2D to 3D Ray Casting for Completeness Model Verification of Airplane Inspections and Color Mapping

Final Presentation of Master Thesis
by Huiying Zhai

Supervisor: Martin Oelsch

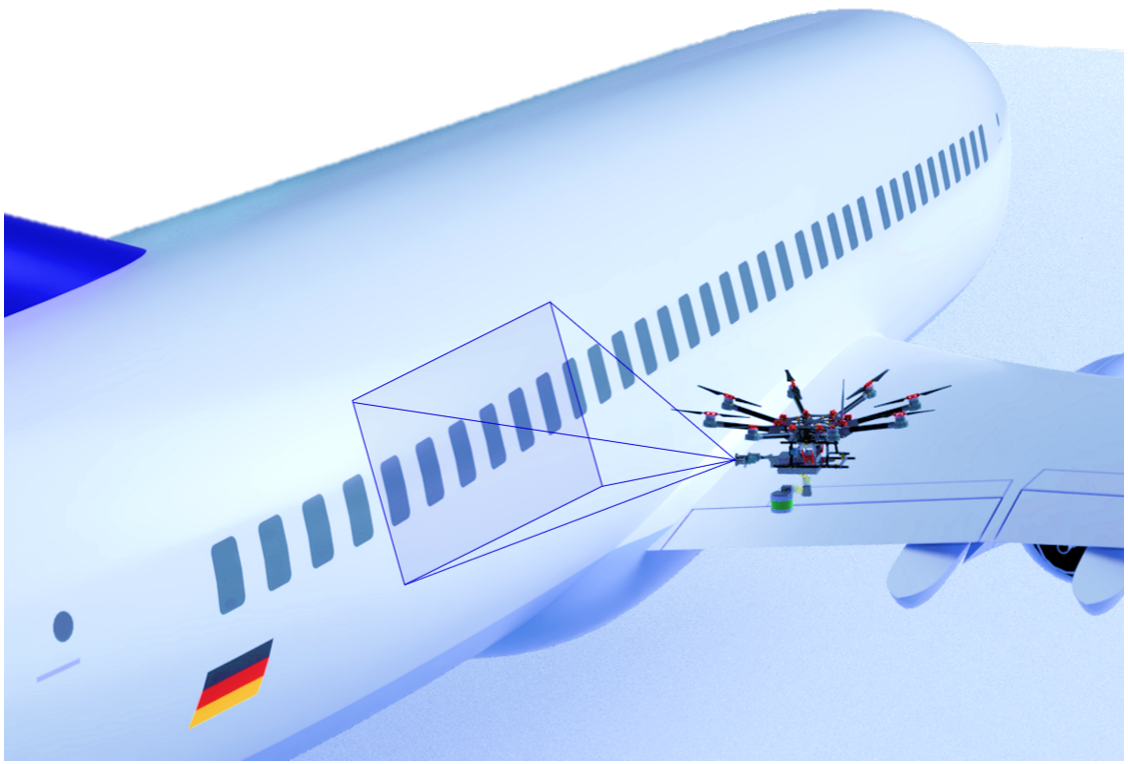
January 2021

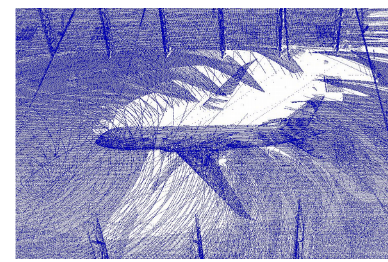
Overview

- Goals and Motivation
- Pipeline
- Methodology
- Experimental setup
- Results

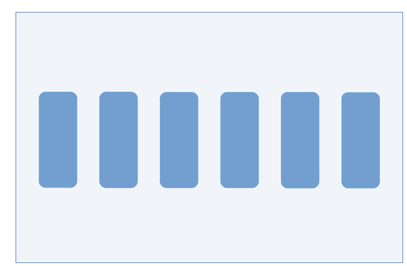
Goals and Motivation

- Metrics as completeness score
- Metrics to verify and improve the ray casting algorithm itself



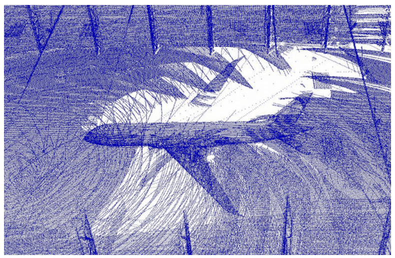


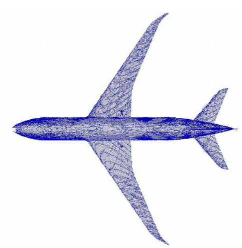
scene cloud



image

Pipeline

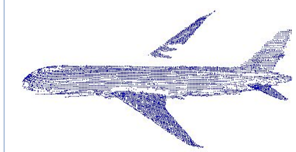

scene cloud

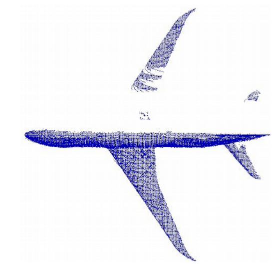

model cloud

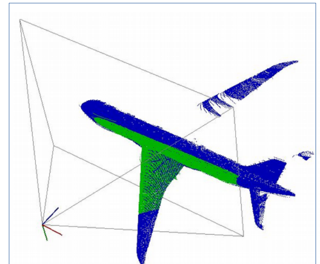
airplane pose detection

airplane segmentation

ray casting

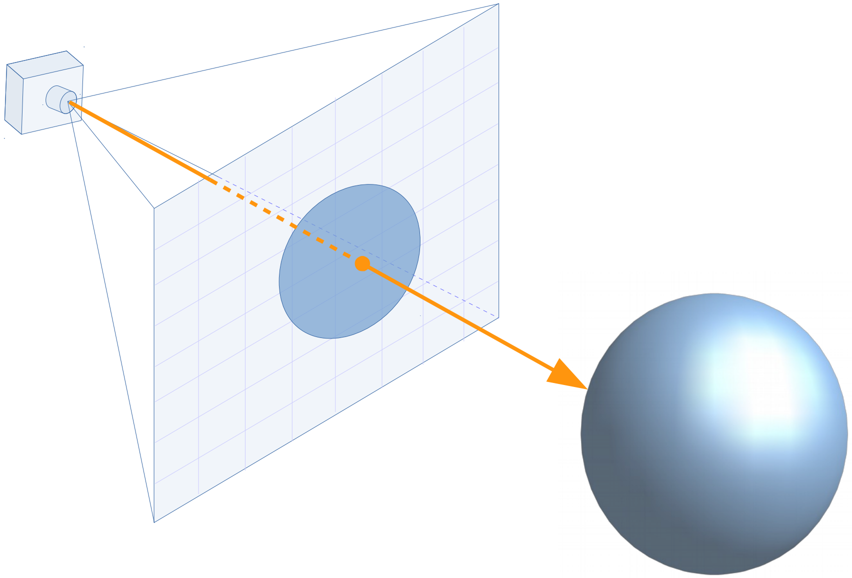

correspondence cloud

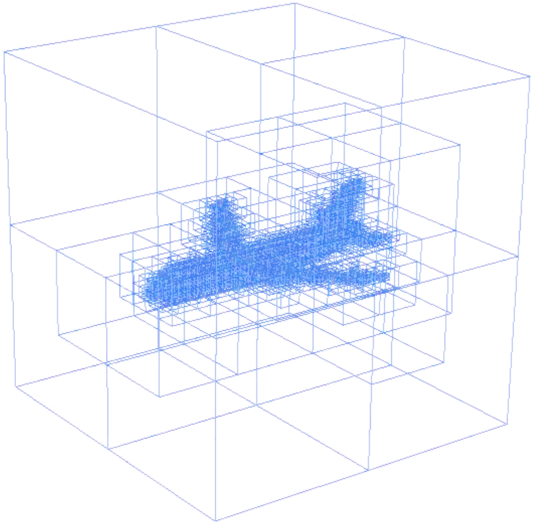

airplane cloud


single image ray casting

Methodology

- **Ray casting**





Methodology

- **Ray casting evaluating metrics**

  - Similar to the evaluation of binary classifiers.

  - Evaluation with two different kinds of ground truth: metric I and II

binary classifier:

	classified as	label
TP	P	P
FP	P	N
TN	N	N
FN	N	P

ray caster:

	hit	ground truth
TP	yes	yes
FP	yes	no
TN	no	no
FN	no	yes

$$sensitivity = recall = \frac{TP}{P} = \frac{TP}{TP+FN}$$

$$specificity = \frac{TN}{N} = \frac{TN}{TN+FP}$$

$$precision = \frac{TP}{TP+FP}$$

$$F_1 = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

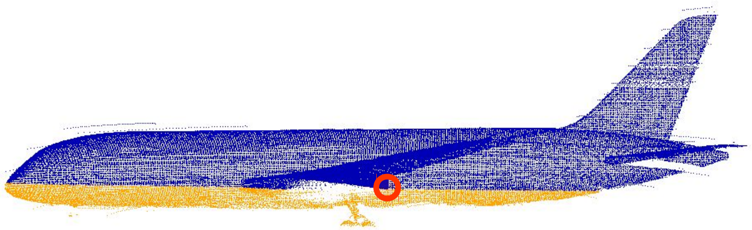
Methodology

- **Ray casting evaluating metric I**

  - Evaluation with two different kinds of ground truth: metric I and II

  - Metric I: Cut the points that are below the wing to reject some noise points. The remand point set is the ground truth.

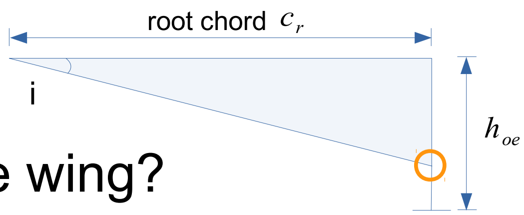
	hit	upper
TP	yes	yes
FP	yes	no
TN	no	no
FN	no	yes

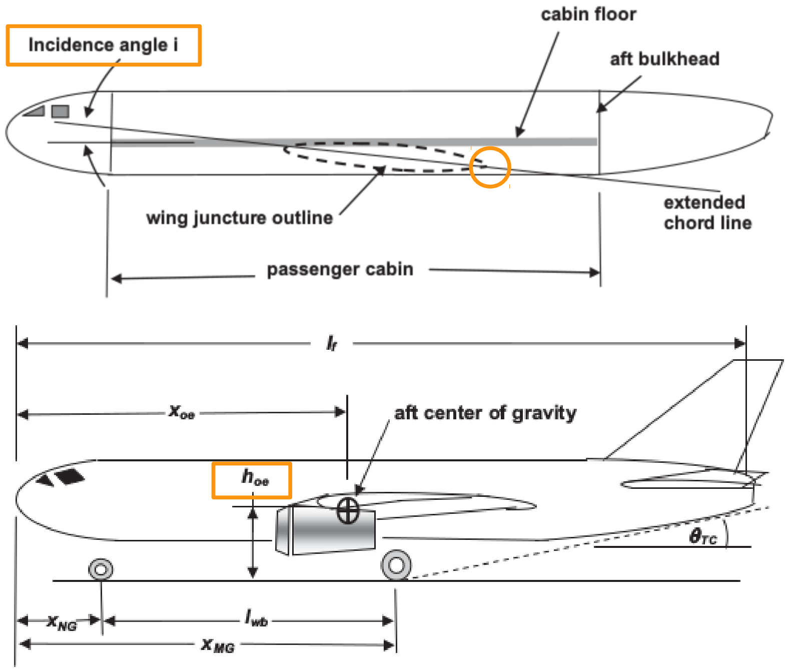


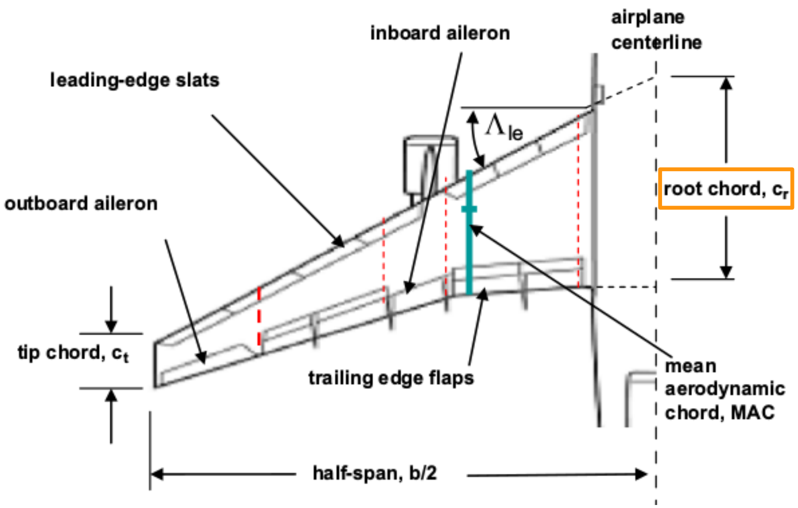
Results

- **Ray casting evaluating metric I**

  - metric I: How to find the lowest point on the wing?







$$h_{lowest} = h_{oe} - \frac{c_r}{\cos(i)} \cdot \sin(i) \approx h_{oe} - c_r \cdot i$$

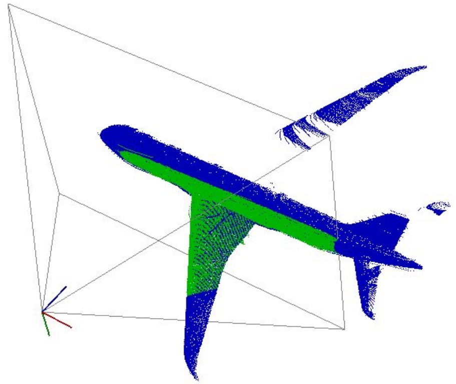
Methodology

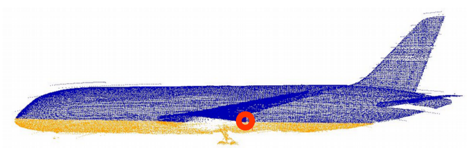
- **Ray casting evaluating metric II**

  - Evaluation with two different kinds of ground truth: metric I and II

  - Metric II: Frustum Culling. Given the information of camera field of view(FOV). The ground truths are the points within the range of FOV.

	hit	FOV
TP	yes	yes
FP	yes	no
TN	no	no
FN	no	yes





Methodology

- **Completeness metric I**

  - Describes the percentage of inspected airplane model.

  - Accumulated hit ratio for 900 images: $r_{hit} = \frac{n_{hit}}{n_{total}}$

  - Assumption: all hit points are TP.

  - The quality of ray casting will affect the completeness score.

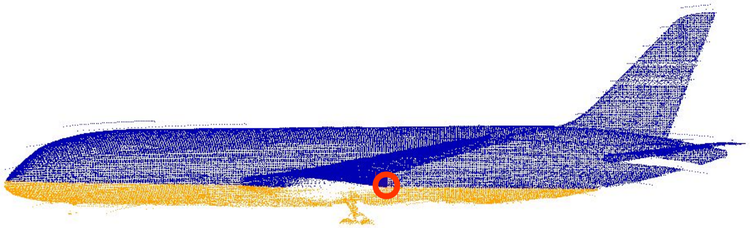
Methodology

- **Completeness metric II**

  - Computes accumulated sensifivity/sepecificity/precision/F1-score.

  - Ground truth: the same as ray casting evaluating metric I.

  - The quality of ray casting will affect the completeness score.



Methodology - Improvement summary

- Algorithm research, analysed their pros and cons
- Improved airplane segmentation
- Improved the old algorithm:

	old algorithm	**new algorithm**
hit points	Check the first 100 voxels, randomly pick one of the points as hit.	Check the first hit voxel, mark the point(s) inside it as hit(s).
FOV	FOV is sphere-shaped.	FOV is pyramid-shaped.
distance threshold	60m/1000m	30m
octree voxel size	5cm/10cm/20cm	4.7cm/7cm/17cm
visualizer	offline	online

Experimental setup

- **Datasets:**

  - 900 images with pixel resolution 4112 x 3008

  - point cloud with resolution 5cm/10cm/20cm

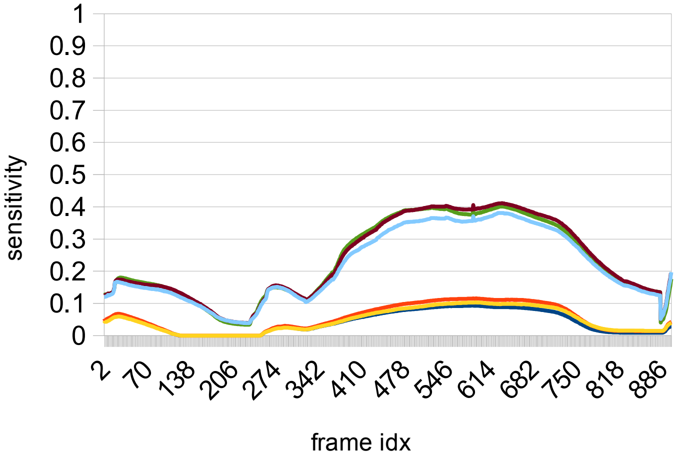
- **Octree resolution:** 4.7cm/7cm/17cm

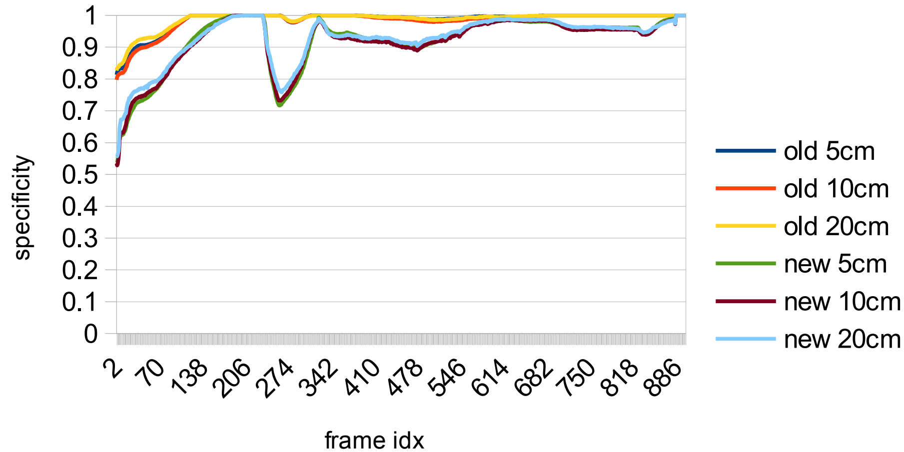
- **Distance threshold:** 30m

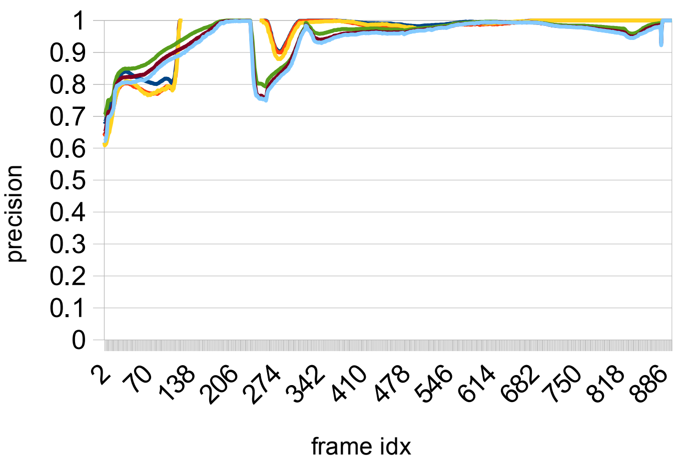
Results - metric I

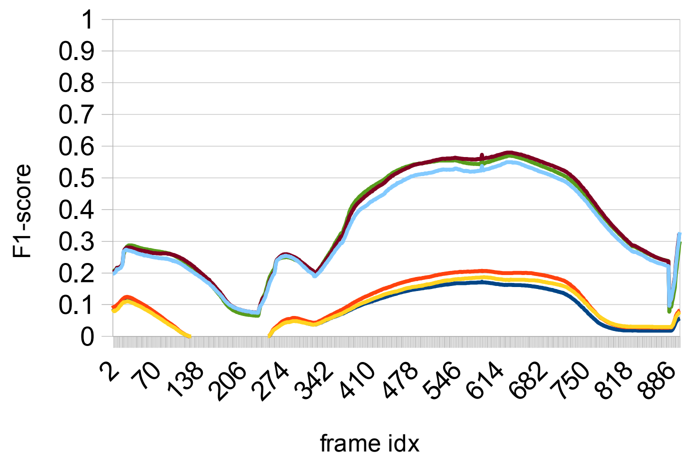
$$sensitivity = recall = \frac{TP}{P} = \frac{TP}{TP+FN} \qquad specificity = \frac{TN}{N} = \frac{TN}{TN+FP}$$

$$precision = \frac{TP}{TP+FP} \qquad F_1 = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

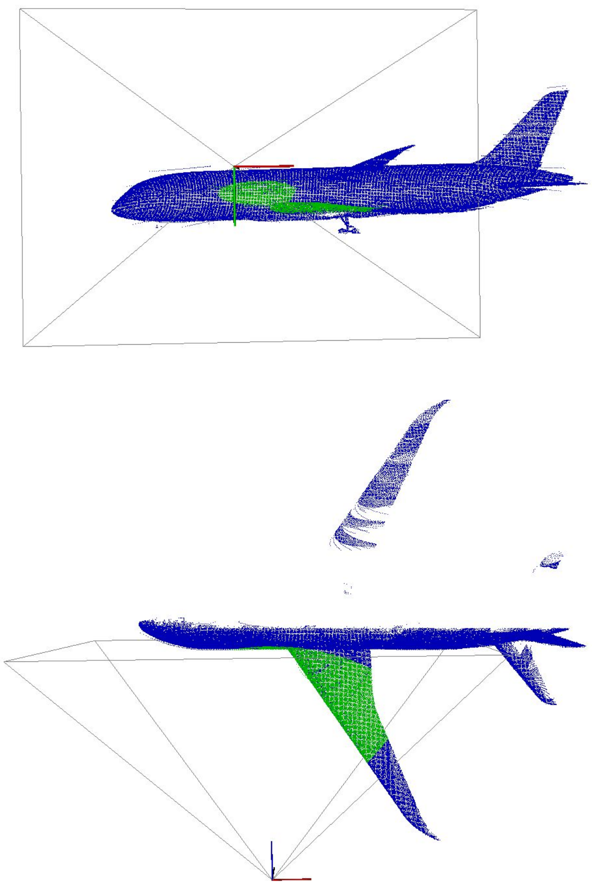




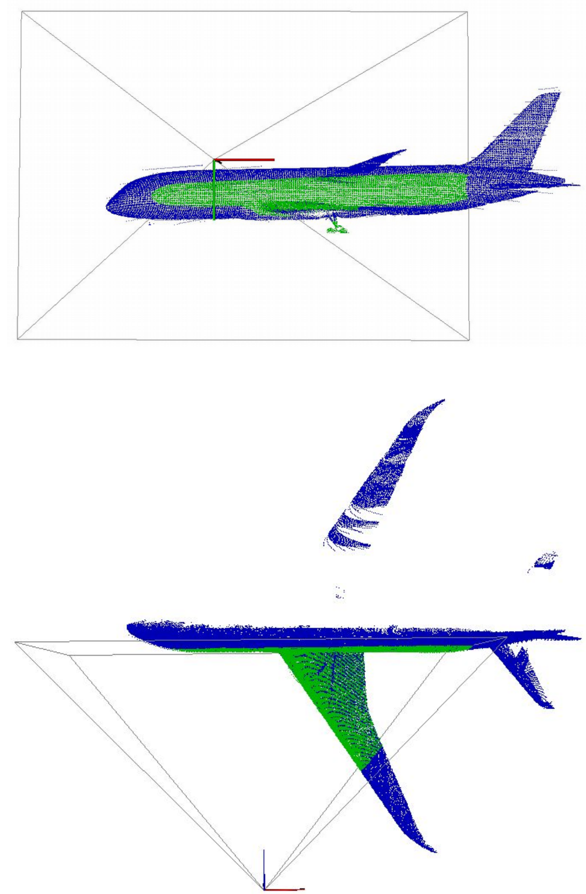




Results - metric I



Old algorithm, cloud res 10 cm

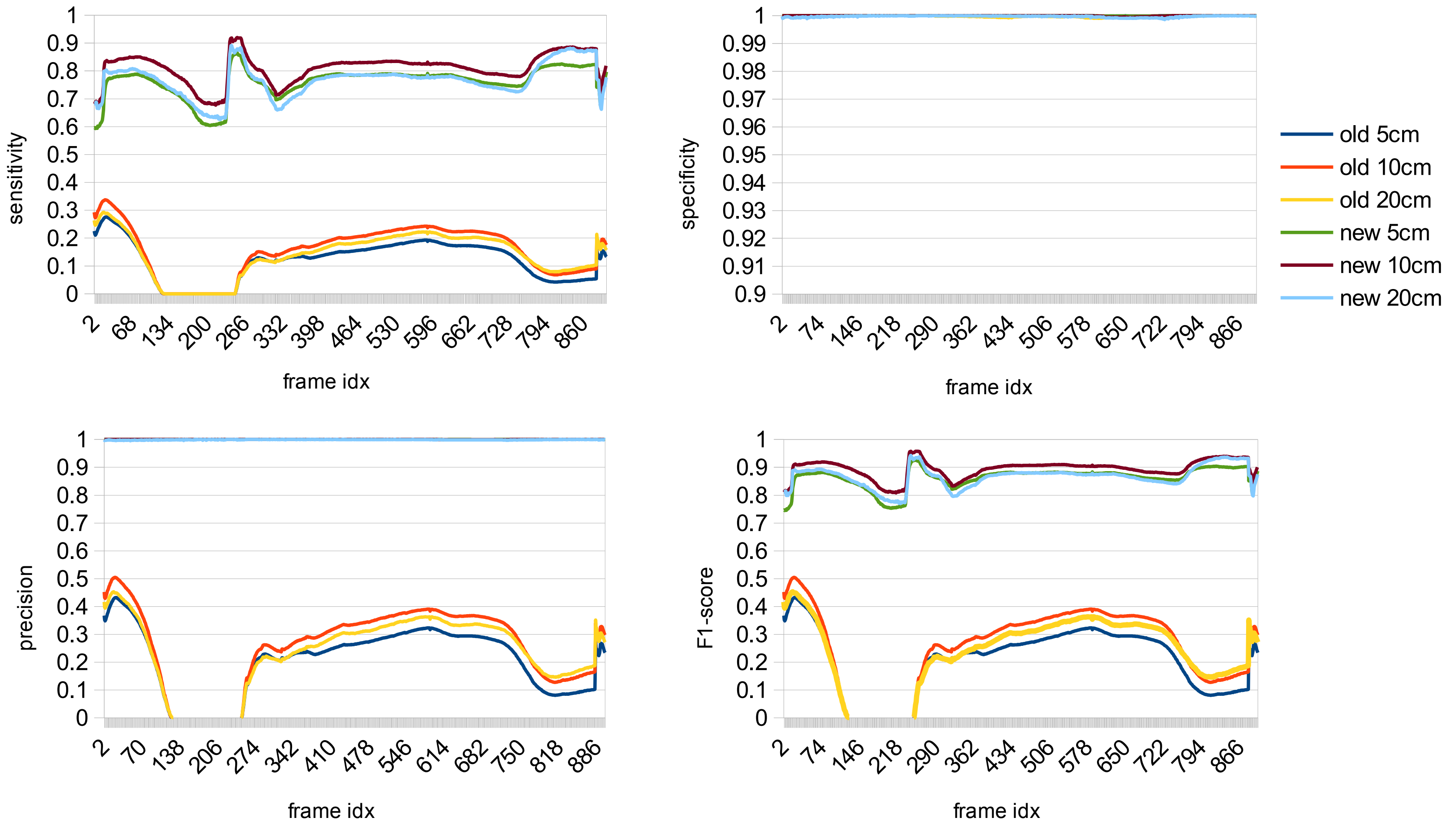


New algorithm, cloud res 10cm

Results - metric II

$$sensitivity = recall = \frac{TP}{P} = \frac{TP}{TP+FN} \qquad specificity = \frac{TN}{N} = \frac{TN}{TN+FP}$$

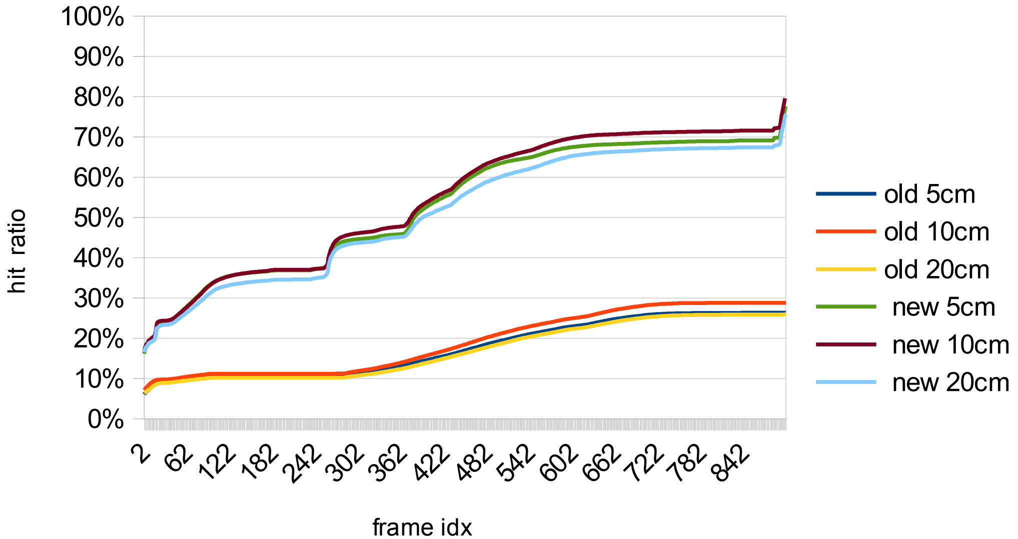
$$precision = \frac{TP}{TP+FP} \qquad F_1 = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

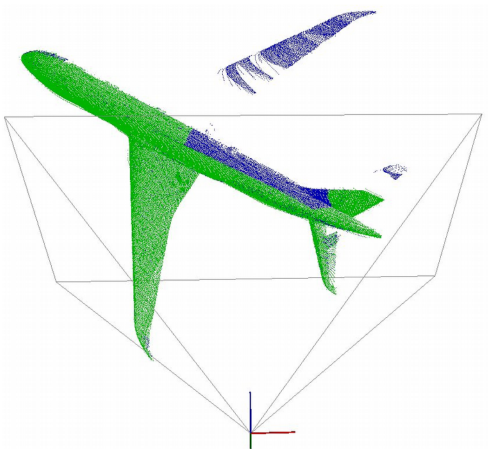


Results – Completeness score I

- Accumulated hit ratio

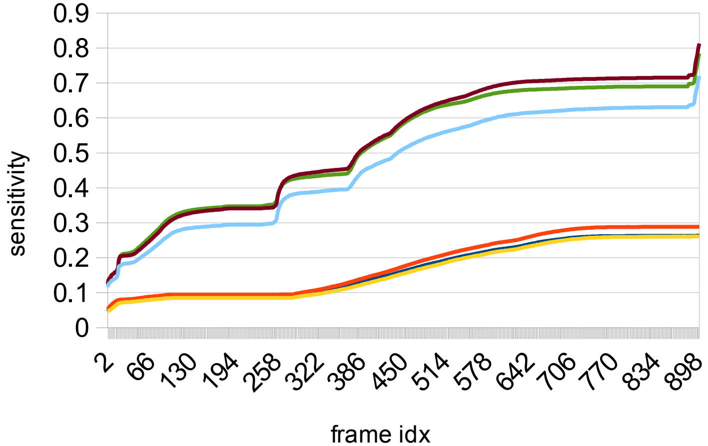
  - the distance threshold and octree resolution will affect the completeness score

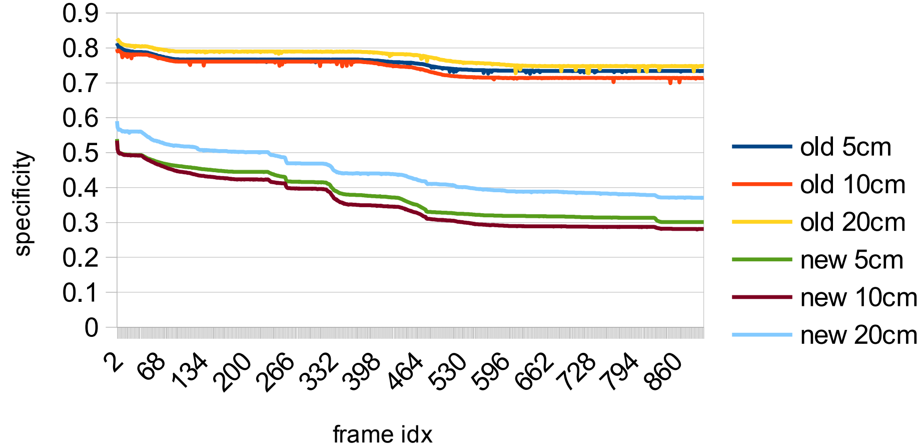


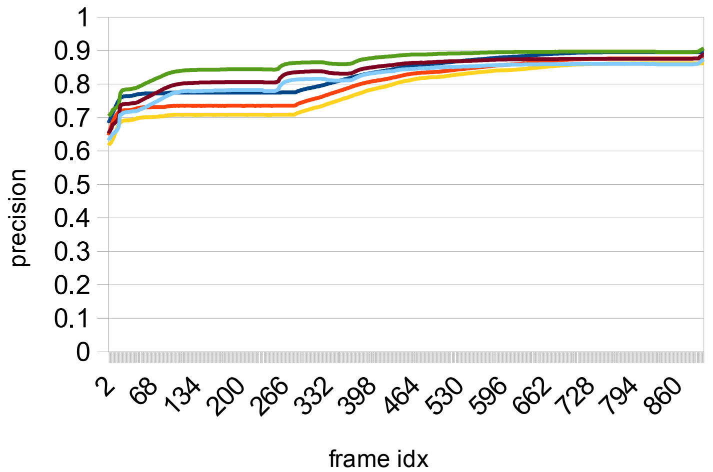


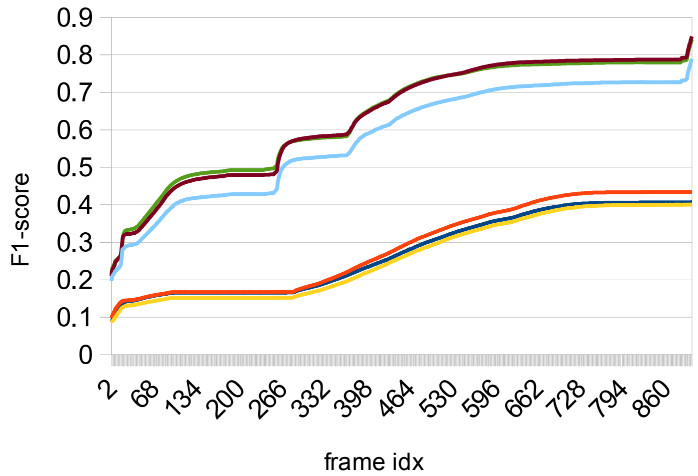
Final ray casting result using the new algorithm

Results - Completeness score II









Conclusion

- Improved ray casting algorithm
- Investigated two metrics for ray casting performance

  - The metric I ground truth point set is not perfect.

  - The metric II groud truth has noise points → could be improved by the combination of I and II.
- Investigated two metrics for completeness score

  - The score I assumes all hits are TP.

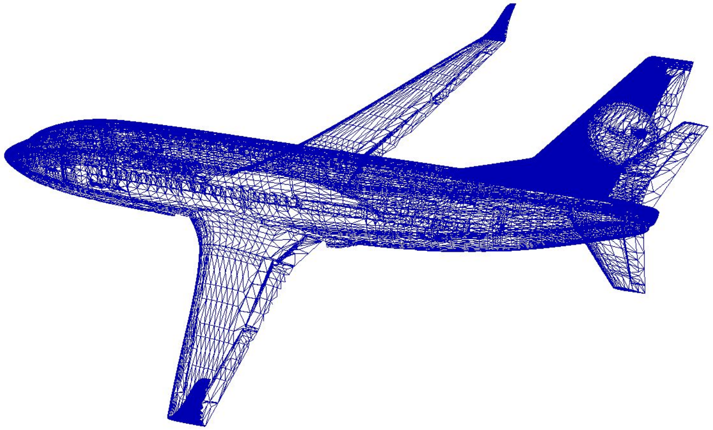
  - The score II is ideal.

Plan for next stage (Deadline: 31. December)

- A new ground truth combines metric I and II.
- To compute completeness score for full scan datasets.
- To use mesh file evaluate the completeness score.

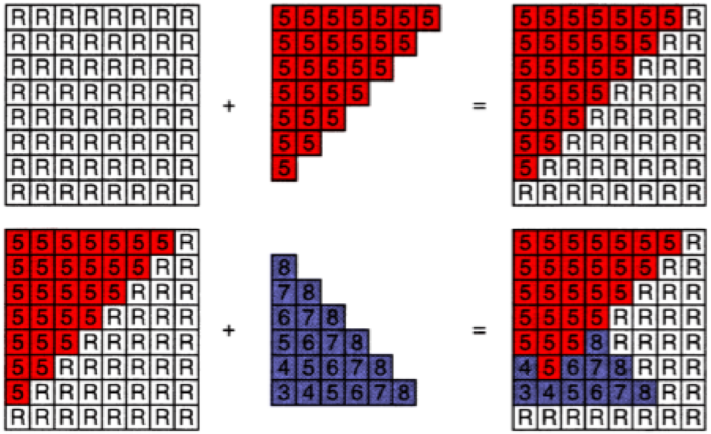
  - Rays hit triangles instead of points.

  - Ground truth: triangle set.



Methodology

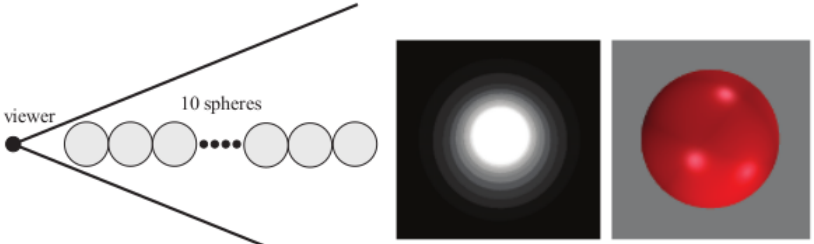
- **3D → 2D (Z-Buffer)**



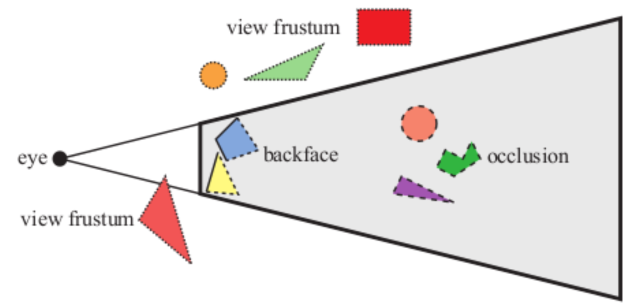
Initialize depth buffer to ∞

During rasterization:

```
for (each triangle T)
    for (each sample (x,y,z) in T)
        if (z < zbuffer[x,y])          // closest sample so far
            framebuffer[x,y] = rgb;    // update color
            zbuffer[x,y] = z;          // update depth
        else
            ;                  // do nothing, this sample is occluded
```



depth complexity   rendered image



Methodology

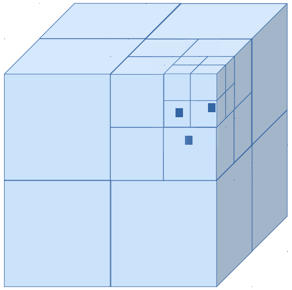
- **2D → 3D(ray casting) v.s. 3D → 2D(z-buffer)**

	Ray Casting	**Z-Buffer**
advantages	1. requires no extra algorithms 2. flexible 3. easy to implement 4. easy to parallize 5. can be speed up further with acceleration structure and efficient traversal algorithm	1. hardware z-buffer is faster
disadvantages	1. slower than hardware z-buffer	1. requires culling algorithms 2. extra memory for ground truth cloud and depth 3. hardware z-buffer is overkill and not flexible

Methodology

- **Spatial partition**

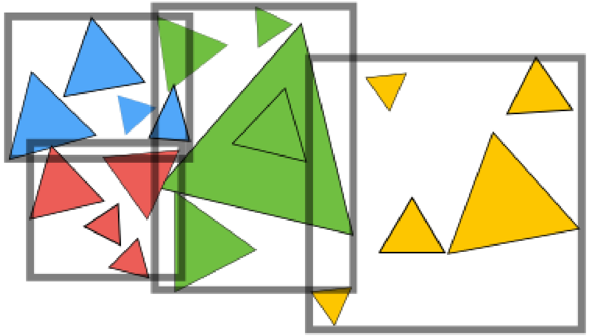
  - octree



- **Object partition**

  - bouding volume hierarchy(BVH)

  - AABB-tree



Methodology

- **Spatial partition v.s. object partition**

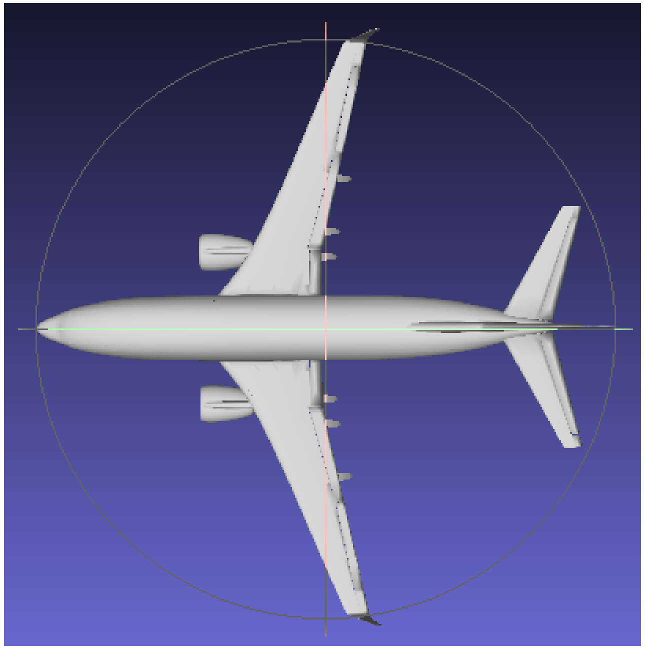
	spatial partition	**object partition**
principles	Splits space into non-overlapping subregions.	Splits objects into disjoint subsets.
disadvantages	One single object may occupy multiple subregions.	Bounding volumes may overlap in space.

- Spatial partition is suitable for point cloud ray casting.

- Object partition is better choice for object/polygon ray casting.

Methodology

- **Mesh preprocessing – cut the mesh**

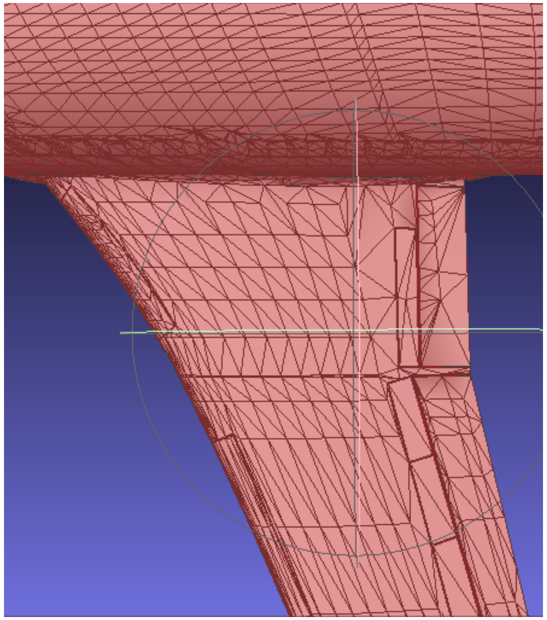


before

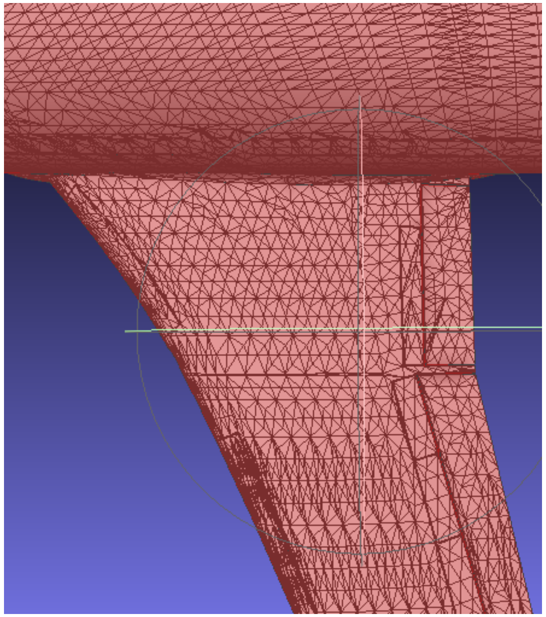
after

Methodology

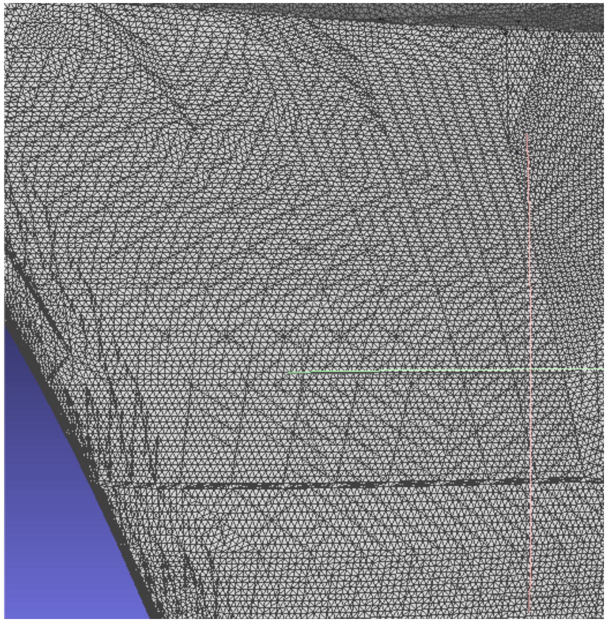
- **Mesh preprocessing – mesh subdivision**



original



5th iteration



final

Methodology

- **Completeness score I**

  \- triangle hit percentage: $\dfrac{n_{hit}}{n_{total}}$

- **Completeness score II**

  \- surface area hit percentage: $\dfrac{A_{hit}}{A_{total}}$

Methodology

- **Color mapping with point cloud**

  - direct approach

- **Color mapping with mesh I**

  - direct approach

- **Color mapping with mesh II**

  - weighting mask blending(angle, depth, border)

  - coloring with mean of weighted sum of pixel color values

- **Color mapping with mesh III**

  - parameterization and texturing

  - coloring by selecting the best pixel color

Experimental setup

- **Datasets:**

  - 9420 images with pixel resolution 4112 x 3008 (full scan)

  - triangle meshes: original and subdivided with 10cm triangle size

- **Distance threshold:** 30m/35m/40m

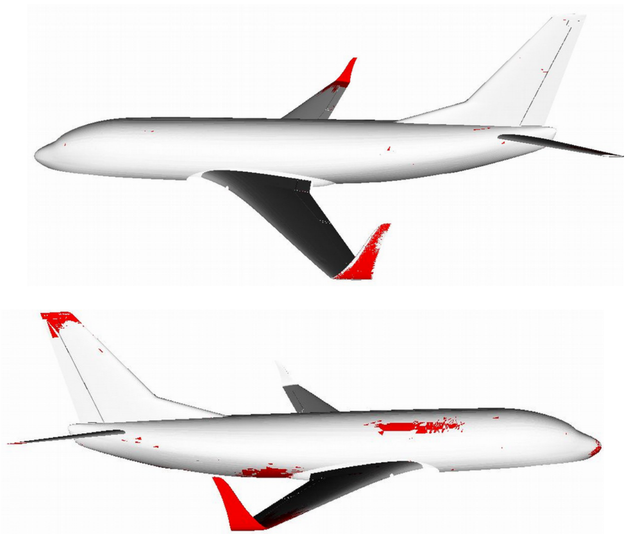
Results - Completeness score I

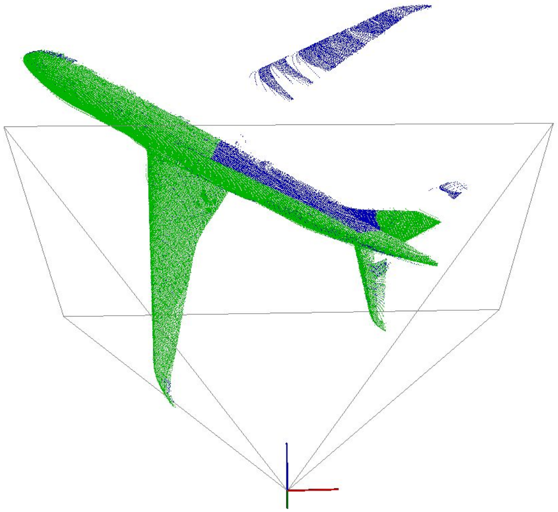
- **Data**

mesh, dist. threshold	**triangle hit %**
original, 30m	80.268
original, 35m	80.5618
original, 40m	82.3294
subdivided, 30m	86.4189

Results - Completeness score I

- **Visualization**



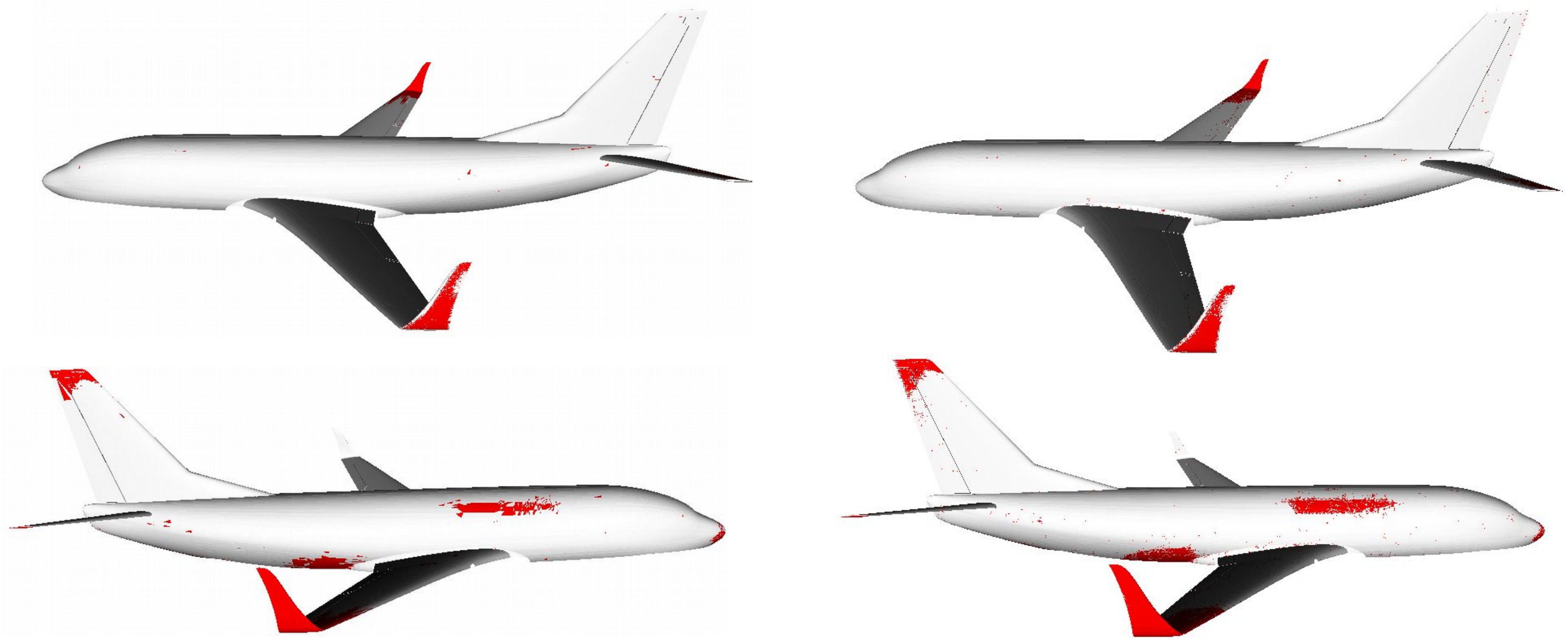


Mesh-based ray casting, dist = 30m, completeness score = 80.260%

Cloud-based ray casting, dist = 30m, completeness score is about 80 %

Results - Completeness score I

- **Visualization after mesh subdivision**



Ray casting with original mesh

Ray casting with subdivided mesh

Results - Completeness score II

- **Data**

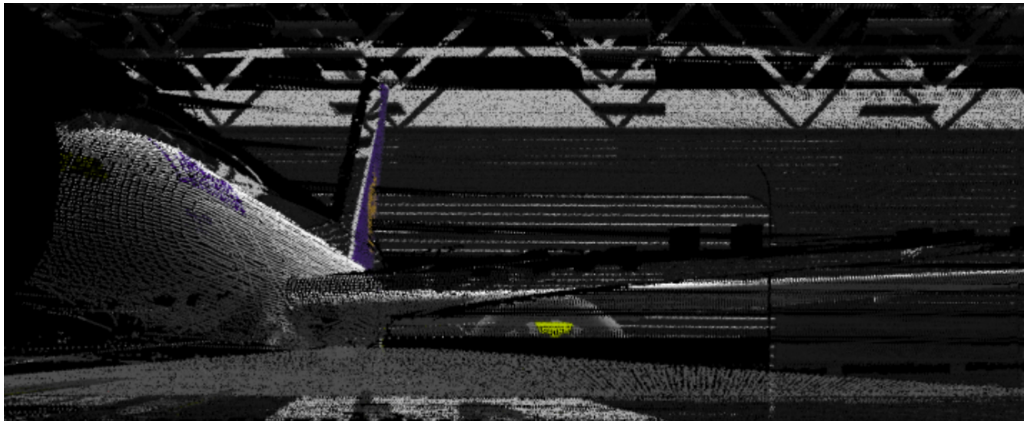
mesh, dist. threshold	triangle hit %	surface are hit %
original, 30m	80.268	95.667
original, 35m	80.5618	96.5160
original, 40m	82.3294	97.5827
subdivided, 30m	86.4189	94.3704

Results – Color Mapping with Point Cloud

- **original**

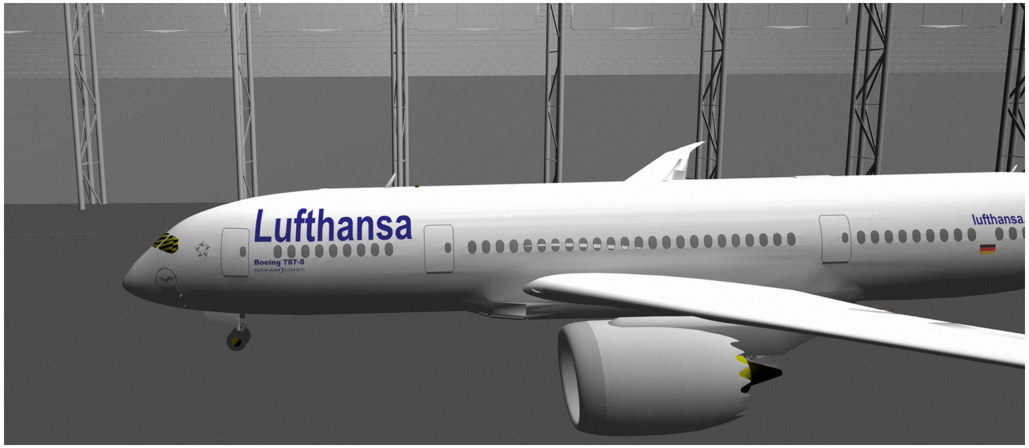
- **result 1**

  - scaling err



Results – Color Mapping with Point Cloud

- **original**

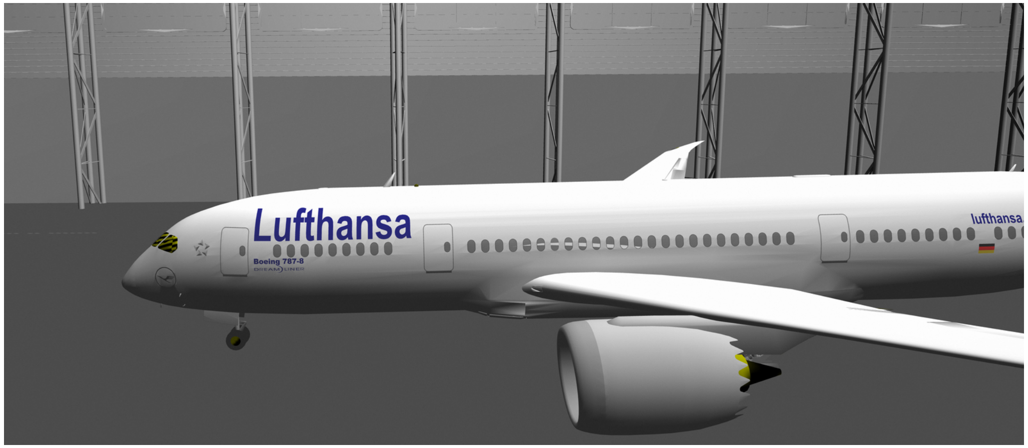


- **result 2**

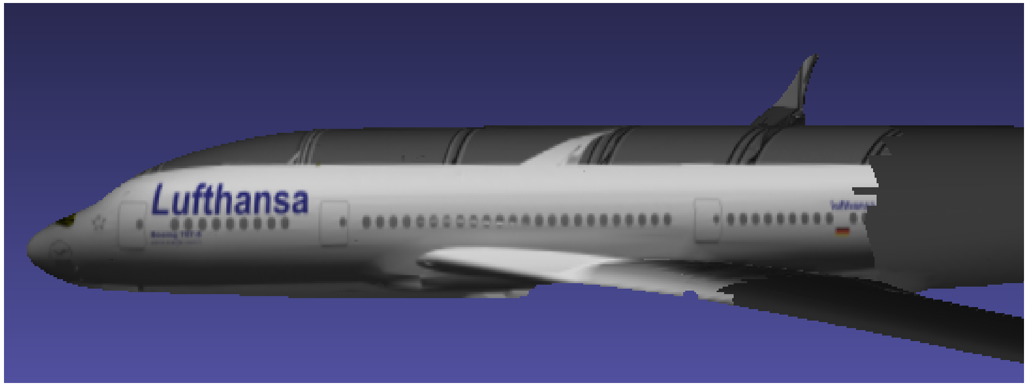
  - scaling err

Results – Color Mapping with Mesh I

- **original**

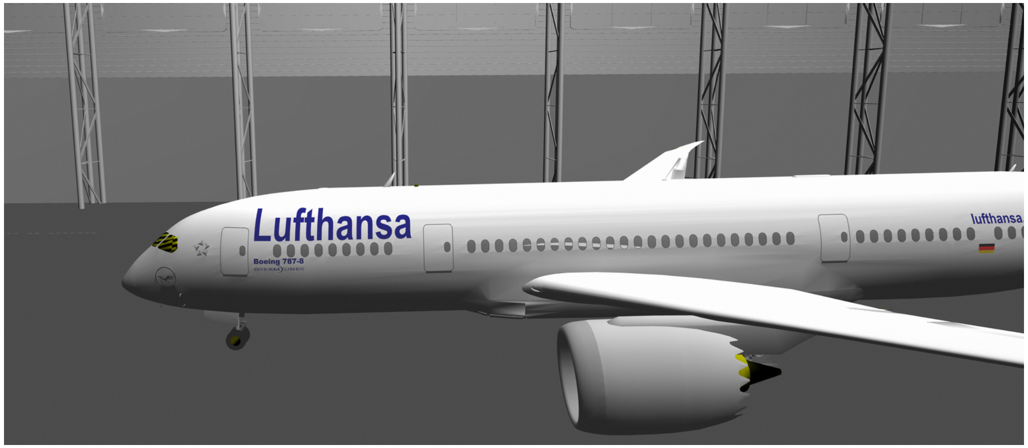


- **result 1**
  - extrinsic err
  - intrinsic err



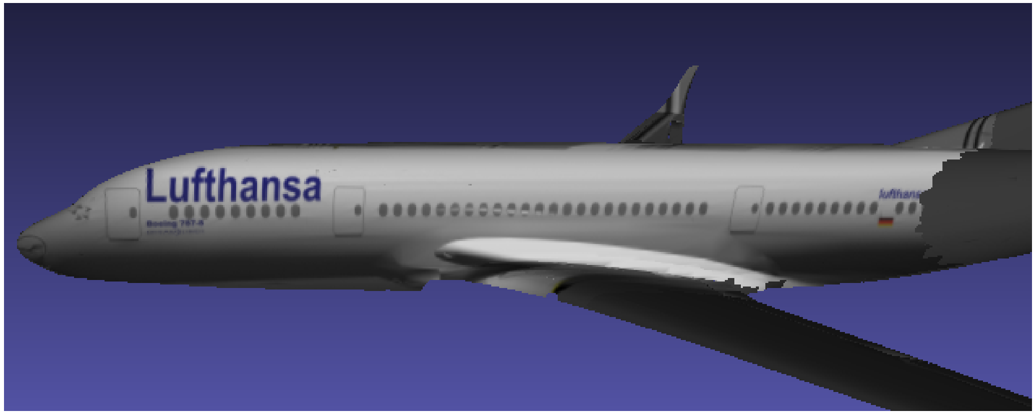
Results – Color Mapping with Mesh I

- **original**



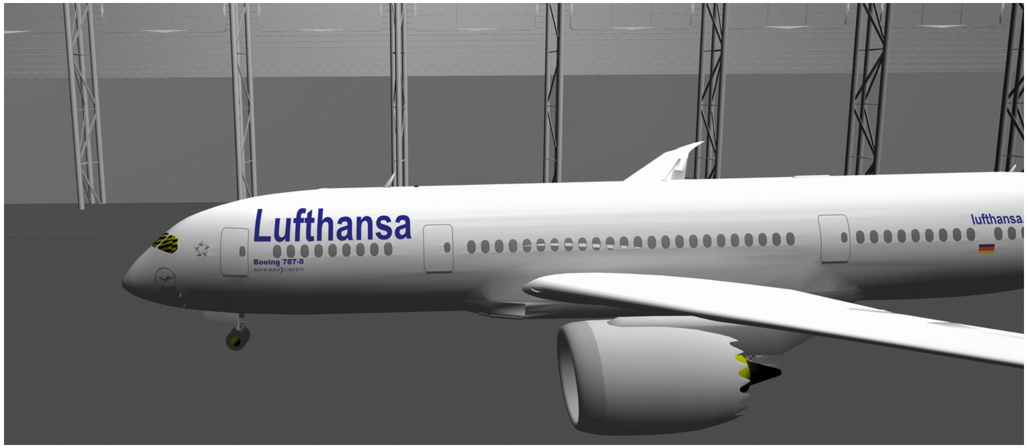
- **result 2**

  - correction

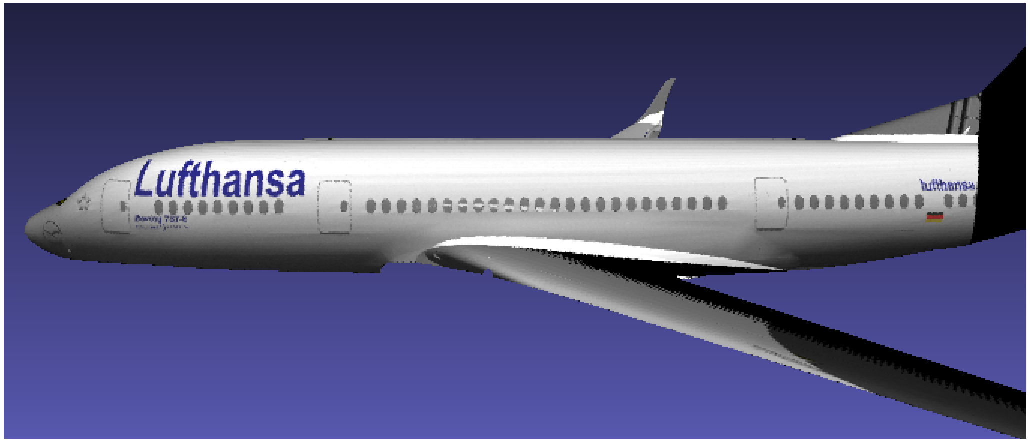


Results – Color Mapping with Mesh II

- **original**

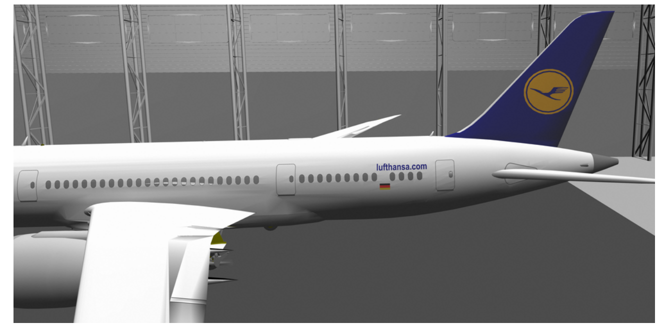


- **result 1**



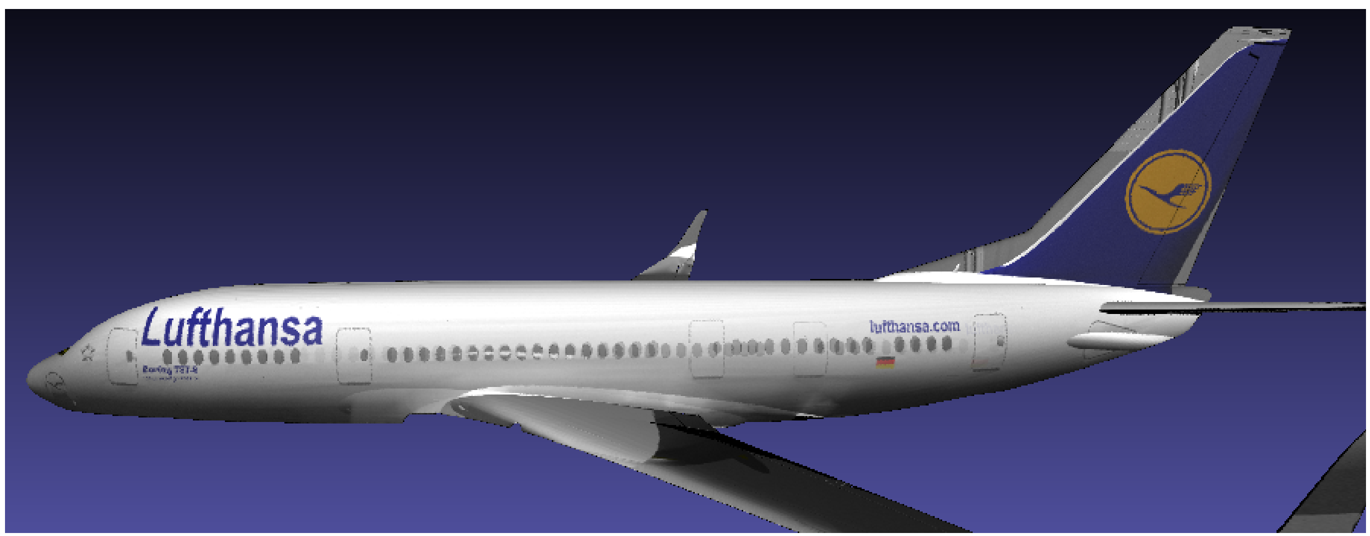
Results – Color Mapping with Mesh II

- **original**



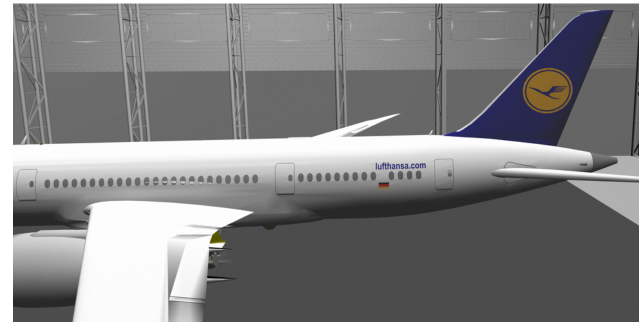
- **result 2**

  - ghosting



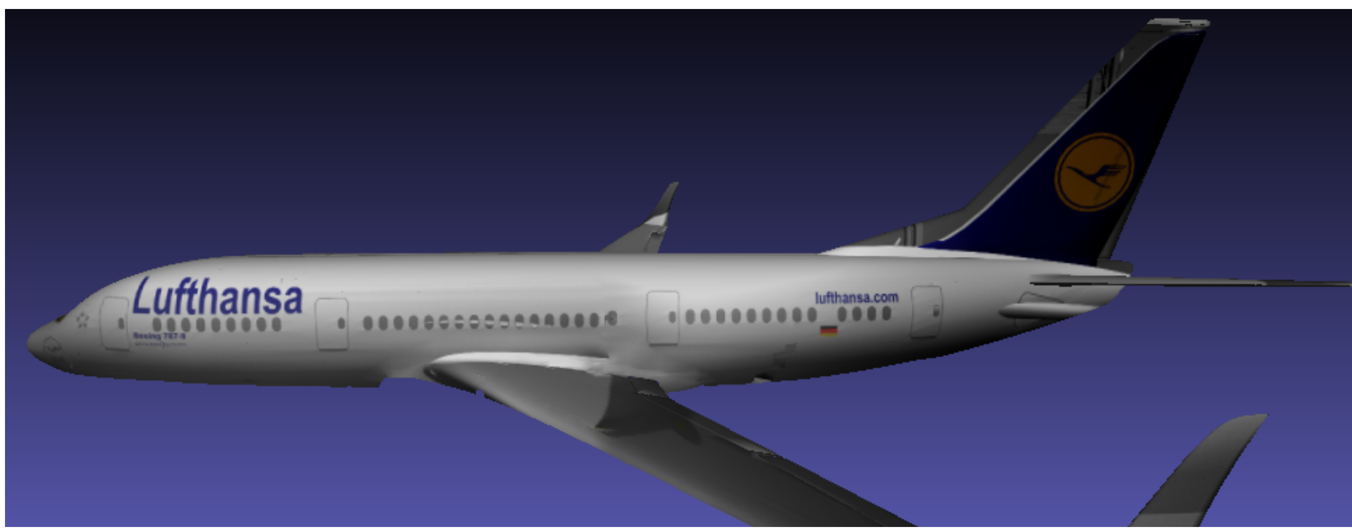
Results – Color Mapping with Mesh III

- **original**



- **result**

  - darker



Conclusion

- Completeness score I:

  - Underestimation caused by non-even triangle distribution(size).

  - Can be balanced by mesh subdivision. However cannot be completely avoided.

- Completeness score II:

  - Holes ocurr due to large drone-plane distance.

  - Drone should be driven closed enough to airplane.

  - Results correspond to visualization.

- Optimal metric: score II with mesh subdivision.

Conclusion

	cloud-based approach	mesh-based approach
acceleration data structure	octree	AABB-Tree
advantages and disadvantages	- results influenced by octree resolution - large: overestimation - small: underestimation - requires parameter tuning - almost impossible actually hit a point(noisy measurement) - set ray-point distance threshold	+ each mesh correspondes to an unique AABB-tree + precise triangle hit by geometrical calculation

- However both approaches are affected by drone-airplane distance

  → The inspection drone should be driven closed enough to airplane

Conclusion

- Color mapping

  - direct color mapping approaches suffer the texture distortion(strech on the edges)

  - requires precise extrinsics and intrinsics

  - mask blending is ideal for model with high resolution

  - parameterization is suitable for coarse model

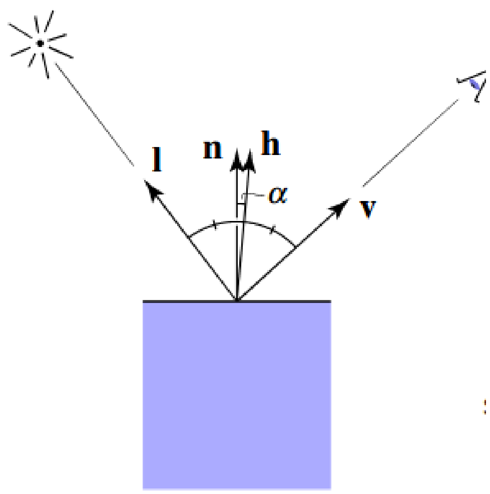
Future work

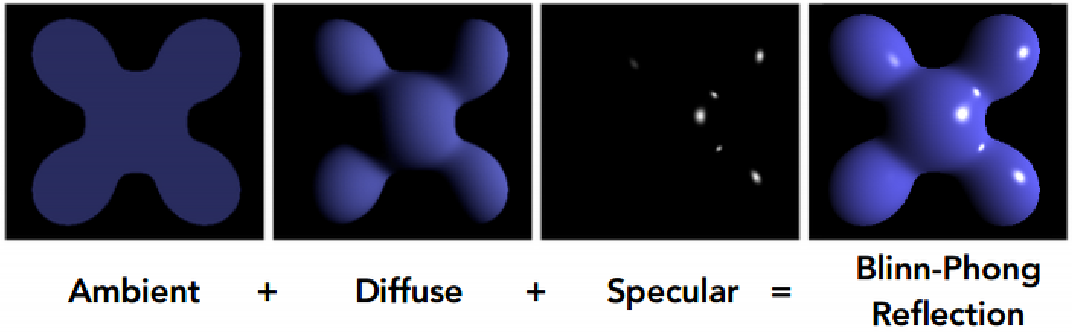
- Airplane types database and automatic airplane recognition for cloud- and mesh-preprocessing
- Ray casting to implicit surface representation(volume) with marching cubes for colored mesh generation
- Dynamic tessellation and level of detail(LOD) for hole filling and watertight texture mapping

Appendix

- **Color mapping with mesh II – angle mask**

  - angle between surface normal and view direction





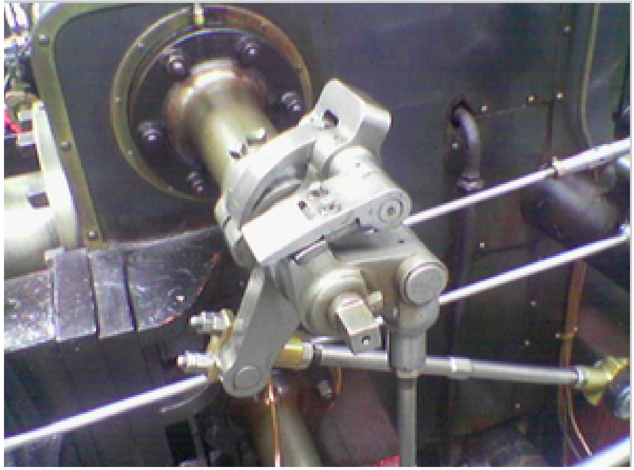
$$
\begin{aligned}
L &= L_a + L_d + L_s \\
&= k_a I_a + k_d\left(\frac{I}{r^2}\right) max(0, n \cdot l) + k_s\left(\frac{I}{r^2}\right) max(0, n \cdot h)^p
\end{aligned}
$$

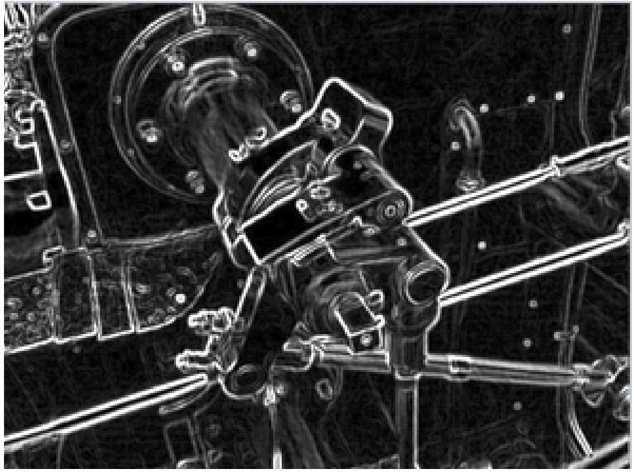
Appendix

- **Color mapping with mesh II – border masks**

  image border detection: Sobel filter in image space

  depth discontinuities detection(border between 3D model and background): Sobel filter applied to depth map





Appendix

- **Color mapping with mesh III – parameterization**

  Transform decomposition: T = T1 * T2

  Parameterization: To find the mapping between mesh and texture, **with angle and area preservation**.

  M: 2-manifold(mesh), D: 2D domain(texture), E: interpolation domain