Vehicle Geolocalization from Drone Imagery

David Novikov[1], Paul Sotirelis[2], Alper Yilmaz[3]

[1] Weizmann Institute of Science, Faculty of Mathematics and Computer Science, Rehovot, Israel - david.novikov@weizmann.ac.il

[2] AFRL, RYAP, Dayton, United States - paul.sotirelis@us.af.mil

[3] The Ohio State University, Department of Civil Environmental and Geodetic Engineering, Department of Computer Science and Engineering, Columbus, United States - yilmaz.15@osu.edu

KEY WORDS: Object Geolocalization, Drones, Drone Cameras, Image Registration, Aerial Maps, Deep Learning, Object Tracking.

ABSTRACT:

We have developed a robust, novel, and cost-effective method for determining the geolocation of vehicles observed in drone camera footage. Previous studies in this area have relied on platform GPS and camera geometry to estimate the position of objects in drone footage, which we will refer to as object-to-drone location (ODL). The performance of these techniques is degraded with decreasing GPS measurement accuracy and camera orientation problems. Our method overcomes these shortcomings and reliably geolocates objects on the ground. We refer to our approach as object-to-map localization (OML). The proposed technique determines a transformation between drone camera footage and georectified aerial images, for example, from Google Maps. This transformation is then used to calculate the positions of objects captured in the drone camera footage. We provide an ablation study of our method’s configuration parameter, which are: feature extraction methods, key point filtering schemes, and types of transformations. We also conduct experiments with a simulated faulty GPS to demonstrate our method’s robustness to poor estimation of the drone’s position. Our approach requires only a drone with a camera and a low-accuracy estimate of its geoposition, we do not rely on markers or ground control points. As a result, our method can determine the geolocation of vehicles on the ground in an easy-to-set up and cost-effective manner, making object geolocalization more accessible to users by decreasing the hardware and software requirements. Our GitHub with code can be found at `https://github.com/OSUPCVLab/VehicleGeopositioning`

1. INTRODUCTION

Drones have a variety of use cases, typically centered on capturing data about an environment with a camera and/or other complementary sensors. These include agriculture (Daponte et al., 2019), delivery of medicines in remote regions (Nyaaba and Ayamga, 2021), delivery of products in urban settings (Rodrigues et al., 2022), and land surveying (EL Meouche et al., 2016). Two essential tasks for drones are determining where they are and where other objects are in the scene. These tasks enable useful data for the end user and prevent the drone from colliding with its environment or other drones.

Today’s approaches for drone camera-based object geolocalization rely on precise camera geometry or elevation maps to determine a relationship between a drone camera and its environment. We will refer to these schemes as object-to-drone localization (ODL) methods. The position of an object relative to the camera is determined, and then the object is localized using the drone position. This requires precise and accurate knowledge of the drone and camera parameters to georeference the object of interest from the image (parameters such as field of view, intrinsic and extrinsic camera calibration, camera tilt/viewing angle, drone altitude, and GNSS sensor readings). This presents a challenge for the end user, as these conversions lead to opportunities for error and require precise and accurate knowledge of the drone and its relationship to the environment. The user must either guarantee the accuracy and precision of these parameters for each image the drone captures or accept a significant margin of error.

We present a novel method that requires only the low-accuracy geoposition of the drone. We will refer to this new scheme as object-to-map localization (OML). Our approach uses the reported geoposition of the drone with apriori reference mapping information provided through Google Maps and other platforms such as Bing Maps or Apple Maps. Our reduced parameter count minimizes hardware and software requirements to perform geolocalization in unseen environments. We note that our method is robust to spatiotemporal and seasonal differences between drone camera-acquired images and reference data in the form of georectified reference images. This scene-centric approach is similar to the parallax approach introduced originally by (Irani et al., 1998), which shifts the focus of 3D vision tasks away from the camera’s explicit relationship to its environment.

The rest of the paper is organized into related work discussed in the next section. The proposed approach is discussed in Section 3. This is followed by experiments and their results, and finally conclusions in Sections 4 and 5.

2. RELATED WORK

There are two main methods for ODL using drone cameras. The first is to use the drone’s altitude and camera parameters to determine object positions and to localize them using the drone’s

position. The second is to determine the position of objects relative to the drone via depth maps or LiDAR data.

In (Zhang et al., 2019), the authors propose using drone altitude and camera angle information to find a ground plane and locate objects with the intrinsic camera calibration matrix. This is done by backpropagating object positions from 2D to 3D and placing them on the ground plane. (Liu and Li, 2021) provides a similar approach; however, they also consider the orientation of the camera mount and the pitch, roll, and yaw of the drone. They report their localization accuracy via AprilTags (visual fiducials) which they put on the ground with an established geolocation. The average error between the AprilTags’ locations and the true locations is between 4.38 and 6.88 meters when flying at 6.5 and 10 meters, respectively. In comparison, our flights occur at 20 to 45 meters.

Another approach uses the field of view and altitude of a camera to determine the position of an object (Dwyer, 2022). They provide a clear image of the general framework of the geometric method, which is shown in Fig. 1. (Lygouras, 2020) uses a similar method for ODL; however, their approach uses a real-time kinematic positioning (RTK) correction system. This provides a more precise geolocation for the drone and improves the quality of ODL. However, the authors note that if the drone is outside of the range of the RTK correction, it uses a GNSS system to localize the drone and its error significantly propagates to the predicted positions of the geolocated objects. The practical challenge this method presents is that RTK systems must be set up and configured before flight in the desired area and are available for use at all times, which limits the effectiveness of this method. We should note that the cost of a single RTK system is often high and it must be calibrated before use. If many are needed to maintain a reliable GPS position, the cost could be prohibitive. Finally, the RTK correction addresses only one parameter out of many required for ODL.

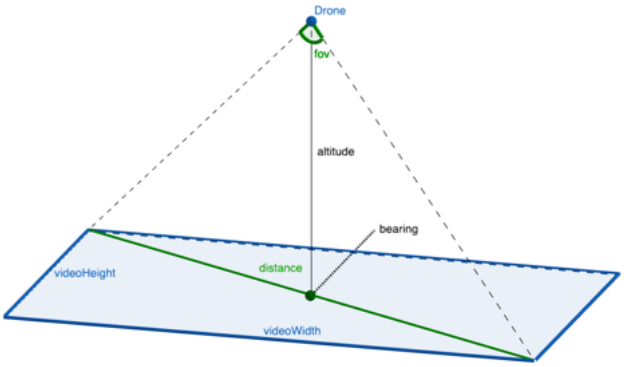

Figure 1. Geometry Based Localization Approach (Dwyer, 2022)

The open source software OpenAthena (mkrupczak3, 2023) expands on previous ODL methods, taking into account the topology of the ground it observes. This allows for more better localization and fewer errors, given that the drone’s geolocation is reported accurately as well. OpenAthena’s method is shown in Fig. 2. Depth maps are generated by (Carrio et al., 2020) during flight to position objects relative to the drone using stereo vision. Their use case is to avoid collisions with other drones (or other objects) during flight.

ODL methods use the drone as an anchor in the scene that the drone observes. This is undesirable in object geolocalization

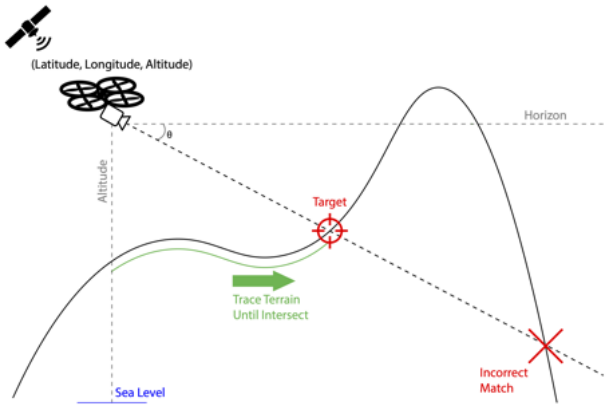

Figure 2. Geometry Based Localization with Topological Constraints (mkrupczak3, 2023).

and is tiresome for drone platform operators. The user must determine the drone camera and sensor parameters with sufficient sensitivity to minimize error and then ensure that these parameters are accurate and precise throughout the flight. To satisfy geometric assumptions, more accurate and precise sensors are used to meet the requirements. These sensors are larger, heavier, and more expensive. In practical scenarios, the requirement of knowing a camera’s exact orientation and relationship to the environment is easily violated due to the dynamic nature of drone flight, such as wind, turbulence, platform vibration, and rapid changes in drone motion.

Multiview geometric analysis was done in (Irani et al., 1998) by using homographic transformations to describe the scene. This allows for 3D scene reconstruction and novel view rendering within the scene. The primary assumption is that the scene has discernible planar surfaces, which can be mapped to each other across images via the homography transform. The advantage of this method is that is describes the 3D nature of the scene without overly relying on the camera parameters and from where the camera took pictures.

During a survey of object geolocalization, (Wilson et al., 2023) found no datasets with objects observed from the air with reported object geolocations. We were unable to find such a dataset as well. This gap in the literature will not be addressed in this paper, but the result of this gap is that there are no metrics to evaluate the accuracy of methods to geolocate objects from drone footage.

Our method’s contributions are as follows:

1. Cost-effective method for localization, with no hardware requirements beyond any drone with a GPS receiver and camera,
2. Robustness to the dynamic nature of drone flights,
3. Minimum requirements to start using for the first time and in a new environment

3. METHODOLOGY

We will briefly describe the pipeline to give the reader a complete picture of the process. This process is summarized in

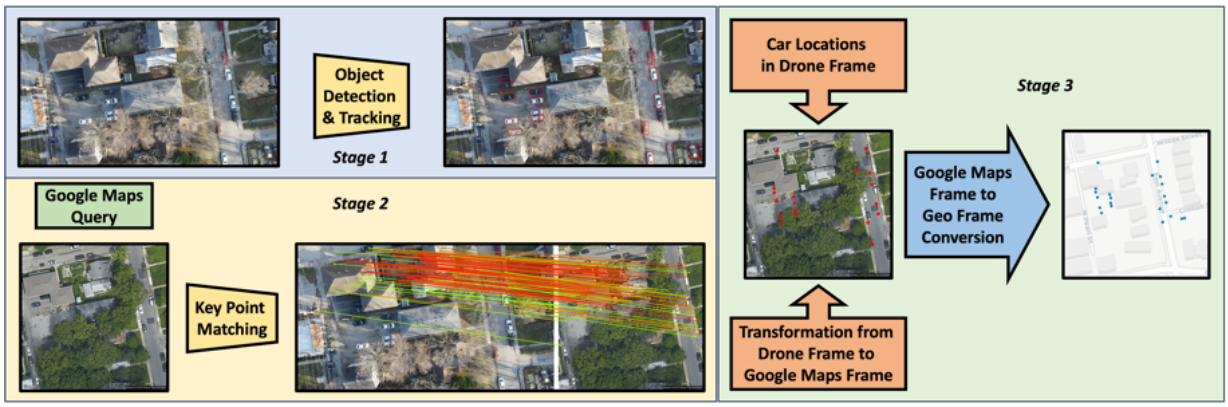

Figure 3. Pipeline Summary: In *Stage 1*, all objects of interest are detected for each drone image, and object tracking is applied in the sequence. During *Stage 2*, georectified reference image is queried from a provider (in this paper, we used Google Maps) that is closest to the drone's reported geoposition to estimate the projective transformation. In *Stage 3*, the objects from *Stage 1* are transferred to the reference frame using the transformation estimated in *Stage 2*. Finally, the geopositions of the objects are estimated and placed on the map.

Fig. 3. The approach starts with detecting the vehicles in the drone footage. This process is followed by matching key points between the drone image and a queried georectified reference image and estimating the transformation between them. Finally, the detected vehicle image positions per frame are transformed to the reference frame. We can then geolocate the objects once they're in the reference frame because reference frame is already georectified. In the context of an image sequence, visual object tracking is applied to track and determine many object geopositions per object (which can then be post processed into one position for each object).

3.1 Detecting and Tracking vehicles

To detect vehicles in the drone footage, we finetuned a YOLOv8 model (Jocher et al., 2022) for 25 epochs. Since the vehicles are small objects at higher altitudes, during training, we chose to tile the images in the dataset. The augmentation to the dataset was performed using Roboflow (Dwyer et al., 2022). We adopted the SAHI approach reported in (Akyon et al., 2021) to perform sliced window inference to improve detection performance, especially when a large number of vehicles co-occur.

To track vehicles in consecutive frames, we adopted the DeepSORT approach (Wojke and Bewley, 2018). The tracking history is used to associate the vehicle geolocations estimated in consecutive images. This allows us to develop a distribution of the predicted locations for the vehicles.

3.2 Matching and Filtering Features between Images

As shown in Fig. 4, there is a spatiotemporal domain gap between the orthorectified reference image and the drone images. In the drone imagery, the trees have no leaves and cast shadows occur due to early morning capture. Furthermore, the georectified Google Maps imagery was captured in 2017, while drone imagery was captured in 2022, so some of the buildings and roads have changed across the images. In our experiments, we observed that the best method for keypoint extraction and matching that mitigates the spatio-temporal gap is SuperGlue (Sarlin et al., 2020) and LoFTR (Sun et al., 2021). We apply no fine-tuning and use the pretrained vanilla models for both. During the initial testing of our method, we found that classical methods for keypoint extraction and matching, such as SIFT (Lowe, 1999) and ORB (Rublee et al., 2011), cannot handle the spatiotemporal gaps between the images.

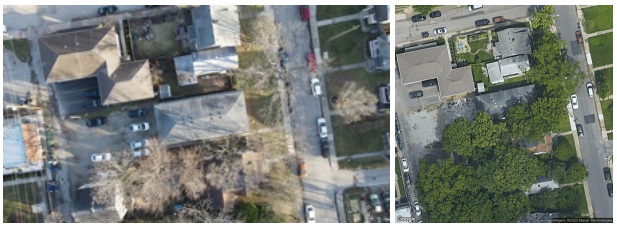

Figure 4. Drone Image (left), Corresponding Queried Google Maps Image (right)

We use the matching keypoints to estimate a projective transform (homography), H, or a 2D affine transformation, A, between them using RANSAC (Fischler and Bolles, 1981), implemented in (Bradski, 2000). This allows us to georeference any keypoint in the drone image on the map. We show the effect of homography by overlaying the drone image on the georectified reference image in Fig. 5. Note that we assume that the camera points generally down.

Figure 5. Drone Image Fused onto Google Maps Image (vehicle positions shown in red)

We implemented three filtering operations to eliminate what we consider to be unreliable keypoints. We refer to the filters as the Road, Building, and Vehicle filters. These filters are discussed in the following text, and sample filtering results are shown in Fig. 6. The way these filters work is by applying a mask to the detected keypoints.

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-X-2-2024-171-2024 | © Author(s) 2024. CC BY 4.0 License.

- The road filter removes keypoints that are not located on or near roads. This was implemented because vehicles exist on or near the roads, and we expect the best matching features to be near the roads in the Google Maps image. The mask for this filter is found by upper and lower thresholding a semantic Google Maps image from the same position as the Aerial Google Maps image. The thresholding is set up to only keep the points which are on the road. Image dilation is applied to the mask to extend the valid mask regions slightly beyond the road.
- For the building filter, we remove key points detected on buildings, as the tops of buildings do not lie on the ground plane where we observe vehicles. Using keypoints in buildings could lead to a poor transformation estimate, as they are not in the same plane as the ground. Like the road filter, the building filter thresholds a semantic Google Maps image. Next, image erosion and dilation are applied to clean up artifacts left by the semantic thresholding.
- The vehicle filter is used to prevent similar vehicles across the images from being used as matching features (since these vehicles are extremely unlikely to be the same vehicles with the 5-year temporal gap). For the vehicle filter, we remove matching key points that are within the detected vehicles’ bounding boxes in the drone footage.

Figure 6. Sample Camera Image Key Points with Filters Applied: Original Features (top left), Road Filter (top right), Vehicle Filter (bottom left), Building Filter (bottom right).

3.3 Summary of Configuration Parameters

This subsection will briefly review which parameters we can adjust in our method. For the keypoint extraction portion, we can either use SuperGlue, LoFTR, or both (3 options). For the filtering portion, we can apply any combination of the Road, Vehicle, or Building filters, or we can apply no filtering (8 options). Finally we can either use a homography or a 2D affine transformation to describe the relationship between the drone image and the Google Maps image (2 options). There are 48 different parameter configurations we can have, we test all of them in Section 4.

3.4 Calculating the geolocation of vehicles

Once a transformation is determined between the georectified reference image and the drone image, it is applied to the vehicle geolocations to place them on the reference image. Since Google Maps is adopted for georectified image retrieval, in the following discussion, we will provide the conversion from the drone image to georectified coordinates in terms of Google Maps parameters.

In the following formulas; x_{gm} and y_{gm} are the x, y pixel positions after they’ve been transformed from the drone image to the Google Maps image, gmi_w and gmi_h are the width and height of the Google Maps image in pixels, deg_per_m is the number of degrees latitude per meter on earth, m_per_pix is how many meters each pixel in a Google Maps image represents, $zoom$ is the Google Maps zoom level (in our case 20), lat_factor is the adjustment when computing the latitude change at varying latitudes, and lat, $long$ are the latitude and longitude of the Google Maps image (located at its center in the image).

$$\delta x = x_{gm} - \frac{gmi_w}{2} \tag{1}$$

$$\delta y = \frac{gmi_h}{2} - y_{gm} \tag{2}$$

$$deg\_per\_m = \frac{360}{2 \times \pi \times 6378137} \tag{3}$$

$$m\_per\_pix = \frac{156543.03392}{2^{zoom}} \tag{4}$$

$$lat\_factor = cos(\frac{lat \times \pi}{180}) \tag{5}$$

Then we can see that the change in latitude and longitude come from equations (1) through (5) as follows:

$$\delta_{lat} = \delta x \times deg\_per\_m \times m\_per\_pix \times lat\_factor \tag{6}$$

$$\delta_{long} = \delta y \times deg\_per\_m \times m\_per\_pix \tag{7}$$

Finally we can get the latitude and longitude of the object of interest:

$$obj_{lat} = \delta_{lat} + lat \tag{8}$$

$$obj_{long} = \delta_{long} + long \tag{9}$$

4. EXPERIMENTS

In this section, we first introduce the in-house generated dataset. We will then discuss the evaluation metrics and finally provide a discussion of our results. We will also discuss some additional experiments we performed to evaluate how resilient our algorithm is to faulty GPS devices and position estimation with respect to the the drone’s true geoposition.

4.1 Data

The drone images used in this project are from videos released in (Wei et al., 2022). The data was captured using a DJI Mavic Air2. We use the method proposed in (Wei et al., 2022) to determine approximate drone geopositions. Eighty-eight images from three videos were labeled using Roboflow (Dwyer et al., 2022) to generate a small vehicle training data set. Some sample annotations are shown in Fig. 7. The footage generally was captured during the early morning, as a result we expect the predicted vehicle positions to be tightly clustered and consistent across frames.

Figure 7. Sample Annotations

Drone geopositions are used to query a reference Google Maps georectified reference image during the analysis of the captured footage. Fig. 4 shows a drone image and the Google Maps reference image at the reported geoposition.

4.2 Results and Evaluation Metric

To evaluate our method and various configurations of parameters that can be used with the proposed pipeline, we sampled 100 frames of drone footage in 5 different regions. These regions have varying heights (20-45 meters) and constantly changing camera orientations (due to the drone motion and interaction with the environment). We only provide metrics for precision as we do not have the ground-truth geolocations for the vehicles captured in this dataset. We convert the car predicted positions to meters and operate in this space for easier interpretability.

For all tracked vehicles, we remove spurious detections and vehicles that appeared once during the video. For each vehicle, we run (Pedregosa et al., 2011)’s implementation of DBSCAN (Ester et al., 1996), with the eps parameter set to 4.48 meters, the average length of a car’s major axis. We then select only the points inside the largest cluster found. We observed that we remove approximately 20% of points when using any parameter configuration with SuperGlue and 40 to 50 % of points when using only LoFTR. There is some variation in the number of points due to other parameters, but the keypoint extraction and matching makes the largest difference. Next we perform PCA to identify the major and minor axes the car’s predicted positions lie on. We scale the values so that the major axis components have the same range as the minor access components. A sample cleaning of points can be see in Fig. 9. We added the scaling via PCA to minimize the error due to projection.

If our method predicts car positions consistently, we expect the distance of all the predicted car positions to the car’s predicted position center to be low. In Fig. 8 we show qualitative results of plotting the geolocations of the cars observed in the drone footage after post processing.

In Table 1 we provide the mean and standard deviation of the distances of all car’s predicted positions to their cluster center. The best geoposition estimates are generated using keypoints generated using SuperGlue, no filters, and homography as the transformation. This configuration typically places vehicles

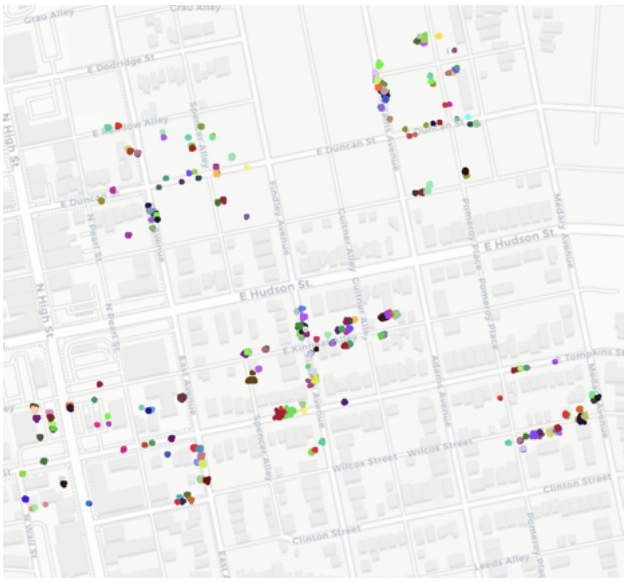

Figure 8. Sample Scattering of Vehicle geolocations for 500 Frames (one color per vehicle)

within 0.7 meters of the vehicle’s geolocation cluster center. We consider this to be a strong result since the dimensions of a typical vehicle are 4.48 x 1.77 meters (Meyer, 2023).

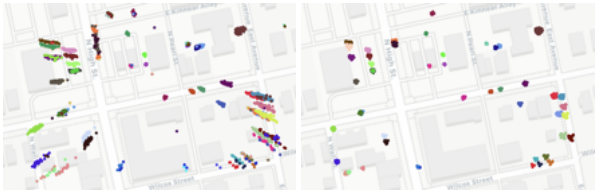

Figure 9. Sample Outlier Removal and Post Processing done with DBSCAN and PCA. Original Car Predicted Positions (left), Cleaned Car Positions (right).

We also preformed t-tests using (Virtanen et al., 2020) on the distributions seen in Table 1, if two distributions only had one configuration parameter changed. Our p-value threshold for this was 0.01. We summarize the results of these t-tests in Table 2.

4.3 Faulty GPS

We simulate flying a drone with a faulty GPS to show how robust our method is to the estimate of the geoposition of the drone. To do this we take the drone’s geolocation and randomly add up to ± 1, 5, 10, 20, 30, 40, 50, or 60 meters to the drones x and y positions. We then input these noisy geolocations to our method and compare how much each predicted car position changes. Sample geolocations which we input into our method can be seen in Fig. 10, and the change in each predicted car position can be seen in Fig. 11.

4.4 Discussion

Across all of our experiments that use LoFTR for feature extraction, the vehicle geolocalization performance was determined to be the worst. Generally speaking, LoFTR + SuperGlue typically under-performs just using SuperGlue. We hypothesis that this gap in performance comes from the variations in performance we expect to see from difference models on different

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-X-2-2024-171-2024 | © Author(s) 2024. CC BY 4.0 License.

Filtering Performed			Feature Extraction Method					
Road	Building	Vehicle	LoFTR		SuperGlue		LoFTR + SuperGlue	
			4.101 ± 4.424	5.927 ± 5.027	**0.665 ± 0.640**	1.740 ± 1.604	0.668 ± 0.610	1.853 ± 1.729
		✓	4.182 ± 4.502	6.148 ± 5.238	0.718 ± 0.743	1.731 ± 1.556	0.748 ± 0.817	1.895 ± 1.746
	✓		4.759 ± 4.791	6.146 ± 4.738	0.791 ± 0.782	1.674 ± 1.600	0.817 ± 0.930	1.855 ± 1.662
	✓	✓	4.821 ± 5.000	6.203 ± 4.912	0.793 ± 0.755	1.817 ± 1.674	0.779 ± 0.823	1.888 ± 1.788
✓			5.675 ± 4.776	6.525 ± 4.800	1.281 ± 1.395	1.906 ± 1.720	1.272 ± 1.391	2.153 ± 2.111
✓		✓	5.701 ± 4.712	6.434 ± 4.728	1.131 ± 1.101	1.937 ± 1.745	1.250 ± 1.300	2.331 ± 2.175
✓	✓		5.849 ± 4.869	6.503 ± 4.623	1.425 ± 1.618	2.406 ± 2.319	1.328 ± 1.557	2.459 ± 2.449
✓	✓	✓	5.929 ± 4.790	6.255 ± 4.616	1.405 ± 1.570	2.394 ± 2.235	1.479 ± 1.619	2.725 ± 2.607
			Homography	**2D Affine**	**Homography**	**2D Affine**	**Homography**	**2D Affine**
			Transformation					

Table 1. Mean and Standard Deviation of Distance to Vehicle Cluster Center for each Configuration (measured in meters) (bold indicates best).

Configuration Feature	Count
Vehicle Filter	10/24
Road Filter	23/24
Building Filter	18/24
Homography vs Affine 2D	24/24
SuperGlue	16/16
LoFTR	12/16

Table 2. Count of Times when Adding or Removing a Configuration Parameter causes a Statistically Significant Change in the Evaluation Metric.

Figure 10. Original (top left) and Noisy Geolocations (top right - 5 m, bottom left - 20 m, bottom right- 60 m) for Drone Flight.

datasets. We also see from Table 2 that changing the SuperGlue or LoFTR configuration parameter almost always causes a statistically significant change in the output.

Surprisingly to us, the use of filters tends to decrease performance and the difference is often statistically significant as seen in Table 1 and Table 2. This means that using these filters typically does not improve performance in a statistically significant way. Though there are a few exceptions. We hypothesize that there are two causes for this; the first being that RANSAC is already designed to remove outliers and the second being that the filters may remove points which are geometrically valid for the transformation. As a result, we are decreasing how many valid points we can use to find the transformation, resulting in a poorer transformation.

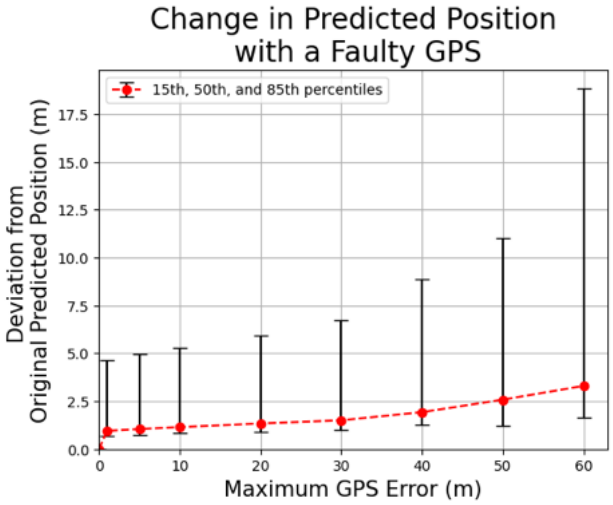

Figure 11. Change in each Predicted Position (measured at 15th, 50th, and 85th percentiles).

We observed that the homography transform tends to provide more consistent predictions compared to 2D affine transformations. From Table 2 we can see that this difference is always statistically significant. This is likely because homography transforms can account for changes in different perspectives on the same plane as the drone moves. Georectified reference images and drone images have different perspectives for the same scene, and homography can better capture the appropriate geometric transformation. Likely the drone camera doesn't point exactly down, and the ground it observes is also on a slight slope, so the homography transform captures the transformation better.

Our faulty GPS experiments show that our method can localize objects consistently even with a low accuracy estimate for the drone's GPS position. Note that in traditional ODL methods, any shift in the GPS position would result in the same shift to the predicted object position, however in our case we are typically significantly lower than this shift as seen in Fig. 11. For reference, consumer-grade smartphones as of 2015 have a mean accuracy of 4.9 meters (van Diggelen and Enge, 2015).

One limitation to consider is that this method can only be applied in places which have been apriori mapped. High resolution orthorectified maps are no available for all locations. However, many cities have these orthorectified maps, and platforms such as Google Maps have made many of them publicly avail-

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-X-2-2024-171-2024 | © Author(s) 2024. CC BY 4.0 License.

able. In the context of indoor or custom environments however, these maps would need to be generated on an individual basis.

5. CONCLUSIONS AND FUTURE WORK

We have developed a method for vehicle geolocalization that removes requirements for known platform parameters such as altitude, roll, pitch, and yaw, and camera parameters such as focal length and orientation. Our method standardizes the reference to a single source; in our experiments, we chose Google Maps as the reference. We note that in our approach any mapped system can be used as a reference. We also note that, by eliminating the large number of parameters used in alternative approaches, it is possible to deploy our solution with any camera-drone combination. The immediate effect of this is the reduced cost of deployment to unseen environments and to new users. In the future, we would like to apply our method to a dataset with ground truth geolocations for the vehicles observed in the drone footage to measure our method’s accuracy.

Acknowledgements

This work was supported in part by the U.S. Air Force Research Lab. under Grant AWD-111867. We would like to thank them for their support throughout the course of this research.

REFERENCES

Akyon, F. C., Cengiz, C., Altinuc, S. O., Cavusoglu, D., Sahin, K., Eryuksel, O., 2021. SAHI: A lightweight vision library for performing large scale object detection and instance segmentation.

Bradski, G., 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

Carrio, A., Tordesillas, J., Vemprala, S., Saripalli, S., Campoy, P., How, J. P., 2020. Onboard Detection and Localization of Drones Using Depth Maps. *Photogrammetria*, 8, 30480-30490.

Daponte, P., Vito, L. D., Glielmo, L., Iannelli, L., Liuzza, D., Picariello, F., Silano, G., 2019. A review on the use of drones for precision agriculture. *IOP Conference Series: Earth and Environmental Science*, 275(1), 012022.

Dwyer, B., 2022. Using computer vision with drones for georeferencing. Roboflow Blog,https://blog.roboflow.com/georeferencing-drone-videos/.

Dwyer, B., Nelson, J., Solawetz, J., 2022. Roboflow.

EL Meouche, R., Hijazi, I., Poncet, P.-A., Abu Nemeh, M., Rezoug, M., 2016. Uav photogrammetry implementation to enhance land surveying, comparisons and possibilities. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, AAAI Press, 226–231.

Fischler, M. A., Bolles, R. C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6), 381–395. https://doi.org/10.1145/358669.358692.

Irani, M., Anandan, P., Weinshall, D., 1998. From reference frames to reference planes: Multi-view parallax geometry and applications. H. Burkhardt, B. Neumann (eds), *Computer Vision — ECCV'98*, Springer Berlin Heidelberg, Berlin, Heidelberg, 829–845.

Jocher, G., Chaurasia, A., Stoken, A., Borovec, J., NanoCode012, Kwon, Y., Michael, K., TaoXie, Fang, J., imyhxy, Lorna, Yifu), Wong, C., V, A., Montes, D., Wang, Z., Fati, C., Nadar, J., Laughing, UnglvKitDe, Sonck, V., tkianai, yxNONG, Skalski, P., Hogan, A., Nair, D., Strobel, M., Jain, M., 2022. ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation.

Liu, J.-S., Li, D.-W., 2021. A drone patrol system for target object counting and geolocalization. *2021 18th International Conference on Ubiquitous Robots (UR)*.

Lowe, D., 1999. Object recognition from local scale-invariant features. *Proceedings of the Seventh IEEE International Conference on Computer Vision*.

Lygouras, E., 2020. Vision and Geolocation Data Combination for Precise Human Detection and Tracking in Search and Rescue Operations. *International Journal of Intelligence Science*, 10.

Meyer, S., 2023. Study: Average car size is increasing - the zebra. The Zebra. https://www.thezebra.com/resources/driving/average-car-size/.

mkrupczak3, 2023. Open athena.

Nyaaba, A. A., Ayamga, M., 2021. Intricacies of medical drones in healthcare delivery: Implications for Africa. *Technology in Society*, 66, 101624.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

Rodrigues, T. A., Patrikar, J., Oliveira, N. L., Matthews, H. S., Scherer, S., Samaras, C., 2022. Drone flight data reveal energy and greenhouse gas emissions savings for very small package delivery. *Patterns*, 3(8), 100569.

Rublee, E., Rabaud, V., Konolige, K., Bradski, G., 2011. Orb: an efficient alternative to sift or surf. 2564–2571.

Sarlin, P.-E., DeTone, D., Malisiewicz, T., Rabinovich, A., 2020. SuperGlue: Learning Feature Matching With Graph Neural Networks. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sun, J., Shen, Z., Wang, Y., Bao, H., Zhou, X., 2021. LoFTR: Detector-Free Local Feature Matching with Transformers. *CVPR*.

van Diggelen, F., Enge, P. K., 2015. The world's first gps mooc and worldwide laboratory using smartphones.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., SciPy 1.0 Contributors, 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17, 261–272.

Wei, J., Karakay, D., Yilmaz, A., 2022. A gis aided approach for geolocalizing an unmanned aerial system using deep learning. *2022 IEEE Sensors*.

Wilson, D., Zhang, X., Sultani, W., Wshah, S., 2023. Image and Object Geo-Localization. *International Journal of Computer Vision*, 1-43.

Wojke, N., Bewley, A., 2018. Deep cosine metric learning for person re-identification. *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 748–756.

Zhang, H., Wang, G., Lei, Z., Hwang, J.-N., 2019. Eye in the sky. *Proceedings of the 27th ACM International Conference on Multimedia*.