Universität Rostock Traditio et Innovatio

Project CSI WS 2025/26

Visual Computing

Prof. Dr. Oliver Staadt
MSc Sven Kluge
MSc Max Kröger

Goals

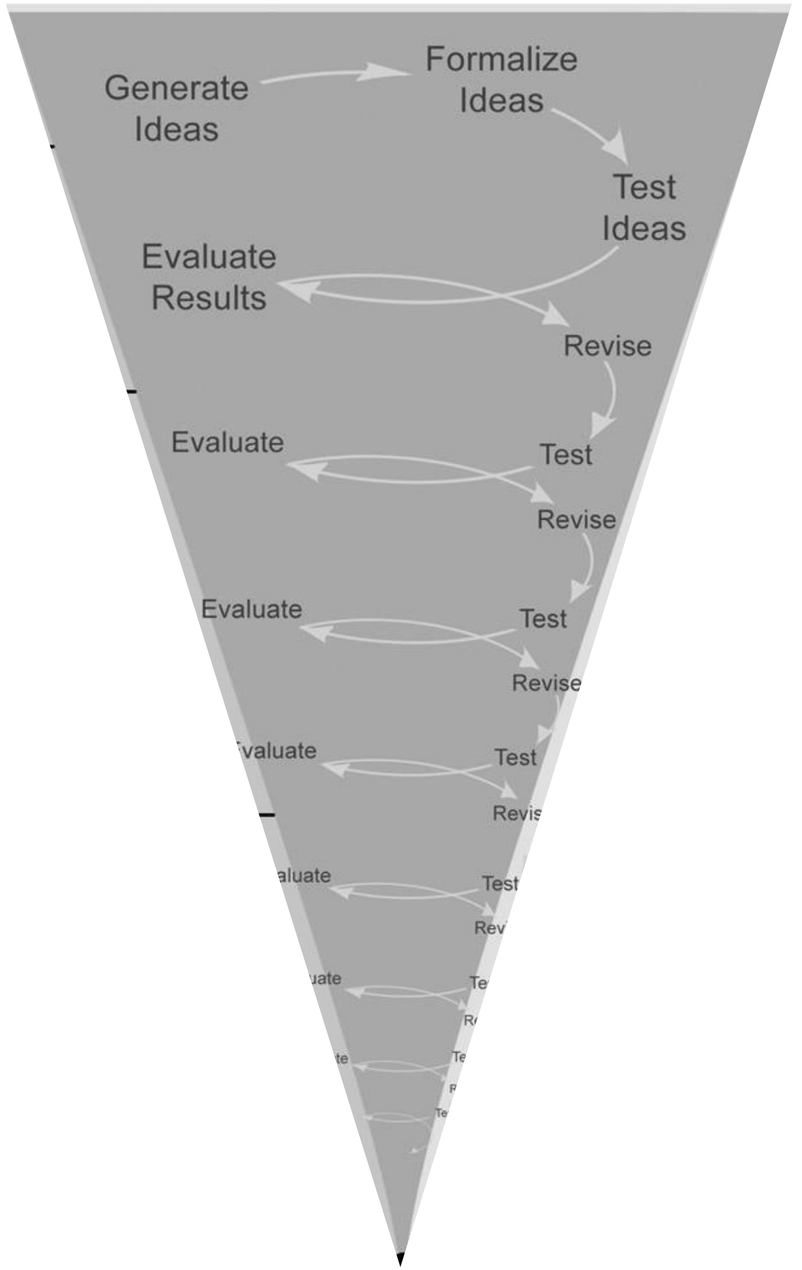
- Design and implementation of a system/application in visual computing
- System design and programming
- Practical application of visual computing topics
- Learning soft skills in teamwork

Prerequisites

- Interest in visual computing technology
- Basic knowledge of visual computing (e.g., computer graphics module in the bachelor’s degree, virtual reality, computer vision)
- Enjoyment of teamwork
- Creativity and motivation!

Three phases

1. Design and Planning (25%)
2. Development (50%)
3. Evaluation and Improvement (25%)



Project structure

- The development of a software system is not trivial!
- Project management
  - Software engineering
  - Proposal/Requirements specification
    - Project idea
    - Tasks
    - Schedule
  - Critical discussion

Project structure

- Introduction to tools used (tutorials) and team building (2 weeks, October 30)
- Proposal/specifications document (2 weeks, November 13)
- Interim presentation (4 weeks, December 11)
- Final demo (7 weeks including Christmas break, January 29)

Teams

- Ideally three students per team
- Each team member should contribute equally to the project
- Considerations
  - Interests
  - Skills
  - Working hours
  - Meeting location
  - …

Tools

- Information exchange between teams in the Stud.IP wiki/forum
- Upload documents to Stud.IP (each team has a directory)
- Google Docs for shared documents/spreadsheets
- Unity/Unreal/PyTorch/OpenCV/Jupyter, etc.

Proposal/Specifications

- 1-3 pages description of the project idea
- Sketches
- Justification of design decisions
- Divide tasks into layers
- Detailed schedule, milestones, deliverables

Five layers

- **Functional Minimum:** What needs to be done so that it can be called a functioning system/application?
- **Minimum Goal:** What is necessary to be reasonably “okay” with the result?
- **Desired Goal:** What should be done if everything goes according to plan?
- **Maximum Goal:** What additional things can be done if everything goes much better than planned?
- **Extras:** What could be done if there were more time than one semester?

Task list

Task	Description	Who	Hrs	Actual
1	Brainstorm design	All	4	8
2	Character modeling	Stan	12	26
3	Camera control	Kyle	6	
4	Prepare presentation	All	6	
5	Explosion effect	Kenny	12	

Schedule

Task	Wk1	Wk2	Wk3	Wk4		Wk5	Wk6	Wk7	…
1	A				Part 3 Due				
2		L	L						
3			T						
…									

Think Big?

Think Small!!!

Think Small

Better to do one thing right than many things mediocre

Intermediate presentation

- Layer 2 completed
- Layer 3 in progress
- Functional minimum must be present!
- Short presentation/demo
- Feedback to other projects/teams

Final demo

- Last lecture week
- Demos of all teams

AI Policy:
If you use AI (e.g, LLMs), disclose how, what, and where you use it

Questions?

Project Ideas

Kerbl, Bernhard, et al. "3D Gaussian splatting for real-time radiance field rendering." *ACM Trans. Graph.* 42.4 (2023): 139-1.

Task: Create a 3D Gaussian Splatting scene of an environment of your choice.

Project: Gaussian Splatting

Kerbl, Bernhard, et al. "3D Gaussian splatting for real-time radiance field rendering." *ACM Trans. Graph.* 42.4 (2023): 139-1.

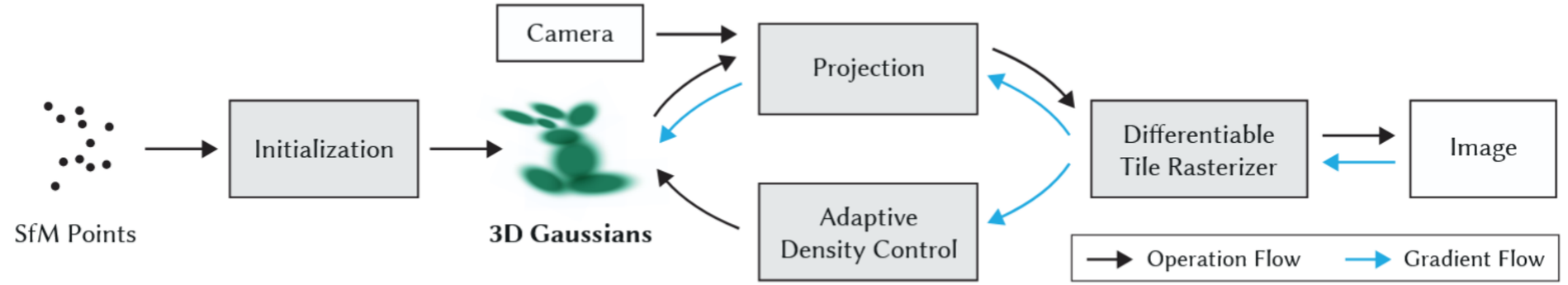
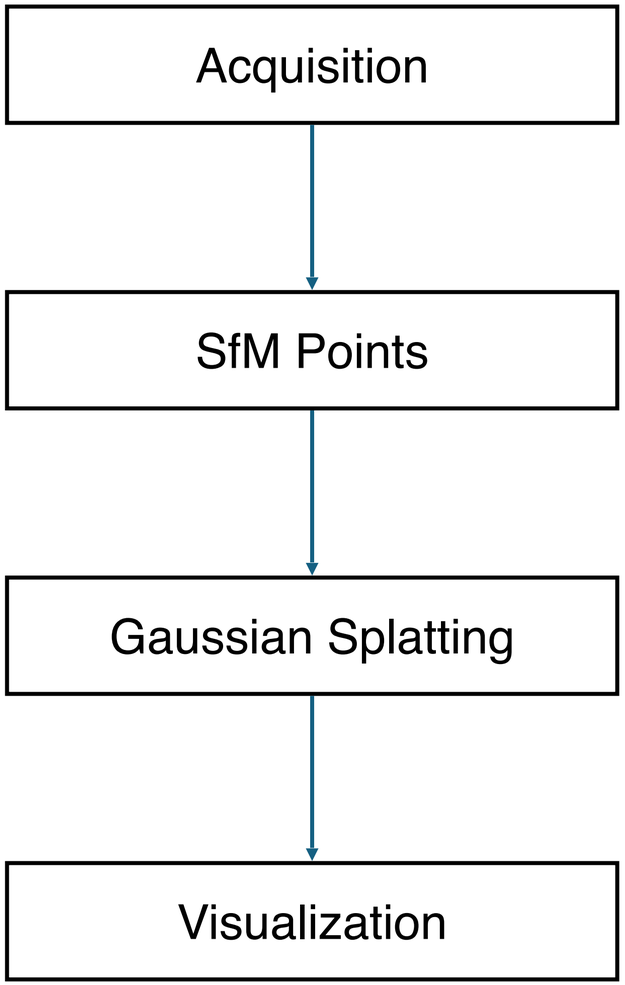


Fig. 2. Optimization starts with the sparse SfM point cloud and creates a set of 3D Gaussians. We then optimize and adaptively control the density of this set of Gaussians. During optimization we use our fast tile-based renderer, allowing competitive training times compared to SOTA fast radiance field methods. Once trained, our renderer allows real-time navigation for a wide variety of scenes.

3D Gaussian Splatting is an explicit, point-based radiance-field representation that enables real-time novel-view synthesis from calibrated multi-view images by optimizing and rasterizing a set of anisotropic 3D Gaussians

Examples can be found at:
https://splatter.app/
https://poly.cam/explore?type=splat&feed=top

General Workflow:



- Take some photos or a video of a real world scene of your choice (could be a landmark, building, statue or even just your room)

- Use tools like ffmpeg and COLMAP to extract a SfM (structure from Motion) Pointcloud and register the cameras for the views

- Use Gaussian Splatting to create your 3D scene

- Use one of the many gaussian splatting visualization platforms to visualize your scene in realtime

Additional infos:

- An additional goal is that you should understand the basics of how Gaussian Splatting works

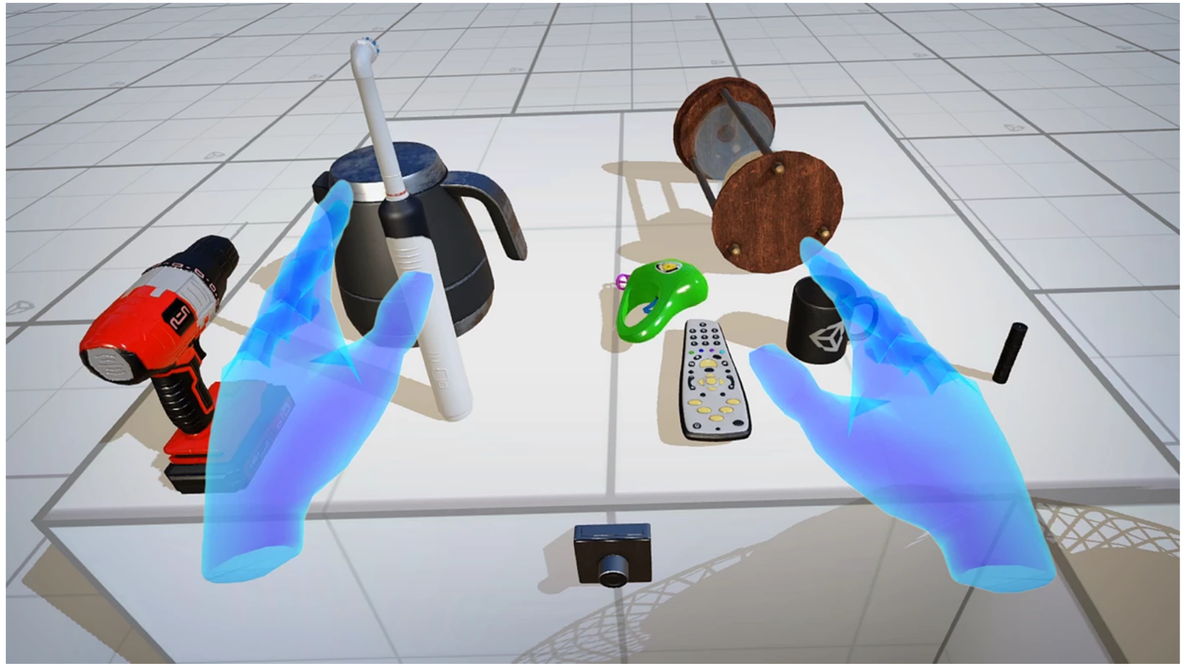
- You will get specific informations which libraries you can/should use

- Gaussian Splatting is well documented
  - There are many Articles, YouTube Videos and Tutorials

- Commercial All-in-One solutions are not allowed
  - Steps must be documented

VR in Unity (or Unreal)

- Create a VR world of your choice
- Use Unity (or Unreal) Engine
- Implement
  - Interaction
  - Locomotion
  - Stereoscopy
  - etc.
- E.g. use Meta XR Simulator



Use MetaHumans in Unity

- MetaHumans: a feature of Unreal Engine
- In the past: MetaHumans only for Unreal
- Now: MetaHumans can be used in other engines (changed license)
- The aim is to have a functional and animated MetaHuman in Unity (e.g. dancing)

Alternative:

- Natively create a MetaHuman of yourself and animate it (Unreal Engine)

You can suggest your own project ideas!