Fachhochschule Nordwestschweiz
Hochschule für Architektur, Bau und Geomatik

Geometrieverarbeitung mit Python

Vom Höhen- und Stadtmodell zur 3D-gedruckten Visualisierung

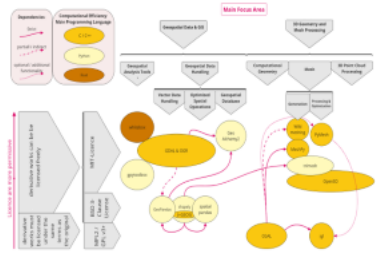
Motivation

- Erst durch die **Visualisierung** werden Geodaten für den Menschen direkt nutz- / interpretierbar
- Vor jeder Nutzung / Visualisierung müssen **Raumbezogene Daten verarbeitet** werden
- Programmiersprachen wie **Python** bieten breite Paletten an Werkzeugkästen (Libraries genannt)
- Besonders im Fachbereich der **Architektur**, aber auch allgemein, trotzt die Beliebtheit **physicher Modelle** neueren virtuellen Möglichkeiten
- Der **3D-Druck** bietet eine moderne, automatisierte Alternative zur Erstellung solcher Modelle

Abb. 1: Titelbild mit diversen 3D-gedruckten Produkten dieser Bachelorarbeit

10. Juni 2024 Autor: Timon Kuhn, Betreuer: Prof. Martin Christen, Matus Gasparik, Experte: Dozent Robert Wüest © IGEO FHNW

Zusammenstellung der Geometrie-verarbeitungs-Libraries in Python



- Abgrenzung unter dem Begriff Geometrie**verarbeitung**, also der **Manipulation** der Geometrien selbst
- **Weglassen** aller Libraries deren Funktionalität beschränkt ist auf:
  - lesen und schreiben von raumbezogenen Daten
  - Visualisieren von Geometrien
  - Analyse und Informationsgewinnung
  - Koordinatentransformation
- Viele **Abhängigkeiten** untereinander sind gegeben

10. Juni 2024 Autor: Timon Kuhn, Betreuer: Prof. Martin Christen, Matus Gasparik, Experte: Dozent Robert Wüest © IGEO FHNW

10. Juni 2024 Autor: Timon Kuhn, Betreuer: Prof. Martin Christen, Matus Gasparik, Experte: Dozent Robert Wüest © IGEO FHNW

Geometrieverarbeitung

- **Standard-Workflow** auf der linken Seite beschreibt den Weg **vom DGM und Stadtmodell zum STL** (siehe Basel)
- Diese **Funktionalität** wurde in einem **Python-Modul implementiert**
- Weitere Optionen durch weitere Funktionalitäten / andere Daten möglich:
  - DOM anstatt DGM + Gebäudemodell (siehe Zürich)
  - Andere Ausdehnung (siehe ganze Schweiz)

Grundlagedaten:
- .tiff: swissalti3D, swissSurface3D Raster, DHM25/200
- .obj: Stadtmodell Basel

Genutzte Geometrieverarbeitung Libraries:
- Polygone: shapely, Geopandas
- Punktwolken & Mesh: open3D, Trimesh
- Raster *lesen*: rasterio, Numpy

10. Juni 2024 Autor: Timon Kuhn, Betreuer: Prof. Martin Christen, Matus Gasparik, Experte: Dozent Robert Wüest © IGEO FHNW

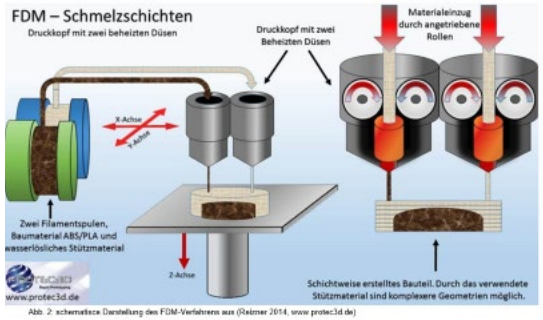


Abb. 2: schematische Darstellung des FDM-Verfahrens aus (Reimer 2014, www.protec3d.de)

3D-Druck

- Ein **Slicer** (Cura) wandelt ein **STL-Modell** in **G-Code** (Maschienensteuerung)
- Der G-Code beschreibt Bewegungen (XY und Z), Temperaturen, usw.
- Gedruckt wird mit PLA und ähnlichen Materialien
- **Slicing** minuten bis Stunden
- **Drucken** Stunden bis Tage
- 2 – 200 Gramm, je nach Modellgrösse und -stärke

10. Juni 2024 Autor: Timon Kuhn, Betreuer: Prof. Martin Christen, Matus Gasparik, Experte: Dozent Robert Wüest © IGEO FHNW

Einschätzungen & Ausblick

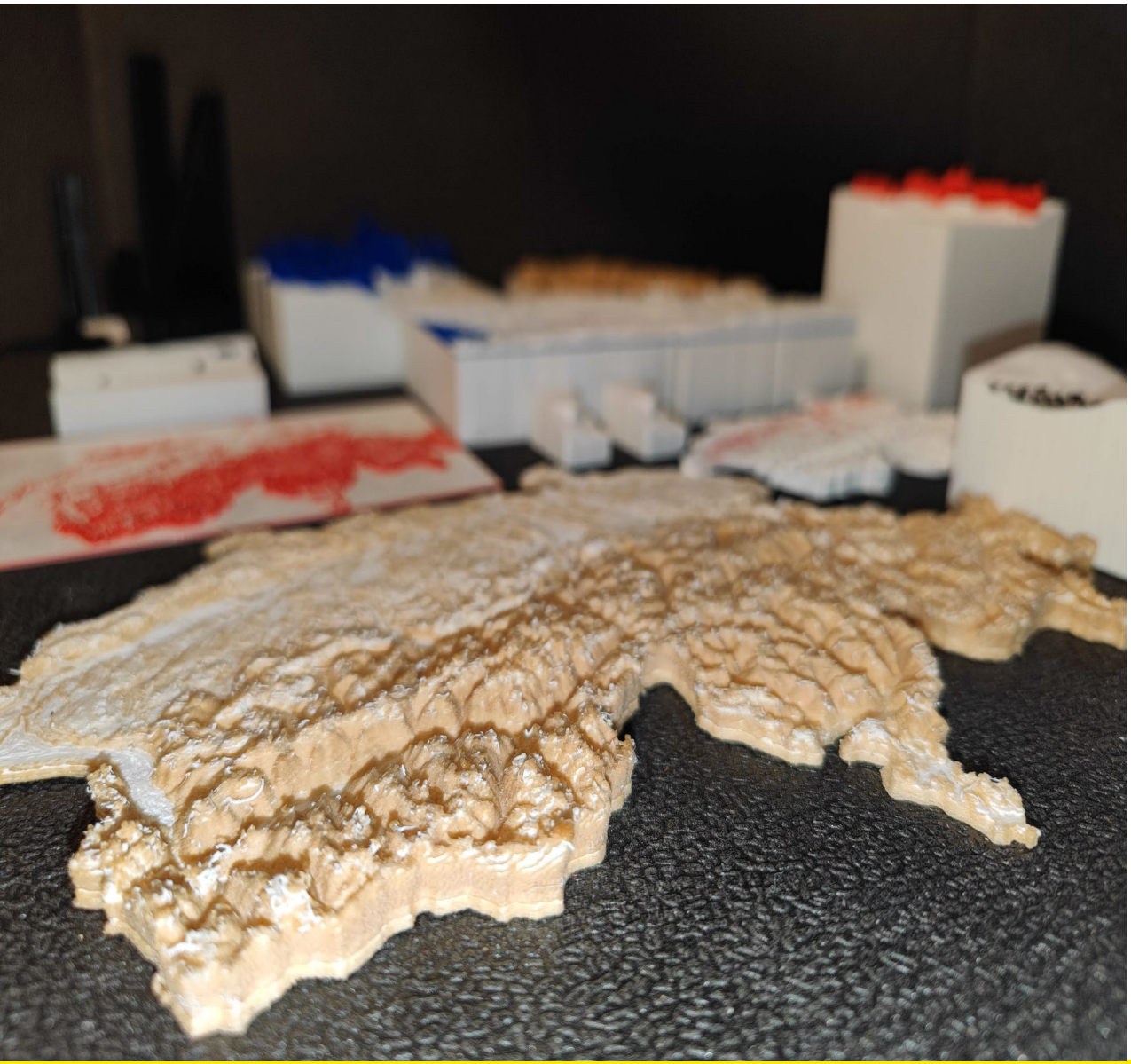
Python zur Geometrieverarbeitung
- Funktionsweise / Dateninteraktion / -manipulation wiegewohnt in Blender und Qgis
- «What you mean is what you get» anstatt «what you see is what you get»
- Für repetive Aufgaben sehr empfehlenswert, aber über Python Konsolen der Programme auch vieles machbar!

Erreichte Funktionalität mit dem Modul datato3D
- Zufridenstellend mit DHMs
- Einigermassen zufriedenstellend mit Stadtmodellen

Wahl **Geometrieverarbeitungs-Libraries**
- Open3D: sehr performant, lücken in Funktionalität
- Trimesh: weniger performant
- Shapely und Geopandas: zu triviale Nutzung für Einschätzung

Diese Weiterentwicklungen würden die **Nutzbarkeit deutlich erhöhen**:
- GUI
- Wasserdichte Lösung für die fehlenden Faces, auch mit non-rectangular-bbox
- Weitere Datenformate (SwissBuildings3D mit Fiona und GeoPandas einbringen)
- Weitere Linien, Punkte oder Polygone druckbar machen als zweites Modell in anderer Farbe
- Mosaikartige Unterteilung mit verdecktem Puzzle-verschluss
- Dächer und Bäume über Pinnsystem separat drucken und so von 2,5 D auf 3D
- Beschriftungsmöglichkeit

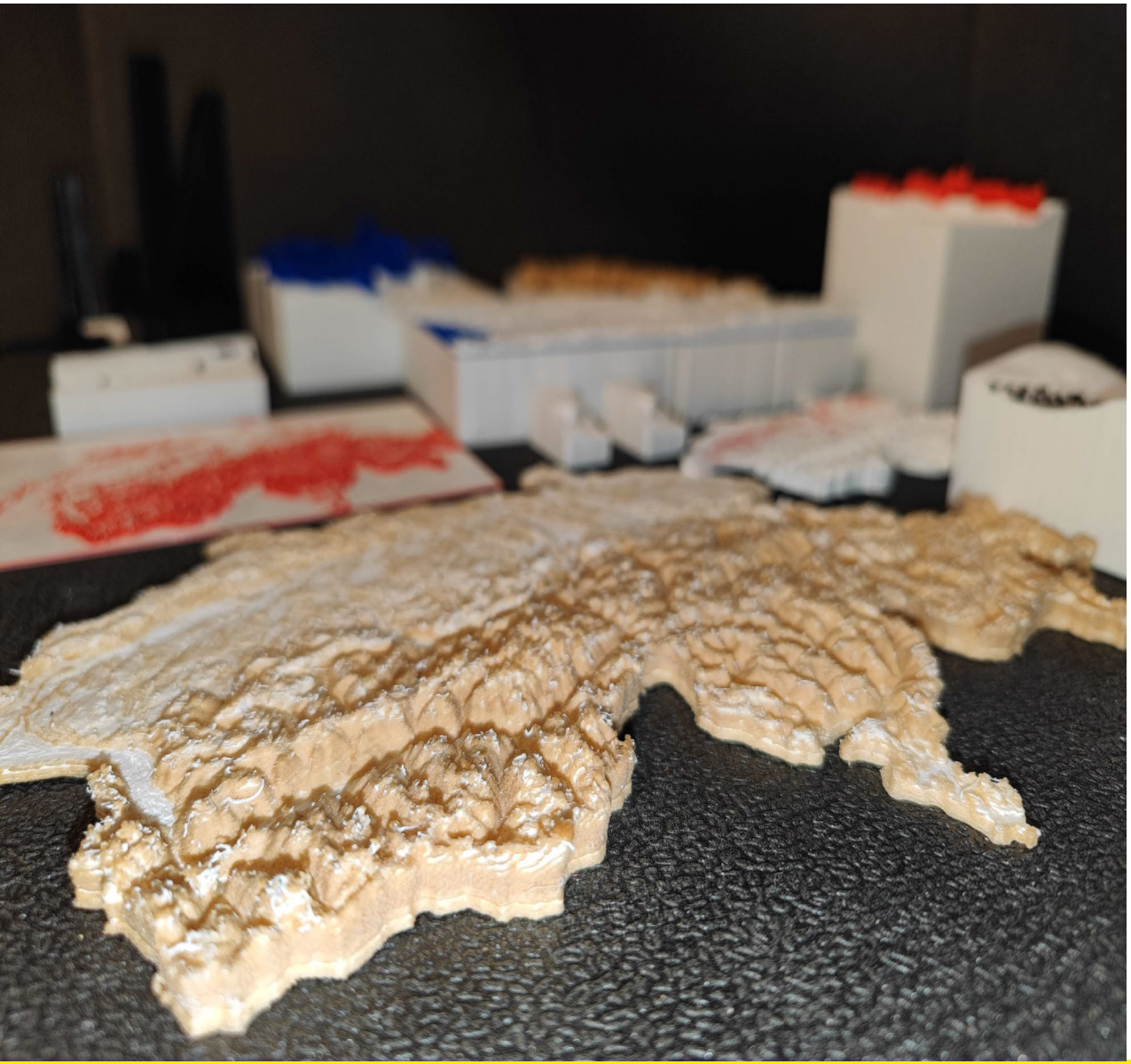
10. Juni 2024 Autor: Timon Kuhn, Betreuer: Prof. Martin Christen, Matus Gasparik, Experte: Dozent Robert Wüest © IGEO FHNW



Motivation

- Erst durch die **Visualisierung** werden Geodaten für den Menschen direkt nutz- / interpretierbar
- Vor jeder Nutzung / Visualisierung müssen **Raumbezogene Daten verarbeitet** werden
- Programmiersprachen wie **Python** bieten breite Paletten an Werkzeugkästen (Libraries genannt)
- Besonders im Fachbereich der **Architektur**, aber auch allgemein, trotzt die Beliebtheit **physicher Modelle** neueren virtuellen Möglichkeiten
- Der **3D-Druck** bietet eine moderne, automatisierte Alternative zur Erstellung solcher Modelle

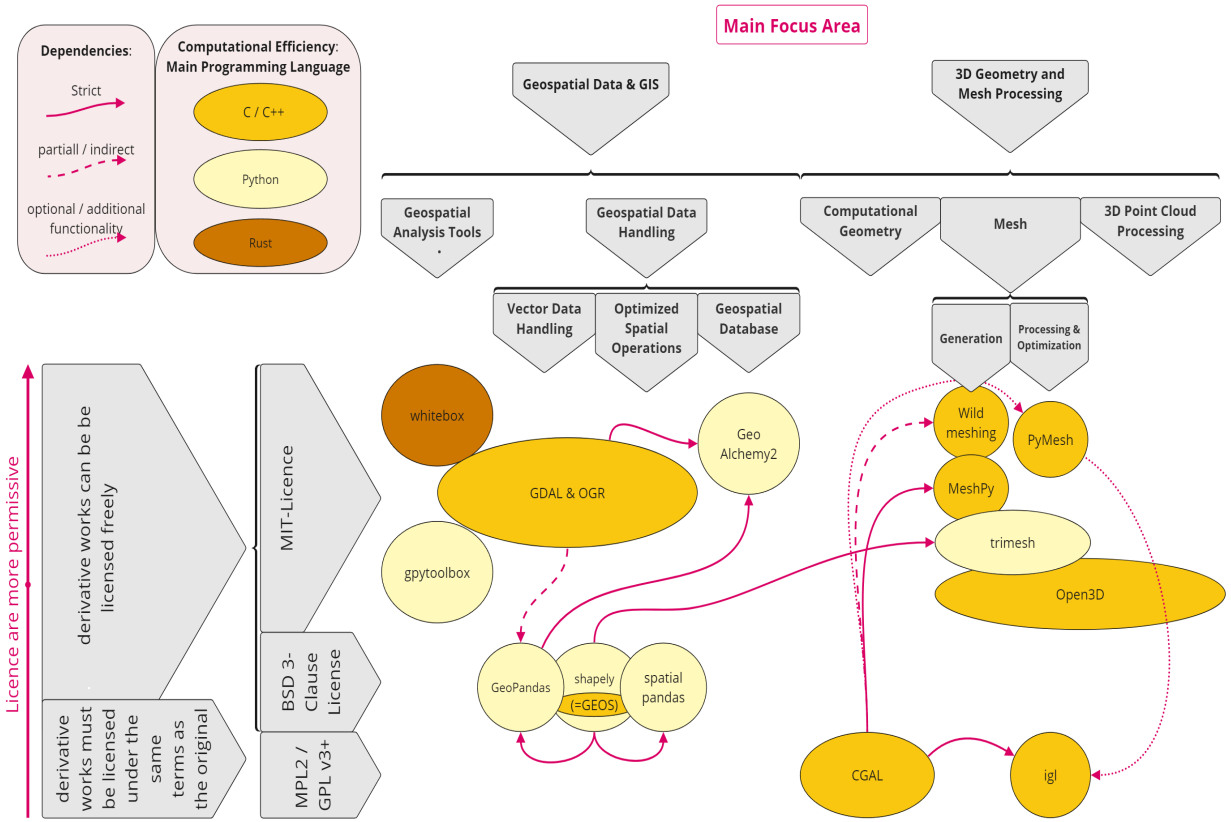
Abb. 1: Titelbild mit diversen 3D-gedrucktenProdukten dieser Bachelorarbeit



Fragestellung

- Was gibt es für Möglichkeiten, mit **Python Geometrien zu verarbeiten**?
- Wie kann man den **Workflow** (vom DHM und Stadtmodell zum STL) mit **Python umsetzen**?
- Wie kann man diese Drucke zur bestmöglichen visualisierung der Daten **optimieren**?

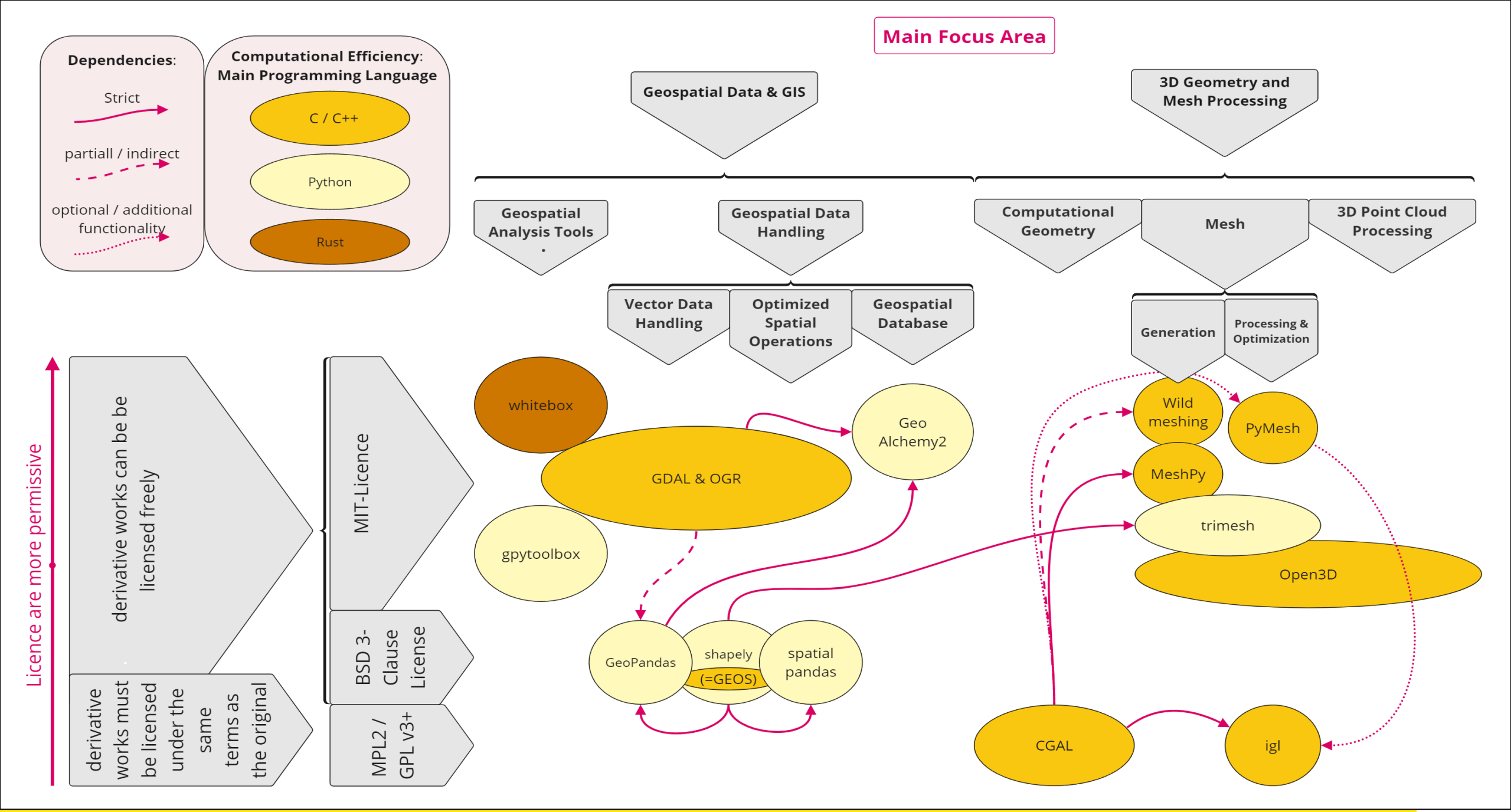
Zusammenstellung der Geometrie-verarbeitungs-Libraries in Python

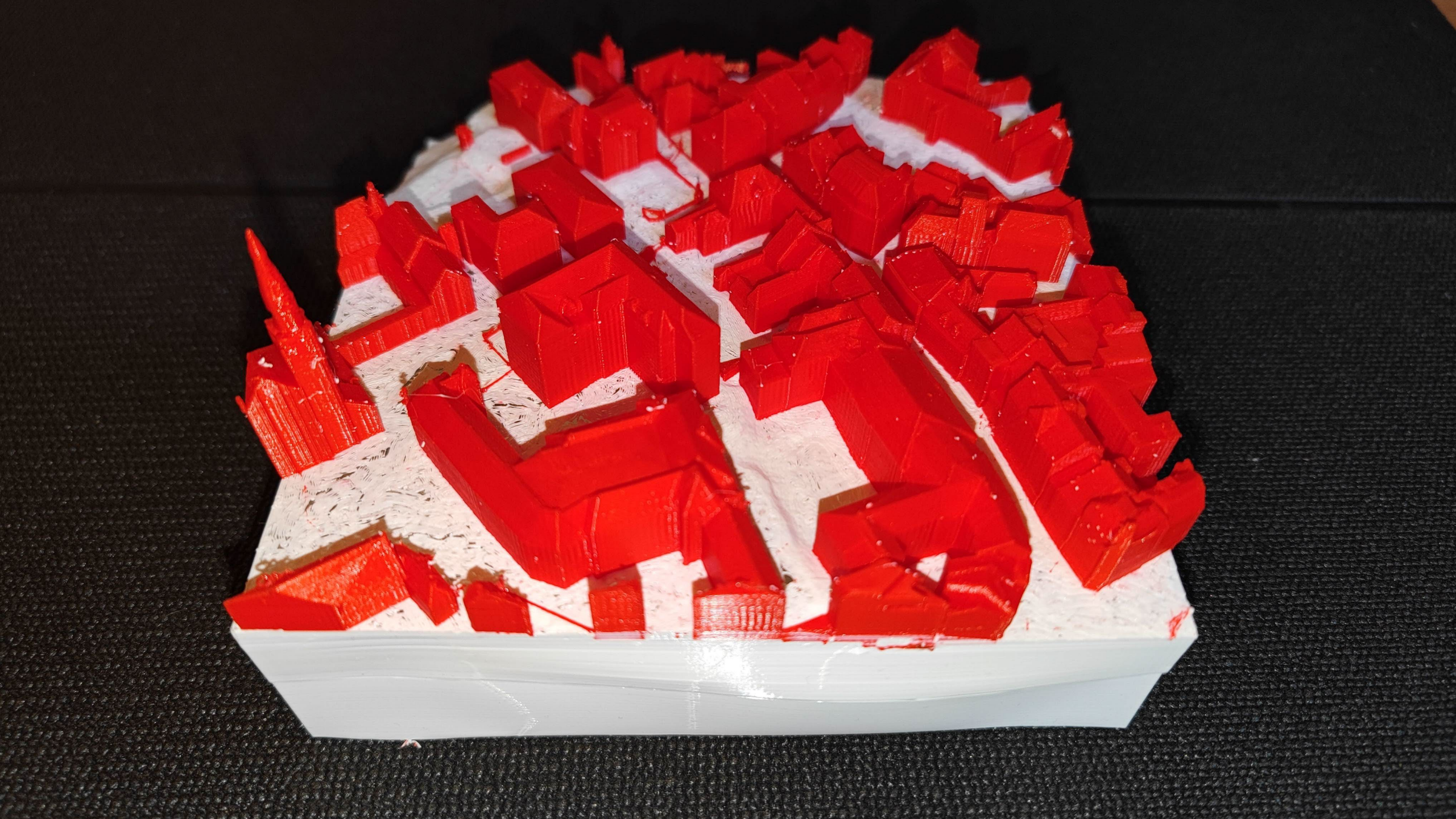


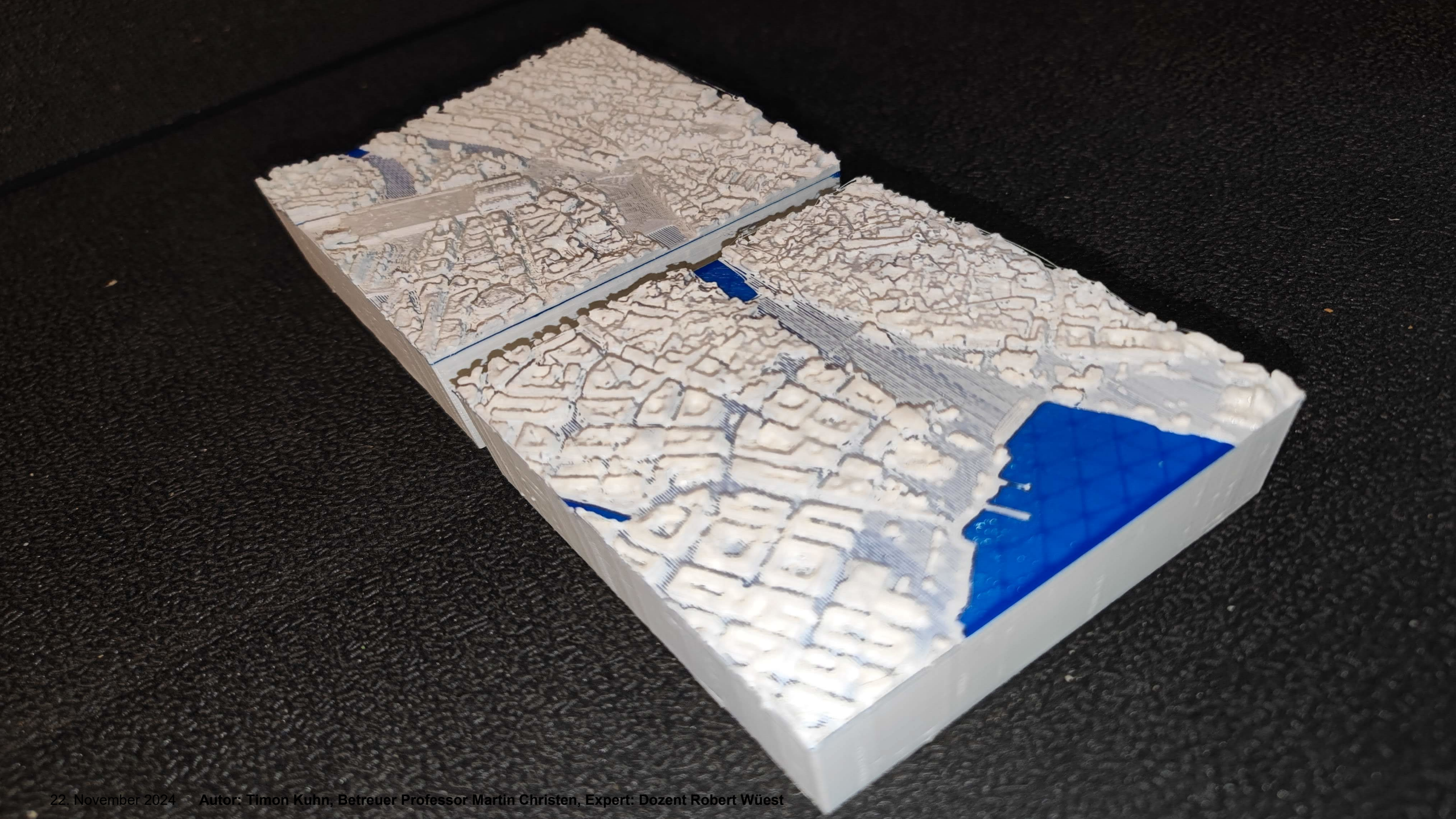
- Abgrenzung unter dem Begriff Geometrie**verarbeitung**, also der **Manipulation** der Geometrien selbst

- **Weglassen** aller Libraries deren Funktionalität beschränkt ist auf:
  - lesen und schreiben von raumbezogenen Daten
  - Visualisieren von Geometrien
  - Analyse und Informationsgewinnung
  - Koordinatentransformation

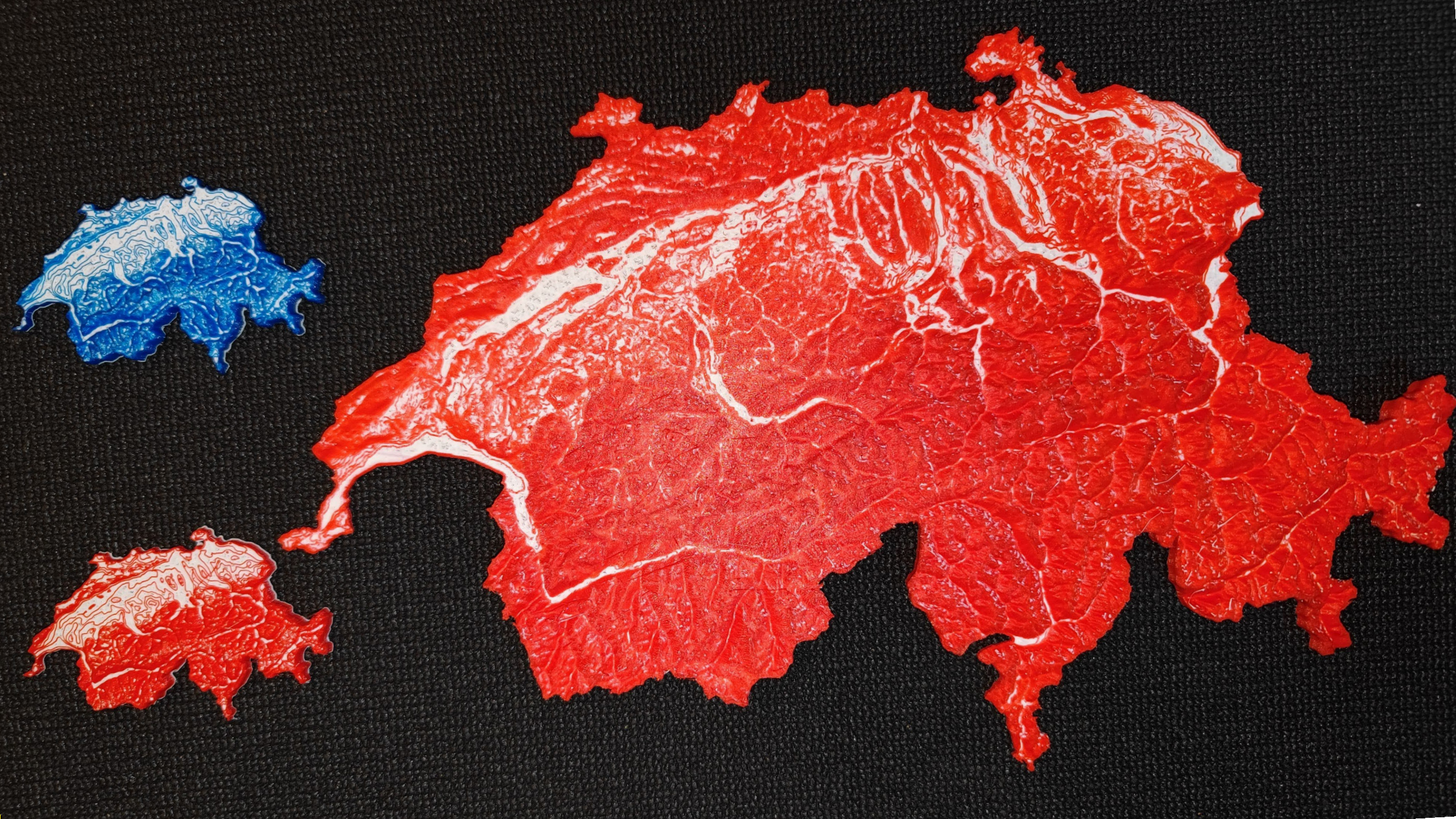
- Viele **Abhängigkeiten** untereinander sind gegeben

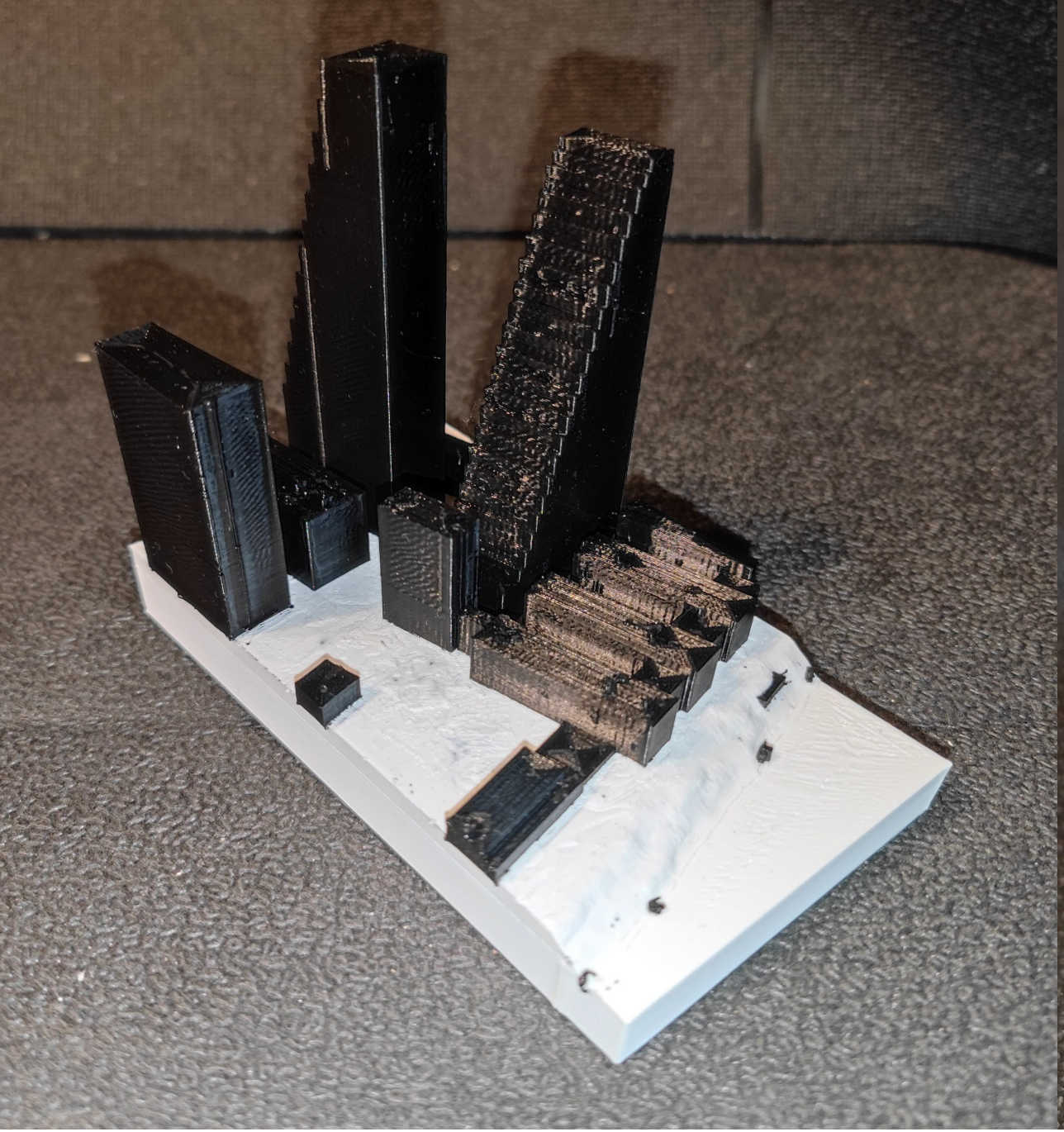

Main Focus Area
Dependencies:
Strict
partiall / indirect
optional / additional functionality
Computational Efficiency: Main Programming Language
C / C++
Python
Rust
Geospatial Data & GIS
3D Geometry and Mesh Processing
Geospatial Analysis Tools
Geospatial Data Handling
Computational Geometry
Mesh
3D Point Cloud Processing
Vector Data Handling
Optimized Spatial Operations
Geospatial Database
Generation
Processing & Optimization
whitebox
GDAL & OGR
gpytoolbox
Geo Alchemy2
GeoPandas
shapely
(=GEOS)
spatial pandas
Wild meshing
PyMesh
MeshPy
trimesh
Open3D
CGAL
igl
Licence are more permissive
derivative works can be be licensed freely
derivative works must be licensed under the same terms as the original
MIT-Licence
BSD 3-Clause License
MPL2 / GPL v3+

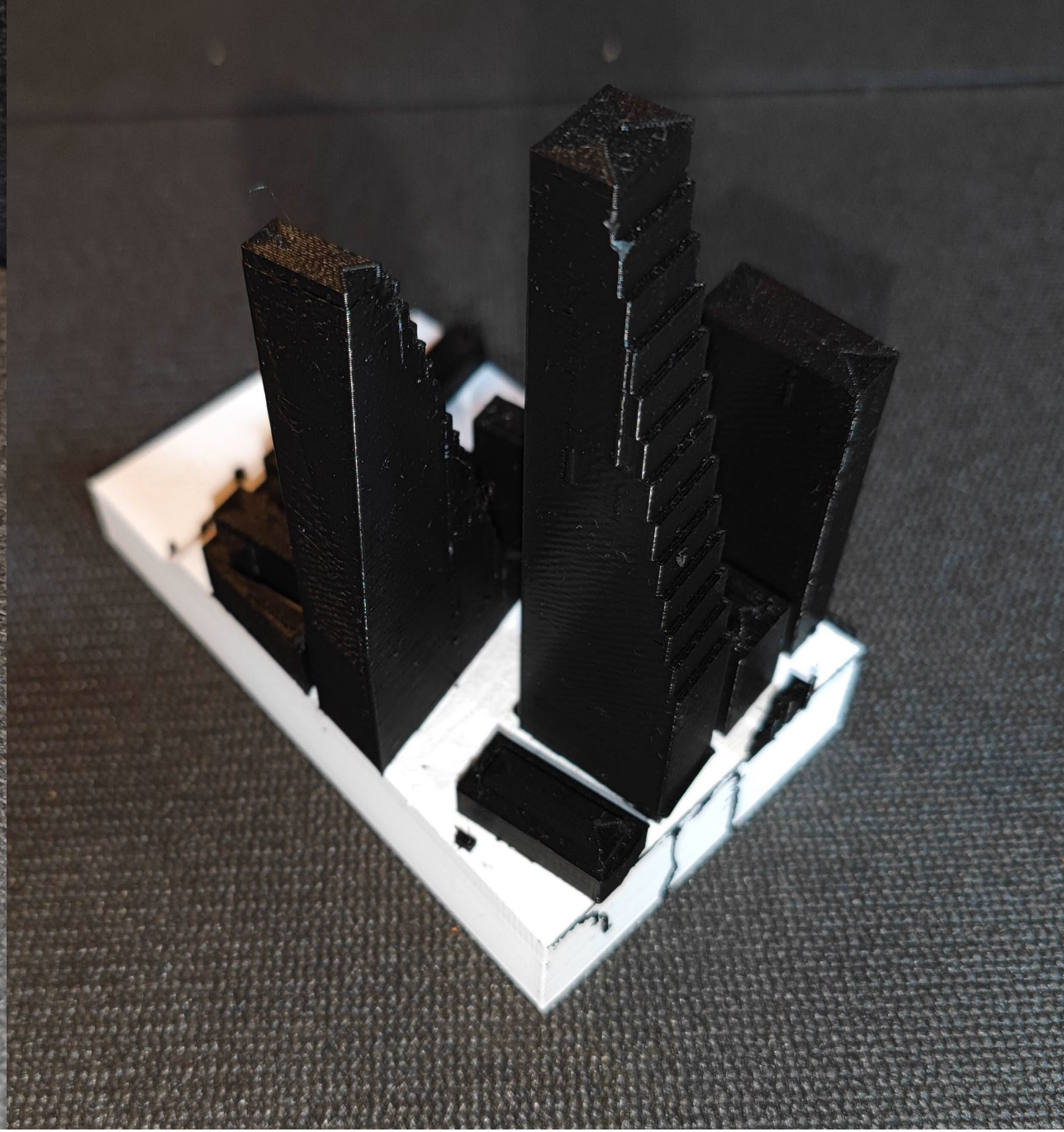


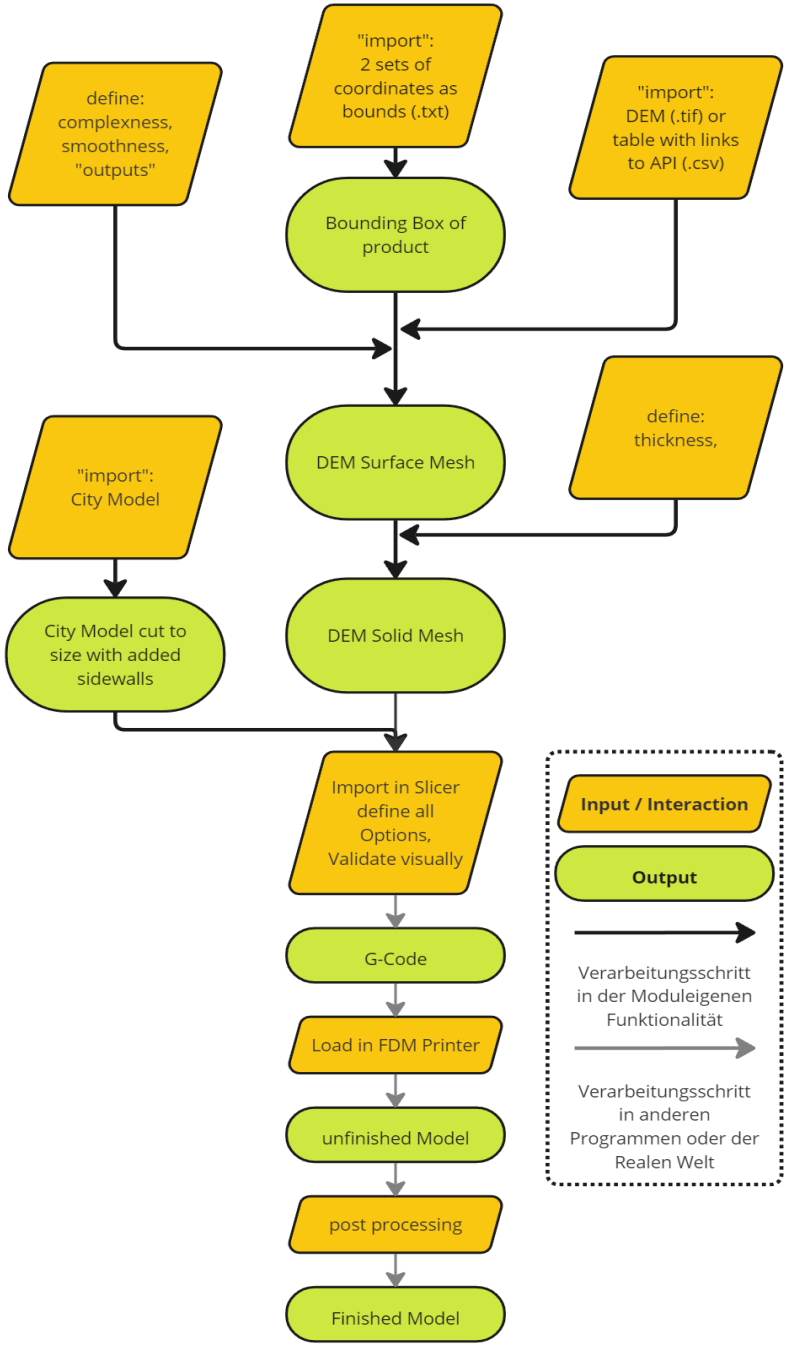


22. November 2024 **Autor: Timon Kuhn, Betreuer Professor Martin Christen, Expert: Dozent Robert Wüest**









Geometrieverarbeitung

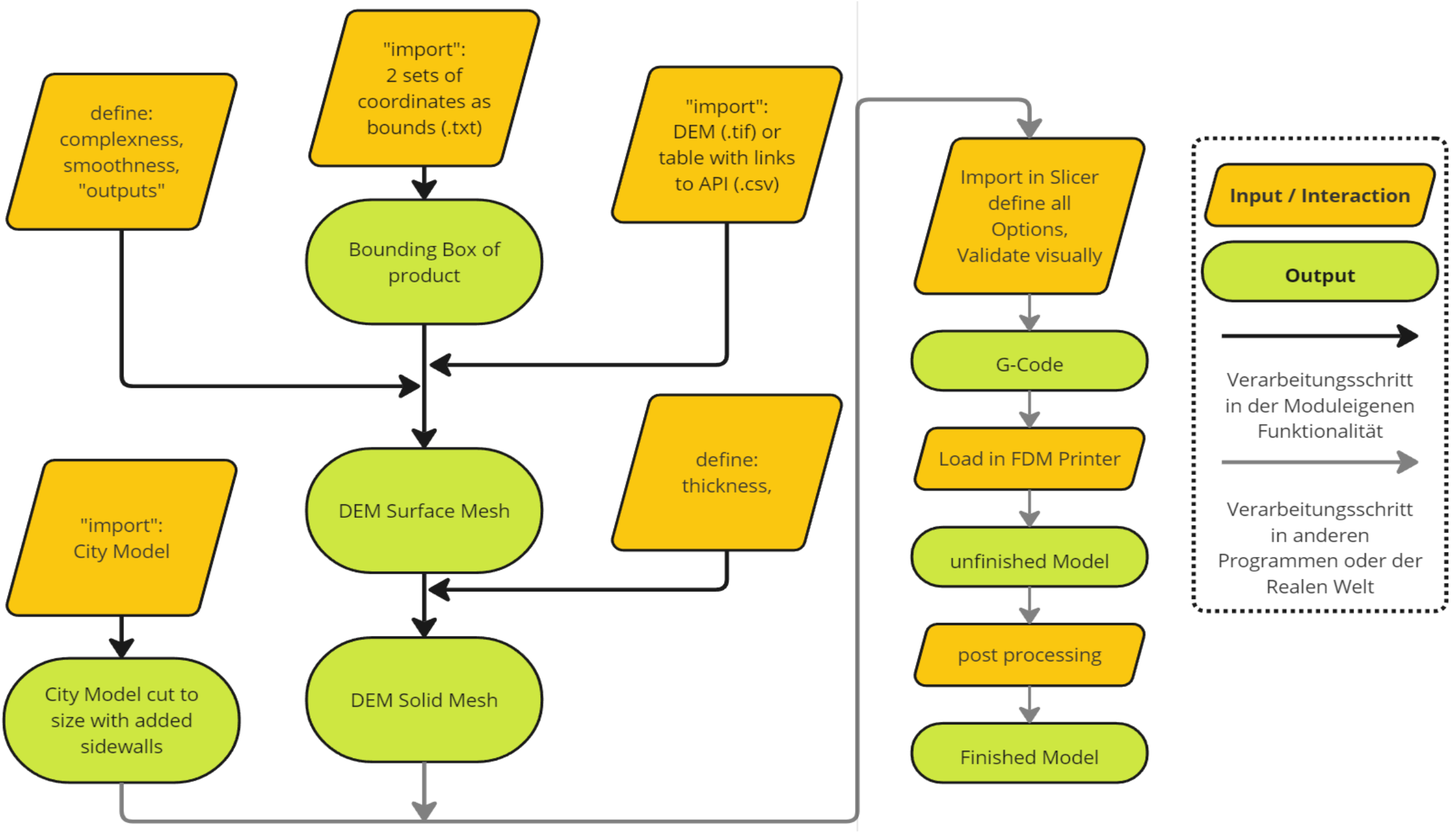
- **Standard-Workflow** auf der linken Seite beschreibt den Weg **vom DGM und Stadtmodell zum STL** (siehe Basel)

- Diese **Funktionalität** wurde in einem **Python-Modul implementiert**

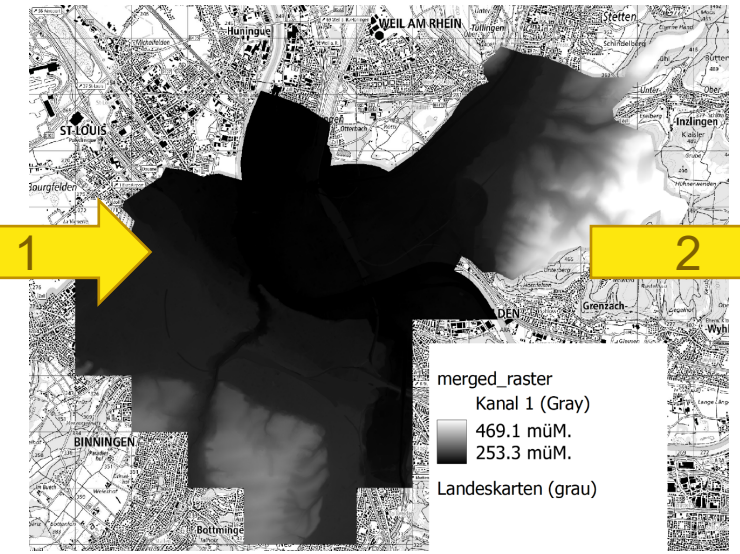
- Weitere Optionen durch weitere Funktionalitäten / andere Daten möglich:
  - DOM anstatt DGM + Gebäudemodell (siehe Zürich)
  - Andere Ausdehnung (siehe ganze Schweiz)

Grundlagedaten:
- .tiff: swissalti3D, swissSurface3D Raster, DHM25/200
- .obj: Stadtmodell Basel

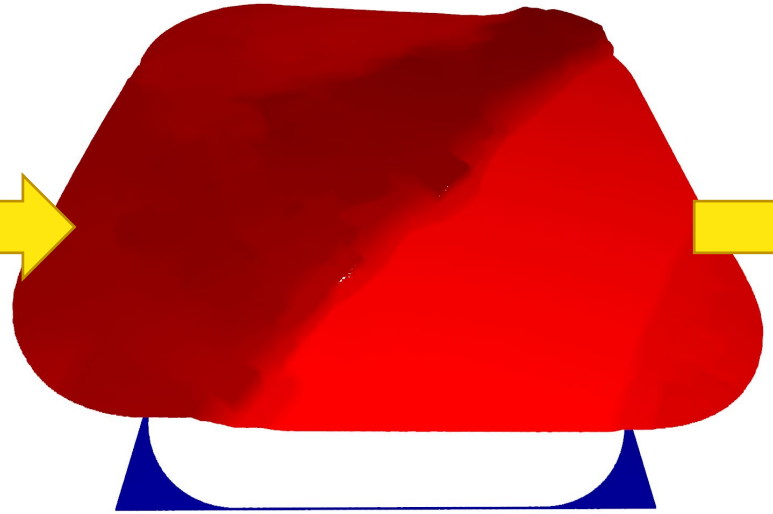
Genutzte Geometrieverarbeitung Libraries:
- Polygone: shapely, Geopandas
- Punktwolken & Mesh: open3D, Trimesh
- Raster *lesen:* rasterio, Numpy


define:
complexness,
smoothness,
"outputs"
"import":
2 sets of
coordinates as
bounds (.txt)
"import":
DEM (.tif) or
table with links
to API (.csv)
Bounding Box of
product
"import":
City Model
DEM Surface Mesh
define:
thickness,
City Model cut to
size with added
sidewalls
DEM Solid Mesh
Import in Slicer
define all
Options,
Validate visually
G-Code
Load in FDM Printer
unfinished Model
post processing
Finished Model
Input / Interaction
Output
Verarbeitungsschritt
in der Moduleigenen
Funktionalität
Verarbeitungsschritt
in anderen
Programmen oder der
Realen Welt

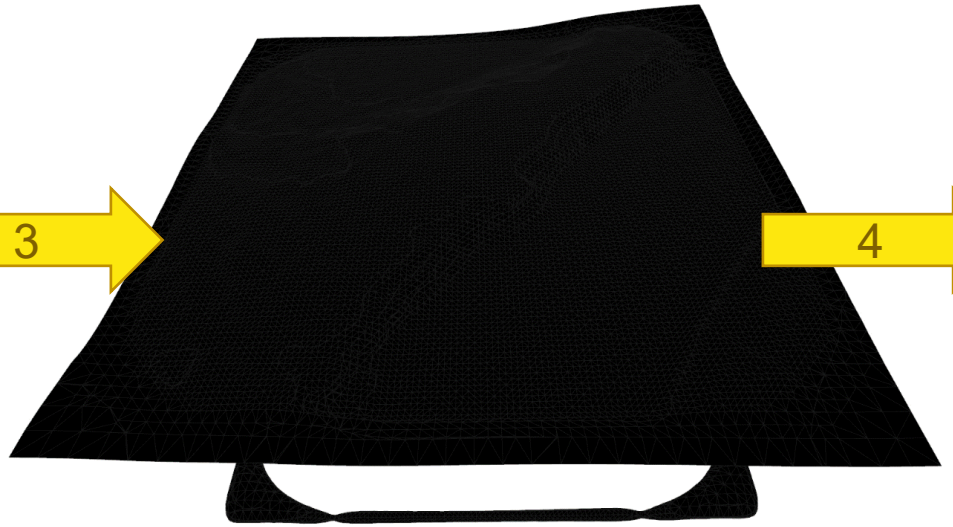
Verarbeitung der Höhenmodelle



1

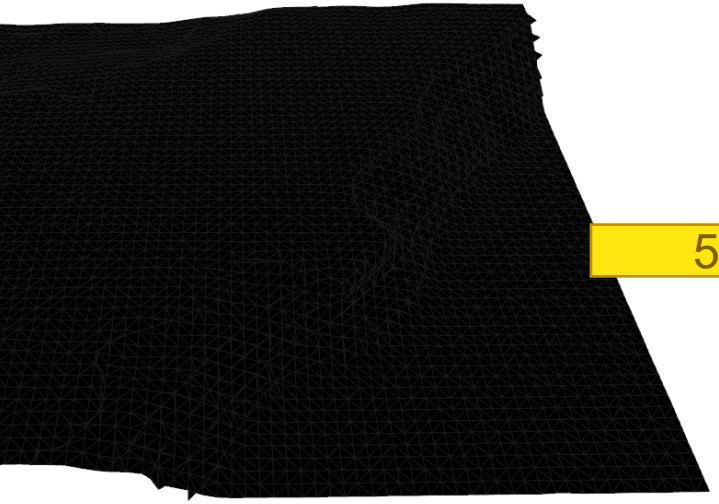


2

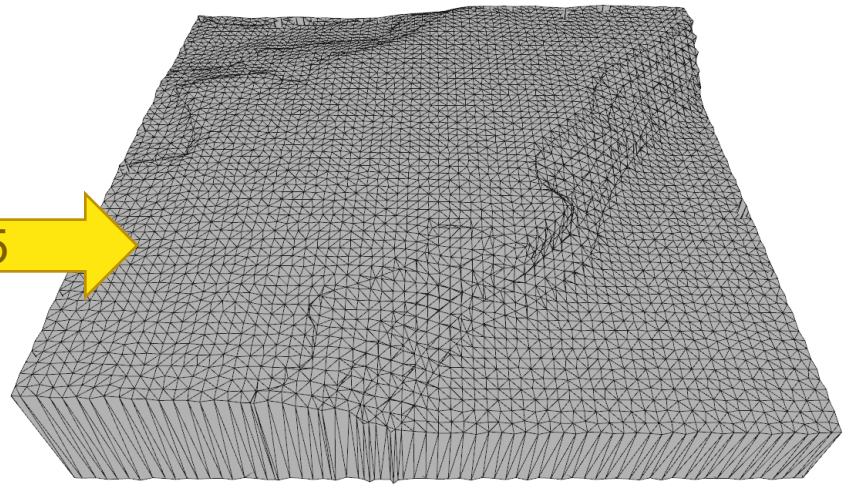


3

4



5



1.) ein .tiff Rasterfile wird gelesen (Optional: es werden angegebene Kacheln heruntergeladen & gestiched

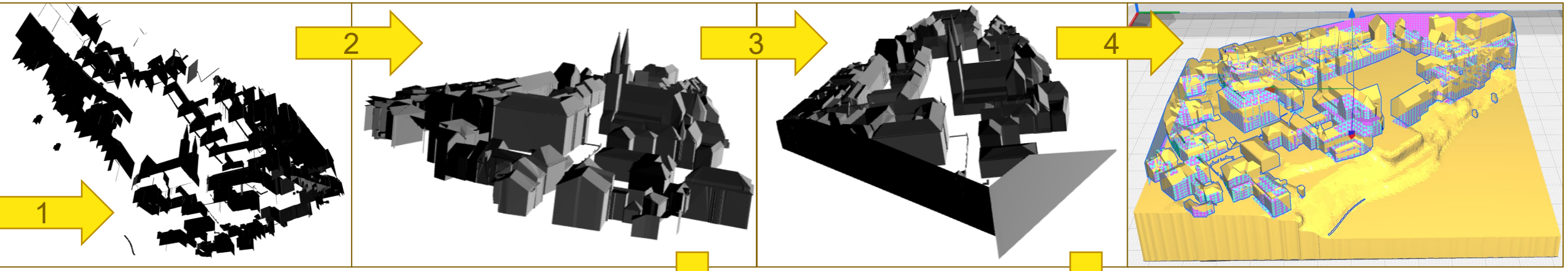
2.) Jeder Rasterwerte wird ein Punkt in einer Punktwolke

3.) Dreiecksvermaschung

4.) Zuschneiden auf Bbox (Optional: non-rectangular)

5.) «thickness» Extrudieren

Verarbeitung der Stadtmodelle



1.) Stadtmodell als .obj laden

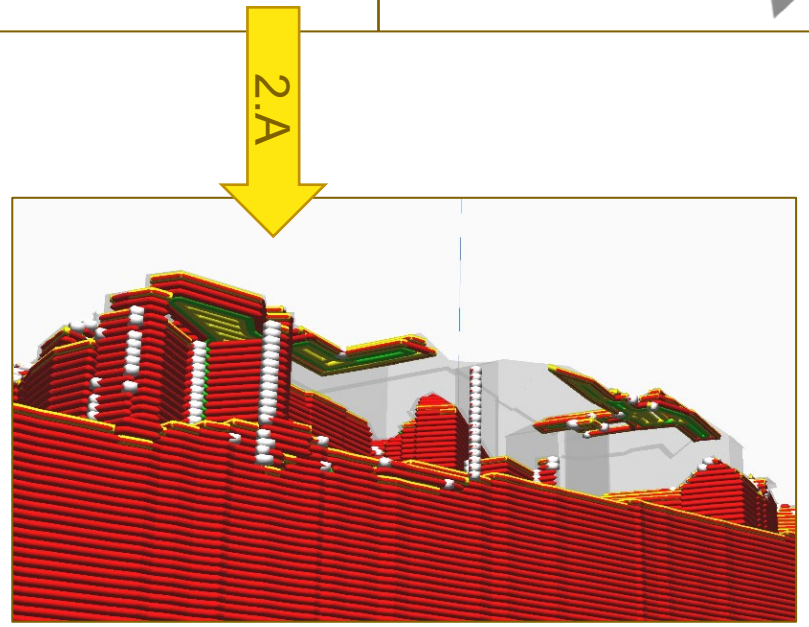
2.) auf Bbox zuschneiden über Ebenen

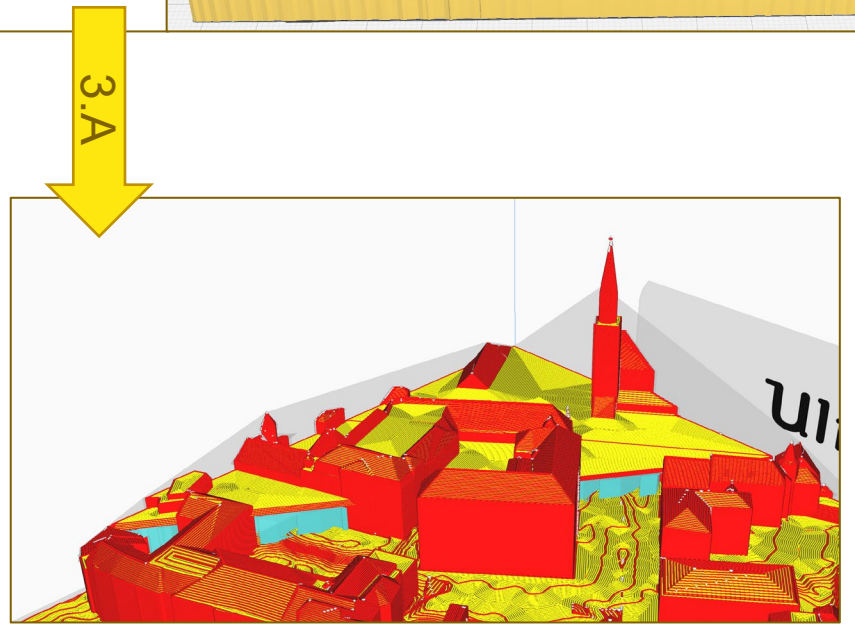
2.A) Ein solcher Slice misslingt häufig, da die Volumen nicht geschlossen sind

3.) Hinzufügen von Faces an den Seiten

3.A) ein solcher Slice kann ebenfalls misslingen, da zu viele Wände definiert sind

4.) Beste Ergebnisse werden erzielt, wenn nur ein Teil der Seitenwände rekonstruiert werden





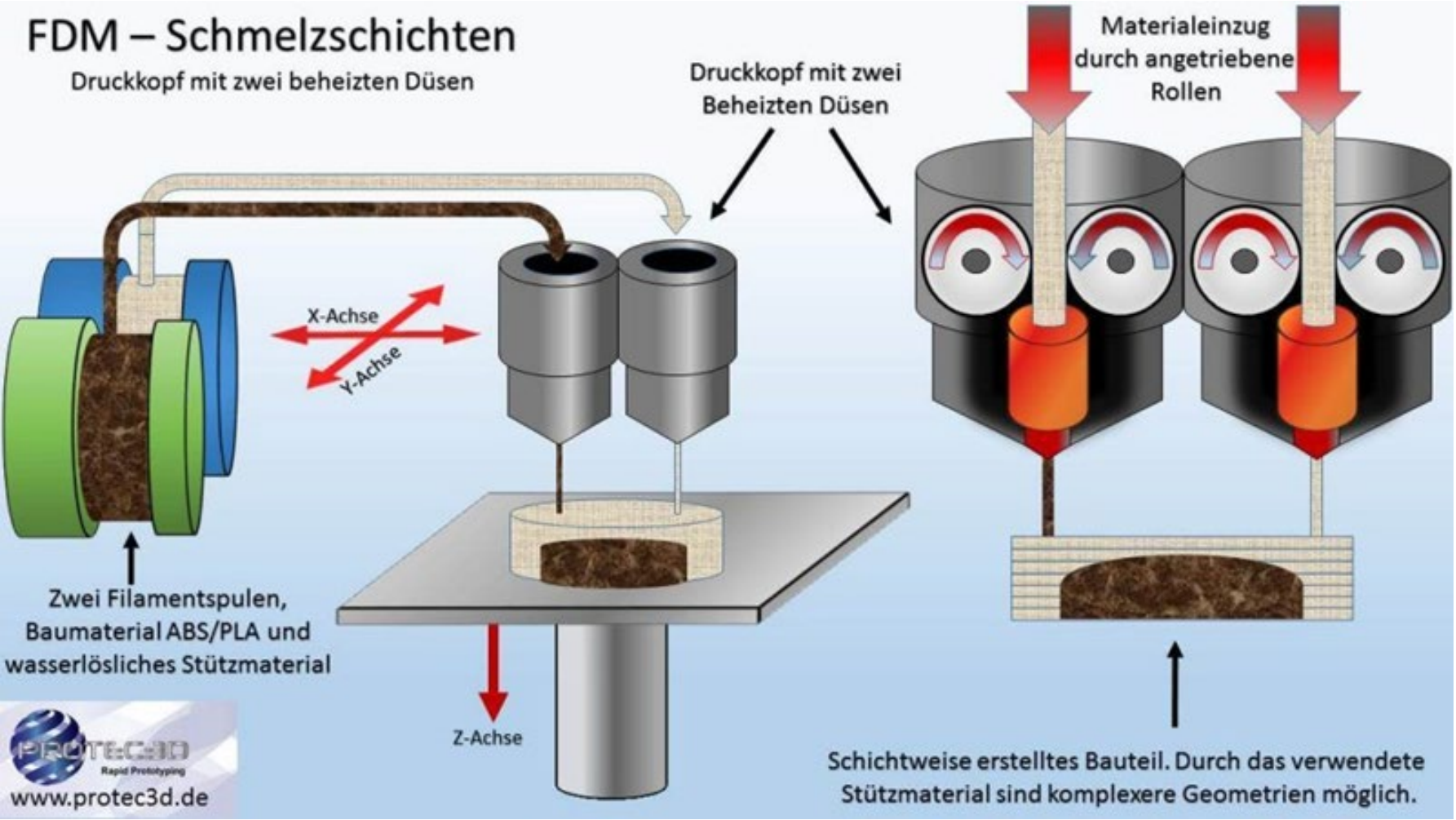
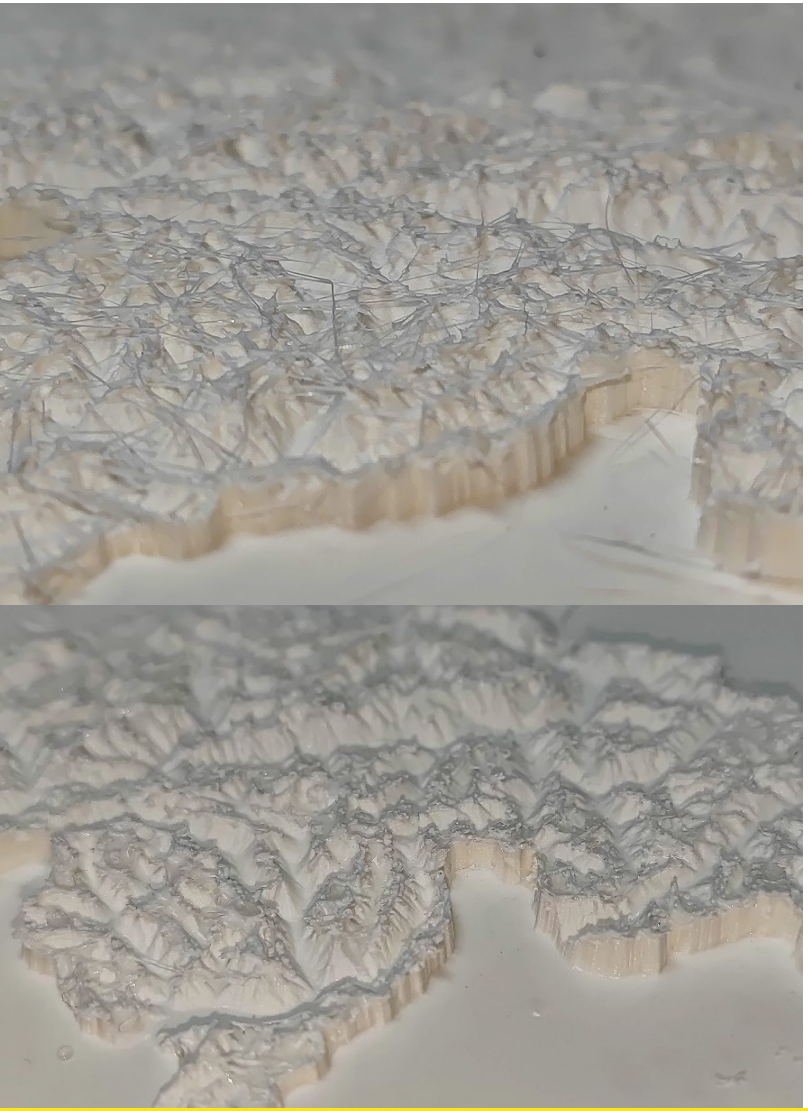


Abb. 2: schematisce Darstellung des FDM-Verfahrens aus (Reizner 2014, www.protec3d.de)

3D-Druck

- Ein **Slicer** (Cura) wandelt ein **STL-Modell** in **G-Code** (Maschienensteuerung)
- Der G-Code beschreibt Bewegungen (XY und Z), Temperaturen, usw.
- Gedruckt wird mit PLA und ähnlichen Materialien
- **Slicing** minuten bis Stunden
- **Drucken** Stunden bis Tage
- 2 – 200 Gramm, je nach Modellgrösse und -stärke



Optimierungen

- **Trial & Error** als Grundprinzip, denn:
  - 450 Einstellungen
  - Dutzende Materialien
  - Verschiedene Düsengrössen
  - ➢ Unberechenbare Umweltfaktoren, reproduzierbare Modellfehler und Fehleinstellungen führen gleichermassen zu **fehlschlagenden Drucken**

- **Mehrfarbigkeit** über drei Möglichkeiten:
  - Mehrere separate Modelle laden
  - Support Blocker
  - «Top Surface» oder «Outer Wall» Extruder überschreiben

- **Materialreduzierung**
  - Boden aus 1-2 Layer, Decke aus 3-5 Layer
  - Infill als Lightning, Wandstäke auf 1 reduzieren
  - ➢ Im Zweifelsfall **besser dicker, als zweimal drucken**

- Nachbearbeitung, «Adaptive Layers», «Ironing» & «Coasting» beachten.

Abb. 3 udn 4: vergleich zwischen starkem Stringing und mit Heissluftföhn nachbearbeitetem Modell

Einschätzungen & Ausblick

Python zur Geometrieverarbeitung
- Funktionsweise / Dateninteraktion / -manipulation wiegewohnt in Blender und Qgis
- «What you mean is what you get» anstatt «what you see is what you get»
- Für repetive Aufgaben sehr empfehlenswert, aber über Python Konsolen der Programme auch vieles machbar!

Erreichte Funktionalität mit dem Modul datato3D
- Zufridenstellend mit DHMs
- Einigermassen zufriedenstellend mit Stadtmodellen

Wahl **Geometrieverarbeitungs-Libraries**
- Open3D: sehr performant, lücken in Funktionalität
- Trimesh: weniger performant
- Shapely und Geopandas: zu triviale Nutzung für Einschätzung

Diese Weiterentwicklungen würden die **Nutzbarkeit deutlich erhöhen**:
- GUI
- Wasserdichte Lösung für die fehlenden Faces, auch mit non-rectangular-bbox
- Weitere Datenformate (SwissBuildings3D mit Fiona und GeoPandas einbringen)
- Weitere Linien, Punkte oder Polygone druckbar machen als zweites Modell in anderer Farbe
- Mosaikartige Unterteilung mit verdecktem Puzzle-verschluss
- Dächer und Bäume über Pinnsystem separat drucken und so von 2,5 D auf 3D
- Beschriftungsmöglichkeit