
 * /rosversion: 1.11.21

NODES
  /
    Inspection_Planner (koptplann

ROS_MASTER_URI=http://localho

core service [/rosout] fo
process[Inspection_Pla
[ INFO] [1493248050.
```

```
ROS Time: 1493248651.10   ROS Elapsed: 925.84   Wall Time: 1493248651.13   Wall Elapsed: 925.81
```

Summary & Conclusions

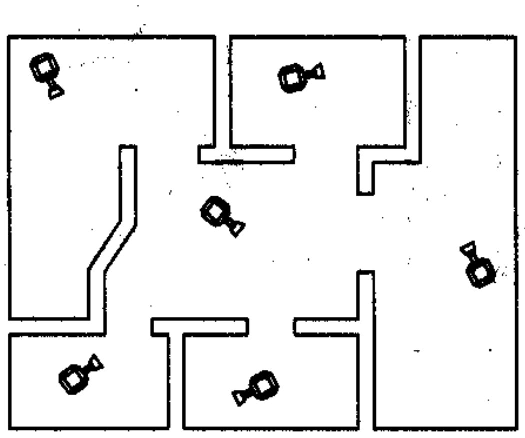
- A practically–oriented fast inspection path planning algorithm capable of computing efficient solutions for complex 3Dstructures represented by triangular meshes was presented.
- With the help of 3D reconstruction software, the recorded inspection data were post-processed to support the claim of finding full coverage paths and the point cloud datasets are released to enable evaluation of the inspection quality.
- https://github.com/ethz-asl/StructuralInspectionPlanner

- **Thanks**

Appendix.1

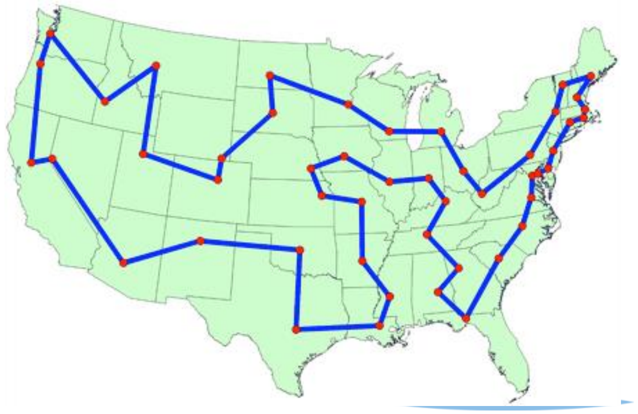
- **Art Gallery Problem (AGP)**

  Suppose you have an art gallery containing priceless paintings and sculptures. You would like it to be supervised by security guards, and you want to employ enough of them so that at any one time the guards can between them oversee the whole gallery. How many guards will you need?



- **Travelling Salesman Problem (TSP)**

  Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city?



Appendix.2

- **Boundary Value Problem (or Solution)**
  - **A boundary value problem has conditions specified at the extremes ("boundaries") of the independent variable in the equation**

Example: Find a solution to the BVP problem

$$\frac{d^2y}{dx^2} - y = 0;\ \ y(0) = 0, y(1) = 1 \ \textit{if we know}$$

$y(x) = c_1e^x + c_2e^{-x}$ *is a general solution to the differential equation.*

  - **whereas an initial value problem has all of the conditions specified at the same value of the independent variable (and that value is at the lower boundary of the domain, thus the term "initial" value).**

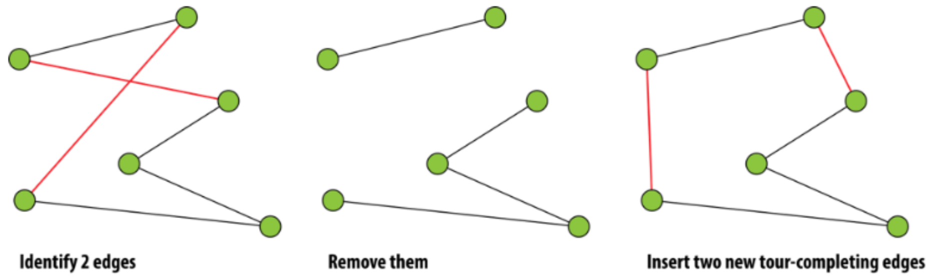
Example: Find a solution to the initial value problem

$y'' + 4y = 0;\ \ y(0) = 1; y'\,(\pi/2) = 2$ *if we know*

$y(x) = c_1cos(2x) + c_2sin(2x)$ *is a general solution to the differential equation.*

Appendix.3

- **Lin-Kernighan-Helsgaun heuristic (LKH, K. Helsgaun, 1998)**
  - **LKH is an effective implementation of the Lin-Kernighan heuristic for solving the traveling salesman problem.**
  - **Even though the algorithm is approximate, optimal solutions are produced with an impressively high frequency.**
  - **Employs the concept of k-opt moves**



Visualization of the k-moves process