# From Points to Prints

Monthly Presentation (2)

Alexandre Bry

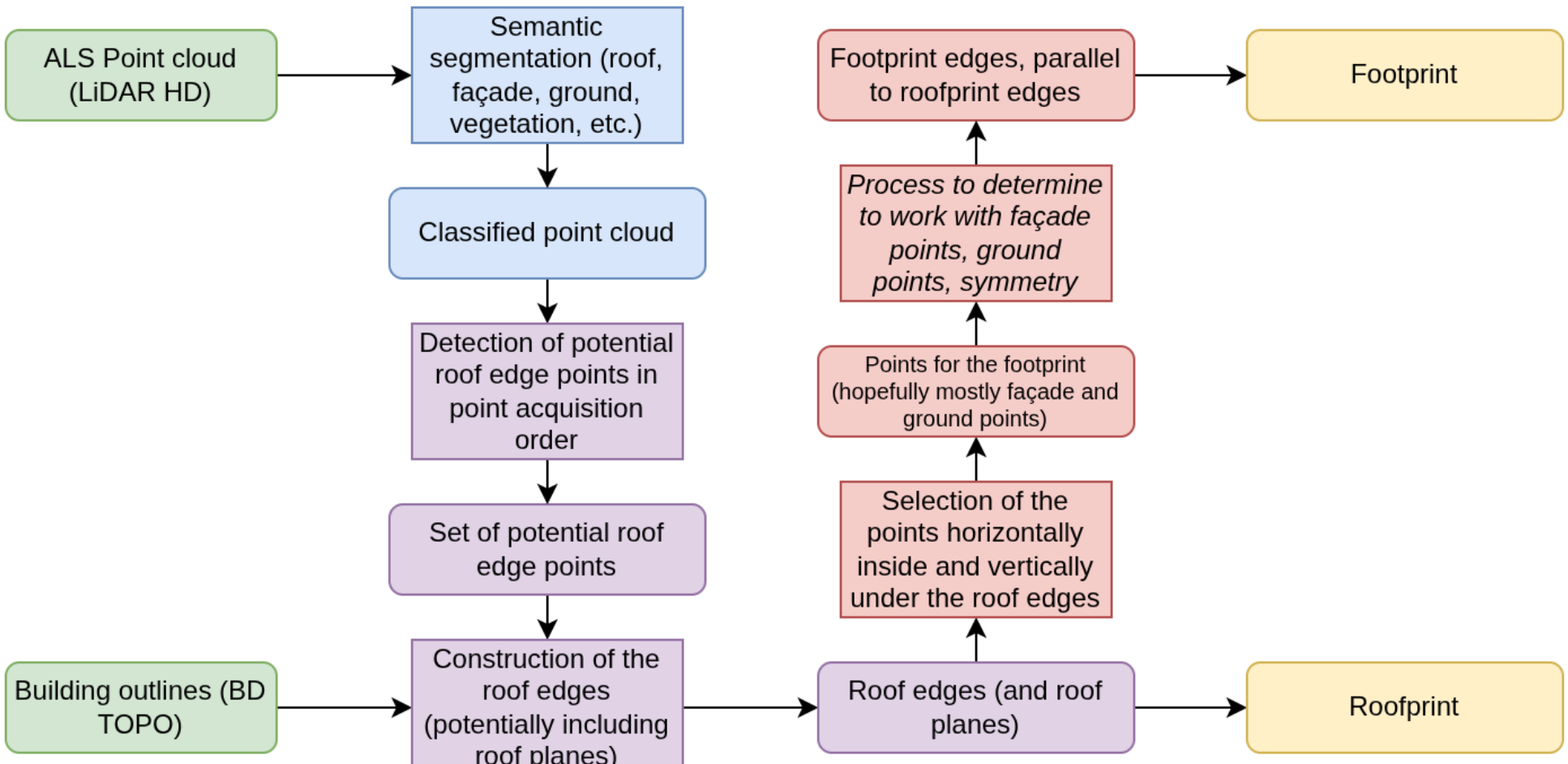
IGN, TU Delft

March 16, 2026

# Outline

# Context

# Planned pipeline



# Work done until now

# Point cloud topology

### Pulse

A single emission of the LiDAR sensor, which may result in **zero, one or more echoes** (points) depending on the number of surfaces the pulse hits.

### Scan line

A **set of pulses** emitted during one rotation of the LiDAR scanner.

### Flight strip

A **set of scan lines** collected along one pass of the aircraft over the ground.

# Extraction of the topology

- LiDAR HD: **cloud-optimized tiles** with **spatially ordered points**
- Topology must be extracted in multiple steps:
  1. Flight strips: using the **Point Source ID** field
  2. Scan lines: using the **GPS Time** field and the **Scan Direction Flag** field
  3. Pulses: using the **GPS Time** field

> The **Number of Returns** field is not reliable for pulses in the LiDAR HD dataset, as it is not updated when points are filtered out during the processing of the raw data.

# Extraction of the topology

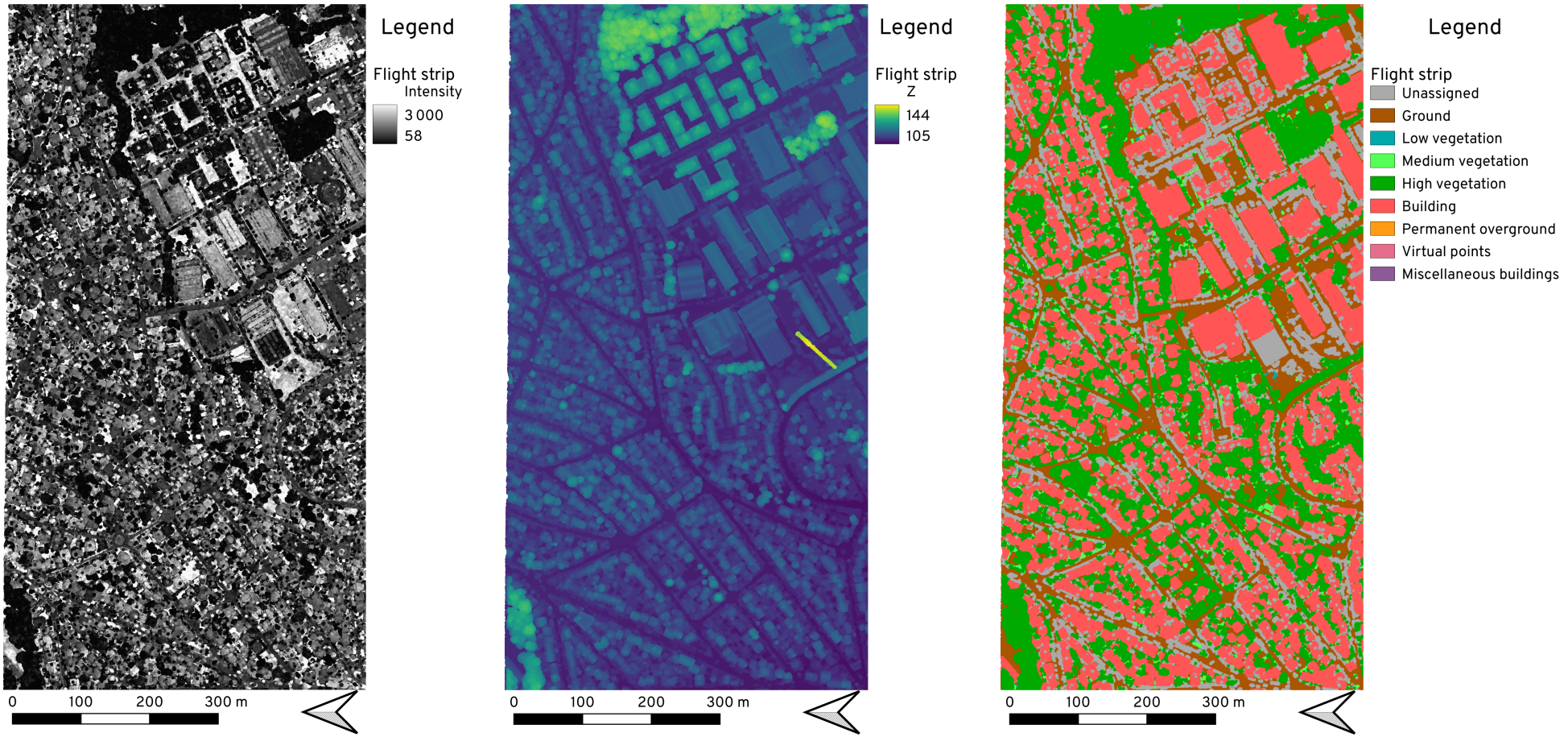



Figure 1: Visualization of the topology of the point cloud.

# Extraction of the topology

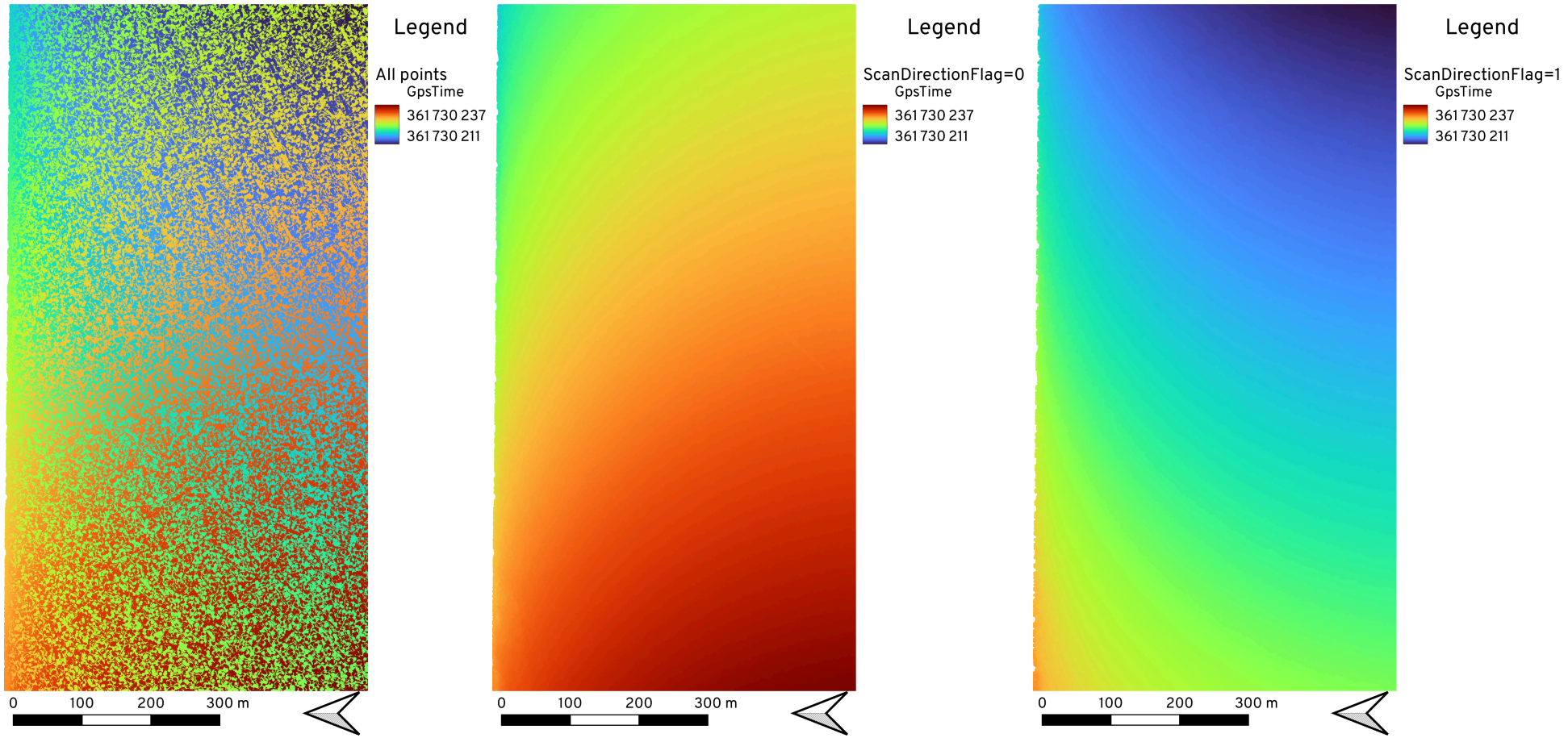



Figure 2: Visualization of the topology of the point cloud.

# Flight strips trajectories

- Trajectories computed using **multi-echo pulses** (code written by **Wu Teng**)
- Necessity to **reconcatenate the different parts of each flight strip** for better precision

> This method allows to use the trajectory **without relying on its availability**.

# Identification of edge-indicator points

## Multi-echo pulse

1. Find the **lowest point** in the same pulse
2. Compute the **height difference** ($\Delta h$) between the point and this lowest point
3. Set the point as a **edge-indicator point** if $\Delta h > (\Delta h)_{\min} = 2\ m$

## Single-echo pulse

1. Find the **lowest point** in the previous and next pulses
2. Compute the **height difference** ($\Delta h$) between the point and these lowest points and the **temporal difference** ($\Delta t$) between the pulses
3. Set the point as a **edge-indicator point** if

$$\frac{\Delta h}{\Delta t} > \frac{(\Delta h)_{\min}}{(\Delta t)_{\min}} = \frac{2}{10^{-6}} = 2.10^{6}\ ms^{-1}$$

for any of the two comparisons.

# Identification of edge-indicator points

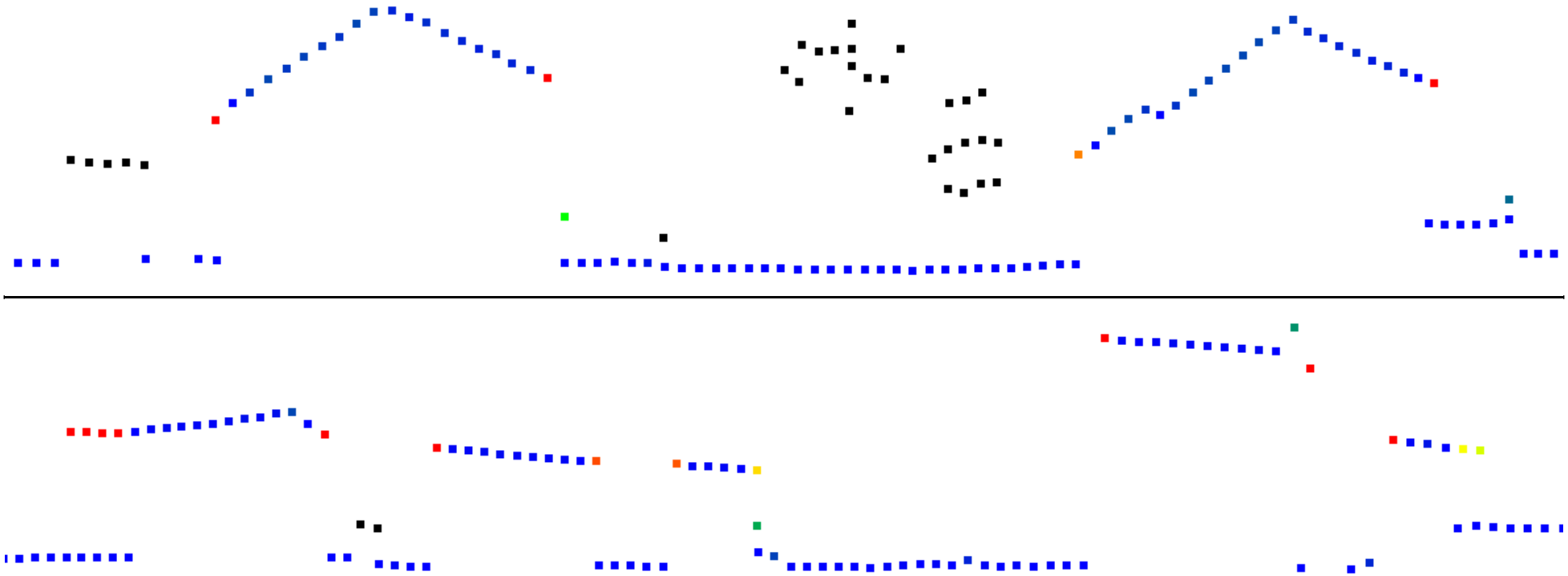


Figure 3: Two examples of the height differences obtained on a scan line. Black points are vegetation points, and for the other points the color represents the value of $\Delta h$, with blue-green-yellow-red from low to high values.

# Placement of edge points

## Multi-echo pulse

The edge-indicator point directly gives the position of the **edge point**.

# Placement of edge points

## Single-echo pulse

The edge-indicator point **needs to be refined** to get a better estimation of the position of the edge point.

### Case 1: Aligned points

If the 3 previous points $(p_1, p_2, p_3)$ in the same scan line (in the direction of the roof) are aligned with the edge-indicator point $(p_0)$, then the edge point $p_e$ is generated by **translating** $p_0$ with

$$p_e = p_0 + \frac{p_1 - p_0}{2}$$

### Case 2: Non-aligned points

Otherwise we use the **position of the scanner** $p_s$ at the time of the pulse to generate the half-way pulse direction between $p_0$ and the next point $p_g$ (supposed to be on the ground or on the façade) with

$$v_e = \frac{\text{normalize}(p_0 - p_s) + \text{normalize}(p_g - p_s)}{2}$$

Then we compute the edge point with

$$p_e = p_s + \text{norm}(p_0 - p_s) v_e$$

# Weighing of edge points

Edge points are weighted using:

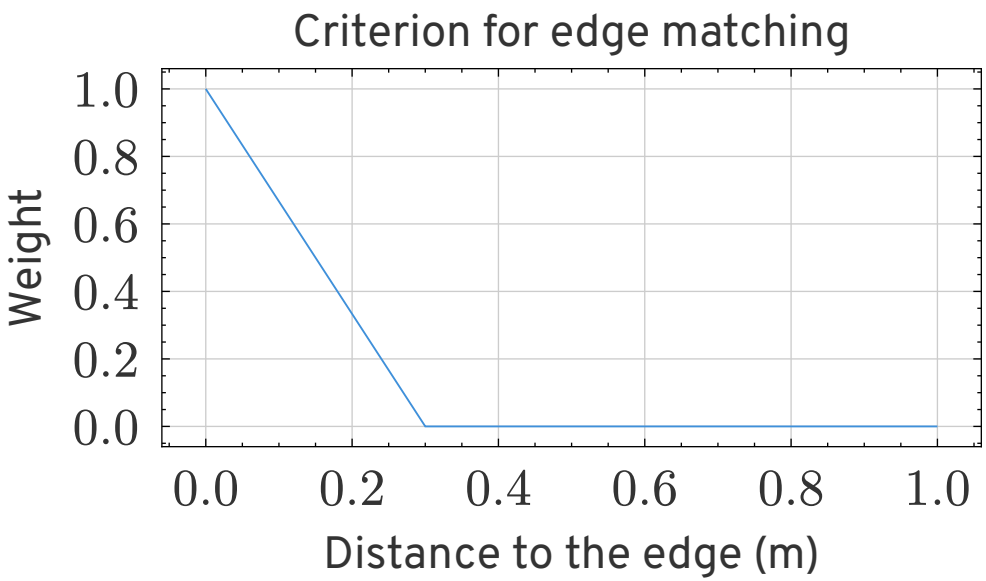
1. Their **height** (normalized over the area of the building) to a factor $w_h \in [1.0, 3.0]$
2. Their **origin** to a factor $w_o \in \{0.1, 0.5, 1.0\}$:
   - Multi-echo: 1.0
   - Single echo with good estimation of position: 0.5
   - Single echo with less precise estimation of position: 0.1
3. Their **classification** to a factor $w_c \in \{1.0, 2.0\}$:
   - Building: 2.0
   - Other: 1.0

These values were picked arbitrarily and would need to be optimized. The final weight $w$ is given by:

$$w = w_h \times w_o \times w_c$$

# Displacement of BD TOPO edges

The value of a point is not only determined by its weight, but also by its **distance to the edge**. Distances are computed **in 2D** after getting rid of the vertical component of the position.




This distance is evaluated by projecting the point onto the edge, keeping only the points that project on the segment.

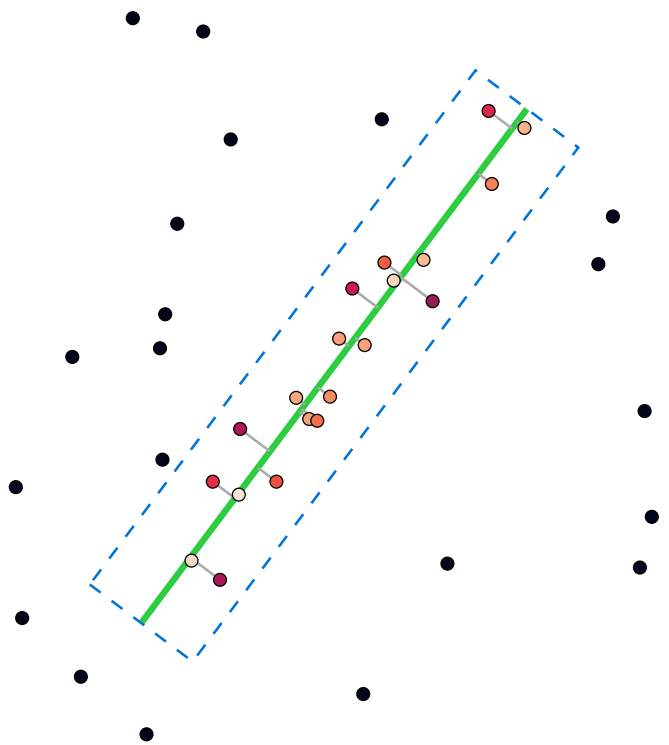
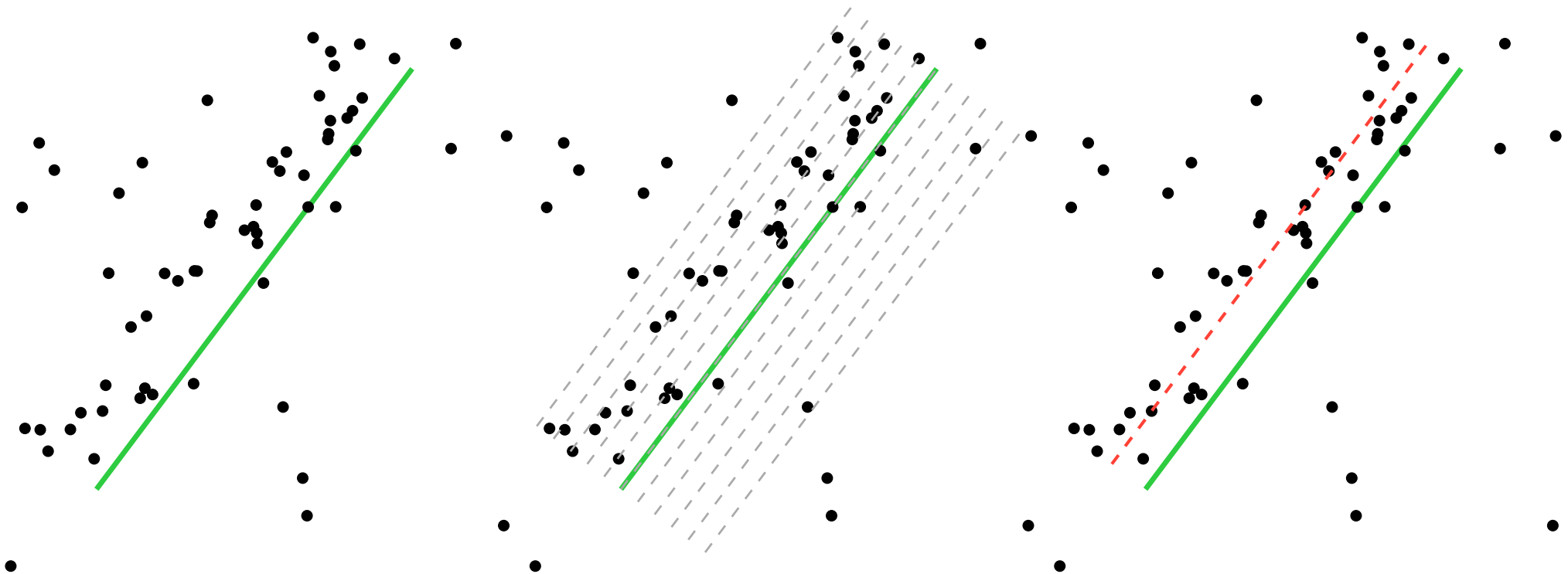


Figure 5: Illustration of the criterion that scores edges.

# Displacement of BD TOPO edges



(a) Step 0 (b) Step 1 (c) Step 2

Figure 6: Process of finding a better edge position.

# Preliminary results

# Identification of edge points

Figure 7: Edge points in blue and LiDAR HD data in gray scale from low (black) to high (white) intensity.

# Identification of edge points

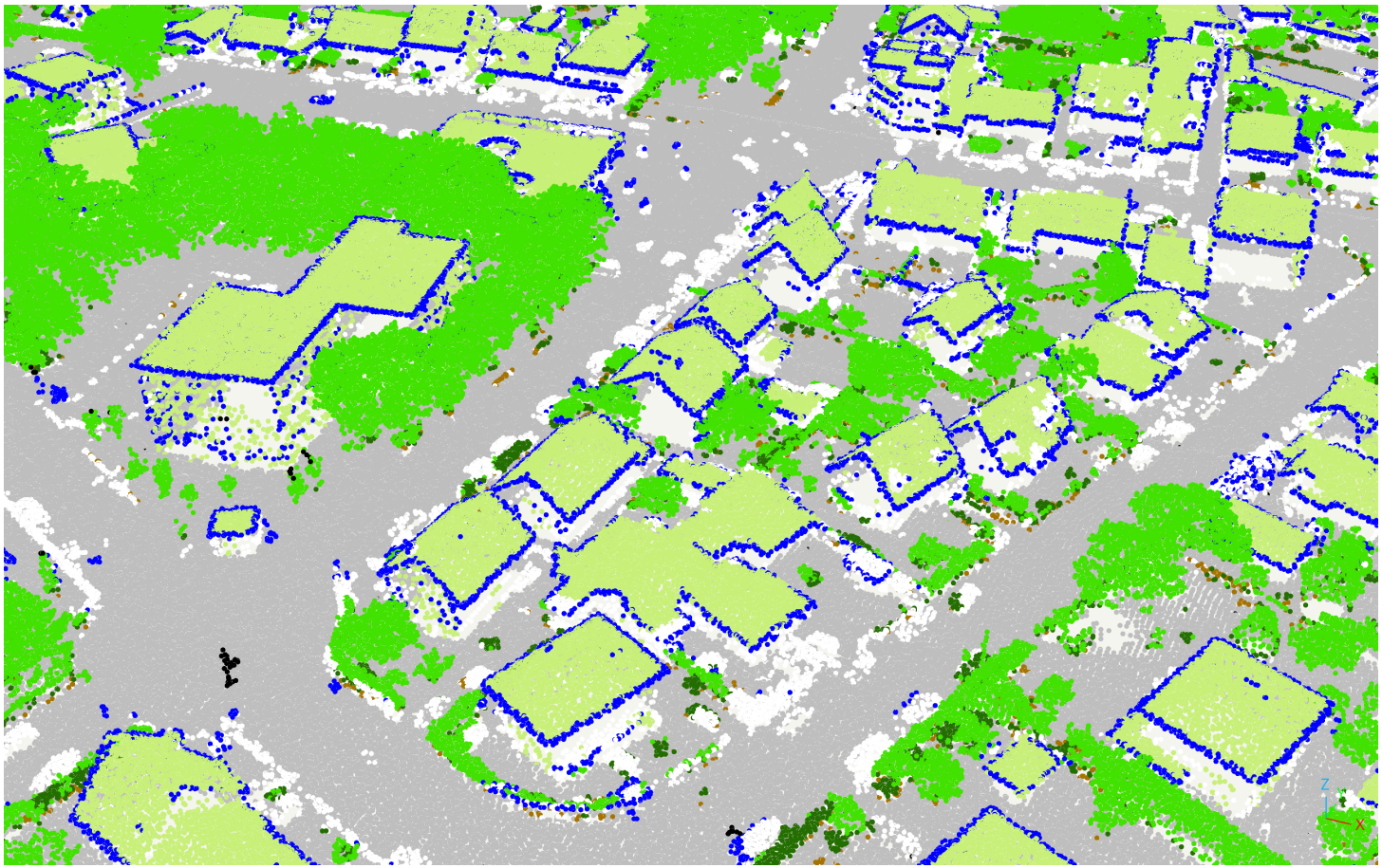


Figure 8: Edge points in blue and LiDAR HD data coloured by classification.

# Identification of edge points

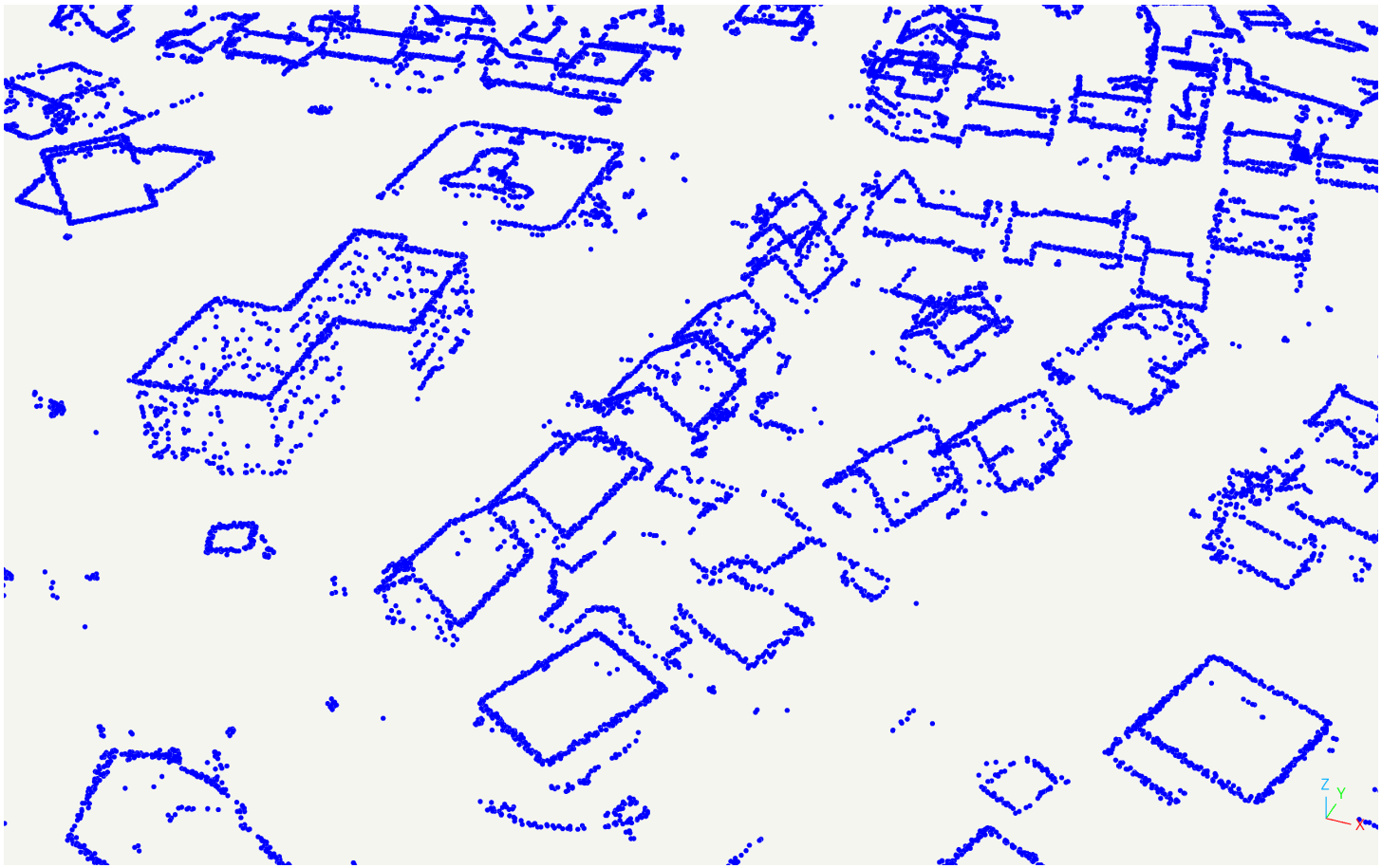



Figure 9: Edge points.

# Identification of edge points

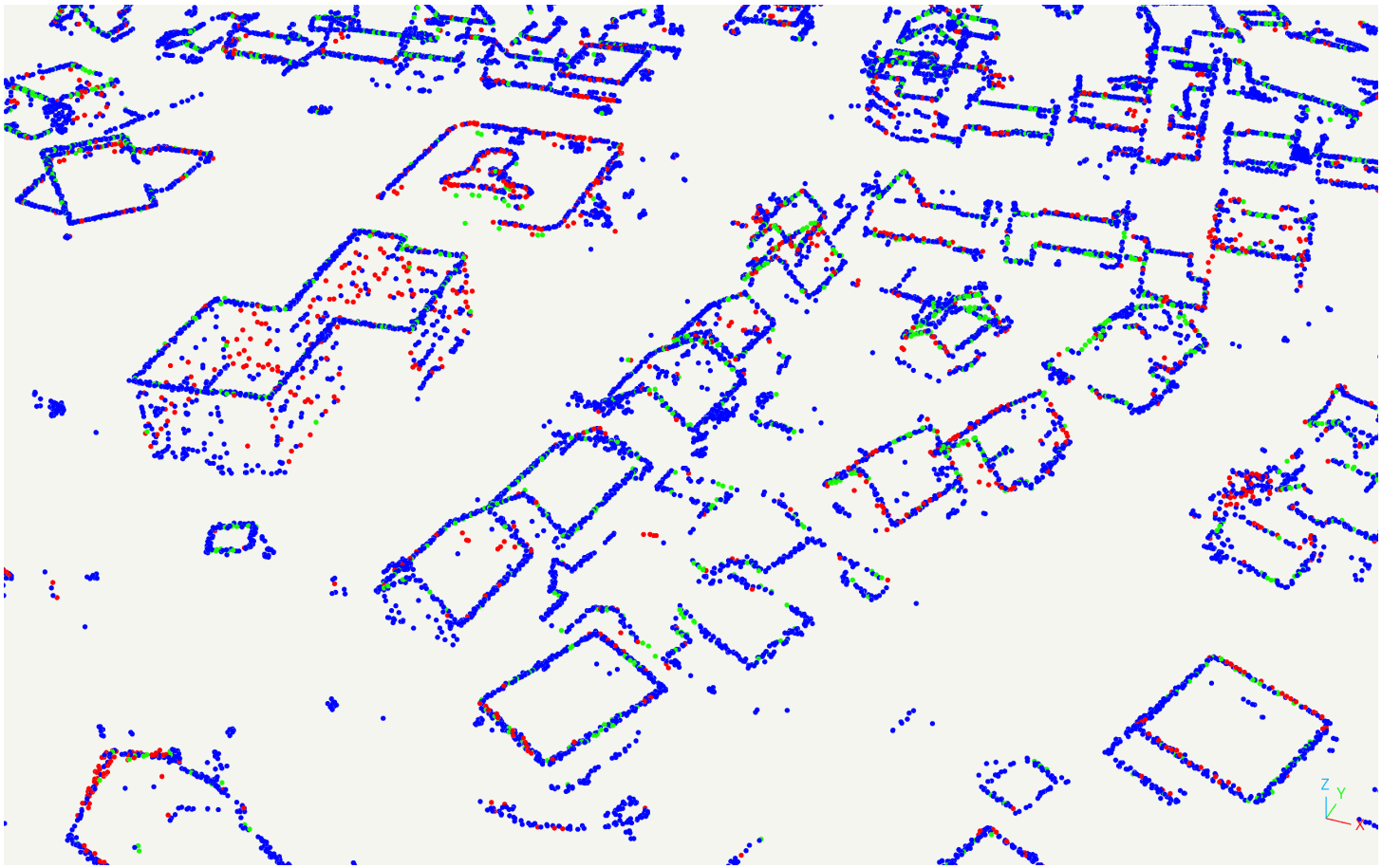
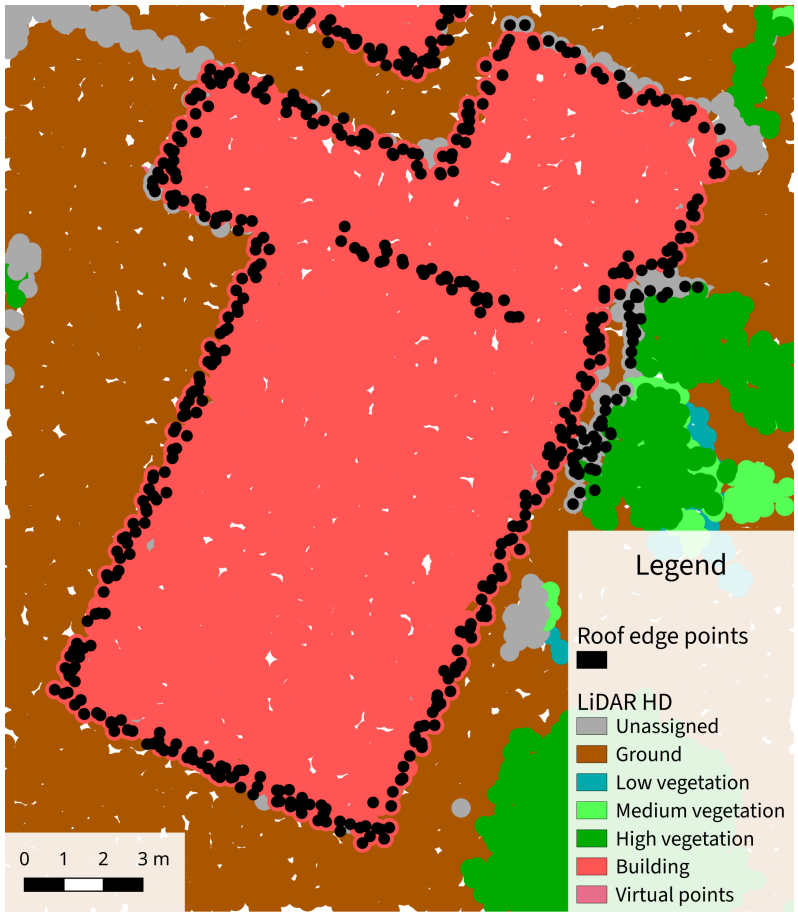



Figure 10: Edge points coloured per type (blue: real point, green: generated by translation, red: generated with trajectory).

# Computation of roofprints from BD TOPO



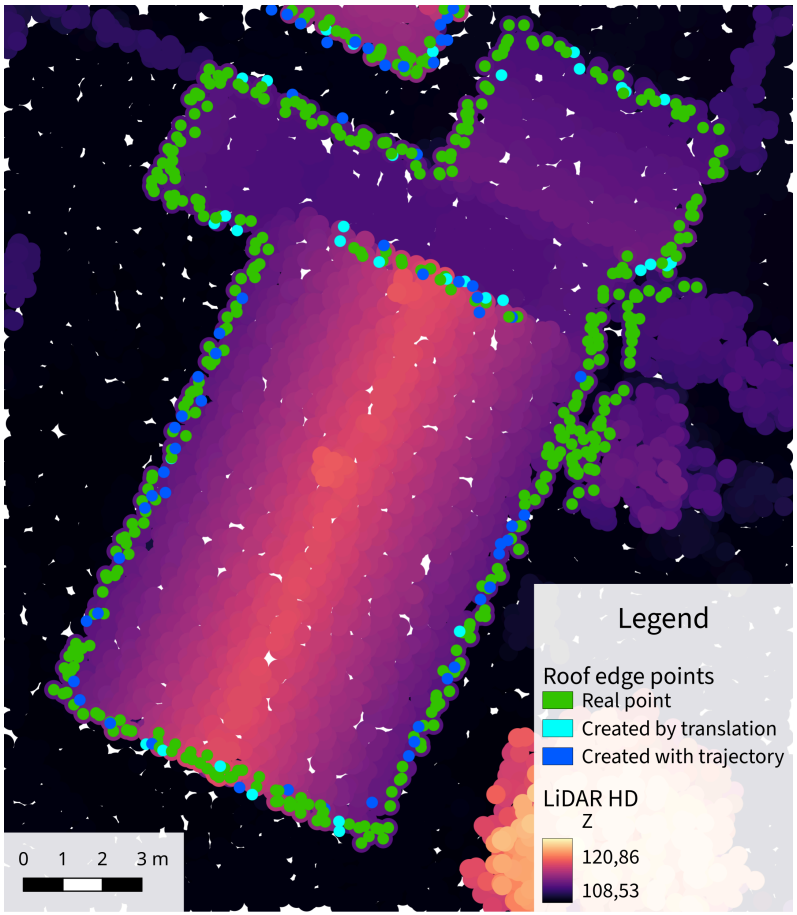




Figure 11: The potential roof edge points.

# Computation of roofprints from BD TOPO

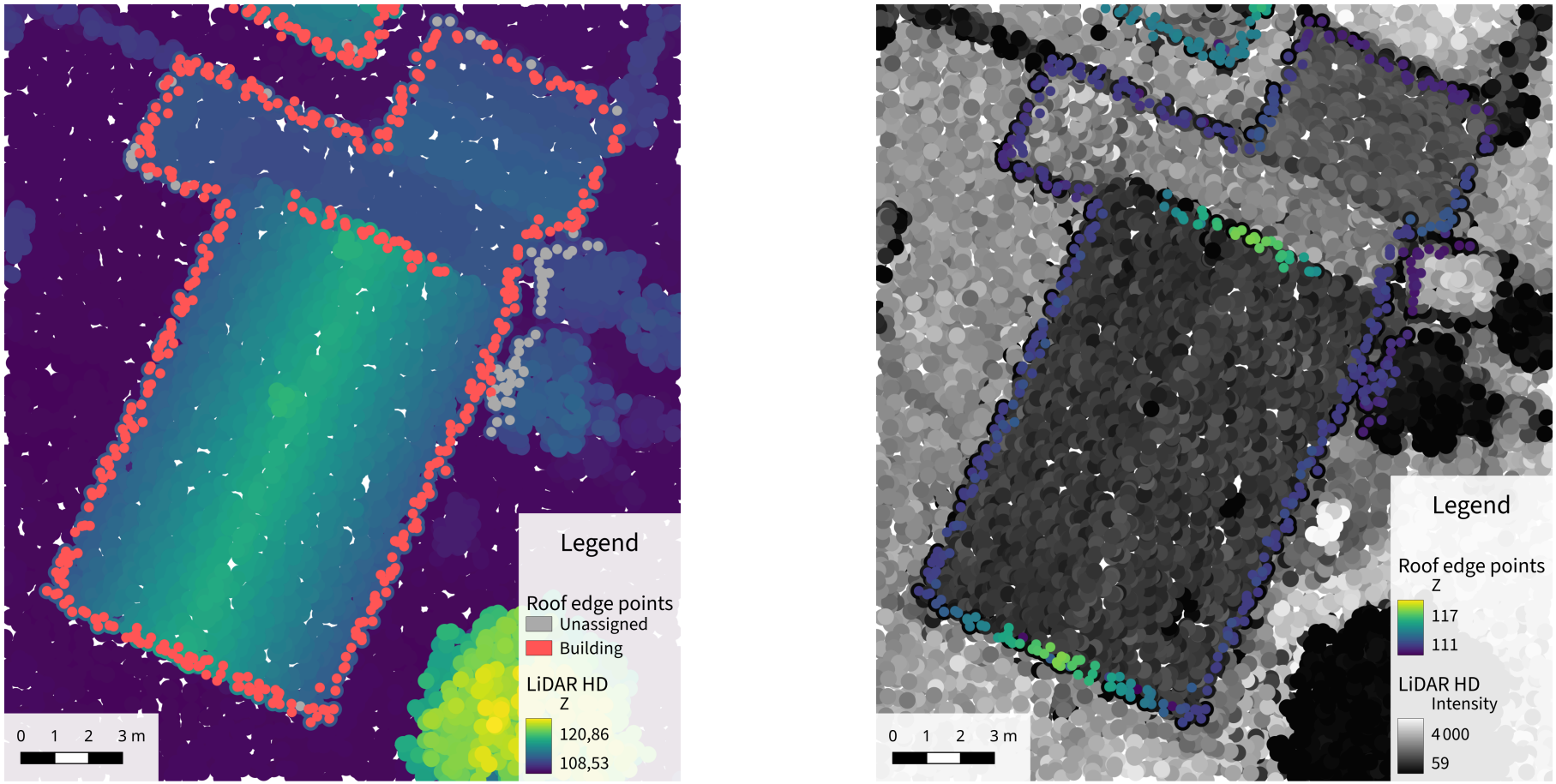



Figure 12: The potential roof edge points.

# Computation of roofprints from BD TOPO

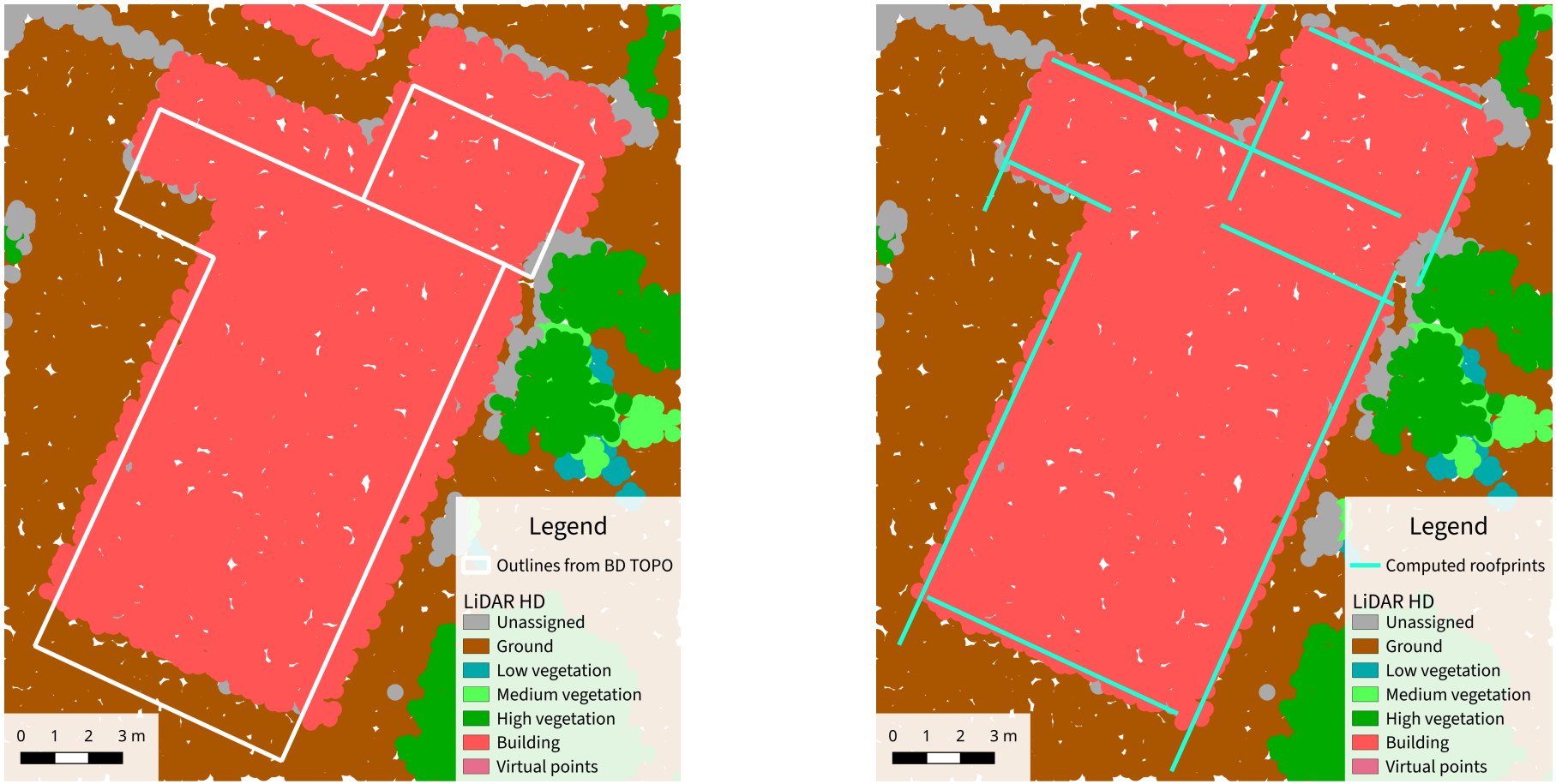



Figure 13: Comparison of BD TOPO outlines and computed roofprints.

# Computation of roofprints from BD TOPO

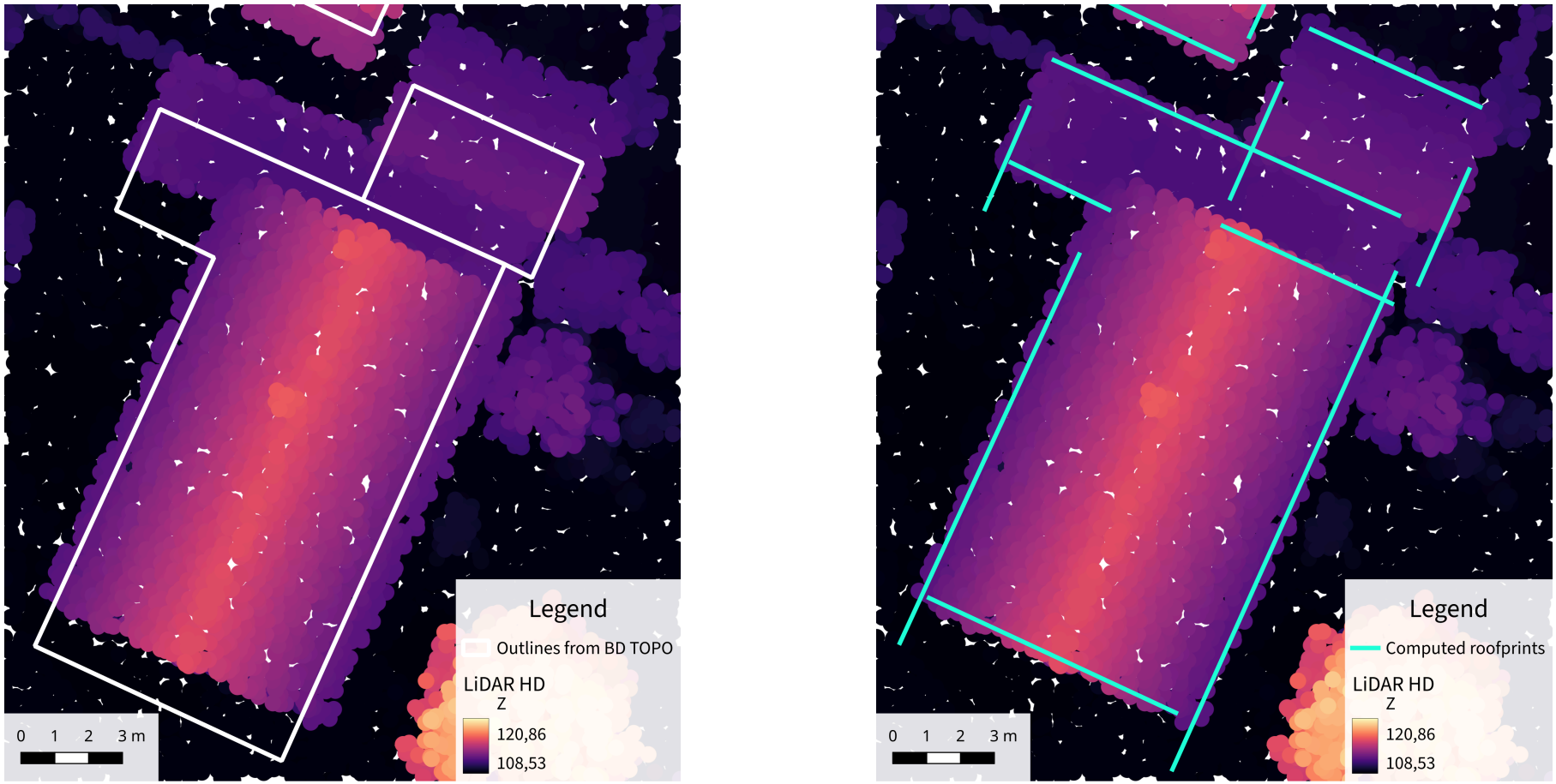



Figure 14: Comparison of BD TOPO outlines and computed roofprints.

# Next objectives

# Edge points

- Identify cases with a **dense set of points on a façade**. Possibility to compute normals to identify these cases.
- Consider **neighbours in other directions** than scan order (such as perpendicularly to it).
- Identify **lines in the point cloud topology** with Wu Teng's method:
  - Could be used to estimate the **normals locally** in cases where segments are found to estimate if points are on façades or on roofs.
  - Could be used to identify the breaks of roof planes (out of scope for now).

# Edge matching - Improvements

## For each edge

- **Group edges** with angles **close to 180°** (instead of currently merging them).
- Match each edge by also moving and computing the **score of its two (or more) neighbours**.
- Use the **repartition of points** on the edge in their score.
- Try (maybe) to use the **distance to the current edge** as an indication as well.

# Edge matching - Improvements

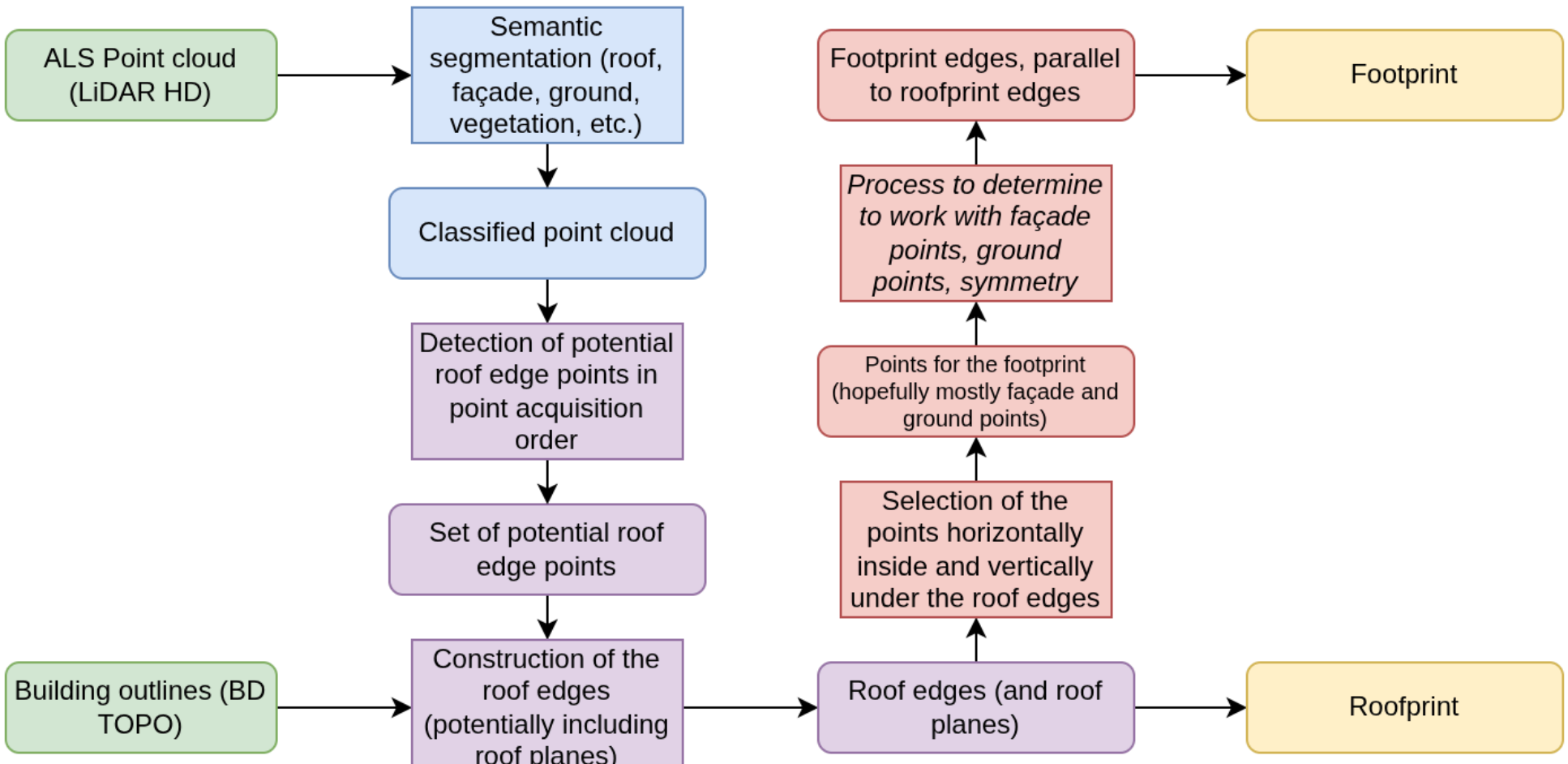
## For the whole building

- Matching **longer edges first**.
- Run the matching **algorithm multiple times in a row**, or until it converges.
- Use **combinatorial optimization** to pick the best set of edges with multiple solutions for difficult edges.

# Edge matching - Other ideas

- Use **regularization** before or after.
- Use the **classification** of the points in combination with the outlines to crop the point clouds better.
- Try to make better edges:
  - With a **RANSAC-like** approach (potentially using the BD TOPO to guide the process) in **2D or 3D**.
  - By other means in 3D?
  - By **recomputing the edge** after identifying the inlier points with the current process.
  - Try to estimate **where the edge starts and stops** from the point cloud.

# Rest of the pipeline



# Thank you for your attention!

Any questions?

alexandre.bry@ign.fr