INTERNSHIP REPORT

Real-time Change Detection Between Drone Video and 3D Reference Model

MATEOS Esteban

Applied Mathematics and Computer Sciences

01/2023 – 07/2023

School supervisor: Fabien Vergnet

Company supervisor: Emilie Le Moël

Abstract

This report presents the findings and outcomes of my internship, focusing on the feasibility of real-time change detection using a 3D reference model and drone-captured videos. The objective was to rapidly identify changes in the environment, such as fallen trees or suspicious objects.

During the familiarization phase, extensive research was conducted on existing algorithms and feature detection and matching techniques. This involved studying various approaches to identify keypoints in images, which were then used for matching and generating differences between images.

The internship involved working with drone-captured videos, a 3D model generated through photogrammetry, and telemetry data. The alignment of the 3D model and video frames enabled the creation of an image database for further analysis. Challenges, such as geographic offsets and manual corrections, were encountered during this process.

To address the limitations of existing algorithms, the Segment Anything Model (SAM) was explored. SAM, a promptable segmentation model, provided high-quality object masks and flexible segmentation based on specific prompts. The SAM model was utilized to identify and compare objects in the images.

The comparison process involved masking, sorting masks based on area and color, and determining differences between the reference image and video frames. Results were visually presented, highlighting the detected differences on an interactive 3D map.

Throughout the internship, various challenges were encountered, including image differences, a unique perspective, limited datasets, and software utilization. Collaboration and knowledge-sharing within the company played a crucial role in overcoming these challenges.

Furthermore, GPU computing challenges were faced due to the resource-intensive nature of the SAM model. Limitations in accessing CUDA on the personal computer resulted in time losses during mask generation. The computational demands of SAM made it impractical to embed the solution in a drone, necessitating a division of tasks between onboard real-time image generation and comparison on a more powerful system.

In conclusion, the internship provided valuable insights into real-time change detection using 3D models and drone-captured videos. The study highlighted the effectiveness of the SAM model for promptable segmentation. The findings contribute to understanding image comparison techniques and the challenges associated with utilizing big machine learning models in resource-constrained environments.

Résumé

Ce rapport présente les résultats et les conclusions de mon stage, axé sur la faisabilité de la détection en temps réel des changements à l'aide d'un modèle de référence en 3D et de vidéos capturées par des drones. L'objectif était d'identifier rapidement les changements dans l'environnement, tels que des arbres tombés sur la route ou des objets suspects.

Pendant la phase de familiarisation, j'ai effectué des recherches approfondies sur les algorithmes existants et les techniques de détection et de correspondance des caractéristiques. Cela a impliqué l'étude de différentes approches pour identifier les points clés dans les images, qui ont ensuite été utilisés pour générer les différences entre les images.

Le stage a impliqué de travailler avec des vidéos capturées par des drones, un modèle 3D généré par photogrammétrie et des données de télémétrie. L'alignement du modèle 3D et des images vidéo a permis de créer une base de données d'images pour une analyse ultérieure. Des défis tels que les décalages géographiques et les corrections manuelles ont été rencontrés au cours de ce processus.

Pour pallier les limitations des algorithmes existants, le modèle "Segment Anything" (SAM) a été exploré. SAM, un modèle de segmentation pouvant être commandé, a fourni des masques d'objets de haute qualité et une segmentation flexible en fonction de commandes spécifiques. Le modèle SAM a été utilisé pour identifier et comparer les objets dans les images.

Le processus de comparaison impliquait la création de masques, le tri des masques en fonction de leur surface et de leur couleur, et la détermination des différences entre l'image de référence et les images vidéo. Les résultats ont été présentés visuellement, mettant en évidence les différences détectées sur une carte 3D interactive.

Tout au long du stage, divers défis ont été rencontrés, notamment des différences d'image, une perspective unique, des ensembles de données limités et l'utilisation de logiciels. La collaboration et le partage des connaissances au sein de l'entreprise ont joué un rôle crucial pour surmonter ces défis.

De plus, des problèmes liés à l'utilisation des GPU et à CUDA ont été rencontrés en raison de la nature gourmande en ressources du modèle SAM. Les limitations d'accès à CUDA sur mon ordinateur personnel ont entraîné des pertes de temps lors de la génération des masques. Les exigences de calcul de SAM rendaient impossible l'intégration de la solution dans un drone, ce qui nécessitait une répartition des tâches entre la génération d'images en temps réel à bord du drone et la comparaison sur un système plus puissant.

En conclusion, ce stage a fourni des informations précieuses sur la détection en temps réel des changements à l'aide de modèles 3D et de vidéos capturées par des drones. L'étude a mis en évidence l'efficacité du modèle SAM pour la segmentation commandée. Les résultats contribuent à la compréhension des techniques de comparaison d'images et des défis associés à l'utilisation de modèles d'apprentissage automatique complexes dans des environnements contraints en ressources.

Acknowledgements

I wish to express my deepest gratitude to all who made my internship at Carmenta Geospatial Technologies such a valuable and enriching experience.

I am thankful for the guidance and support provided by my supervisors, Emilie Le Moël and Fabien Vergnet. Emilie's expertise and encouragement have been pivotal in my professional growth, and the trust and autonomy granted by Polytech significantly contributed to my development during the internship.

Particularly, I want to acknowledge Jonas Envall and Annika Kirschner for their invaluable assistance during our weekly meetings, where they generously shared their expertise.

My appreciation extends to all my colleagues at Carmenta. Your willingness to share knowledge, collaborate, and provide feedback has been truly invaluable.

Furthermore, I would like to acknowledge the management and staff, with a special mention of Carl-Johan Lundell, for providing me with this opportunity and fostering a positive and conducive work environment.

Lastly, I would like to thank all those who may not be directly mentioned but have played a part in my internship journey at Carmenta.

Content

Introduction

This report presents the findings and conclusions of my internship at Carmenta Geospatial Technologies, focusing on the feasibility study of real-time detection of environmental changes using a 3D reference model and videos captured by drones. This topic is important in the current context where timely surveillance and analysis of environmental changes are crucial for applications such as public safety, natural disaster management, and optimization of industrial operations.

Drones have become versatile tools for capturing video footage over vast territories. However, it is often critical to quickly identify significant changes in these videos, such as the appearance of suspicious objects or alterations in the environment, in order to take preventive measures or optimize real-time operations. The detection of these changes in videos can be achieved by comparing the real-time video stream with a pre-generated 3D reference model of the same geographic area.

In this context, my mission during this internship was to study the feasibility of this approach, focusing specifically on performing change analysis on board the drone itself. The objective was to develop a solution that would enable the rapid detection of important environmental changes as soon as the drone captured the video or even in real-time during the drone's flight.

Real-time detection of environmental changes offers numerous advantages. It allows for a prompt response to emergency situations, such as traffic accidents or natural disasters, by immediately alerting the relevant authorities. It can also contribute to incident prevention by quickly detecting obstacles or suspicious objects on roads or in restricted areas. Furthermore, this technology can be used for continuous surveillance of industrial sites, detection of changes in infrastructure, and accurate data collection for urban planning purposes.

Studying the feasibility of this approach also presents interesting technical and technological challenges. It requires the development of efficient algorithms for real-time video and reference model comparison, while considering the computational and storage constraints of drones. Additionally, the use of 3D reference models necessitates precise synchronization of telemetry data and a deep understanding of photogrammetry techniques.

Throughout the internship, extensive research, experimentation, and algorithm development were undertaken to address these challenges. As a result, a solution has been developed, demonstrating the potential of real-time detection of environmental changes using a 3D reference model and drone videos.

Hence, this topic is particularly relevant and intriguing to explore in the field of computer vision and artificial intelligence as it combines advanced technical aspects with practical real-world applications. The findings from this study could have a significant impact on the development of real-time detection systems for environmental changes, thereby enhancing safety, operational efficiency, and decision-making in various application domains.

Company Presentation

During my end-of-studies internship, I had the opportunity to work at Carmenta Geospatial Technologies, a company founded in 1985 and a leading player in the geospatial technology field. Carmenta offers cutting-edge software solutions used in critical applications worldwide.

Carmenta started with a project initiated to meet the real-time management needs of geospatial data for the new Swedish Air Force fighter jets. This demanding requirement led to the creation of Carmenta Engine, a powerful software for geospatial data visualization and analysis. Since its establishment in the 1990s, Carmenta has continued to develop its product, which is now operationally used in various fields.

Carmenta provides a comprehensive range of products and services for geospatial data visualization and analysis. Their flagship product, Carmenta Engine, is a versatile software development kit (SDK) that enables the creation of high-performance 2D/3D geospatial applications. Carmenta Engine supports multiple programming languages and graphical user interfaces, allowing customers to choose the ones that best suit their needs. It is possible to develop applications using languages such as C++, .NET, Java, and Python, and select from a variety of graphical user interface libraries such as WPF, WinForms, Qt, OpenJDK, and Xamarin. Furthermore, thanks to its cross-platform design, the same code can be executed on Windows, Linux, and Android.

Carmenta's software solutions are used in critical applications across various sectors, including defense (land, air, sea), transportation, public safety, and automotive. Their clients, located worldwide, include demanding players in these industries, such as Airbus, Atos, Thales, and the Swedish Transport Administration (Trafikverket).

The company has a presence in five different countries, with offices in Gothenburg and Stockholm in Sweden, Munich in Germany, Paris in France, Reading in England and Washington DC in the United States.

Carmenta Engine

Carmenta Engine is a software development kit (SDK) designed to develop high-performance, interactive 2D and 3D geospatial applications on different platforms. It offers a range of features and benefits that make it ideal for mission-critical systems requiring advanced maps and situational awareness.

One of Carmenta Engine's key assets is its extensibility. It offers various extensions that enhance its functionality and enable developers to customize and extend its features.

For example, there's a tactical extension that lets you use tactical symbols and graphics, a nautical and aeronautical extension, a 3D extension that I use in my project, a vehicle analysis extension that lets you calculate the interactions of ground vehicles with terrain and roads. It also includes a terrain analysis extension that facilitates the calculation of terrain clearance and obstacles along flight routes. Other extensions are available and are all listed on the Carmenta website.

With its hardware-accelerated rendering and advanced data processing capabilities, Carmenta Engine enables rapid creation of visually impressive maps. Its real-time visualization and analysis capabilities enable dynamic data such as live video feeds and radar plots to be superimposed on maps, guaranteeing up-to-date situational awareness.

The SDK supports a wide range of geospatial data formats and provides seamless integration with file-based data sources and geospatial databases. It eliminates the need for data conversion, allowing developers to work with original formats and combine different types of data effortlessly.

Having served mission-critical applications for over 20 years, Carmenta Engine has a proven track record of reliability. It excels in dynamic data management and offers real-time geospatial analysis capabilities, enhancing the value and usability of geospatial data sources.

In conclusion, Carmenta Engine is a high-performance, reliable SDK that enables developers to create geospatial applications with advanced functionality. Its scalability, cross-platform compatibility, speed and support for different data formats make it an excellent choice for mission-critical systems requiring higher-level situational awareness.

The mission

Drones can be used to efficiently capture video footage over large areas. A post-processing mechanism called photogrammetry makes it possible to recreate 3D representation of that geographical area from such videos.

In certain scenarios it is important to identify changes in the environment as quickly as possible – ideally as soon as the drone has captured the video. For example, fast detection of a tree fallen on a road could prevent traffic blockage, or the drone video could be used to detect suspect objects or intruders in an area restricted to the public.

A possible way to do this would be to compare the new video stream in real-time against a previously generated 3D “reference model” of the same area.

The purpose of my internship was to study the feasibility of this approach, including performing the change analysis on board the drone itself and to develop a solution.

1/ Familiarization

Firstly, I familiarized myself with the subject by conducting research on existing algorithms and contemplating how I could utilize them. My research led me to the utilization of feature detection and feature matching.

Feature detection is a fundamental task in computer vision that involves identifying distinctive and meaningful structures or patterns in images. These structures, also known as features or keypoints, can be points, edges, corners, blobs, or any other localized structures that have unique characteristics. Feature detection plays a crucial role in various computer vision tasks, such as image matching, object recognition, tracking, and 3D reconstruction.

Feature detection algorithms aim to locate these keypoints based on specific criteria, such as their visual saliency, uniqueness, and invariance to transformations like scale, rotation, and illumination changes. These algorithms analyze the image's local properties, such as intensity gradients, color variations, texture patterns, or higher-order image derivatives, to identify regions of interest.

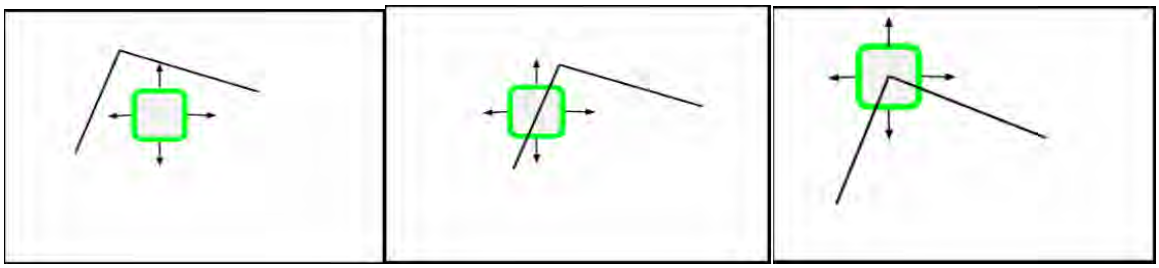
Corner detectors are utilized to identify points in an image that exhibit corner-like characteristics, typically points that display a significant response to a differential operator. The term "corner detectors" originated from the early stages of automated feature detection.

For example, one algorithm that I studied is the Shi-Tomasi Corner Detection that was published by J.Shi and C.Tomasi in their paper “Good Features to Track”.

In this algorithm, the fundamental idea is to identify corners by observing substantial changes in all directions.

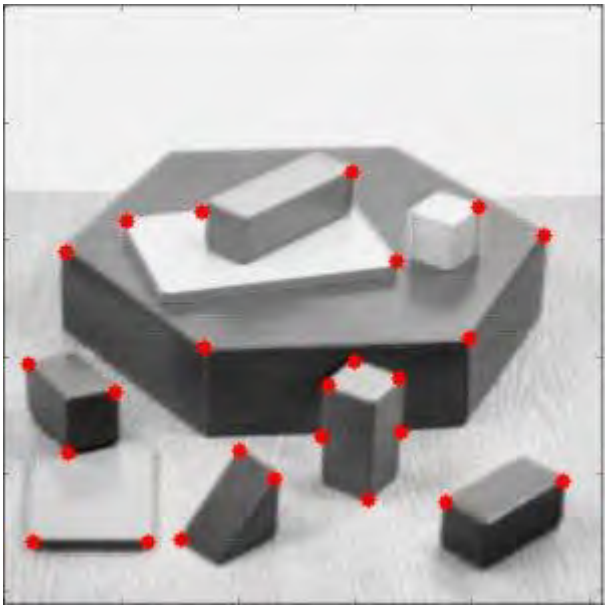
The approach involves selecting a small window within the image and systematically scanning the entire image in search of corners. When the small window is positioned over a corner, shifting it in

any direction will cause a noticeable alteration in the visual appearance of the contents within the window:



(1) Images from https://www.geeksforgeeks.org/python-corner-detection-with-shi-tomasi-corner-detection-method-using-opencv/

Here, on the left image, moving the window won't cause any change. On the middle one, there will be change on the axis perpendicular to the edge, but won't be any change along the edge. Finally, on the right, whatever the direction, changes will always be observed, so it's an edge.



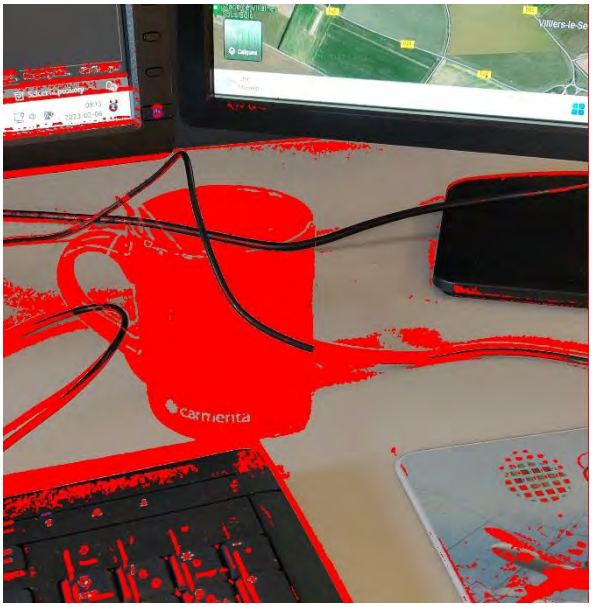
(2) Example of keypoint detection using this algorithm. Image from https://docs.opencv.org/3.4/d4/d8c/tutorial_py_shi_tomasi.html

After detecting the keypoints, a descriptor is computed for each keypoint by analyzing the local image gradients in its vicinity. This descriptor captures intensity gradient information and orientations, creating a robust representation of the keypoint.

The same process is performed for both images, allowing us to compare the keypoints and perform keypoint matching. The objective is to compare the descriptors of keypoints in different images to find correspondences using techniques such as nearest neighbor matching and ratio tests.

By applying this algorithm to pictures taken from my phone, I was able to overlap the images and utilize bitwise operations to display the differences between them.

This is the result obtained when I apply the process to two images from my phone: in one of the images, there was a cup that I removed in the second image. We can see in red the differences obtained:

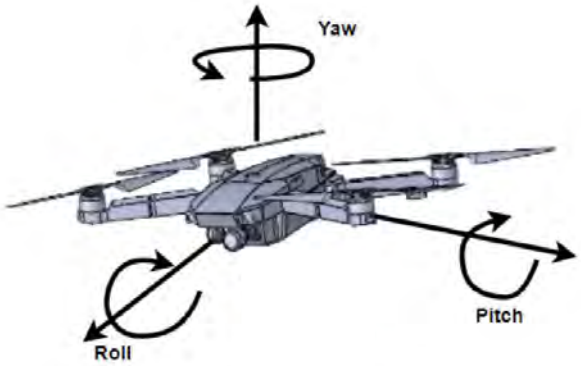


(3) Result of the utilization of the algorithm

2/ Utilization of Provided Equipment

Next, I familiarized myself with all the files and data that I would be using during my internship:

- A video recorded from a drone, capturing an aerial view of a farm.
- A 3D model of the same farm generated via photogrammetry a few months prior to the new video capture. Photogrammetry is a method that utilizes pictures to create precise three-dimensional models of objects, buildings, or landscapes.
- A CSV file containing all the telemetry data of the drone during the video capture. This file allows me to access the drone's position (longitude/latitude) for each millisecond, as well as the values for the pitch, roll, and yaw of the camera.



(4) Image from https://www.mdpi.com/2504-446X/6/4/98

With this data, I can load the 3D model into Carmenta Engine and position a camera at a specific point that allows me to capture images similar to those from the video. By configuring the virtual camera with the same settings as the actual camera (resolution, field of view, location, and camera orientation), I obtained the following result:

(5) On the left we have the frame from the video, and on the right side we have the projection in the 3D model.

The entire process has been automated, from video processing and metadata parsing to projecting onto the 3D model and capturing screenshots. This allowed me to generate a database of image pairs: one from the 3D model and one from the video.

During the analysis of several image pairs, it was discovered that there was a slight geographic offset between the video images and the ones generated by the 3D model. This offset could be attributed

to inaccuracies in the drone's information, a misconfiguration of the 3D model, or both. Fortunately, the offset was manually corrected by adjusting the model parameters. Unfortunately, no automatic solution was found for this correction.

3/ Algorithm Selection

When I tested the feature detection and matching algorithm previously explored on these images, it failed to produce satisfactory results. This algorithm is designed to compare similar images, taken from the same camera with consistent color shades: as mentioned earlier, the algorithm computes descriptors for each keypoint by calculating local gradients. However, due to significant differences between the two images, the computed gradients are vastly different. Consequently, the keypoint matching cannot be achieved as expected. Hence, I had to find or develop a new image comparison algorithm.

One common issue I encountered with existing algorithms was that, like for the feature detection and matching, existing algorithms expect similar images and color shades. These algorithms primarily focused on identifying missing objects or variations in angles, rather than images displaying the same location with different textures. As a result, the machine learning models and algorithms I tried did not produce satisfactory results.

Since these algorithms proved ineffective, I decided to take a different approach. I leveraged the fact that both images were captured from the exact same position, allowing me to focus on identifying and comparing the elements present in the images.

To pursue this new method, I initially explored the use of the "You Only Look Once" (YOLO) model. YOLO is a real-time object detection algorithm based on the Darknet framework.

Darknet is an open-source framework written in the C programming language. It provides the tools necessary for building and training convolutional neural networks (CNNs). Darknet is used for computer vision tasks, including object detection.

Unlike other object detection models that perform multiple passes over an image, YOLO takes a different approach: this algorithm divides the image into a grid of cells and predicts bounding boxes and object classes for each cell.

The process of object detection with YOLO involves three main steps:

1. Bounding Box Detection: YOLO predicts the bounding boxes that enclose the objects present in each grid cell. Each bounding box is represented by a set of coordinates (x, y, width, height).
2. Class Prediction: For each bounding box, YOLO predicts the class of the object it contains. These classes can include predefined object categories such as "car," "dog," "person," etc.
3. Prediction Filtering: The predictions of bounding boxes and classes are filtered using confidence thresholds to eliminate unreliable detections. The remaining predictions are then used to identify and locate the detected objects in the image.

The YOLO model is trained on large amounts of annotated data, where the objects of interest are marked with their corresponding bounding boxes and classes. It is important to have a representative and diverse dataset to achieve good detection performance.

However, during our testing, we encountered two main problems. Firstly, most of our images were captured from aerial viewpoints, whereas the model was initially trained on frontal images.

Consequently, the results were unsatisfactory, and we did not have a sufficiently large dataset to train a new model.

Furthermore, we observed that the model was only trained to recognize specific labeled objects. It was designed to identify pre-defined objects, such as cars and bikes. While this is useful for scenarios like road images, our goal was to detect any kind of object, regardless of whether the model was familiar with it. The objects could range from cars to random boxes in the middle of a street.

Therefore, we needed an agnostic model capable of recognizing various types of objects, rather than being restricted to specific objects.

That's where I found the Segment Anything Model, released in early 2023 by Meta AI Research under the Apache License 2.0. This pretrained model produces high quality object masks from input prompts such as points or boxes, and it can be used to generate masks for all objects in an image.

(6) Image from https://github.com/facebookresearch/segment-anything/

The Segment Anything Model (SAM) is designed for the task of promptable segmentation, where the goal is to generate valid segmentation masks given various types of prompts. SAM consists of three components: an image encoder, a prompt encoder, and a mask decoder.

The image encoder is based on a pre-trained Vision Transformer (ViT) that processes high-resolution inputs. It computes an image embedding, which represents the visual information of the input image.

The prompt encoder handles different types of prompts, such as points, boxes, and text. It encodes these prompts into embeddings that capture the relevant information for segmentation.

The mask decoder takes the image embedding and prompt embeddings as inputs and generates a segmentation mask. It uses a modified Transformer decoder block combined with a dynamic mask prediction head. The decoder block performs self-attention and cross-attention to update the embeddings, and the mask prediction head maps an output token to a foreground probability at each location in the image.

SAM is designed to handle ambiguity in prompts by predicting multiple masks for a single prompt. This allows the model to handle cases where the prompt could refer to multiple objects. During training, the model is trained to minimize the loss over the predicted masks.

To ensure strong generalization to new data distributions, SAM is trained on a large and diverse set of masks using a data engine. The data engine consists of three stages: assisted-manual, semi-automatic, and fully automatic. In the assisted-manual stage, annotators label masks with the help of SAM in an interactive segmentation setup. In the semi-automatic stage, automatic mask predictions are combined with manual annotation. In the fully automatic stage, SAM generates masks without any annotator input using a regular grid of points as prompts.

Overall, SAM provides a flexible and efficient solution for promptable segmentation, allowing for real-time interactive use and handling ambiguity in prompts.

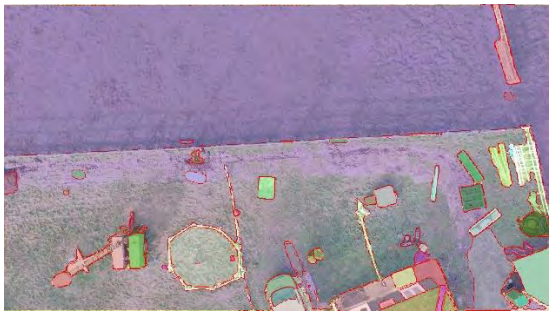
4/ Utilization of SAM

Segment Anything Model is used in two different ways in the image comparison process: first by generating the masks in the full image, then by using SAM only in specified positions of the image.

To illustrate the complete process, I'll apply all the steps using this image:

(7) Image used to show all the steps applied.

First, we initiate the process by generating masks from the original image using the Segment Anything Model (SAM). The primary objective of SAM is to segment all the elements present in the image, resulting in individual masks for each detected object. Subsequently, it becomes necessary to sort these generated masks. The purpose of this sorting step is to select masks that are relevant for the comparison process while eliminating those that are insignificant or could potentially distort the results. Thus, the mask sorting process enables a reduction in the number of masks to be considered, allowing us to focus on the elements of interest for our study.



(8) Masks generated with SAM.

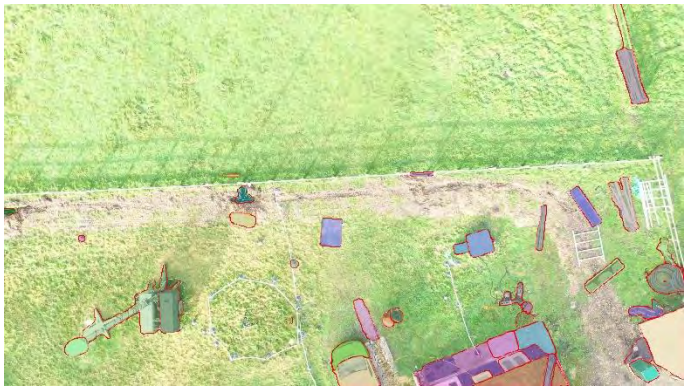
The first step of sorting involves determining whether to keep or reject masks based on their area. The rationale behind this approach is that some masks may represent unwanted regions in the image, such as the sky or vast grassy areas. In the context of my project, these regions are not relevant and do not require in-depth analysis. Therefore, we choose to eliminate large masks, which cover more than 20% of the whole image, as they typically correspond to these non-informative regions. Similarly, very small masks that reduce to mere points are also discarded. These extremely small masks are often unreadable and challenging to interpret in terms of their significance. By sorting the masks based on their area, we can eliminate masks that are either too large or too small, allowing us to focus on more meaningful regions of interest for our analysis.

(9) First sort based on area.

After the initial sorting based on area, the masks undergo another sorting process, which is optional and based on color. This step involves comparing the colors of the inner and outer contours of each mask. The goal is to identify and eliminate objects, such as trees, that may have distinct shading but are part of a larger context, such as a forest. The comparison is carried out by calculating the average pixel values and computing the mean squared difference. If this value is below a certain threshold, indicating minimal color dissimilarity, the mask is discarded.

This color comparison is performed twice, once in the HSV color space and again in the RGB color space. Including the HSV color space is crucial because even though colors may appear very similar in the RGB color space, they can exhibit significant differences in terms of hue, saturation, and value in the HSV color space. By considering both color spaces, a more accurate and comprehensive analysis of color dissimilarities can be achieved.

However, it's important to note that this color-based sorting may remove some useful information, such as people or objects with camouflage, as their color variations might be deemed similar to the surrounding context.



(10) Second sort based on the color.

It is important to note that despite our efforts, the results obtained are not perfect. For example, some cars may have poorly defined contours, or grass may still be present in certain regions. However, it quickly becomes complex to precisely sort the information we want to extract without risking the loss of important details. Our main objective is to preserve as much relevant information as possible from the image and let the person using this information make their own further sorting and decisions. Although some imperfections may remain, we aim to maximize the quality of the generated masks while maintaining a holistic approach that avoids losing valuable information.

Following this sorting process, we have a list of masks representing the different detected objects in the image. Our goal now is to determine whether these objects were already present in the reference image or if they are new elements that are particularly important for our analysis.

To do this, we apply SAM to the reference image, but only in the locations where the objects are expected to be found. The model provides us with a mask corresponding to these locations, which we need to compare with the reference mask. We perform this comparison based on the area of the masks. If the ratio or difference between these areas is significant, it indicates a substantial difference between the detected objects and those present in the reference image. Conversely, if the mask areas are similar, it means that the objects are the same.

It would be interesting to explore other methods of comparison; however, at present, we have not found any other suitable approaches. For example, we considered comparing differences in color between the masks, but this approach has limitations as elements such as snow, sand, or dust can affect these comparisons, rendering the method unreliable.

(11) This is the result obtained at the end: the elements highlighted in red are considered as differences.

We can see here certain limitations of the process. For example, the entire scaffolding is not 'considered' as a difference. This is because the model can sometimes exhibit 'white spots' that are likely caused by lighting variations during photogrammetry. These spots are identified as objects by the model, and since the object comparison is solely dependent on location and size, actual objects in the original image may not be recognized as differences. Therefore, it is crucial to take these anomalies into account when interpreting the results. It would be advantageous to enhance the object comparison method to fix this issue. Here is an example of these scaffolding masks found on the right side, and on the left side, the same location but captured from the perspective of the 3D model. At the end of the comparison process, no masks are kept.

(12) Example of limitation of the process

5/ The result

Now that the algorithm is working, it was necessary to find a solution to present the results. While displaying the results on the images is satisfactory, a problem remains: it is challenging to geographically locate the detected differences. Therefore, we decided to adopt an innovative approach available in the Carmenta Engine SDK, to display the result on an interactive 3D map.

To do so, I used a tool called ‘camera project operator’, that allows us to provide an image along with the necessary telemetry information (the camera position and orientation for example) and project these elements to the appropriate location on the interactive 3D map. Through this method, we can visualize the detected differences clearly and accurately locate them on the map:

(13) Results displayed on a 3D map.

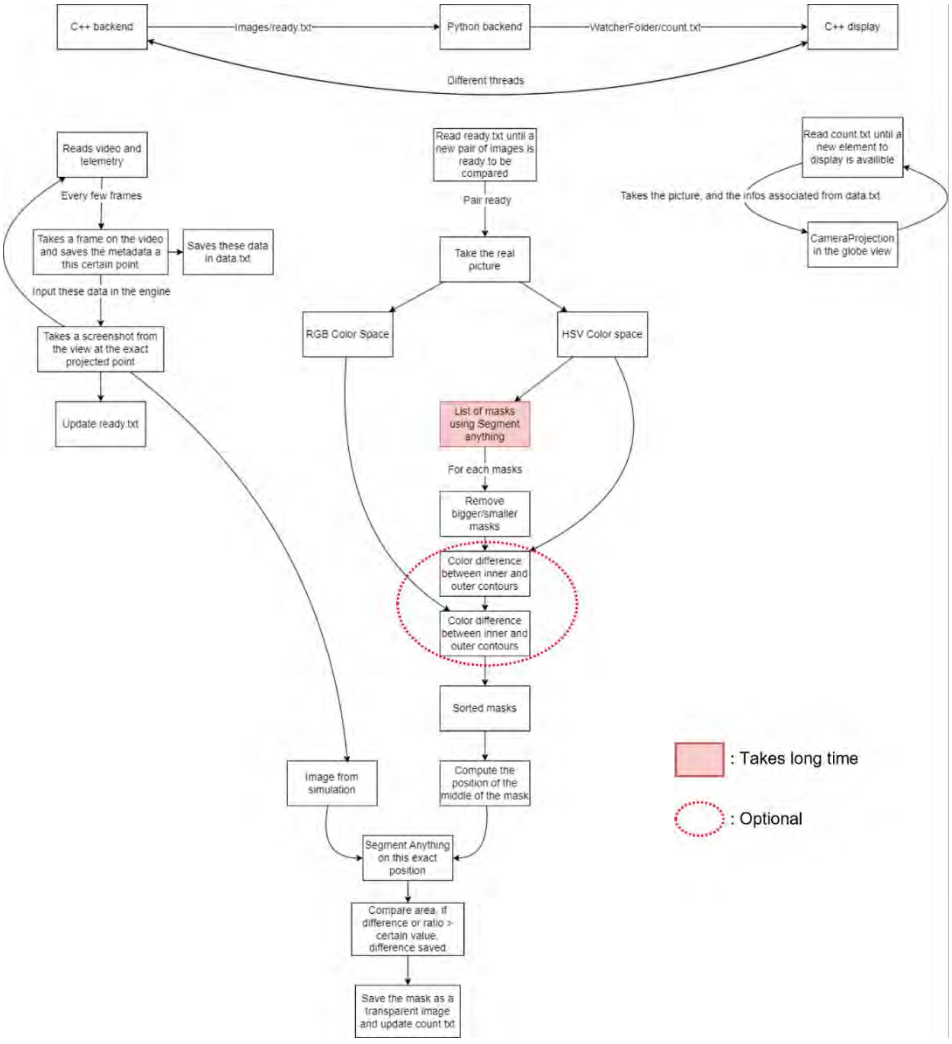
Thus, the entire project consists of three interconnected processes:

1. The backend process parses the metadata and video and saves the images for processing. Once the images are ready, a message is sent to the comparison process.

2. The comparison process waits for a pair of images to be ready for comparison. When this happens, it performs the masking, followed by mask sorting, and finally carries out the comparison. The result is saved as an image with a transparent background. It then sends a message to the frontend process indicating that a new image is ready to be displayed.

3. The frontend process displays the 3D map of the location. When new differences are ready to be shown, the CameraProjectOperator is called to place them at the correct location.

Here is a diagram that provides a more detailed explanation of the steps performed by the tool:



(14)Diagram explaining all the steps from the process

Issues Encountered During the Internship

1/ Image Differences: Attempting Image Processing for Improved Results

One of the significant challenges I faced during the internship was dealing with the differences between the images captured by the drone and the 3D model. To overcome this challenge, I had to develop a robust image processing pipeline capable of effectively handling various image conditions and enhancing the overall quality of the captured images.

To address this issue, extensive research was conducted to explore techniques such as shadow removal, photo correction, and color correction. I thoroughly studied relevant image processing methodologies, delved into existing literature, and experimented with different approaches to find the most effective solutions. I faced the challenge of dealing with the diversity of image processing techniques and the process of finding, coding, verifying their functionality, and assessing their usefulness. The difficulty lay in exploring various processes, implementing them, and ensuring that they produced better results.

2/ Unique Perspective and Scarcity of Datasets

Another major difficulty I encountered was related to the unique aerial/satellite top-down perspective of our project. Unlike readily available datasets and models that are primarily trained on front-facing perspectives, there was a scarcity, if not a complete absence, of datasets and pre-trained models specifically designed for overhead imagery analysis. This presented a significant hurdle as our objective was to accurately segment and analyze all objects within the images, rather than focusing solely on labeled objects or specific categories.

The lack of readily available models and datasets for our desired perspective required extensive exploration and experimentation to identify or develop suitable solutions. I conducted a thorough search for relevant models, considering factors such as adaptability to our specific use case and compatibility with our existing infrastructure. Additionally, installing, training, and evaluating these models posed challenges, often requiring substantial debugging and efforts to ensure smooth integration into our pipeline.

In my search for relevant models and datasets, I took a comprehensive approach. This involved reading research papers, searching extensively on the internet, and exploring GitHub repositories. I explored various sources to find datasets and pre-trained models that could be adapted to our unique top-down perspective.

One specific challenge I encountered was the difference in platforms. Since I was working on Windows, while most of the tools and resources were primarily designed for Linux, there were compatibility issues during the installation, training, and evaluation of the models.

3/ Challenges with the Carmenta Engine

Although the Carmenta Engine is a powerful and robust tool, I had limited familiarity with it, as my specific tasks did not extensively require its use. As a result, even simple tasks or manipulations posed challenges, and I struggled to navigate certain aspects of the software. I heavily relied on the documentation to understand and utilize its functionalities. I spent time studying the documentation to navigate through the software effectively.

Without the assistance and support of my colleagues, resolving some of the encountered issues would have been significantly difficult. Some features crucial to our project were not prominently highlighted in the documentation, making it challenging to find the appropriate configurations and techniques to achieve the desired performance.

One specific example of such a challenge was optimizing the performance of the engine. Typically, the model is loaded and processed gradually, incrementally increasing the resolution to ensure smooth and efficient operation. However, for my screen captures and analysis, immediate access to the highest resolution was necessary to accurately capture fine details. Consequently, I had to explore various approaches and configurations before receiving guidance from my colleagues on the appropriate settings and techniques to achieve the desired outcome.

The collaboration and knowledge-sharing within the team played a crucial role in overcoming these challenges. With their guidance and expertise, I was able to navigate the intricacies of the Carmenta Engine more effectively, overcoming obstacles and optimizing its performance to meet my specific requirements.

4/ GPU Computing: CUDA Challenges

The last aspect of my internship that posed a challenge was the utilization of big machine learning models, particularly the SAM model. One of the main goals of the project was to achieve real-time performance. However, SAM, being a resource-intensive model, presented significant computational demands. As a result, attaining real-time performance became a primary objective.

Unfortunately, I encountered limitations due to the unavailability of CUDA on my personal computer. To address this issue, I had to explore alternative solutions. Whenever necessary, I conducted tests on platforms like Colab, which provided access to GPUs for faster computation. However, even after setting up the entire pipeline, I faced difficulties in directly utilizing CUDA, resulting in considerable time losses.

One specific process that significantly impacted the overall processing time was the mask generation using SAM. Utilizing CUDA on Colab, the mask generation process for an image took an average of 7 seconds. In contrast, without CUDA, it could take between 3 and 5 minutes. It's worth noting that SAM requires a powerful graphics card to leverage CUDA effectively. Unfortunately, the graphics card in my laptop did not meet the necessary specifications, making it impractical to run SAM with CUDA.

Due to the high computational demands of SAM, it is not feasible to embed such a solution in a drone. A part of my internship project involved studying the feasibility of running the entire process onboard the drone. However, considering the power requirements of SAM, it is impossible to execute the complete solution on the drone itself. Instead, a viable approach is to divide the solution into two parts: generating real-time images onboard the drone and performing the comparison and presentation of results once the drone is connected to a more powerful system.

Personal review

During my six-month internship as a software engineer at Carmenta Geospatial Technologies in Stockholm, I had the opportunity to explore and appreciate the work culture in Sweden. It was a highly rewarding experience both professionally and personally.

I worked on an individual project but had weekly meetings with a German colleague and a Swedish colleague. These exchanges were beneficial as they allowed me to share ideas and receive constructive feedback.

What struck me about Swedish work culture was the openness and trust given to employees. I observed that unlike some other professional environments, there was a strong belief in individuals' ability to autonomously accomplish their tasks. Supervisors and the management team were not constantly monitoring employees. Instead, there was an implicit understanding that everyone was responsible for their own work and outcomes.

This trust in employees fostered a sense of autonomy and independence, allowing for personal growth in the workplace. One felt free to organize their time and tasks in the way that suited them best, while still respecting goals and deadlines. This freedom and trust enabled me to develop a sense of responsibility and take initiative to achieve set objectives.

Another significant aspect of my internship was English communication. I almost exclusively spoke English throughout my internship, which significantly improved my language skills. Working in an international environment also allowed me to develop intercultural communication skills and adapt to different ways of working.

On a technical level, I naturally acquired new skills and enhanced my knowledge. I was exposed to innovative software development technologies and methodologies. However, I realized that developing my soft skills was equally important. I learned to work independently, be proactive, and take initiative. I also developed my problem-solving ability in a creative manner and effectively collaborated with my colleagues.

Overall, my internship as a software engineer in Stockholm was an extremely beneficial experience. Not only did I enhance my technical skills, but I also developed my communication, teamwork, and adaptability in a different work culture. I am grateful to the company that provided me with this opportunity and to my colleagues for their support throughout the internship. I am confident that this experience will serve me in my future career as a software engineer.

To go further

During the internship, several potential avenues for improving the algorithm and enhancing the capabilities of the system were explored. Although no specific methods have been identified at this stage, continuous exploration of potential approaches is crucial to address existing challenges and optimize the project’s performance.

One aspect that requires attention is the reduction of missed details caused by lighting variations and the enhancement of accuracy by mitigating false positives and false negatives. A promising solution to consider is the training of a specialized machine learning model dedicated to this purpose. This model could be designed to improve the algorithm's ability to compare elements, thereby minimizing errors and ensuring more reliable results.

Another area of interest is the recognition of overlapping elements. By developing a method to identify when two or more elements overlap, we can enhance the system's ability to display a single, clear representation of these elements. This improvement is particularly valuable in scenarios where multiple objects overlap, as it enhances recognition and understanding. Additionally, the exploration of photogrammetry techniques can be considered to generate detailed 3D representations of the overlapping elements, providing a more immersive visualization.

Optimizing the model to run onboard the drone presents a significant challenge due to current hardware limitations. Real-time comparison, a crucial requirement, necessitates the use of CUDA, which relies on a high-performance graphics card. Unfortunately, this is not compatible with our embedded drone solution. To overcome this obstacle, a hybrid solution can be adopted. By performing a portion of the process onboard the drone itself, we can achieve real-time image generation during the drone's flight over the area of interest. The remaining comparison process would then be conducted on a computer after the drone is connected to it. This hybrid approach allows for efficient utilization of resources, with the drone handling image generation and the computer performing the comparison.

Interactions with elements can be further improved by implementing an 'OnClick' functionality. This feature would enable users to interact with the system by removing elements or retrieving additional information simply by clicking on them.

These potential developments and enhancements offer exciting prospects for the future of the project. By continuing to explore and implement these ideas, we can strive for increased accuracy, real-time performance, and a more interactive user experience.

Utility for the company

This project initially started as a research project and was not commissioned by a specific client. While the exact utilization of this project is still under consideration, during my internship, three potential use cases were suggested:

1. Airport Applications: One potential application is in the field of airport operations. The system can be utilized to automate the verification process, ensuring that everything is in its designated place and detecting any signs of damage or irregularities within the airport premises. This automation can significantly streamline the monitoring and maintenance procedures, enhancing overall efficiency and security.

2. Battlefield Reconnaissance: In hazardous or volatile environments like battlefields, it is often risky to deploy personnel. In such scenarios, drones are frequently employed for reconnaissance purposes. In this context, the developed system can be utilized to perform automated preliminary reconnaissance tasks. By deploying the drone equipped with the system, a quick and automated assessment of the area can be conducted, providing valuable information without risking human lives.

3. Tool for Demonstrations: The developed system can serve as a valuable demonstration tool to showcase the capabilities of Carmenta Engine. By providing examples of what can be achieved using Carmenta technology, the company can gauge potential client interest and explore opportunities for collaboration.

These potential applications demonstrate the versatility and practicality of the developed system. As the project progresses, further exploration of these use cases and the identification of additional potential applications will be crucial to maximize the project's impact and value for the company.

Ethical and Societal impact

Carmenta Geospatial Technologies operates in the field of cartography, a discipline widely utilized across various domains. However, the primary area of focus for the company lies in defense, serving international clients. This specific industry positioning entails significant responsibility for Carmenta in terms of the clients it chooses to work with, resulting in significant ethical and societal implications.

By collaborating with clients in the defense sector, Carmenta must consider the ethical ramifications of its partnerships. It is crucial for the company to ensure that its products and services are not employed in ways contrary to human rights, global peace, or international ethical standards. This necessitates thorough evaluation of potential clients, their activities, and their commitments to upholding human rights.

Furthermore, the defense domain can raise complex societal issues. The technologies and tools developed by Carmenta can be employed in sensitive contexts such as surveillance or military intervention. Consequently, the company must reflect upon the societal impact of its work and take measures to minimize the risks of misuse or misappropriation of its technological solutions.

Carmenta must not only comply with trade and defense regulations and international standards, but also adopt a proactive approach to integrate ethical and societal considerations into its decision-making processes. This may include establishing clear criteria for client selection, implementing monitoring mechanisms to ensure responsible use of its products, and raising awareness among its employees about these ethical and societal issues.

In summary, by operating in the field of cartography with a strong focus on defense, Carmenta faces pivotal ethical and societal choices. By making responsible decisions and integrating these considerations into its business practices, the company can contribute to promoting ethical uses of its technologies and minimizing negative impacts on society and human rights.