Combining and extracting features across map types

Tekshov Norling, Sander
sandertn@ifi.uio.no

Oldernes, Joar
joaro@ifi.uio.no

Gjøstøl Strømsvåg, Emil Christopher
ecstroms@ifi.uio.no

May 2022

Contents

Introduction

This report highlights some of the methods we have attempted to match aerial photographs with known maps of an area. The inspiration comes from Google Maps, which obtains keypoints from a phone camera and matches these against a database of known photos of landmarks to do localization and GPS calibration. For an airplane, this matching could be done with a map of the plane's last known position and a photo taken of the ground. The aerial case is a little more challenging because we do not have access to a vast dataset of known locations, and the large regions being flown over are relatively homogeneous. To match outside of cityscapes, the initial plan was to extract roads because, although rare, intersections and turns should be more prevalent than buildings while also being locally unique. Aerial photos would be segmented using a U-Net, and maps would be segmented using color thresholding and morphological operations to clean up the result. We assumed these would be similar enough to estimate an affine transform between the segmentations.
Irregularities between the segmentations proved to be significant, and therefore, we suggest applying SuperGlue to match correspondences instead. Keypoints can be extracted using a handcrafted feature like SIFT/ORB or SuperPoint, before being subjected to an affine transform or homography estimation. This means that matching can only be performed in inhomogeneous cityscapes instead of the planned desired use case. The following methods and discussion sections will highlight the limitations of this approach, why we think it works for some cases, and some suggestions for future work to improve results and novelty in the approach.
Code available at `https://github.com/2ec/TEK5030_project`
The GitHub lacks weights from SuperGlue due to their size. Please use the zipped file this PDF was delivered in to run our code.

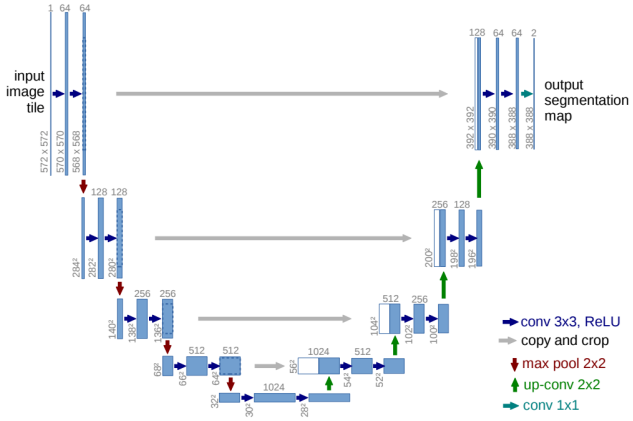

Figure 1: U-Net architecture. Retrieved from Ronneberger et al.[4]

Related work

U-Nets

Ronneberger et al. proposed the U-Net architecture for neuronal structures segmentation in electronic microscopic stacks.[4] These networks get their name from their shape, where layers are both down-sampled and up-sampled like in regular convolutional networks and also copied and cropped between layers of similar size. Previously, segmentation was typically performed by using a sliding window strategy, and classifying each pixel as either part of the object or the background.[1] An U-Net instead trains at different resolutions, massively reducing complexity and introducing a significant performance increase. The authors point out that the network can train on few samples and is suitable for augmentations, in stark contrast to the state-of-the-art classification methods at that time which required millions of labeled examples.[2]

SuperGlue

SuperGlue applies a Graph Neural Network (GNN) to find correspondences for matching sets of features.[7] Inspired by exploiting epipolar geometry in a static scene, SuperGlue can estimate pose transformations between cameras and find correspondences between keypoints in images. Since the method learn relevant priors directly from the data, as opposed to finding them through hand crafted features, SuperGlue also sometimes possesses the ability to find poses in planar scenes. SuperGlue takes visual descriptors from an image as input, and outputs a set of strong matched keypoints. Input features are added together with a learned position encoding, and then fed to an attention module. This module computes matching descriptors, by relating intra- and inter-pixel positions in two images through keys, queries, and values resulting in a weighted sum corresponding to how important the relations are for all pairs. Further, a message passing method is used to send information between edges of pixels. From the candidate connections, a partial assignment matrix is computed by maximizing a score matrix with the constraints of only having pairs of matching solutions. To optimize the network, homographies between images are applied to true matches which are compared to the negative log-likelihood for the predicted assignments. To extract keypoints for matching, SuperGlue recommend the self-supervised matching network SuperPoint.[6] The authors also recommend using non-robust homography estimators for the matched keypoints.

Methods

Aerial road segmentation

We employed a pre-trained U-Net[9] trained on the Massachusetts Roads Dataset[3] for aerial road segmentation. A blue shift was observed in the original dataset, and to improve segmentation, data augmentation by shifitng the blue channels to match those in the original dataset was attempted. This yielded poor segmentations, often without any road structures, and was later omitted. The pre-trained U-Net was trained on images of size 256x256, and therefore, centre-cropping was performed on test images. Visual inspection of segmentation results were preformed to determine optimal zoom

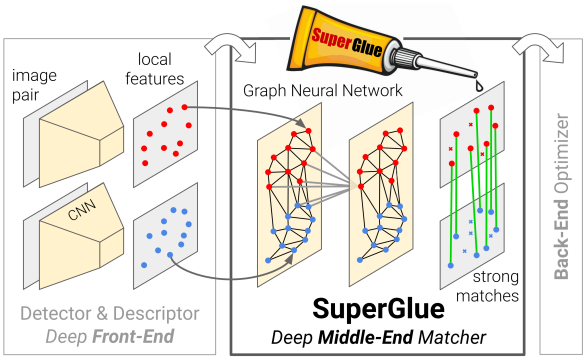

Figure 2: Overview of SuperGlues usecase. Retrieved from Sarlin et al.[7]

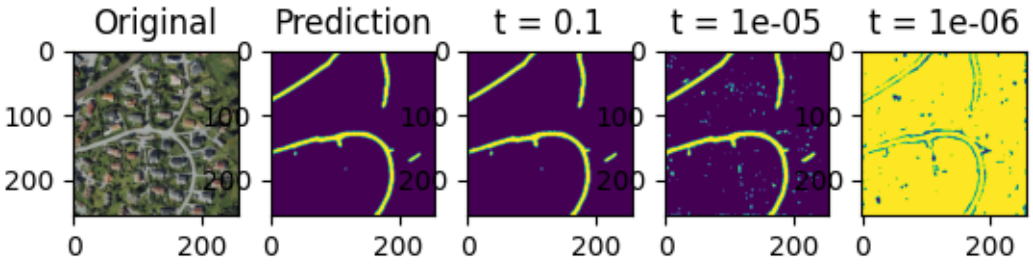

Figure 3: Example of segmentation with uncompleted road. Different threshold values applied to illustrate what the network learns.

level to be around 50 meters above the ground. The skeleton of the roads was extracted by morphological operations to better match the roads extracted from OpenStreetMap [8]. Segmentations still proved poor, sometimes omitting roads completely if no buildings are nearby, and sometimes breaking off roads due to for instance shadows. An experiment plotting different thresholds was performed to see if this parameter could be tuned for better segmentations, as seen in Figure 3. From this, it could be observed that the network cannot fill the roads completely for any value of threshold and one can confirm that the pre-trained network using roads in Massachusetts does not generalize to Norwegian roads. Overall, the approach of matching road structures proved unsuccessful, and we therefore moved away from it, in favor of using keypoints.

Map road segmentation

To have visual localization by using the segmented roads after the U-Net, we needed to have a ground truth of where the roads are in the map. At first we used the map from Kartverket [10], which we then used classical color segmentation to retrieve a map with only the roads. To achieve this, we found the gray color that Kartverket represents roads with. When extracting the roads we first found every shade of gray, between white and black, which was done by checking that the different color planes had the same value. By this method we had a map that only the parts that where drawn in gray shades, before we thresholded these pixel values to only have the shade of gray that corresponded with the roads. To further clean up the image, which had some noise, we used morphological opening to make the roads more complete. In Figure 5 we have an example of the map we used to segment and in Figure 6 is the corresponding image showing the extracted roads. As seen by Figure 5 and 6 we are able to extract the roads, but some segments are not closed correctly and some road segments are missing because the road name label are blocking the segment, removing the connection between connected roads.
This splitting of connected roads is a weakness of using images with labels, and also it is harder to locate where the areas in the image are located in the real world using coordinates, since then you will need a pixel-to-coordinate transform.

An easier and more robust method to get the segmented roads, is to use the same road data as the maps services use. To achieve this, we utilized an API [5] to access the database of road network

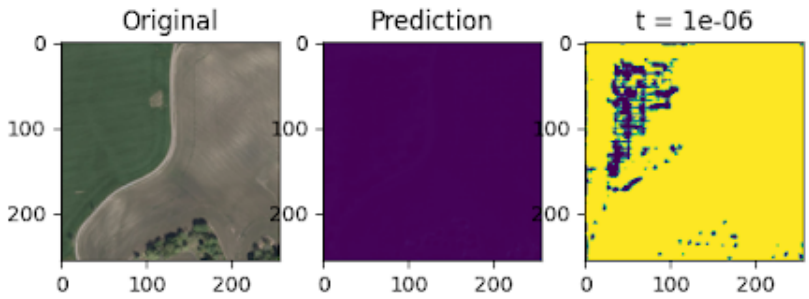

Figure 4: Example of segmentation on road in field. The network struggles to segment roads in areas without buildings. Ruins the assumption that roads can be used for matching outside of cityscapes.

Figure 5: Example map from Kartverkets Norgeskart[10].

from OpenStreetMap [8]. This approach allowed us to get other features that maps usually offer, like the ability to just get roads made for cars, ignoring pedestrian and bicycle streets. Alongside these road networks, we also get map features like buildings, shortest paths from point A to point B and off course map coordinates, so that we more efficiently can locate where in the world we are. In contrast to the segmented roads in Figure 5, we can clearly see from Figure 7 that the roads and features we get using the API is far superior to segment the map manually.

Keypoint matching

We used aerial images and map sections and used SuperGlues own image matching and recommended keypoint extraction SuperPoint. This gave us the corresponding keypoints for each image. We chose the best matches of keypoints and found an affine transform to overlap these images and create a merged image with details from both the map and the aerial photo.

Early stages

In the beginning we used a pretrained network together with the segmented roads from the map. This sadly showed no usable results, as we were not able to find any corresponding keypoints for the images as shown in Figure 8. Since this did not lead anywhere, we changed the images we wanted to

Figure 6: Example of roads using color segmentation on image of map in Figure 5.

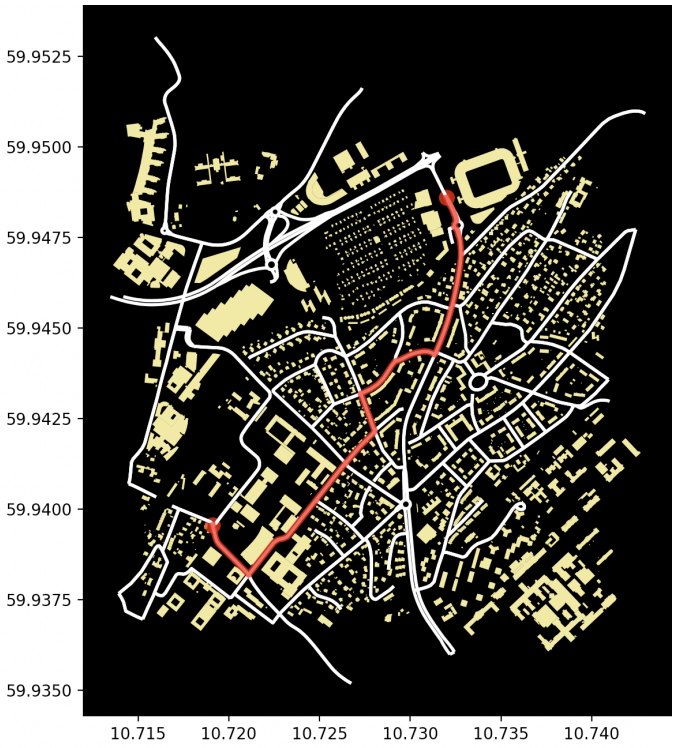

Figure 7: Example of roads, buildings and shortest path on Blindern, Oslo, using API.

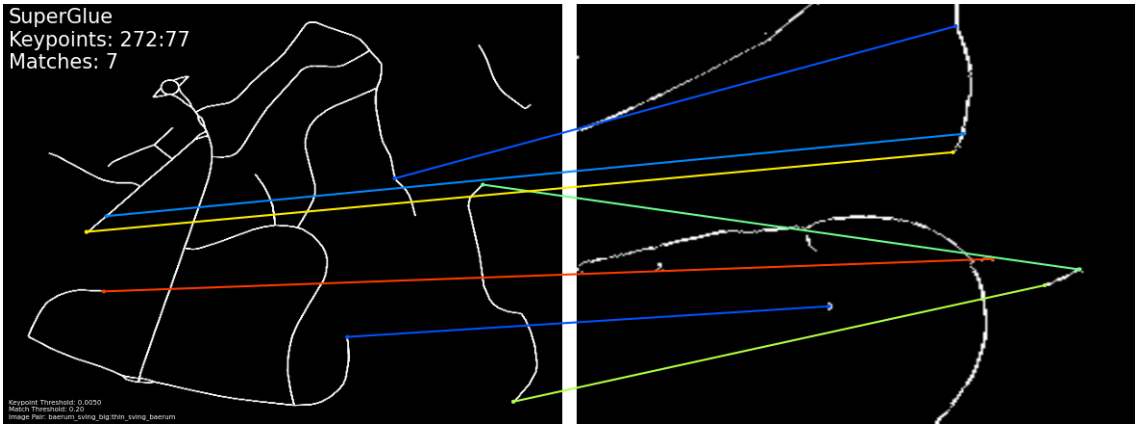

Figure 8: Bad matching of keypoints

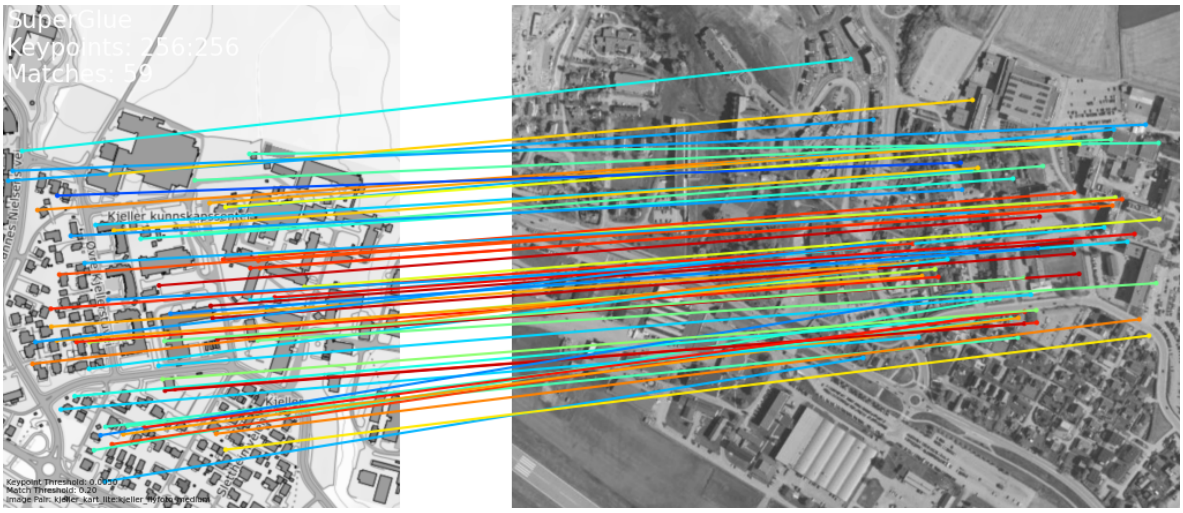

Figure 9: Clear matches with map and aerial photo

match. When using aerial images and map sections, we got better results and some clear keypoints with a strong match confidence, as shown in Figure 9

Overlapping

With the corresponding matches in each estimate, we created an affine transformation to rearrange the map into the original aerial photo. The overlapping image was given an opacity to get the advantages from both images. To find the affine transform, we used OpenCVs getPerspectiveTransform. This takes in four correspondences between images and uses gaussian elimination to solve the equations. We also used findHomography with RANSAC/the least squares method to estimate homography. The section Results / Discussion will outline how we observed these different methods to work, and which one we ended up using.

Results / Discussion

Since SuperGlue works on grayscale images, the resulting map and airphoto turn out pretty similar, primarily due to the high-quality Norgeskart.no map we used from Kartverket[10], which also includes accurate outlines of buildings. From the red lines in Figure 9 we can observe that SuperGlue can extract fairly accurate matches of correspondences between the images. This will only work for scenes rich with structures and information where maps correspond highly to the images, with few occlusions. When we estimate the affine transform, good overlappings are observed when the extracted matches are spread out in the images, especially towards the corners. Inaccurate results, especially in the corners, likely stem from the affine solution's inability to accurately model the transform when these are not present in the matched points. To improve results, using a scheme to choose high-scoring points that are far apart within the image could be applied. Since affine transformations are subsets of homographies, we also tried using OpenCVs homography estimator. Since SuperGlue scores matches, we assumed that using a scheme like RANSAC to detect outliers is less useful. However, we observed some high scoring matches that are entirely wrong, which lead us to apply RANSAC on all keypoints correspondences, instead of just the four best scoring ones. Since RANSAC detects outliers, it can get rid of high scoring correspondences that does not match and allows lower scoring matches to be used to estimate the homographies. As seen in Figure 12, when using RANSAC, the aerial photos roads and buildings match well with roads and buildings drawn on the map. Since SuperGlue cautions against using RANSAC, we also tried using the least squares method when estimating the homography, leading to worse matches than with RANSAC.
Even though we are able to match maps and aerial photos through SuperGlue, SuperPoint and RANSAC, the method is still fairly brittle. If the scene is homogeneous, SuperGlue often finds too few and the wrong correspondences, which leads to the overlapping missing entirely. Our method is also the opposite of novel. This is a result of spending to much time on getting the segmented matching to work, leading us to rush through the implementation of the keypoints method.
One way we could see the results improve is to use a feature extractor that is optimized for this

Figure 10: Airphoto overlapped with map with hand selected correspondences using an affine transform. We believe this result motivates future explorations with a scheme to choose matches that are more spread out instead of just using scores.

Figure 11: Airphoto overlapped with map with four best correspondences from SuperGlue using an affine transform. Here, we can see that the roundabout close to the region there are multiple correspondences is decently matched, while the top edges are entirely wrong, possibly due to lack of correspondences in that area. This motivates using homography estimations instead of an affine transform.

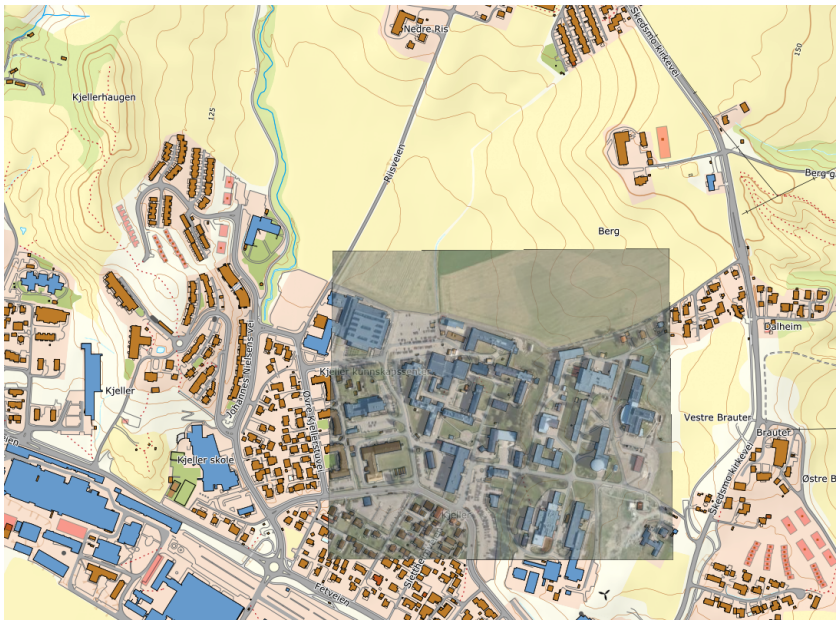

Figure 12: An proof of concept for our use-case of matching aerial images with a known map to do localization. Here, the centre of the image would approximately correspond to the planes location. We applied an homography estimator with RANSAC, which yielded the best results. Roads in the image correspond very well.

task. While SuperPoint finds good correspondences across the two domains we have tested on, maps and aerial photography, there is still room for improvement in extracting features. A feature that could be interesting to explore in the future is a feature extractor that works on homogeneous scenes. This could maybe be done by applying some structure on the aerial scene and match this up with a similarly structure in the map. This structure could take form in multiple ways, and one possibility could be to project a dot or grid structure from the air plane on the ground. Another way could be to use the movement of the air plane to construct a depth map of the environment, using structure from motion. This approach would make it possible to estimate depth and 3D structures to better match to known 3D structures in the real world. This way it could be possible to match an estimated 3D world to known topological maps and get corresponding match points even if the scene is homogeneous.

Conclusion

We have explored a brittle approach for matching aerial photos with map types. We find that given high quality map and no occlusions in the images, we are able to localize a photo within a map using SuperGlue, SuperPoint and homography estimation with RANSAC. Still, new methods needs to be explored in order to perform this task for homogeneous scenes and at a larger scale.

References

[1] Dan Ciresan et al. “Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images”. In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper/2012/file/459a4ddcb586f24efd9395aa7662bc7c-Paper.pdf.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[3] Volodymyr Mnih. “Machine Learning for Aerial Image Labeling”. PhD thesis. University of Toronto, 2013.

[4] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Ed. by Nassir Navab et al. Cham: Springer International Publishing, 2015, pp. 234–241. ISBN: 978-3-319-24574-4.

[5] G Boeing. "OSMnx: New Methods for Acquiring, Constructing, Analyzing, and Visualizing Complex Street Networks." In: *Computers, Environment and Urban Systems. 65, 126-139.* 2017. URL: doi:10.1016/j.compenvurbsys.2017.05.004.

[6] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. "SuperPoint: Self-Supervised Interest Point Detection and Description". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2018), pp. 337–33712.

[7] Paul-Edouard Sarlin et al. "SuperGlue: Learning Feature Matching with Graph Neural Networks". In: *CVPR*. 2020. URL: https://arxiv.org/abs/1911.11763.

[8] OpenStreetMap contributors. *Road network from OpenStreetMap*. https://www.openstreetmap.org. 2022.

[9] Shivam Jalotra Jerin Paul. *Skeyenet - Read Our Planet Like a Book*. https://github.com/Paulymorphous/skeyenet. Accessed: 2022-01-05.

[10] Kartverket. *Norgeskart*. https://www.norgeskart.no/. Accessed: 2022-01-05.