geofront

VISUAL COMPUTING

geofront is an Austrian R&D company for computer vision and general purpose computation on graphics hardware (aka GPUs).

Landscape Heatmap Simulation: Combining Photogrammetry & WebGL-based GPGPU in the texture domain

We develop novel cutting-edge algorithms, applying real-time graphics concepts to computer vision tasks and other spatial signal processing, such as light wavefront simulation, video compression and computer tomography.

We supply contracted research and software development, leading engineers to implemented solutions. Services include task assessment and white board advice on solution approaches in workshops. These can also be combined with training for GPU development.

Finally, our on-demand advice supports your engineering staff during their graphics

Business Background

- geofront e.U., privately owned, was founded in January 2016.
- Algorithmic consulting and training in general purpose GPU computing.
- Main focus on **computer vision** and visual computing, e.g. GPU-based computer vision algorithmics.
- Spatial computing as well, e.g. volume compression or tomography reconstruction.
- Occasionally: **Open investigations.**

geofront
VISUAL COMPUTING

OPEN INVESTIGATIONS: PURPOSE

- Create new algorithms and technologies
  as part of geofront's algorithmic portfolio.
- **GPGPU mixed with Real-time game graphics techniques**
  (e.g. shadow mapping, Z buffer, data parallelism)
- Extend practical multi-view camera system knowledge
  (Image Based Reconstruction and Rendering)
- Computer Vision on Mobile (WebGL, Android)

TASK SETTING: SUNLIGHT SIMULATION FOR LANDSCAPES

- Input: Landscape 3D model, GPS position
- How many hours per day (per month, per year) is the landscape and its buildings in light?
- Use: Solar Panel planning (in „hard“, e.g. half-occluded areas)
- Use: Construction planning (do lower floors have enough sunlight?)

LANDSCAPE ACQUISITION: PRACTICAL PROCESS

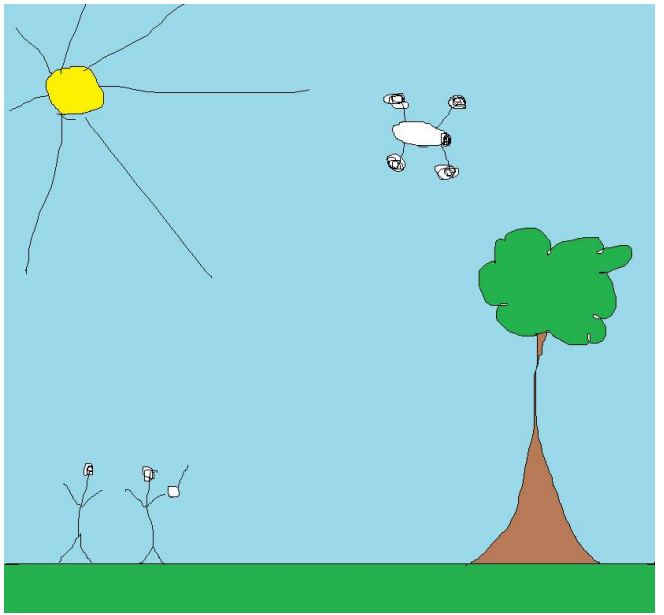
- Location: Porto San Giorgio, Italy
  https://en.wikipedia.org/wiki/Porto_San_Giorgio
- Remnants of the City walls

LANDSCAPE ACQUISITION: PRACTICAL PROCESS

How professional
we
thought we were

Parrot Bebop2
1920x1080 Video

15 minute long flight



How professional
we
really were...

LANDSCAPE ACQUISITION: PRACTICAL PROCESS

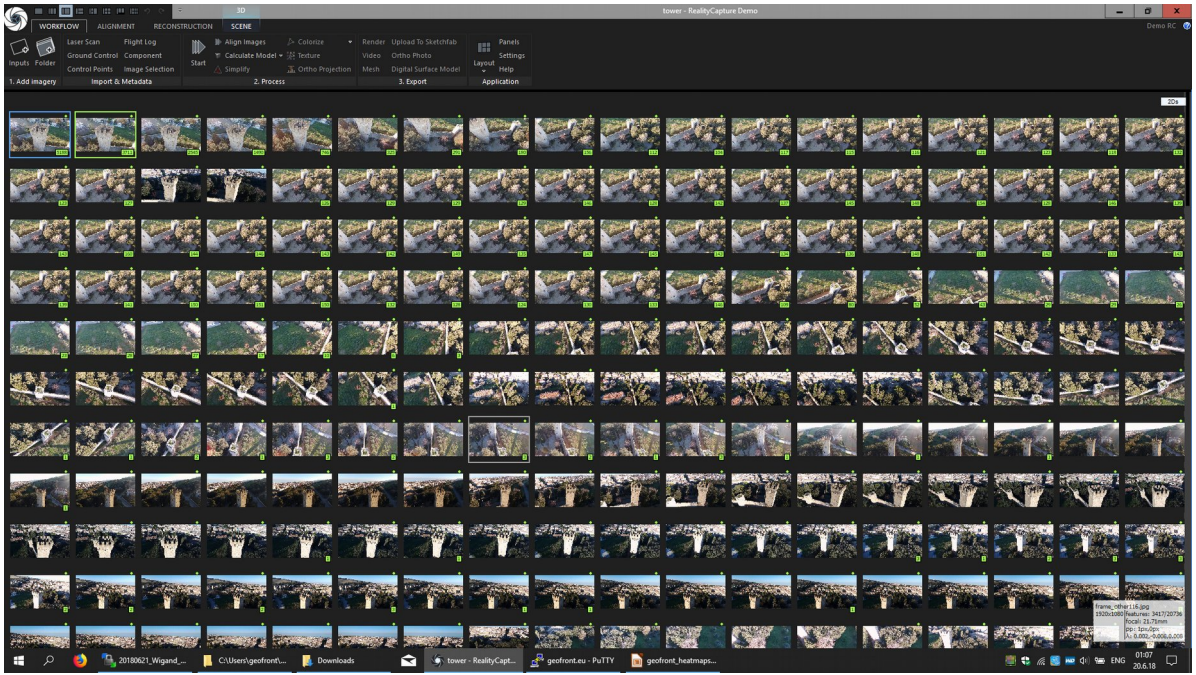
Example video frames:

How to get 3D model from drone footage? -> Photogrammetry!

LANDSCAPE ACQUISITION: PRACTICAL PROCESS

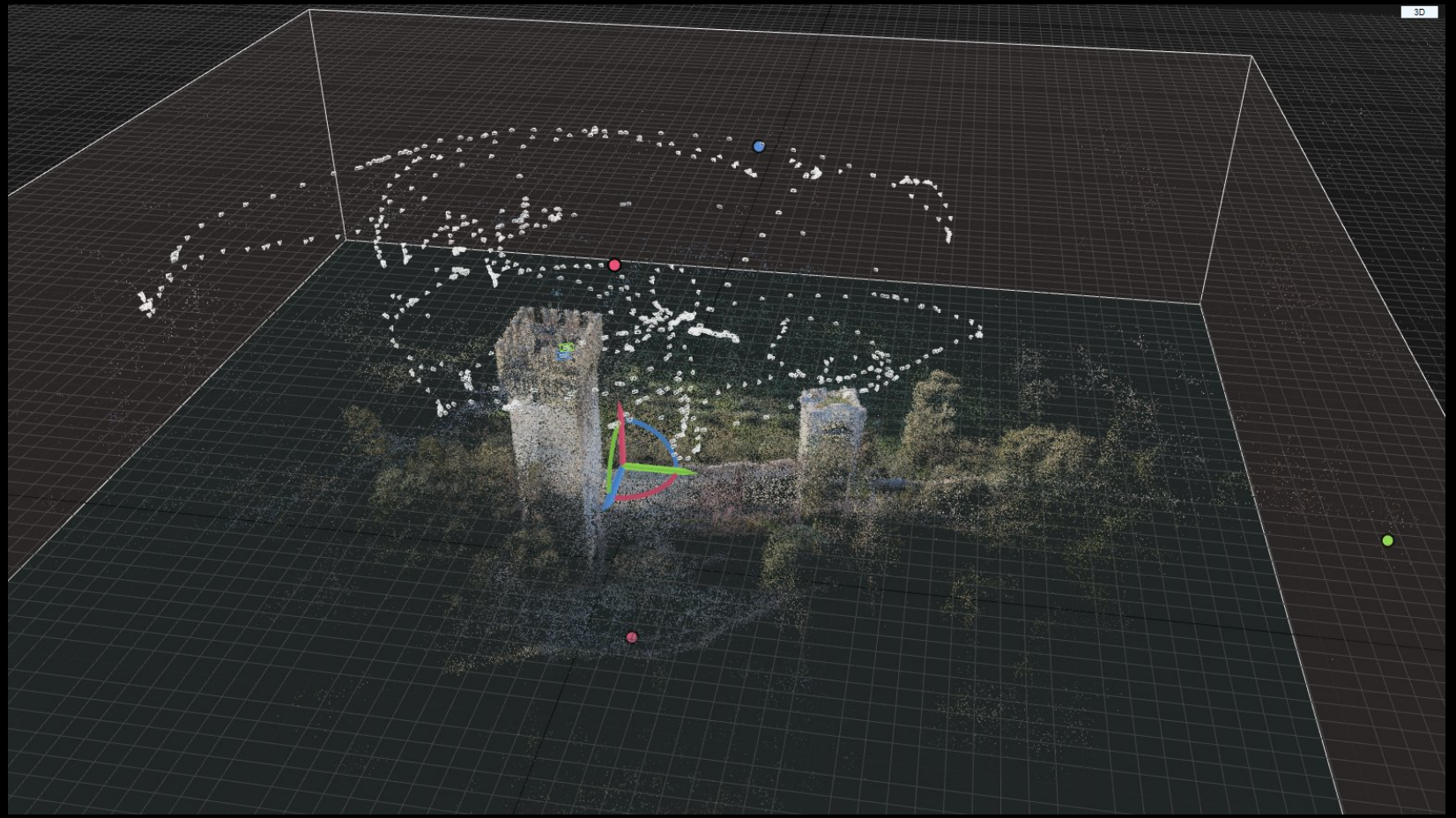
- RealityCapture [1], pro-level photogrammetry software from Bratislava, uses GPS information and photo correlations to:
  - reconstruct camera positions
  - sparse points clouds
  - dense 3D mesh (textured).
- GPU-Compute some minutes.
- Photogrammetry software reconstructs texture from camera views
  (assuming photoconsistency - color agreement between camera views)

[1] http://www.capturingreality.com



- RealityCapture, fed with 918 video frames.

Photogrammetry: Result

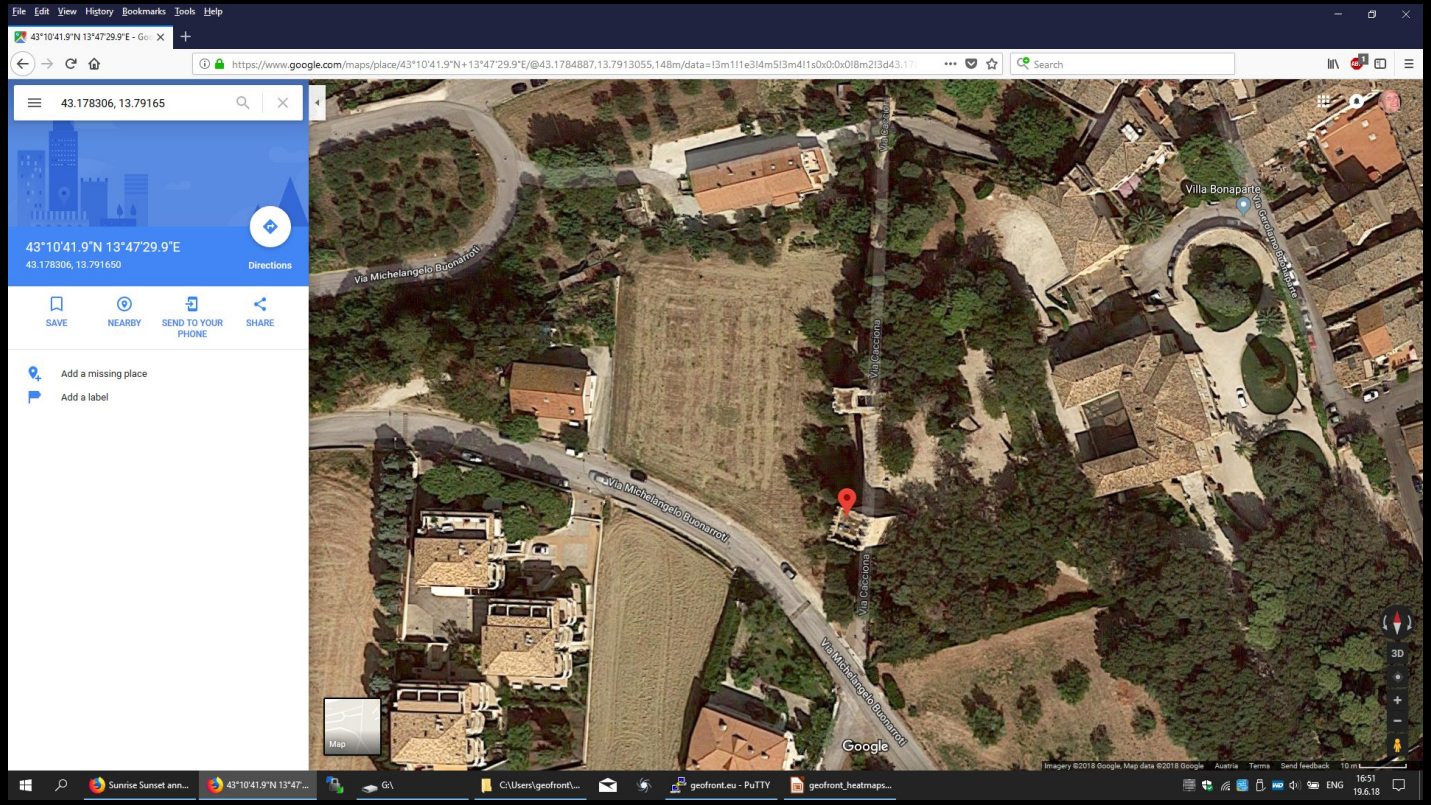


GRAPHICS WITH WEBGL AND HTML5

- Now bring textured 3D model of landscape on screen
- We use WebGL to ensure platform portability (mobile, desktop, even VR)
- Use Canvas element in HTML5 -
  but instead of a 2D drawing context, acquire a 3D (WebGL2) context
- WebGL2 is programmed with an OpenGL-like API in Javascript.
- **Draw it! [DEMO]**
- **Side Problem: Landscape Mesh is quite large ((300k vertices, with texcoords - 27 Megabytes, 4 Megabytes texture) -> massive download for every page view.**
  Solved: Google Draco differential mesh compression: 1 Megabyte
- Landscape Color Texture:
  4 Megabytes for 4096x4096 (could be increased to 8K).
- **Now where is the Sun ?**

GEOGRAPHICAL LOCATION

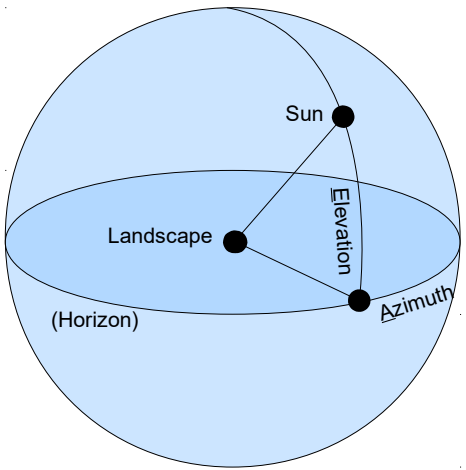
- GPS position from drone (Latitude: 43.178306, Longitude: 13.79165)
- Verification via Google Maps https://goo.gl/maps/3QgVPDkiy4T2:

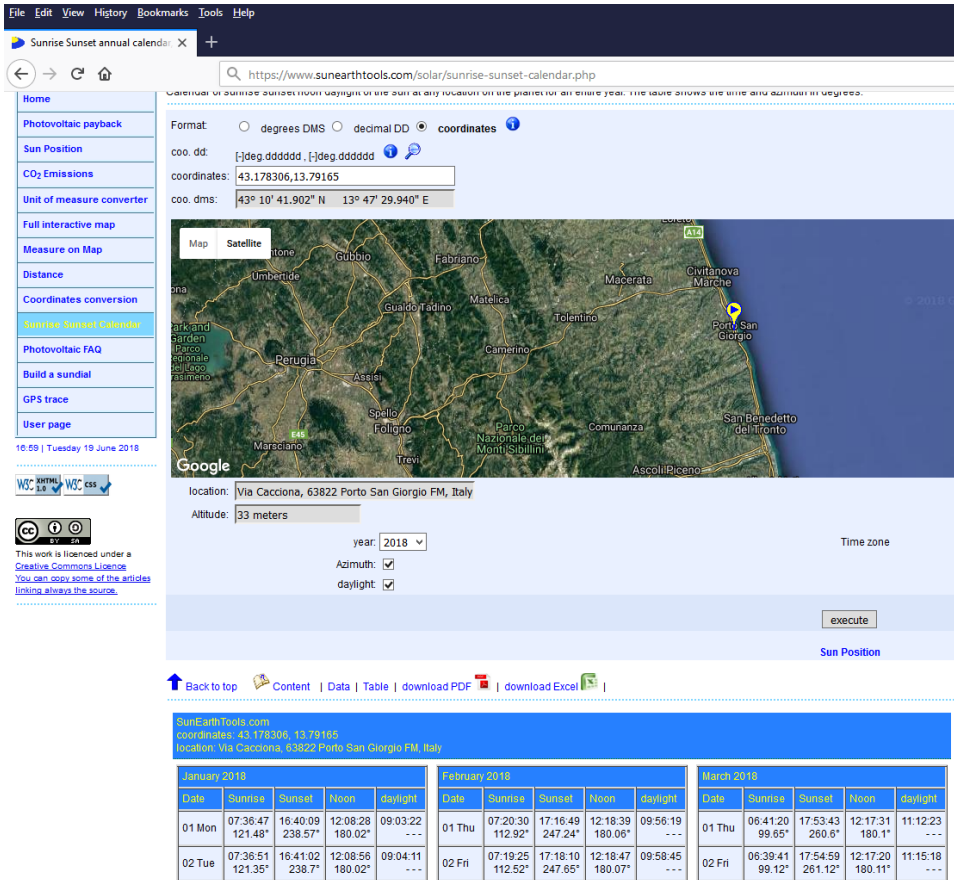


SUN POSITION TABLE

- SunEarthTools [1] offers sun position table given the GPS position from drone:
  Longitude: 43.178306, Latitude: 13.79165
- We used CSV file format, with
  sun positions of 2017 at 5 min granularity:
  A(zimuth) and E(levation)

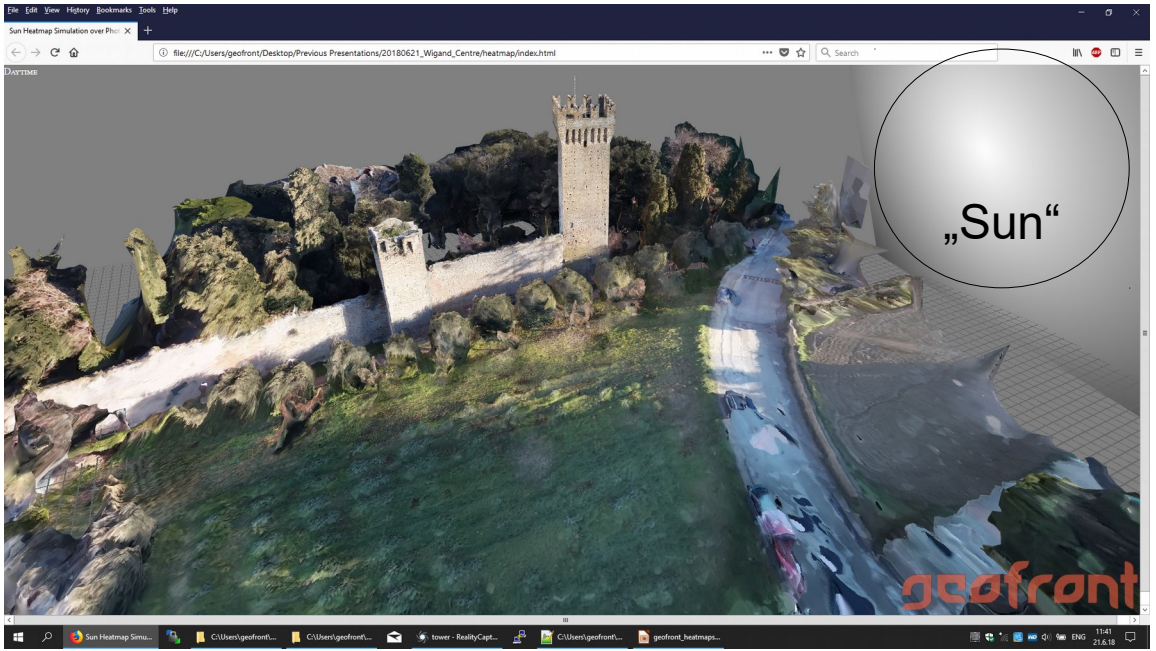
  [1] www.sunearthtools.com





OVERVIEW APPROACH TO HEATMAP (1/2)

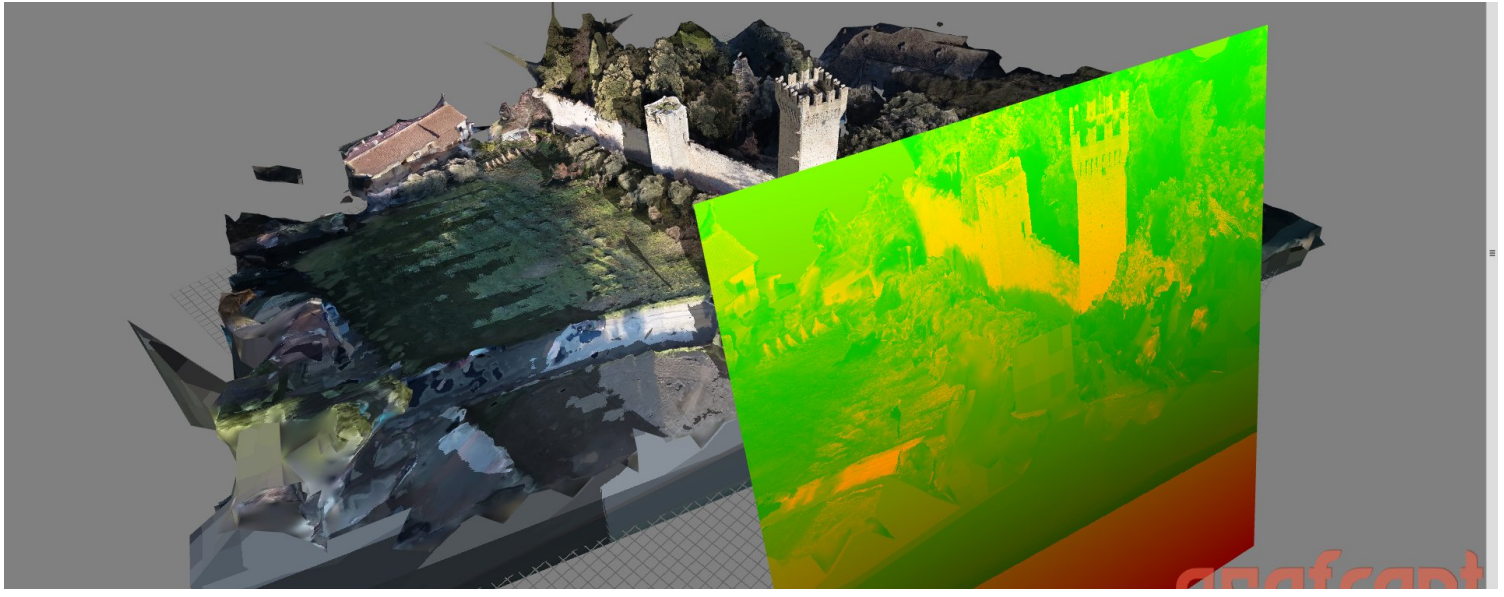
- Subdivide sun exposure into time steps, simulate sun position
- [DEMO of SUN positions moving, use „+“ and „-“]



Now: Find out how the landscape is exposed to sun over time

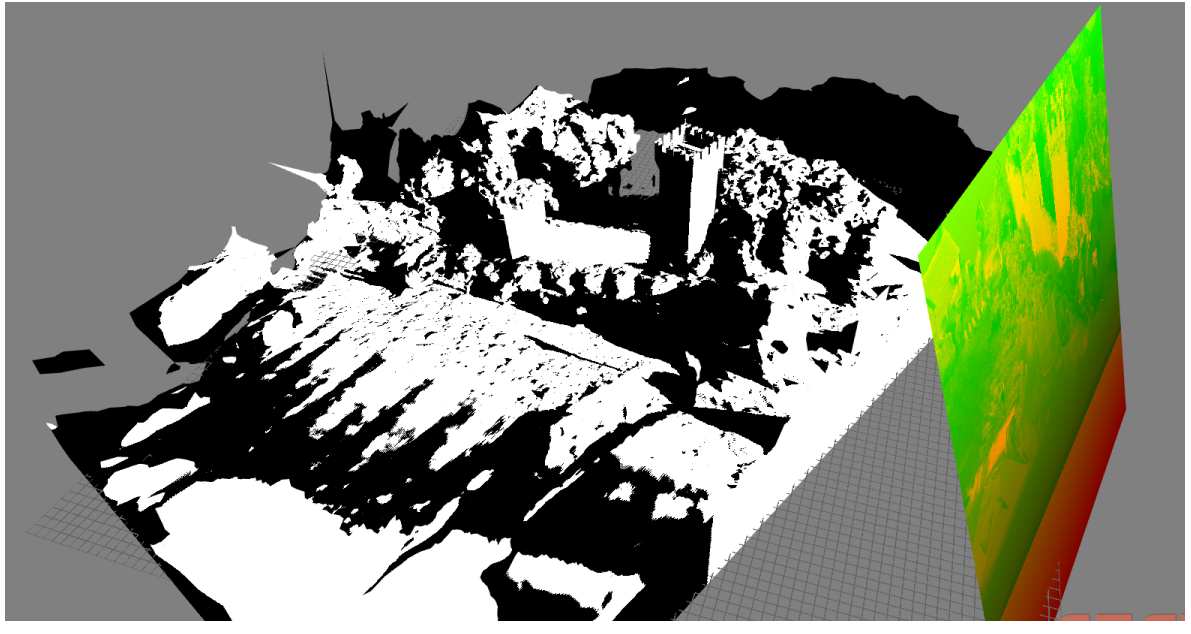
SHADOW MAPPING COMPUTES SUN VISIBILITY!

- How to determine that a rendered pixel is in shadow or not?
- Shadow computation in rendering is a **visibility computation.**
- First step: Check if pixel is visible from light source view!
  -> Render depth map from light source „virtual camera“ (sun: orthocam)
  [DEMO and below: showing color, not depth, as from light source]



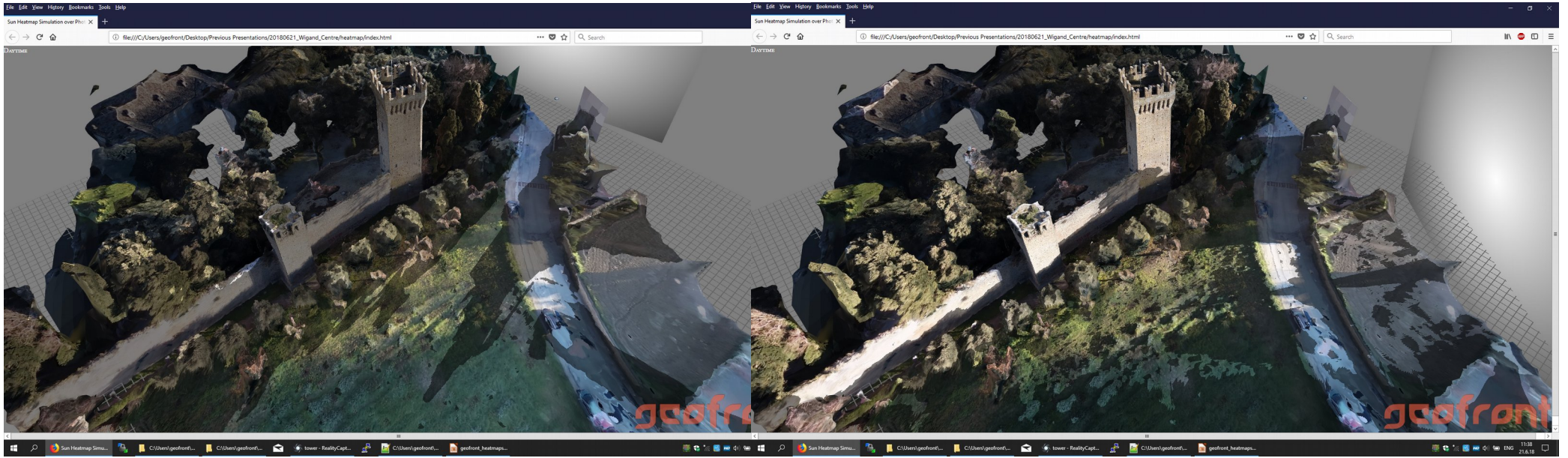
SHADOW MAPPING COMPUTES SUN VISIBILITY!

- Second step: While rendering scene, transform each pixel´s 3D position into light source „virtual camera“ projection space.
- Compare shadow depth map value with pixel´s depth value.
  if (near) identical: in light
  if smaller: pixel is not visible to light source -> **pixel in shadow**!



geofront

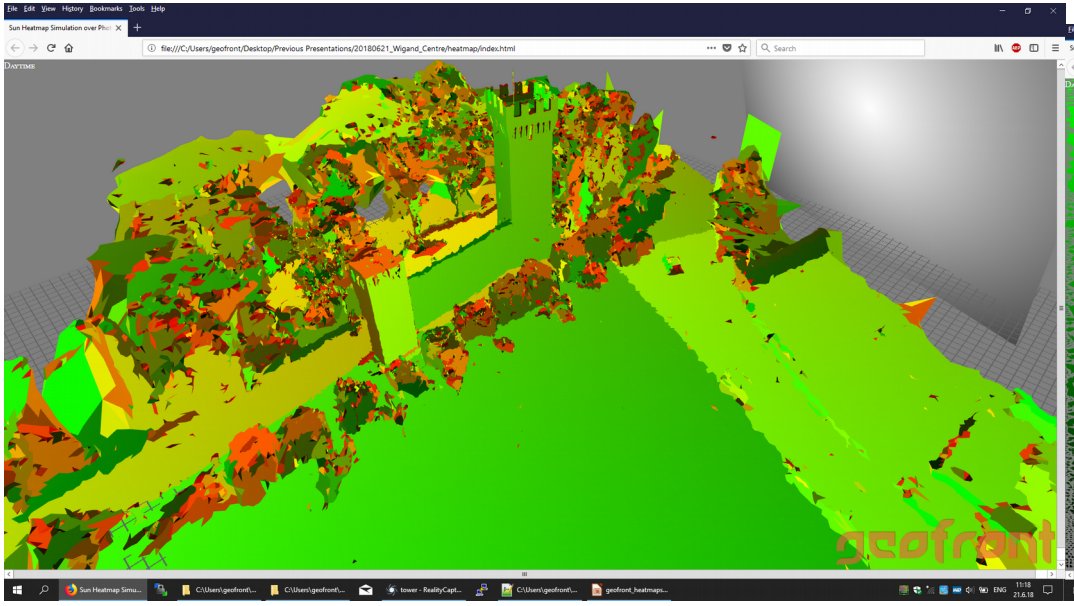
OVERVIEW APPROACH TO HEATMAP (2/2)



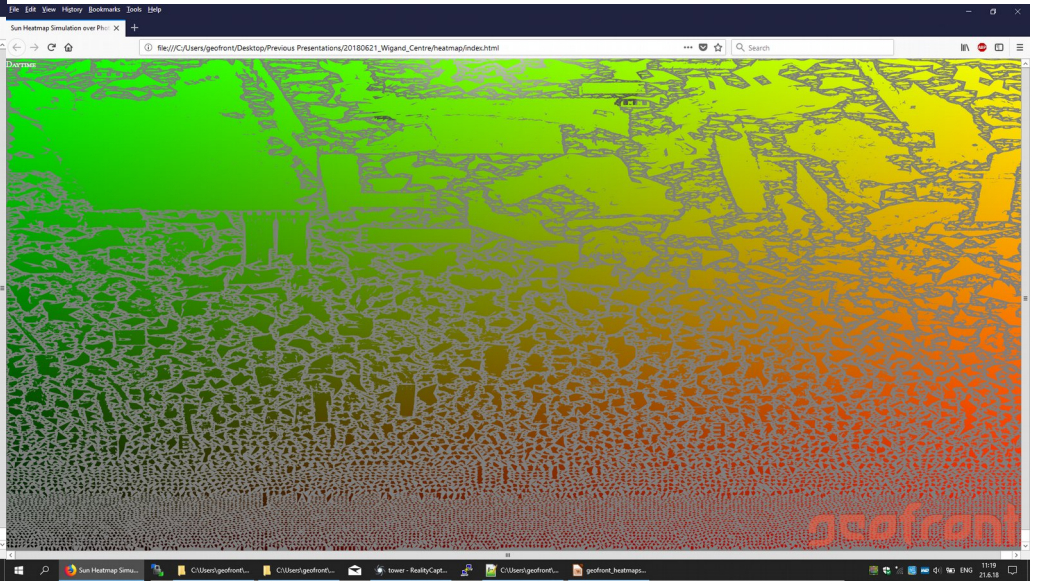
- Good, now "Accumulate" sun shadow over several sun positions
- **Problem: Shadow mapping only computes "sun visibility" for user-visible pixels! (wouldn´t be able rotate final heatmap)**
- **Solution: Move to heatmap computation to texture domain!**

WHAT IS THE TEXTURE DOMAIN

- Textures are a way to color 3D meshes. The mesh has a texture parameterization (2D texture coordinates per vertex) - during rendering, every pixel is a lookup from a color value.



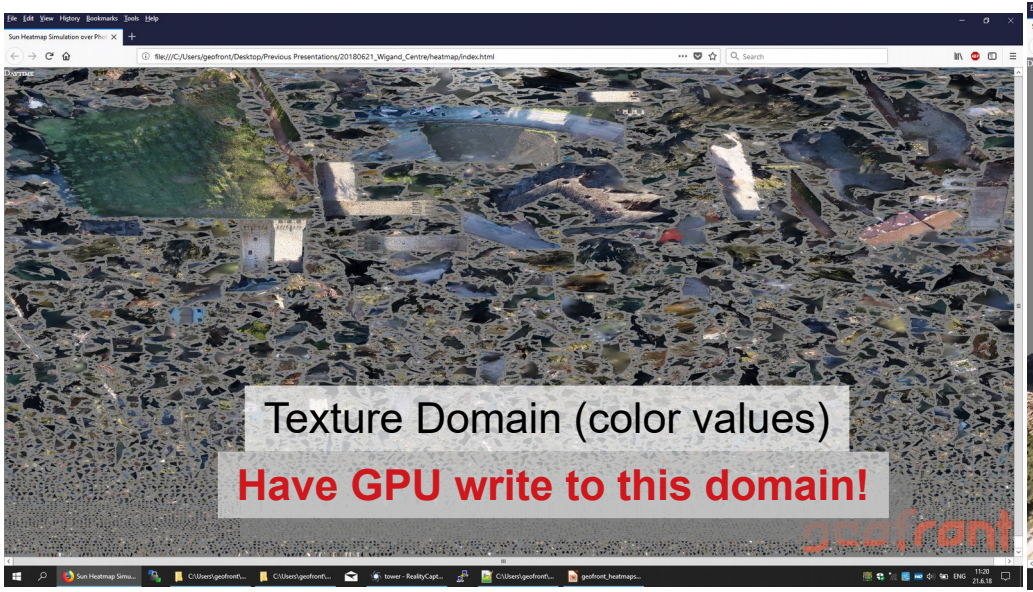
View Domain
(texture coordinates as red-green)

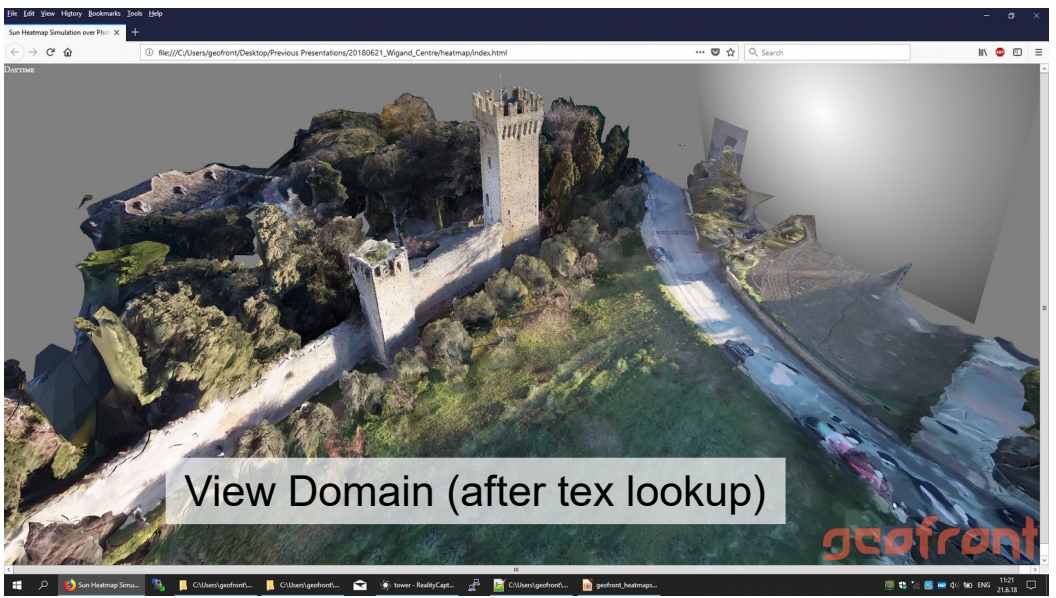


Texture Domain
(texture coordinates as red-green)

WHAT ARE TEXTURES

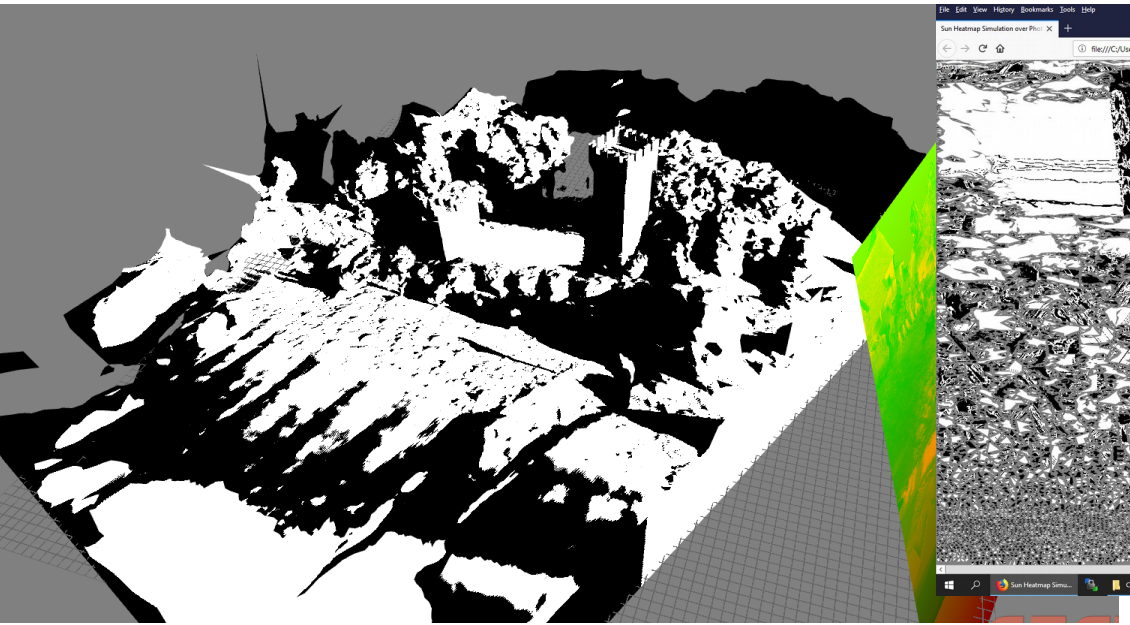
- Textures are a way to color 3D meshes. The mesh has a texture parameterization (2D texture coordinates per vertex) - during rendering, every pixel is a lookup from a color value.
- BTW: Photogrammetry software reconstructs texture from camera views (assuming photoconsistency - color agreement between camera views)



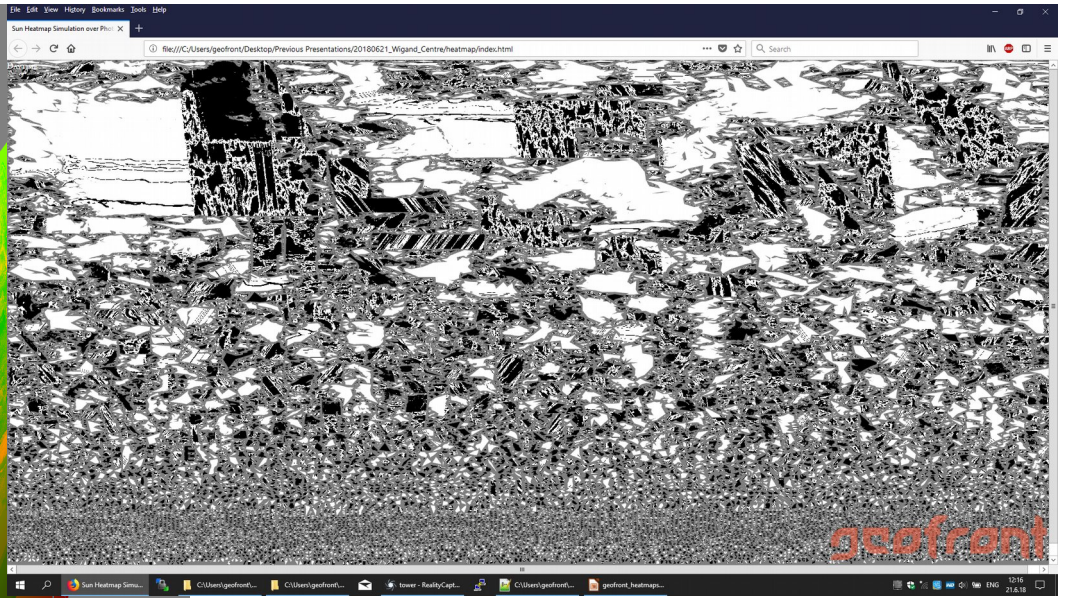


WRITE-VISIBILITY- TO TEXTURE

- Render all landscape mesh triangles INTO texture domain, where we compute sun visibility for current sun position
- Output is a visibility value (0.01 or 0.0) for the current sun position. (similar to shadow)



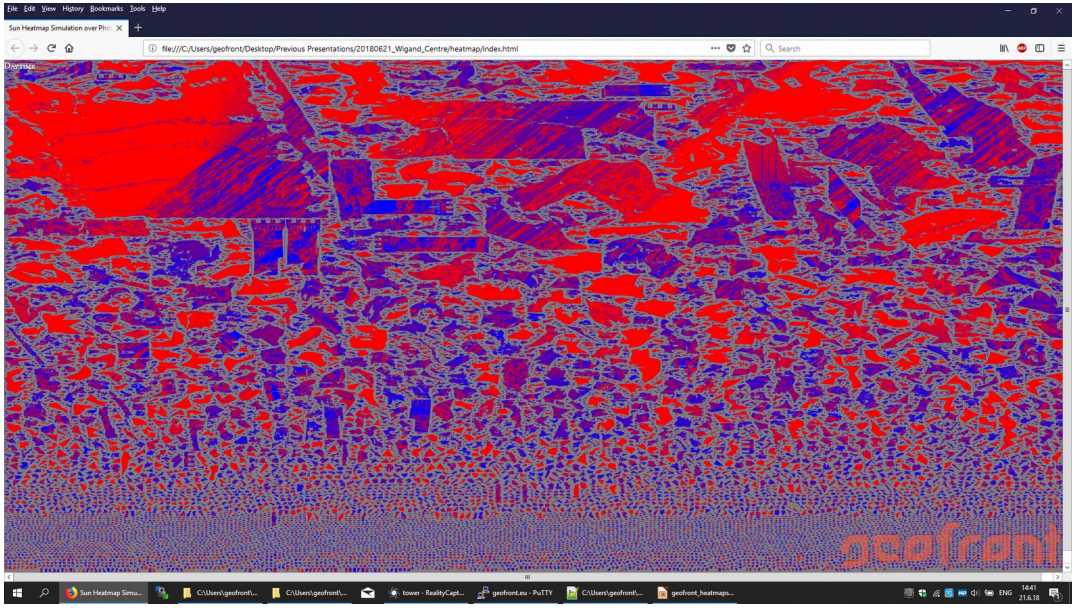
View domain (from before)



Texture domain

IMPLEMENTATION ALGORITHM

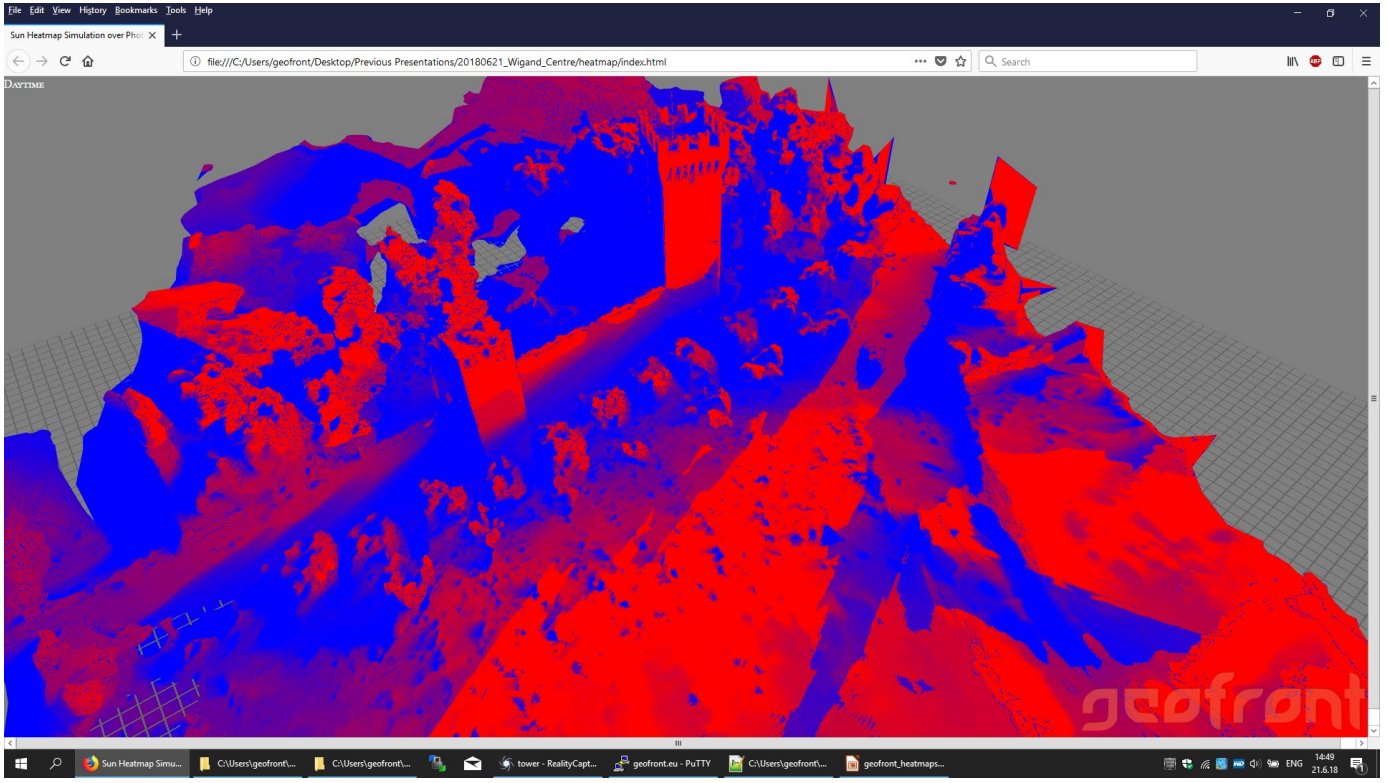
- „Heat accumulation“
  from (start time:end time),
  repeatedly landscape mesh triangles into texture domain,
  using OpenGL blend mode to add visibility value to framebuffer value
  (= accumulated values in heatmap texture) gl.blendFunc(gl.ONE, gl.ONE);
  Shadow map test: glsl/render_heatmap.glsl



- Heatmap, texture domain in false colors (blue: low, red: high)

IMPLEMENTATION RUN

- [Final DEMO, heatmap uses false colors blue: low, red: high]



- **DONE.**

CONCLUSION

- CapturingReality produces usable 3D landscape models with photogrammetry, even from video frames of hobbyist drone flights.
- WebGL works great to visualize photogrammetry results
  (Draco helps with mesh compression)
- Sun exposure over time can be simulated using
  - GPU-based shadow mapping and
  - GPGPU-like accumulation of „daytime sun exposure samples“ by using (a) the texture parameterization of the 3D model and
    (b) rendering into the texture domain.
- **GPGPU is possible in WebGL**
  (and GPGPU thus portable between mobile (Android, iPhone...)/desktop/VR).

CHALLENGES AND FUTURE WORK

- „Compute away“ original shadow in the landscape
  (or record footage on a cloudy day)
- Modify sun´s light flux over day and season
-
- Come up with more applications where
  - self-occlusion of meshes (only fast on GPU) and
  - repeated visibility/shadowing computations (only fast on GPU)
    are useful!

Let me know if you have any suggestions!

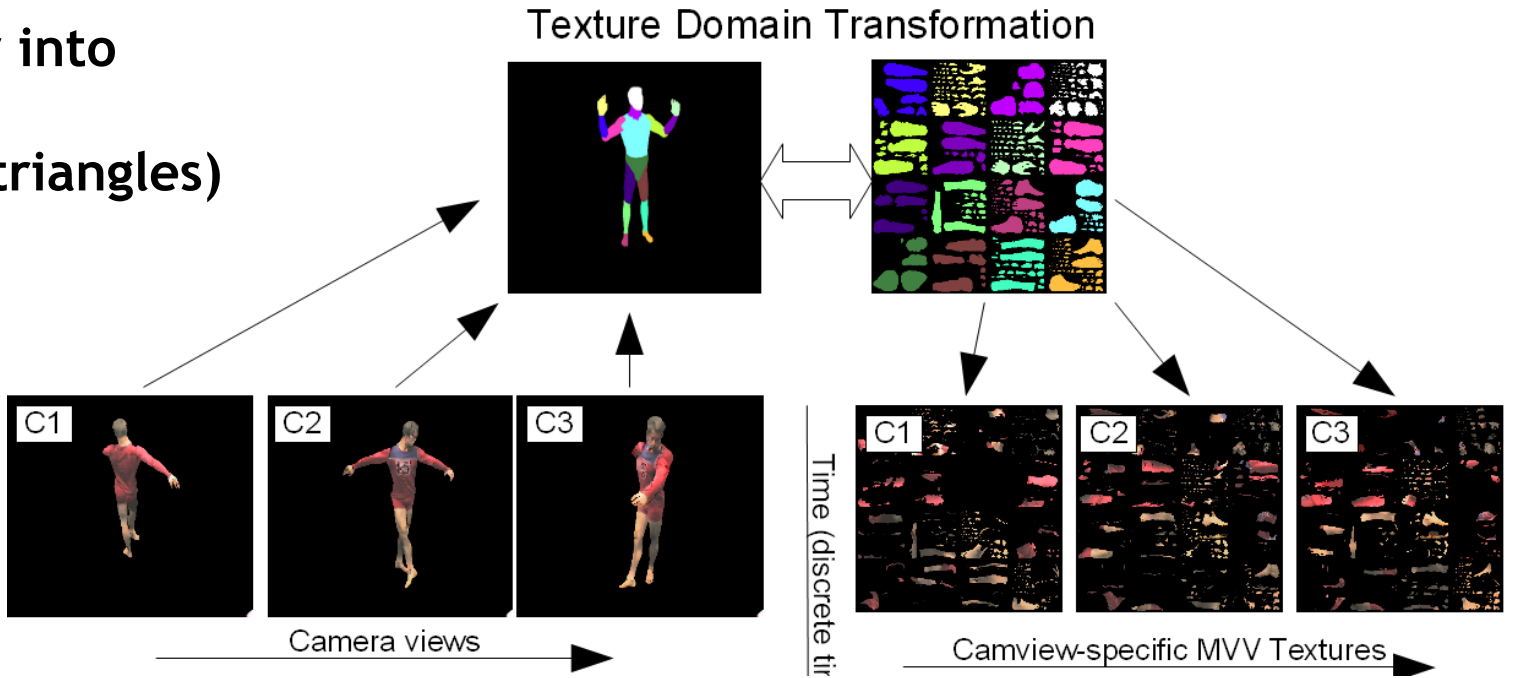
Thank you.

Questions?

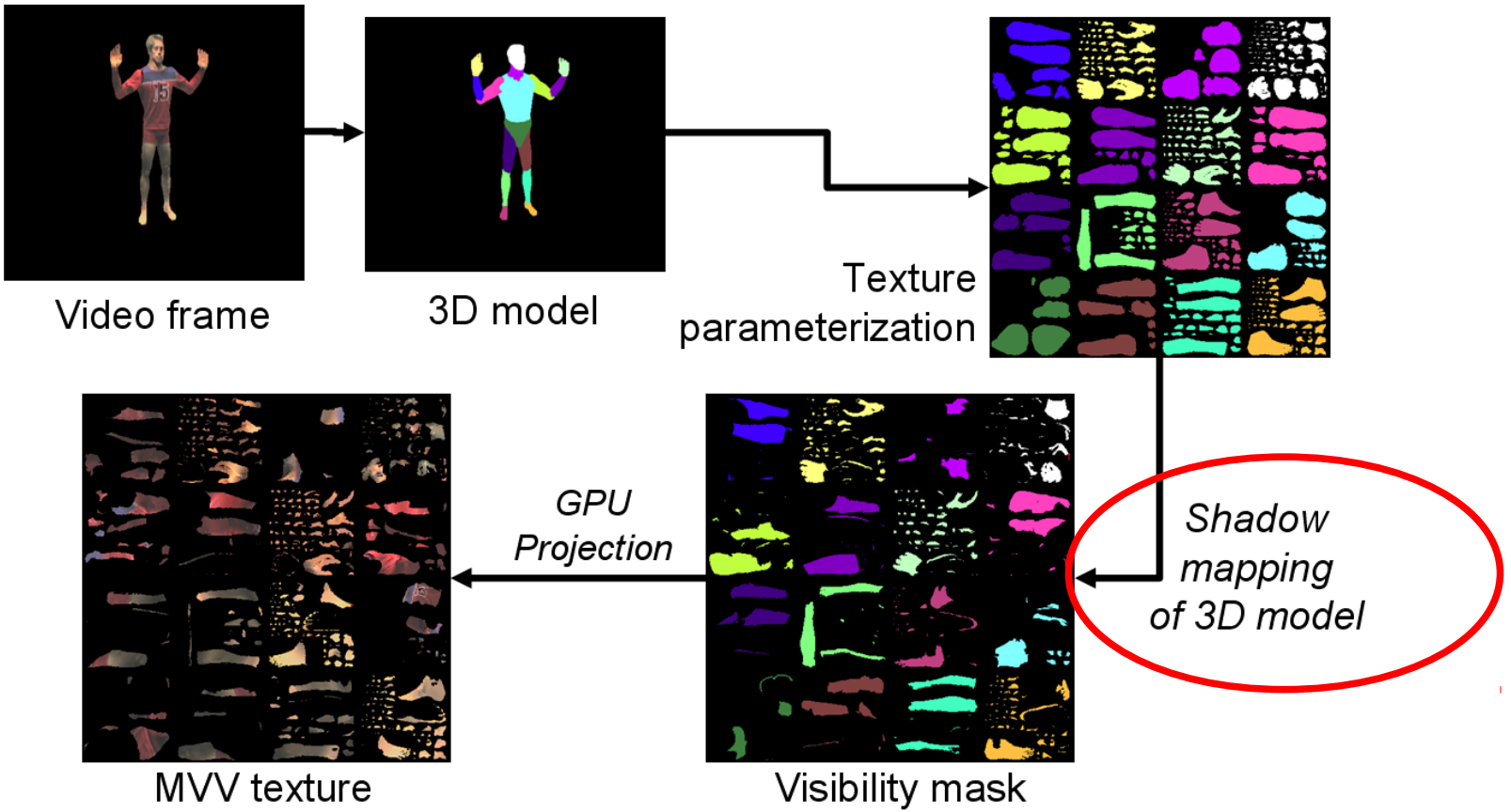
geofront.eu/demos/heatmap

INSPIRATION: VISIBILITY IN MULTI VIEW VIDEO

- 3D model tracked from multiple camera views
- Texture coordinates mapping present
  (Triangles can be mapped and rendered to texture domain)
- Camera visibility determined by shadow mapping
- **Render visibility into texture domain (using 3D mesh triangles)**



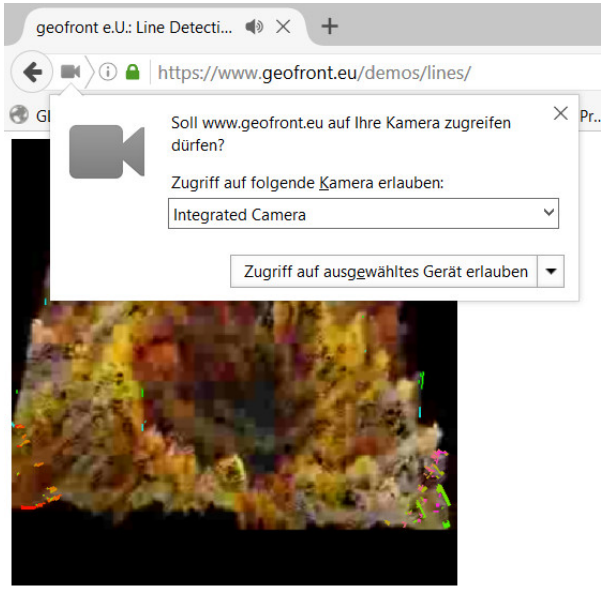
INSPIRATION: VISIBILITY IN MULTI VIEW VIDEO



- IDEA: Use the same approach for landscape heat maps !

OTHER PROJECTS

PROJECTS „WEBGL COMPUTER VISION„



- HTML5 enables real-time video input
- WebGL enables GPU access from Javascript
- Limited (only OpenGL ES level), but 2008 algos for data compaction apply!
  -> can *extract sparse feature lists*, build quadtrees, octrees, **geometry shader**, etc.
- Line detection, Object tracking, …
- Contact me if you are curious and I give you a short intro!

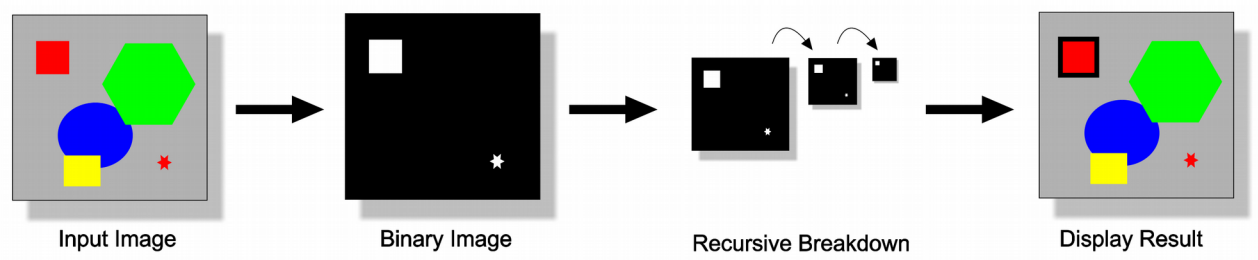
BOUNDING BOX TRACKING IN WEBGL

https://www.geofront.eu/webgl/bboxpyramid

- Quick tracking of largest pixel group of certain color
- Threshold pixels, assume small bounding boxes
- Data Parallel Reduction:
  Merge bounding boxes (if adjacent) OR
  Choose largest one (if competing)

Interactive run times on mobile GPUs

Doesn´t have to be color (e.g.group local motion



Landscape Heatmaps

OTHER DATA-PARALLEL RESEARCH

- HistoPyramids have been extended to create quadtrees and octrees through bottom-up analysis:

  www.geofront.eu/thesis.pdf

- Summed Area Ripmaps fix precision issues of Summed Area Tables / Integral Images:

  http://on-demand.gputechconf.com/gtc/2012/presentations/S0096-Summed-Area-Ripmaps.pdf
  (Note for implementation: US patent filed!)

- Connected Components using full GPU parallelism:
  http://on-demand.gputechconf.com/gtc/2013/presentations/S3193-Connected-Components-Kepler.pdf