LOCALIZATION OF UAV[1] IN THE WILDERNESS: ADDRESSING RELIABILITY OF FEATURE MATCHING ALGORITHM

Chua Qiao Jun[1] , Fu Yu[2] , Teo Tiat Siak Justin[3] , Wong Ruiming Jeremy[4]
[1]NUS High School of Math and Science, 20 Clementi Avenue 1, Singapore 129957
[2]Nanyang Girls' High School, 2 Linden Dr, Singapore 288683
[3]Dunman High School, 10 Tanjong Rhu Rd, Singapore 436895
[4]Defence Science and Technology Agency, 1 Depot Road, Singapore 109679

Abstract

UAV technology has seen rapid evolution throughout the years, transforming itself into an invaluable asset for applications such as search and rescue operations, inspections, mapping and agriculture. One major challenge in these applications is the heavy reliance on GPS for a drone to effectively localize itself for UA operations. GPS is susceptible to jamming, and may not function properly in remote and non urbanised areas, where GPS signals may be weak or completely unavailable. This paper aims to investigate the novel approach of geo-referencing satellite imagery using feature matching algorithms to enhance the localization capabilities of drones in such environments, focusing on overcoming the limitations of traditional satellite-based navigation systems. By examining algorithms like SuperPoint and SuperGlue, we aim to assess the accuracy, reliability and efficiency of satellite imagery feature matching drone localization, especially in the context of Singapore, in situations when GPS/GNSS is unavailable. Our findings suggest that while these intricate algorithms are capable of fabricating apt results under highly specific conditions, it faces several limitations by numerous factors such as zoom level, resolution of query and reference images, and others. This paper aims to scrutinize a UAV localization algorithm to see if it is apt for real-world solutions, in the context of Singapore.

Introduction

This project expands on a previous research paper "Vision-based GNSS-Free Localization for UAVs in the Wild" [1], which details the increasing need for localizing these aerial systems in non-urban environments. The algorithm implemented is based on matching salient features of RGB photographs captured by the drone camera to features of sections of a pre-built map consisting of georeferenced open-source satellite images. Doing so, the algorithm is able to output a prediction of the drone's localization data as a set of numbers, longitude and latitude, so as to pinpoint the location of the drone. In order to compute geographical coordinates, the algorithm uses a fairly simple approach. It starts by rotating the drone photograph using the image metadata providing the orientation of the UAV and the camera gimbal to match the heading of the map sections (always oriented northwards). The rotation is absolutely necessary because it improves the number of features which can be matched for a pair of images. We intend to test out this localization algorithm with our own provided data to measure the extent of

[1] an unmanned aerial vehicle (an aircraft piloted by remote control or onboard computers):

its functionality, which will be able to provide information to future developers on possible improvements that can be made to the algorithm.

Research Question

To what extent is the Vision-based GNSS-free localization Algorithm effective? The effectiveness of a vision-based GNSS-free localization algorithm is largely determined by its ability to accurately estimate a drone's position and orientation in remote or GPS-denied terrains such as forests or mountainous regions, traditional GNSS-based systems often fail due to signal blockage or interference. The algorithm's effectiveness depends on factors such as the quality of visual data, the drone speed, the complexity of the terrain and the robustness of the algorithm in handling lighting changes and dynamic environments.

Hypothesis

Our hypothesis is that the localization algorithm was very sensitive to any discrepancies to the nature of the "query" image and the satellite image provided, such as the resolution of the image and the pixels of the feature matching. As an analogy, we are predicting that the localization algorithm will be affected given the geo-referenced satellite images are of different magnification and resolution.

Methodology

The Methodology for evaluating feature matching algorithms in wilderness UAV localization encompasses three main phases: dataset collection, algorithm implementation, and performance evaluation. Firstly, we studied the existing literature paper and understood the concepts. This involves an extensive review of existing literature on GNSS-free drone localization methods. Key studies are identified to analyze the current state of the field, including the various algorithms and technologies used for localization in wild terrains. The review explores challenges of GNSS localizations and identifies gaps in the existing research.

The query image and numerous sets of reference satellite maps were collected. Reference satellite maps were taken from Google Maps. Due to logistical and time constraints, only 1 query image (taken by UAV) was available. The query image was fed into the feature matching algorithm and tested against different sets of satellite maps (of varying quality and condition), where results such as predicted location and amount of feature matches were collected.

We conducted 4 separate tests to examine the algorithm's full capability and limitations. The first test was to vary the zoom level (measured in m/pix) of the reference satellite map, to test the algorithm's sensitivity to the reference maps' quality and resolution, etc. This test was conducted with a single query image against a single map image, to test the algorithm's ability to identify feature matches and predict the GPS coordinates of the query image.

The second test was multi-map test. The reference map was a set of 5 images, where the satellite map was split into 5, all of which had a standard zoom level. This was to test the program's ability to scan through a larger reference map, and identify which image had the most appropriate feature matches among a pool of numerous images.

The third test was also a multi-map test, but all the images within the reference map set had similar features as the query image, to assess the algorithm's ability to correctly identify and feature match the map image that matches the query image amongst a pool of similar images, in order to predict the location of the query image. This test was to assess the algorithm's accuracy when dealing in multi image map sets, where some images are similar.

The last test was a "split test", where the correct map image was split into 4, to assess the algorithm's ability to identify the correct matching image(s) to the query image when there is more than one correct image.

1. Dataset Collection

For the query image, we used an existing aerial image taken by a UAV that was available.
Variability in Conditions:
Equipment: DJI Air 2S
Camera Resolution: 5280x2970

When fed into the algorithm for tests, the image was rotated such that the image faced north, and was downscaled to 337x600px.

The reference maps were taken from Google Maps. Zoom level of the maps (measured in m/pix) was standardised within each set of maps, and vary between different sets (zoom level test). For the multi tests and split tests, zoom level was standardised to 0.715m/pix. The top left corner and bottom right corner of each map image has their coordinates (lat, long) recorded and inputted in the program via a csv file, such that the algorithm is able to predict the coordinates of the query image when it matches with one of the map images.

2. Feature Matching algorithms

Classical Algorithms that were introduced to us included SIFT, SURF, ORB. Deep Learning-Based algorithms included SuperGlue, SuperPoint. The Algorithm was evaluated based on feature extraction, descriptor matching and computational efficiency.

The program used was "Vision based GNSS-free localization of UAVS in the wild" by TerboucheHacene, an extension of the original implementation by Wildnav. (https://github.com/TerboucheHacene/visual_localization)

The program was installed via Powershell in Visual Studio Code terminal, and was mainly accessed and ran through VSC.

Trying out the Software with our own images

Hardware:

- CPU: Intel i9-13900H.
- GPU: NVIDIA RTX 3080M.
- RAM: 32 GB.

Software:

- OpenCV (v4.5) for classical algorithms.
- PyTorch (v1.12) for deep learning models.
- ROS (Robot Operating System) for UAV localization integration.
- Visual Studio Code.

3. Performance Evaluation

To evaluate the accuracy and reliability of our localization results, we considered the following metrics that potentially determine the accuracy of the algorithm. Firstly, Error in Position – we measured the distance between the ground truth positions and the estimated positions from the feature matching algorithm and calculated the magnitude of the error, to give us an indication of the localization accuracy. Secondly, we conducted Error in Orientation, we assessed if feature matching accuracy would be affected by orientation of the query image. Then, we evaluated the consistency of feature matching and localization over time — consistent feature matching across a sequence of images indicates a reliable localization process.

Metrics and Benchmark

To evaluate the impact of zoom levels (zoom level test) on feature matching precision, we introduced a controlled experiment where images were scaled to simulate different zoom levels. Precision: True positive matches as identified by ground truth homographics. We calculated the ratio as Error Magnitude against Zoom Level. Defined as the deviation in feature match locations between the scaled image and the baseline image. The average error magnitude was computed. The number of matched features that remained consistent across zoom levels was calculated as a percentage of the total detected features in the baseline image. A graph was created to display the relationship between zoom levels and error magnitude. For the second to fourth tests, we determined whether the algorithm was able to accurately identify the corresponding satellite map and feature match, and to what extent was feature matching done.

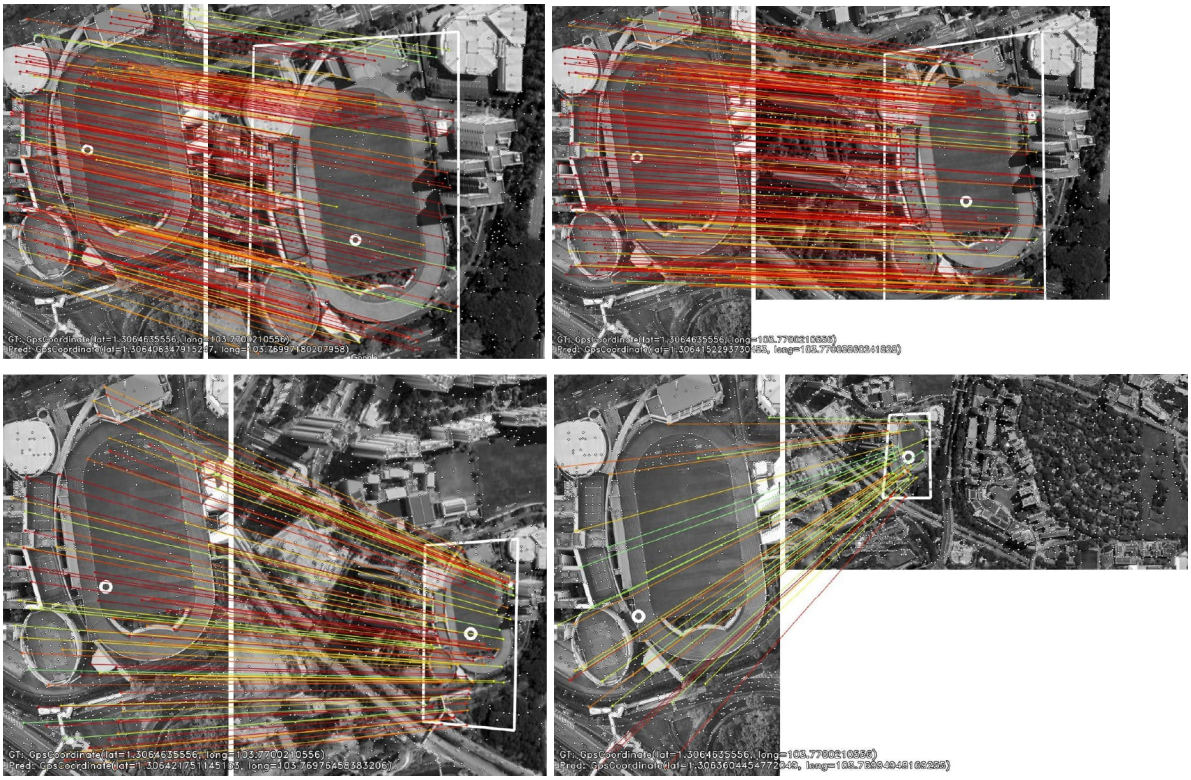
Literature Review

The use of UAVs in wilderness environments, where GPS signals are unreliable, has spurred the development of advanced localization techniques. Feature matching is a central approach in visual-based localization – a key area in computer vision, pattern recognition and machine learning. Key points are detected in images matched across multiple views to estimate the UAVs position and orientation. Feature matching typically involves three stages – Feature extraction, Feature Description and Feature matching. This section reviews the existing literature on the challenges of localization in such environments, focusing on the role of feature matching algorithms like SuperPoint and SuperGlue. The traditional method of Feature Matching is the SIFT (Scale-Invariant Feature Transform) [3] and ORB (Oriented FAST and Rotated BRIEF) [4] algorithm, which makes use of Scale-space peak selection to locate keypoints. However, these algorithms often struggle in environments with low-texture areas or poor feature contrast, which is common in some settings like rocky landscapes or forests. Like SIFT, SuperPoint is a deep learning-based feature detection and description method introduced by DeTone et al. (2018) [5] used for feature extraction and feature description. Unlike traditional algorithms, SuperPoint is a fully-convolutional neural network (CNN) architecture which operates on a full-sized image and extracts and describes key points directly from the image. In their experimental results, DeTone et al. demonstrated that SuperPoint outperforms traditional methods like SIFT and ORB, especially in areas with low texture, which are common in wilderness environments. SuperGlue, introduced by Sarlin et al. (2020) [6], is a deep learning-based method for feature matching that builds upon SuperPoint's feature extraction. Sarlin et al. showed that SuperGlue significantly improves matching accuracy and robustness, outperforming previous methods like SIFT and MIND (a traditional matching algorithm) in both real and synthetic environments. Algorithms like SuperPoint and SuperGlue represent significant advances in feature extraction and matching. Despite their strengths, challenges like low-texture areas, occlusions, and environmental dynamics remain.

Results

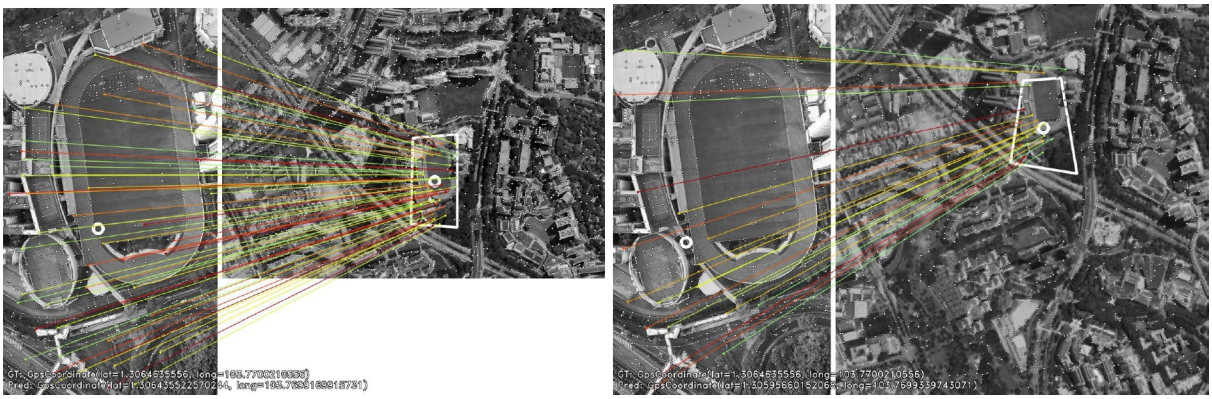
ZOOM LEVEL TEST

Zoom level for the reference map was measured in m/pixel. The images below show the query image (left side of each image) feature-matched to the satellite image of varying zoom level (right side of each image).



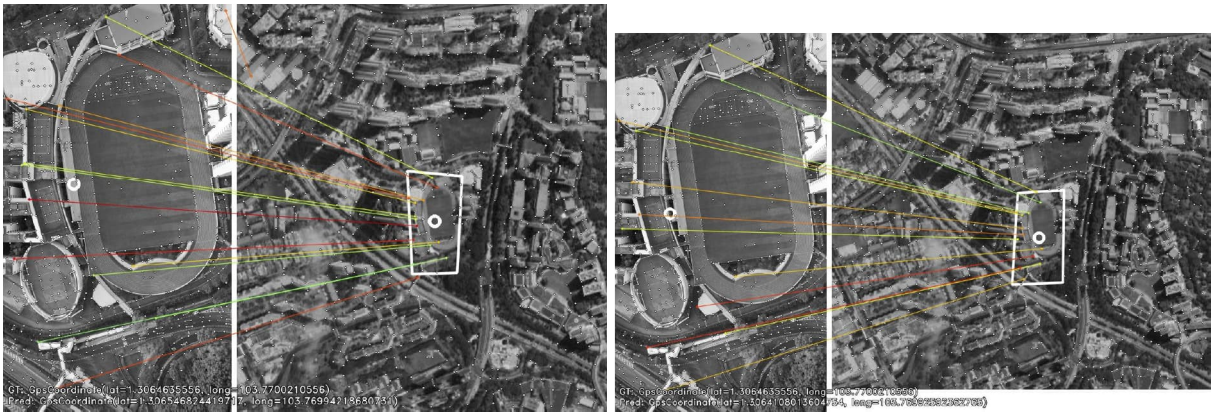
Top left: Figure 1(a) (Zoom: 0.301m/pix); Top right: Figure 1(b) (Zoom: 0.503m/pix); Bottom left: Figure 1(c) (Zoom: 0.990m/pix); Top left: Figure 1(d) (Zoom: 1.18m/pix)

Figure 1



Left: Figure 2(a) (Zoom: 1.17m/pix); Right: Figure 2(b) (Zoom: 1.64m/pix);

Figure 2



Left: Figure 3(a) (Zoom: 1.39m/pix); Right: Figure 3(b) (Zoom: 1.66m/pix);

Figure 3

For maps beyond the zoom level of 1.66m/pix, the algorithm was unable to find a proper image match and failed to deliver an output. The maps we tried that were beyond 1.66m/pix zoom level are below.

Figure 4

MULTI IMAGE TEST

The following set of 5 images was the map set used for the multi image test. All images had a standardised zoom level of 0.715m/pix.

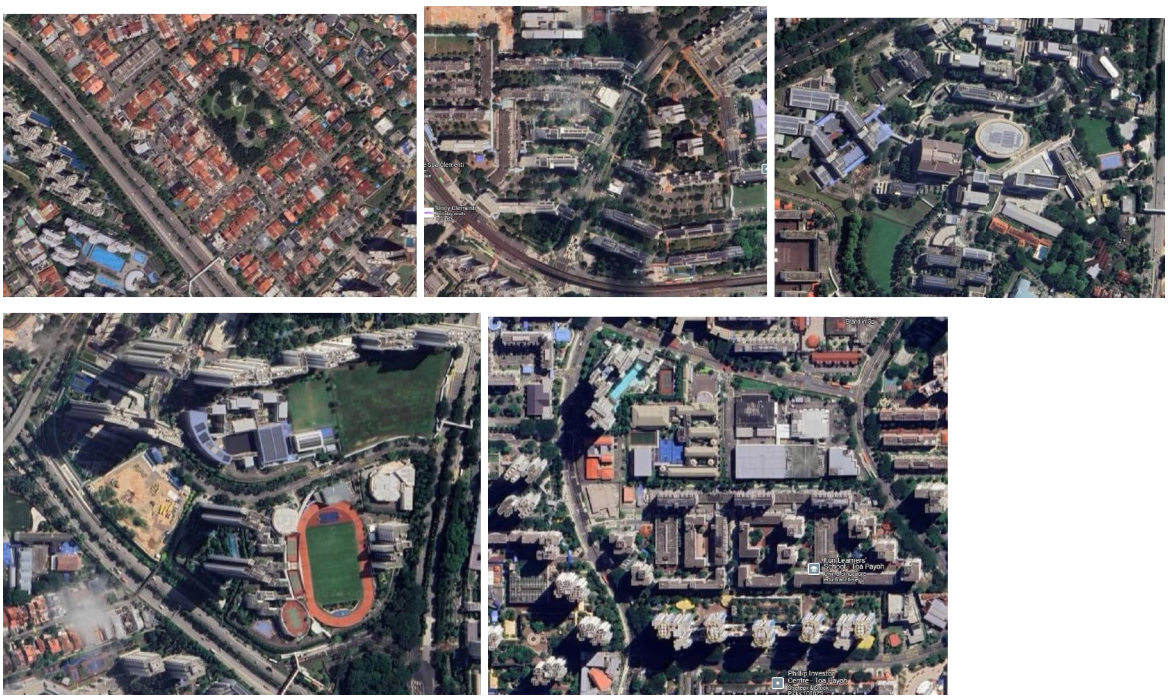


Figure 5(a)

The output of the aforementioned test is seen in Figure 5(b). The algorithm managed to identify the correct image out of the 5 and feature map it rather extensively.

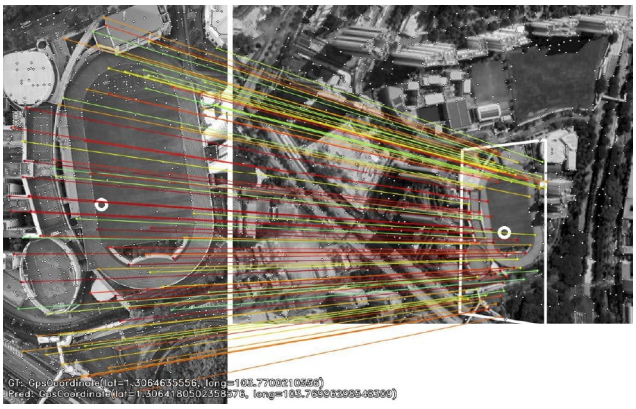


Figure 5(b)

MULTI IMAGE WITH SIMILAR FEATURES TEST

The following set of 5 images was the map set used for the multi image with similar features test. All images had a standardised zoom level of 0.715m/pix.

Figure 6(a)

The output of the aforementioned test is seen in Figure 6(b). The algorithm managed to identify the correct image out of the 5 and feature map it rather extensively.

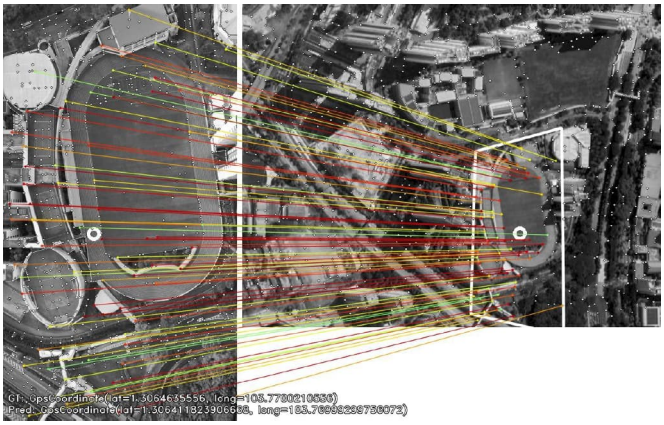


Figure 6(b)

SPLIT TEST

The first split test was conducted with the correct map image split into 2, and an extra irrelevant image. The images used are below.

Figure 7(a)

The output of the aforementioned test is seen in Figure 7(b).

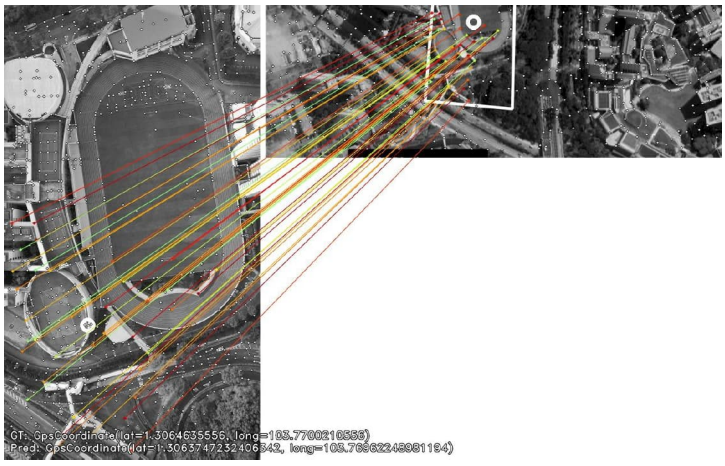


Figure 7(b)

The second split test used the following images below, where the correct image was split into four.

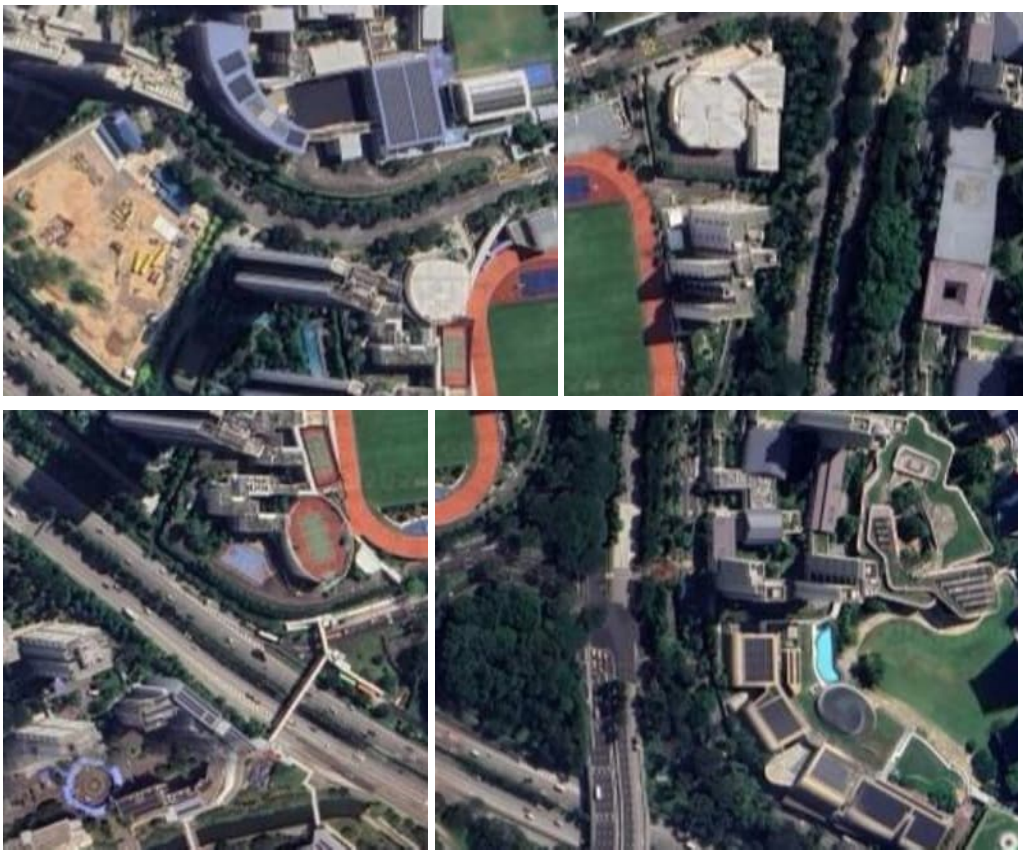


Figure 8(a)

The output of the aforementioned test is seen in Figure 8(b).

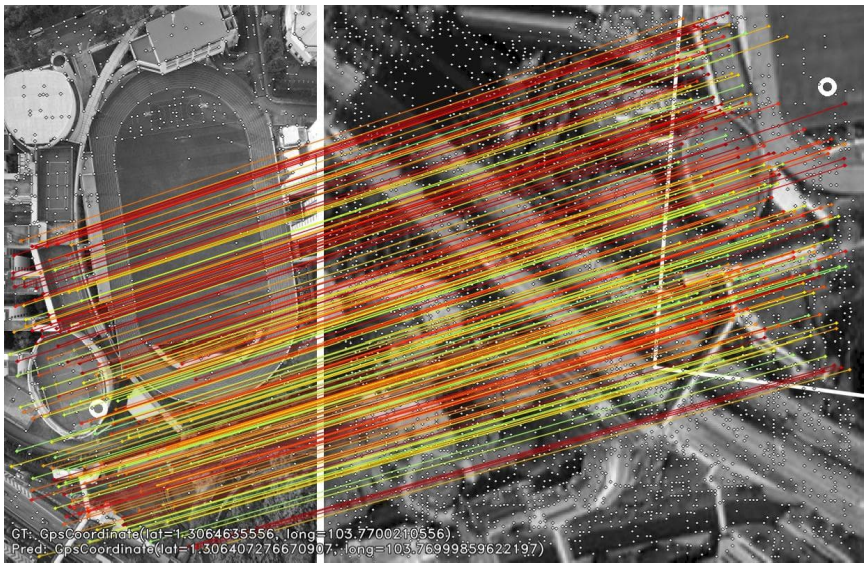


Figure 8(b)

Analysis

Zoom level test

For all output images from the test, the predicted coordinates of the query image and the actual ground truth coordinates were both generated. The magnitude of the error between the predicted coordinates and the actual coordinates was measured for the zoom level test, and the data points are plotted in Figure 9.

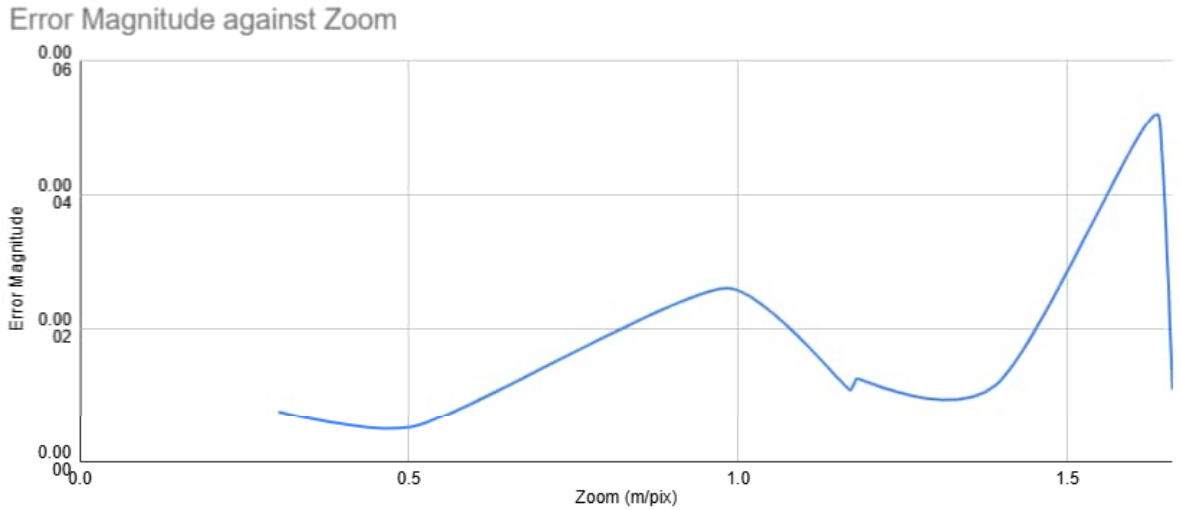


Figure 9

Our hypothesis was that as zoom level increases, quality and resolution of the satellite map decreases, and the magnitude of error increases. We expected an exponential increase in the error magnitude. From Figure 9, it seems that our hypothesis was correct to a certain extent, but there were anomalies at the zoom levels of 1.17-1.18m/pix, where error magnitude dipped down. This was observed again between 1.64m/pix to 1.66m/pix, with a staggering fall in the error magnitude. We believe these 2 data points to simply be outliers, and presumed that the algorithm got lucky and managed to predict the coordinates of the query image rather accurately. There may also be the possibility of human error when inputting lat,long coordinates for the map images leading to such anomalies, but we were unable to confirm any errors. The general increasing trend in error magnitude is still observed.

We conducted tests trying reference maps beyond 1.66m/pix zoom level, as seen in Figure 4, but were unable to obtain any matches or output. We can reasonably assume that the algorithm stops working past the zoom level of around 1.66m/pix-1.70m/pix.

Multi image test

When the program was fed with a set of 5 images that are dissimilar with one another (Figure 5(a)), the program was able to identify the correct image and feature match it, producing the output shown in Figure 5(b). We can see that the program was not only able to identify the correct map image, but also managed to feature match between the query image and map image rather extensively.

Multi image with similar features test

When the program was fed with a set of 5 similar images (Figure 6(a)), where all images had similar features, it was still able to identify the correct image and feature match it rather extensively (Figure 6(b)). We can conclude that the program is pretty versatile, and is intelligent enough to rely on numerous sources of feature matches (trees, bridge, roads) to identify the correct image, and will not misidentify the images even when there are similar features (track/field).

It's possible that the orientation of the images played a big part in helping the algorithm identify the correct image too.

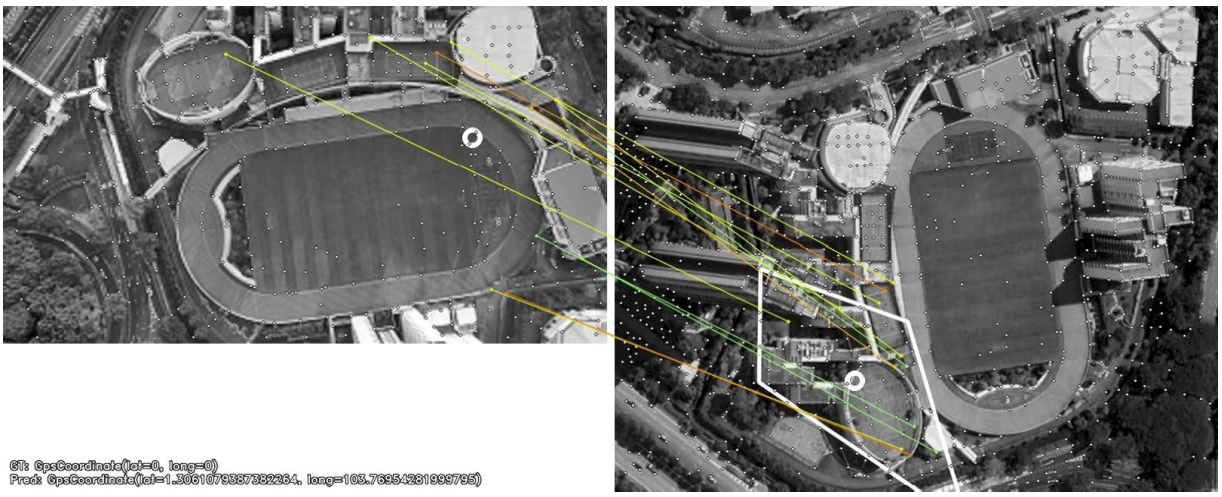


Figure 10

As seen in Figure 10, when the query image had its orientation changed from facing north to facing east, the algorithm was not able to feature match the query and reference images properly. We can conclude that the program relies heavily on image orientation for feature matching, which helped it to identify the correct image in this particular test.

Notably in the future, we could reuse the same set of images, but orient the similar features (track/field) in the same direction as each other to see how the program handles it.

Split test

As seen in Figure 7(b) and 8(b), the program was able to feature match correctly, but only partially, and failed to detect that there was more than one correct image in the reference map set. Notably, the algorithm preferred the bottom right corner of the query image to feature match, and would always match it to the corresponding bottom/bottom right area image in the reference map set. Its plausible that as earlier mentioned, the program relied more heavily on the roads/trees/bridge found in the bottom/bottom right area of the image for feature matching, together with other features, hence explaining this observation.

Limitations and Future Work

These limitations stem from aspects of methodology, scope of testing, and the complexities of real-world UAV localization tasks. Below are the key limitations of the report:

1. Logistical challenges

We faced many logistical challenges in this process. Firstly, we were lacking on Development Laptops, which hindered our progress in generating data. We were not able to get our hands on the drone model most preferred for taking footage of the query image. It also required personnel with more specialized skills in robotics, computer vision and sensor fusion. Hence, it took us some time to get used to the algorithm. This caused some data collection difficulties.

2. Short-term data collection

The tests presented in the report are based on a limited number of images. Furthermore, our internship lasted only about 4 weeks, and with 2 weeks of understanding the algorithm, we only had 2 weeks left for data collection. Longitudinal data over extended flight periods and continuous localization testing are essential for assessing the consistency and robustness of feature matching algorithms.

3. Lack of real-world conditioning

Real-world UAV missions typically span longer durations and testing for performance degradation over time could not be included in this study. Hence, there was a lack of real-world field validation with Ground truth GPS data for comparison. This report also assumes that the environment remains mostly static between image captures, which is often not the case in real-world UAV missions. While the algorithm handles feature matching well in controlled settings, the presence of dynamic objects such as people, vehicles and animals in the environment could introduce challenges that were not accounted for in the report. Once again, we did not have enough understanding and logistical materials to conduct further research and testing on further real-world scenarios. Hence, our report does not explore or consider the difficulty of matching features in environments with significant movement or fast-changing scenarios.

Future works/Studies

Hence, while this report provides an evaluation of the algorithm for UAV localization in the environment, the limitations outlined above should be seriously considered when interpreting the results. These are key challenges that were not comprehensively addressed. Future studies should focus on extending the scope of testing to diverse environments, to a greater range of orientation, position and zoom of the images. It should also include improving real-world processing capabilities, such as considering detecting dynamic objects. This can be done by validating results under dynamic, long-duration missions to provide a deeper understanding of how these academic algorithms can be implemented in real-world scenarios and support UAV localization in challenging, GPS-weak scenarios.

References

[1] Gurgu, M.-M., Peña Queralta, J., & Westerlund, T. (n.d.). Vision-based GNSS-free localization for UAVs in the wild. Turku Intelligent Embedded and Robotic Systems (TIERS) Lab, University of Turku.

[2] c

[3]G. Lowe, D. (n.d.). Distinctive image features from scale-invariant keypoints. Distinctive Image Features from Scale-Invariant Keypoints. https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf

[4] Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (n.d.). (PDF) Orb: An efficient alternative to SIFT or surf. ORB: an efficient alternative to SIFT or SURF. https://www.researchgate.net/publication/221111151_ORB_an_efficient_alternative_to_SIFT_or_SURF

[5] DeTone, D., Malisiewicz, T., & Rabinovich, A. (2018, April 19). SuperPoint: Self-supervised interest point detection and description. arXiv.org. http://arxiv.org/abs/1712.07629

[6] Sarlin, P.-E., DeTone, D., Malisiewicz, T., & Rabinovich, A. (2020, March 28). Superglue: Learning feature matching with Graph Neural Networks. arXiv.org. https://arxiv.org/abs/1911.11763