From Data to Digital Twin for Smart Mobility

Course for the PhD program Information Engineering
2023-2024

Lesson #4

Smart City Digital Twin

3rd July 2024

Marco Fanfani – Stefano Bilotta

Introduction

- A **Digital Twin** is a virtual replica of a real entity
- Originally named *Mirrored Space Model* by M. Grieves in 2002
- Renamed digital twin by J. Vickers of NASA

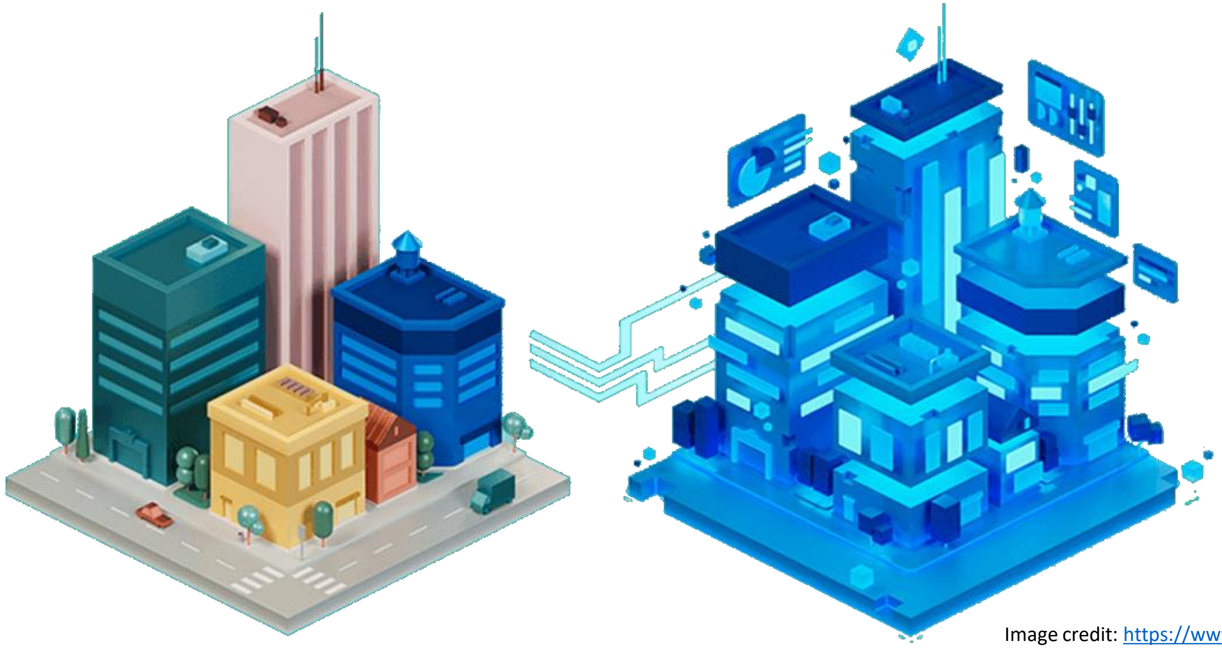
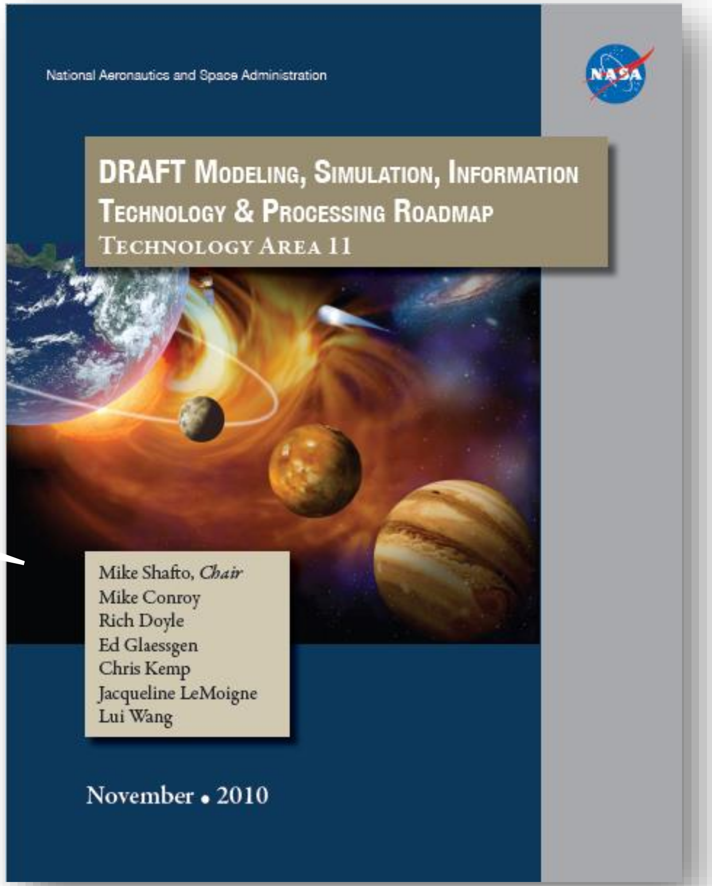


Image credit: https://www.12dsynergy.com/innovation-showcase/digital-twins-explained/

Introduction

- First applications of digital twins emerged in the **aerospace** field

*A digital twin is an **integrated** multi-physics, multi-scale, probabilistic simulation of a vehicle or system that uses the best available physical models, sensor updates, fleet history, etc., to **mirror the life of its flying twin**.*



Introduction

- First applications of digital twins emerged in the **aerospace** field
- Then, digital twins appeared in the **manufacturing** and **construction** industries

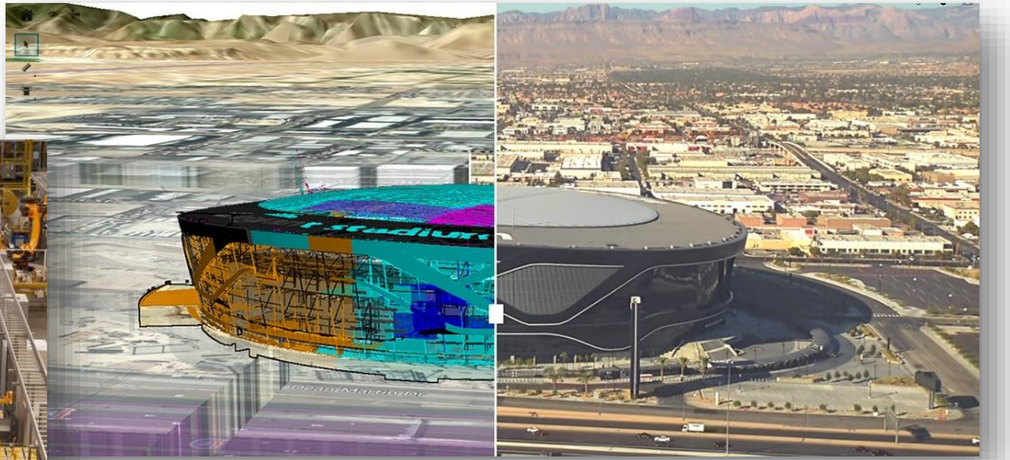


Image credits: https://aec-business.com/earthcam-4d-and-bentley-bring-construction-digital-twins-to-life/

Image credits: Siemens Digital Industries

Introduction

- Digital twins are powered by
  - Internet-of-Things/Web-of-Things (IoT/WoT)
  - Big Data
  - Industry 4.0

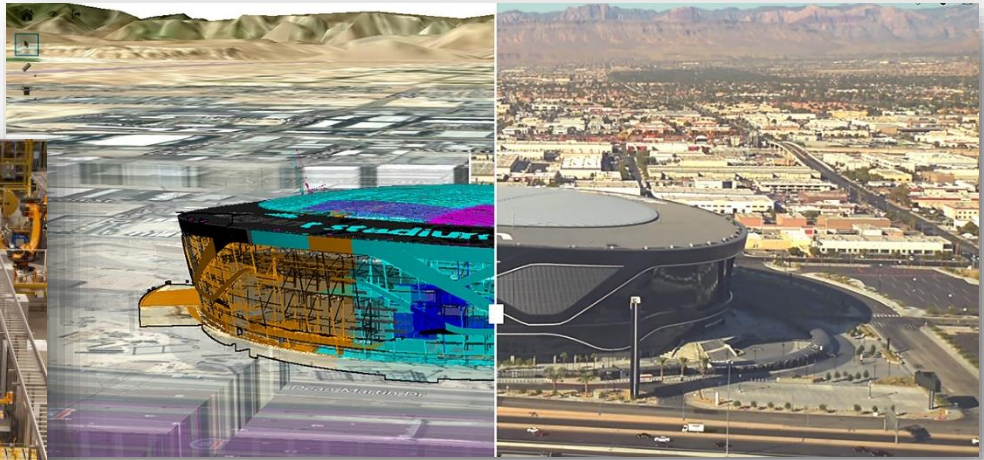


Image credits: https://aec-business.com/earthcam-4d-and-bentley-bring-construction-digital-twins-to-life/

Image credits: Siemens Digital Industries

Introduction

- More recently the concept of digital twins has been adopted in the context of **Smart Cities**
- Digital twin technology can undoubtfully help to face future urban developments in several domains
  - Mobility
  - Environment
  - Energy
  - Urban planning
  - …

Introduction

- According to *VirtualCity* project a smart city digital twin should have six characteristics
  - **Realistic**: The digital twin is a realistic 4-dimensional (in both space and time) visual and acoustic virtual experience of the physical counterpart.
  - **Interactive**: The digital twin is intuitive, accessible, and supports multi-user interaction.
  - **Simulated**: The digital twin is a simulation of the physical twin.
  - **Integrated**: The digital twin is continuously synchronized with the physical twin.
  - **Scalable**: The digital twin is open-ended, scalable from the building to the district to the city level.
  - **Open**: The digital twin is driven by open data and models.

Chalmers University of Technology (2018) Virtual City@Chalmers. https://virtualcity.chalmers.se/about/

Introduction

- According to *VirtualCity* project a smart city digital twin should have si

A smart city digital twin, even if based on a **3D digital replica** of the urban environment, must go beyond, including **dynamic real-time information**, aggregated in an **accessible interactive interface** where the users can perform **analysis** and **simulations** to study possible evolution of the cities, and from which (with some limits) should be possible to **apply changes** in the real counterpart

Chalmers University of Technology (2018) Virtual City@Chalmers. https://virtualcity.chalmers.se/about/

Introduction

- In (Ketzler et at. 2020) a digital twin is said to be composed by three main layers

Ketzler, B., Naserentin, V., Latino, F., Zangelidis, C., Thuvander, L., & Logg, A. (2020). Digital twins for cities: A state of the art review. Built Environment, 46(4), 547-573.

Introduction

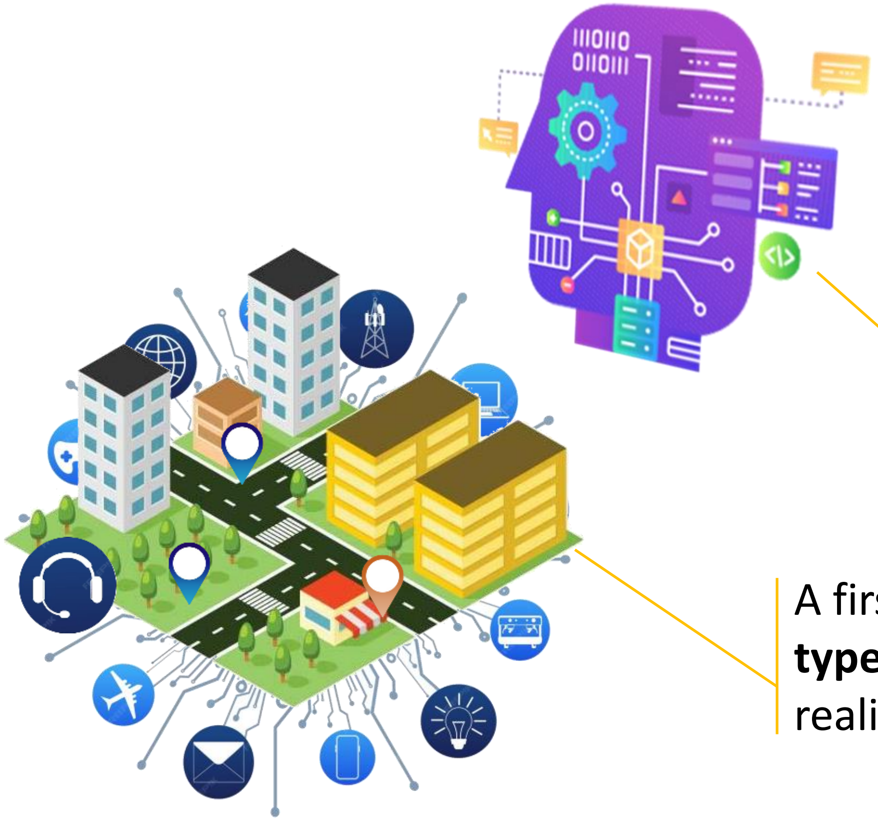
- In (Ketzler et at. 2020) a digital twin is said to be composed by three main layers

A first layer a first layer included the **heterogeneous data types** such as 3D buildings, maps, data from sensors, etc., realizing the so-called **City Information Model (CIM)**.

Ketzler, B., Naserentin, V., Latino, F., Zangelidis, C., Thuvander, L., & Logg, A. (2020). Digital twins for cities: A state of the art review. Built Environment, 46(4), 547-573.

Introduction

- In (Ketzler et at. 2020) a digital twin is said to be composed by three main layers



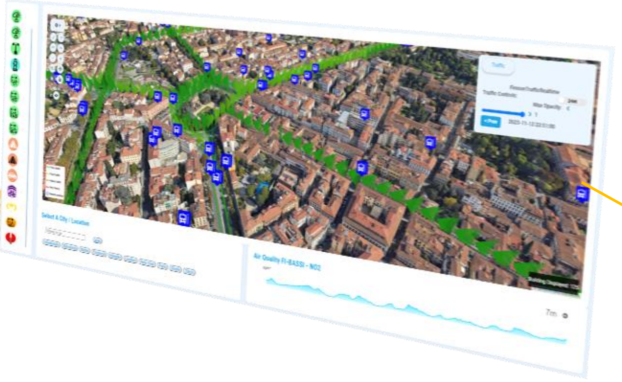
A second layer aimed to cover basic functionalities, communications and can implement **analytics** and **software**. This can be achieved with an adequate IoT/IoE platform able to manage and manipulate huge amount of data.

A first layer a first layer included the **heterogeneous data types** such as 3D buildings, maps, data from sensors, etc., realizing the so-called **City Information Model (CIM)**.

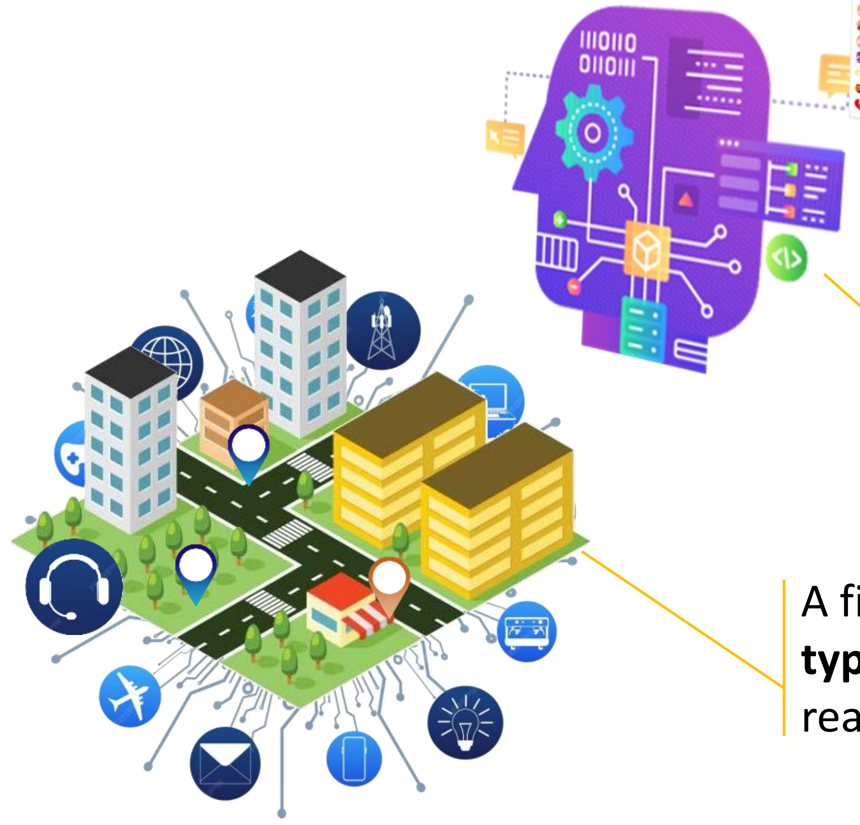
Ketzler, B., Naserentin, V., Latino, F., Zangelidis, C., Thuvander, L., & Logg, A. (2020). Digital twins for cities: A state of the art review. Built Environment, 46(4), 547-573.

Introduction

- In (Ketzler et at. 2020) a digital twin is said to be composed by three main layers



A final layer devoted **to visualization and distribution** thought 3D engines and web applications



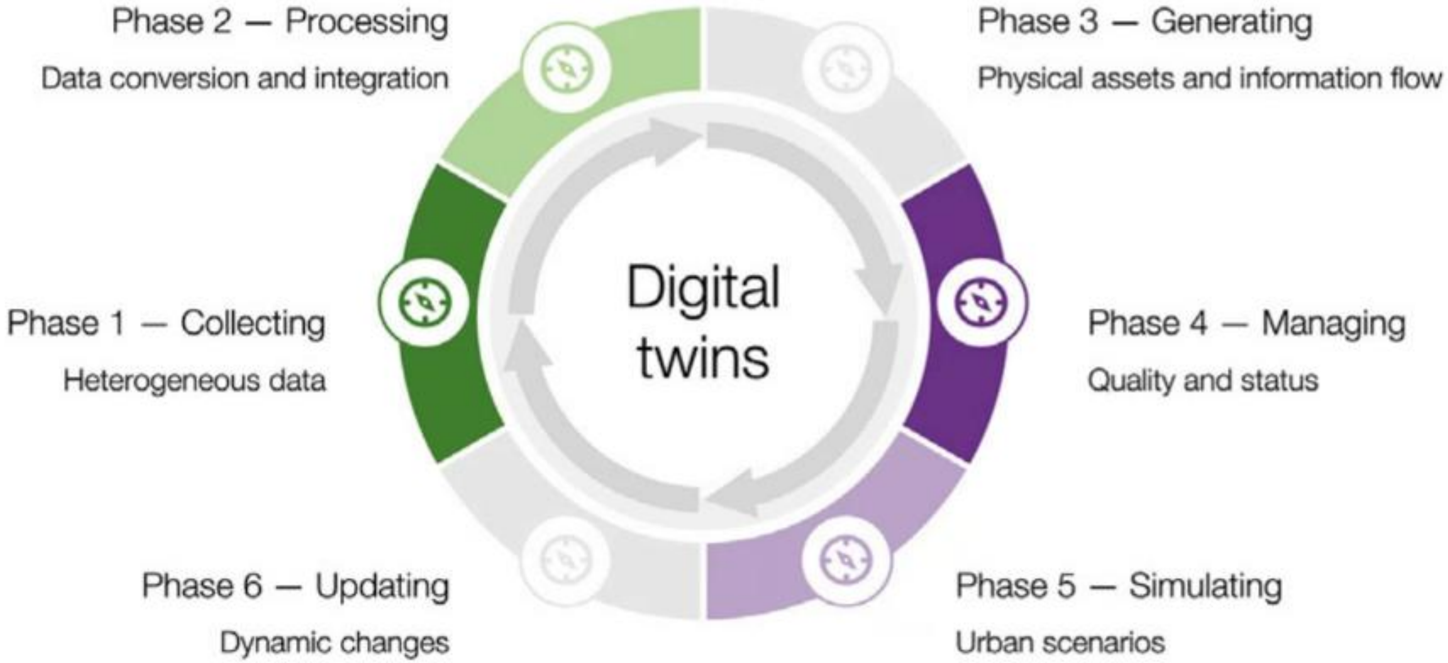
A second layer aimed to cover basic functionalities, communications and can implement **analytics** and **software**. This can be achieved with an adequate IoT/IoE platform able to manage and manipulate huge amount of data.

A first layer a first layer included the **heterogeneous data types** such as 3D buildings, maps, data from sensors, etc., realizing the so-called **City Information Model (CIM)**.

Ketzler, B., Naserentin, V., Latino, F., Zangelidis, C., Thuvander, L., & Logg, A. (2020). Digital twins for cities: A state of the art review. Built Environment, 46(4), 547-573.

Introduction

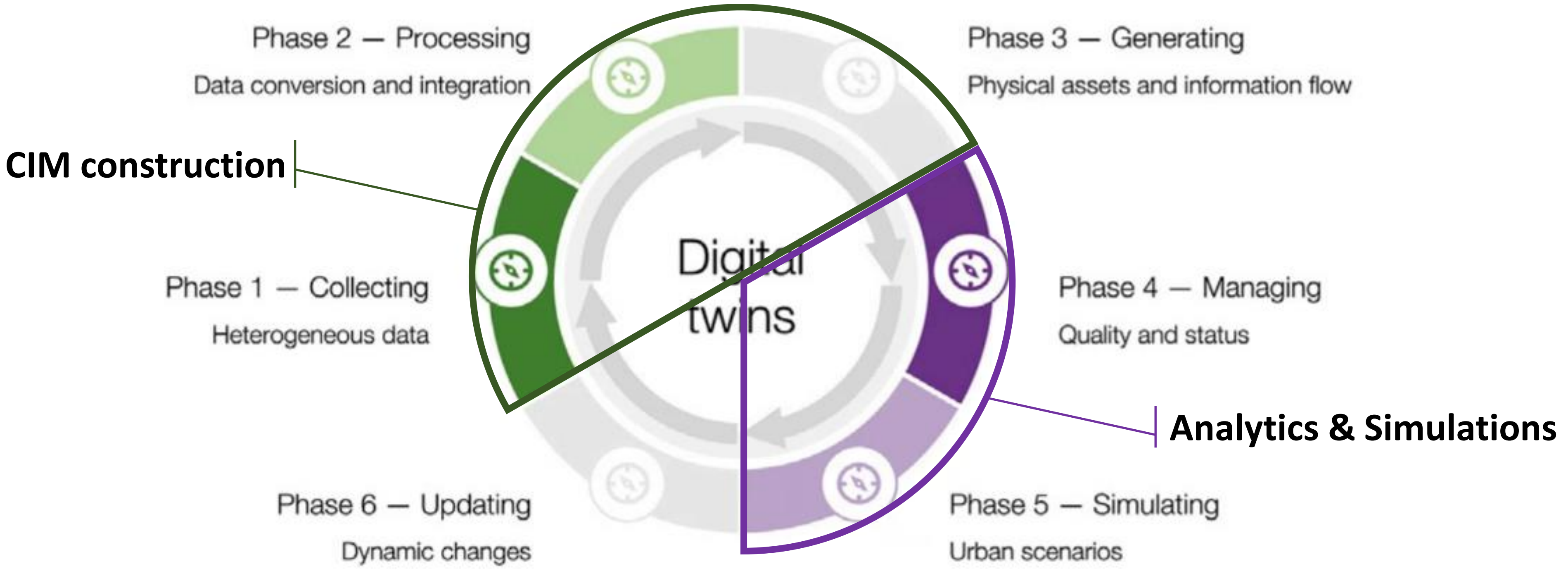
- The three-layer definition is in line with the smart city digital twin life-cycle proposed in (Lei et al. 2023)



Lei, B., Janssen, P., Stoter, J., & Biljecki, F. (2023). Challenges of urban digital twins: A systematic review and a Delphi expert survey. Automation in Construction, 147, 104716.

Introduction

- The three-layer definition is in line with the smart city digital twin life-cycle proposed in (Lei et al. 2023)



Lei, B., Janssen, P., Stoter, J., & Biljecki, F. (2023). Challenges of urban digital twins: A systematic review and a Delphi expert survey. Automation in Construction, 147, 104716.

Requirements

- Based on the literature, we defined a series of requirements to guide the development of a SCDT framework

Field Interoperability

Which function should be included to guarantee an accessible, integrated, and affordable SCDT solution?

Data and Computing for Representation

Which data and software should be included into a SCDT?

Distribution and Interaction

Which kind of user interaction and animation should be offered by the SCDT interface?

Field Interoperability

ID	Name	Description	
R1	Network Modeling and Management	The modeling of data networks including gateways, brokers, devices, services and API, external services as web pages, and protocols.	Field Interoperability
R2	Hierarchical modeling of entities	To model data for terrain, city building and shapes (gardens, roads), services, heatmaps, traffic flows, services, IoT devices, public transportation, etc. To be retrievable with relational, geographical, and temporal queries.	
R3	Logics for data transformation	To transform data collected, for example from IoT sensors, and other sources and transform them into different data models and formats. For example, collecting data from some web service, GIS, FTP and process them for interoperability and for ingestion.	
R4	Smart Data Model compatibility	To guarantee interoperable and replicable Smart Cities, interoperability at level of data formats, FIWARE Smart Data Models, etc.	
R5	Smart City Federation	Model and data federation among platforms at level of protocols and APIs	
R6	Integration with workflow management systems	To enable ticket/event management. For example, when a fault is detected, it is highlighted in the SCDT and linked to a CMMS (Computerized Maintenance Management System).	

Data and Computing for Representation

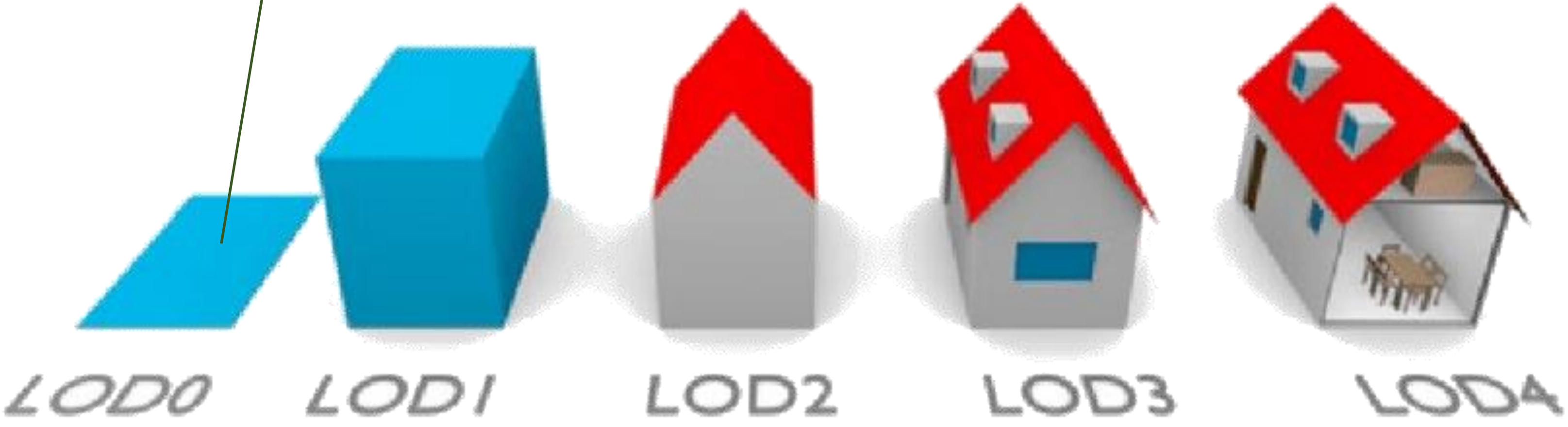
ID	Name	Description	
R7	Terrain information and elevation	Terrain elevation must be taken into account to properly elevate the city buildings and to model city hills and surrounding mountains	Data and computing for representation
R8	Ground information	Road shapes and names, names of squares and localities, etc., exploiting orthomaps, with eventual real aerial view patterns, and the graph road.	
R9	Heatmaps	To be superimposed (with variable transparency) on the ground level without overlapping the buildings, to represent distribution of temperature, pollutant, noise, humidity, vegetation, etc.	
R10	Paths and areas	To be used to describe perimeters/shapes of gardens, cycling paths, trajectories, borders of gov areas, elements of origin destination matrices, traffic flows, people flows, trajectories, pipes, severs, etc.	
R11	Data analytic	Data analytic processes must be available to let the user develops and/or execute specific data analytics: prediction, traffic flow reconstruction, anomaly detection.	
R12	Single Services	To mark the positions of services, IoT Devices, Point of Interest (POI), Key Performance Indicator (KPI), moving devices as fleets, etc.	
R13	Buildings of the city	Each single building should be represented. Multiple LoD could be included: (i) simple LoD1 structures, or (ii) higher LoD structures represented as 3D meshes, and (iii) BIMs	
R14	Automated 3D building construction	(i) 3D buildings must be created automatically, to be able to scale and replicate the SCDT framework; and (ii) the used software must be released with open or free license.	
R15	Additional 3D entities	To augment the realism of the 3D representation. For example (i) trees, benches, fountains, semaphores, and any other city furniture, and (ii) water bodies to better represent rivers, lakes, etc.	

Data and Computing for Representation

Level of details – LoD

- A system to categorize the building models included in a SCDT in five level
- Based on the CityGML Specification

LoD 0: simple 2D ground shapes

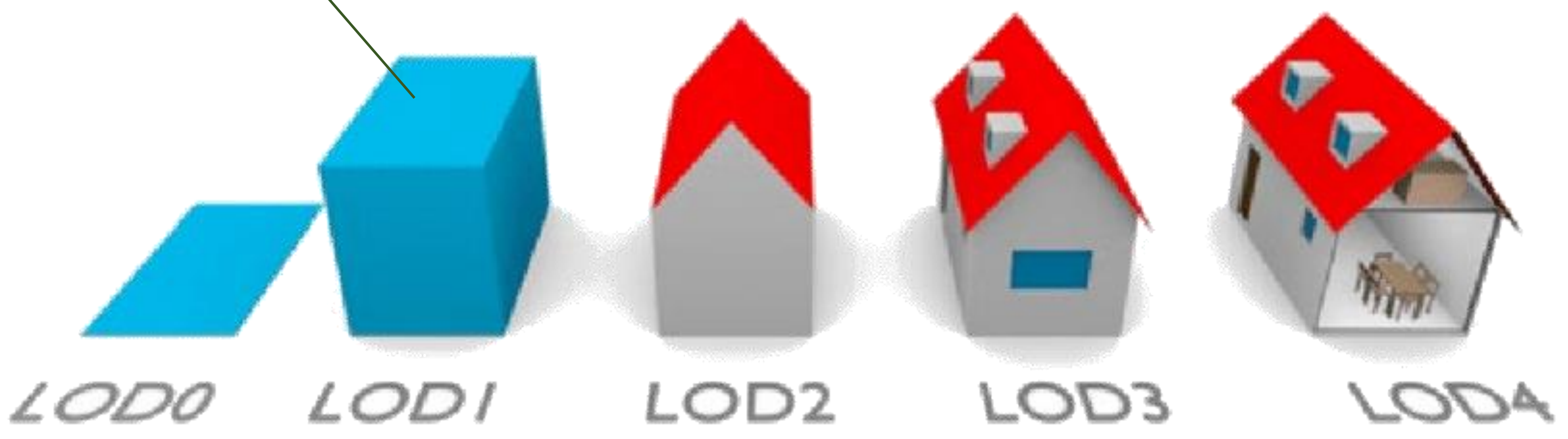


Data and Computing for Representation

Level of details – LoD

- A system to categorize the building models included in a SCDT in five level
- Based on the CityGML Specification

LoD 1: 3D volumes obtained by extrusion using building height

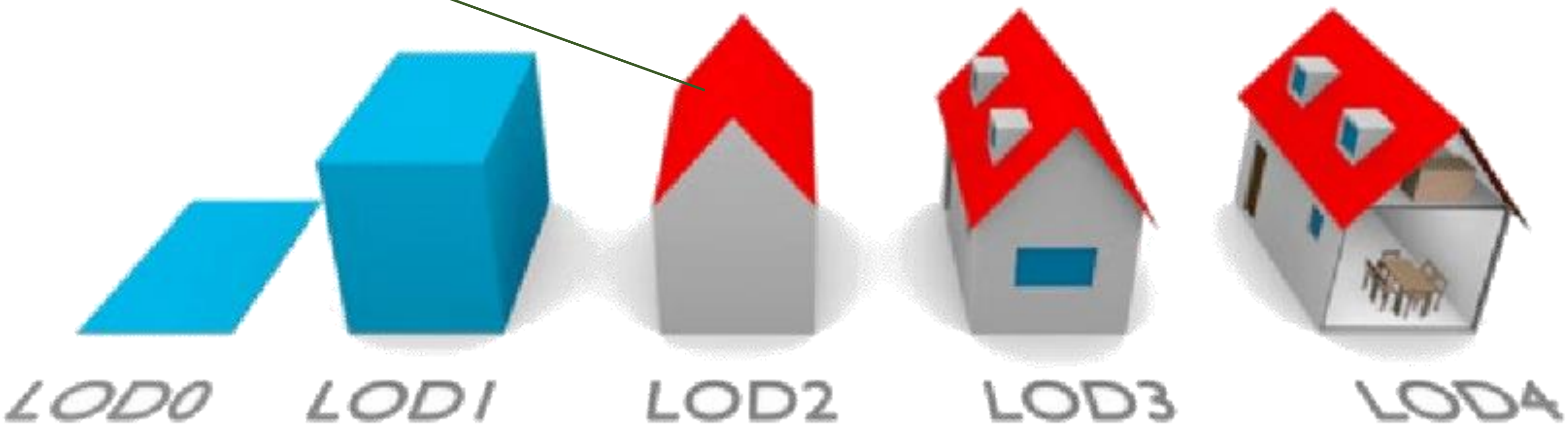


Data and Computing for Representation

Level of details – LoD

- A system to categorize the building models included in a SCDT in five level
- Based on the CityGML Specification

LoD 2: 3D volumes with 3D rooftops

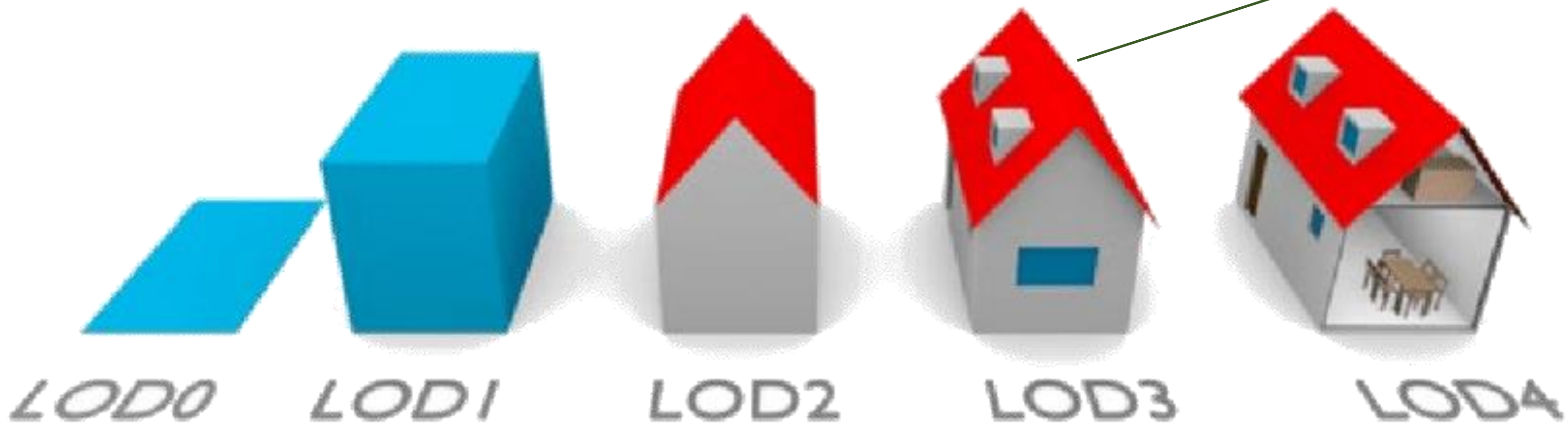


Data and Computing for Representation

Level of details – LoD

- A system to categorize the building models included in a SCDT in five level
- Based on the CityGML Specification

LoD 3: 3D volumes with 3D rooftops, details, and textures

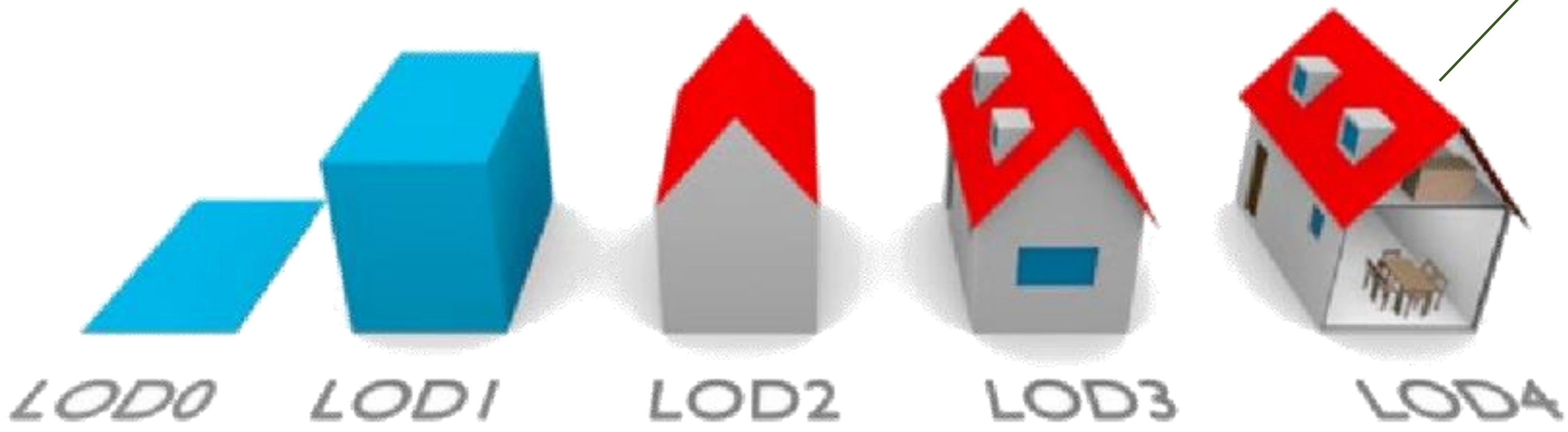


Data and Computing for Representation

Level of details – LoD

- A system to categorize the building models included in a SCDT in five level
- Based on the CityGML Specification

LoD 4: 3D volumes with 3D rooftops, details, and textures + indoor



Distribution and Interaction

ID	Name	Description	
R16	Dynamic 2D/3D structures	Elements such as PINs, shapes, paths, should be represented in 3D dynamically, changing color and shape according to their kind or some real-time value.	Distribution and interaction
R17	Dynamic data management	To have elements to be automatically reported in the SCDT as soon as they are included in the platform, event driven rendering of data.	
R18	No reloading	Changes in the SCDT must be rendered without the needs of a full reload of the map.	
R19	View Map controls	To change the point of view by zooming, rotating, tilting, and panning the scene.	
R20	Dynamic sky and lighting	To model and show different sky conditions and to change the light source position, simulating different times of day/night.	
R21	Building picking/manipulation	To select single building to: (i) show detailed information, or (ii) move into a BIM view of the building, or (iii) to change the building 3D model.	
R22	Services and element data access	To show data associated with IoT Devices, POI, KPI, shapes, paths, etc., including real time and historical data.	
R23	Independent element management	To hide, show, replace specific elements (e.g., to disable the building view to see only the city PINs, or to load different heatmaps or paths)	
R24	Web player	The SCDT must (i) be accessible thought a web browser without additional plugins, and (ii) the player must be released with open or free license.	
R25	Business logic call-back	To provide the possibility of selecting an element (3D, PIN, ground, heatmap) to provoke a call back into a business logic tool for intelligence activities, analytics, etc.	
R26	Underground and elements inspection	To provide the possibility of selecting and inspecting specific areas and see detailed 3D elements placed underground, such as water pipes, metro lines, etc.	

State of the Art

- Efforts to produce SCDTs have been undertaken by several cities around the world. For example:
  - Helsinki, Gothenburg, Paris, Rennes, London, New Castle, Rotterdam, Berlin, Stockholm, Zurich, Herrenberg, Munich in Europe;
  - Toronto, Boston in Nord America;
  - Amaravati, Singapore, Shanghai in Asia;
  - Wellington in Oceania.
- Most of these solutions seems to be work-in-progress or limited case of studies
- They do not provide open access application to inspect and test their SCDT

State of the Art - Helsinki

✓LoD3 Models

✓Ground orthomap

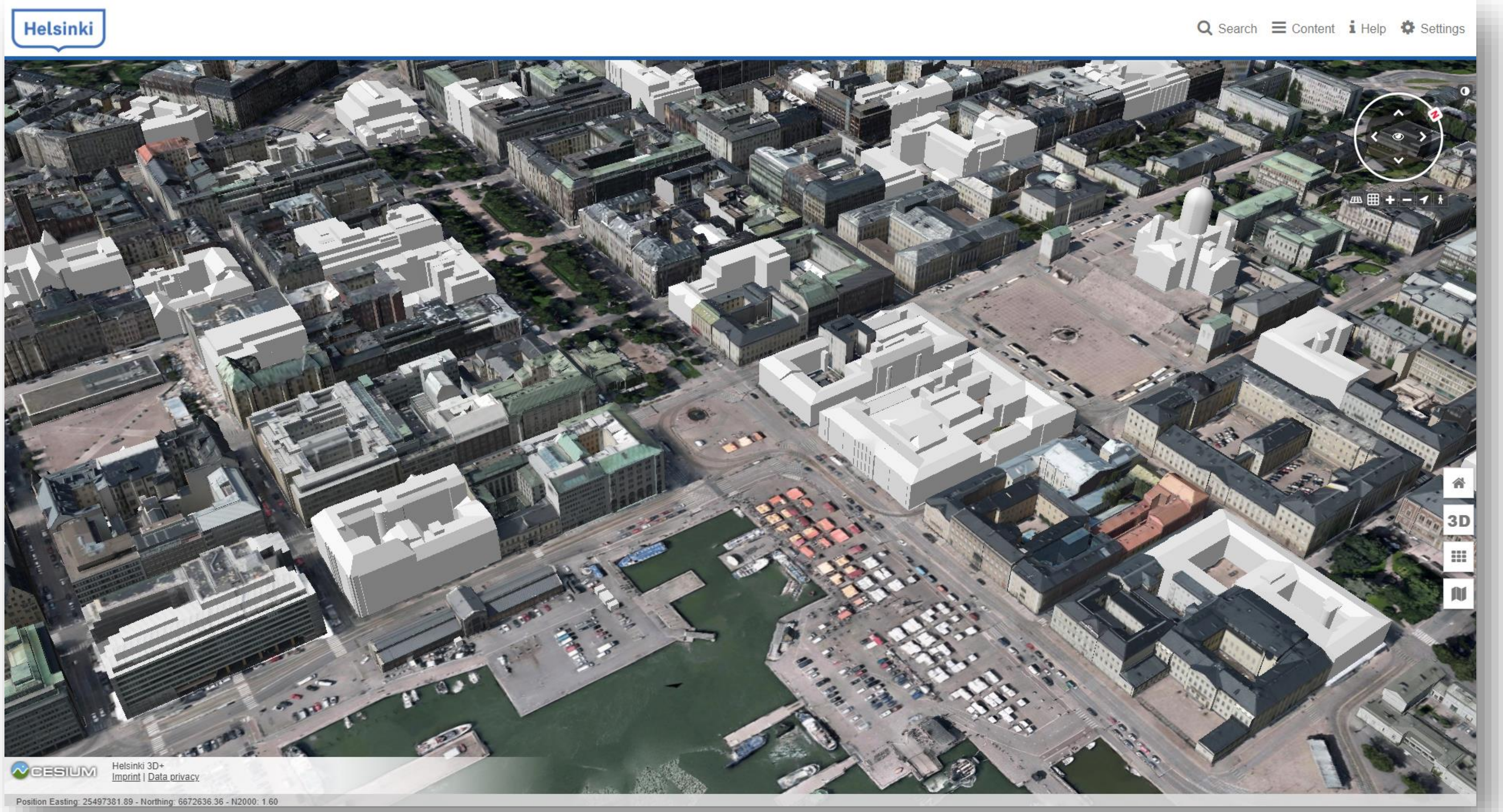
✓Terrain elevation

X PINs

X Heatmaps

X Paths

X Additional 3D entities

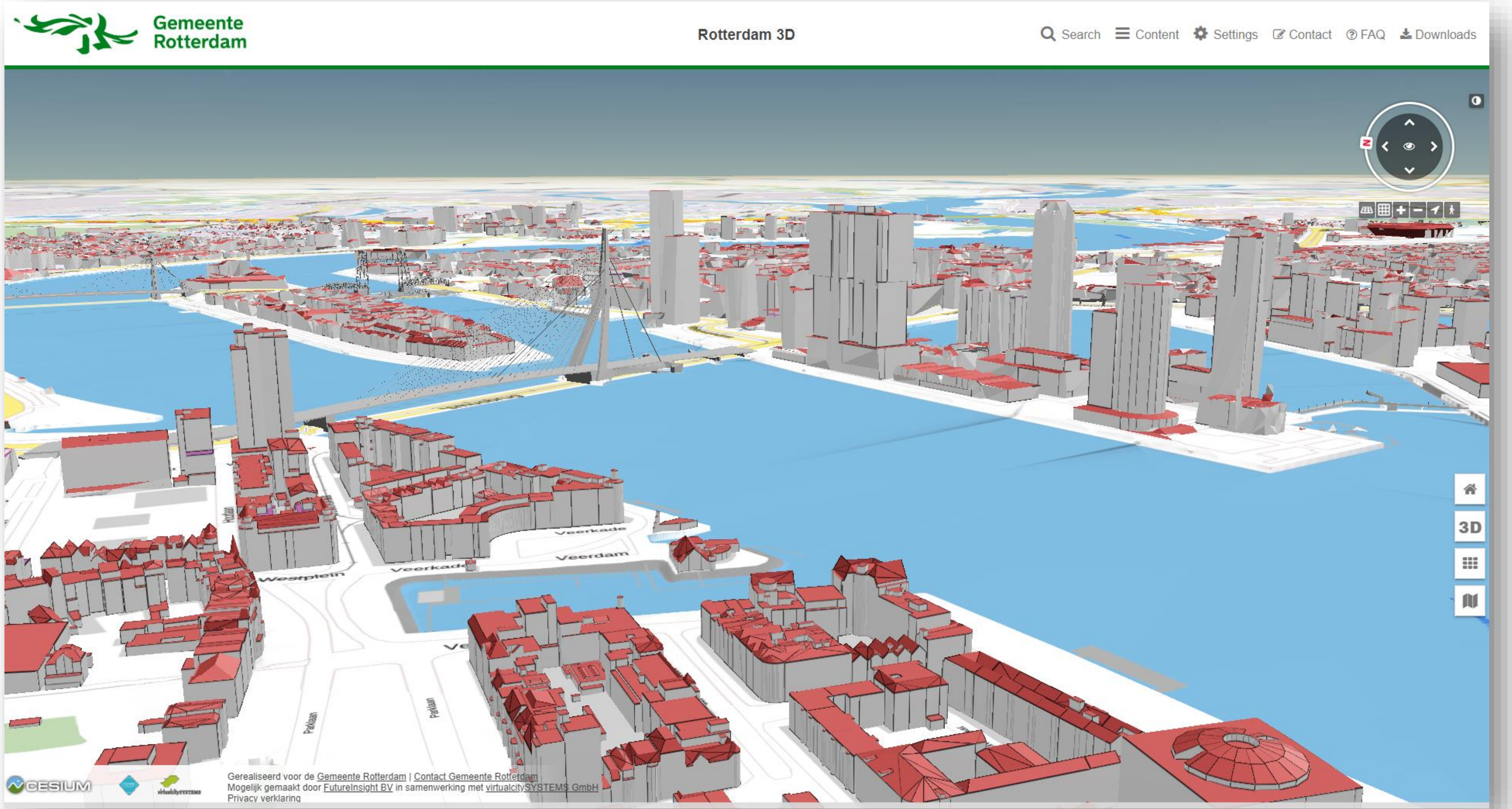


https://kartta.hel.fi/3d/#/

State of the Art - Rotterdam

- ✓LoD2 Models
- ✓Ground orthomap
- ✓Paths

- X Terrain elevation
- X PINs
- X Heatmaps
- X Additional 3D entities



https://www.3drotterdam.nl

State of the Art - Berlin

✓LoD3 Models

✓Ground orthomap

✓Paths

X Terrain elevation

X PINs

X Heatmaps

X Additional 3D entities

https://www.3dcitydb.org/3dcitydb-web-map/1.7/3dwebclient/index.html?title=Berlin_Demo&batchSize=1&latitude=52.517479728958044&longitude=13.411141287558161&height=534.3099172951087&heading=345.2992773976952&pitch=-44.26228062802528&roll=359.933888621294&layer_0=url%3Dhttps%253A%252F%252Fwww.3dcitydb.org%252F3dcitydb%252Ffileadmin%252Fmydata%252FBerlin_Demo%252FBerlin_Buildings_rgbTexture_ScaleFactor_0.3%252FBerlin_Buildings_rgbTexture_collada_MasterJSON.json%26name%3DBrlin_Buildings_rgbTexture%26active%3Dtrue%26spreadsheetUrl%3Dhttps%253A%252F%252Fwww.google.com%252Ffusiontables%252FDataSource%253Fdocid%253D19cuclDgIHMqrRQyBwLEztMLeGzP83IBWfEtKQA3B%2526pli%253D1%2523rows%253Aid%253D1%26cityobjectsJsonUrl%3D%26minLodPixels%3D100%26maxLodPixels%3D1.7976931348623157e%252B308%26maxSizeOfCachedTiles%3D200%26maxCountOfVisibleTiles%3D200

State of the Art - Stockholm

✓LoD3 Models

✓Ground orthomap

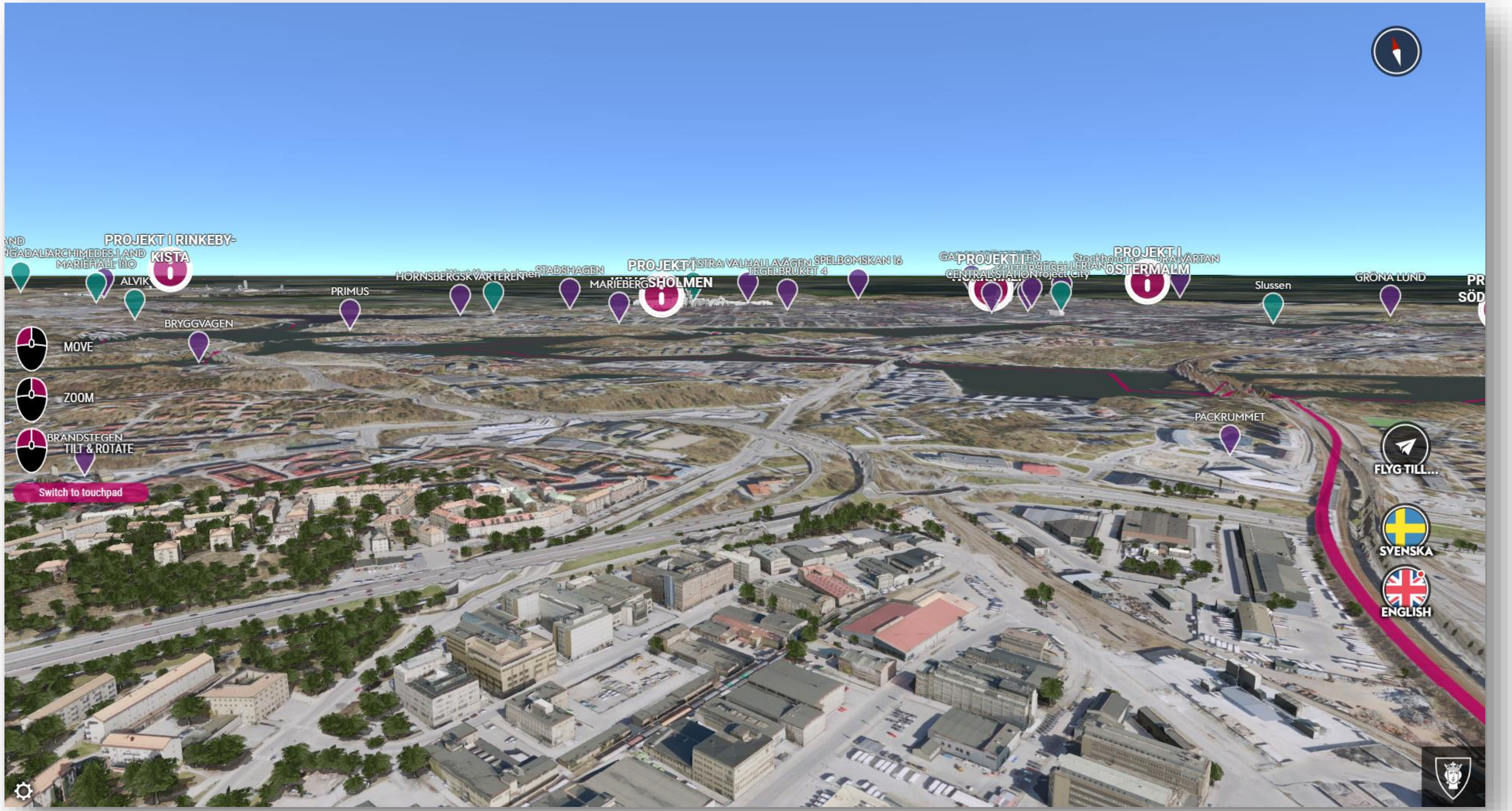
✓Terrain elevation

✓PINs (for POIs)

✓Additional 3D entities

X Heatmaps

X Paths

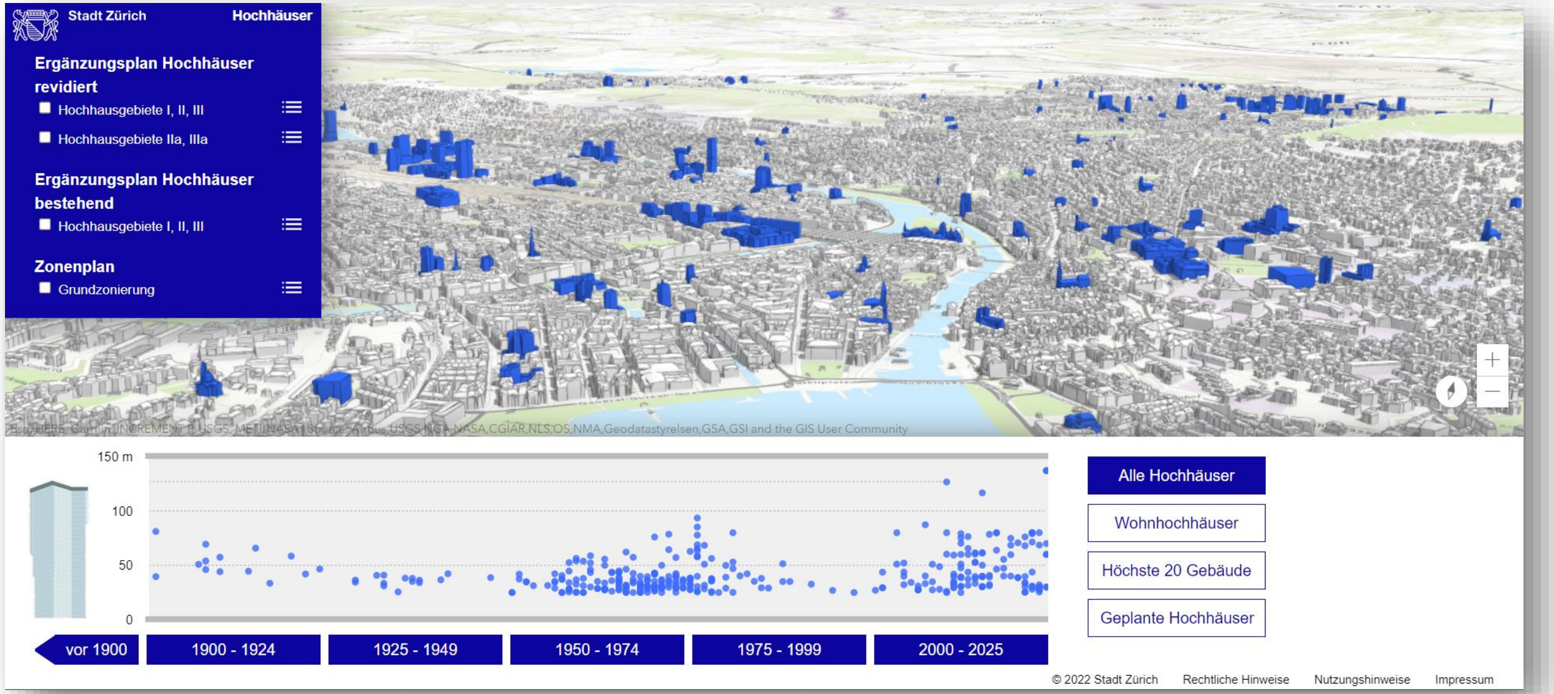


https://eu.opencitiesplanner.bentley.com/stockholm/stockholmvaxer

State of the Art - Zurich

- ✓LoD2 Models (pickable)
- ✓Additional 3D entities
- ✓Terrain elevation
- ✓Paths
- ✓Illumination conditions

- X Heatmaps
- X PINs

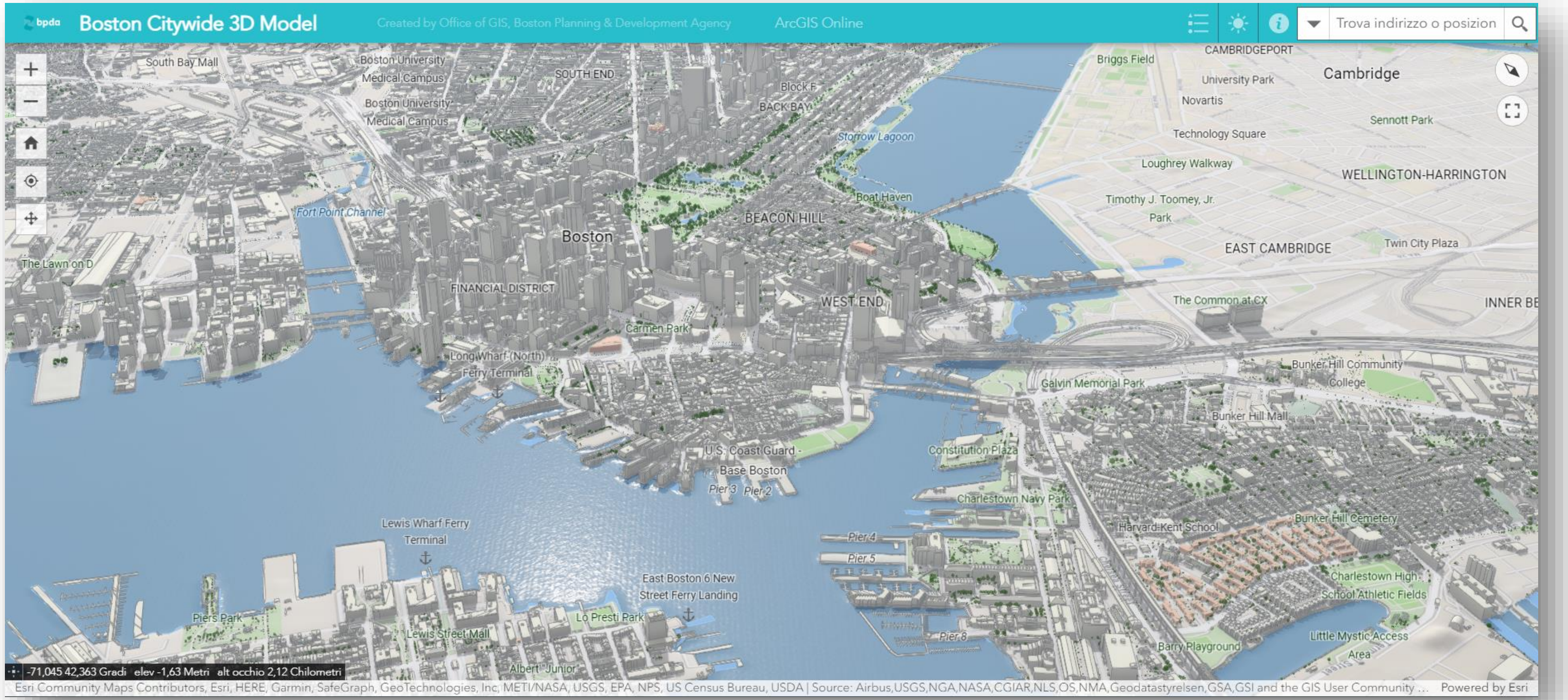


https://hochhaeuser.stadt-zuerich.ch/

State of the Art - Boston

- ✓LoD2 Models
- ✓Additional 3D entities
- ✓Terrain elevation
- ✓Illumination condition

- X Heatmaps
- X PINs
- X Paths



https://boston.maps.arcgis.com/apps/webappviewer3d/index.html?id=cf3415dea19d480caa71eb5dbdce185f

State of the Art – Wellington & Shanghai

- Based on Unreal Engine 5
- Seems to be advanced projects
- No open access
- The usage of a high-end 3D engine limits the distribution and accessibility

https://buildmedia.com/work/wellington-digital-twin

https://www.unrealengine.com/en-US/spotlights/51world-creates-digital-twin-of-the-entire-city-of-shanghai

State of the Art – Considerations

- The open access solution are more 3D city models than SCDT
- Low level of interoperability
- Low integration of real-time data
- Limited user experience

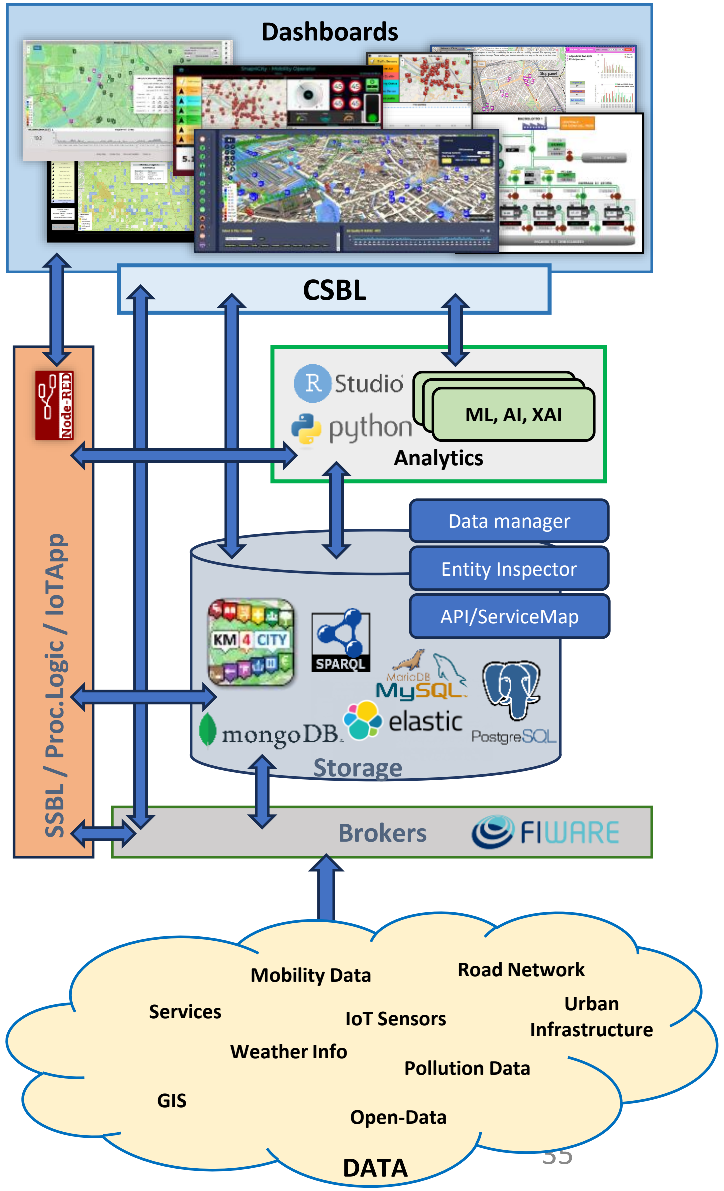
	Helsinki	Rotterdam	Berlin	Stockholm	Zurich	Boston	Snap4City
R1	No	No	No	No	No	No	Yes
R2	Yes (PostGIS)	No	No	No	Yes (Geoportal)	No	Yes (Km4City)
R3	No	No	No	No	No	No	Yes
R4	No	No	No	No	No	No	Yes
R5	No	No	No	No	No	No	Yes
R6	No	No	No	No	No	No	Yes (*)
R7	Yes	No	No	Yes	Yes	Yes	Yes
R8	Yes	Yes	Yes	Yes (fixed)	Yes	Yes (fixed)	Yes
R9	No	No	No	No	No	No	Yes
R10	Yes	Yes	No	Yes	Yes	No	Yes
R11	No	No	No	No	No	No	Yes
R12	Yes	No	No	Yes	No	No	Yes
R14.i	Yes	Yes	No	No	Yes	No	Yes
R14.ii	Yes (LoD3)	Yes (LoD3)	Yes (LoD2)	Yes (LoD3)	Yes (LoD2)	Yes (LoD2)	Yes (LoD3)
R14.iii	No	No	No	No	No	No	Yes
R15.i	Yes	Yes	No	Yes	No	No	Yes
R15.ii	Non free	Non free	n/a	Non free	Non free	n/a	Yes
R16.i	Yes	No	No	Yes	Yes	Yes	Yes
R16.ii	No	Yes	No	No	No	Yes	No
R17	Yes	No	No	Yes	No	No	Yes
R18	No	No	No	No	No	No	Yes
R19	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R20	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R21	Yes	Yes	Yes	No	Yes	Yes	Yes
R22.i	Yes	Yes	Yes	No	Yes	Yes	Yes
R22.ii	No	No	No	No	No	No	Yes
R22.iii	No	No	No	No	No	No	Yes
R13	Yes	Yes	No	Yes	No	No	Yes
R23	Yes	Yes	Yes	No	Yes	No	Yes
R24.i	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R24.ii	Non-free	Free	Free	Non-free	Limited	Non-free	Free
R25	No	No	No	No	No	No	Yes
R26	Yes (*)	Yes (*)	No	No	No	No	No

State of the Art – Considerations

- Differently from SotA solutions, our Smart City Digital Twin framework is built on top of a **IoT platform**
- An IoT platform is fundamental to meet **most of the requirements on data and integration**
- In the development of the SCDT of Florence, the **Snap4City** platform was used

Snap4City Platform

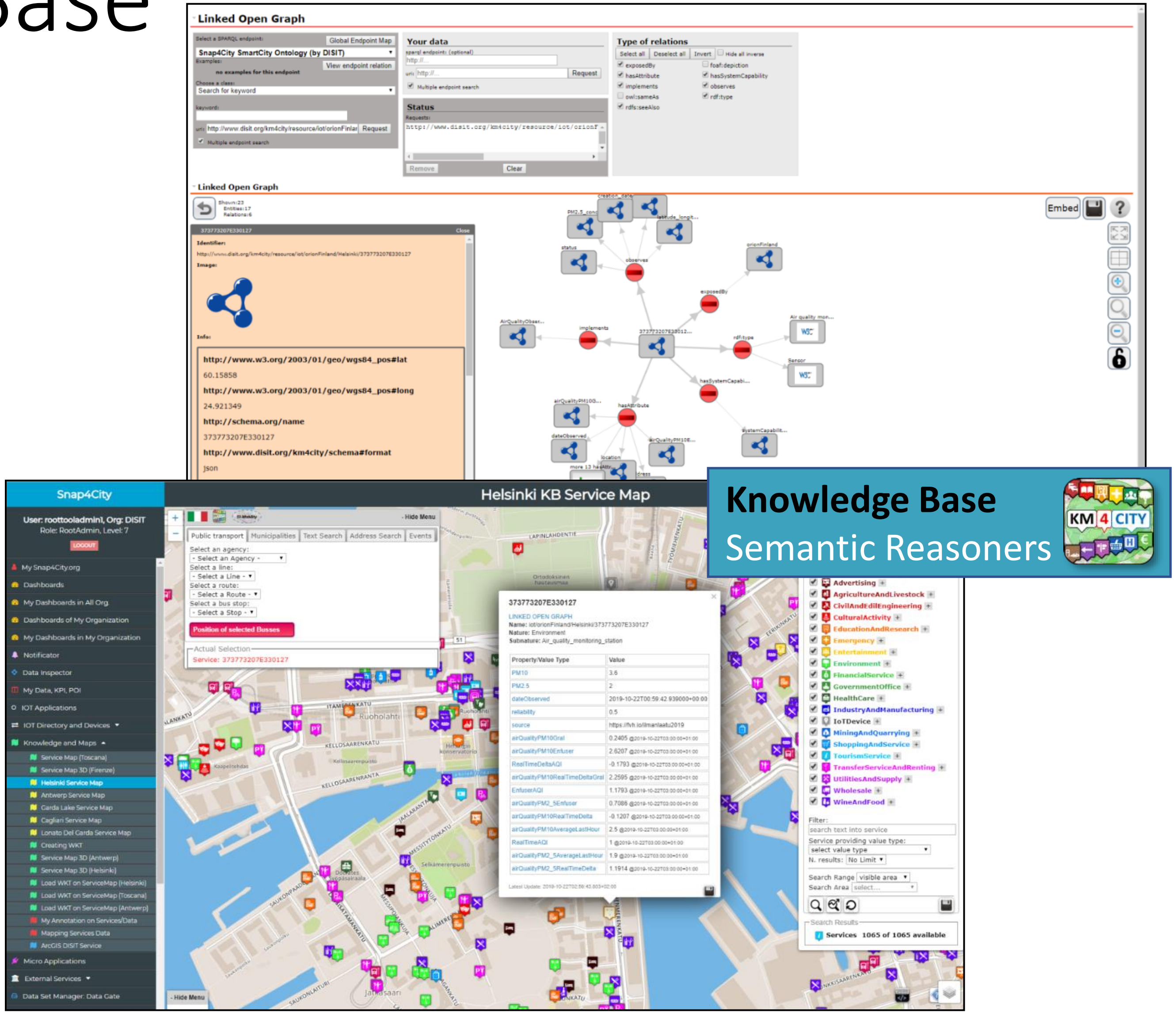
- Snap4City is an **open-source IoT platform** developed at DISIT Lab of the University of Florence, with continuous evolution from 2016
- The platform handles **heterogeneous data sources** (IoT devices, Open Data, external services, etc.) exploiting multiple storages with semantic enrichment
- Data are retrieved by **dedicated APIs** and exploited by **data analytic processes** and **IoT applications**
- **2D and 3D dashboards** can be built using a large number of widgets



KM4City Knowledge Base

- Collected static and real-time data are **semantically indexed** in a graph based RDF Knowledge Base, named **KM4City**

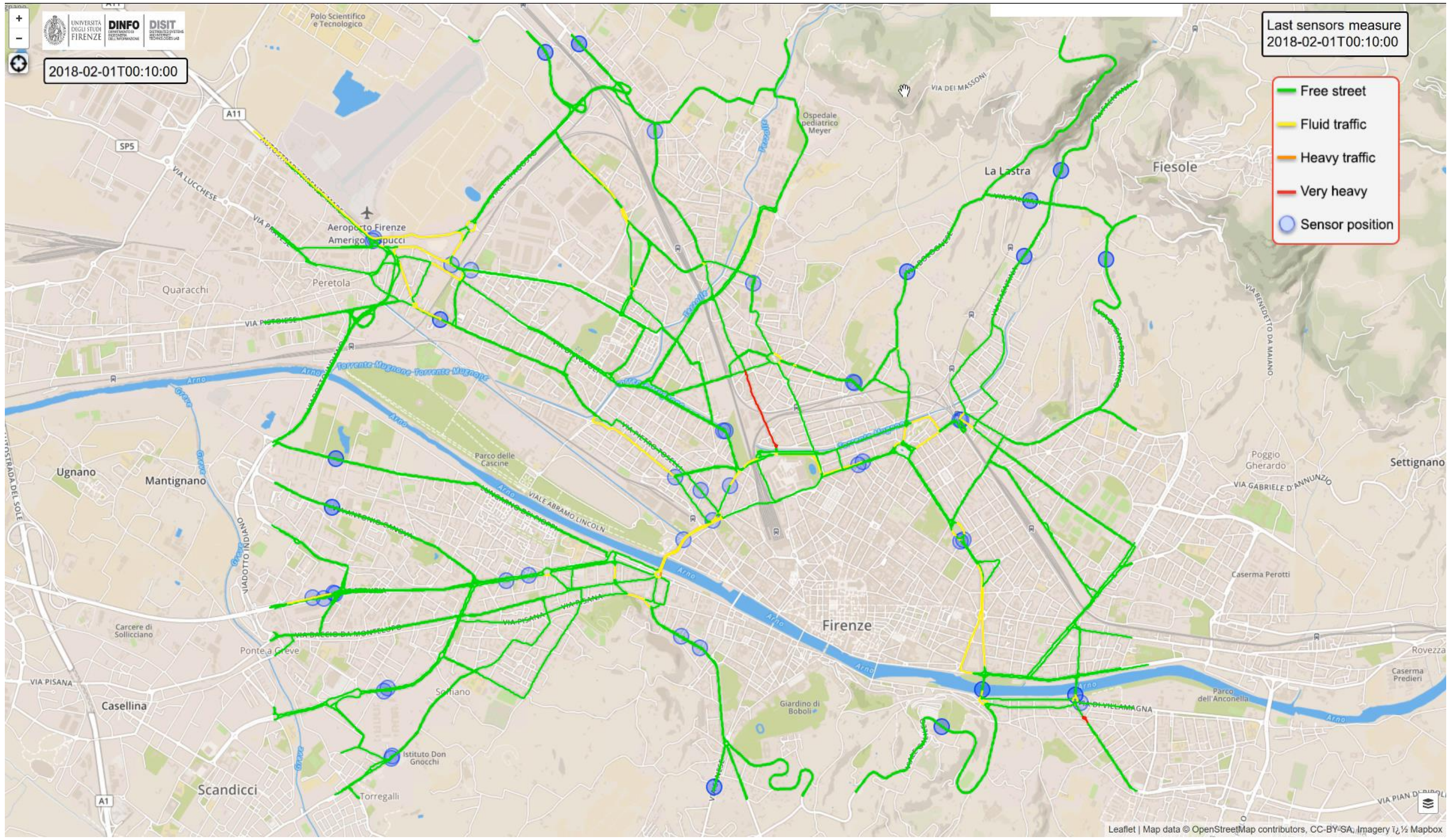
- Several API have been defined in order to **retrieve data from the KB** using relational, spatial, and temporal queries



Data Analytics in Snap4City

More than **90 data analytics processes** are available!

- **Traffic flow reconstruction** based on a fluid-dynamic model

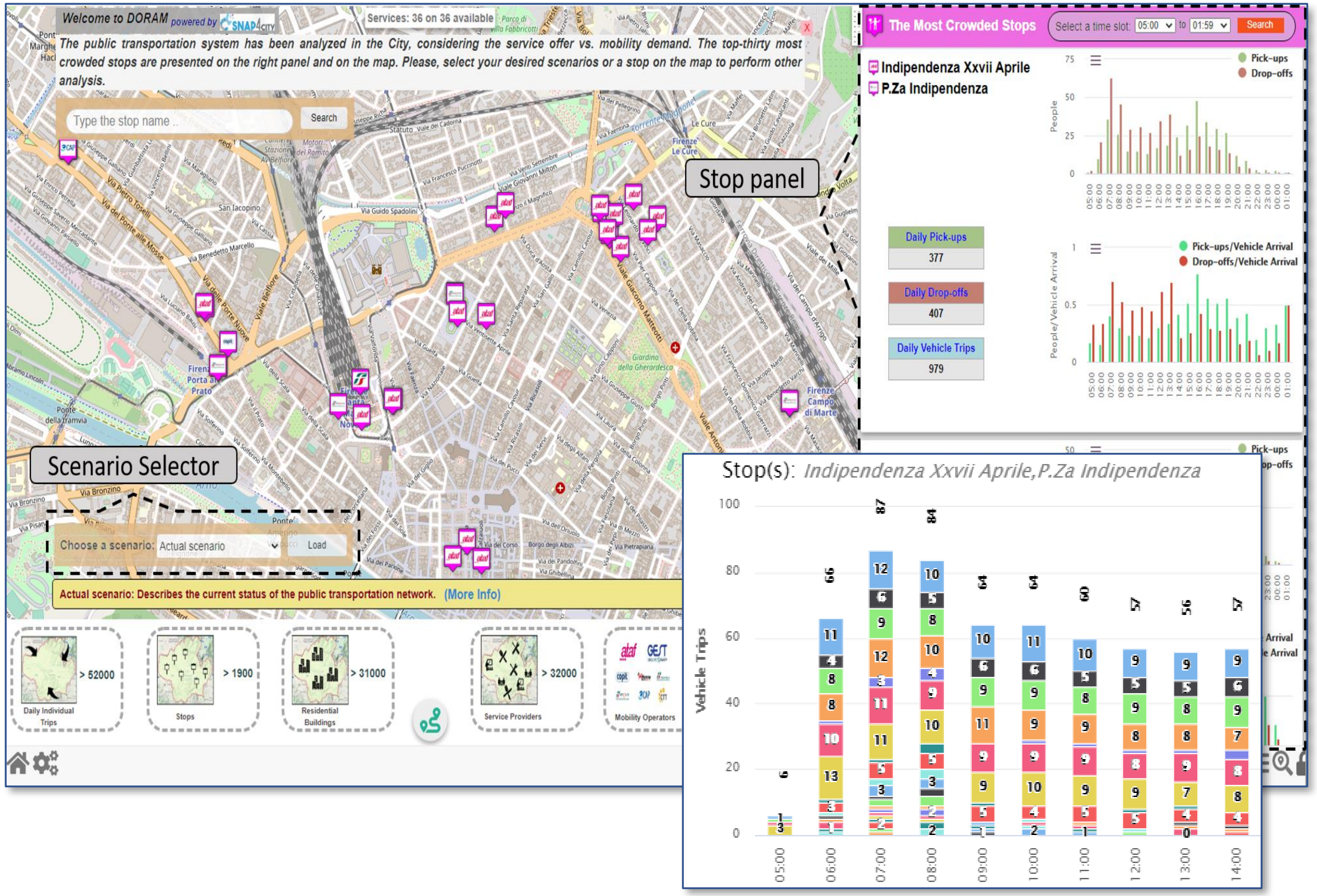


Data Analytics in Snap4City

More than **90 data analytics processes** are available!

- **Traffic flow reconstruction** based on a fluid-dynamic model

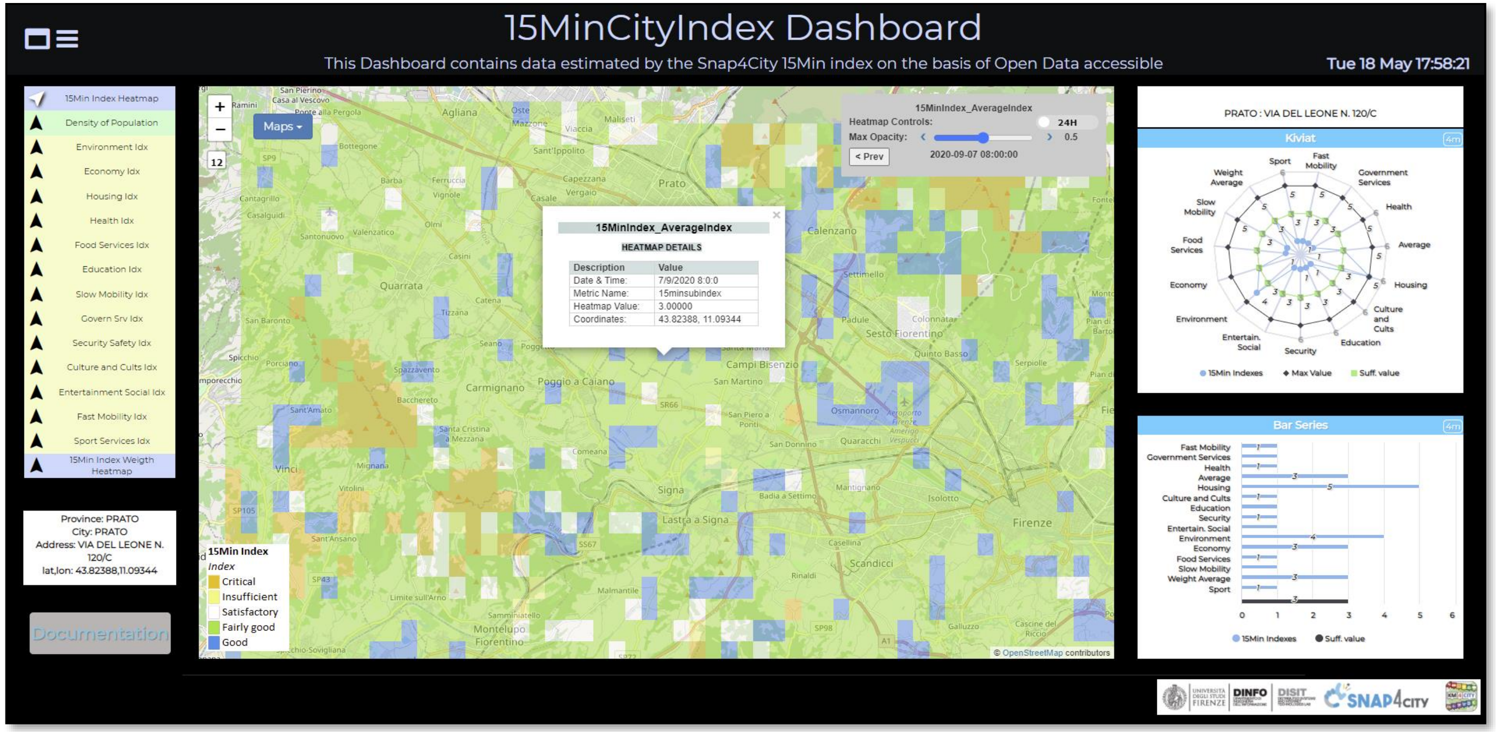
- Analysis of **public transport offer and demand**



Data Analytics in Snap4City

More than **90 data analytics processes** are available!

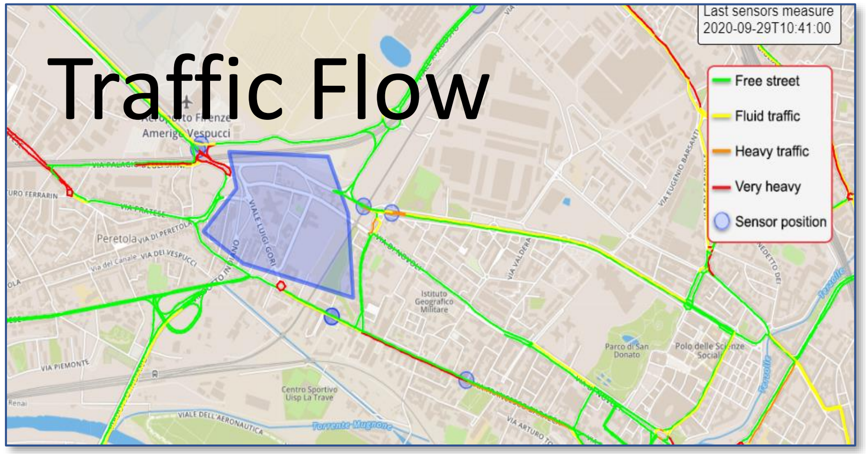
- **Traffic flow reconstruction** based on a fluid-dynamic model
- Analysis of **public transport offer and demand**
- Assessment of the **15-minute index**



Data Analytics in Snap4City

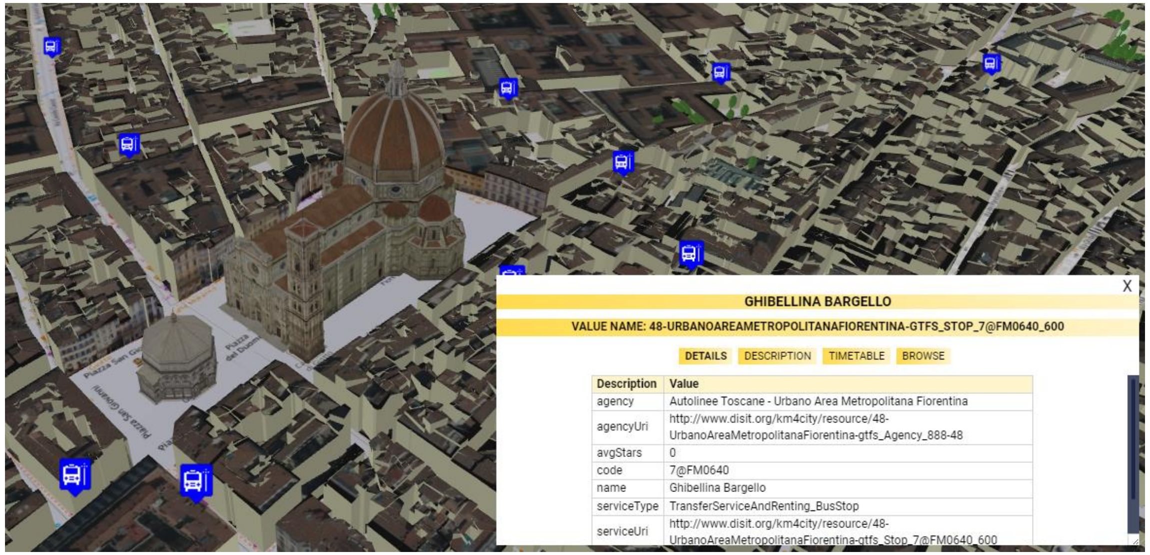
More than **90 data analytics processes** are available!

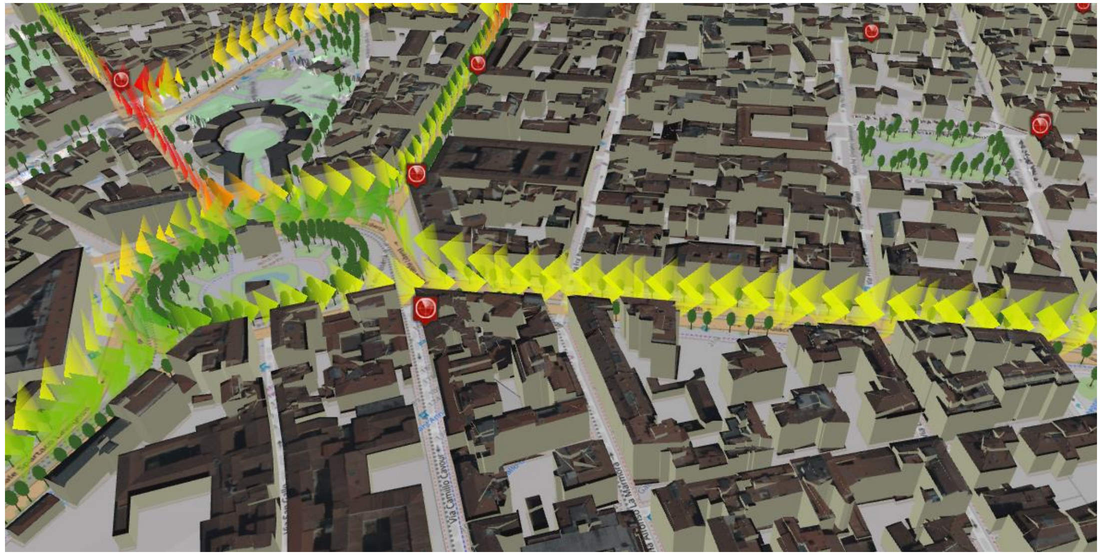
- **Traffic flow reconstruction** based on a fluid-dynamic model
- Analysis of **public transport offer and demand**
- Assessment of the **15-minute index**
- Capabilities to perform **What-If analysis** for routing and traffic congestion



Data Analytics in Snap4City

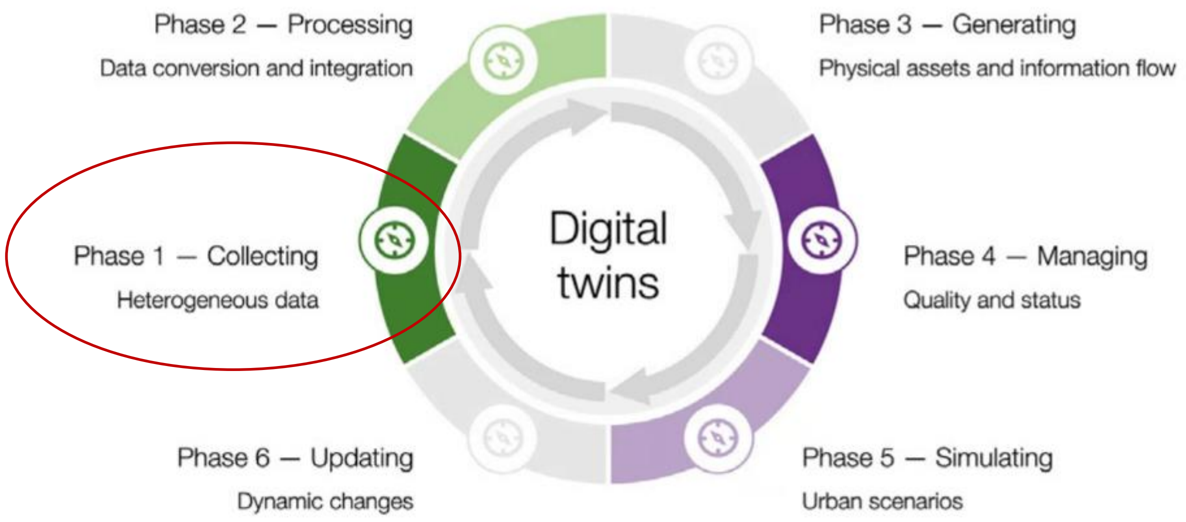
- All the data available in the KB and produced by the analytics can eventually be **visualized into the SCDT**





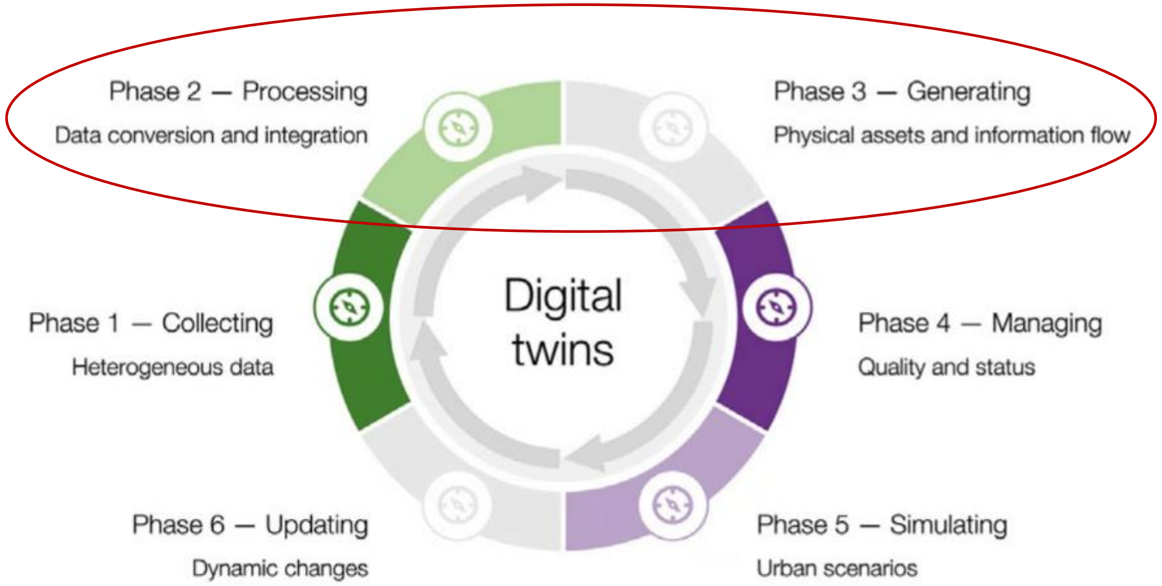
SCDT Development Phases

- **Data acquisition**: city graph, IoT sensor/actuators, POIs, orthomaps, paths, digital surface model (DSM) and terrain elevation (DTM), images, etc.



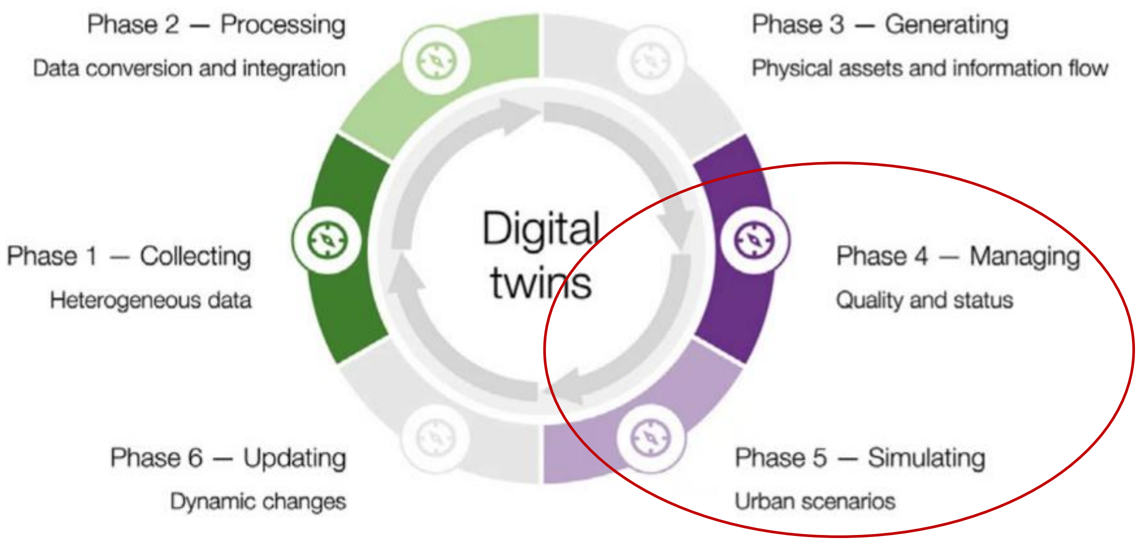
SCDT Development Phases

- **Data acquisition**: city graph, IoT sensor/actuators, POIs, orthomaps, paths, digital surface model (DSM) and terrain elevation (DTM), images, etc.
- **Production**: Heatmaps computation, traffic flow reconstruction, ODM productions, 3D building construction, etc.



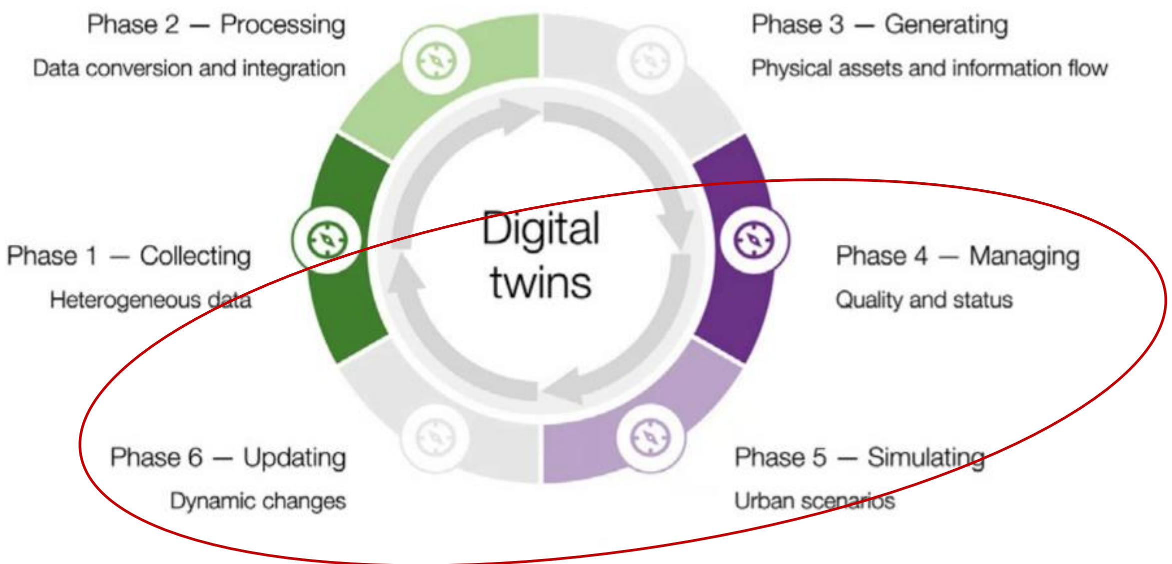
SCDT Development Phases

- **Data acquisition**: city graph, IoT sensor/actuators, POIs, orthomaps, paths, digital surface model (DSM) and terrain elevation (DTM), images, etc.
- **Production**: Heatmaps computation, traffic flow reconstruction, ODM productions, 3D building construction, etc.
- **Analyses and simulations**: acquired and produced data are exploited by analytics



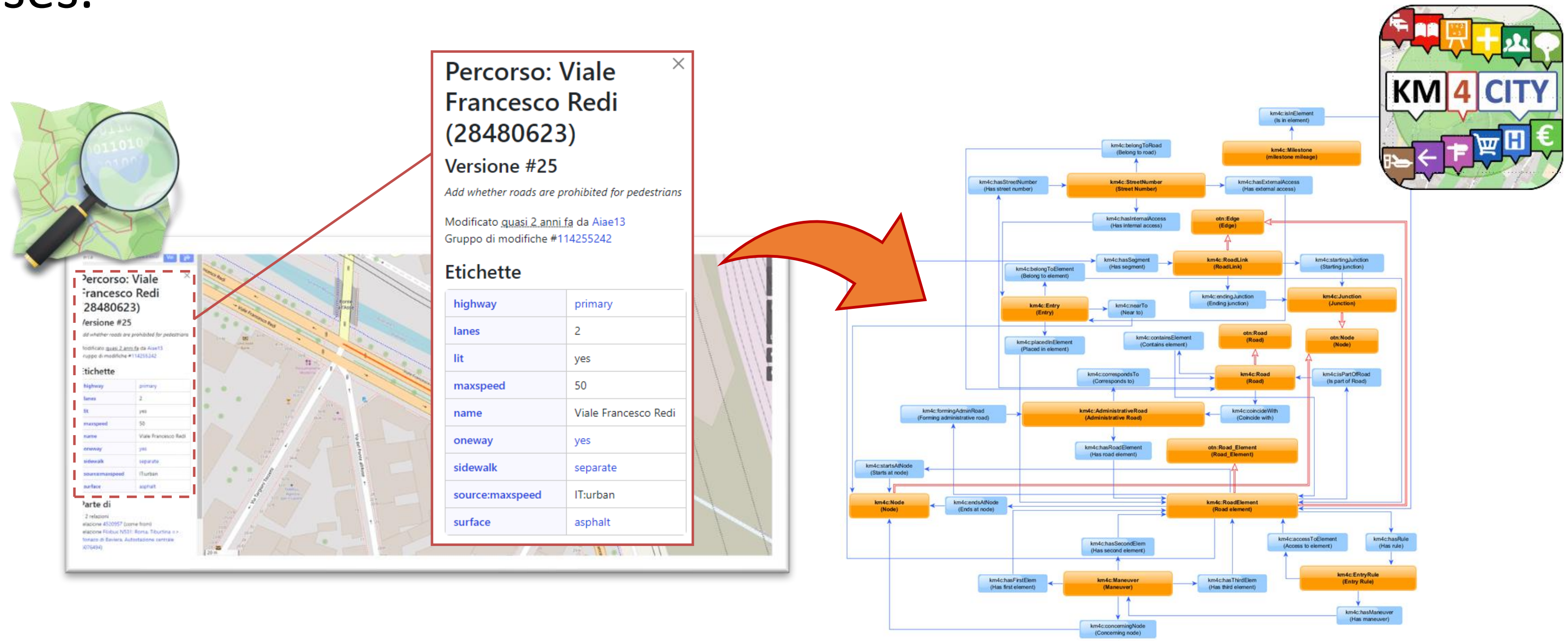
SCDT Development Phases

- **Data acquisition**: city graph, IoT sensor/actuators, POIs, orthomaps, paths, digital surface model (DSM) and terrain elevation (DTM), images, etc.
- **Production**: Heatmaps computation, traffic flow reconstruction, ODM productions, 3D building construction, etc.
- **Analyses and simulations**: acquired and produced data are exploited by analytics
- **User interaction**: acquired and produced data are integrated into a global digital twin model and rendered as 3D multi-data map and distributed as an interactive web interface.



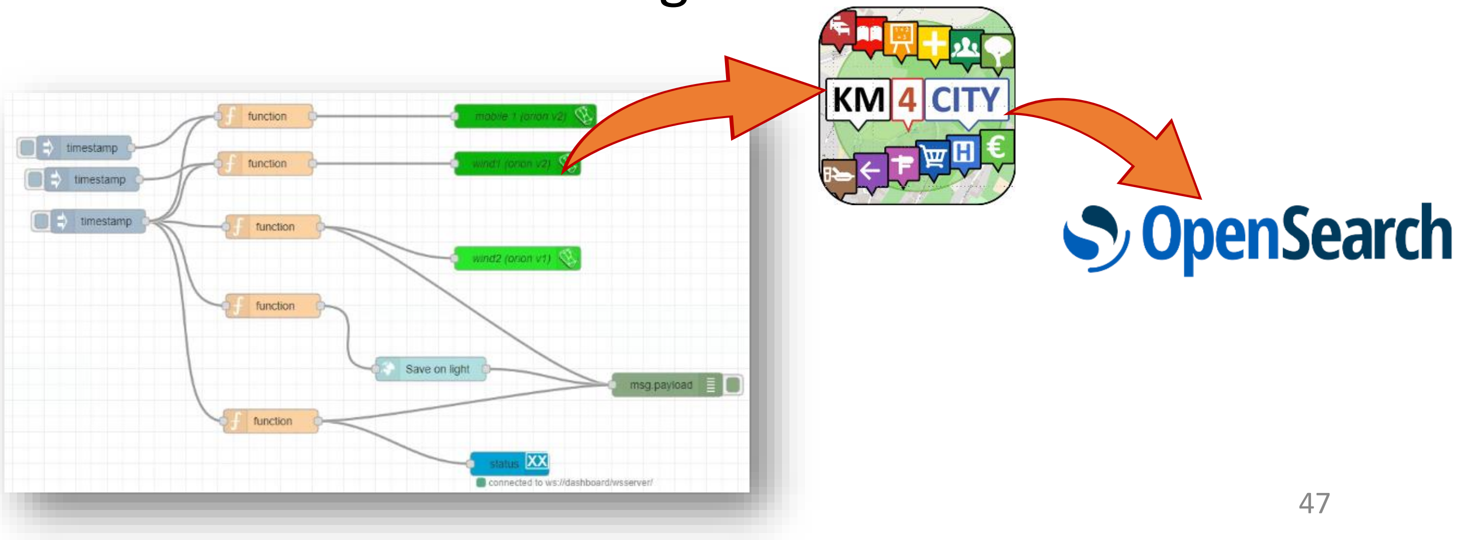
Data acquisition

- Data acquisition is handled directly by the Snap4City platform
  - **Static data** are ingested with ETL process and sent to the KB or to specific databases.



Data acquisition

- Data acquisition is handled directly by the Snap4City platform
  - **Static data** are ingested with ETL process and sent to the KB or to specific databases. We talked about the road graph ingestion from OSM to KM4City
  - **Real-time data** from IoT sensors can be ingested using IoT App (i.e., NoreRED flows) and sent to an OpenSearch cluster for indexing

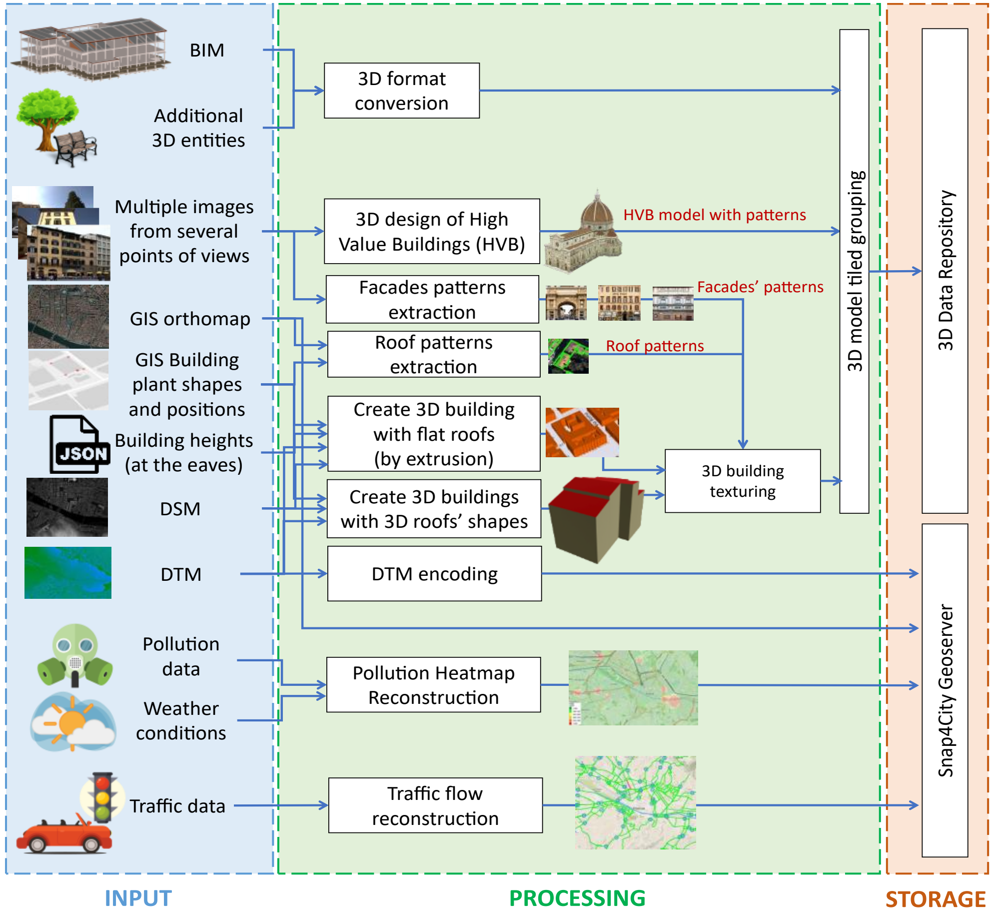


UNIVERSITÀ DEGLI STUDI FIRENZE
DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE
DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB DISTRIBUTED DATA INTELLIGENCE AND TECHNOLOGIES LAB
SNAP4CITY

Production

- Ingested data can then be used as input for analytic processes in order to obtain
  - Traffic flow reconstruction
  - Heatmaps, e.g., to describe the pollutant dispersion
  - Etc…

- In this phase, the 3D map to be shown in the SCDT is built



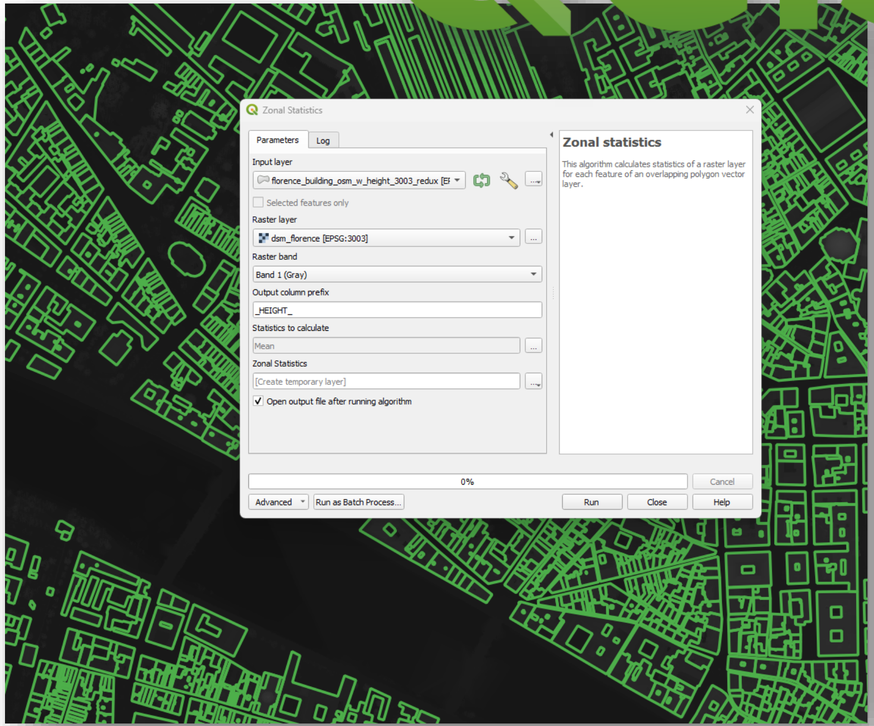
3D Map Production

- To obtain 3D building models, a **Digital Surface Model (DSM)** of Florence was used

- Our algorithm exploits also the **OSM building shapes**, in order to segment building patches from the DSM

3D Map Production – LoD1 Models

- LoD1 model can be easily obtained using a **GIS software** (e.g., QGIS)
  - Load the OSM shapefile (Esri shp, GeoJSON, etc.)
  - Load the DSM file (GeoTIFF, ASC, etc.)
  - Use the *Zonal Statistic* algorithm of the processing toolbox to compute the average DSM value (i.e., height) within each polygon



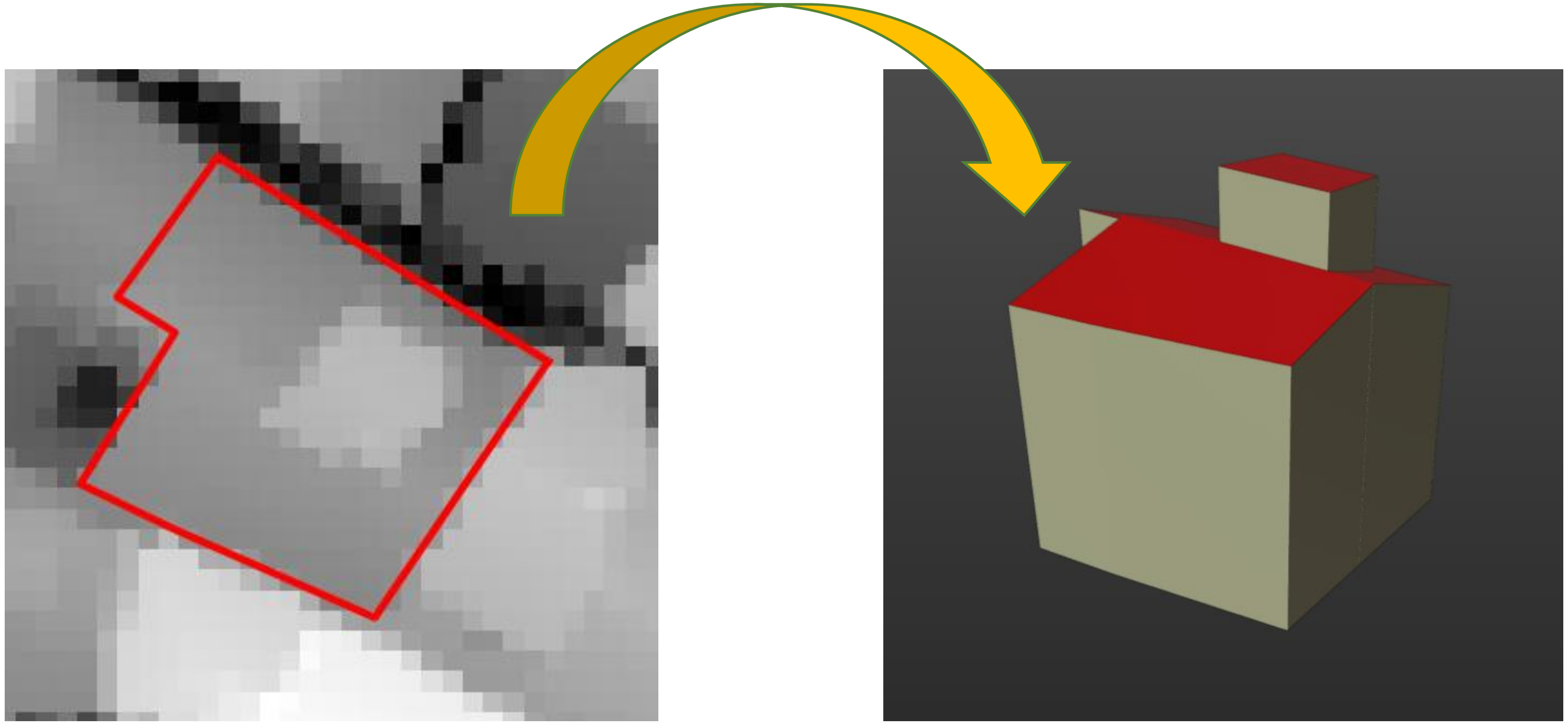
3D Map Production – LoD1 Models

- LoD1 model can be easily obtained using a **GIS software** (e.g., QGIS)
- LoD1 model can also be available from free services
- For example, **OSM Buildings** offer 3D models for several cities freely available for non-commercial use

3D Map Production – LoD3 Models

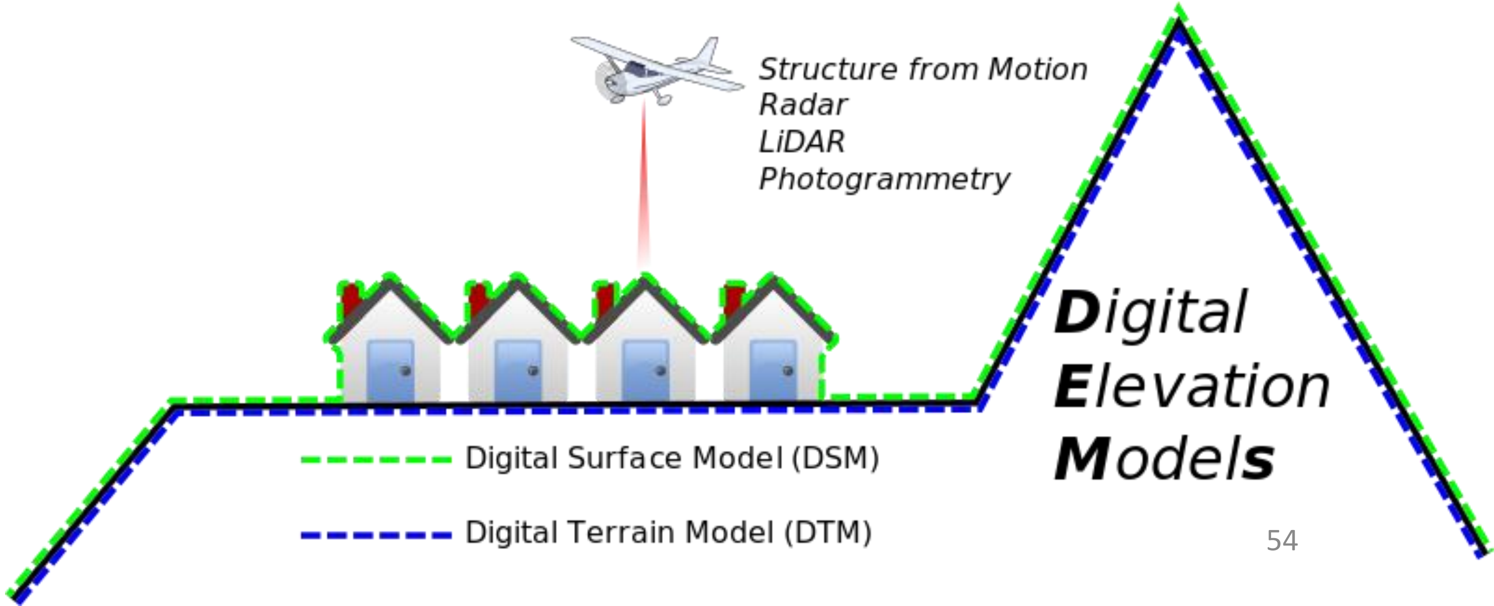
- Construction of LoD3 models is more complex
- The **3D shape of the rooftops** must be estimated from the DSM
- **Roof texture** must be extracted from aerial orthophotos
- **Facade texture** must be acquired and applied to the models

Rooftop modelling



Digital Elevation Model

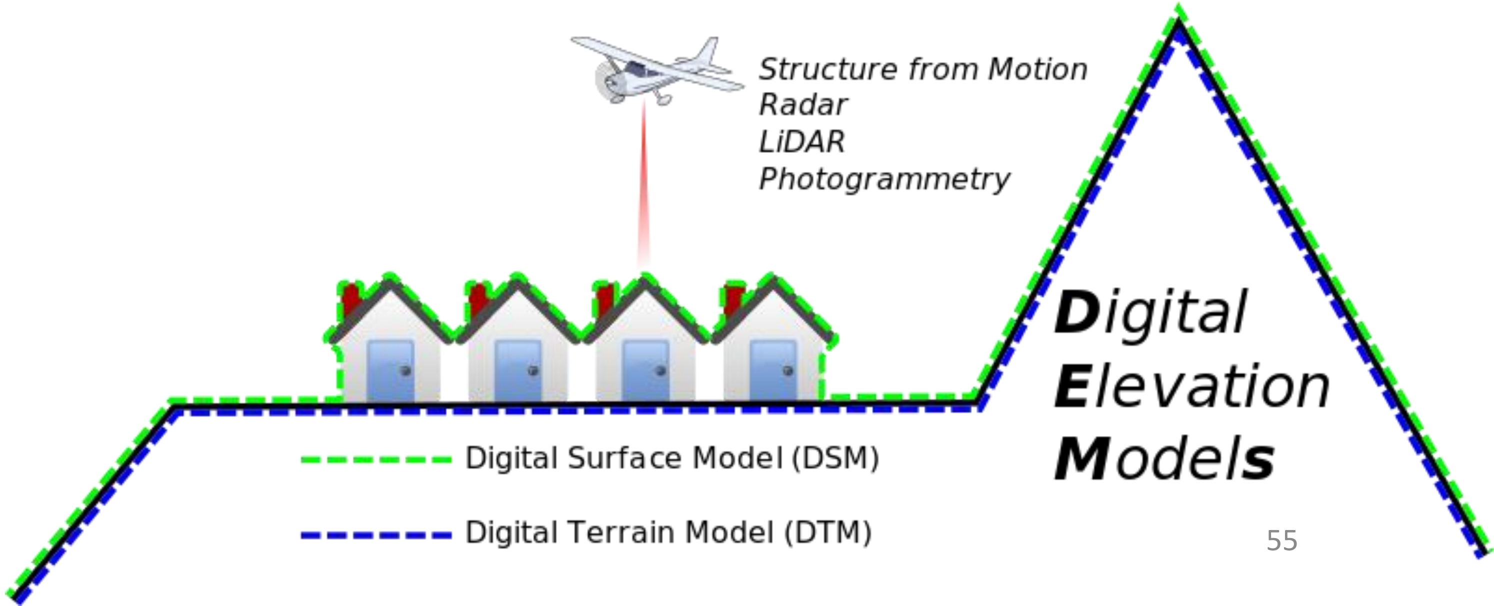
- DEM is obtained by aerial acquisition using
  - Lidar
  - Radar
  - Photogrammetry



Digital Elevation Model

- DEM is obtained by aerial acquisition using
  - Lidar
  - Radar
  - Photogrammetry

- DEM is composed by
  - DTM: Digital **Terrain** Model
  - DSM: Digital **Surface**

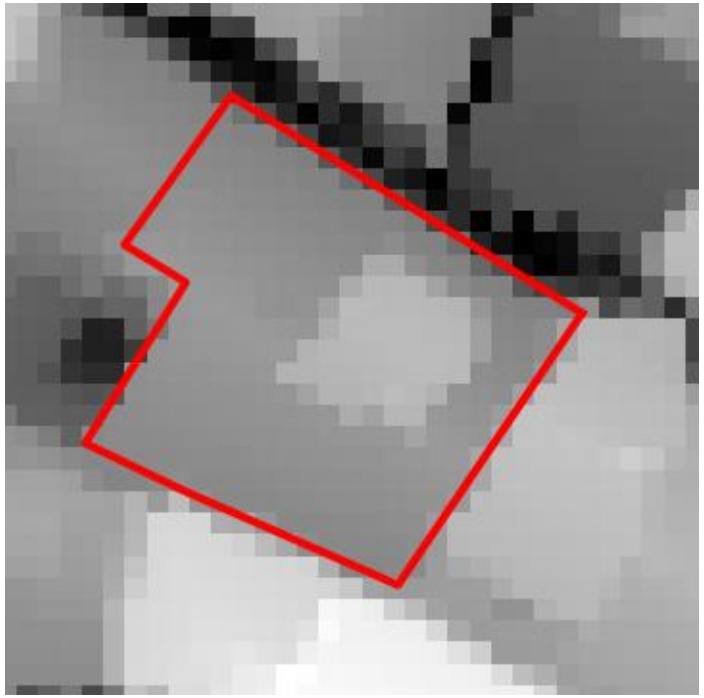


Digital Terrain Model

Digital Surface Model

Rooftop modelling

1. Buildings cadastral maps used to segment the DSM



Rooftop modelling

1. Buildings cadastral maps used to segment the DSM
2. **Region-Growing** algorithm to cluster different building elevation

Rooftop modelling

1. Buildings cadastral maps used to segment the DSM
2. **Region-Growing** algorithm to cluster different building elevation

$$\begin{bmatrix} 3 & 6 & 4 & 9 \\ 2 & 6 & 5 & 7 \\ 3 & 5 & 4 & 7 \\ 2 & 3 & 3 & 8 \\ 1 & 6 & 8 & 9 \\ 9 & 8 & 9 & 9 \end{bmatrix}$$

UNIVERSITÀ DEGLI STUDI FIRENZE DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB DISTRIBUTED DATA INTELLIGENCE AND TECHNOLOGIES LAB SNAP4CITY

Rooftop modelling

1. Buildings cadastral maps used to segment the DSM
2. **Region-Growing** algorithm to cluster different building elevation

$$\begin{bmatrix} 3 & 6 & 4 & 9 \\ 2 & 6 & 5 & 7 \\ 3 & 5 & 4 & 7 \\ 2 & 3 & 3 & 8 \\ 1 & 6 & 8 & 9 \\ 9 & 8 & 9 & 9 \end{bmatrix} \Rightarrow \begin{bmatrix} 3 & 6 & 4 & 9 \\ 2 & 6 & 5 & 7 \\ 3 & 5 & 4 & 7 \\ 2 & \color{red}{3} & 3 & 8 \\ 1 & 6 & 8 & 9 \\ 9 & 8 & 9 & 9 \end{bmatrix}$$

Rooftop modelling

1. Buildings cadastral maps used to segment the DSM
2. **Region-Growing** algorithm to cluster different building elevation

$$\begin{bmatrix} 3 & 6 & 4 & 9 \\ 2 & 6 & 5 & 7 \\ 3 & 5 & 4 & 7 \\ 2 & 3 & 3 & 8 \\ 1 & 6 & 8 & 9 \\ 9 & 8 & 9 & 9 \end{bmatrix} \Rightarrow \begin{bmatrix} 3 & 6 & 4 & 9 \\ 2 & 6 & 5 & 7 \\ 3 & 5 & 4 & 7 \\ 2 & \color{red}{3} & 3 & 8 \\ 1 & 6 & 8 & 9 \\ 9 & 8 & 9 & 9 \end{bmatrix} \Rightarrow \begin{bmatrix} 3 & 6 & 4 & 9 \\ 2 & 6 & 5 & 7 \\ \color{blue}{3} & \color{blue}{5} & \color{blue}{4} & 7 \\ \color{blue}{2} & \color{red}{3} & \color{blue}{3} & 8 \\ \color{blue}{1} & \color{blue}{6} & \color{blue}{8} & 9 \\ 9 & 8 & 9 & 9 \end{bmatrix}$$

Rooftop modelling

1. Buildings cadastral maps used to segment the DSM
2. **Region-Growing** algorithm to cluster different building elevation

$$\begin{bmatrix} 3 & 6 & 4 & 9 \\ 2 & 6 & 5 & 7 \\ 3 & 5 & 4 & 7 \\ 2 & 3 & 3 & 8 \\ 1 & 6 & 8 & 9 \\ 9 & 8 & 9 & 9 \end{bmatrix} \Rightarrow \begin{bmatrix} 3 & 6 & 4 & 9 \\ 2 & 6 & 5 & 7 \\ 3 & 5 & 4 & 7 \\ 2 & {\color{red}3} & 3 & 8 \\ 1 & 6 & 8 & 9 \\ 9 & 8 & 9 & 9 \end{bmatrix} \Rightarrow \begin{bmatrix} 3 & 6 & 4 & 9 \\ 2 & 6 & 5 & 7 \\ {\color{blue}3} & {\color{blue}5} & {\color{blue}4} & 7 \\ {\color{blue}2} & {\color{red}3} & {\color{blue}3} & 8 \\ {\color{blue}1} & {\color{blue}6} & {\color{blue}8} & 9 \\ 9 & 8 & 9 & 9 \end{bmatrix} \Rightarrow \begin{bmatrix} 3 & 6 & 4 & 9 \\ 2 & 6 & 5 & 7 \\ {\color{orange}3} & {\color{blue}5} & {\color{orange}4} & 7 \\ {\color{orange}2} & {\color{red}3} & {\color{orange}3} & 8 \\ {\color{blue}1} & {\color{blue}6} & {\color{blue}8} & 9 \\ 9 & 8 & 9 & 9 \end{bmatrix}$$

Rooftop modelling

1. Buildings cadastral maps used to segment the DSM
2. **Region-Growing** algorithm to cluster different building elevation

$$\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow$$

$$\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}$$

Rooftop modelling

1. Buildings cadastral maps used to segment the DSM
2. **Region-Growing** algorithm to cluster different building elevation

$$\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow$$

$$\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\ldots\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}$$

Rooftop modelling

1. Buildings cadastral maps used to segment the DSM
2. **Region-Growing** algorithm to cluster different building elevation

$$\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow$$

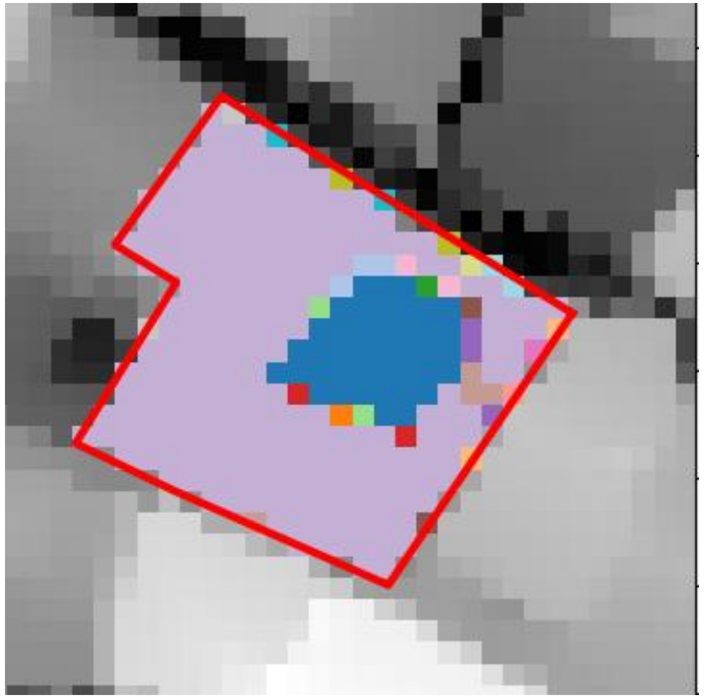
$$\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\dots\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}$$

Rooftop modelling

1. Buildings cadastral maps used to segment the DSM
2. **Region-Growing** algorithm to cluster different building elevation

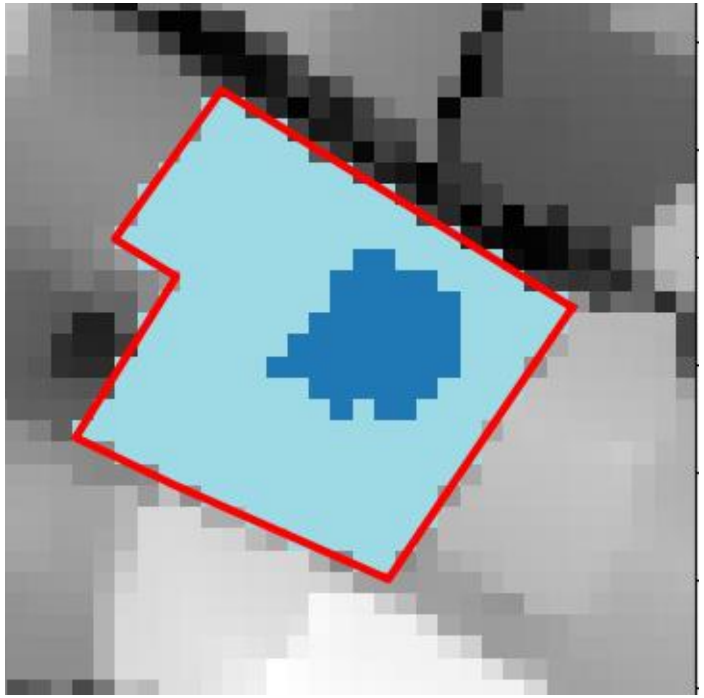
$$\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow$$

$$\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}\Rightarrow\ \dots\ \Rightarrow\begin{bmatrix}3&6&4&9\\2&6&5&7\\3&5&4&7\\2&3&3&8\\1&6&8&9\\9&8&9&9\end{bmatrix}$$



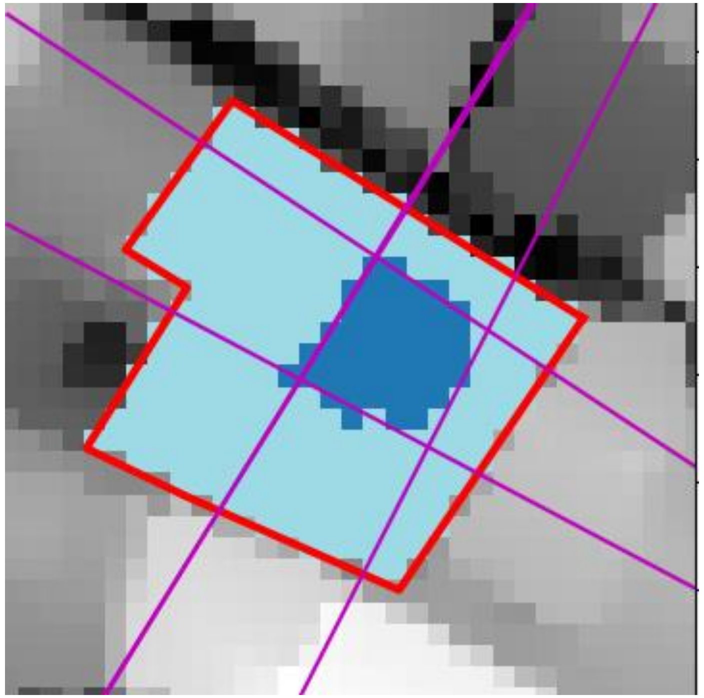
Rooftop modelling

1. Buildings cadastral maps used to segment the DSM
2. Region-Growing algorithm to cluster different building elevation
3. Small cluster aggregation



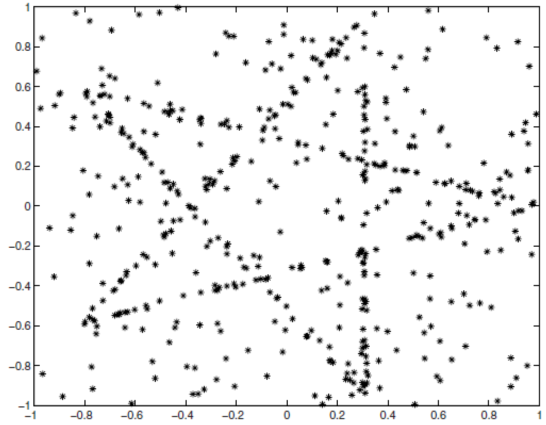
Rooftop modelling

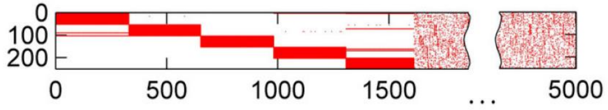
1. Buildings cadastral maps used to segment the DSM
2. Region-Growing algorithm to cluster different building elevation
3. Small cluster aggregation
4. Step-line regression

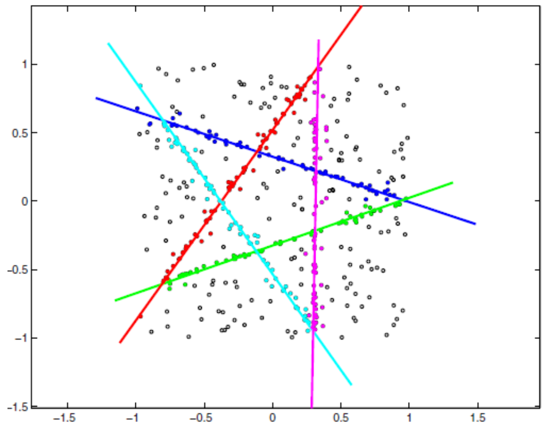


J-Linkage

- To estimate multiple lines from set of points including noise, we used the J-linkage algorithm
- J-Linkage is based on the robust estimation of geometric primitives:
  - Given a minimal set of points (i.e., 2 for lines) the primitive is estimated
  - Then, the estimated primitive is scored w.r.t. all the other data points
  - Repeating the process several times a consensus matrix is computed, where for each point is reported its consensus respect to each estimated primitive
  - By clustering the consensus matrix, the predominant primitives can be found



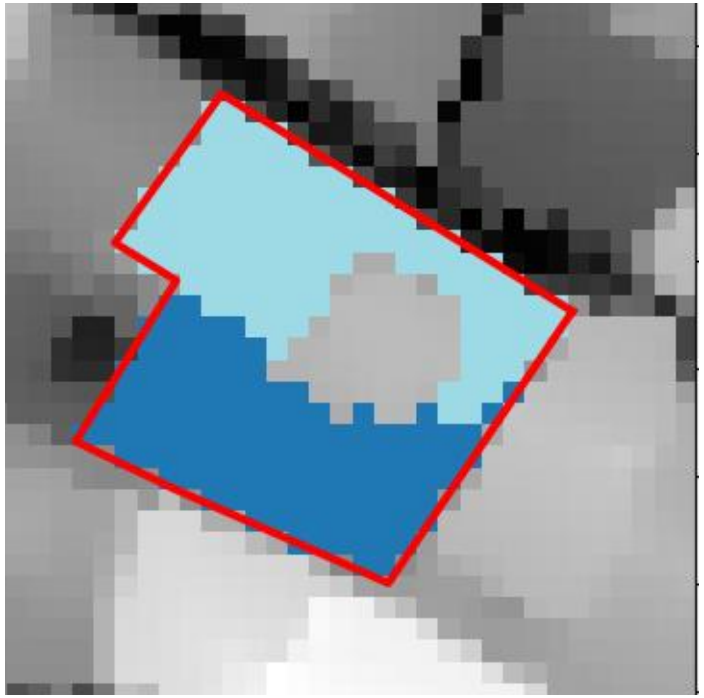




Toldo, R., Fusiello, A. (2008). Robust Multiple Structures Estimation with J-Linkage. ECCV 2008

Rooftop modelling

1. Buildings cadastral maps used to segment the DSM
2. Region-Growing algorithm to cluster different building elevation
3. Small cluster aggregation
4. Step-line regression
5. **HDBSCAN algorithm** based on a custom weight matrix W= $\{w_{ij}\}$



$$w_{ij} = \begin{cases} \infty, & if\ L_{ij} > 6 \\ w_N N_{ij} + w_M M_{ij} + w_D D_{ij} + w_L L_{ij}, & otherwise \end{cases}$$

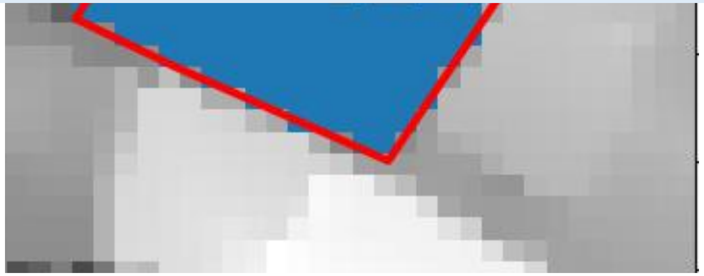
Rooftop modelling

1. Buildings cadastral maps used to segment the DSM
2. Region-Growing algorithm to cluste building elevation
3. Small cluster aggregation
4. Step-line regression
5. **HDBSCAN algorithm** based on a custom weight matrix W= $\{w_{ij}\}$

HDBSCAN is a density-based clustering algorithm

- It can find clusters of arbitrary shape
- It is a hierarchical clustering method
- It favors dense clusters, discarding isolated data points (noise)
- It automatically determines the number of clusters
- It can handle user-specified distance

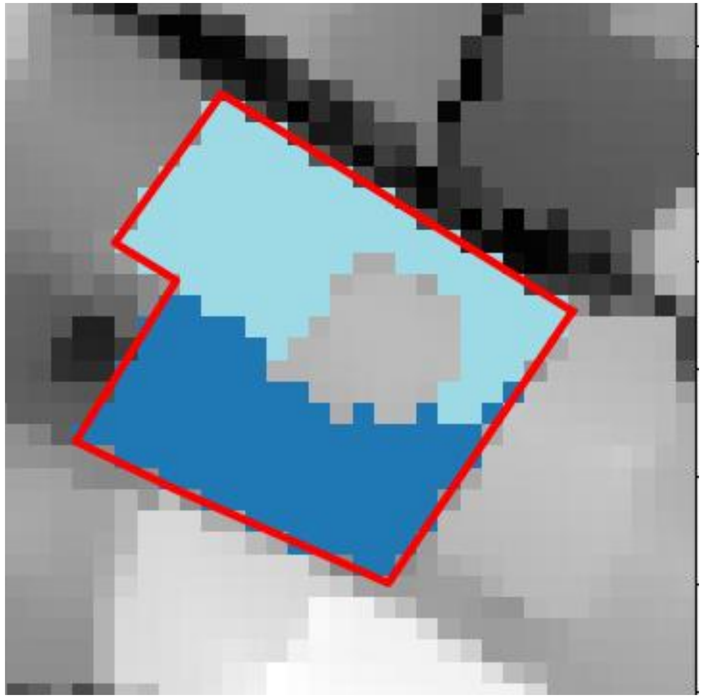
See Campello, et al. Density-Based Clustering Based on Hierarchical Density Estimates. PAKDD 2013



$$w_{ij} = \begin{cases} \infty, & if\ L_{ij} > 6 \\ w_N N_{ij} + w_M M_{ij} + w_D D_{ij} + w_L L_{ij}, & otherwise \end{cases}$$

Rooftop modelling

1. Buildings cadastral maps used to segment the DSM
2. Region-Growing algorithm to cluster different building elevation
3. Small cluster aggregation
4. Step-line regression
5. HDBSCAN algorithm based on a custom weight matrix W= $\{w_{ij}\}$



grad magnitude difference

grad direction difference

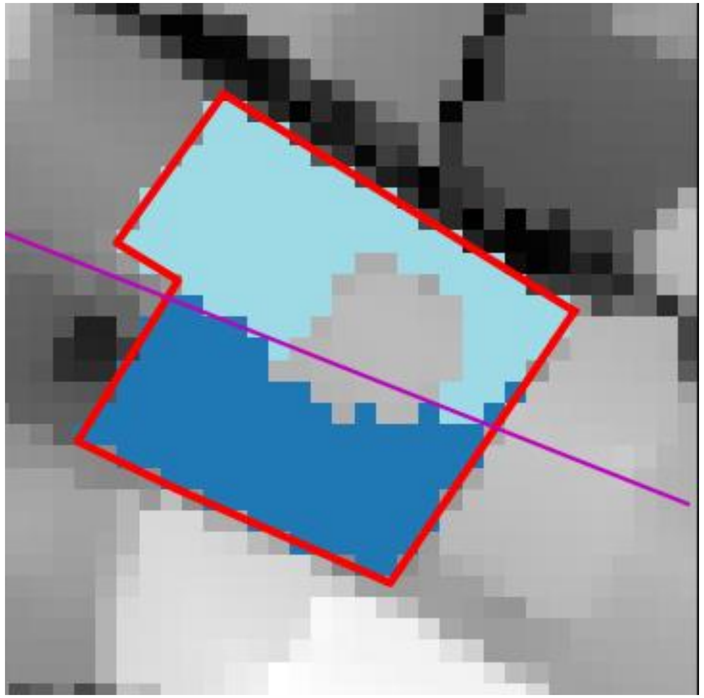
L1 distance

normal angular difference

$$w_{ij} = \begin{cases} \infty, & if\ L_{ij} > 6 \\ w_N N_{ij} + w_M M_{ij} + w_D D_{ij} + w_L L_{ij}, & otherwise \end{cases}$$

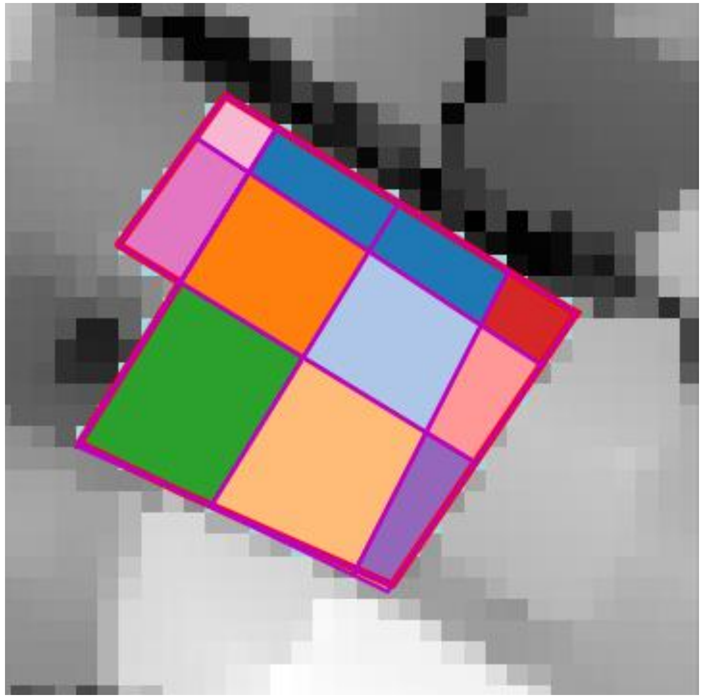
Rooftop modelling

6. Hip-line regression



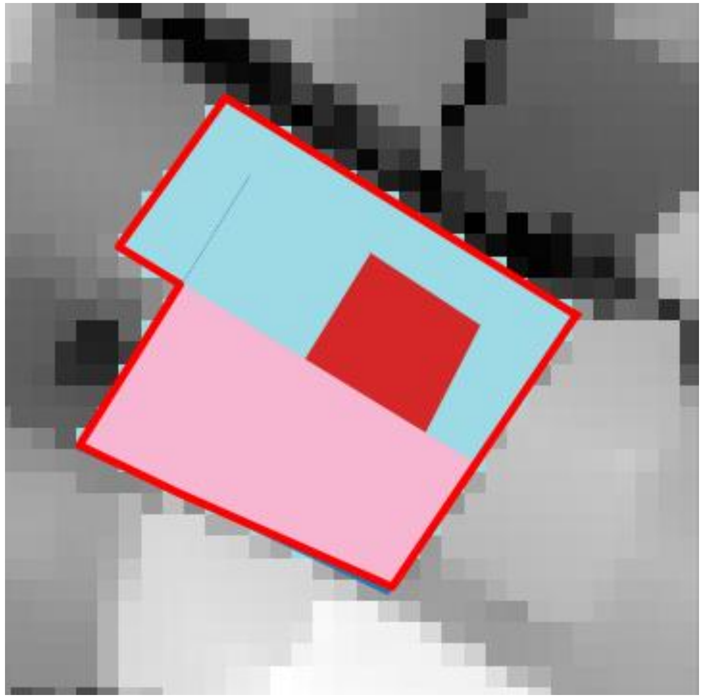
Rooftop modelling

6. Hip-line regression
7. Planar patch definition



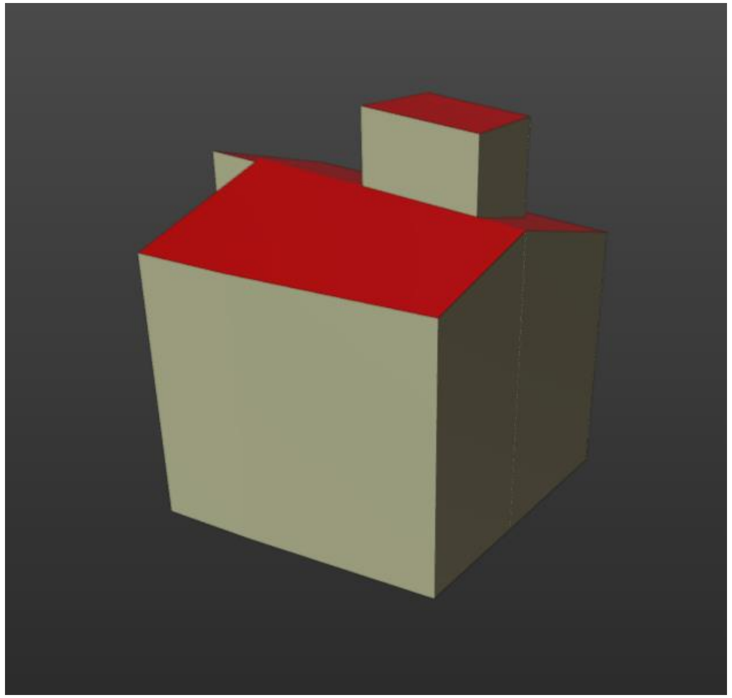
Rooftop modelling - DA RIVEDERE E AMPLIARE

6. Hip-line regression
7. Planar patch definition
8. Planar patch labeling and **grouping**



Rooftop modelling

6. Hip-line regression
7. Planar patch definition
8. Planar patch labeling and grouping
9. Compute the 3D roof planes by **robust regression** for each planar patch



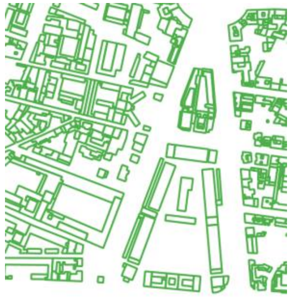
Code is freely available at https://github.com/disit/3d-building-modelling

Rooftop texturing

- Aerial orthophoto are **roughly georeferenced**
- Building shapes are **not aligned** with the orthophotos
- To extract roof textures the alignment between shapes and images **must be corrected**

Rooftop texturing

Orthomaps

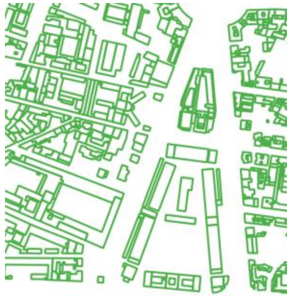


Building shapes

Input

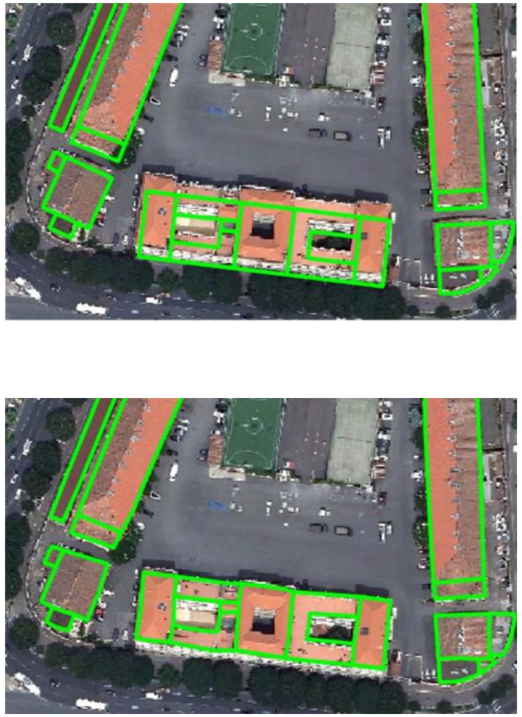
Rooftop texturing

Orthomaps



Building shapes

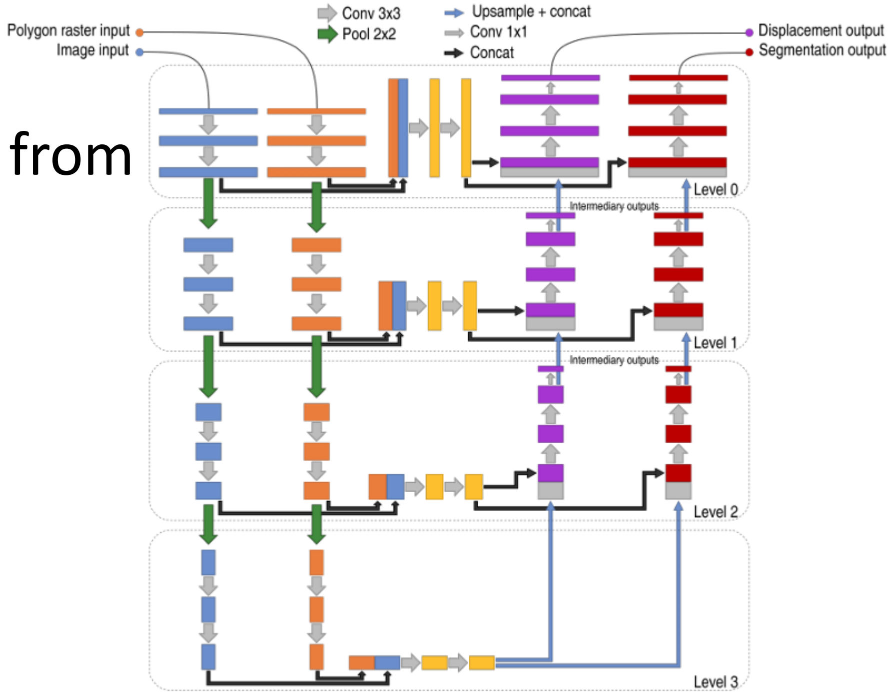
Input



Deep network alignment

A **double U-Net architecture** exploiting multi-resolution and multi-task learning was used to

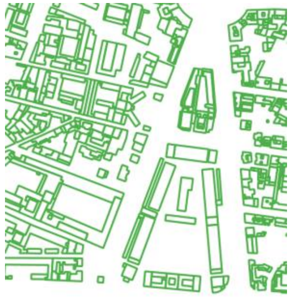
- **detect the rooftops** from the aerial photos
- **align** them with the building shapes



N. Girard et al., *Aligning and Updating Cadaster Maps with Aerial Images by Multi-task, Multi-resolution Deep Learning*, ACCV, 2018.

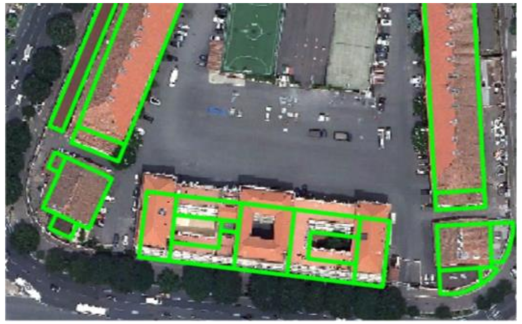
Rooftop texturing

Orthomaps

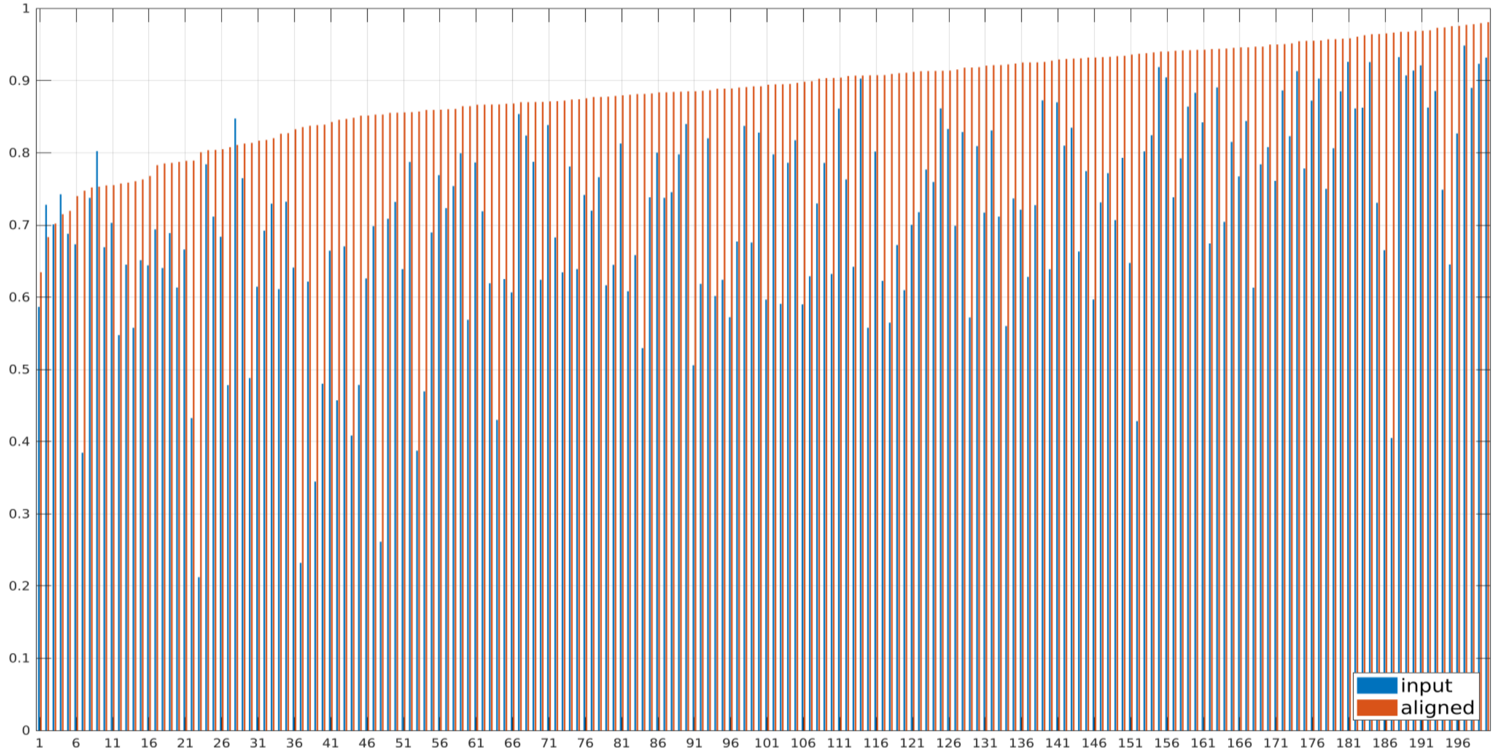


Building shapes

Input

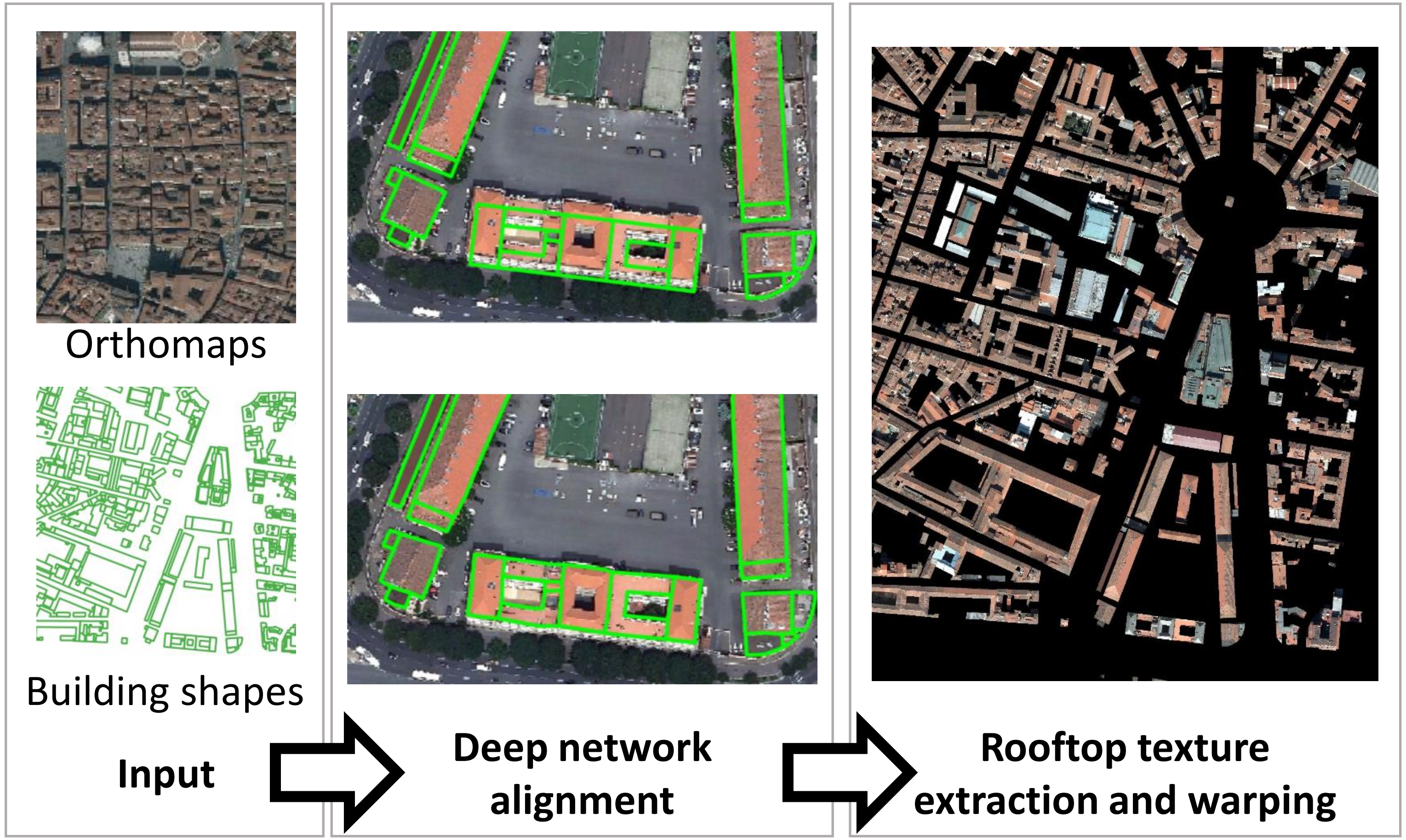


Deep network alignment



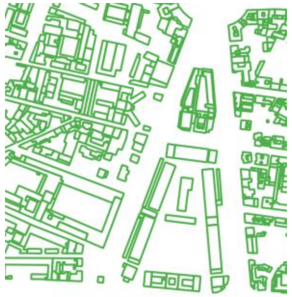
After the alignment, **an increases of almost 15%** of the IoU between the building shapes and the orthomap rooftops was measured.

Rooftop texturing



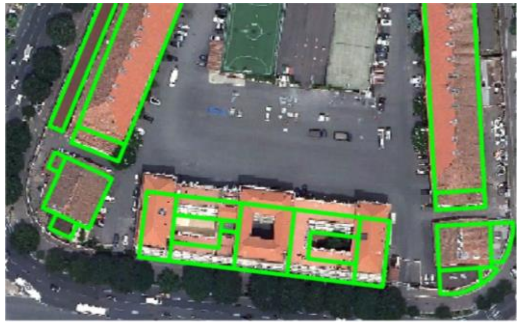
Rooftop texturing

Orthomaps

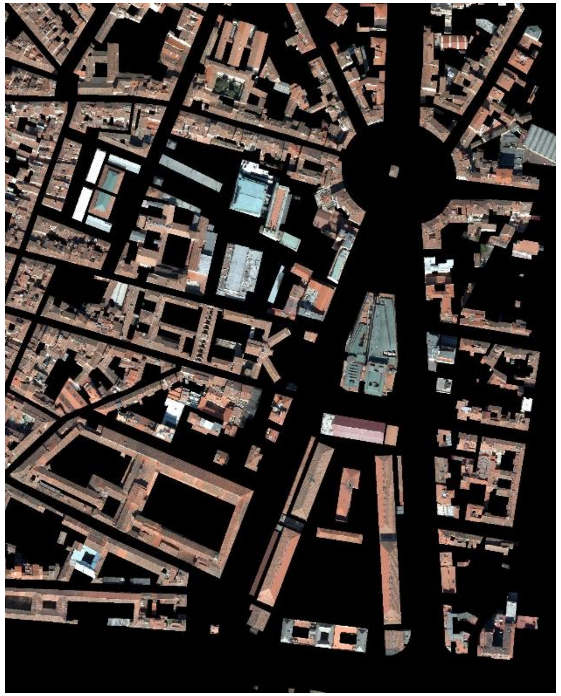


Building shapes

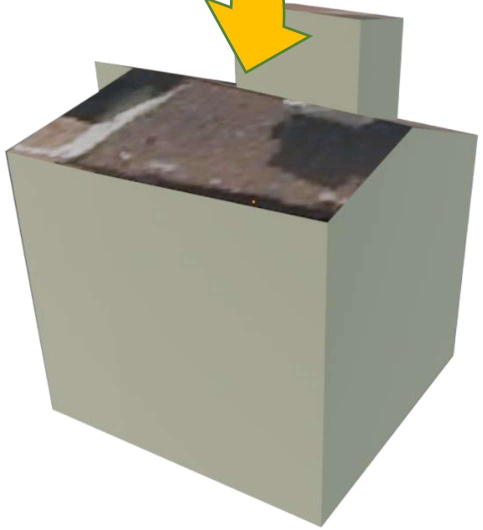
Input



Deep network alignment



Rooftop texture extraction and warping



Apply the texture on the 3D model

Façade texturing

- Façade texturing is more difficult

Façade texturing

- Façade texturing is more difficult
  - It requires specific acquisition campaign



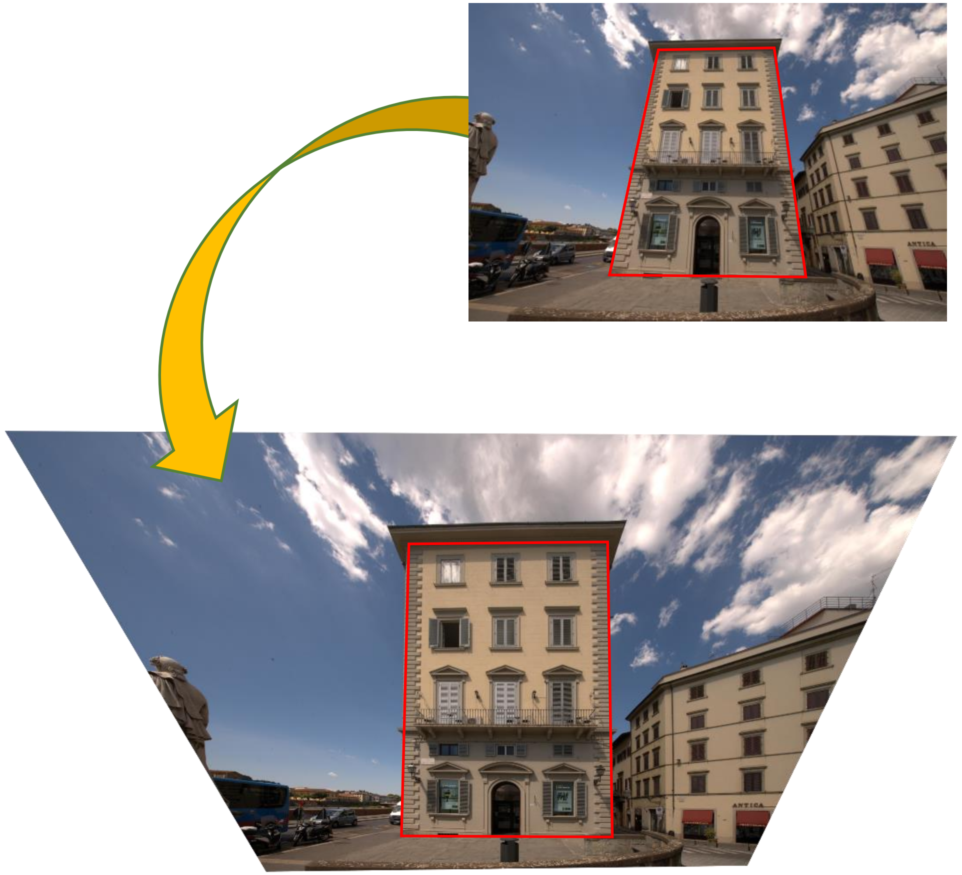


Façade texturing

- Façade texturing is more difficult
  - It requires specific acquisition campaign
  - A single photo can include few buildings simultaneously

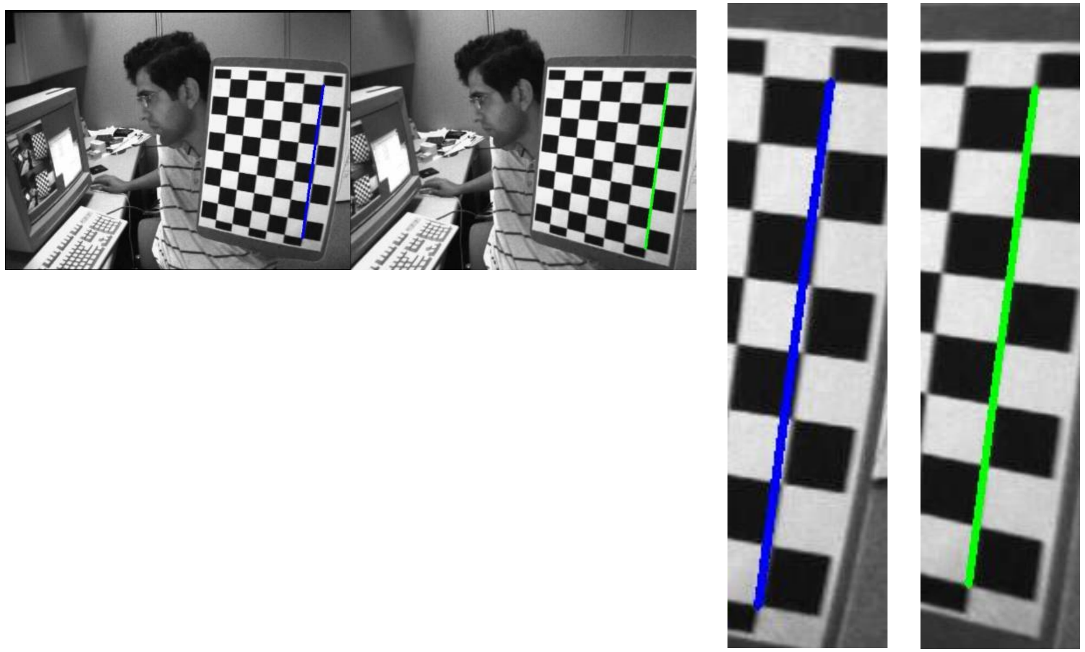
Façade texturing

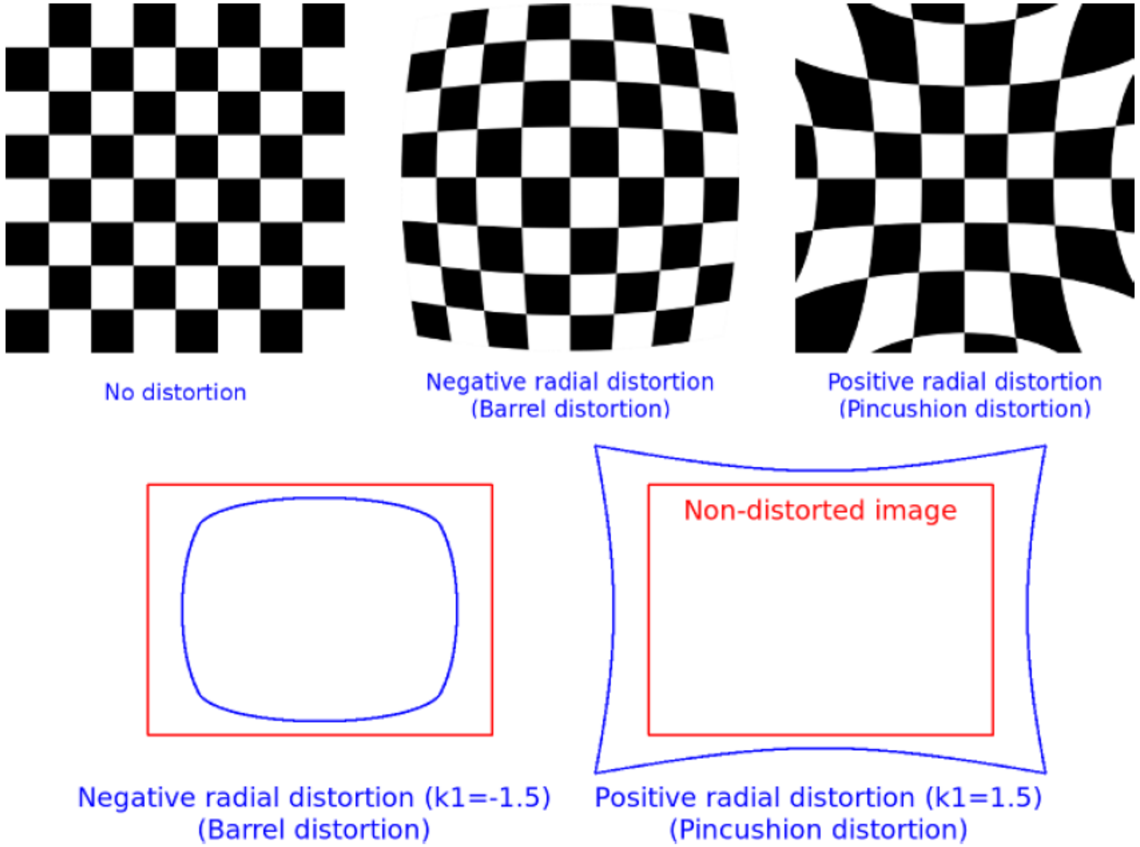
- Façade texturing is more difficult
  - It requires specific acquisition campaign
  - A single photo can include few buildings simultaneously
  - **Radial** and **perspective** distortion must be removed (**HOMOGRAPHY**)



Radial distortion

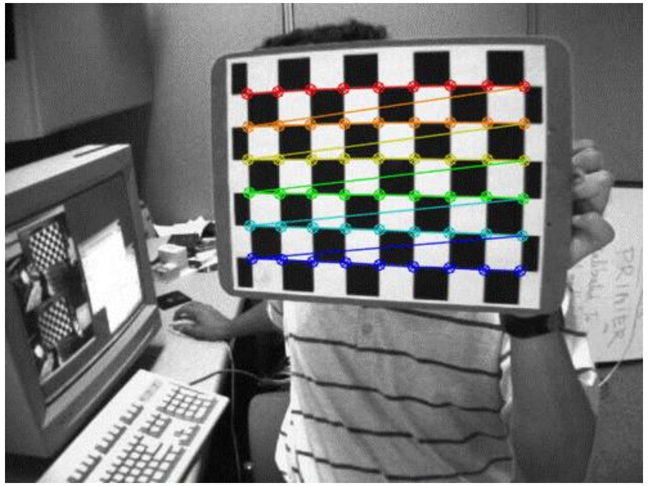
- It is an effect introduced by **camera lenses**
- Straight line becomes **curves**
- It can be recovered together with the camera calibration

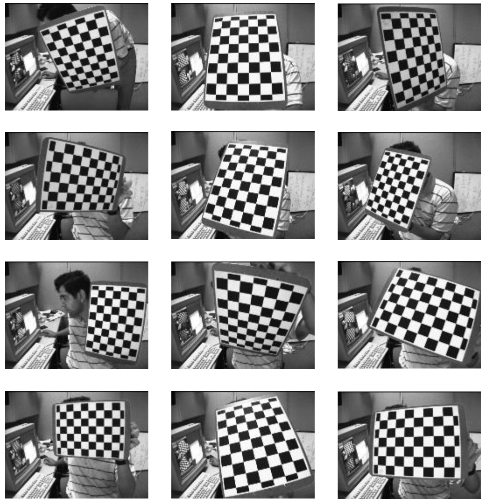


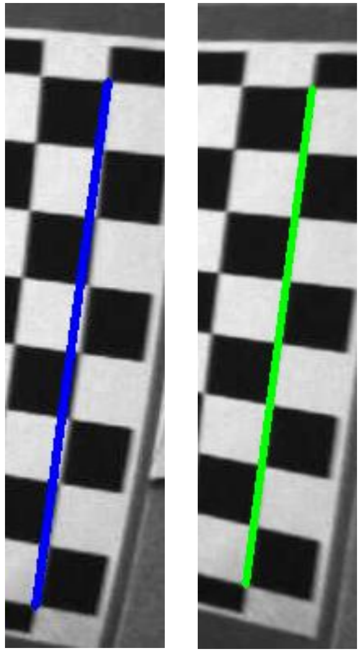


Radial distortion

- If you have the camera, calibration methods can be used to find the distortion parameters automatically using known patterns
- Otherwise, we can manually search image edges that we know should be straight ad use them to estimate the parameters of the distortion

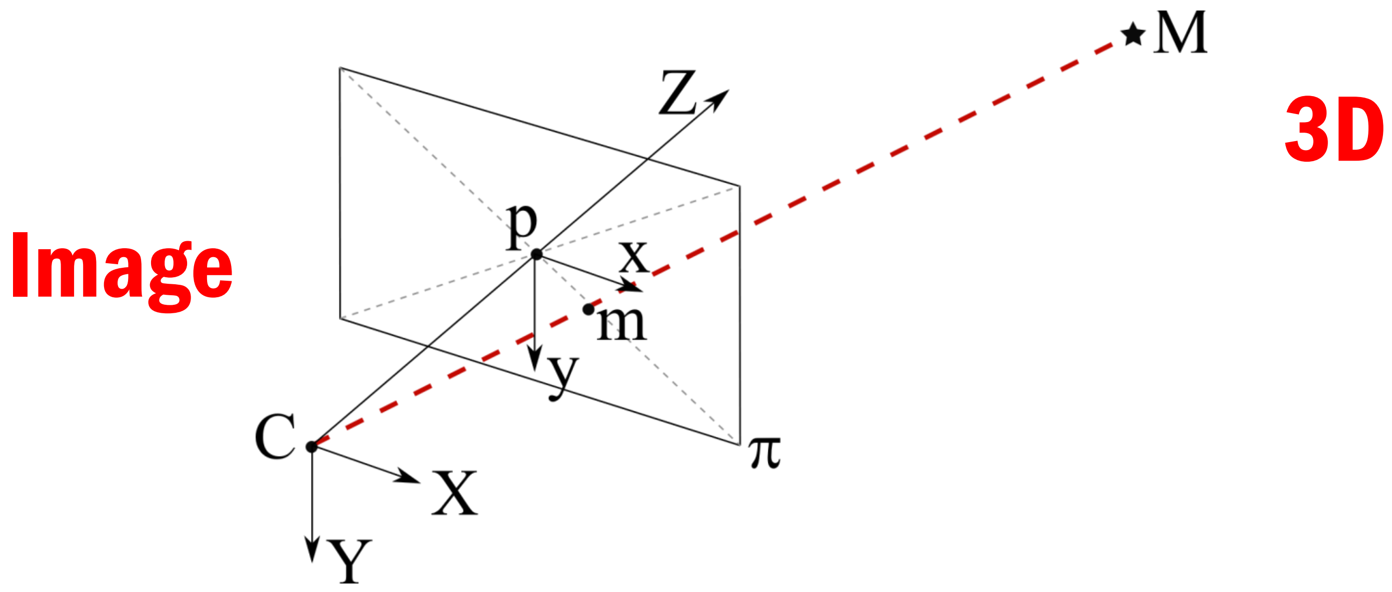






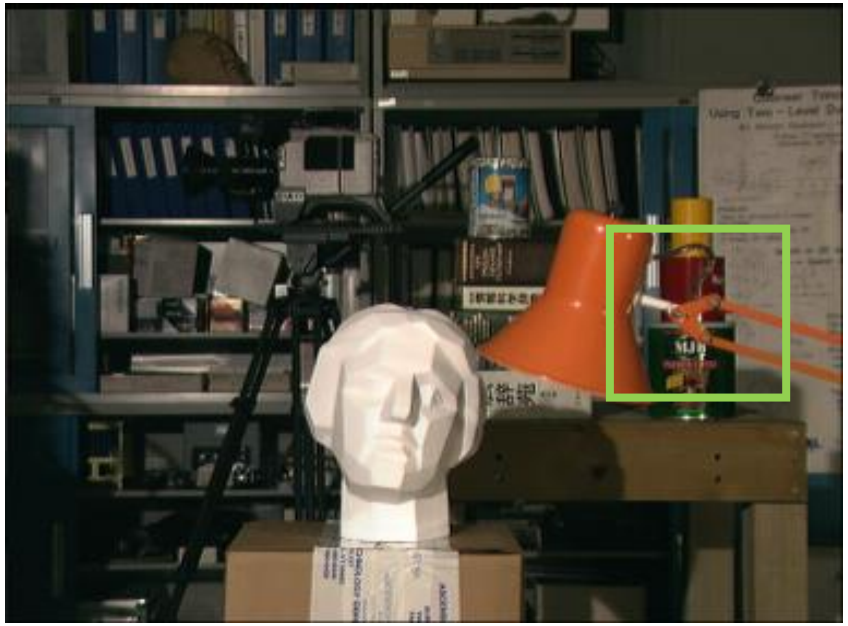
Perspective distortion

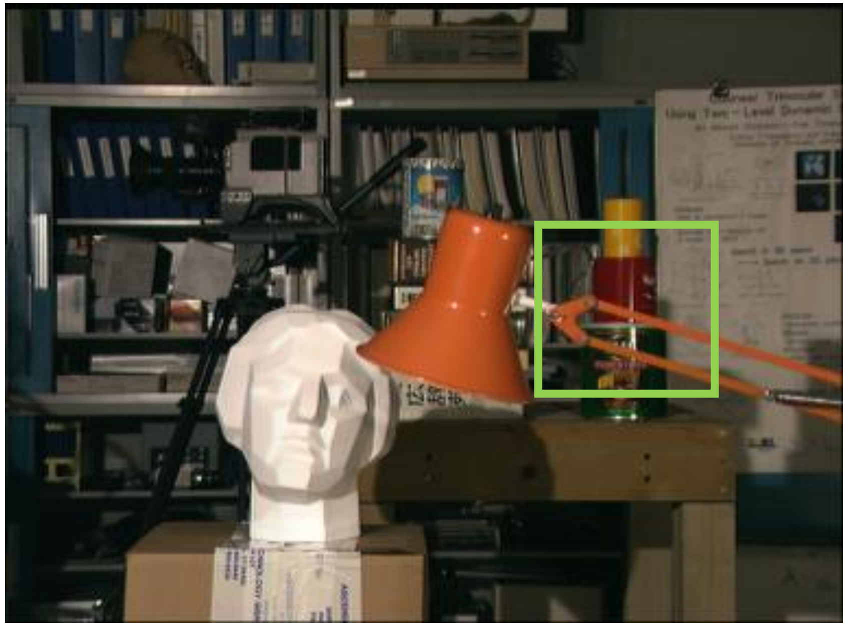
- Perspective distortion is intrinsically due to the camera projection



Perspective distortion

- Perspective distortion is intrinsically due to the camera projection
- In complex 3D scene, the projection introduce the **parallax effect**, where 3D points at different depth can be projected into the same 2D point



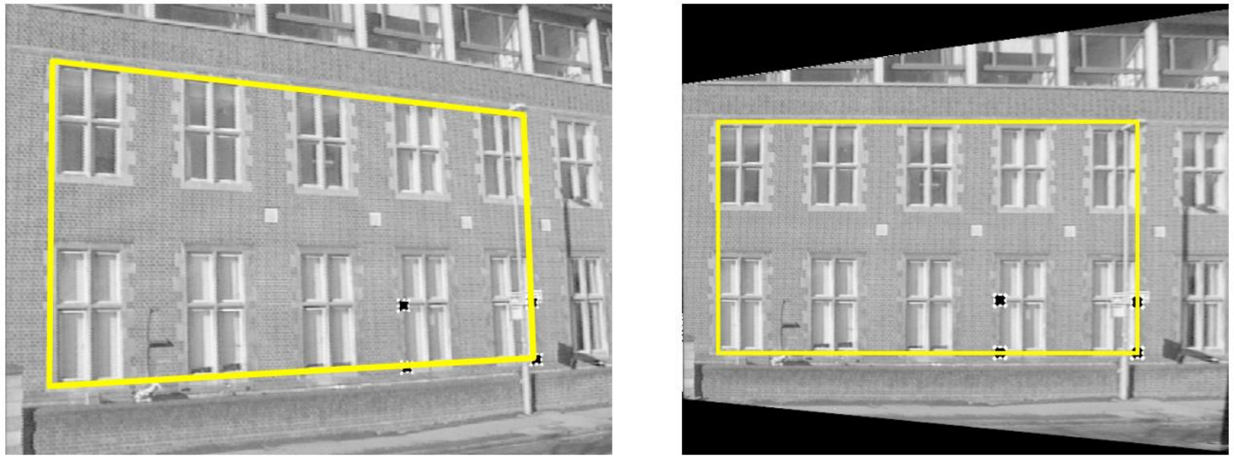


Perspective distortion

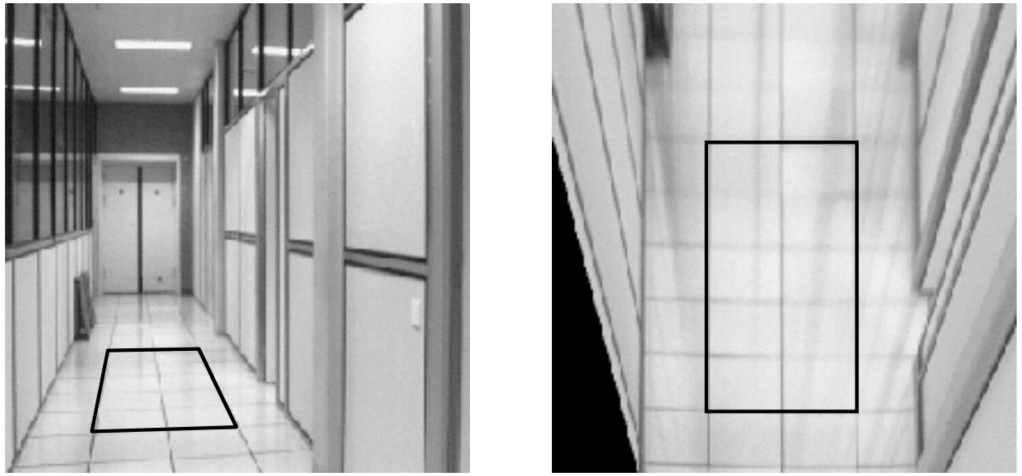
- Perspective distortion is intrinsically due to the camera projection
- In complex 3D scene, the projection introduce the **parallax effect**, where 3D points at different depth can be projected into the same 2D point
- However, when observing **planar scenes,** no parallax is generated

Perspective distortion

- Perspective distortion is intrinsically due to the camera projection
- In complex 3D scene, the projection introduce the **parallax effect**, where 3D points at different depth can be projected into the same 2D point
- However, when observing **planar scenes,** no parallax is generated
- In this case, we can use **planar homographyes** to remove the projective distortion



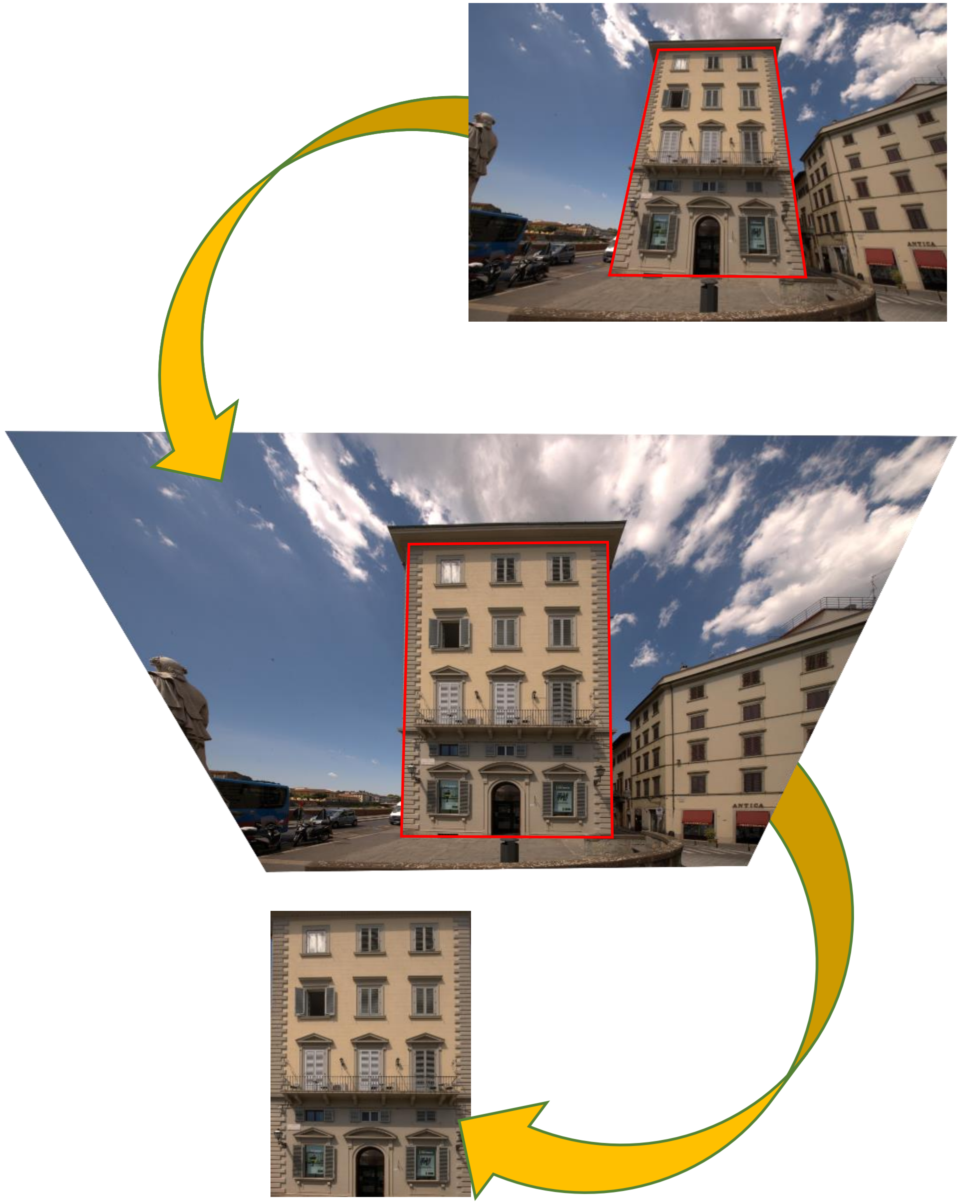
from Hartley & Zisserman



from Hartley & Zisserman

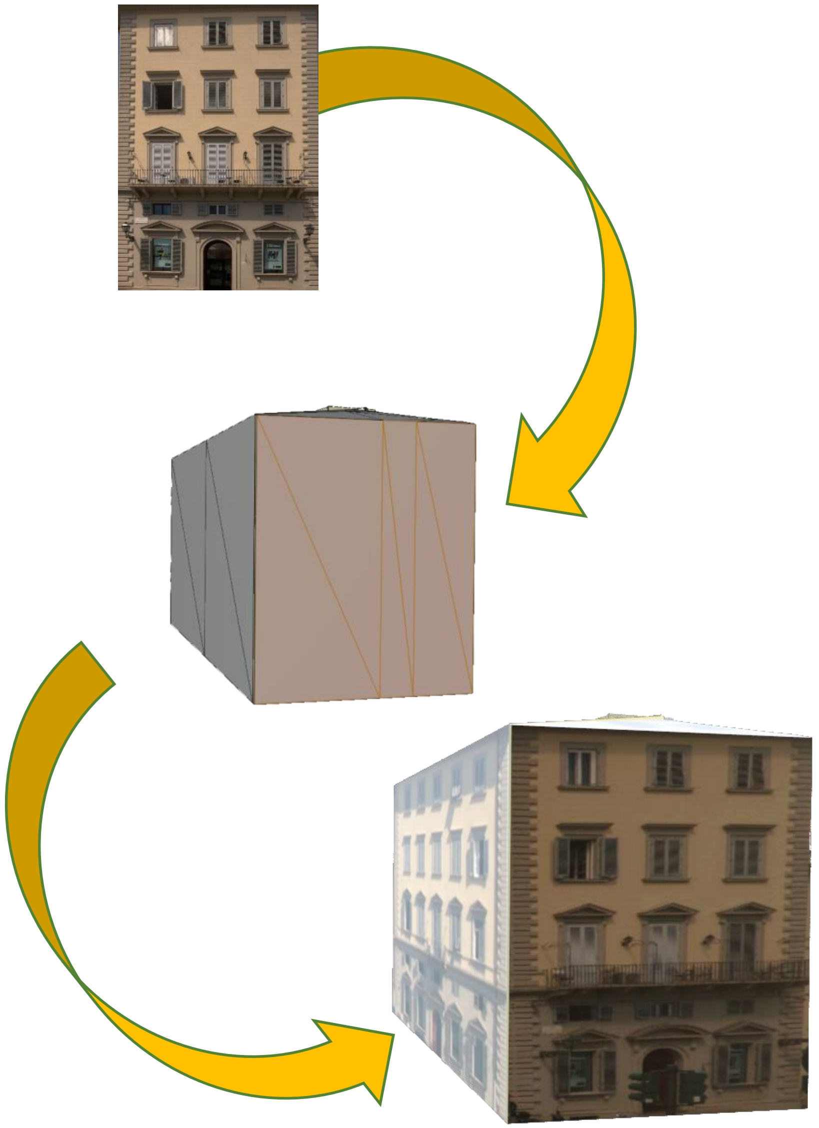
Façade texturing

- Façade texturing is more difficult
  - It requires specific acquisition campaign
  - A single photo can include few buildings simultaneously
  - Perspective and radial distortion must be removed
  - Building façades must be segmented from the image



Façade texturing

- Façade texturing is more difficult
  - It requires specific acquisition campaign
  - A single photo can include few buildings simultaneously
  - Perspective and radial distortion must be removed
  - Building façades must be segmented from the image
  - The correct 3D model face must be selected to apply the texture



Structure from Motion

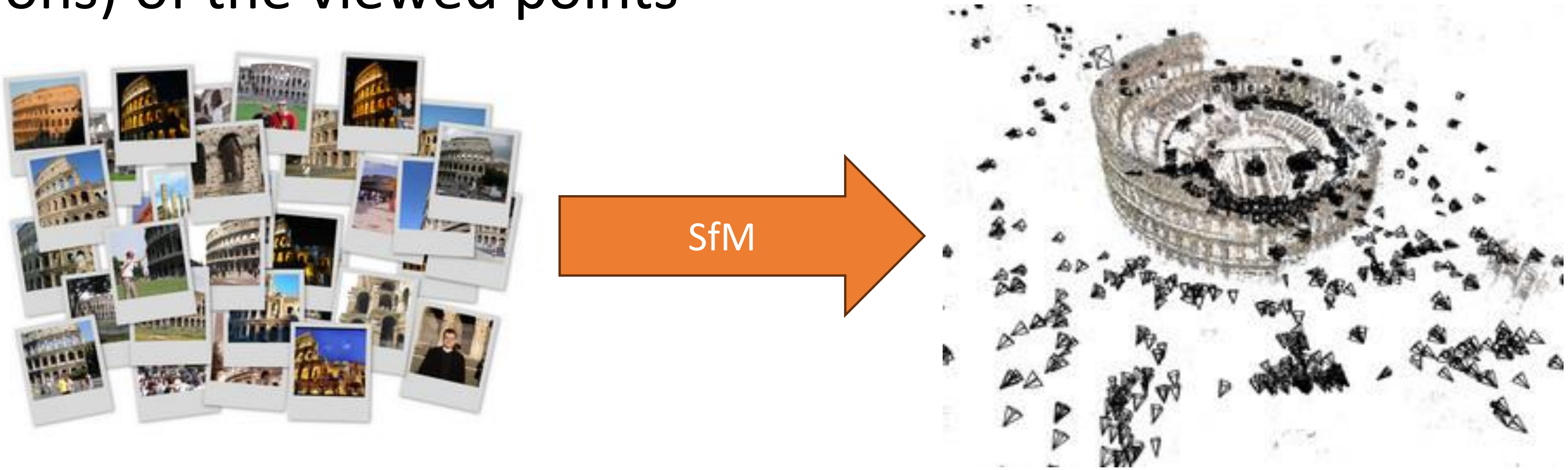
- 3D models can be obtained also using **Structure from Motion** approaches

Structure from Motion

- 3D models can be obtained also using **Structure from Motion** approaches
- If we have a **large collection** of photos and videos of a scene observed from multiple point of views

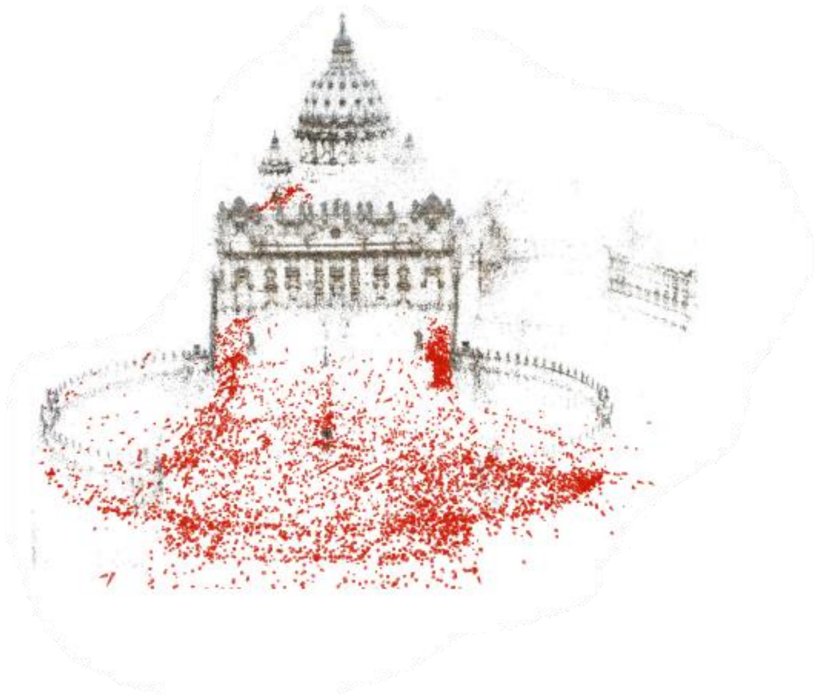
Structure from Motion

- 3D models can be obtained also using **Structure from Motion** approaches
- If we have a **large collection** of photos and videos of a scene observed from multiple point of views
- We can exploit the parallax effect to **recover the depth** (and the 3D positions) of the viewed points



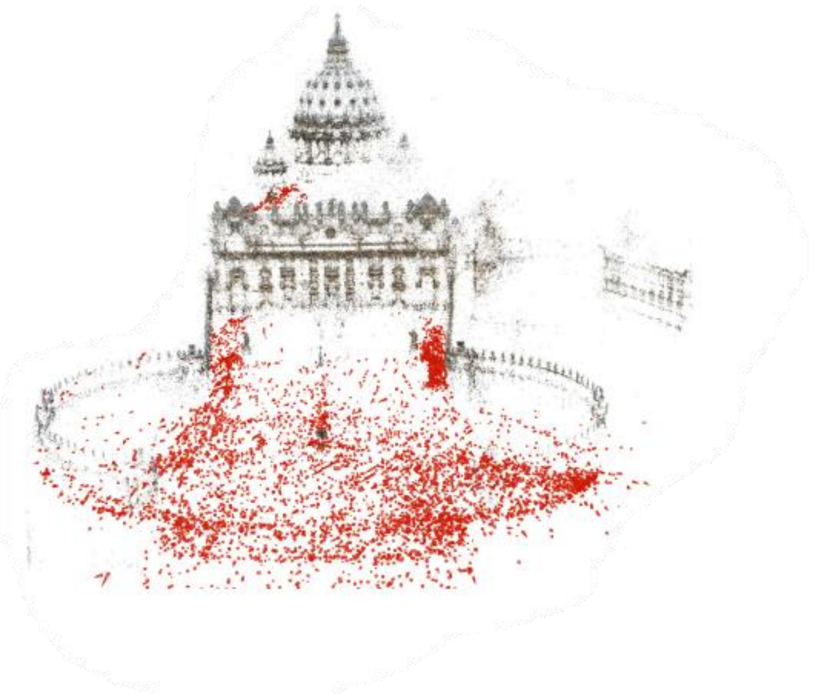
Structure from Motion

- The first result obtained is a **sparse 3D point cloud**



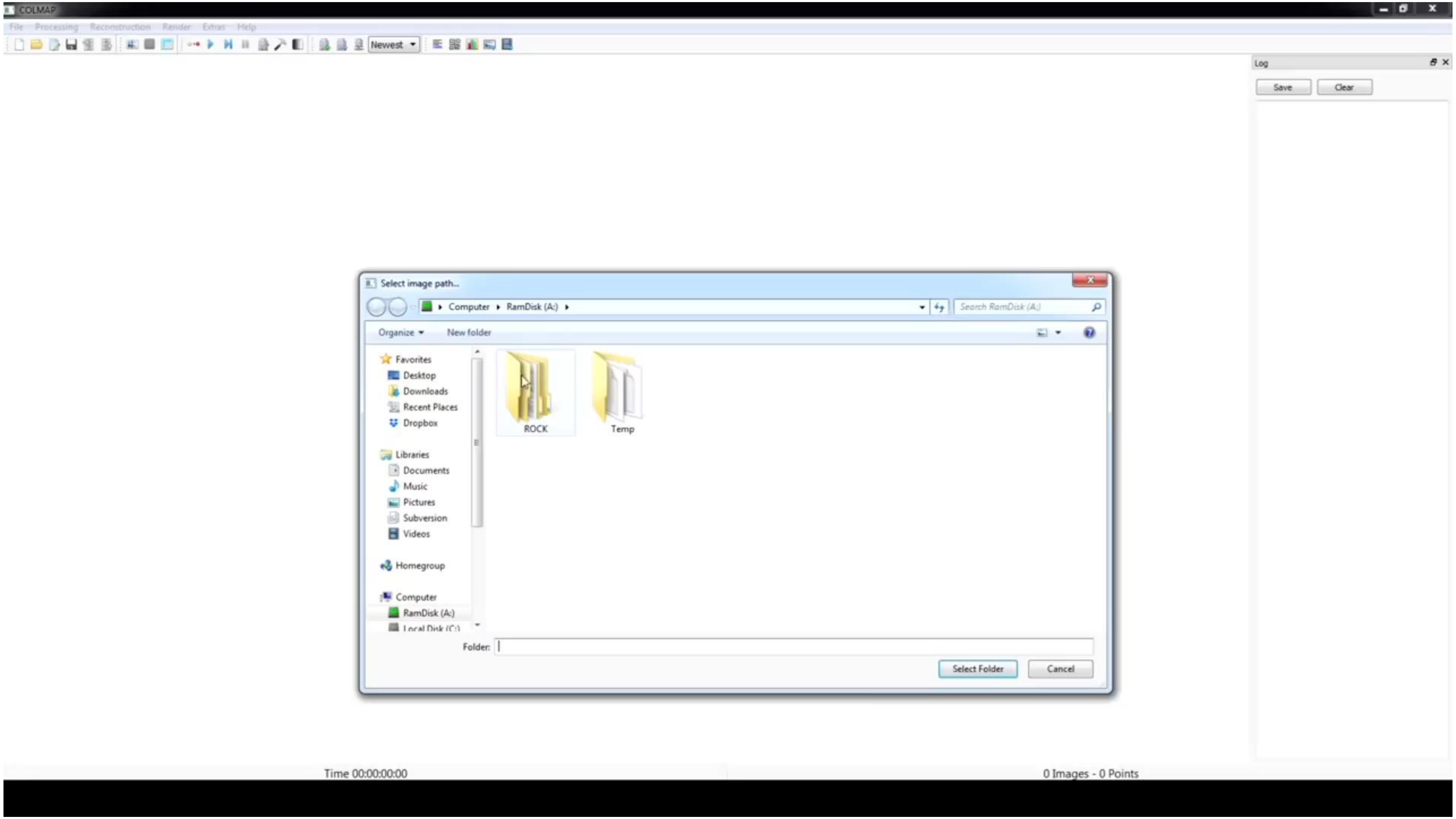
Structure from Motion

- The first result obtained is a **sparse 3D point cloud**
- **Multi-view stereo** methods can then be used to densify the reconstruction



Structure from Motion

- Today there are several off-the-shelf solutions that can be easily used to obtain a **local 3D reconstruction**
- The **OpenCV library** (C/C++, Python) includes functions to build a SfM pipeline
- There are also **software** ready to use, that do not require particular knowledge:
  - PhotoTurism: http://phototour.cs.washington.edu/
  - VisualSfM: http://ccwu.me/vsfm/
  - Colmap: https://colmap.github.io/
  - AliceVision: https://alicevision.org/
- Such software are also able to compute the **mesh** and apply **textures**



https://www.youtube.com/watch?v=bDHJM6nAKtc

Structure from Motion

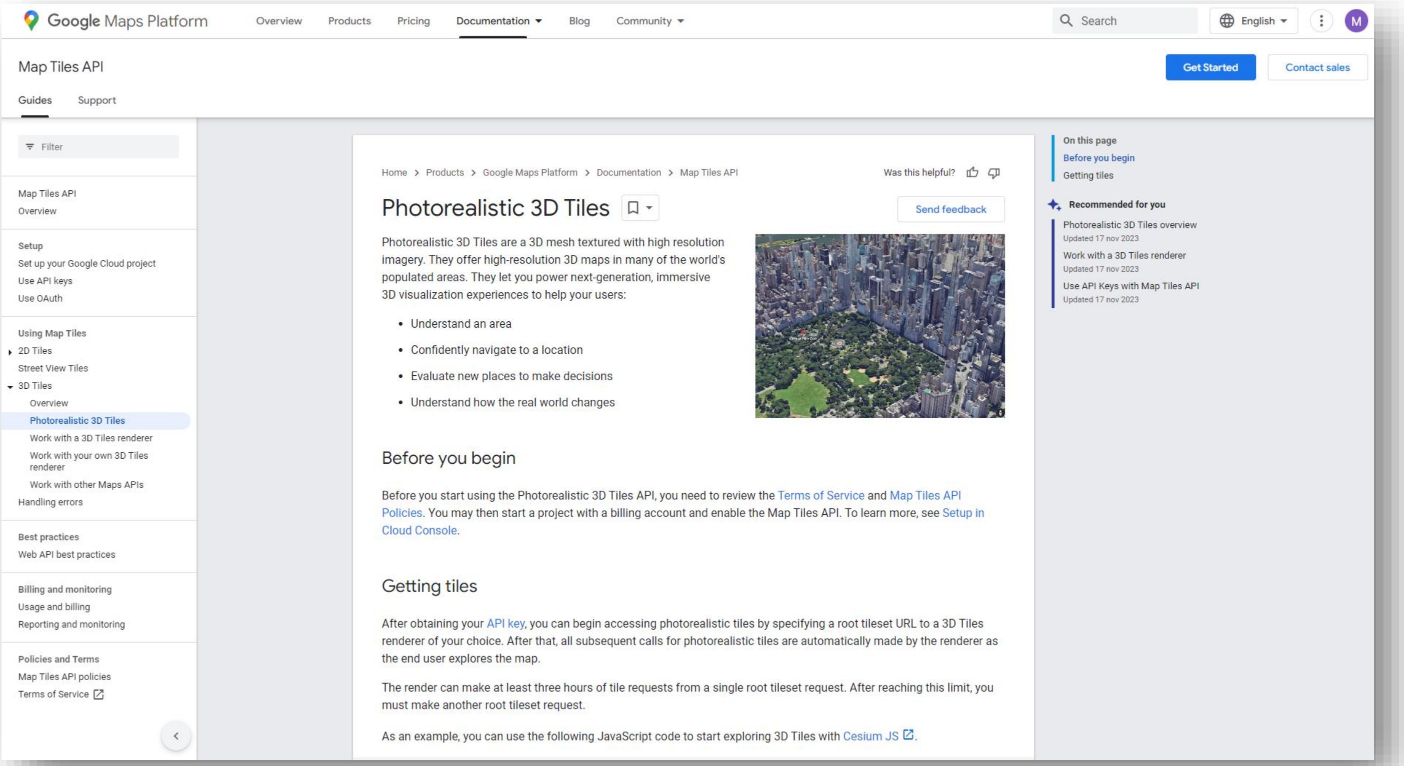
- When dealing with large reconstructions (e.g., city wide) there are however additional difficulties that have to be taken into account
  - The acquisition campaign must cover the whole urban area, both from **street-level** and with **aerial acquisitions**
  - The amount of acquired **data is huge** and requires particular care
  - A single monolithic reconstruction is practically impossible, therefore **multiple localized reconstructions** must be obtained first and then **fused** into a unique coordinate reference frame
  - Moreover, the obtained 3D models must be **geo-localized** in order to be exploited in the context of SCDT

Structure from Motion

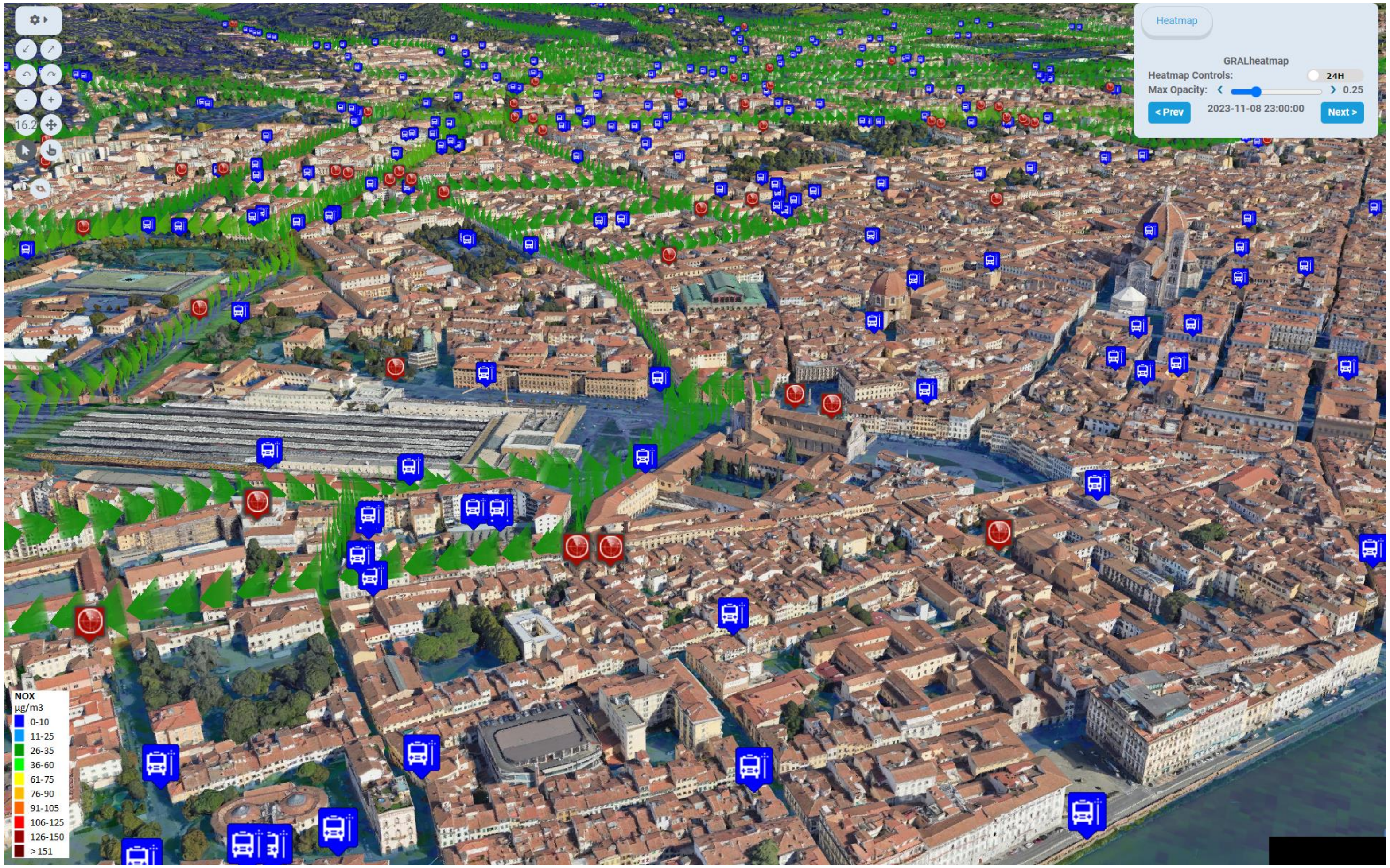
- When dealing with large reconstructions (e.g., city wide) there are however additional difficulties that have to be taken into account
  - The acquisition campaign must cover the whole urban area, both from **street-level** and with **aerial acquisitions**
  - The amount of acquired **data is huge** and requires particular care
  - A single monolithic reconstruction is practically impossible, therefore **multiple localized reconstructions** must be obtained first and then **fused** into a unique coordinate reference frame
  - Moreover, the obtained 3D models must be **geo-localized** in order to be exploited in the context of SCDT
- The conclusion is: exploit Structure from Motion on large scale to obtain a city reconstruction is **extremely expensive**

Google 3D Tiles

- Recently, Google released photorealistic 3D tiles
- REST API can be used to query the Google servers and retrieve 3D tiles of good quality to be shown in your web application



Google 3D Tiles



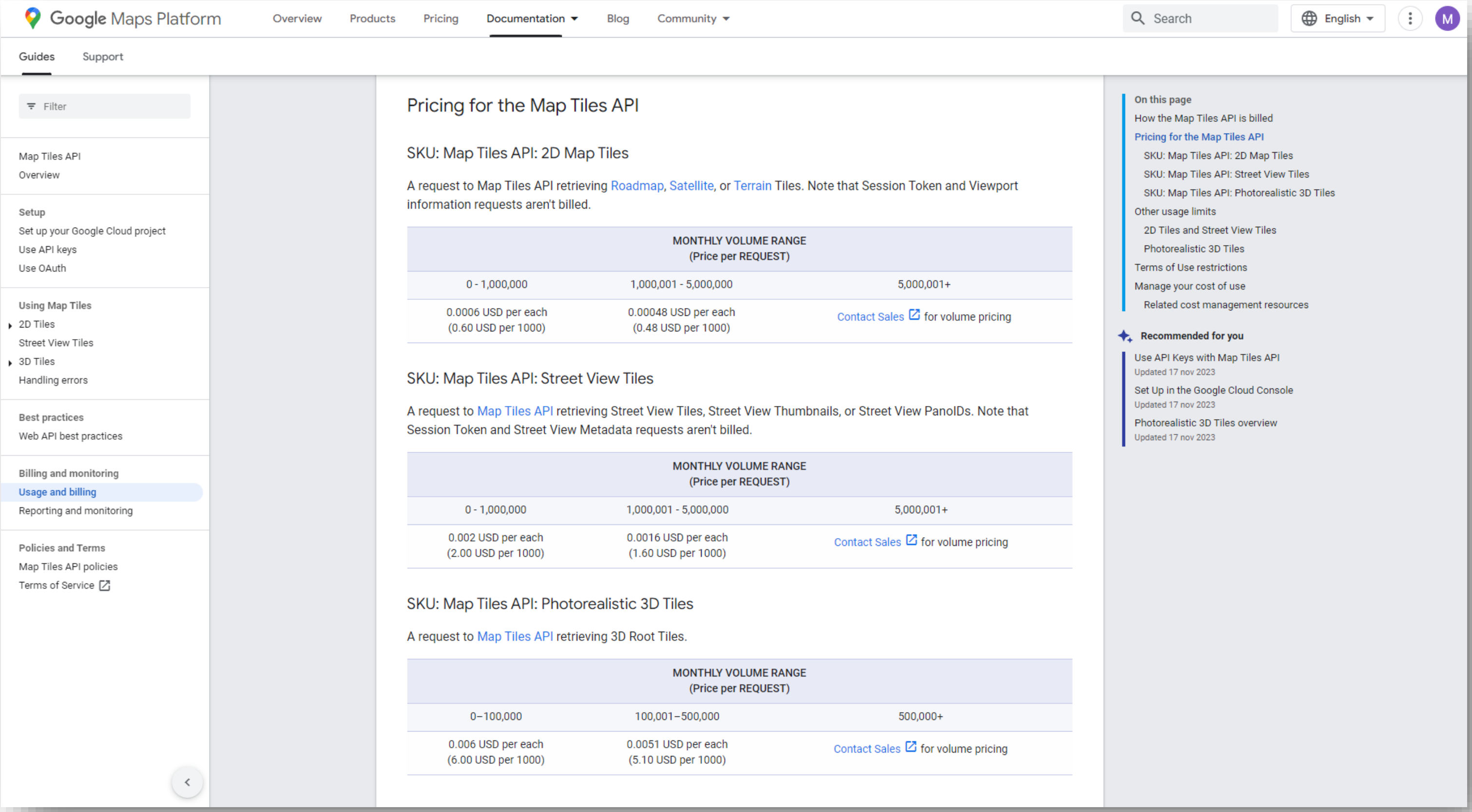
Florence, Italy

Google 3D Tiles

Helsinki, Finland

Google 3D Tiles

- The service is **not free**
- A per-request cost is applied
- Careful consideration must be taken when designing your web application


Google Maps Platform | Overview | Products | Pricing | Documentation | Blog | Community | Search | English

Guides | Support

Filter

Map Tiles API
Overview

Setup
Set up your Google Cloud project
Use API keys
Use OAuth

Using Map Tiles
2D Tiles
Street View Tiles
3D Tiles
Handling errors

Best practices
Web API best practices

Billing and monitoring
Usage and billing
Reporting and monitoring

Policies and Terms
Map Tiles API policies
Terms of Service

Pricing for the Map Tiles API

SKU: Map Tiles API: 2D Map Tiles

A request to Map Tiles API retrieving Roadmap, Satellite, or Terrain Tiles. Note that Session Token and Viewport information requests aren't billed.

MONTHLY VOLUME RANGE (Price per REQUEST)		
0 - 1,000,000	1,000,001 - 5,000,000	5,000,001+
0.0006 USD per each (0.60 USD per 1000)	0.00048 USD per each (0.48 USD per 1000)	Contact Sales for volume pricing

SKU: Map Tiles API: Street View Tiles

A request to Map Tiles API retrieving Street View Tiles, Street View Thumbnails, or Street View PanoIDs. Note that Session Token and Street View Metadata requests aren't billed.

MONTHLY VOLUME RANGE (Price per REQUEST)		
0 - 1,000,000	1,000,001 - 5,000,000	5,000,001+
0.002 USD per each (2.00 USD per 1000)	0.0016 USD per each (1.60 USD per 1000)	Contact Sales for volume pricing

SKU: Map Tiles API: Photorealistic 3D Tiles

A request to Map Tiles API retrieving 3D Root Tiles.

MONTHLY VOLUME RANGE (Price per REQUEST)		
0–100,000	100,001–500,000	500,000+
0.006 USD per each (6.00 USD per 1000)	0.0051 USD per each (5.10 USD per 1000)	Contact Sales for volume pricing

On this page
How the Map Tiles API is billed
Pricing for the Map Tiles API
SKU: Map Tiles API: 2D Map Tiles
SKU: Map Tiles API: Street View Tiles
SKU: Map Tiles API: Photorealistic 3D Tiles
Other usage limits
2D Tiles and Street View Tiles
Photorealistic 3D Tiles
Terms of Use restrictions
Manage your cost of use
Related cost management resources

Recommended for you
Use API Keys with Map Tiles API
Updated 17 nov 2023
Set Up in the Google Cloud Console
Updated 17 nov 2023
Photorealistic 3D Tiles overview
Updated 17 nov 2023

Additional 3D Elements

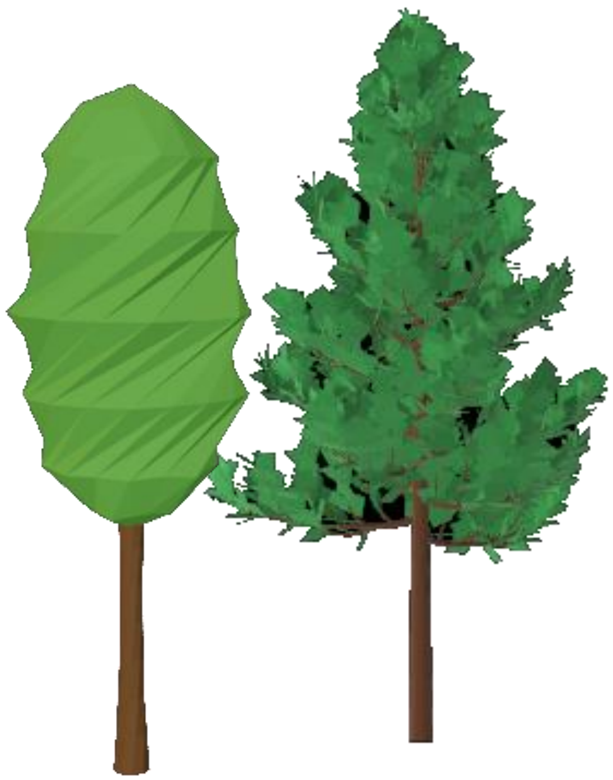
- In the 3D map, additional elements are included

Additional 3D Elements

- In the 3D map, additional elements are included
  - **High Value Building**, as detailed mesh obtained by manual editing or using Structure from Motion

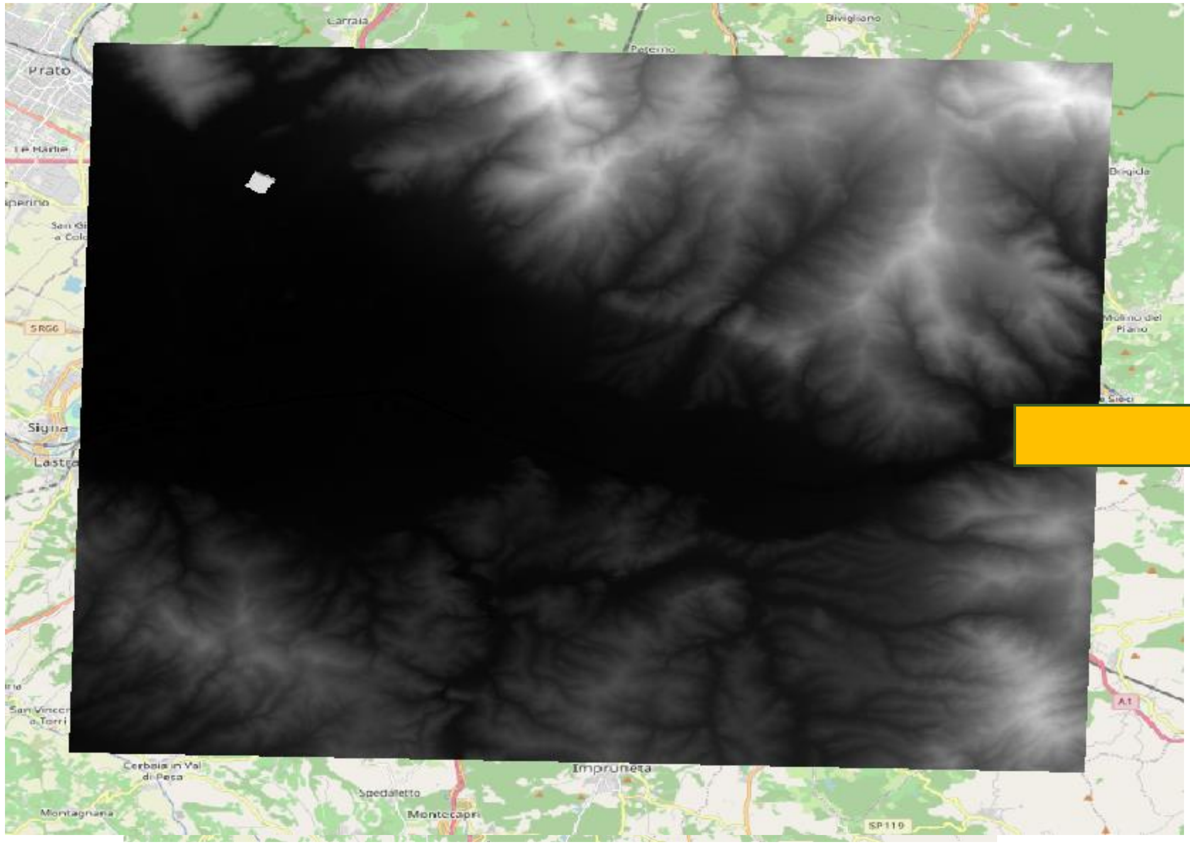
Additional 3D Elements

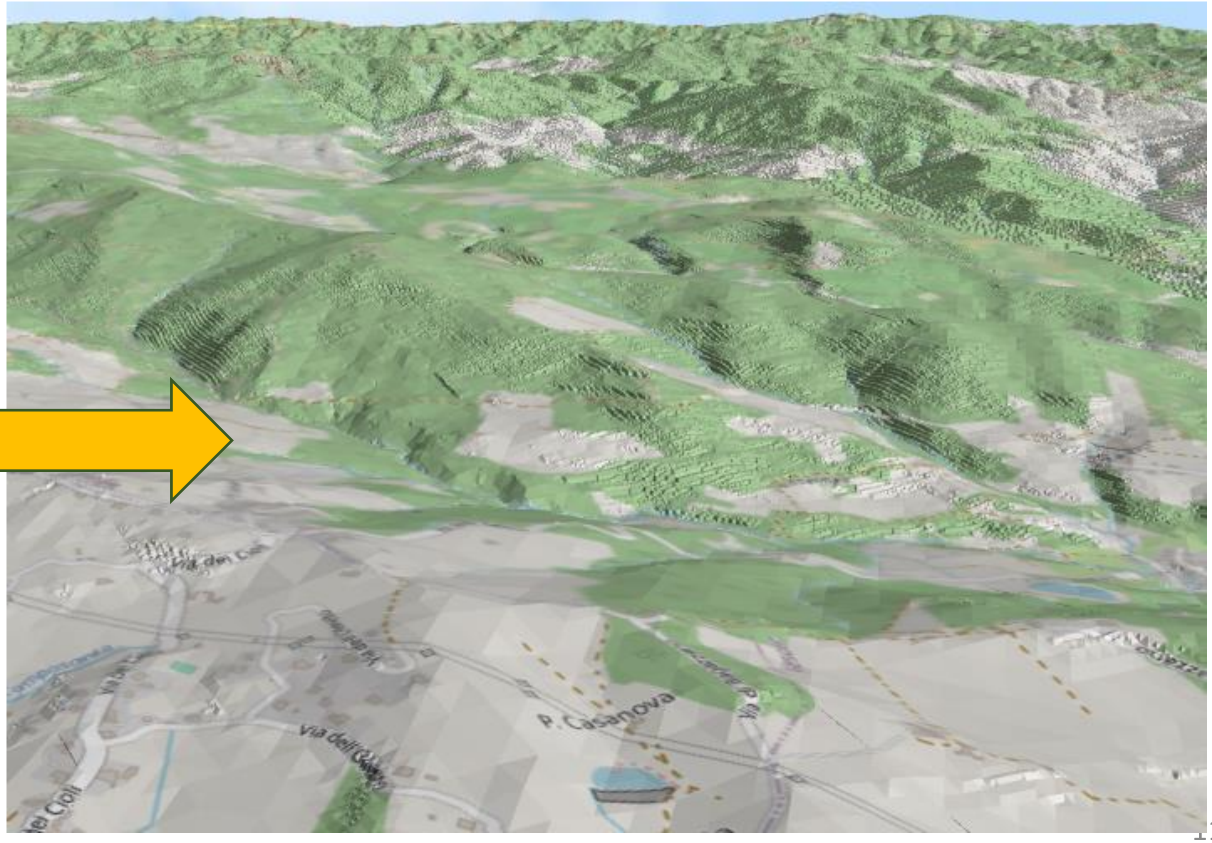
- In the 3D map, additional elements are included
  - **High Value Building**, as detailed mesh obtained by manual editing or using Structure from Motion
  - **Street furniture** like trees, fountains, etc.



Terrain

- Using a Digital Terrain Model (DTM) we can define the 3D terrain layer (useful to show different elevation in a city)





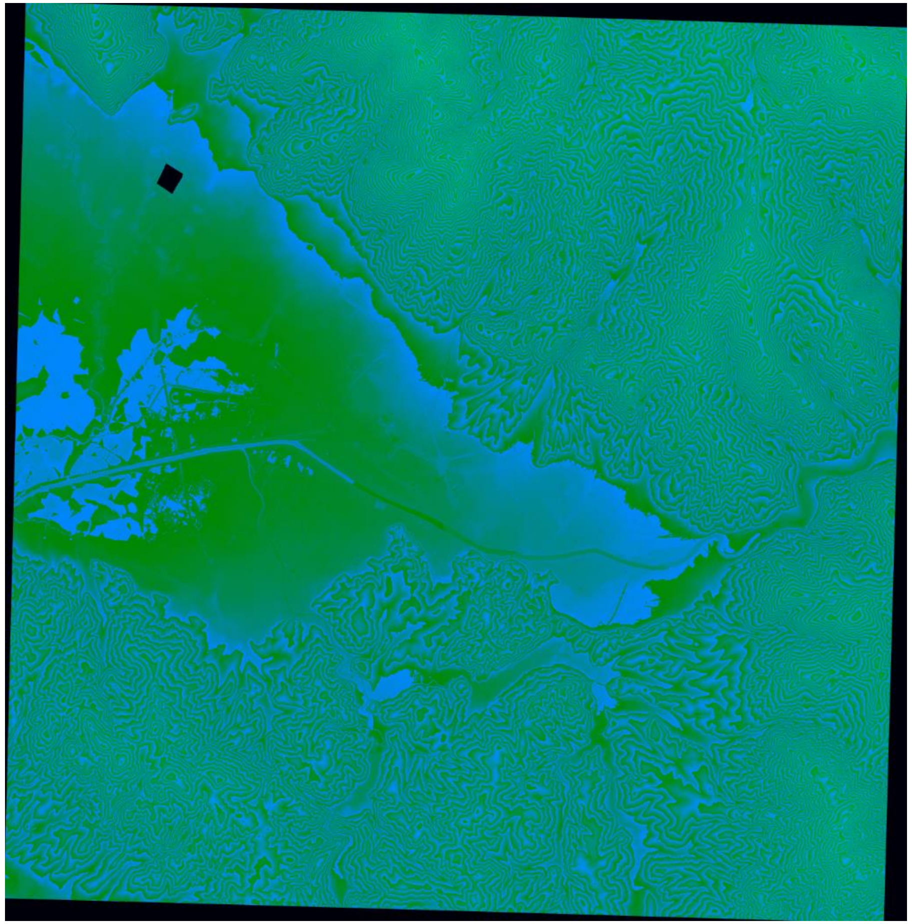
Terrain

- Different elevations are **encoded in RGB** using

$$\begin{cases} R = \left\lfloor \dfrac{100000 + 10v}{256^2} \right\rfloor \\ G = \left\lfloor \dfrac{100000 + 10v}{256} \right\rfloor - 256R \\ B = \lfloor 100000 + 10v \rfloor - 256^2 R - 256G \end{cases}$$

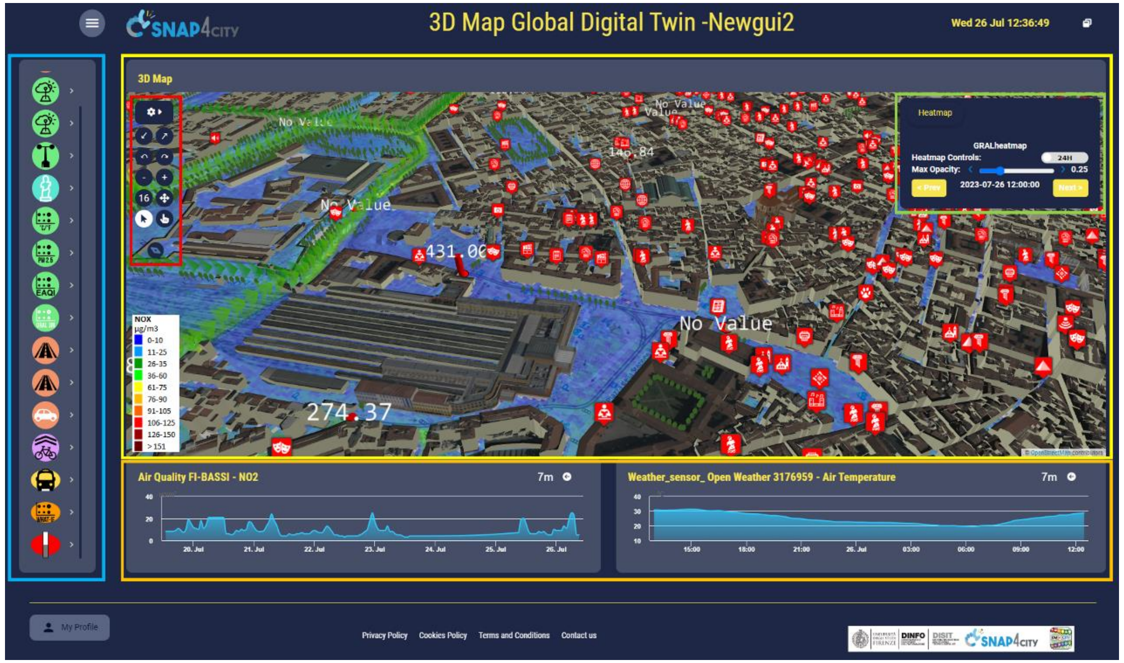
with a precision of 0.1 m

- Then, the RGB DTM is saved into the **Snap4City GeoServer** and provided to clients using REST calls with the possibility to select specific geographic tiles



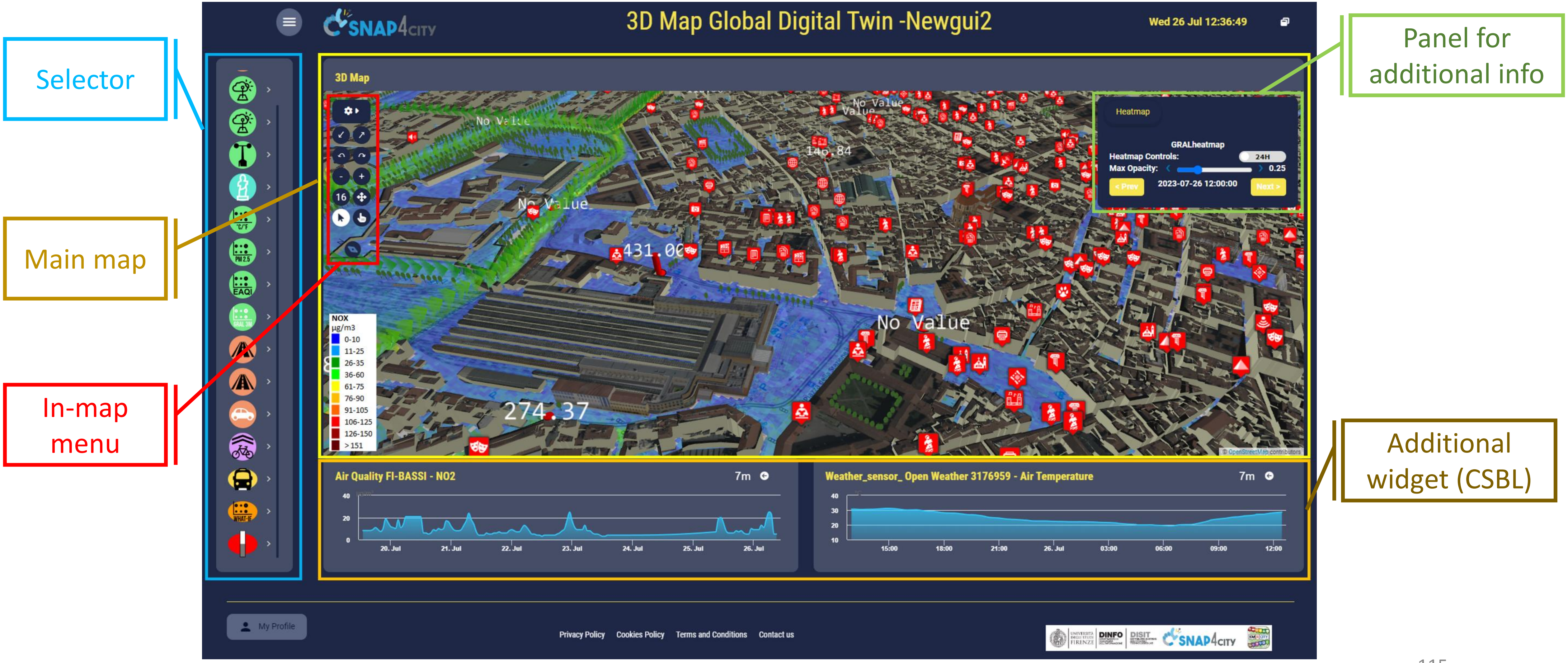
Integration and distribution

- Acquired and produced data are integrated into the digital twin model
- The model is distributed as a **Snap4City dashboard**
- A **3D multi-data map** is built as an interactive web interface



- Additional widget can be included in the same dashboard to focus on particular information

Integration and distribution



Main functionalities

- The SCDT in Snap4City includes the following functionalities:
  - 3D terrain

Main functionalities

- The SCDT in Snap4City includes the following functionalities:
  - 3D terrain, textured with different orthomaps

Main functionalities

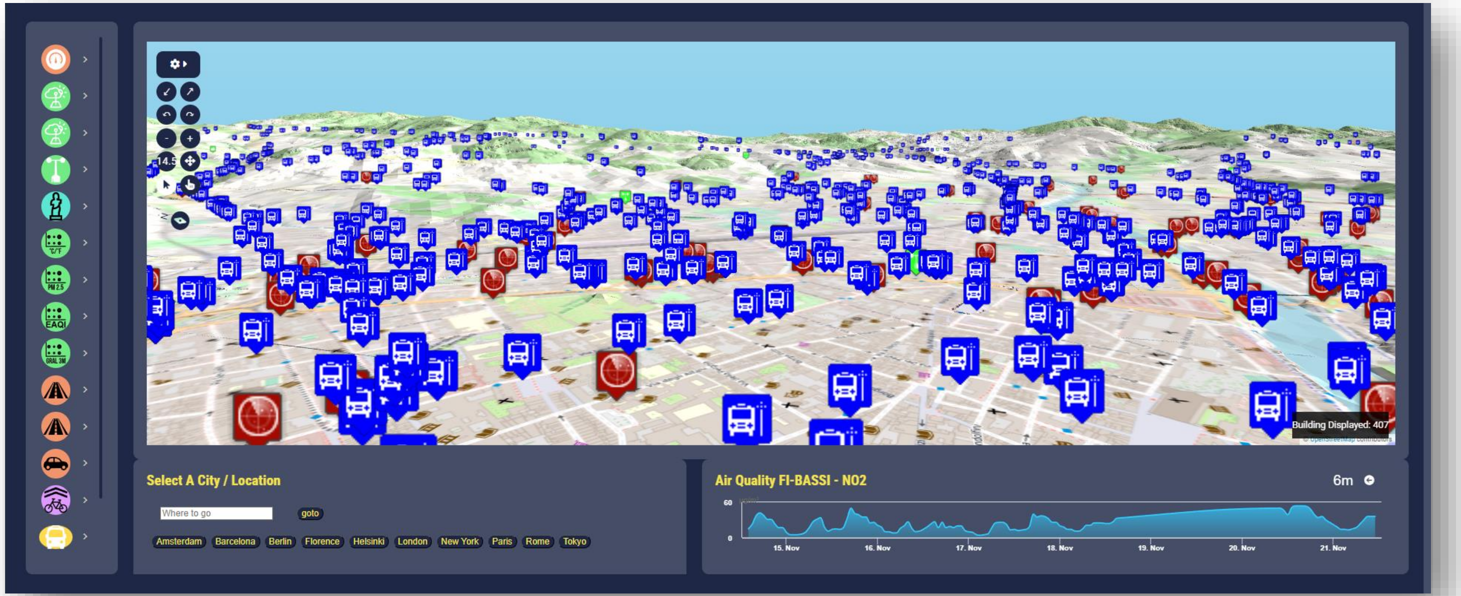
- The SCDT in Snap4City includes the following functionalities:
  - 3D terrain, textured with different orthomaps

Main functionalities

- The SCDT in Snap4City includes the following functionalities:
  - 3D terrain, textured with different orthomaps and heatmaps

Main functionalities

- The SCDT in Snap4City includes the following functionalities:
  - 3D terrain, textured with different orthomaps and heatmaps
  - PINs are used to represent IoT devices, Services, Public Transport stops



Main functionalities

- The SCDT in Snap4City includes the following functionalities:
  - 3D terrain, textured with different orthomaps and heatmaps
  - PINs are used to represent IoT devices, Services, Public Transport stops
  - Each PIN can be clicked to open a pop-up with detailed information

Main functionalities

- The SCDT in Snap4City includes the following functionalities:
  - 3D terrain, textured with different orthomaps and heatmaps
  - PINs are used to represent IoT devices, Services, Public Transport stops
  - Each PIN can be clicked to open a pop-up with detailed information
  - RT-data can be queried and shown in dedicated widget using Client-Side Business Logic

Main functionalities

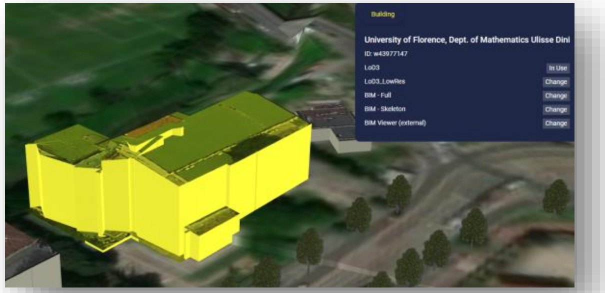
- The SCDT in Snap4City includes the following functionalities:
  - 3D building of the city are represented

Main functionalities

- The SCDT in Snap4City includes the following functionalities:
  - 3D building of the city are represented
  - Each building can be picked, to accesso to additional information

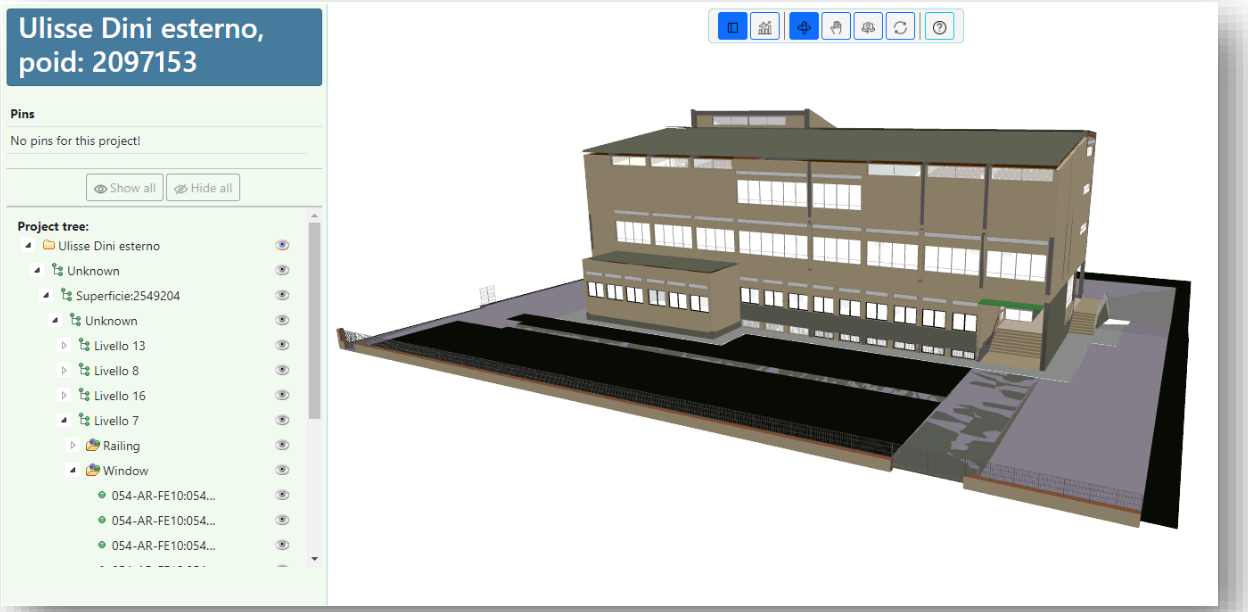
Main functionalities

- The SCDT in Snap4City includes the following functionalities:
  - 3D building of the city are represented
  - Each building can be picked, to accesso to additional information
  - For specific buildings, after selecting it the 3D model can be changed



Main functionalities

- The SCDT in Snap4City includes the following functionalities:
  - 3D building of the city are represented
  - Each building can be picked, to accesso to additional information
  - For specific buildings, after selecting it the 3D model can be changed
  - When available, access to inspect the BIM of the building is provided



Main functionalities

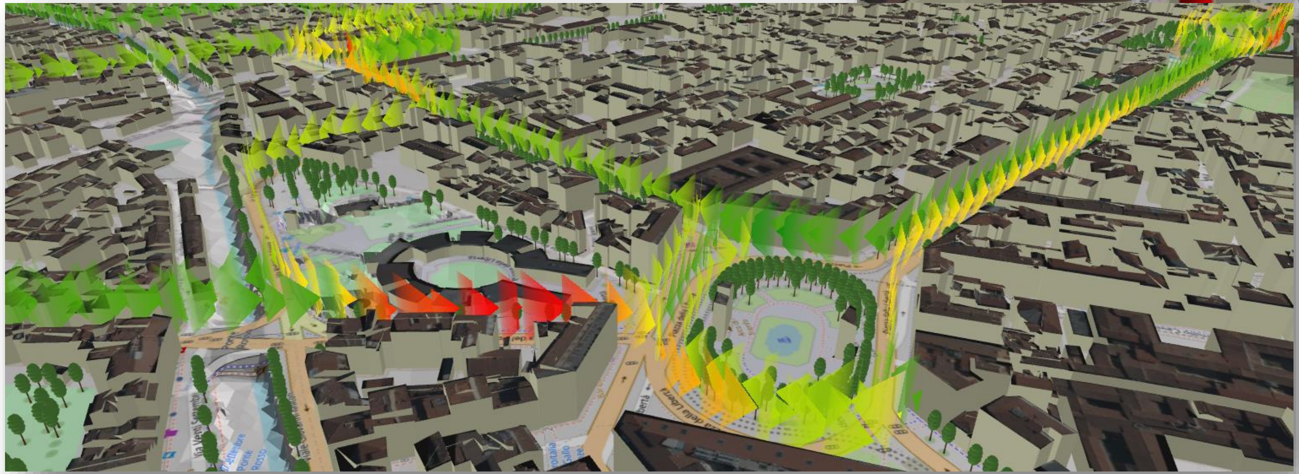
- The SCDT in Snap4City includes the following functionalities:
  - 3D building of the city are represented
  - Each building can be picked, to accesso to additional information
  - For specific buildings, after selecting it the 3D model can be changed
  - When available, access to inspect the BIM of the building is provided
  - In the case of Google 3D tiles, the picking functionality was implemented exploiting invisible extruded buildings

Main functionalities

- Specific 3D representation have been devised to show
  - Sensor measurements

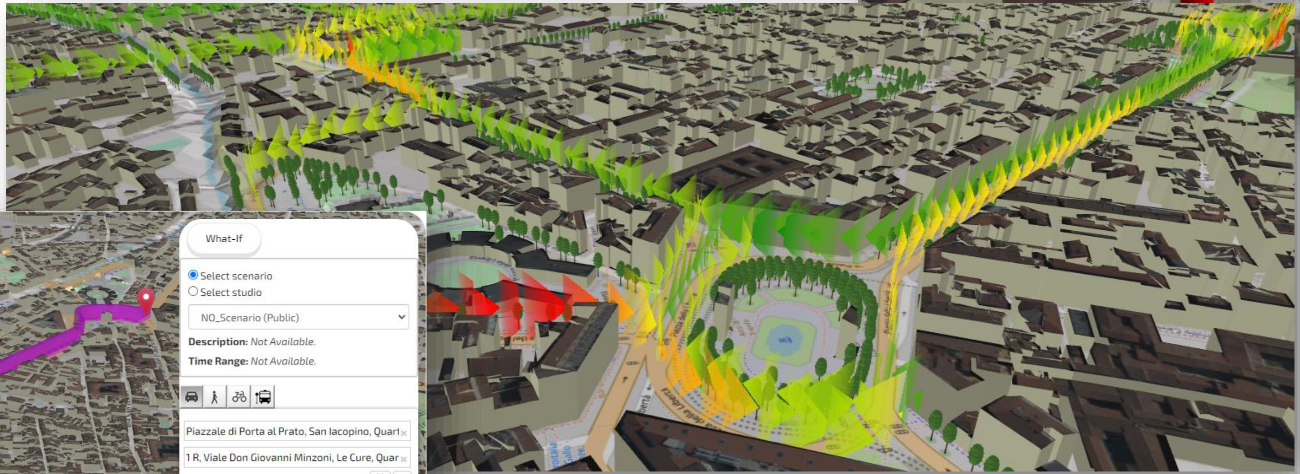
Main functionalities

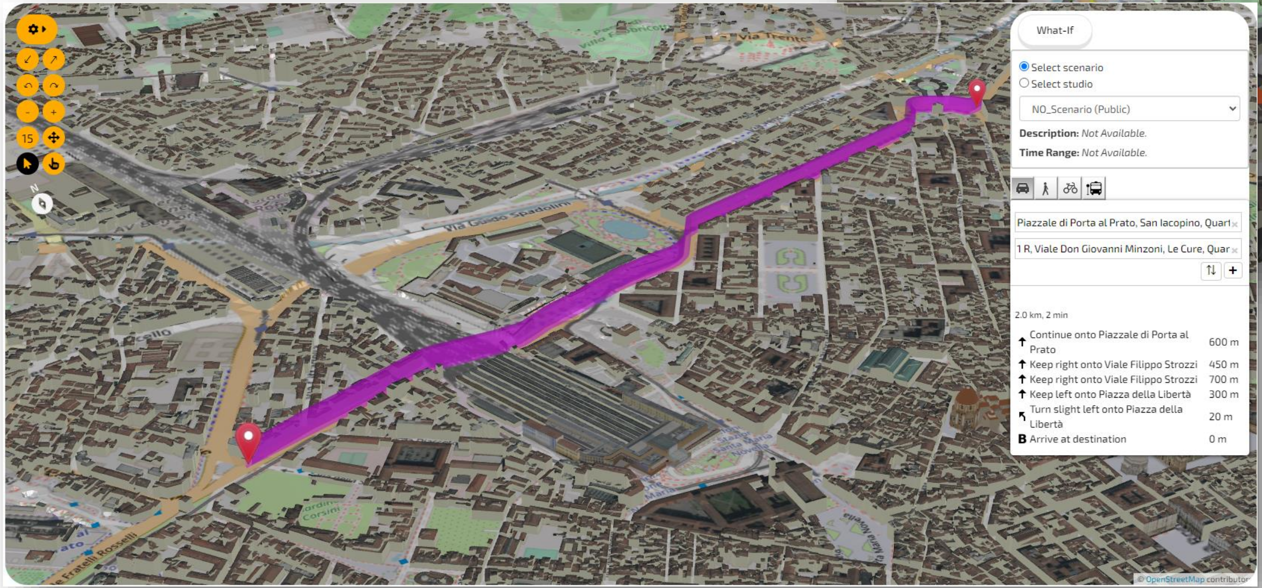
- Specific 3D representation have been devised to show
  - Sensor measurements
  - Traffic flows



Main functionalities

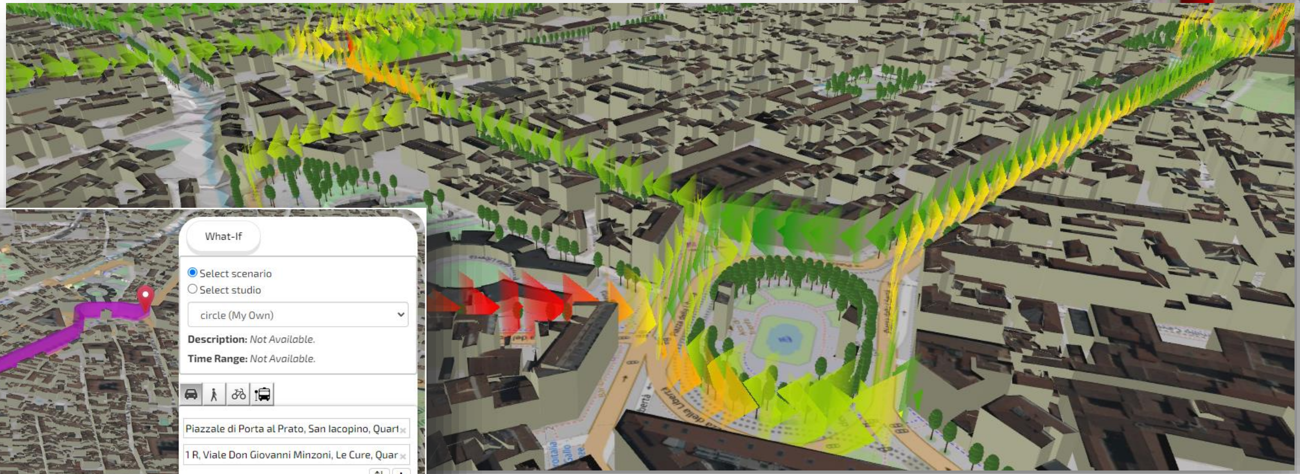
- Specific 3D representation have been devised to show
    - Sensor measurements
    - Traffic flows
    - Routing paths

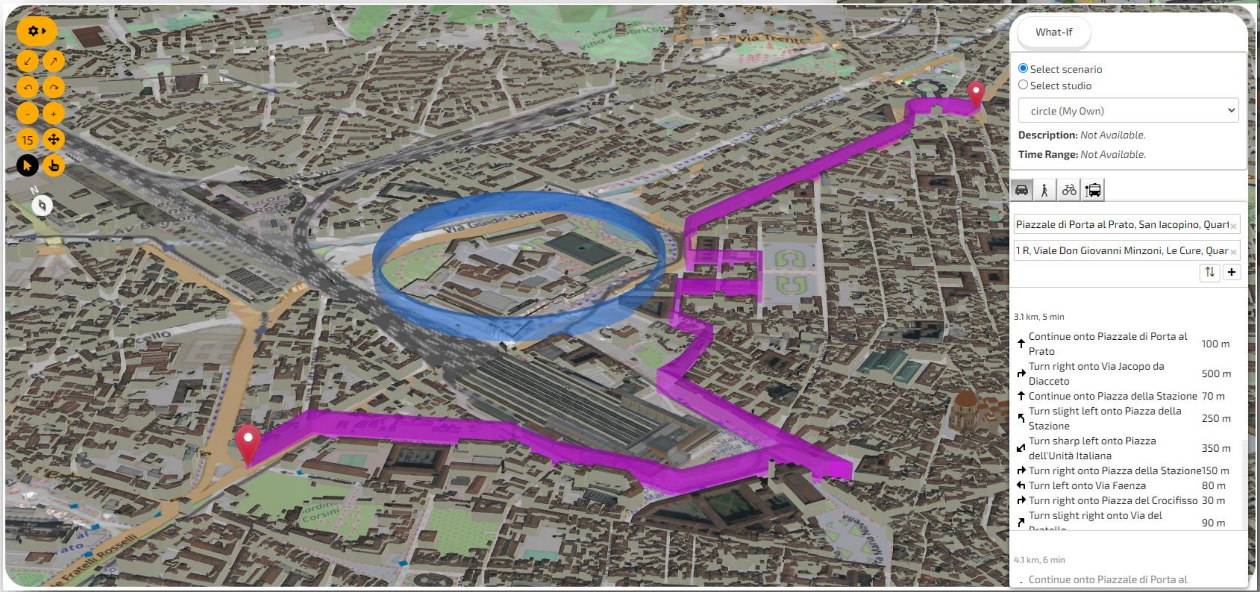




Main functionalities

- Specific 3D representation have been devised to show
  - Sensor measurements
  - Traffic flows
  - Routing path, also for What-if analysis

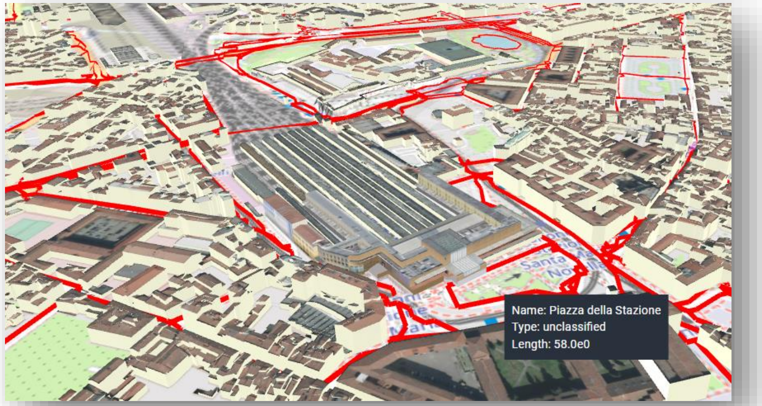


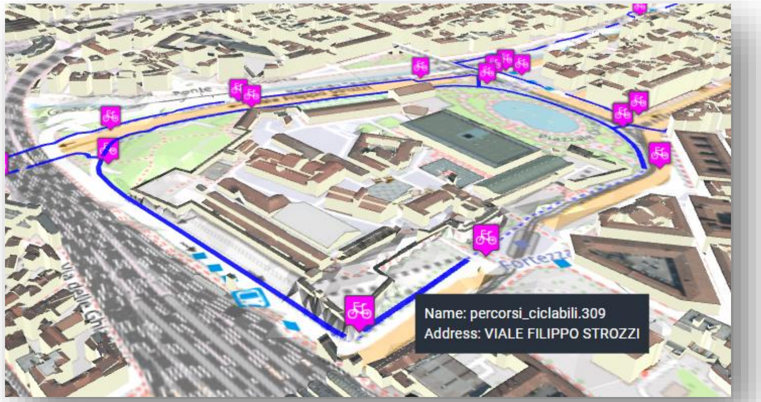


UNIVERSITÀ DEGLI STUDI FIRENZE | DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE | DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB DISTRIBUTED DATA INTELLIGENCE AND TECHNOLOGIES LAB | SNAP4CITY

Main functionalities

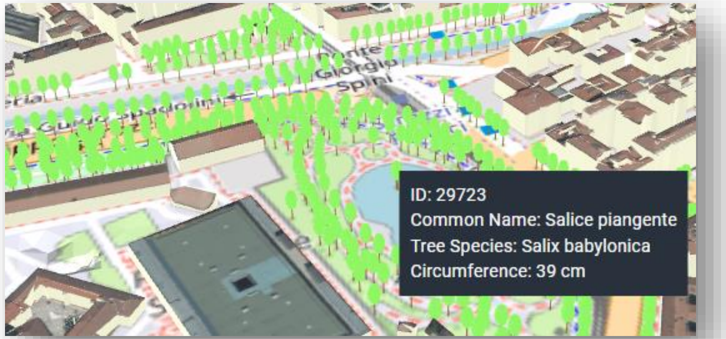
- The SCDT also includes
  - Roads, cycling paths, bus routes



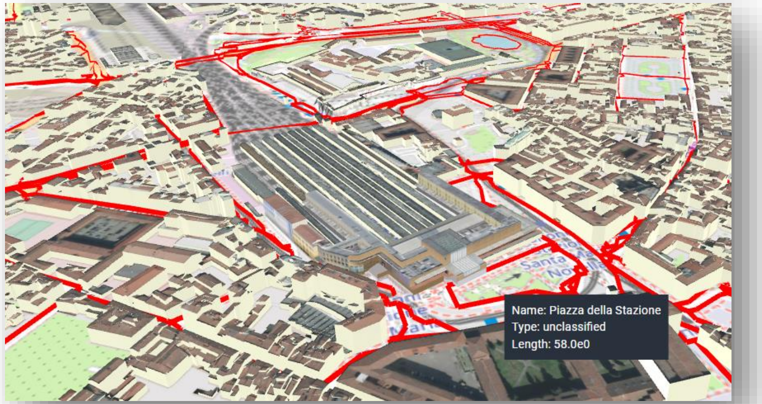


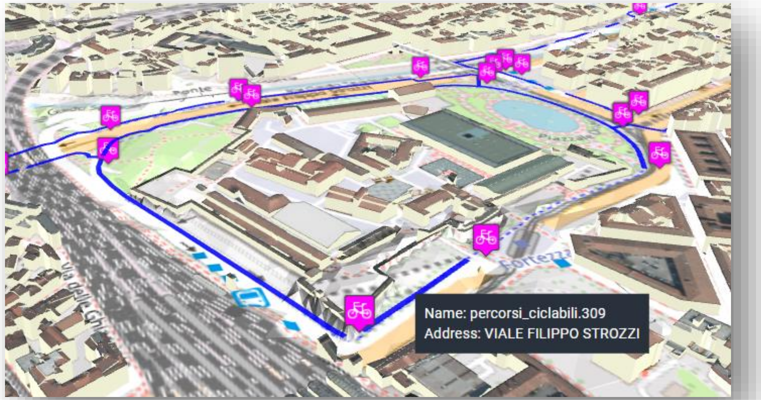
UNIVERSITÀ DEGLI STUDI FIRENZE
DINFO DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE
DISIT DISTRIBUTED SYSTEMS AND INTERNET TECHNOLOGIES LAB DISTRIBUTED DATA INTELLIGENCE AND TECHNOLOGIES LAB
SNAP4CITY

Main functionalities



- The SCDT also includes
  - Roads, cycling paths, bus routes
  - Additional 3D elements like trees, that can be requested with different level of details



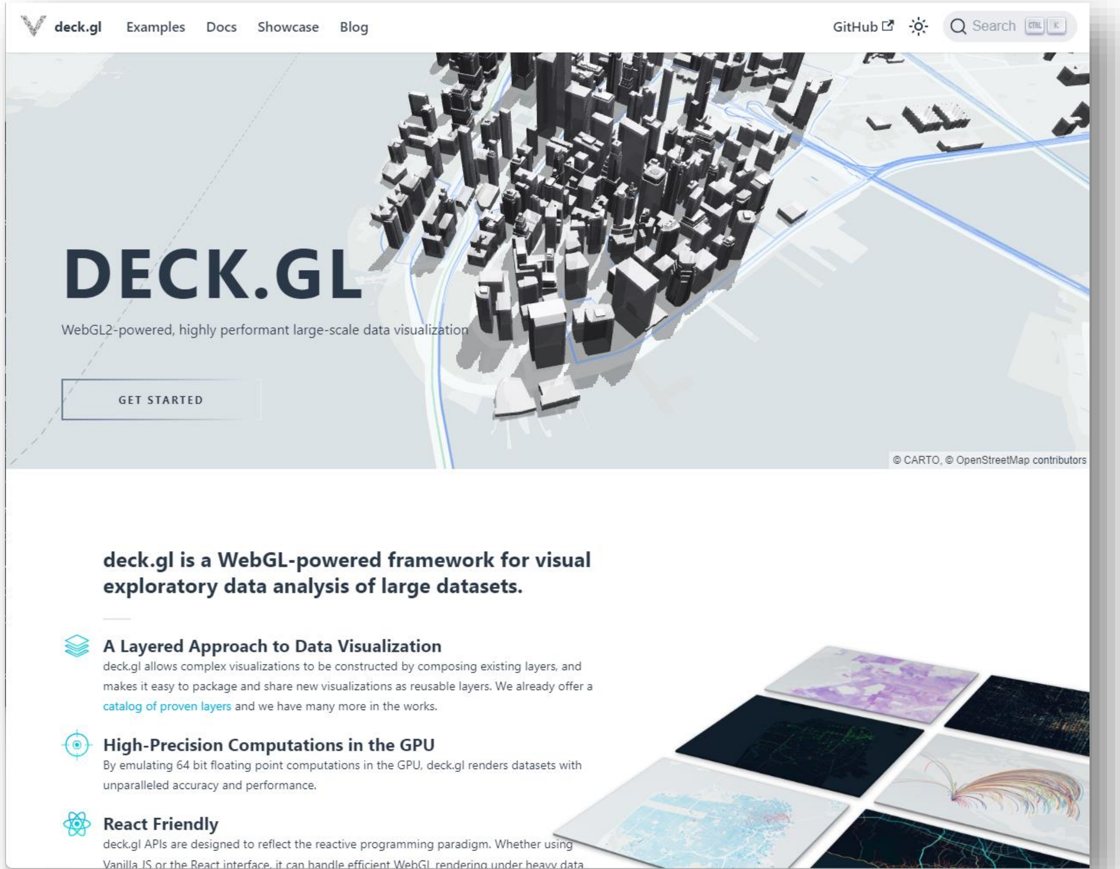


3D Multi-data Map Interactive Web Interface

- The architecture implements a **client-side business logic** developed in JavaScript exploiting the **Deck.gl framework**
- REST calls are used to query Snap4City API and **load data on user demands**
- PINs for POIs or IoT Devices, and paths and areas are **retrieved with geographic queries** toward the Snap4City SuperService Map (the KM4City interface)
- Heatmaps, Orthomaps, and the RGB DTM are instead **provided via WMS protocol** with HTTPS requests toward the Snap4City GeoServer

Deck.gl

- Deck.gl is a **JavaScript** framework to handle large database and create web applications
- It is based on a **layered architecture**…
- …and provide tools to integrate and visualize **geographical data**

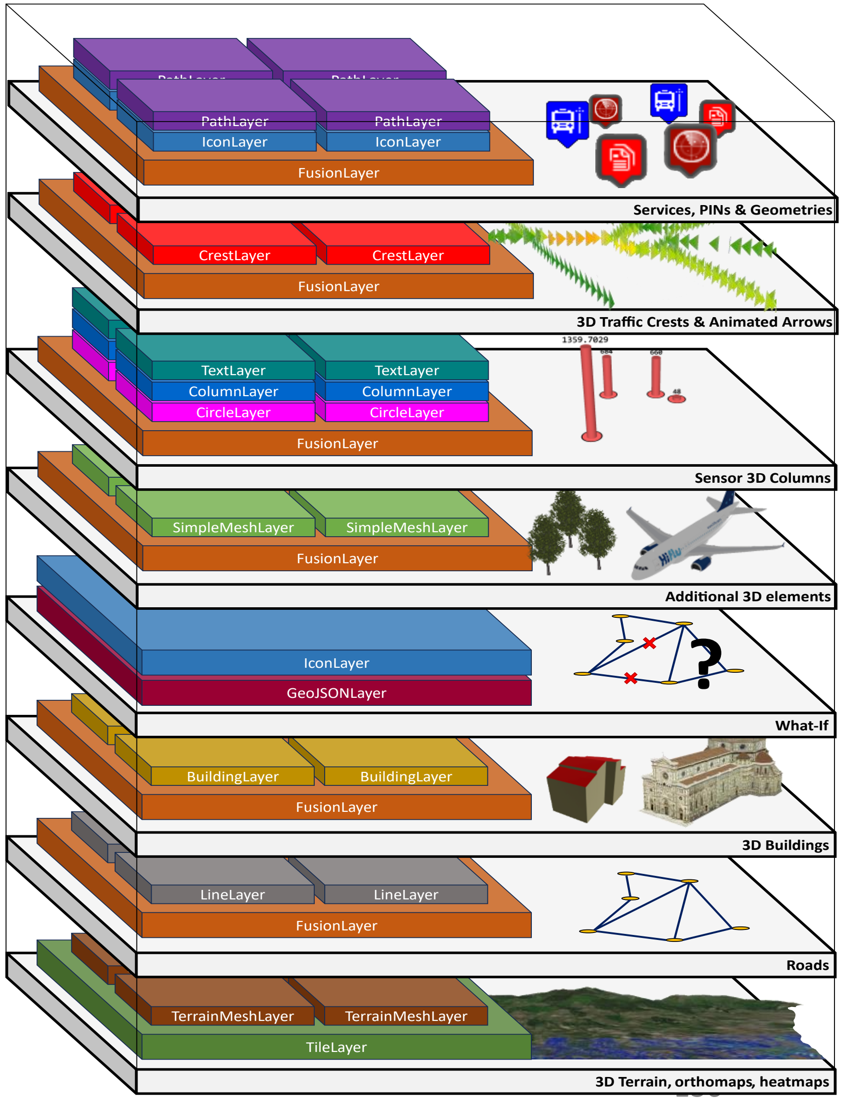


Layered architecture

- Deck.gl **layered structure** was exploited

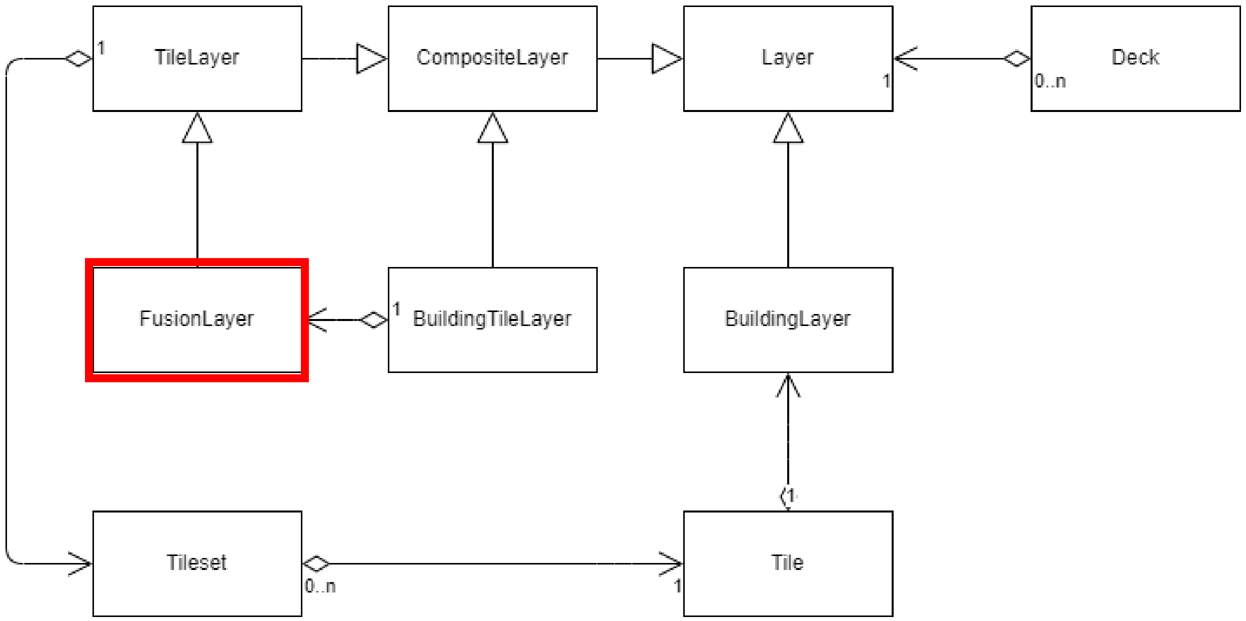
- Different entities are rendered in **different layers**, each with its **own safe context** to avoid interferences and be able to load each of them **independently**

- Data are rendered with a **tiled approach**, in order to display only the visible data. This is fundamental to handle **large SCDT**



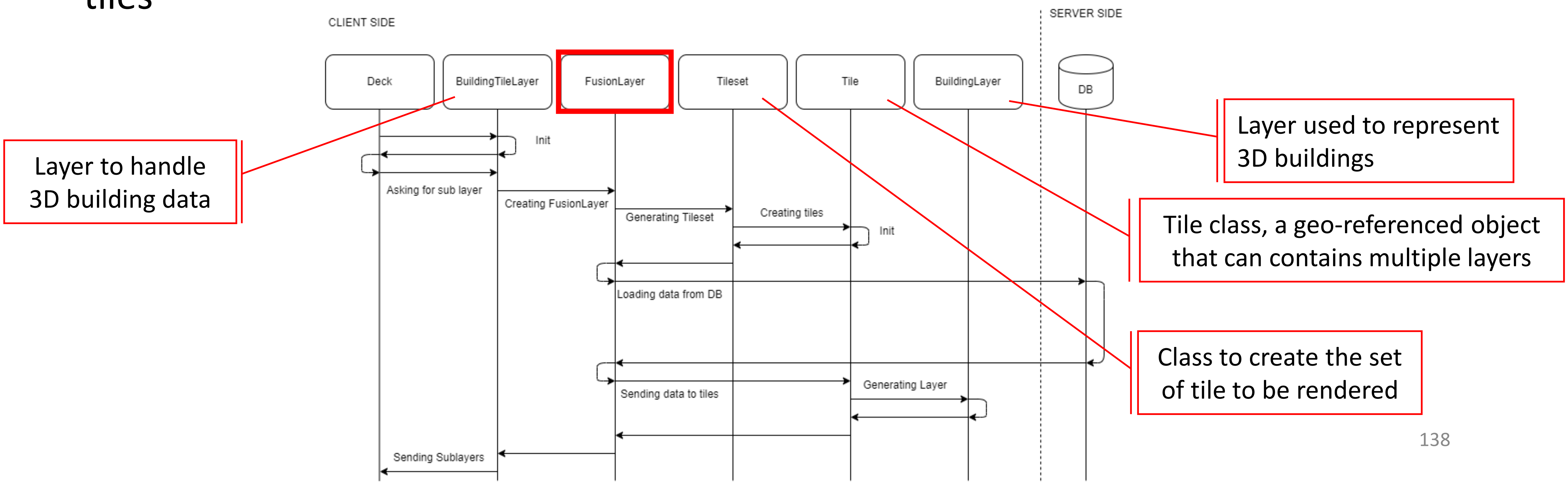
Fusion Layer

- The core of the architecture is the so-called **FusionLayer**
- The FusionLayer **extends** the Deck.gl architecture



Fusion Layer

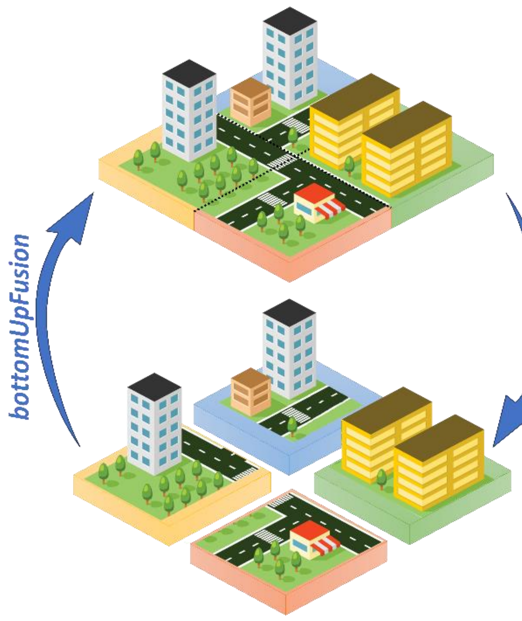
- The core of the architecture is the so-called **FusionLayer**
- The FusionLayer **extends** the Deck.gl architecture
- The FusionLayer **is specialized to manage the data to be rendered** in the SCDT tiles



Fusion Layer

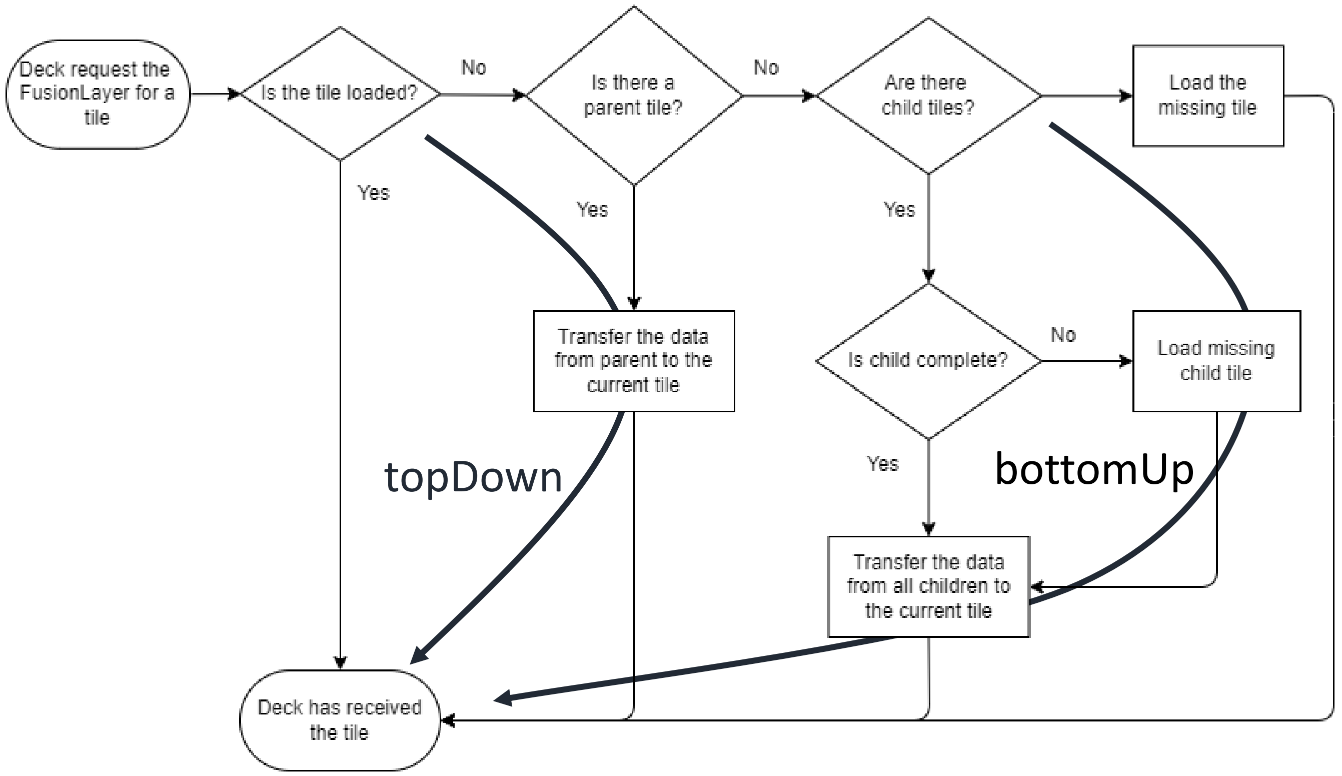
- FusionLayer implements two main functionalities

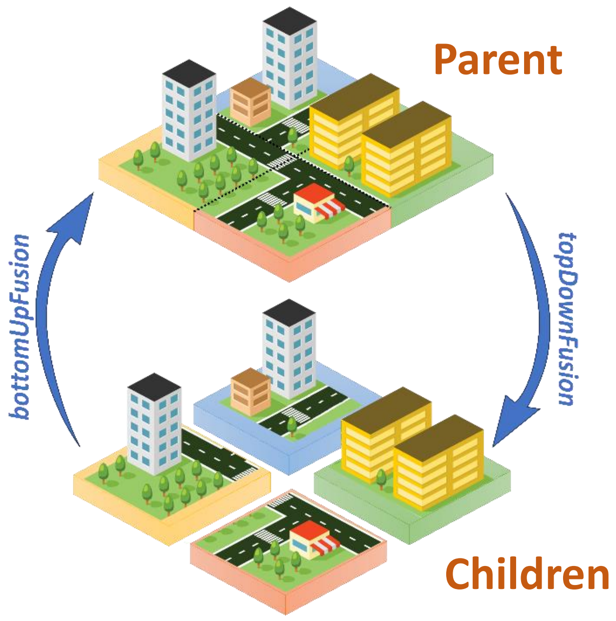
1. The **bottomUp** and **topDown** Fusions
2. The **deepLoad**



Fusion Layer

- The bottomUp and topDown fusion are used to **reduce server calls and exchanged data** by reusing data already downloaded for tile at different zoom levels



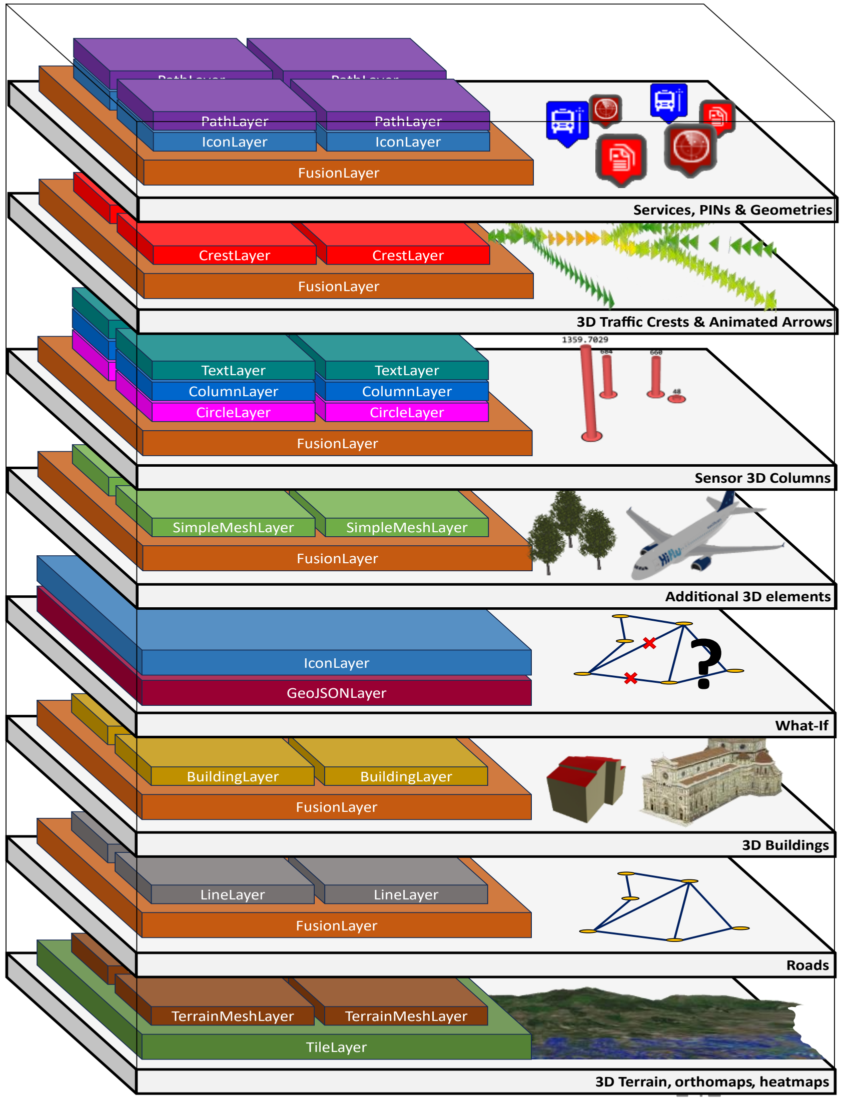


Fusion Layer

- The **deepLoad** function is used when dealing with **data available at a single zoom level** (e.g., traffic flows, 3D buildings)
- In the standard Deck.gl, such data would be retrieved at the available zoom **regardless of the actual zoom** used in the map
- At low zoom level, this behavior would lead to a **very large number of tiles to be rendered**, slowing down the performance
- The deepLoad solve this problem by requiring data at the available zoom, then **creating virtual tiles with dimension according to the actual zoom level** and filling these virtual tiles with the retrieved data, thus **reducing the rendering processes** to be carried out

Layered Architecture

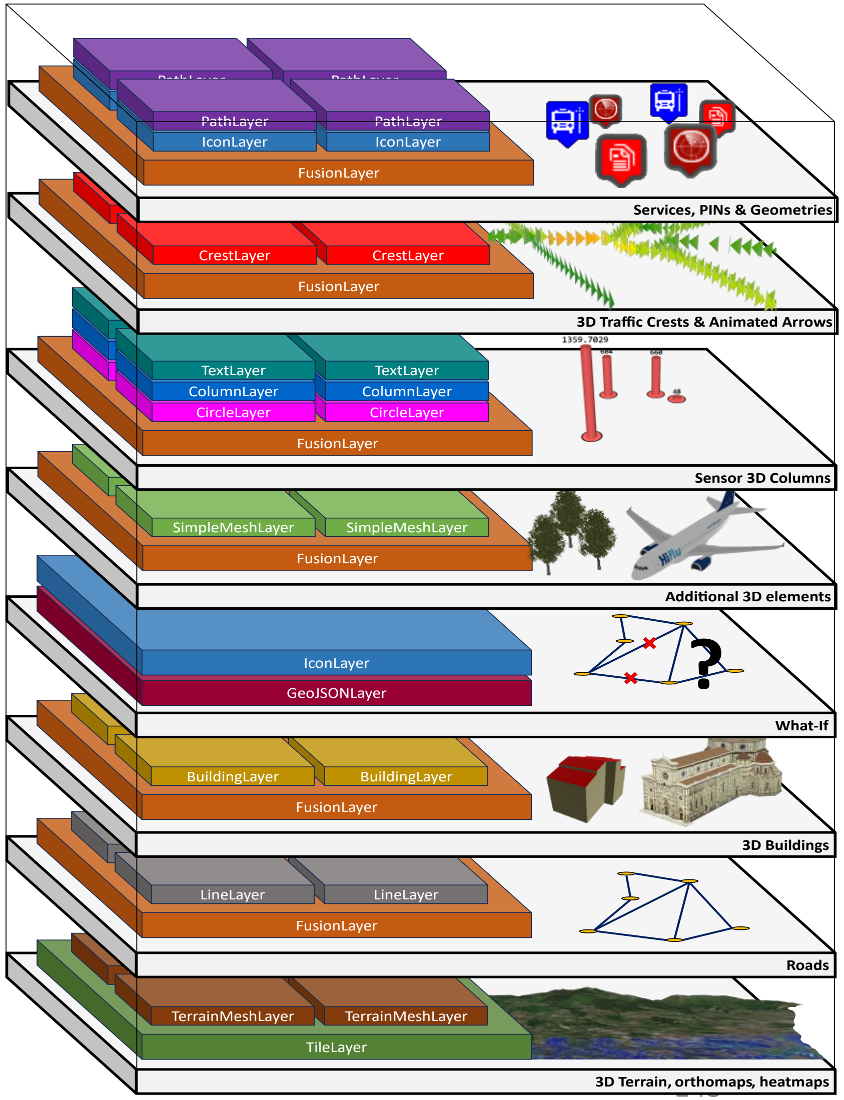
- The **TerrainMeshLayer** is used to display the 3D terrain, othomaps, and heatmaps using a tiled approach
- Each tile can load different resources **using business-logic callbacks** to the WMS GeoServer
- The **Martini tessellation** algorithm was used to obtain the terrain mesh from the RGB-encoded DTM
- Multiple textures can be applied over the terrain using a **merging procedure carried out in the GPU fragment shader**



Layered Architecture

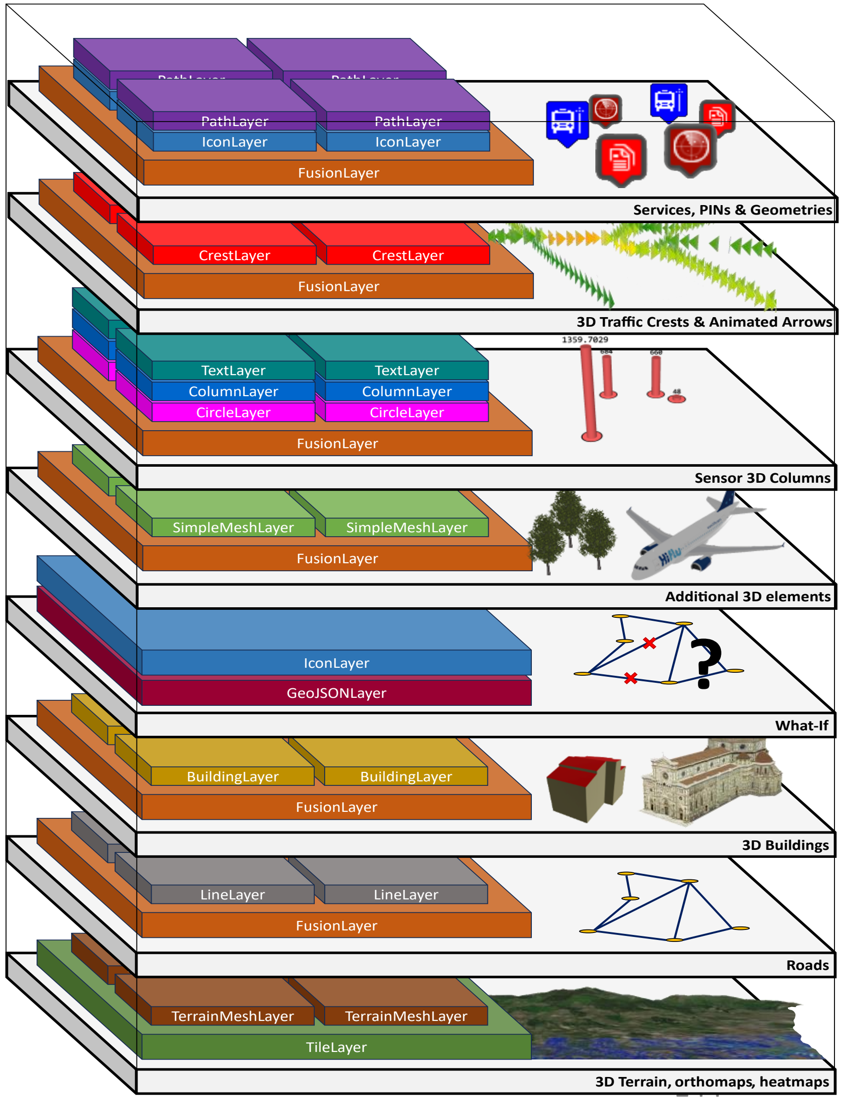
- The **LineLayer** is used to display the road graph retrieved from the KB using SPARQL queries, limited to the tiles in view

```
PREFIX km4c: <http://www.disit.org/km4city/schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdfsn: <http://www.w3.org/2003/01/geo/wgs84_pos#>
PREFIX dct: <http://purl.org/dc/terms/>
SELECT ?road ?roadName ?roadElement ?highwayType ?startLat ?startLong ?endLat ?endLong
?compositionType ?operatingStatus ?trafficDir ?length
WHERE {
   ?road a km4c:Road.
   ?road km4c:extendName ?roadName.
   ?road km4c:containsElement ?roadElement.
   ?road km4c:inMunicipalityOf ?municip.
   ?municip foaf:name "Firenze".
   ?roadElement km4c:highwayType ?highwayType.
   ?roadElement km4c:composition ?compositionType.
   ?roadElement km4c:operatingStatus ?operatingStatus.
   ?roadElement km4c:trafficDir ?trafficDir.
   ?roadElement km4c:length ?length.
   ?roadElement km4c:startsAtNode ?startNode.
   ?startNode rdfsn:lat ?startLat.
   ?startNode rdfsn:long ?startLong.
   ?roadElement km4c:endsAtNode ?endNode.
   ?endNode rdfsn:lat ?endLat.
   ?endNode rdfsn:long ?endLong.
   FILTER (
      xsd:float(?startLat) < {north} && xsd:float(?endLat) > {south} &&
      xsd:float(?startLong) < {east} && xsd:float(?endLong) > {west}
   )
}
```



Layered Architecture

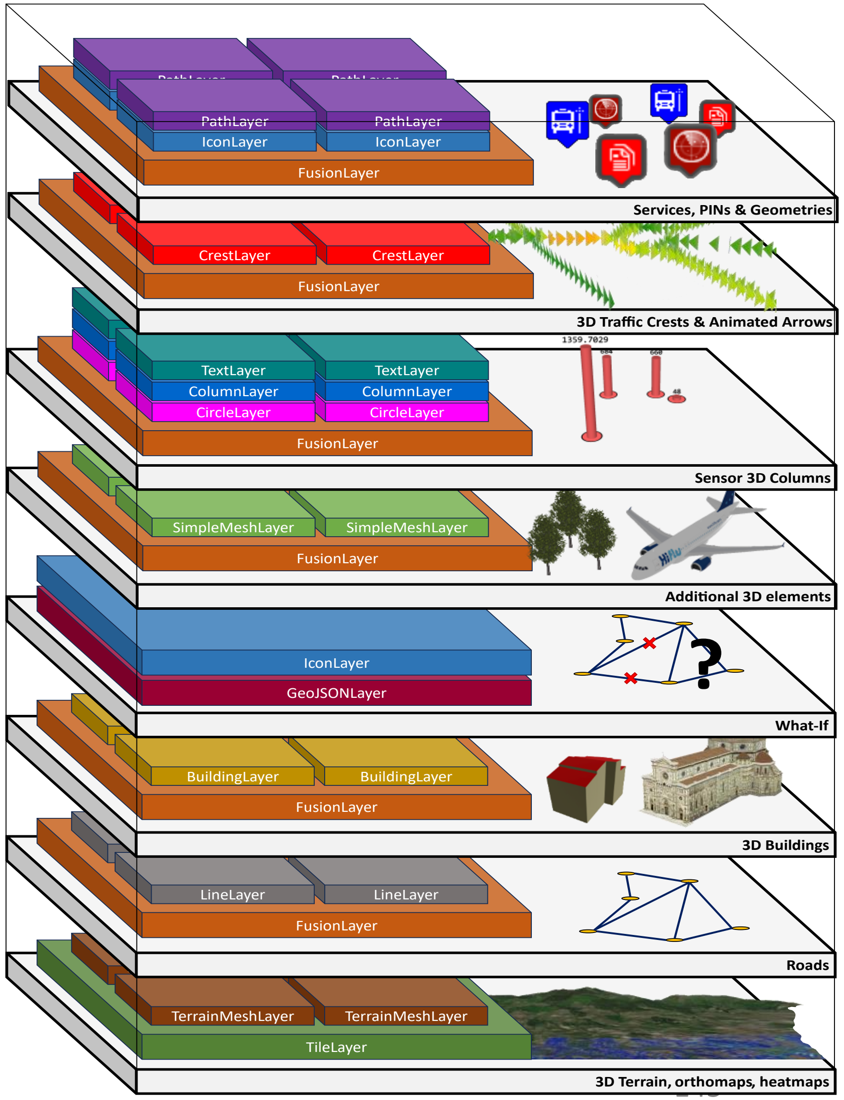
- A tiled approach exploiting the BuildingLayer is used to show LoD3 3D buildings
- 3D model tiles are **dynamically loaded taking into account the viewing position**, prioritizing closer tiles w.r.t. farthest ones
- This approach ensures that **only buildings in visible tiles are rendered**, reducing the use of client resources
- Such solution is **fundamental to handle the huge number of buildings** present in a city



Layered Architecture

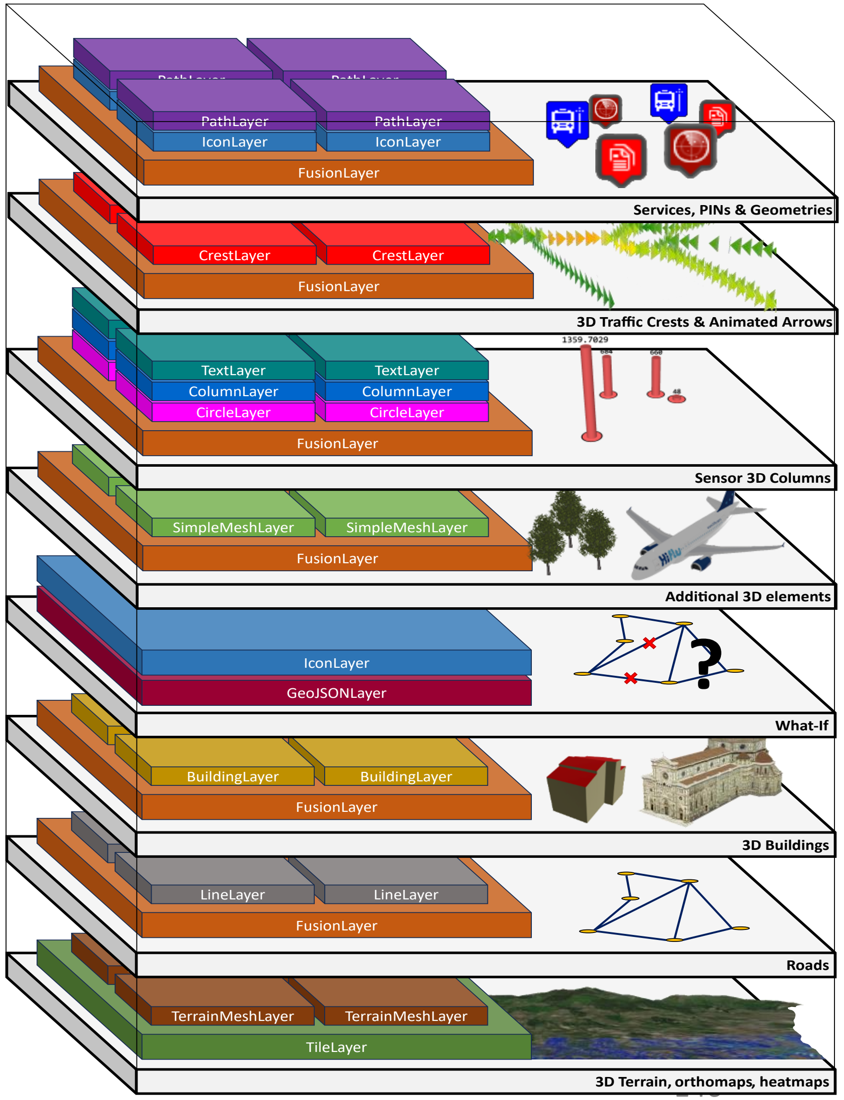
- A novel layer called **WhatIfLayer** has been designed to handle What-If analysis in the SCDT web interface

- It includes a series of sub-layer
    - PathLayer
    - GeoJSONLayer
    - IconLayer



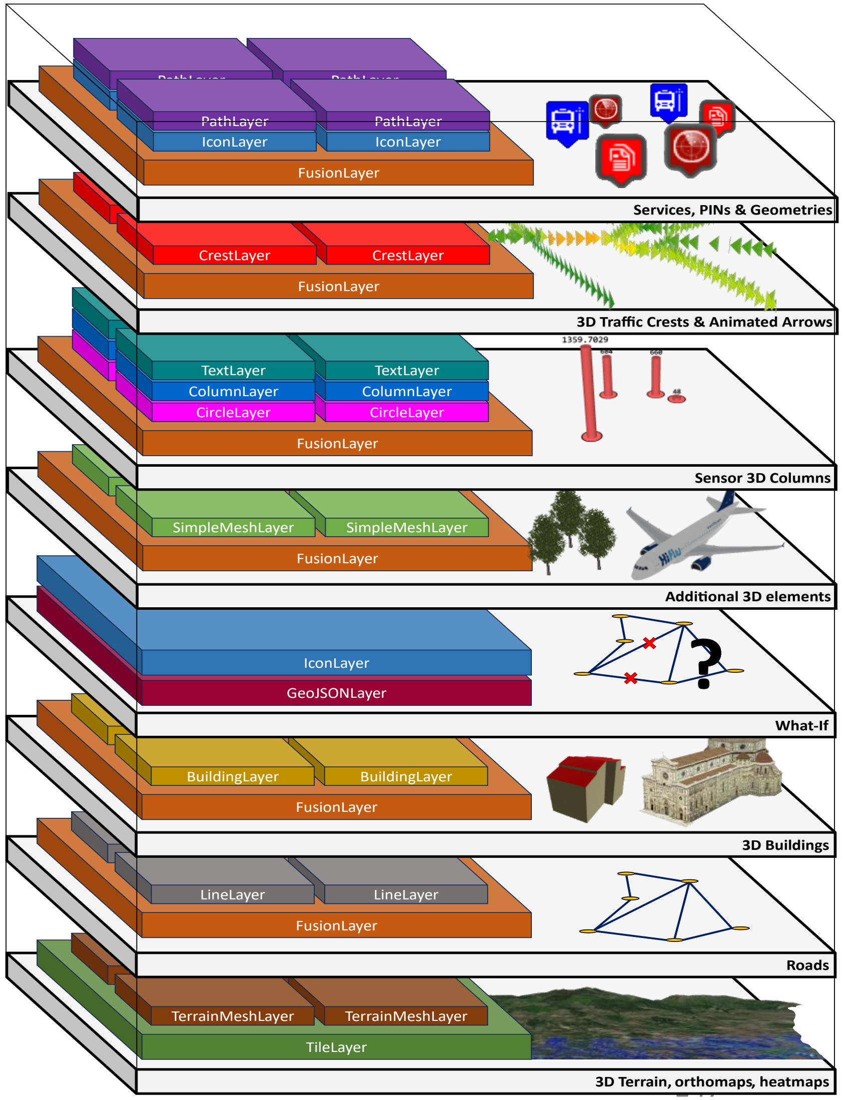
Layered Architecture

- Additional **3D entities**, such as trees, are shown in the 3D map using a layer structure similar to the one used to represent the buildings.



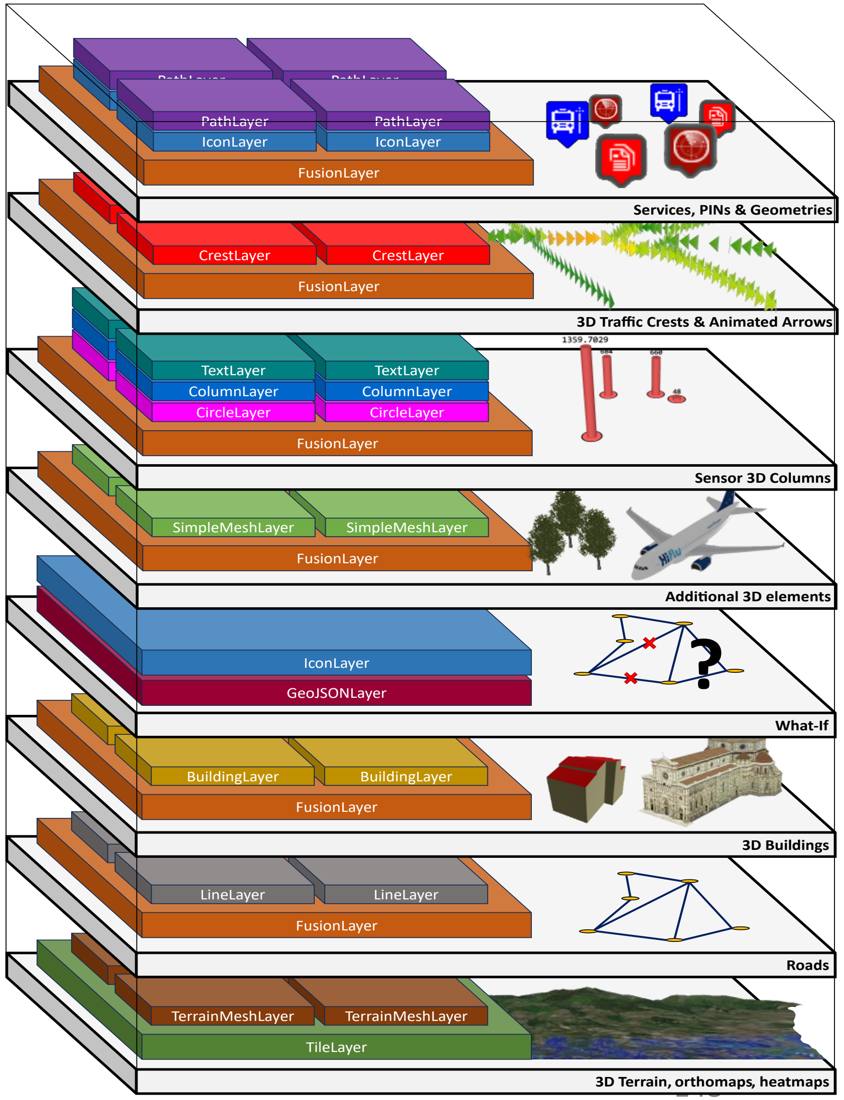
Layered Architecture

- A combination of different layers was used to show 3D representation of IoT Sensor
- In particular, **3D Pillars** are used to indicate the values registered by traffic sensors



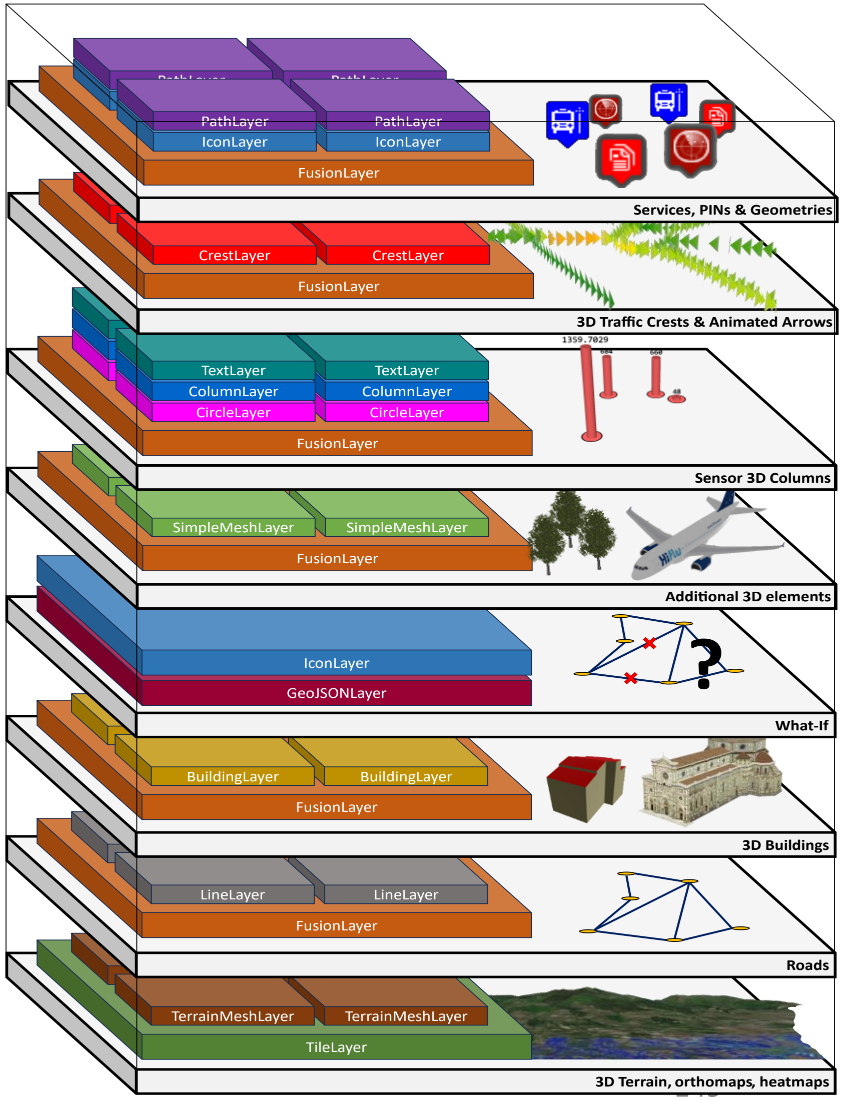
Layered Architecture

- Solutions to show specific information must be reconsidered when passing from a 2D to a 3D representation
- For example, traffic flow information cannot be appreciated if visualized as a 2D heatmap
- We developed the **CrestLayer** in order to show traffic information as static or animated 3D crests

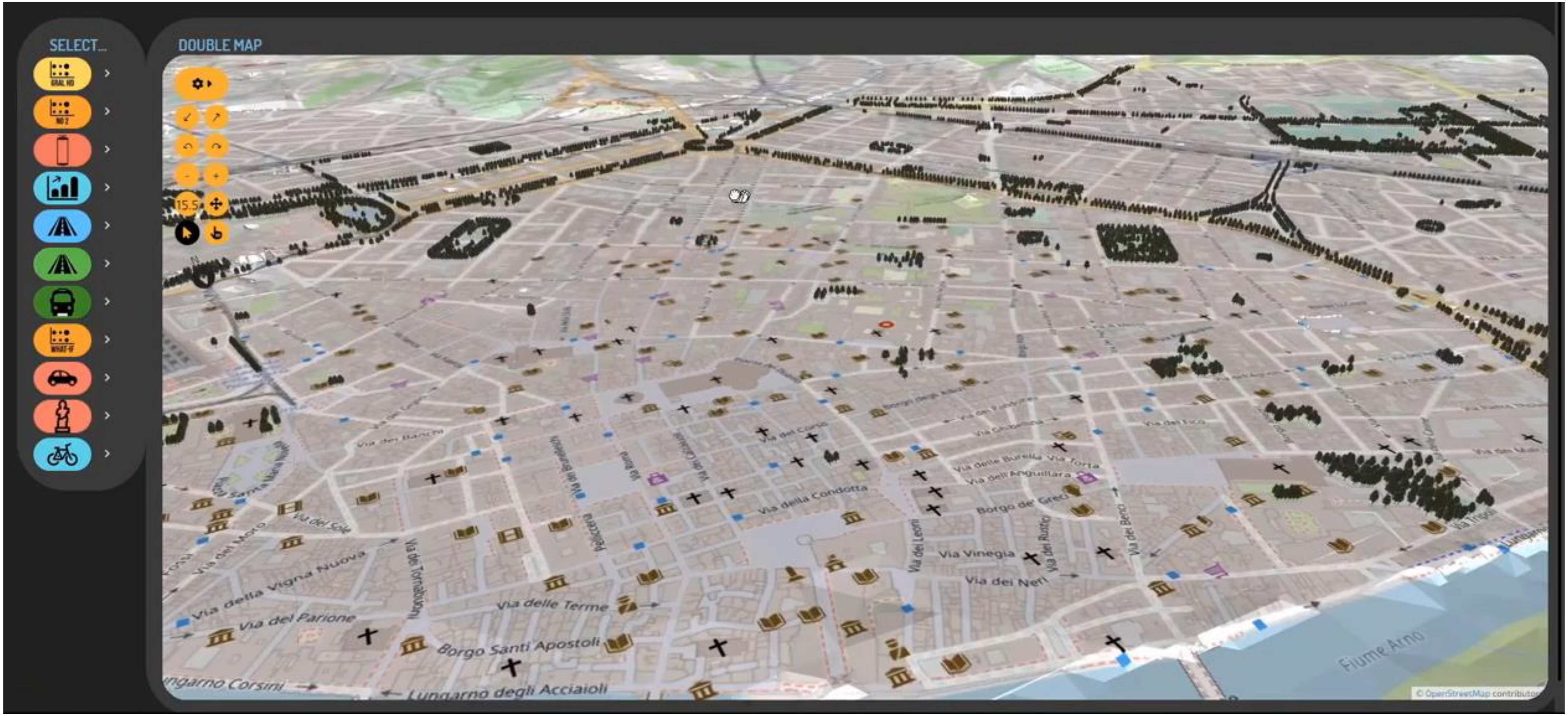


Layered Architecture

- **Paths** and **geometries** are represented in specific layers
- The **PathLayer** was used to display geometries such as cycling paths
- The IconLayer was used to show **pickable PINs** for IoT Devices or POIs
- **PIN can also be dynamic**, changing their shape or color according their kind
- Both layers are composed by a series of sub-layers to handle different elements (e.g., PINs indicating the bus stop locations)

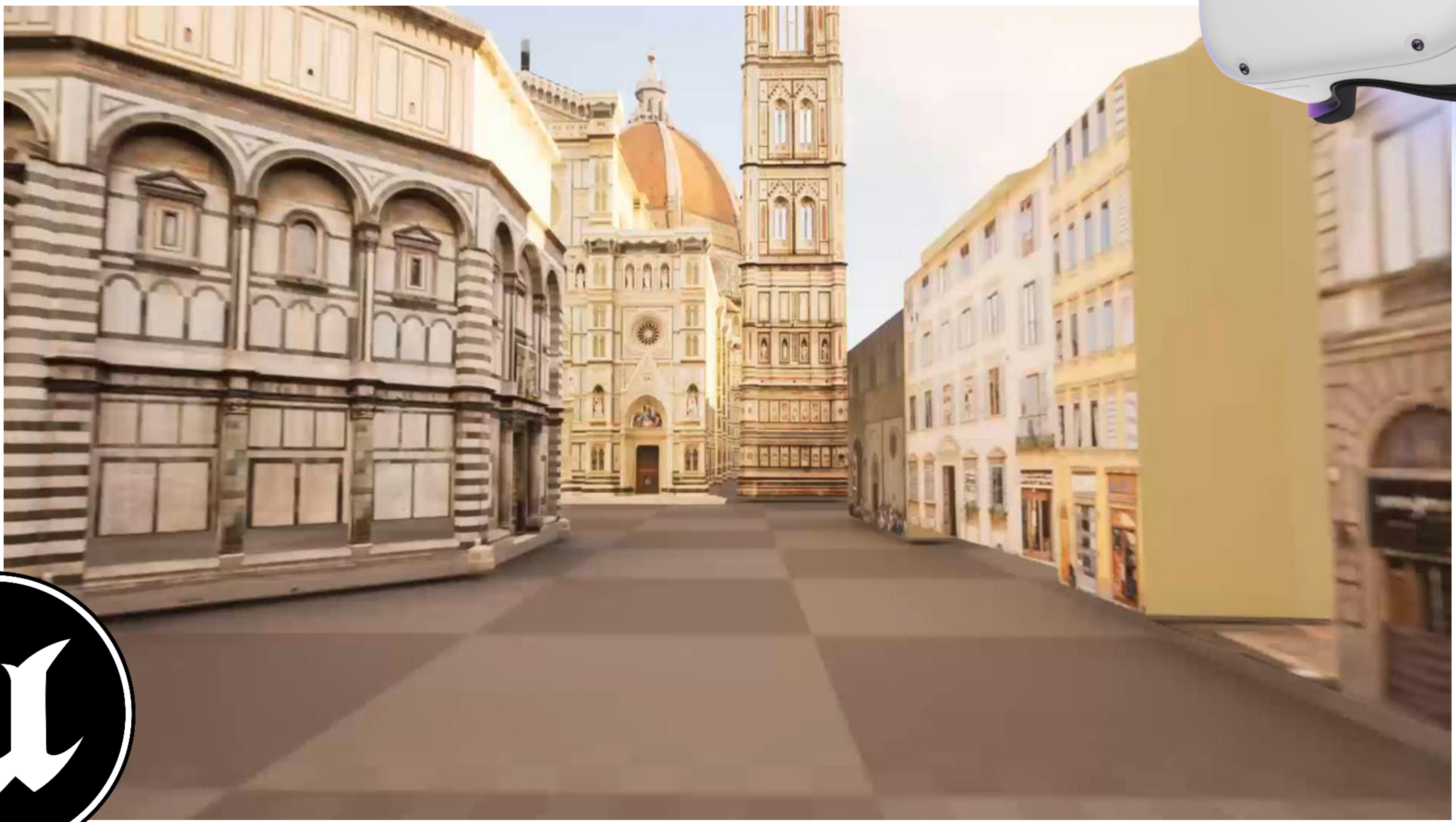


Digital Twin of Florence



https://digitaltwin.snap4city.org - https://www.youtube.com/watch?v=AWy7poKtTLc

Digital Twin of Florence



https://www.youtube.com/watch?v=Rcf_B2_GOio

References

- L. Adreani, P. Bellini, M. Fanfani, P. Nesi, G. Pantaleo, "**Smart City Digital Twin Framework for Real-Time Multi-Data Integration and Wide Public Distribution**", IEEE Access, 2024
- L. Adreani, P. Bellini, C. Colombo, M. Fanfani, P. Nesi, G. Pantaleo, R. Pisanu, "**Implementing Integrated Digital Twin Modelling and Representation into the Snap4City Platform for Smart City Solutions**", Multimedia Tools and Applications, Springer, 2023
- L. Adreani, P. Bellini, M. Fanfani, P. Nesi, G. Pantaleo. "**Design and Develop of a Smart City Digital Twin with 3D Representation and User Interface for What-If Analysis**". In proceedings 23rd International Conference on Computational Science and Its Applications (ICCSA 2023), 2023. (*Best paper award*)
- L. Adreani, P. Bellini, C. Colombo, M. Fanfani, P. Nesi, G. Pantaleo and R. Pisanu. "**Digital Twin Framework for Smart City Solutions**". In proceedings 28th International DMS Conference on Visualization and Visual Languages, DMSVIVA 2022, KSIR Virtual Conference Center, Pittsburgh, USA, 2022. (*Best paper award*)
- L. Adreani, C. Colombo, M. Fanfani, P. Nesi, G. Pantaleo and R. Pisanu, "**A Photorealistic 3D City Modeling Framework for Smart City Digital Twin**". In proceedings IEEE International Conference on Smart Computing (SMARTCOMP), 183-185, 2022
- L. Adreani, C. Colombo, M. Fanfani, P. Nesi, G. Pantaleo and R. Pisanu, "**Rendering 3D City for Smart City Digital Twin**". In proceedings IEEE International Conference on Smart Computing (SMARTCOMP), 183-185, 2022