Out-of-Core Surface Reconstruction via Global TGV Minimization

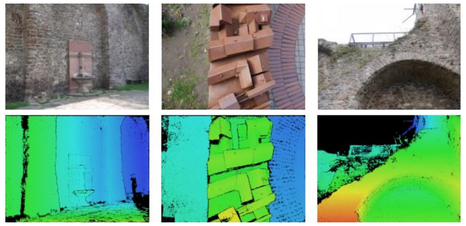
Nikolai Poliarnyi, Agisoft LLC, St. Petersburg, Russia
polarnick@agisoft.com

Agisoft

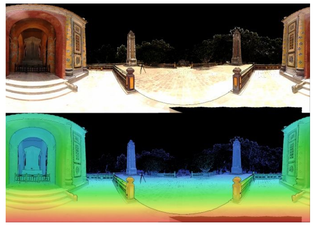
2021 ICCV OCTOBER 11-17 VIRTUAL

Input

- Depth maps reconstructed with methods like **SGM** or **PatchMatch**:

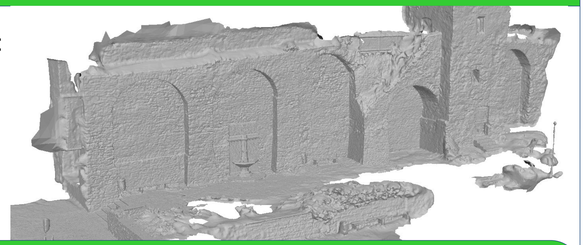


- Terrestrial LIDAR 360-degree scans:



Output

3D polygonal model:



Motivation

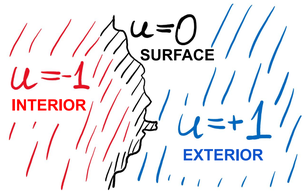
1) **GDMR [1]** has a low memory footprint but is not able to process city-scale datasets (10000+ photos) on the computer with **16 GB RAM.**
2) **GDMR [1]** has two slow stages:
   - **histograms building,**
   - **primal-dual iterations.**
3) **SSR [2]** has strong noise filtering property thanks to **visibility rays**, but is very slow due to computation heavy graph-cut and Delaunay triangulation.

Ideas

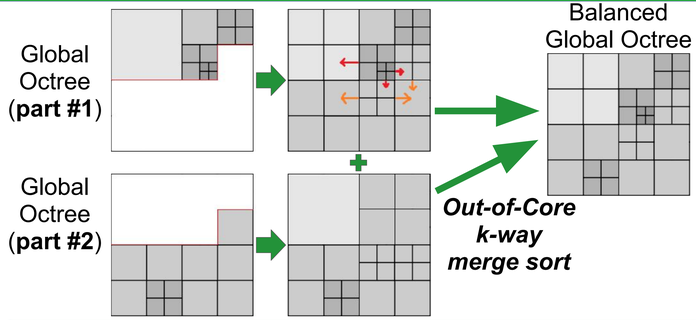
1) **Out-of-Core** method is required (split scene space with octree **treetop**)
2) **Histograms building & primal-dual iterations** - on **GPU via OpenCL**
3) In contrast with **GDMR [1]** we want to exploit strong noise filtering property achieved by respecting **visibility rays**. So we add votes from depth maps not only near the surface, but also on the whole visibility ray

Processing stages

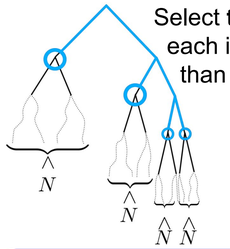
1) Build octree from depth maps
2) Balance octree
3) Build indexed treetop
GPU **4]** Build histograms
5] Iterations to estimate **indicator *u*:**
6) Marching cubes



Out-of-Core? How to balance octree?

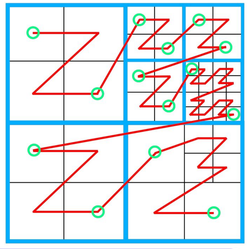


Out-of-Core? How to partition scene space? Treetop!



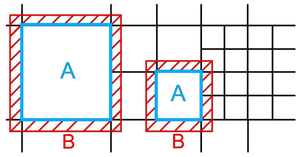
Select **treetop** so that each its leaf will have less than ***N*** cubes in its subtree.

IO-friendly: all cubes from the same treetop's leaf lies sequentially thanks to Morton Codes:



How to prevent seams?

We update the indicator *u* for all cubes inside the current treetop leaf's border (set ***A***) while the indicator *u* for neighboring cubes outside of the border (set ***B***) is frozen. Note that the frozen indicator was upscaled from the previous level of the coarse-to-fine scheme.

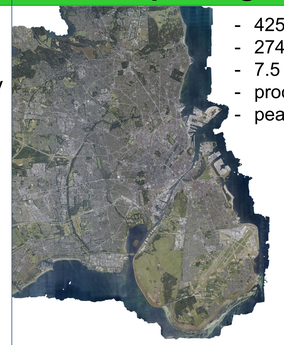


Method properties

Practical properties:
- Out-of-Core
- GPU-accelerated & IO-friendly
- Cluster-friendly
- Support terrestrial LIDAR

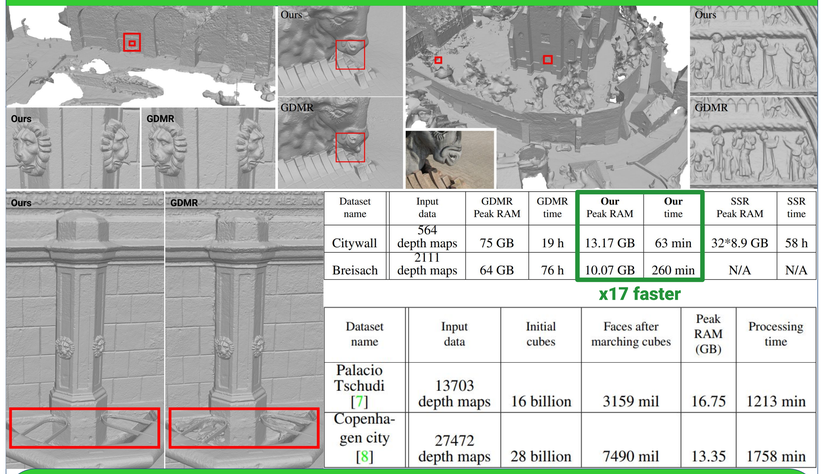
Quality properties:
- Scale-diverse
- Strong visibility-based noise filtering
- Seamless surface

Copenhagen City



- 425 km²
- 27472 photos (566 GB)
- 7.5 billions of triangles
- processed in 29 h
- peak RAM: 14 GB

Results



Dataset name	Input data	GDMR Peak RAM	GDMR time	**Our** Peak RAM	**Our** time	SSR Peak RAM	SSR time
Citywall	564 depth maps	75 GB	19 h	13.17 GB	63 min	32*8.9 GB	58 h
Breisach	2111 depth maps	64 GB	76 h	10.07 GB	260 min	N/A	N/A

x17 faster

Dataset name	Input data	Initial cubes	Faces after marching cubes	Peak RAM (GB)	Processing time
Palacio Tschudi [7]	13703 depth maps	16 billion	3159 mil	16.75	1213 min
Copenha-gen city [8]	27472 depth maps	28 billion	7490 mil	13.35	1758 min

References

[1] B. Ummenhofer, T. Brox. Global, dense multiscale reconstruction for a billion points, 2015
[2] C. Mostegel et al. Scalable surface reconstruction from point clouds with extreme scale and density diversity, 2017
[3] C. Zach et al. A globally optimal algorithm for robust TV-*L*1 range image integration, 2007