Visual Localization of Unmanned Aerial Vehicles using Supervised Learning for Feature Detection in Aerial Images

1st Daniel Sander Isaksen
Department of Informatics
University of Oslo
Oslo, Norway
daniesis@ifi.uio.no

2nd Artem Chernyshov
Department of Informatics
University of Oslo
Oslo, Norway
artemch@ifi.uio.no

3rd Clément Dargein
ISMIN
Mines Saint-Etienne
Gardanne, France
clement.dargein@etu.emse.fr

4th Mathias Mantelli
Phi-Robotics Research Group
Federal University of Rio Grande do Sul
Porte Allegre, Brazil
mathiasfassini@gmail.com

5th Renan Maffei
Phi-Robotics Research Group
Federal University of Rio Grande do Sul
Porto Allegre, Brazil
rqmaffei@inf.ufrgs.br

6th Kai Olav Ellefsen
Department of Informatics
University of Oslo
Oslo, Norway
kaiolae@ifi.uio.no

***Abstract*—The context of our work is to investigate the effectiveness of machine learning models in the measurement model of the Monte Carlo Localization algorithm, to provide a more robust localization of the unmanned aerial vehicle, using aerial imagery. It is shown that machine learning methods can outperform traditional methods of computer vision, while the Monte Carlo Localization is the golden standard for localization. Therefore we propose that the right machine learning method can give the most effective visual localization of flying robots. This paper presents a study on the issues of the implementation of Terrain classification and Object detection in Monte Carlo Localisation algorithm, the advantages and disadvantages of each solution and how to perform it.**

***Index Terms*—Visual Localization, Unmanned Aerial Vehicle, Aerial Imagery, Monte Carlo Localization, Measurement Model, Computer Vision, Terrain Classification, Object Detection**

I. Introduction

In recent years, the use of Unmanned Aerial Vehicles (UAVs) has risen to a whole new level, owing to their versatility. Possible applications are many, ranging from supply deliveries and recreational photography to surveillance and rescue operations [11]. For most use cases, a positioning system is needed, mostly to find and keep a desired spatial position. A Geo-spatial Positioning System (GPS) is usually the preferred system for this task, but not always, as explained in [22] [4], the system is vulnerable to interference from an increasing amount of signals in the ether, as well as environmental changes caused by nature.

With the constant improvements in commercially available technology, UAVs are equipped with increasingly advanced hardware, allowing for expanded range of communications and longer flight times. Such developments create the need for redundant positioning and localization systems that would enable UAVs to operate in GPS-denied areas.

UAVs can be equipped with a variety of sensors, and it is expected that at least one camera is present. Therefore, vision-based auxiliary localization systems should be usable by most UAVs, compared to systems that make use of sensors other than camera. An example of a visual localization solution can be a system that compares images taken by the on-board camera with parts of a pre-loaded two-dimensional map of the flight area. Then, the part of the map that best matches the camera view is a likely approximation of the true location.

Numerous strategies for visual UAV localization have been proposed, most of them built around Monte Carlo Localization algorithm (MCL) [5]. The strategies introduce novel ways of making and measuring observations for MCL. In this work, we explore the efficiency of observation methods that make use of supervised terrain classification and object detection. We also propose measurement models for such observations.

The robustness of the observation method is heavily dependent on the sensors and techniques applied in the model. For instance, observation models based on image processing suffer with variations in lighting, colors, image resolution,

The paper is organized as follows. Section II presents some works that are related to our research. Section III introduces the challenges of finding and processing the required data for a machine learning based measurement model. Section IV introduces our proposal for the measurement model. Section V describes the setup used for the experiments, including datasets, and results supporting our choices. Finally, Section VI discusses the implication of our strategy, both in terms of accuracy and efficiency, together with future directions.

II. Related Work

There are related works that propose solutions for visual UAV localization, using different types of measurement mod-

els, and there are related works that present different ways of processing aerial imagery using machine learning. We first present the works related to the localization problem, and then the works related to the observation model. Additionally, there are works which present novel ways to handle the visual localization problem with machine learning all together in one model, and this is presented as an argument for why to use machine learning in visual localization.

The current state-of-the-art for visual localization of UAVs can be argued to be the work done by Mantelli et al. [16]. This work uses the abBRIEF descriptor (BRIEF was first introduced by Calonder et al. [3]) to compare the image from the downward-facing camera of the UAV, with different patches from the satellite image, to estimate the robots pose - position, altitude and orientation - by use of an iterative particle filter algorithm, namely the Monte Carlo Localization algorithm [5]. The work optimizes the descriptor by running it for different color-channel representations of the images. The high performance of the localization is partly due to how abBRIEF is a relatively cheap operation. This allows for a high number of particles, which may lead to an increased rate of convergence. However, as abBRIEF compares color intensities between pairs of pixels, the method suffers from variation in tone which is inevitable due to changing weather conditions. In addition, particles that have a incorrect pose might still be given a high weight, as similarities between color intensities might still be found. Other works on localization, Masselli et al. [17], uses preprocessed maps and measurement models that are more robust towards variations in lighting conditions. The work uses supervised learning to train terrain classifiers for their measurement model, using labeled terrain data. Masselli uses Random Forests [2] which were trained on descriptor data using ORB [21] and TSURF [13]. There are works which uses Convolutional Neural Networks [14] for terrain classification of drone images, Khan [13] and Hudjakov [8]. Hudjakov does classification to provide path planning for a unmanned ground vehicle, but the classification itself seems promising for use in localization of a UAV, as it seems robust against environmental changes.

To use terrain classification for localization of a UAV the classifier need to be work for both satellite imagery and drone imagery. Works on terrain classification of satellite imagery are done by Helber et al. [6] and Basu et al. [1]. These works provide labeled datasets with labeled terrain classes, and discuss the challenges of classifying satellite images. Namely the lack of good datasets and the high variability inherent in satellite data, which may result in misclassifications. The works compare the accuracy of various network architectures, and they also compare to random forest classifiers. We have based our terrain classifier on these works.

The other measurement model proposed in our work is based on object detection. The detector used is the You Only Look Once model (YOLOv3), proposed by Redmon et al [19]. YOLOv3 is state-of-the-art in terms of detection speed, compared to region based object detectors like R-CNN [20]. The algorithm is supposed to be light enough to work real-time on light computation devices, like smartphones, so this is thought to be ideal to run on a robot with limited hardware constraints [18].

	Length (m)	Altitude (m)	Images
Flight 1	1,800	35-130	358 (1FPS)
Flight 2	2,400	40-300	500 (1FPS)

TABLE I: Parameters of the UAV flights.

Fig. 1: Map 1, Arroio do Meio, January, 2014

III. Data overview

The data used in this work differ from origin and use. Satellite images data-sets are used as supervised learning data and other data is used to perform offline tests.

A. UAV Flight Data

We reuse data of two real UAV flights from a past work to assess the performance of our algorithms [16]. The recorded flights were over the city of Arroio do Meio (Rio Grande do Sul municipality, Brazil), with the ground truth provided by GPS. The flight parameters are stated in Table I.

Several maps of the flight area in Arroio do Meio are included in the data set, each covering 1160×1160 meters and represent the exact same area. The only changes between the images are the contrast, sharpening and color correction. We use three of these: Map 1 was recorded in January, 2014. Map 2 is from March, the same year and Map 3 is from October 2010, see figure 1.

B. Supervised Learning Data

Acquisition of appropriate data is an unavoidable challenge for any supervised learning application. In particular, we require aerial images that are comparable to both the camera images produced by the drone and the global map of the area in which the drone navigates. The drone's altitude is not fixed, meaning that the spatial resolution of the resulting camera

images is not fixed either, further magnifying the difficulty of finding the appropriate data. Moreover, the data labels should ideally describe robust features that are frequently present in the flight area and do not change their properties or location over long periods of time.

We considered a number of data sets, but had to discard most of them for the above reasons. Some data sets had inappropriate spatial resolution (over one meter per pixel), while others focused too much on moving objects, such as cars. There are two data sets that showed promise: SAT6, which is suitable for terrain classification [1] and XView, which is appropriate for object detection [10].

With its 405,000 samples, SAT6 is a very comprehensive data set of six terrain classes: buildings, barren land, trees, grass, roads and water. Though terrain may be affected by human activities or seasonal changes, these classes still seem sufficiently robust. Each sample in the data set is a patch of 28×28 pixels, captured at a one meter per pixel resolution.

xView is a data set of high-resolution satellite images (0.3 meters per pixel). It has been made for object detection applications and allow the detection of 60 different classes. With more than 1 million objects over 1,400 km^2 of imagery, xView helps the training model to detect an object in different environments. Interesting to note is the fact that the xView images have been through ortho-rectification, pan-sharpening, atmospheric-correction and RGB dynamic range adjustments. The ortho-rectification implies geometric correction of the image depending on the tilt of the photograph and on topographical variations in the surface of the earth, meaning that some objects can end up bent after the correction, which is not the case for a drone capturing images downwards.

IV. Methods and Models

Visual localization is a complex problem with multiple subtasks: extraction of information from camera images and the map, pose hypothesis generation and hypothesis assessment. This section focuses on discussion of the methods used to solve each of the sub-tasks.

We use MCL as the cornerstone of our localization solution. Often referred to as Particle Filter localization, this algorithm earned its recognition due to its ability to represent a multimodal belief, meaning that it can generate multiple pose hypotheses at each time-step.

The algorithm initially spreads a set of particles (pose hypotheses) $\mathbf{P} = \{\mathbf{p}_1, ..., \mathbf{p}_N\}$ in a uniform manner. The particles are then assigned equal weights $\mathbf{w} = \{\frac{1}{N}, ..., \frac{1}{N}\}$ that represent their estimated probability of being the true state. At each time-step t, the algorithm updates every particle $\mathbf{p}_i = (x, y, z, \theta)$ based on robot's motion data $\mathbf{u}_t$, while sensor data $\mathbf{z}_t$ is used to calculate new weights $\mathbf{w}_t$. Finally, the particles are resampled with replacement according to their weights. By iterating through the update and resampling operations, the filter eventually converges to a narrow distribution close to the real pose of the robot. However, MCL cannot perform on its own without models that provide it with $\mathbf{u}_t$ and $\mathbf{w}_t(\mathbf{z}_t)$ at every time-step.

The motion data $\mathbf{u}_t$ are normally obtained via odometry. Vision-based odometry methods have been researched to a great degree and even successfully fielded in the Mars Exploration Rovers, thereby proving their effectiveness [15]. Such methods can be divided into two categories: monocular visual odometry methods that rely on a single camera and stereo visual odometry methods that rely on multiple cameras. As civilian UAV rarely have more than one camera installed, a monocular solution is more feasible. We opted for an odometry procedure that tracks ORB features throughout camera images and calculates their sparse optical flow using the iterative implementation of the Lucas-Kanade method with Gaussian pyramids. While this method is likely to be less precise than its more sophisticated counterparts, it does not rely on intrinsic parameters of the camera. Without the added need for camera calibration, motion estimation via optical flow is more universally available.

While the sensor data $\mathbf{z}_t$ for visual localization methods are often the images taken by the robot's on-board camera, it may be beneficial to add an image processing stage. We investigate how application of two supervised learning techniques to camera images affects localization. The techniques in question are terrain classification and object detection.

A. Terrain Classification

A relatively simple convolutional neural network, see figure 2, is trained on SAT6 dataset, achieving 95.71% accuracy on validation data. The learning model with the saved weights can then be applied to the flight area map and the UAV camera images, after they are divided into 28×28 pixel patches. Ideally, the network should be able to work with these data and output a six-dimensional vector of terrain class certainties. One issue that may negatively impact the prediction quality is the disparity in spatial resolution of images: SAT6 patches have a 1 meter per pixel resolution, the global maps are close to 25 centimeters per pixel, while camera images are usually between 7 and 15 centimeters per pixel, depending on the drone's height. Although resizing the map can equalize the spatial resolution, it also increases the real area covered by each patch and adds some uncertainty to localization, even if the correct map patch is identified.

B. Object Detection

The object detection model we used is YOLOv3 from Redmon et al. [19]. Its network is made from top level convolutional layers that extract features from the image, and the anchor boxes for the objectness prediction mechanism. When training YOLOv3 with a specific dataset, one has to perform a pre-definition of the prior anchor sizes for each class. To this extent, we used the anchors k-means clustering on xView data developed by Ultralytics LCC under MIT licence. [10]. With this configuration, the validation reached a precision of 55%, recall of 69% and mAP of 16%. Knowing that xView contains 60 classes, a mAP of 16% can be sufficient for our use case, especially when more than a half of the

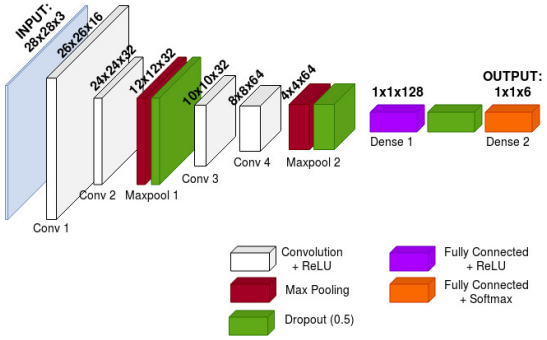


Fig. 2: Convolutional Neural Network for classification of SAT6 dataset. Validation accuracy: 95.71%

classes from xView are irrelevant for us and consequently, misleads the mAP interpretation. Moreover, as the training data has a fixed spatial resolution of 30cm per pixel and specific corrections, the images taken from the drone in flight will have to be processed.

C. Measurement Models

A measurement model in the context of Monte Carlo Localization algorithm is a model for comparing sensor data (image predictions, in our case) with particle observations. Particle observations are sampled from a datastructure containing predictions. Based on the comparison, a weight is assigned to each particle via a function of choice. Since predictions of images retain some spatial information, conventional error functions like Error Sum of Squares would not work directly, unless the prediction tensors are perfectly aligned. Thus, the selected measurement model needs to either be somewhat robust against geometric transformations or completely deal away with spatial information.

There are a number of point descriptors, such as BRIEF, that perform well under distortions. BRIEF takes in random pixel pairs and compares their intensities. If the first pixel in the pair has higher intensity than the other pixel, the descriptor stores a "1" for that pixel pair. Otherwise, it stores a "0". We can directly calculate a BRIEF descriptor for the prediction tensor of the camera image and that of the area where a particle is located. Since BRIEF is normally calculated for grayscale images, we have to evaluate each channel separately (there is one "color channel" for each prediction class). Then, once we obtain descriptors for the camera image prediction and a particle area prediction, we can count the number of times the descriptor elements matched each other (with XNOR operation). After normalization (division by number of pixel pairs n), the result can be assigned the as the weight for the inspected particle:

$$\mathbf{d}(\hat{\mathbf{I}}, \hat{\mathbf{P}}) = \mathrm{BRIEF}(\hat{\mathbf{I}}) \bigodot \mathrm{BRIEF}(\hat{\mathbf{P}}) \tag{1}$$

$$w_{brief} = \sum_i \frac{\mathbf{d}_i(\hat{\mathbf{I}}, \hat{\mathbf{P}})}{n} \tag{2}$$

where $\hat{\mathbf{I}}$ is the prediction tensor for a given camera image and $\hat{\mathbf{P}}$ is the prediction tensor for the area of a given particle.

Another possible measurement model can be based on disregarding the spatial relationship between the predictions and simply counting the total number of instances for each prediction class:

$$\mathbf{s}(\hat{\mathbf{I}}) = \left(\sum_{i,j} \hat{\mathbf{I}}_{i,j,1}, \quad ..., \quad \sum_{i,j} \hat{\mathbf{I}}_{i,j,C} \right) \tag{3}$$

$$\mathbf{s}(\hat{\mathbf{P}}) = \left(\sum_{i,j} \hat{\mathbf{P}}_{i,j,1}, \quad ..., \quad \sum_{i,j} \hat{\mathbf{P}}_{i,j,C} \right) \tag{4}$$

where $\mathbf{s}(\hat{\mathbf{I}})$ and $\mathbf{s}(\hat{\mathbf{P}})$ are vectors of class instance sums for a given camera image and a given particle area respectively, while C is the number of prediction classes.

Thus, the result for each prediction tensor is a vector with one element per prediction class k. Unlike tensors, vectors of class instance sums are reasonable to compare through Error Sum of Squares (SSE), since the elements cannot be misaligned or distorted. We add a normalization step (division of each term by the total number of detected instances) in order to avoid large values:

$$\mathrm{SSE}(\hat{\mathbf{I}}, \hat{\mathbf{P}}) = \sum_{k=1}^{C} \left(\frac{\mathbf{s}_k(\hat{\mathbf{I}})}{\sum_{k=1}^{C} \mathbf{s}_k(\hat{\mathbf{I}})} - \frac{\mathbf{s}_k(\hat{\mathbf{P}})}{\sum_{k=1}^{C} \mathbf{s}_k(\hat{\mathbf{P}})} \right)^2 \tag{5}$$

But only comparing the number of elements of a certain class in the images is not enough, as the object detection implies a lot of other parameters that can be studied. For example, we can adjust the measurement model by taking into account the total number of elements or the proportion of houses over the rest. With a few modifications, the reciprocal of the result can be used as a particle weight:

$$w_{sse} = \left(\left| \ln \frac{\sum_{k=1}^{C} \mathbf{s}_k(\hat{\mathbf{I}})}{\sum_{k=1}^{C} \mathbf{s}_k(\hat{\mathbf{P}})} \right| \left| \ln \frac{s_a}{p_a} \right| \mathrm{SSE}(\hat{\mathbf{I}}, \hat{\mathbf{P}}) + 1 \right)^{-1} \tag{6}$$

With a referring to the architecture type objects, namely buildings and facilities, and where the cases of the denominators being equal to zero is managed by the software implementation. Incrementing the denominator by one simultaneously prevents possible division by zero and forces the result into $(0, 1]$ range. The ratio of instance sums augments the function in a way that takes the population density of prediction instances into account. Since the number of detected instances grows exponentially with elevation of the camera, we soften the density factor by introducing the natural logarithm.

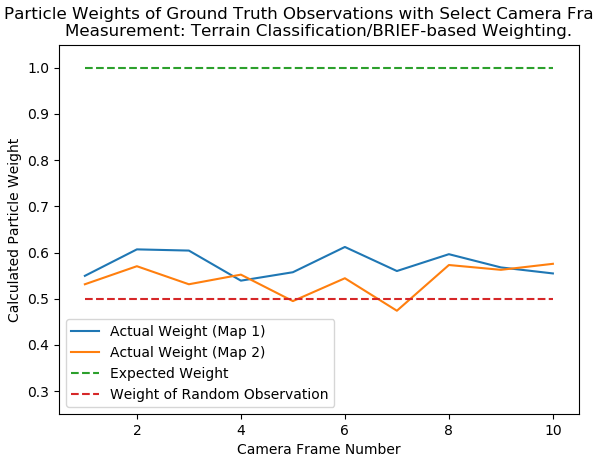


Fig. 3: Terrain classification/BRIEF-based measurement: Particle weights at ground truth UAV poses.

V. Experiments and Results

We separately tested the algorithm performance with observation models based on both terrain classification and object detection. Due to the lack of labels on camera images and global maps, it was not possible to test the observation models directly. However, indirect assessment could be performed when using the observations in conjunction with measurement models and ground truth flight trajectories.

A. Experiments with Terrain Classification

We began by testing the observations made with the terrain classification method, applying the BRIEF-based measurement model. An MCL particle was placed in a known ground truth UAV pose. Then, the observation made by the particle was compared to the corresponding observation made by the UAV camera, and the particle weight was calculated via the measurement function. The results for ten such observations can be seen in Figure 3.

Our BRIEF-based measurement function has the upper limit $w_{max} = 1$ (if the descriptors match completely), and the comparison of two unrelated observations should produce $w_{random} = 0.5$, because BRIEF works as a "greater than" operator, rather than a difference operator. From the experiment results, we observed that the actual particle weights exceeded this w_{random} value by a very small margin. Furthermore, one observation produced a particle weight even lower than w_{random}. Such development could only indicate that there was a fault in our observation model. To substantiate our suspicions, we ran the localization algorithm through the full offline test of Flight 1, using Map 1 and Map 2. The mean trajectory errors over five runs were recorded and as we can see on figure 4, the results is far from acceptable, which confirms our previous stated assumptions.

B. Experiments with Object Detection

The experiment lead for evaluating the SSE-based measurement model went as shown in Figure 5:

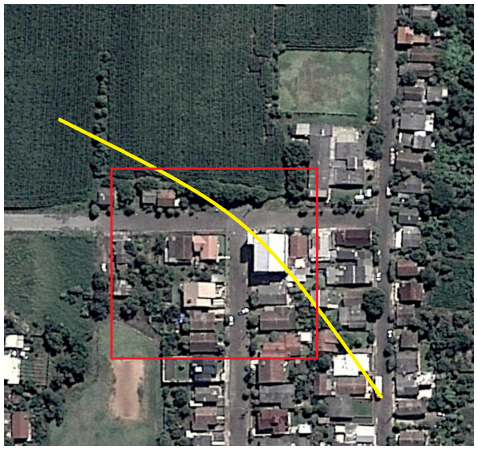


Fig. 5: Object Detection measurement: Particle weights at ground truth UAV poses.

The UAV follows a trace, flying over a specific region of the map. The general idea is that this map area is similar to the image a particle would generate. The drone's trajectory is an actual portion of the recorded flight that was performed over the map presented in figure 1. If the measurement model is well tuned, we expect an increase in weight for the first images and a decrease for the latest, the weight values for the center images being close to 1. The test was run only on Map2 and Map3, as Map1 image corrections (especially the pan-sharpening) made it far different from the data we trained our model with. Thus, as expected, the weight values behave correctly: slowly increasing and decreasing as the drone gets closer to the particle's position and goes away.

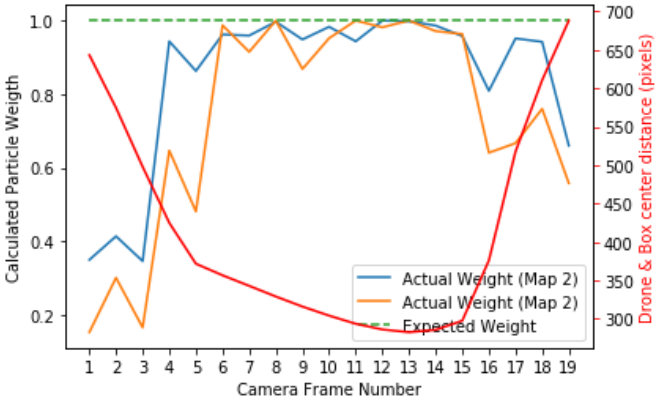
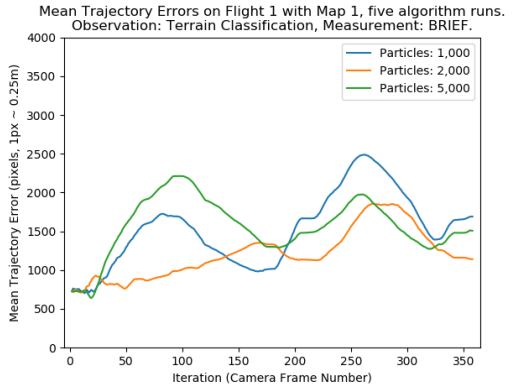
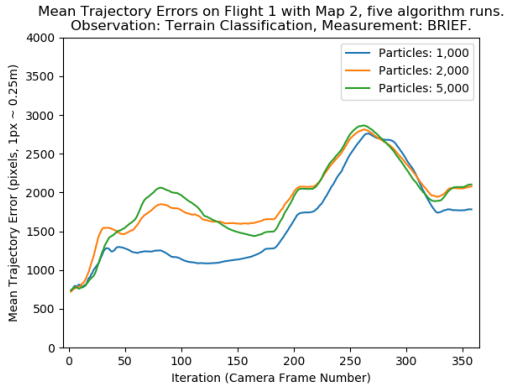


Fig. 6: Object Detection measurement: Particle weights at ground truth UAV poses.

As the measurement model works as expected on this particular test, the next step would be to integrate it to the MCL. Though, the overall object-detection method could be improved in some ways. The main issue is the object detection itself. A solution would be to create a data-set very similar to xView but with more relevant classes. Another way would be to have a more stable trained YOLO. This can easily been done by training it with better configurations than the ones we used.



(a) Trajectory errors on **Flight 1** with **Map 1**



(b) Trajectory errors on **Flight 1** with **Map 2**

Fig. 4: Mean trajectory errors on **Flight 1** with both maps, terrain classification, **BRIEF**-based particle measurement. Five algorithm runs. MCL particles used: 1,000, 2,000 and 5,000

Finally, the measurement model we proposed is far from being perfect, but still demonstrates how good is all the information provided by object detection. Thus, many parameters are yet to be used to discover their potential, such as the object's positions, their sizes etc...

VI. Conclusion & Future Work

The datasets used to train the network must be optimised for classification of both satellite imagery and drone imagery, in which there is significant differences in how the world is represented in terms of angles, colors, contrast, sharpness and resolution. The choice of classes used for training the model must be optimized to represent only the objects that are easiest to classify, and the number of classes must be limited to provide a faster classifier.

There are a lot of parameters to be tuned, for the measurement model. Among them are the weights given to the occurrence of each class, and of much of the particle image to be compared to the reference image. There is also the question of how "fine grained" the preprocessed map should be. A genetic algorithm might be suitable for optimizing the measurement model in terms of the parameters mentioned above, and this in turn might be a way to leave more of the optimization to the computer, rather than to the programmer.

A recurrent thought when using deep learning for localization is to let the network include the particle filter. This is done by Karkus et al. [12], however, they report that a major limitation of this approach is the amount of computation needed for the larger network. For future work we would like to find a balance between using machine learning and traditional particle filters, supposedly using Evolutionary Algorithms [7] to optimise the parameters of the measurement model in the particle filter.

Finally, there is the question of whether supervised learning is the best approach to go for when using machine learning in the measurement model. As mentioned earlier, there is a limitation to how well made the dataset can be in terms handling differences between satellite, and drone, imagery. A better solution might be to use unsupervised learning, that is clustering or segmentation, in the measurement model. Information Invariant Clustering [9] has been proven to yield rather good classification.

References

[1] Saikat Basu et al. "DeepSat: a learning framework for satellite imagery". In: *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '15*. the 23rd SIGSPATIAL International Conference. Bellevue, Washington: ACM Press, 2015, pp. 1–10.

[2] Leo Breiman. "Random Forests". In: *Machine Learning* 45.1 (Oct. 1, 2001), pp. 5–32.

[3] Michael Calonder et al. "BRIEF: Binary Robust Independent". In: ().

[4] James V. Carroll. "Vulnerability Assessment of the U.S. Transportation Infrastructure that Relies on the Global Positioning System". In: *Journal of Navigation* 56.2 (May 2003), pp. 185–193.

[5] F. Dellaert et al. "Monte Carlo localization for mobile robots". In: *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*. International Conference on Robotics and Automation. Vol. 2. Detroit, MI, USA: IEEE, 1999, pp. 1322–1328.

[6] Patrick Helber et al. "EuroSAT: A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 12.7 (July 2019), pp. 2217–2226.

[7] Shinn-Ying Ho, Li-Sun Shu, and Jian-Hung Chen. "Intelligent evolutionary algorithms for large parameter optimization problems". In: *IEEE Transactions on Evolutionary Computation* 8.6 (Dec. 2004), pp. 522–541.

[8] Robert Hudjakov and Mart Tamre. “Aerial imagery terrain classification for long-range autonomous navigation”. In: *2009 International Symposium on Optomechatronic Technologies*. 2009 International Symposium on Optomechatronic Technologies (ISOT 2009). Istanbul, Turkey: IEEE, Sept. 2009, pp. 88–91.

[9] Xu Ji, João F. Henriques, and Andrea Vedaldi. “Invariant Information Clustering for Unsupervised Image Classification and Segmentation”. In: *arXiv:1807.06653 [cs]* (Aug. 22, 2019). arXiv: 1807.06653.

[10] Glenn Jocher. *dust: xView 2018 Object Detection Challenge: YOLOv3 Training and Inference.* 2018.

[11] Christoforos Kanellakis and George Nikolakopoulos. “Survey on Computer Vision for UAVs: Current Developments and Trends”. In: *Journal of Intelligent & Robotic Systems* 87.1 (July 1, 2017), pp. 141–168.

[12] Peter Karkus, David Hsu, and Wee Sun Lee. “Particle Filter Networks with Application to Visual Localization”. In: *arXiv:1805.08975 [cs, stat]* (Oct. 25, 2018). arXiv: 1805.08975.

[13] Yasir Niaz Khan, Andreas Masselli, and Andreas Zell. “Visual terrain classification by flying robots”. In: *2012 IEEE International Conference on Robotics and Automation*. 2012 IEEE International Conference on Robotics and Automation. ISSN: 1050-4729. May 2012, pp. 498–503.

[14] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. In: *Nature* 521.7553 (May 2015), pp. 436–444.

[15] Mark Maimone, Yang Cheng, and Larry Matthies. “Two years of Visual Odometry on the Mars Exploration Rovers”. In: *Journal of Field Robotics* 24.3 (2007), pp. 169–186.

[16] Mathias Mantelli et al. “A novel measurement model based on abBRIEF for global localization of a UAV over satellite images”. In: *Robotics and Autonomous Systems* 112 (Feb. 2019), pp. 304–319.

[17] Andreas Masselli, Richard Hanten, and Andreas Zell. “Localization of Unmanned Aerial Vehicles Using Terrain Classification from Aerial Images”. In: *Intelligent Autonomous Systems 13*. Ed. by Emanuele Menegatti et al. Vol. 302. Cham: Springer International Publishing, 2016, pp. 831–842.

[18] Matija Radovic, Offei Adarkwa, and Qiaosong Wang. “Object Recognition in Aerial Images Using Convolutional Neural Networks”. In: *Journal of Imaging* 3.2 (June 2017), p. 21.

[19] Joseph Redmon et al. “You Only Look Once: Unified, Real-Time Object Detection”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA: IEEE, June 2016, pp. 779–788.

[20] Shaoqing Ren et al. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.6 (June 1, 2017), pp. 1137–1149.

[21] Ethan Rublee et al. “ORB: An efficient alternative to SIFT or SURF”. In: *2011 International Conference on Computer Vision*. 2011 IEEE International Conference on Computer Vision (ICCV). Barcelona, Spain: IEEE, Nov. 2011, pp. 2564–2571.

[22] Mingguo Zheng et al. “Rotation and affine-invariant SIFT descriptor for matching UAV images with satellite images”. In: *Proceedings of 2014 IEEE Chinese Guidance, Navigation and Control Conference*. Proceedings of 2014 IEEE Chinese Guidance, Navigation and Control Conference. ISSN: null. Aug. 2014, pp. 2624–2628.