Kartverket

SENBYGG project

Mathilde Waymel - 01.06.2024

Introduction

The Senbygg project aims to detect changes in buildings using data from the Sentinel-1 satellite. Sentinel-1, a radar satellite, provides information on height variations by comparing images taken at different times. By comparing median images from consecutive years, the code can identify the construction or demolition of buildings and generate annual change maps.

The original project was funded by the Framework Partnership Agreement on Copernicus User Uptake (FPCUP) and the Norwegian Space Agency (NOSA). The action was coordinated by NOSA with the Norwegian Mapping Authority (NMA) as an implementing partner and NORCE as a sub-contractor to the NMA. Several Python code files were developed during the initial project, which Torgeir Ferdinand Klingenberg later adapted for our specific case study, enhancing their usability.

This new Senbygg study aimed to adapt the code for the specific area of study, update calls to external resources if necessary, and test the accuracy of the code by comparing it to field data. It was also planned to explore other potential detection methods if any appeared promising and to test their accuracy.

Updating Senbygg original code

The first step was to adapt the original Senbygg code. It consists of four scripts: the first downloads images from the internet, the second formats the downloaded data for further processing, the third detects changes for each year and outputs the results as a raster image of the changes, and finally, an optional fourth script vectorizes the data to produce a result with building areas represented as polygons.

Despite the well-written instructions provided by Torgeir Ferdinand Klingenberg, adapting the code and obtaining initial results proved to be quite challenging for me. Indeed, certain steps require the use of Docker software (although not feasible for all steps), involving the installation and modification of an existing image each time. Additionally, during this period, the image download routine stopped working, and I did not have the time to explore alternative methods for downloading the images.

I had still managed to test this code on a few areas, and my impressions and conclusions are detailed in the Comparison with original Senbygg python code section. Unable to resolve this issue and obtain results for the study areas, I thus decided to test another promising method: the Google Earth Engine online platform.

New approach with Google Earth Engine

New Senbygg code

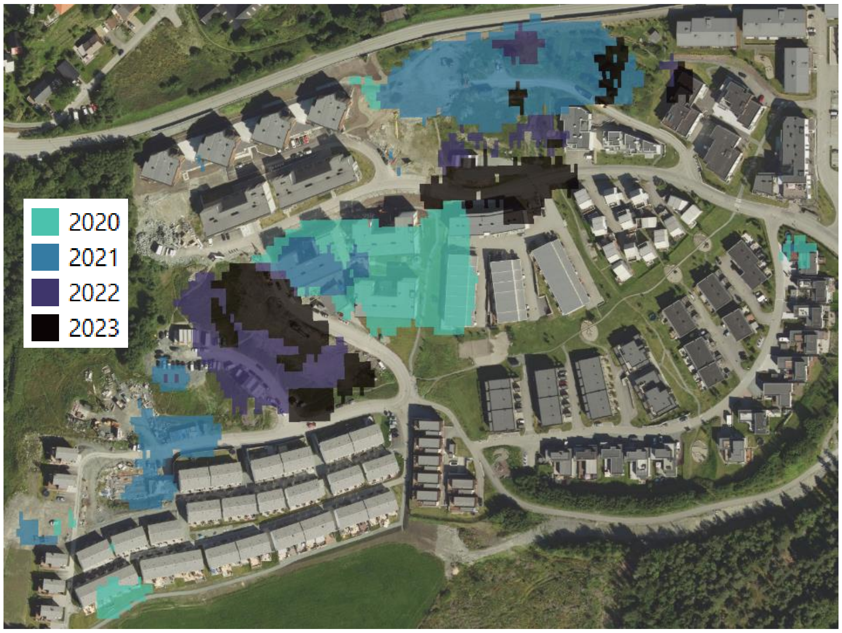
Google Earth Engine (GEE) is an online computing platform that enables easy access to freely available satellite data and provides very limited computing time. Therefore, I wrote a new code for the Senbygg project on the GEE platform, using the same technique as the original Python scripts. However, many steps are immediate on this platform, resulting in a much simplified code. In fact, there is only one script for all the steps.

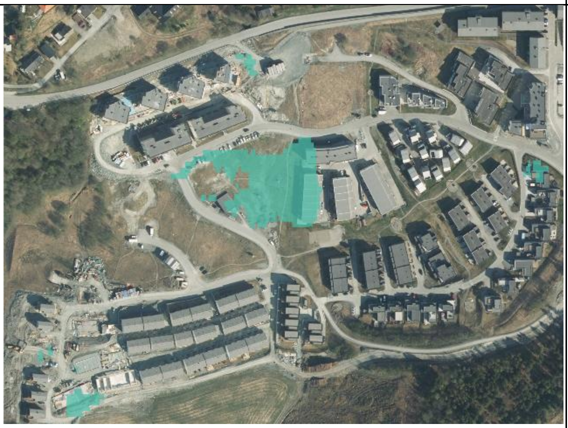
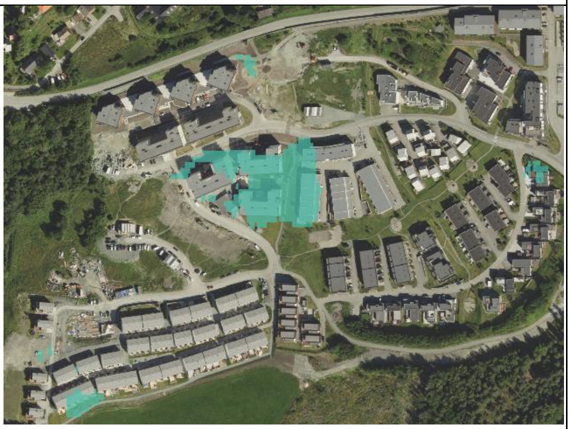
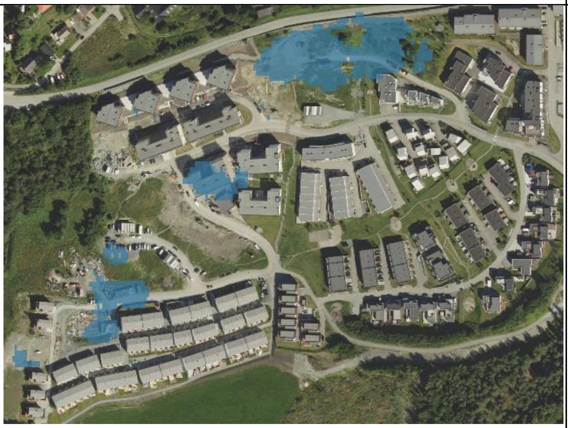
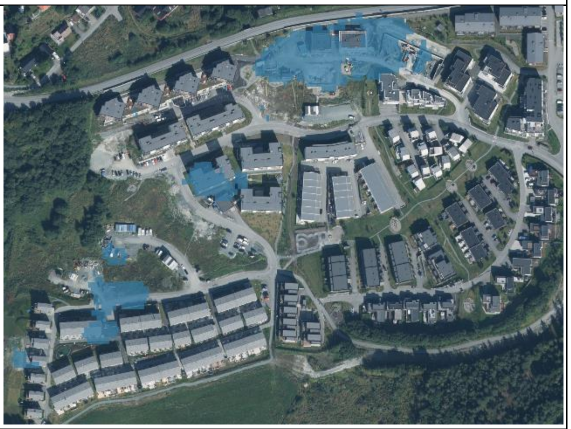
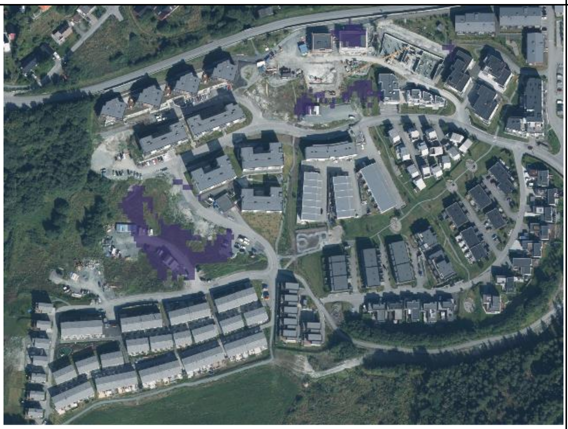
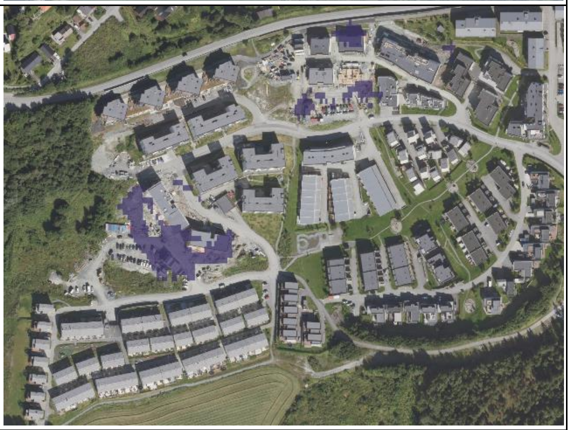
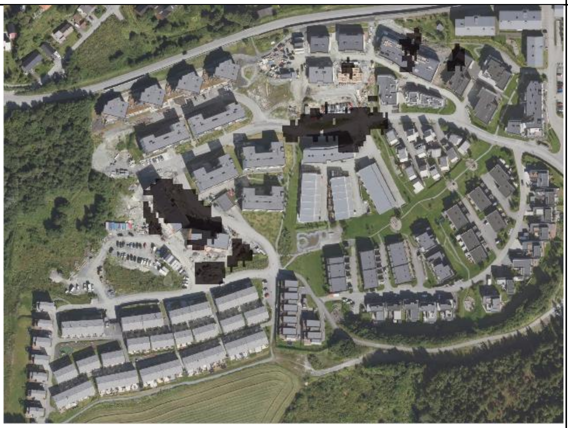
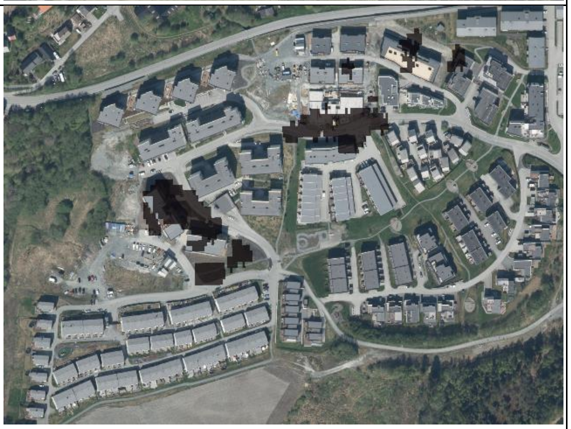
Here are some examples of resulting simplifications: there is no need to provide a digital elevation model (DEM), vegetation mask, or water mask. Indeed, access to Sentinel-2 satellite data allows for the easy reconstruction of these two masks, and the platform manages the projection and overlay of images automatically. Furthermore, there is no need to manually conduct a preliminary study on a website indicating available orbits for the area and then adapt the scripts accordingly. The code retrieves all available images, determines the orbits used, and groups the images by orbits to simultaneously calculate detections for each. Finally, all data formatting steps for the subsequent script are not necessary.

Results

The obtained results appear very promising. The algorithm successfully detects building changes for each year. However, I didn't have time to conduct a thorough comparison between the algorithm's detections and the ground truth data from the cadaster. This step is a bit challenging because the date of declarations in the cadaster does not exactly correspond to the date of completion of construction works, which can make the ground truth data unreliable. Nevertheless, I was able to visually assess the results by displaying the orthophotos for each year and verifying that the detected changes are accurate and that no changes were missed.

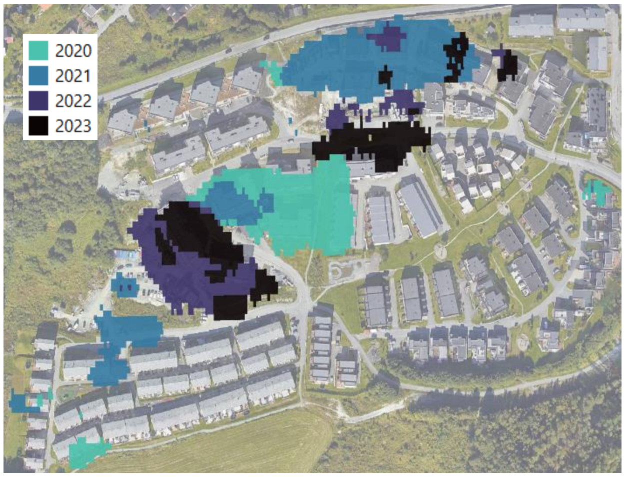
Here's an example of a construction area in Heimdal (Trøndelag). The areas with various shades of blue indicate changes per year. The comparison with the annual orthophotos is shown just below.

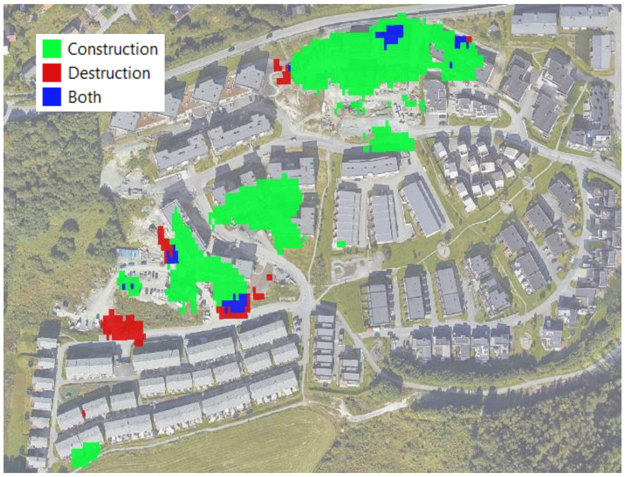


Year	Ortophoto before change	Ortophoto after change
2019 - 2020		
2020 - 2021		
2021 - 2022		
2022 - 2023		

The table comparing the detections with the orthophotos highlights that the results are very good, despite some flaws. It appears, for example, that a few buildings were either incompletely detected or not detected at all, and that the positioning of the detections is sometimes quite offset. However, there seem to be few false detections. Nevertheless, caution is needed with this comparison because we lack information on the date of the orthophotos, which could disrupt our analysis if images from two consecutive years were taken very close together in time.

The algorithm also calculates an image indicating the type of change (see the image below on the right). This allows us to determine whether the change is a construction, a destruction, or both. It can be noticed that the change image by year and the change type image do not completely overlap. I believe this difference is due to the mathematical formulas used, and they should be reviewed.

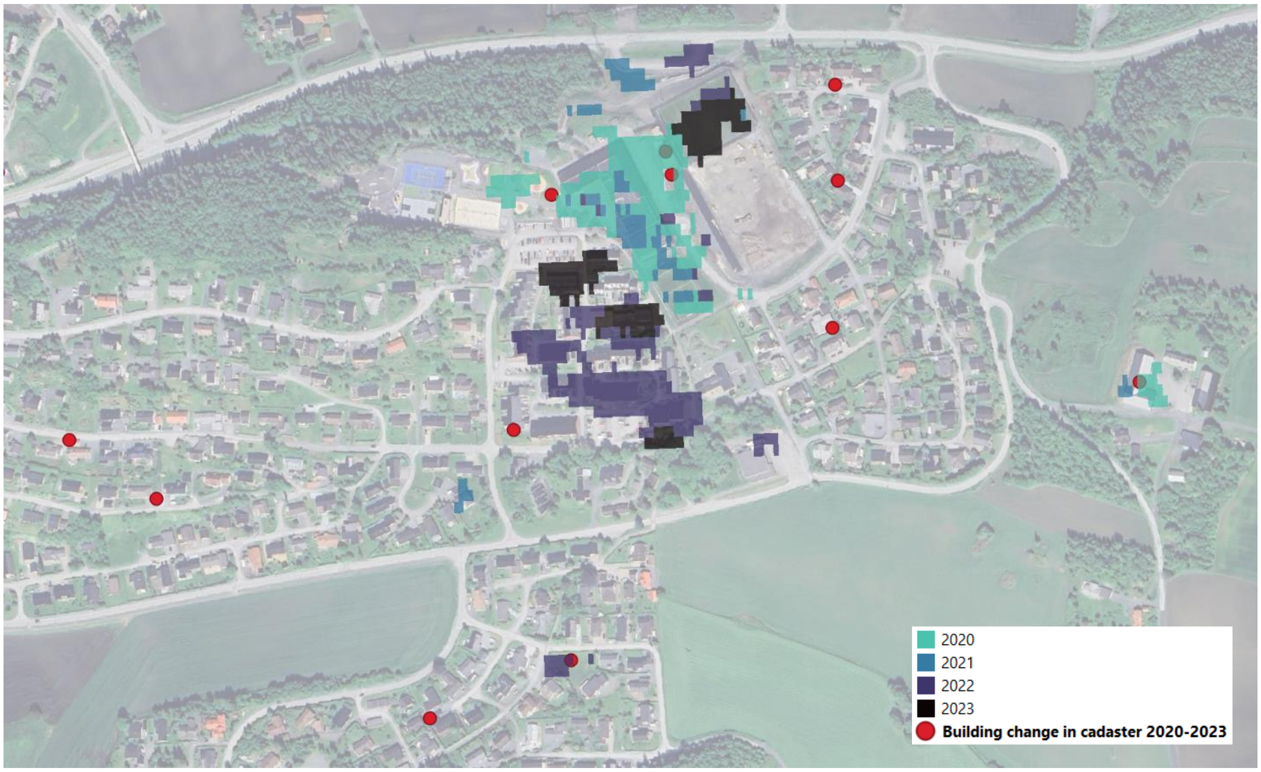


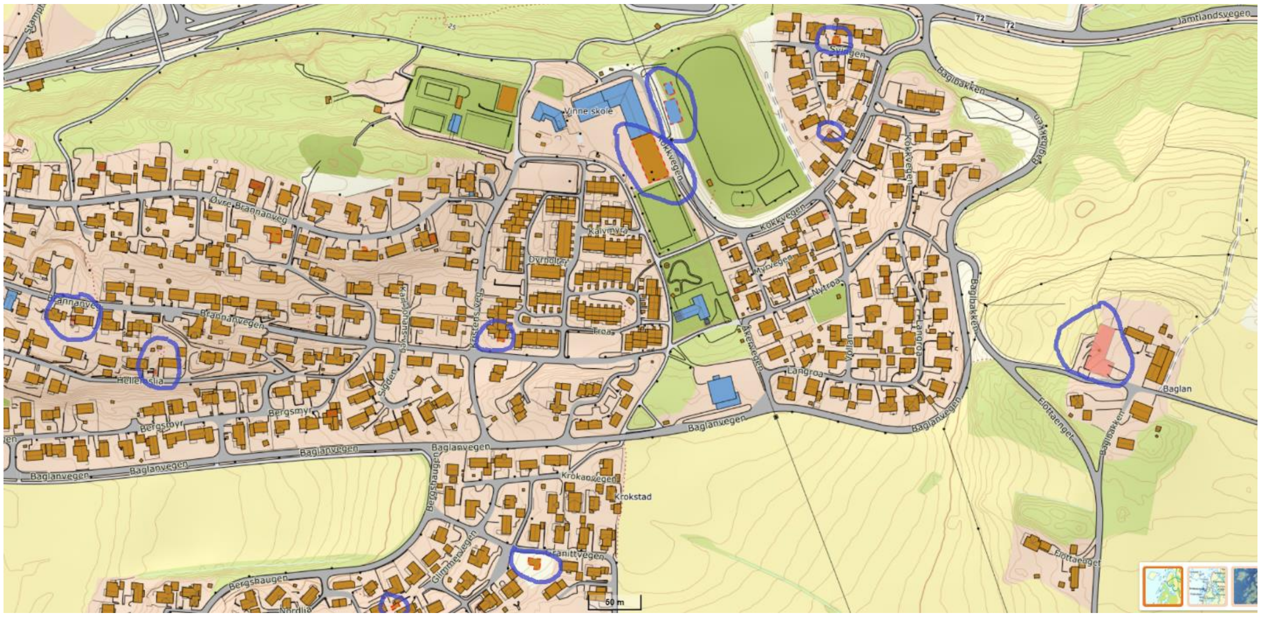


Overview of ground truth comparison

The first image below shows an example of detection in Verdalsøra (Trøndelag). The shades of blue and green represent detected change areas, while the red dots indicate buildings declared in the cadastre as having changed during the detection period. It appears that despite the good results of the visual analysis, the detections do not align at all with the field data, and many detections do not correspond to declared areas. This observation is quite surprising but also raises questions about the cadastre data.

The second image below shows the cadastral map with areas undergoing changes highlighted in blue (visible through red dashed lines outlining the buildings). We can observe that these areas correspond perfectly to the change points in the image above. However, we also notice that the undetected changes from the first image correspond to very minor alterations (such as adding an entrance porch or a small garage, for example). These observations allow us to explain the difference between the ground truth data and the detections: some changes are too small to be detected. I believe that before proceeding with the comparison to cadastral data, we need to find a way to select only ground truth data corresponding to significant changes, and then we can obtain reliable results. The codes for building status in the cadastral data are provided in Appendix I.

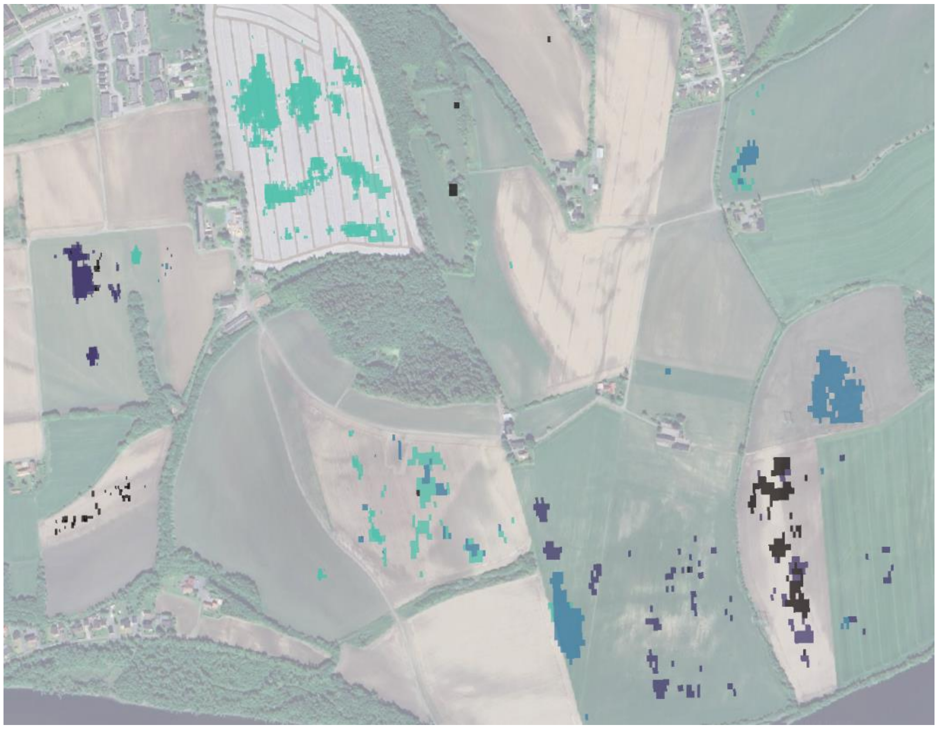




Perspectives of improvements

The Senbygg code for GEE already provides very good results, but improvements are possible.

- Firstly, I have observed some false detections on fields (see image below), so the vegetation filter thresholds need to be adjusted to address this issue.



- I also didn't have time to implement the 'change confidence' feature, which indicates the reliability of each detection.
- Furthermore, there are discrepancies between the 'change year' detection image and the 'change type' image (indicating appearance, disappearance, or both); I believe the mathematical formula needs to be reviewed.
- Lastly, there's a bug preventing the generation of change maps before 2018. I believe this issue is due to changes in the satellite orbits during that period, resulting in some orbits lacking images for certain years, which creates problems due to the missing data. The code should be adapted to handle these years without data.

In first conclusion, while the code on GEE is not yet fully complete, it already shows very promising results and offers many advantages over the original method.

Comparison with original Senbygg python code

Although both methods were developed for the same project and use the same physical technics, the original Python code and the GEE method present significant differences. First, I will list the disadvantages of the original method that the GEE method addresses, followed by the disadvantages of the GEE method.

Here is a list of the advantages of the GEE method that I have identified:

- **Reduced Implementation Time:** The most significant advantage of the new method is the total implementation time. The pure computation time is already much faster on GEE because it is a distributed computing platform. Generating and exporting the results takes only about 5 minutes for a 3km x 3km area with a 5m pixel size. Additionally, there is only one script to run on GEE. However, most of the time saved is due to not needing to download satellite images, a step that takes several hours and is extremely tedious. Thus, the GEE method offers a significant time-saving benefit.

- **Storage Space Savings:** As mentioned earlier, not having to download images also represents a substantial storage space savings.
- **Scalability:** Thanks to the time savings mentioned earlier, it is feasible to use the GEE method at the scale of a kommune and even a fylke. This point, along with computation time, was a major disadvantage of the original method. When it takes a day of script preparation and computation for a small area (2km x 2km), how can one envisage generalizing the use of this technique on a national scale?
- **Simplified File Management:** With the GEE method, there is no need for a complex file structure, which is laborious for updating file paths in the scripts.

However, the current GEE method has some disadvantages:

- **Dependence on Google:** Although GEE is free, it is a Google service. This raises several concerns, such as long-term reliability, legal issues related to data dissemination, and potential data usage by Google. Additionally, a Google account is required to use the platform.
- **Programming Language:** The code is written in JavaScript, which might seem a bit difficult to access at first glance. However, it is quite easy to get the hang of, and the GEE documentation (https://developers.google.com/earth-engine/guides) is very well done and provides many examples.
- **Result Download Complexity:** Practically, while the drawbacks are much less annoying than the implementation of the original method, the process of downloading results is not as immediate as one might expect. The simplest way to save results is to export them to a Google Drive account and then download them from there.
- **Data Upload Requirement:** Another minor inconvenience is that necessary data (study area and potentially ground truth data) must be uploaded individually via the platform's assets.
- **Simplification in Current GEE Code:** Due to my imperfect understanding of the original code's computational details, the GEE version I wrote is likely somewhat simplified, especially in terms of data corrections (e.g. DEM). The entire mathematical part should be thoroughly reviewed.

Despite these disadvantages, the GEE method offers substantial benefits over the original method, particularly in terms of time efficiency and scalability.

However, it is necessary to note that beyond the functional disadvantages of the methods, the accuracy of the results produced by each must be considered. A rigorous comparison with field data has not been conducted for either method, and the conclusions of this analysis are a crucial factor in choosing which method to use. If the results of the two methods prove to be comparable, then I would recommend continuing the development of the GEE method.

Conclusion

To conclude, I would first like to share my opinion on the original Python code for the project. The code is functional, but I find several flaws that make it difficult to use. Getting started and adapting the code is quite time-consuming, mainly due to the use of multiple different software tools. However, the major drawback is the total time required to obtain results. Specifically, the process of saving large satellite images locally is excessively long and tedious. For me, this flaw is prohibitive: it seems impossible to use this method on a scale larger than a few square kilometres, making its generalization at the city or fylke level unfeasible.

In contrast, the GEE method, despite its few drawbacks, offers promising prospects for the continuation of the project. It addresses the major flaws of the original method and appears to produce equally good results. I strongly recommend the development of this method because I believe it is better suited to meet the new challenges of this project.

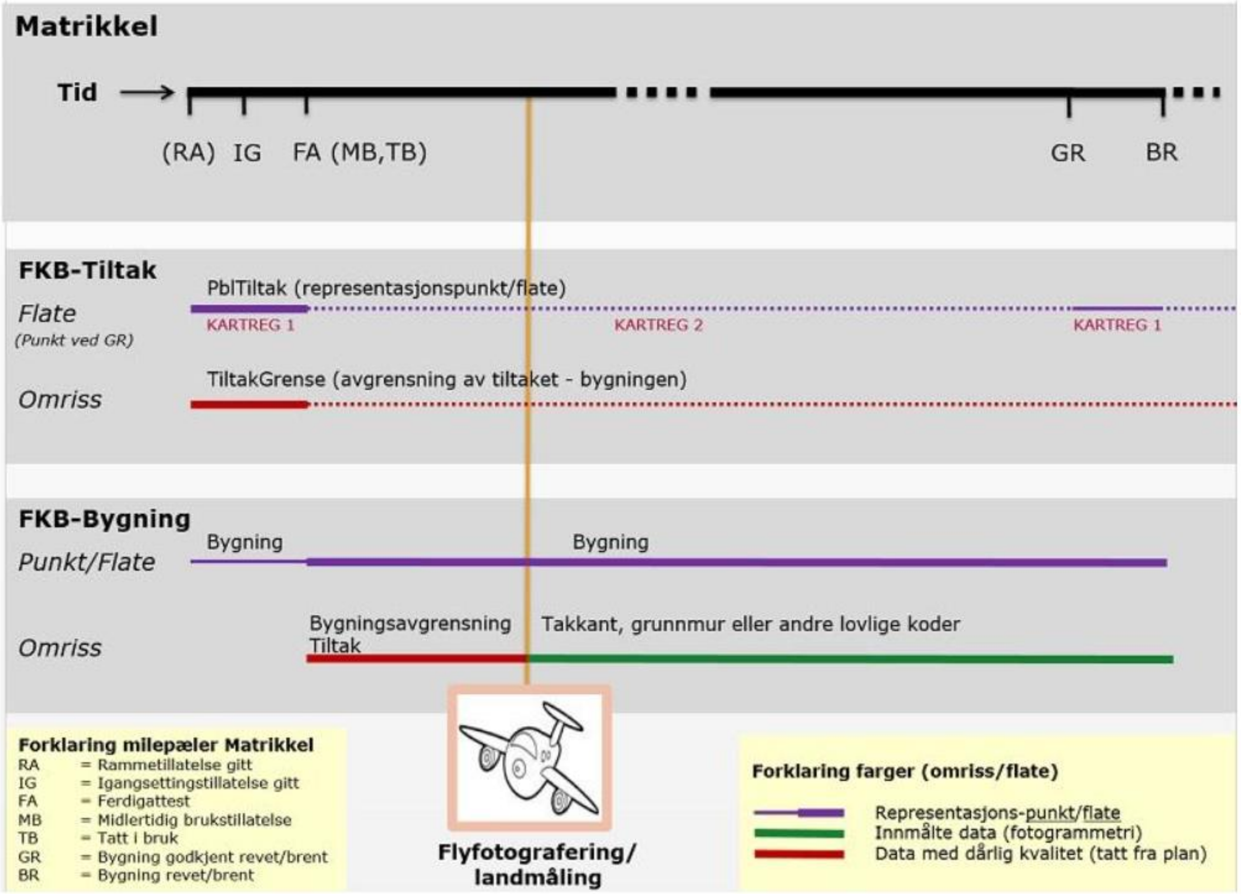
Finally, I reiterate that a rigorous comparison with field data has not been conducted, and this is a crucial step for the project's continuation. All GEE method results for the areas of interest, necessary for this verification, have been generated and are ready for this comparison.

Access to code and data

The GEE code and the data generated can be found on my GitHub account at this link:
https://github.com/mwaymel/GEE_Senbygg

Appendix I: Cadastre status details

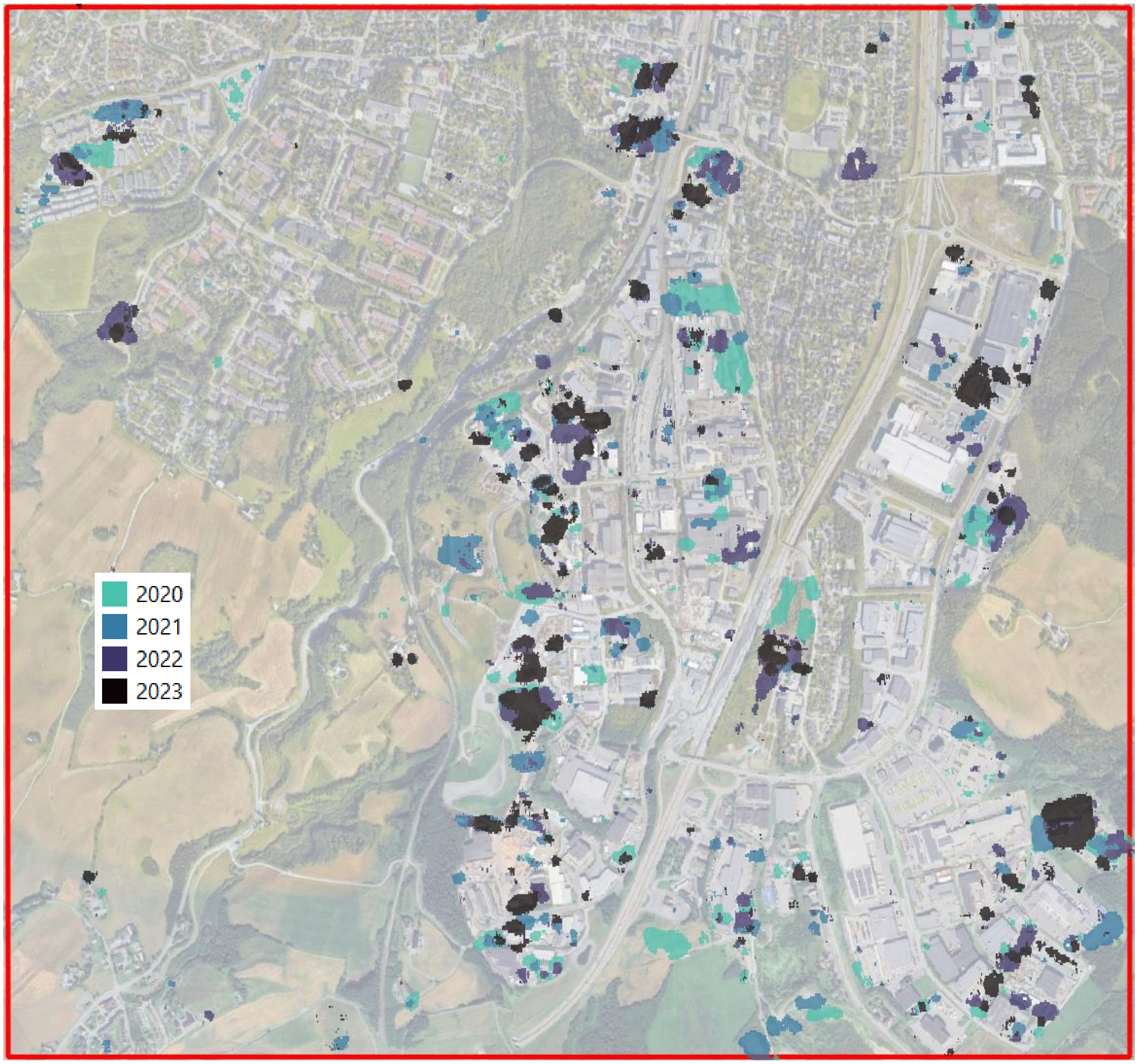
Information regarding cadastral ground data and their status is available at the following address: https://www.kartverket.no/geodataarbeid/forvaltning-drift-og-vedlikehold/byggtema


Matrikkel
Tid
(RA)
IG
FA (MB,TB)
GR
BR
FKB-Tiltak
Flate
(Punkt ved GR)
PblTiltak (representasjonspunkt/flate)
KARTREG 1
KARTREG 2
KARTREG 1
Omriss
TiltakGrense (avgrensning av tiltaket - bygningen)
FKB-Bygning
Punkt/Flate
Bygning
Bygning
Omriss
Bygningsavgrensning Tiltak
Takkant, grunnmur eller andre lovlige koder
Forklaring milepæler Matrikkel
RA = Rammetillatelse gitt
IG = Igangsettingstillatelse gitt
FA = Ferdigattest
MB = Midlertidig brukstillatelse
TB = Tatt i bruk
GR = Bygning godkjent revet/brent
BR = Bygning revet/brent
Flyfotografering/ landmåling
Forklaring farger (omriss/flate)
Representasjons-punkt/flate
Innmålte data (fotogrammetri)
Data med dårlig kvalitet (tatt fra plan)

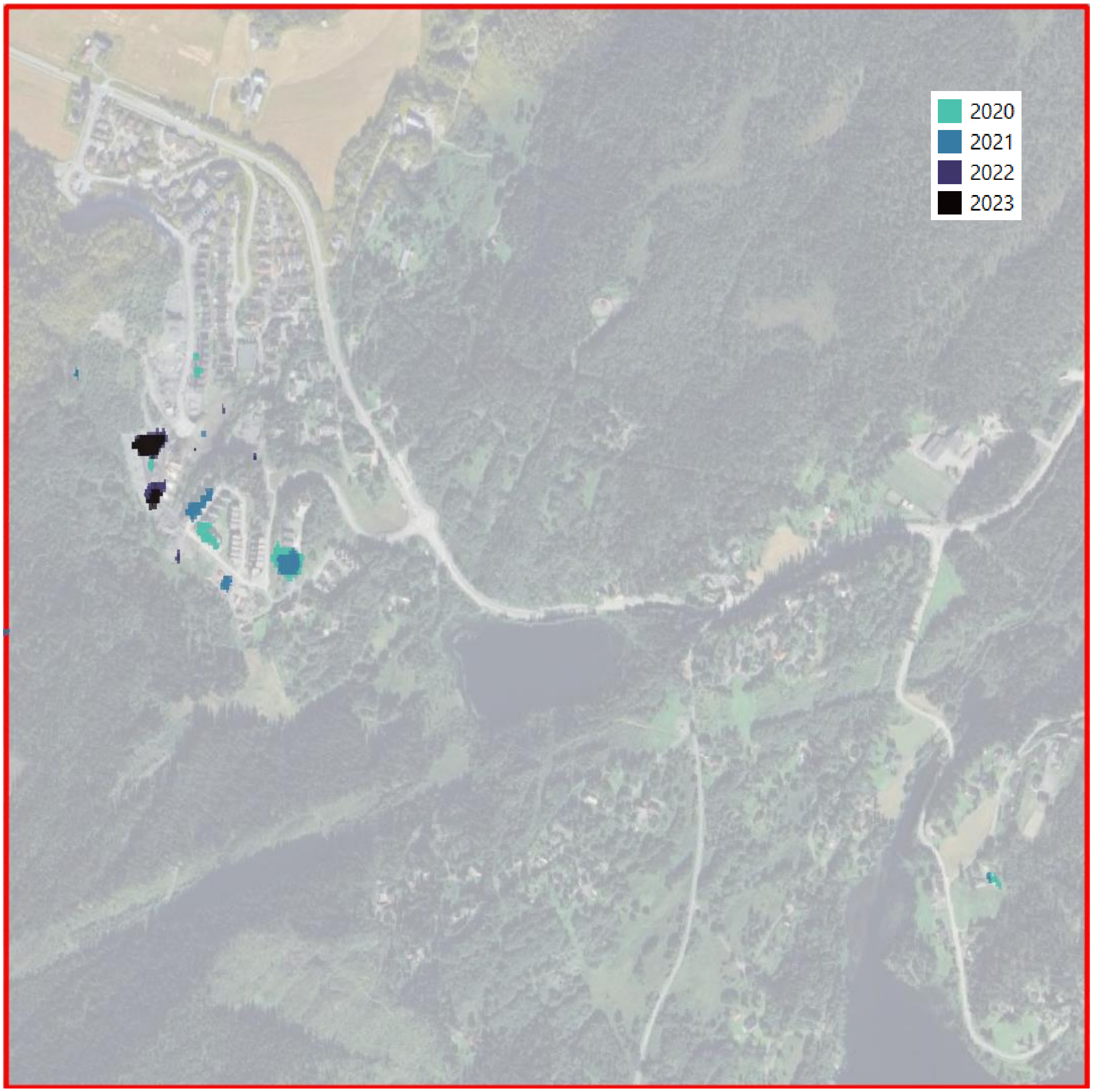
HOVEDMODELL: Bildet viser når informasjon skal registreres i henholdsvis matrikkel, FKB-Tiltak og FKB-Bygning ut fra saksgangen (bygningsstatus) i en sak. Ill: Kartverket.

Appendix II: Results on the 4 areas of interest

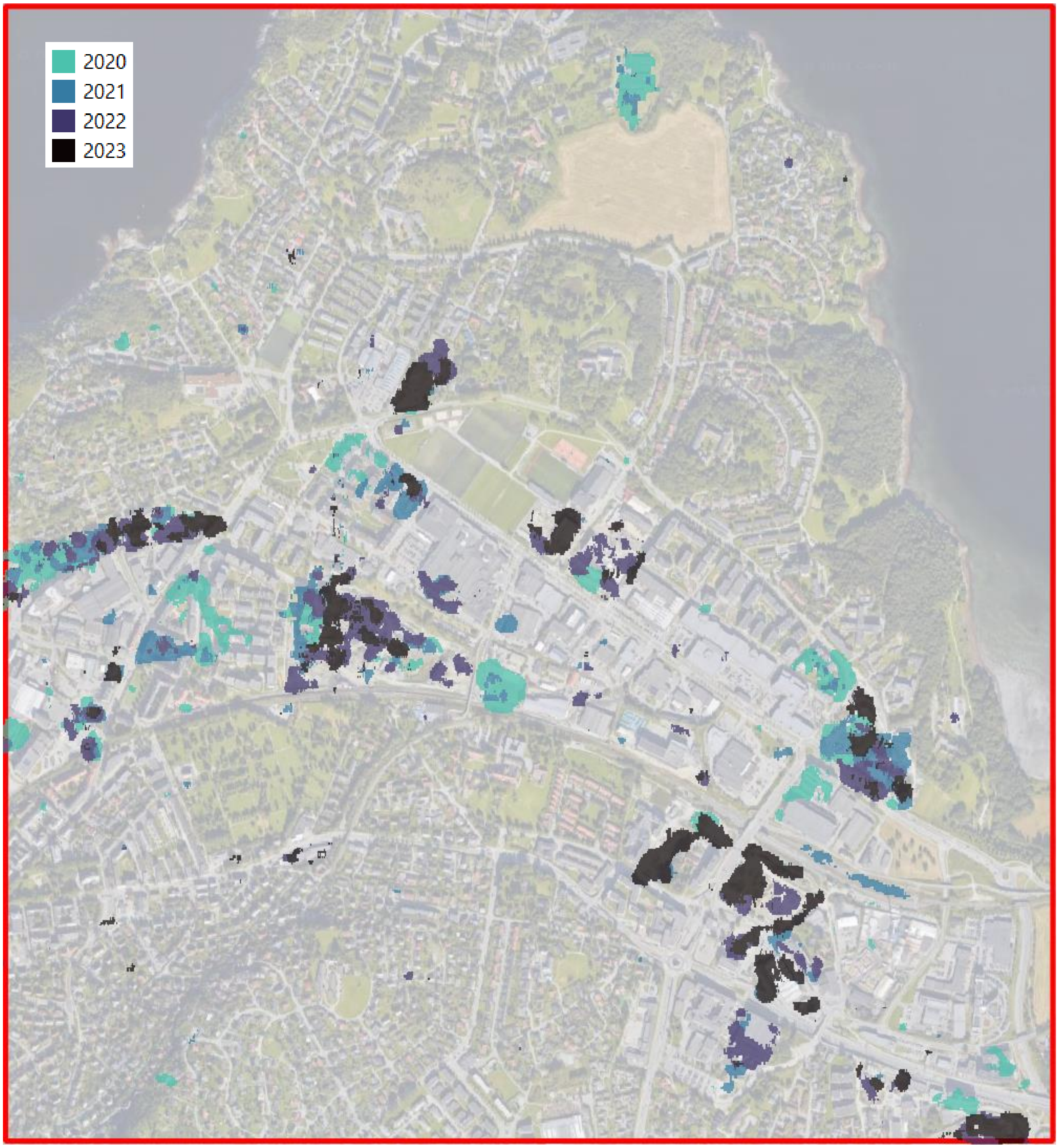
Trondheim - Heimdal



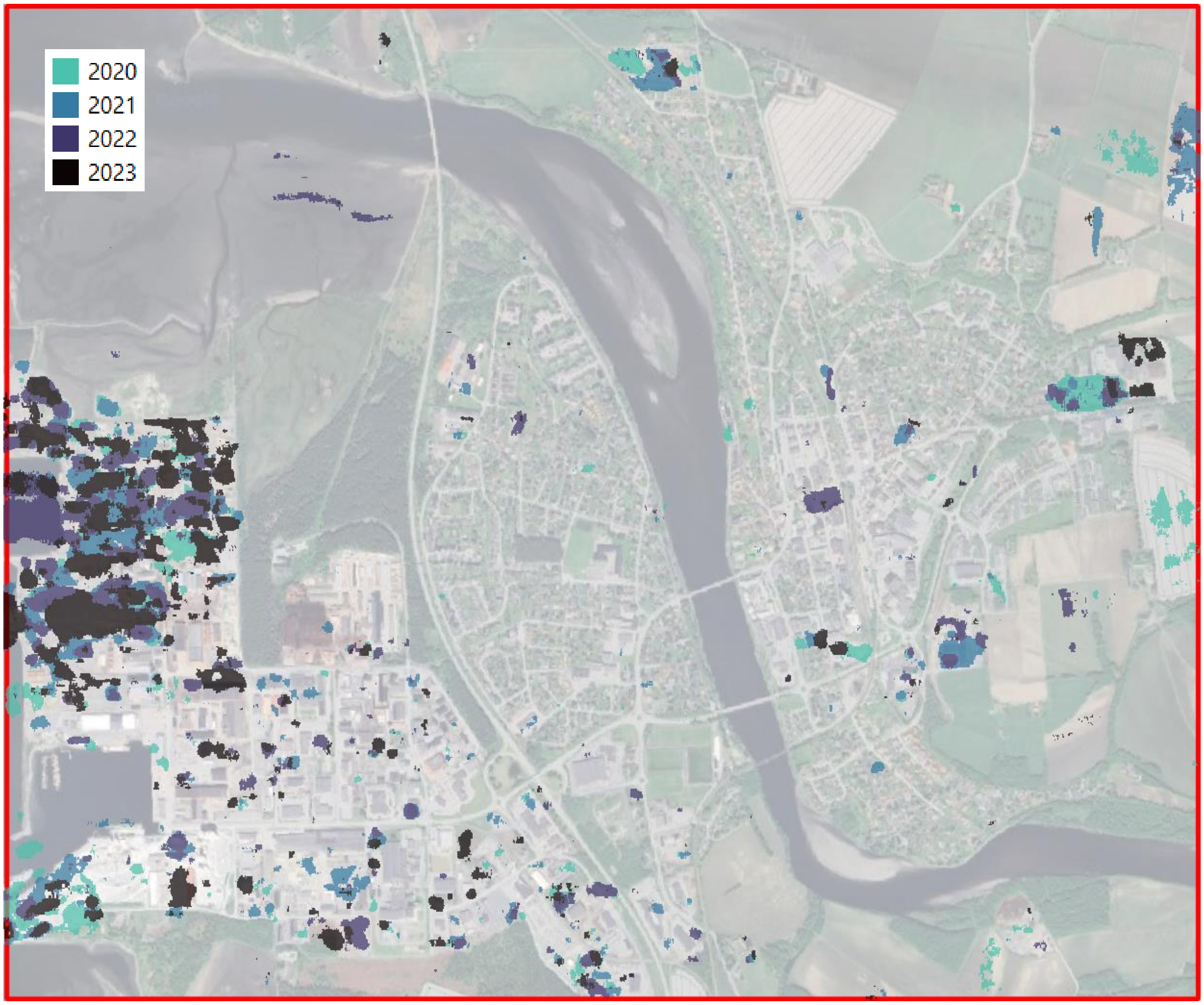
Trondheim - Fortunalia



Trondheim - Lade



Verdalsøra


2020
2021
2022
2023