drones

Article

A Visual Odometry Pipeline for Real-Time UAS Geopositioning

Jianli Wei * and Alper Yilmaz

Photogrammetric Computer Vision Lab., The Ohio State University, Columbus, OH 43210, USA; yilmaz.15@osu.edu
* Correspondence: wei.909@osu.edu

Abstract: The state-of-the-art geopositioning is the Global Navigation Satellite System (GNSS), which operates based on the satellite constellation providing positioning, navigation, and timing services. While the Global Positioning System (GPS) is widely used to position an Unmanned Aerial System (UAS), it is not always available and can be jammed, introducing operational liabilities. When the GPS signal is degraded or denied, the UAS navigation solution cannot rely on incorrect positions GPS provides, resulting in potential loss of control. This paper presents a real-time pipeline for geopositioning functionality using a down-facing monocular camera. The proposed approach is deployable using only a few initialization parameters, the most important of which is the map of the area covered by the UAS flight plan. Our pipeline consists of an offline geospatial quad-tree generation for fast information retrieval, a choice from a selection of landmark detection and matching schemes, and an attitude control mechanism that improves reference to acquired image matching. To evaluate our method, we collected several image sequences using various flight patterns with seasonal changes. The experiments demonstrate high accuracy and robustness to seasonal changes.

Keywords: geopositioning; UAS; visual odometry; real-time pipeline

check for updates

Citation: Wei, J.; Yilmaz, A. A Visual Odometry Pipeline for Real-Time UAS Geopositioning. *Drones* **2023**, *7*, 569. https://doi.org/10.3390/drones7090569

Academic Editors: Hanno Hildmann and Fabrice Saffre

Received: 20 July 2023
Revised: 29 August 2023
Accepted: 1 September 2023
Published: 5 September 2023

1. Introduction

Geopositioning is arguably the most crucial step in navigating Unmanned Aerial Systems (UAS) either autonomously or by an operator. This is true for several tasks, including a safe return home, navigating to a designated destination, and assuring a safe flight. The geopositioning task is currently achieved using the onboard GPS antenna and sometimes with the addition of an inertial system that uses GPS as a correction signal. In cases where GPS is degraded, denied, or suffers from the multi-path problem, as in urban canyons [1], the drift caused by an inertial-only solution becomes unacceptable, and a backup geopositioning mechanism is required. This paper addresses the need to develop a real-time geopositioning solution using a down-looking camera generally available on most commercial drones. The proposed pipeline enables UAS geopositioning by matching the landmarks from the acquired image to a corpus of landmarks generated offline from a reference orthophoto available from either Google Earth or local aerial mapping [2].

Several challenges in vision-based UAS geopositioning include changes in the environment, such as new buildings, roads, etc., rapid changes in attitude, and the size of the region where the UAS will fly. The environmental changes between the reference and detected landmarks may be due to the intensity and direction of sunlight, seasonal changes, and changes in infrastructure. It is expected that a vision-based positioning solution should be robust to such changes. Additional algorithmic challenges arise from the scale changes between the reference region and UAS imagery.

To address the aforementioned challenges, we introduce an efficient pipeline shown in Figure 1 that uses a quad-tree as a geospatial landmark database, a landmark matching module, and an attitude control module. The geospatial database enables scalability and dynamically extracts and updates the look-up reference region based on the UAS motion. The dynamic updates to look-up regions and swift landmark retrieval from the database make the algorithm agnostic to how large a spatial area the flight would cover. The landmark matching module adopts several advanced matching algorithms, such as SuperGlue [3], to be resilient to environmental changes. The geometry control module offsets the UAS altitude change and ensures the UAS image is rectified during flight.

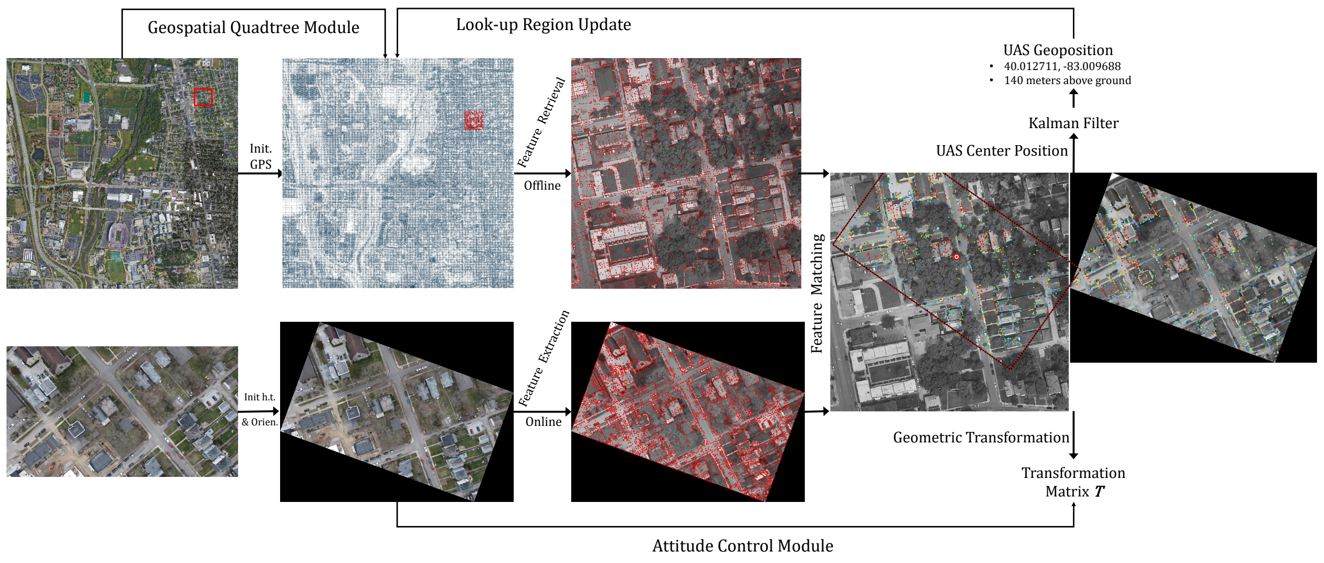

Figure 1. Overview of our proposed UAS geopositioning pipeline. The top left reference orthophoto from Google Earth is processed to generate the reference landmark dataset in a quad-tree data structure. Our algorithm requires initial geoposition to generate a list of reference landmarks from the database. Geometric relation between reference and UAS landmarks to convert pixel coordinates to latitude and longitude. Kalman filter is applied before output final predicted UAS geoposition to enhance predicted trajectory smoothness. With the new estimated UAS position, the look-up region is dynamically adjusted.

The contributions of this paper can be summarized as follows:

- Using an offline geospatial data structure for on-the-fly landmark set retrieval for matching. The database module uses a geospatial data structure to swiftly generate a list of landmarks from the look-up region while reducing computational and data bandwidths.
- Introducing a geometry control module to select appropriate scales that guarantee matching ground sampling distances across reference and UAS landmarks. This, in turn, enables the UAS to change its attitude and altitude during flight freely.
- Our approach achieves absolute UAS geopositioning with high accuracy in vertical and horizontal directions.
- In contrast to our earlier work [1], where the UAS was forced to fly at a fixed altitude and attitude, this paper provides an algorithm that removes fixed altitude and attitude constraints.

2. Related Works

Simultaneous Localization and Mapping (SLAM) is a promising solution for ground platforms that can be used for aerial platforms [4]. Without a very extended list, some of the SLAM algorithms used in this regard include MonoSLAM [5] and VINS-mono [6]. Researchers have also considered the cross-view matching approaches for positioning [7]. These methods utilize a Convolutional Neural Network (CNN) to extract landmarks from UAS-acquired images and reference satellite images and estimate the best-matching position in a query-to-target strategy [8].

2.1. SLAM-Based Solutions

SLAM techniques generate 3D point clouds and estimate the sensors' pose from sensory data. The UAS geopositioning problem can be considered a pose estimation problem; hence, SLAM techniques can help. The SLAM literature in UAS geopositioning uses visual, visual–inertial, and depth camera (RGBD) sensors [9]. Visual-only SLAM relies on one or more cameras to estimate the camera pose and 3D point by extracting and matching landmarks. Monocular SLAM is well explored and arguably considered the most adopted approach due to its reduced hardware requirements. Monocular SLAM is affected by drift due to dead reckoning. Estimating the metric scale is a problem unless ground control points (GCP) are provided [5,10]. Having multiple sensors mounted on the platform, such as an Inertial Measurement Unit (IMU) [6] along with cameras [11–13], generally helps with metric scale estimation; however, drift and loop closure remains a challenge and increases as the platform traverses longer distances. To mitigate the UAS pose estimation, our approach uses geotagged landmarks that can be considered (GCP).

2.2. Cross-View Deep Learning Approaches

Cross-view-based approaches estimate the latent relationship between UAS-collected imagery and the reference orthophoto available from, for instance, Google Earth. In Ref. [7], Sixing et al. proposed CVM-Net for the cross-view-based ground-to-aerial geopositioning task; while this approach was not proposed for UAS, one can apply the method as is to UAS positioning. CVM-Net uses the Siamese architecture to extract feature maps from acquired and reference imagery to perform a query-to-target search strategy. Akshay et al. [14] proposed a cross-view geopositioning method including a Scene Localization Network and a Camera Localization Network achieving UAS pose estimation from acquired UAS images and satellite images. Zhuang et al. [8] introduced the MSBA network architecture to extract landmarks from different views using a multi-branch structure. The network uses a single UAS image and satellite image gallery as input and estimates the UAS geoposition.

The cross-view approaches use attention mechanisms [15] within transformer architecture [16] to generate feature maps for the image; they significantly increase the computational cost for real-time applications running on the edge. The learned networks for the images learn the latent view geometric relations; hence, a slight UAS displacement may result in different feature maps and, consequently, mismatching of cross-view images. In addition, underlying neural networks trained for one site do not transfer to new areas and require retraining. The proposed approach in our paper does not require retraining and can adapt to new flight plans in unseen environments by replacing the look-up database of ground landmarks. The addition of the attitude control module allows UAS to undergo rotation and altitude fluctuations.

3. Methodology

The proposed approach comprises three modules: quad-tree look-up, landmark matching, and attitude control. First, the landmarks extracted offline from the reference orthophoto are saved in a quad-tree data structure. The landmarks extracted from UAS imagery are matched to landmarks retrieved from the quad-tree in real-time. We have tested with multiple landmark extraction and matching algorithms. The matching landmarks estimate the geometric transformation T and predict the UAS geoposition. The estimated geoposition is used within the quad-tree look-up module to update the dataset. The attitude control module uses the recovered transformation to estimate Euclidean platform rotation and altitude to update the UAS global rotation and altitude. These estimated values keep the acquired image and the landmarks extracted from the orthophoto with the same orientation and scale. The information flow of the proposed approach is shown in Figure 1.

3.1. Geospatial Quad-Tree

Considering the potentially extensive coverage of the UAS flight plans and the sheer size of reference imagery required for geolocalization (see Figure 2a), we use a quad-tree look-up module to store geotagged landmarks (see Figure 2b) from the orthophoto for fast landmark retrieval. The quad-tree is populated offline and can store extensive geographic areas due to its compressed size. Figure 2c shows example landmarks in red from the quad-tree. We used the UAS geoposition converted into coordinates of the reference orthophoto to retrieve the look-up region (shown in red rectangle box in Figure 2a) with a pre-defined range in pixels (we used 1400×1400 in experiments).

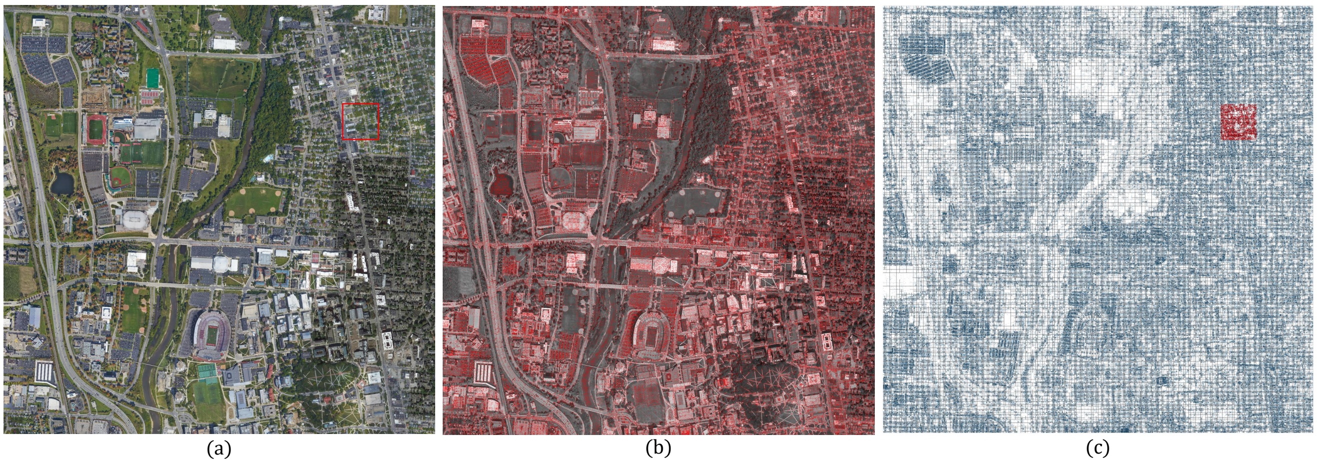

Figure 2. (**a**) is a 2.5×2.5 km^2 reference orthophoto obtained from Google Earth using Google Maps API. The red rectangle box indicates the look-up region corresponding to the estimated UAS geoposition. Reference orthophoto can also be generated using aerial mapping. (**b**) Automatically extracted landmarks with known geopositions that are stored in geospatial quad-tree data structure illustrated in (**c**).

Quad-tree is a tree-based data structure where each internal node has four children partitioning the space into four quadrants [17]. When each node reaches maximum capacity, it splits to extend the tree. The quad-tree recursively decomposes space into adaptive nodes, making it very efficient for geospatial indexing geographic regions, as shown in Figure 3. In Figure 4, we compare data retrieval efficiency between quad-tree and brute force methods. The inference time of a quad-tree exhibits a linear relationship with the retrieval area, while brute force execution maintains nearly constant performance, as it processes all data regardless of the size of the lookup region. Quad-tree takes less than 1% of the brute force's time to find appropriate landmarks demonstrating the retrieval efficiency of quad-tree for real-time UAS geopositioning applications. An added benefit of the quad-tree is its ability to increase geospatial coverage by updating the underlying tree; hence, we suggest using similar data structures.

3.2. Landmark Retrieval and Matching

Given the geoposition of a UAS, a list of landmarks is retrieved from the quad-tree data structure indexed by their geoposition. During the retrieval process, landmarks of an extended area are retrieved to reduce access to the quad-tree. As shown in Figure 5, the look-up region visualized within the aerial orthophoto overlaps with the acquired image at the UAS geoposition. We observe, however, that changes in the scene remain a significant challenge. For instance, differences in the image for a UAS flown in summer and winter using the same flight plan are shown in Figure 5b,c, where the vegetation appears significantly different. Besides seasonal differences, differences due to construction may also be observed, such as the gray building and its surrounding road network within the red box in Figure 5.

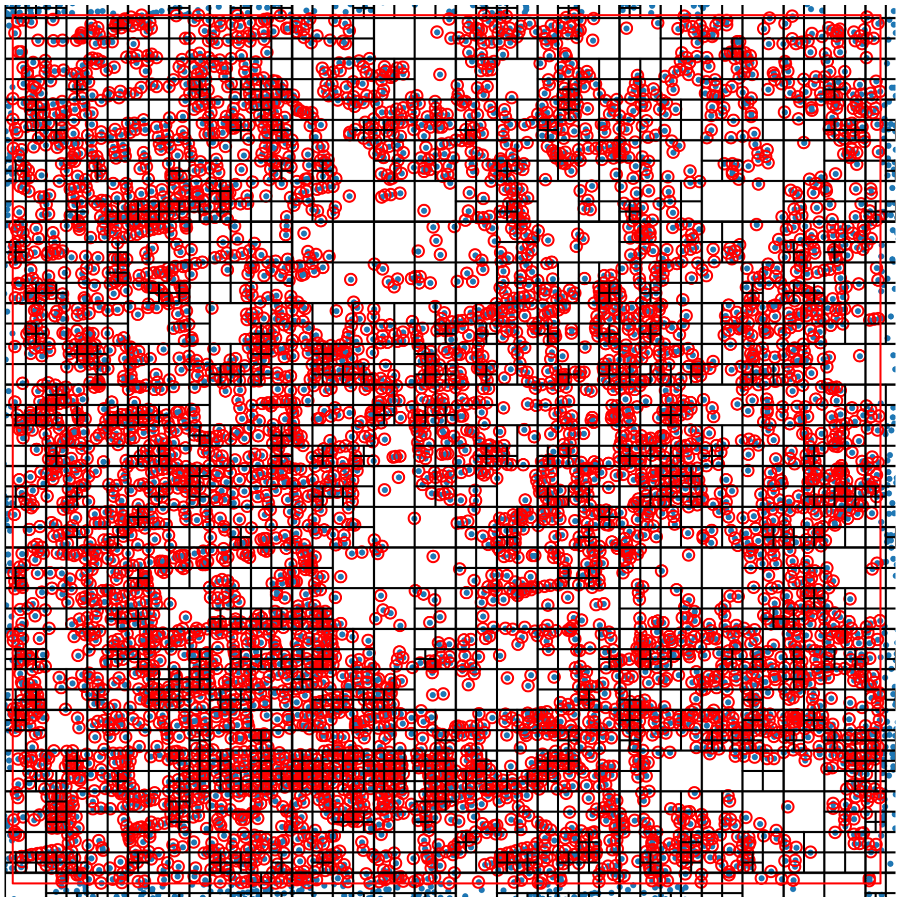

Figure 3. Visualization of the quad-tree for the reference orthophoto landmarks extracted offline.

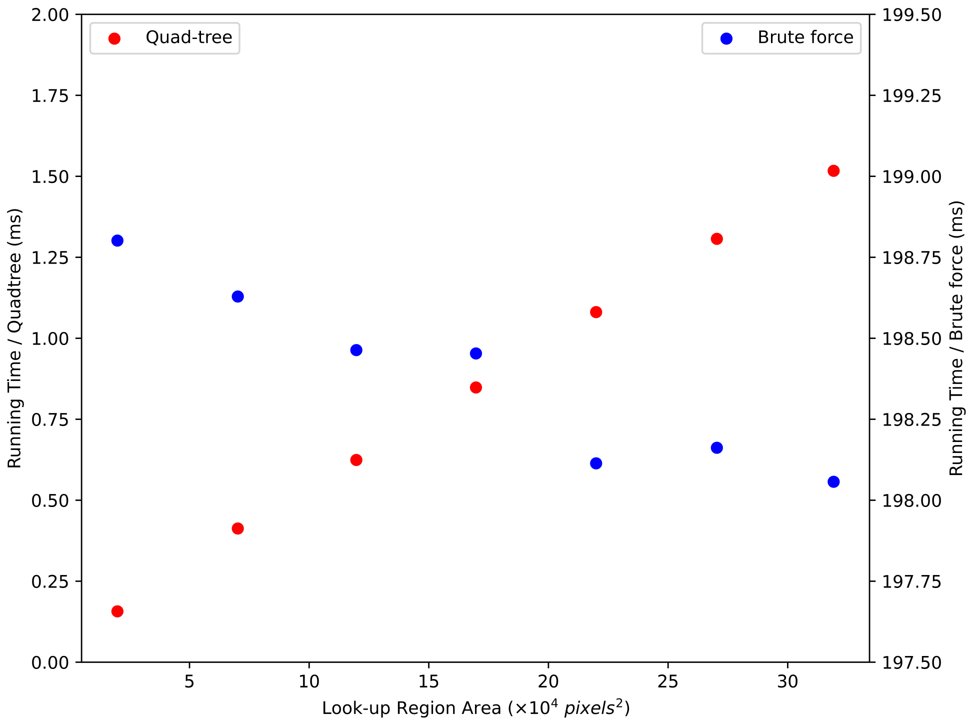

Figure 4. Efficiency of quad-tree compared against brute force search. It is important to note that the y-limit behaves differently in the left and right range considerations.

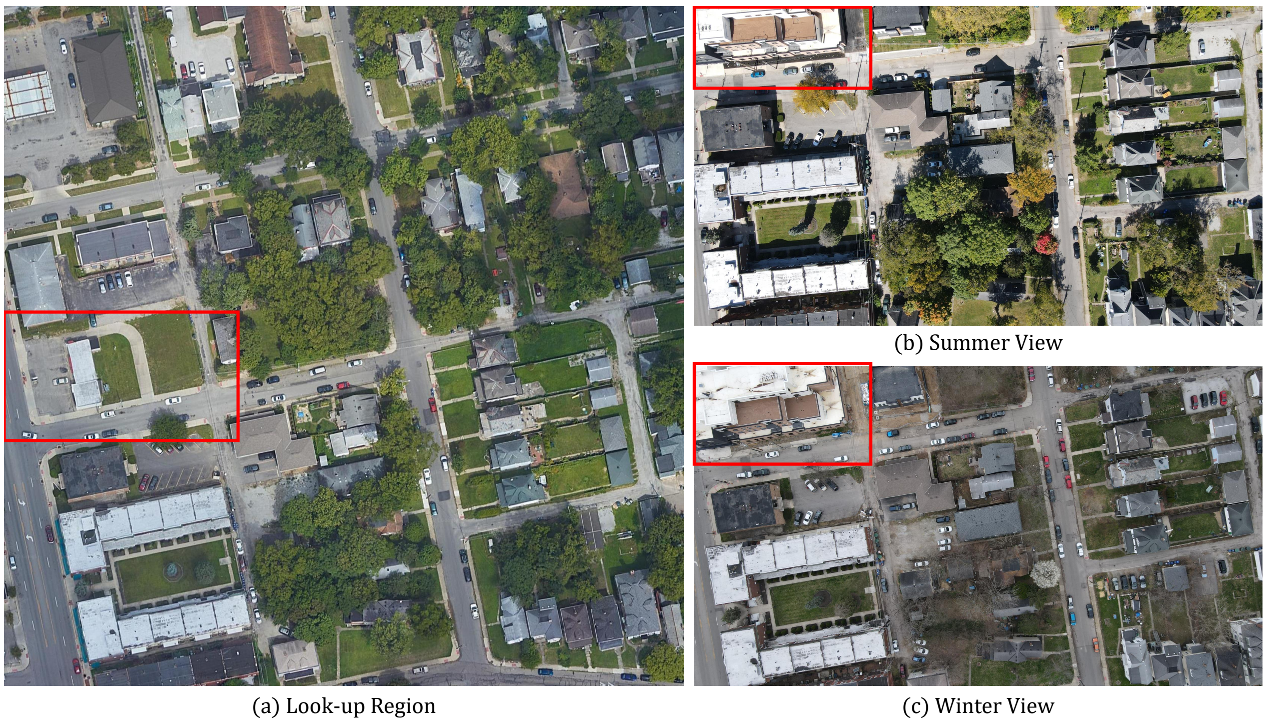

Figure 5. (**a**) The look-up region and landmarks from the reference orthophoto at the UAS geoposition. (**b**,**c**) UAS images acquired in summer and winter, respectively. Appearance differences can be seen due to seasonal changes and construction changes highlighted inside the red box.

The reference orthophoto generated from imagery is generally historical and is different from the imagery acquired during the UAS flight, and traditional landmark (keypoint) matching algorithms collectively perform poorly. We have selected Oriented FAST and Rotated BRIEF (ORB) [18] as a representative of traditional schemes and also adopted the more recent landmark detection/matching algorithm SuperGlue [3]. SuperGlue is observed to be resilient to seasonal changes and provides a more extensive set of matching landmarks from the reference orthophoto and the acquired USA image. This is illustrated in Figure 6, where the algorithm provides matches despite construction changes and seasonal differences.

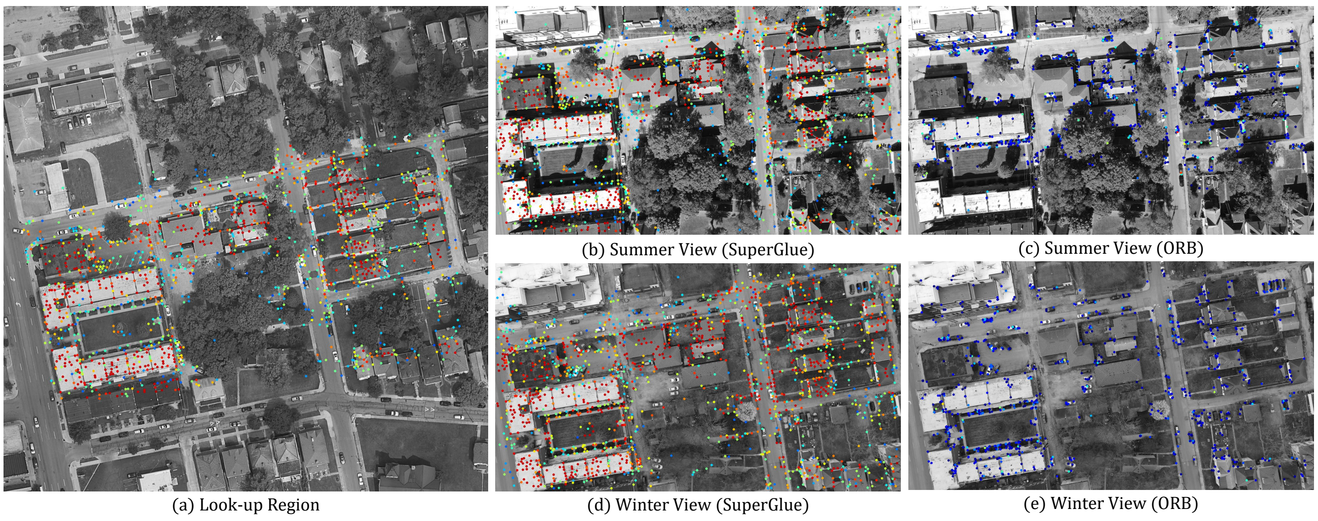

Figure 6. (**a**) Look-up region from the reference orthophoto superimposed with matched landmarks. (**b**∼**d**) UAS acquired images with superimposed matched landmarks for two seasons and changes in the building layout due to construction.

Apart from the above challenges in matching, another essential characteristic typically addressed in most of the published literature is the invariance of the matching schemes to rotation and scaling observed frequently in the UAS geopositioning problem as the platform changes altitude and attitude. We have tested ORB and SuperGlue matching performance under attitude changes (5°–50°) and altitude changes (60–100 m). As shown in Figure 7, SuperGlue performs comparably better than ORB, but its performance drops significantly after 40°. The effect of altitude change on landmark matching is shown in Figure 7.

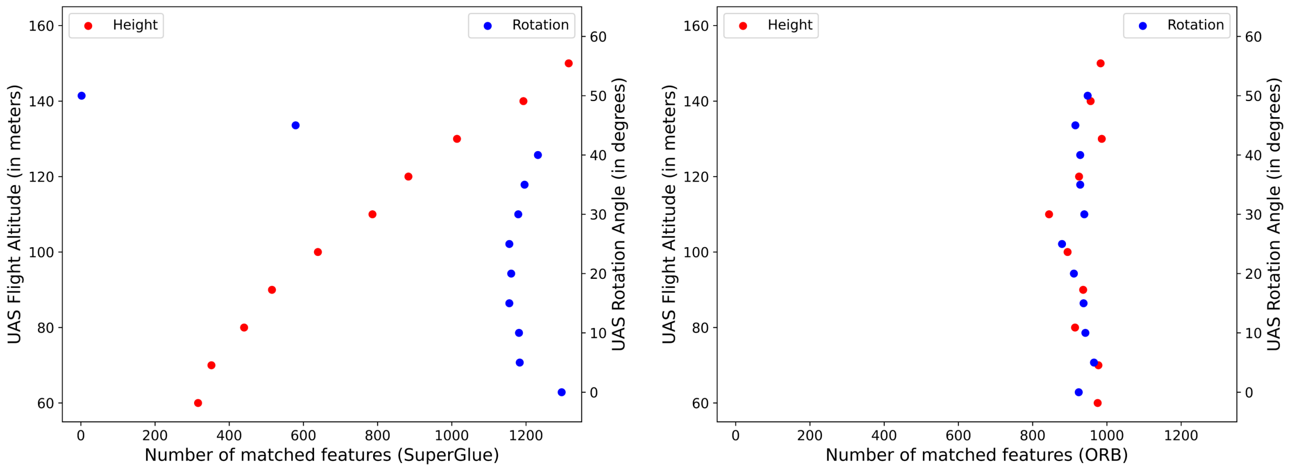

Figure 7. The effect of changes in the attitude on the matching to orthophoto using the SuperGlue and ORB algorithms. Attitude changes are common during UAS flight, and invariance is critical for successful geopositioning. We note that generally, SuperGlue works well in rotations smaller than 40°. Beyond 40°, the performance significantly reduces. ORB's landmark count is consistent at around 1000 due to its internal settings as a handcrafted landmark extractor.

3.3. Geometric Transformation

The geometric transformation T between the UAS-acquired image and the reference orthophoto can be estimated using the matching landmarks. The following discussion will explain the relation in pixel coordinates that can be modified to use the geopositions of the reference landmarks. In our implementation, we use a down-looking camera with a stabilizer generating images with landmarks at $\mathbf{x}_I$. Transformation matrix between $\mathbf{x}_I$ and landmark $\mathbf{x}_L$ from the reference orthophoto is

$$\mathbf{x}_L = \mathtt{T} \times \mathbf{x}_I. \tag{1}$$

The transformation between the down-looking UAS image and the reference data can be computed using 2D transformations as detailed in [19]. In Section 4 of this paper, we compare the pseudo-perspective, affine, and homography transformations listed in Table 1. The pseudo-perspective transformation approximates a perspective transformation using a simpler model that includes rotation and translation and a scale factor based on the object's distance from the camera. The affine transformation adds shearing and independent scaling in the directions of x and y. The homography introduces perspectivity to the affine transformation. We should note that since the reference data are "orthophoto" theoretically, the pseudo-perspective is a more appropriate choice than homography.

Table 1. Transformation variants in 2D space.

	DoF [1]	**Matrix**	**Proprieties**
Pseudo-Perspective	4	$\begin{bmatrix} sr_{11} & sr_{12} & t_x \\ sr_{21} & sr_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$	Rotation, Translation, Scaling
Affine	6	$\begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$	Pseudo-Perspective, Shearing
Homography	8	$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix}$	Affine, non-Parallelism

[1] Degree of Freedom.

In Figure 8, we show an illustration where the UAS flies over the building and acquires two consecutive images CAM_t and CAM_{t+1}. In Figure 8a,b, the ground landmarks B_t and B_{t+1} intersect on the ground, and the building landmark A intersects on the building roof. The difference is that in the projective model, the building landmark A on the roof was projected on the ground as A_t and A_{t+1}, resulting in shearing and non-parallelism. In the pseudo-perspective model, landmark A was vertically projected onto the ground, which preserves the shearing effect and keeps lines parallel. Figure 8c presents an orthographic camera model; all lines are parallel in both images. This ideal state does not practically exist unless a digital elevation model is used along with the image to generate the orthographic image.

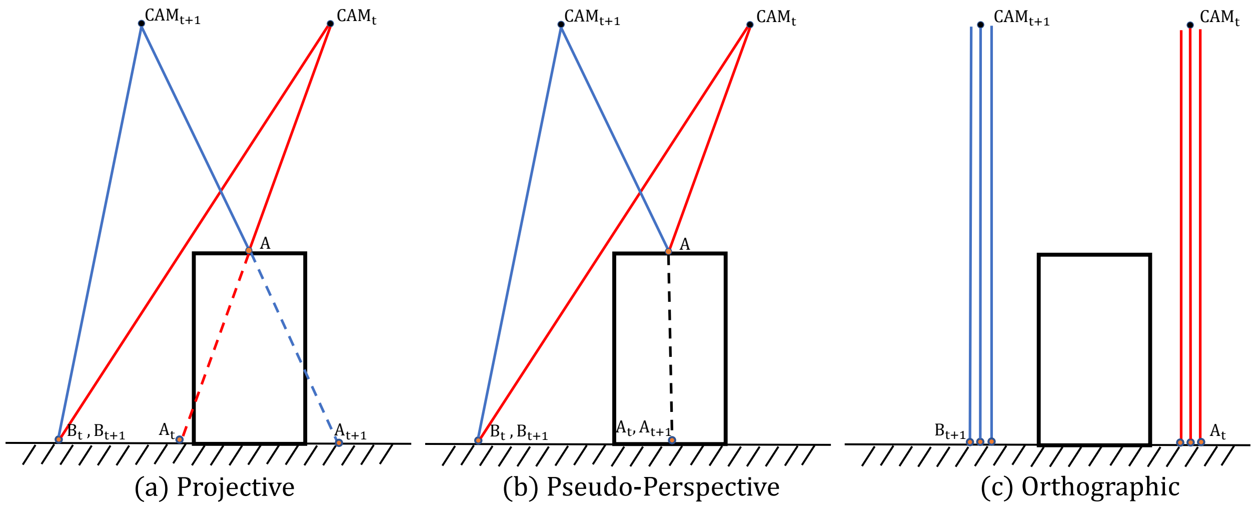

Figure 8. (**a**) Assuming camera model is a pinhole, features from the 3D world coordinates are projected to 2D image coordinates. Taking the ground plane as the reference, the features on the building roof diverge when extending the projection lines to the ground. (**b**) The pseudo-perspective model is a constrained projective model that assumes 3D features have the same depth by ignoring the height of the building. (**c**) The orthographic model is based on the orthographic projection; it preserves parallel lines and does not contain perspective effects.

3.4. Attitude Control

As was discussed, attitude changes cause landmark matching problems (see Figure 7), which results in geopositioning errors. As shown in Tables 2 and 3, with the change in altitude, especially when the platform is below 100 m, landmark matching efficiency significantly reduces, resulting in geopositioning errors. This is due to the scale difference between reference orthophoto and UAS-acquired images.

Table 2. The effect of UAS altitude on landmark matching efficiency by comparing the number of matches (SuperGlue).

Altitude (m)	**Not Adjusted** [1]			**Adjusted**		
	Feat. [2]	**Matches** [3]	**Pct. (%)** [4]	**Feat.**	**Matches**	**Pct.**
50	3488	343	9.83	1059	301	**28.42**
60	3568	406	11.38	1459	343	**23.51**
70	3749	470	12.54	1947	396	**20.34**
80	3954	546	13.81	2434	510	**20.95**
90	4085	639	15.64	2956	604	**20.43**
100	4137	746	18.03	3489	700	**20.06**

[1] Disabling altitude control applied using Equation (3). [2] Extracted landmarks from UAS imagery. [3] Matched landmarks to the reference dataset. [4] Valid landmarks pct., matches/feat.

Table 3. The effect of UAS altitude on landmark matching efficiency by comparing the number of matches (ORB).

Altitude (m)	**Not Adjusted** [1]			**Adjusted**		
	Feat. [2]	**Matches** [3]	**Pct. (%)** [4]	**Feat.**	**Matches**	**Pct.**
50	4000	838	20.95	2872	909	**31.65**
60	4000	968	24.20	3398	993	**29.22**
70	4000	953	23.83	3753	955	**25.45**
80	4000	996	24.90	3893	1008	**25.89**
90	4000	994	**24.85**	3965	931	23.48
100	4000	1000	**25.00**	4000	925	23.13

[1] Disabling altitude control applied using Equation (3). [2] Extracted landmarks from UAS imagery. [3] Matched landmarks to the reference dataset. [4] Valid landmarks pct., matches/feat.

To overcome the landmark-matching problems related to rotation and scaling, we designed an attitude control module that enables UAS to freely change attitude and altitude during flight while ensuring good landmark matching despite the scale and orientation differences between the acquired image and the reference dataset. Using the pseudo-perspective transformation, we estimated the control module transformation as follows:

$$P_L = \underbrace{\begin{pmatrix} \cos\theta_{\text{offs}} * s_{\text{offs}} & -\sin\theta_{\text{offs}} * s_{\text{offs}} & t_x \\ \sin\theta_{\text{offs}} * s_{\text{offs}} & \cos\theta_{\text{offs}} * s_{\text{offs}} & t_y \end{pmatrix}}_{S_{2\times 3}} * P_I \tag{2}$$

where θ_{offs} is the rotation offset in reference to the north, (t_x, t_y) is UAS translation, s_{offs} is the offset ratio from landmark scale between resized UAS acquired image and the landmarks generated from the reference orthophoto. For a UAS with a fixed focal length camera, the scale has a linear relationship with the drone's altitude (see Figure 9):

$$Scale = 0.01114674 + 0.0895404 * Altitude \tag{3}$$

where the slope and intercept are computed from the plot Figure 9, and the slope would change with a change in lens parameters. This equation converts the initial UAS attitude A_0 to the same scale at which the look-up landmark descriptors are generated. By activating the attitude control, we observe increased accuracy and precision in landmark matching as shown in Table 1, especially at low attitudes. The algorithm used by the attitude control is given in Algorithm 1.

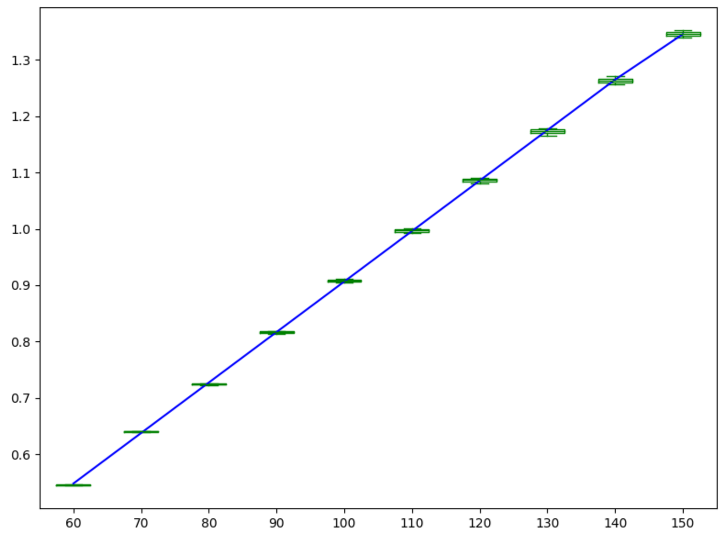

Figure 9. Illustration of the UAS flight height and scale relationship (blue). At each height, we collected over 200 observations with descriptive statistics visualized as candlesticks (green).

Algorithm 1 Attitude control module working process

```
Input: $\theta_{t-1}, s_{t-1}$
Output: $\theta_t, s_t$
  if $N_{matches} \geq T$ then
    $S_{2\times3} = P_I{}^{-1} * P_L$
    $\theta_{offs} = \arctan(s_{21}/s_{11})$                ▷ $s_{11}, s_{21} \in S_{2\times3}$
    $s_{offs} = s_{11}/\cos\theta_{\text{offs}}$
    $s_t = s_{t-1} * s_{\text{offs}}$ and $\theta_t = \theta_{t-1} + \theta_{\text{offs}}$
  else
    $s_{\text{offs}} = 1$ and $\theta_{\text{offs}} = 0$
    $s_t = s_{t-1}$ and $\theta_t = \theta_{t-1}$
  end if
```

While we conjecture the following should not differ for various platforms, in Algorithm 1, T, the threshold for matching landmarks, is empirically set to 50 for the UAS we use in our experiments. To ensure the attitude control module works on other UAS platforms, the scale–altitude linear relationship in Equation (2) must be estimated due to camera focal length changes.

4. Experiments

In this section, we evaluate the proposed UAS geopositioning approach using two variations of landmark matching approaches—ORB and SuperGlue. We use the pre-trained SuperGlue model without additional training. The experiments section is organized into several subsections. First, we discuss the in-house generated datasets and UAS geopositioning accuracy under several scenarios. This is followed by an analysis of the effectiveness of the Geographic Information System (GIS) [20] filter generated using OpenStreetMap [21]. We compare two matching schemes and validate our pipeline running in real-time. All experiments are coded and tested with a 12GB NVIDIA TITAN X GPU. Finally, the geopositioning accuracy of the proposed approach is shown to outperform a common SLAM approach.

4.1. Datasets

The dataset is collected using a DJI Mavic Air 2 drone with an onboard camera. Several flight plans were designed to address various scenarios, including suburban residential areas, university campuses, surface parking lots, and farmlands. We force UAS to fly over

100 m above the ground to ensure flight safety and sufficient features to be collected. Over a flight altitude of 100 m, little variation such as moving vegetation due to wind could be ignored. Please see Figure 10 for sample images from each flight plan. The images are acquired every 3 m at a platform speed of 2.5 m/s. The attitude and altitude of UAS are allowed to change during flight with a maximum angular velocity of 250 °/s, an ascent speed of 4 m/s, and a descent speed of 3 m/s. The UAS speed is limited to 18 m/s. The UAS reference geoposition position has been extracted within a hovering accuracy range of $\pm$1.5 m horizontally and $\pm$0.5 m vertically.

We stored an approximate initial take-off geoposition ($\pm$10 m) and orientation ($\pm 10^\circ$) of UAS for each flight plan. The scaling relationship discussed earlier is embedded into the geopositioning algorithm. We should note that, in practice, the proposed approach has high initial position and attitude tolerances. Experimental evaluations show that the initial geoposition and attitude accuracy can be within $\pm$120 m and $\pm 40^\circ$, respectively. We would like to note that, compared with our approach, alternative geopositioning methods, such as variations of SLAM, require highly accurate initialization.

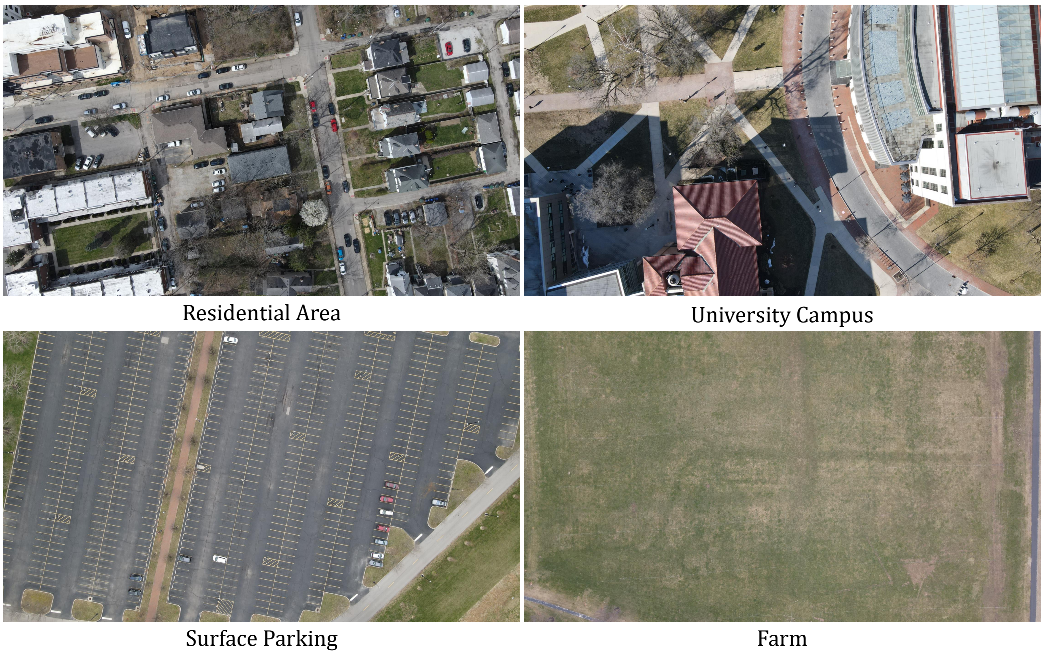

Figure 10. Example images from the dataset containing flight plans for scenarios including residential, university campus, surface parking, and farmland.

4.2. Evaluation

To assess the results qualitatively, we show the predicted UAS trajectories based on the pseudo-perspective projection under different scenarios. The yellow trajectories represent the predicted trajectory compared with the red ones generated from reference GPS sensor readings. Quantitative accuracy and the details of the flight plans from the dataset are given in Table 4. Each flight plan is provided in a column with the information given in the first four rows: traversed distance, initial attitude (orientation) and altitude, and the altitude change for each plan. This is followed by two additional blocks of four rows detailing the geopositioning accuracy with the minimum, maximum, mean absolute error (MAE), and the standard deviation of the predicted horizontal and vertical geoposition against UAS reference geoposition during the flight. Due to the limited reference sensor geopositioning accuracy mentioned in Section 4.1, we reported all quantitative results except the standard deviation in decimeters. For all the scenarios, the MAE for the proposed approach is below 7.1 m horizontally and below 4.5 m vertically, and is within the uncertainty of the onboard GPS sensor positioning specs.

Table 4. The flight-plan information (first quad) and the predicted horizontal (second quad) and vertical (third quad) geopositioning quantitative analysis against reference geospatial data.

	Resi.	**Campus**	**PKG.**	**Farm1**	**Farm2**
Dis. (km)	2.2	0.78	1.4	2.1	1.4
Init. Orien. [1]	0°	0°	60°	90°	180°
Init. Alt. (m)	140	140	90	130	140
Alt. Fluct.(m) [2]	105–140	110–140	90–145	120–145	130–149
Min. Err. (m) [3]	1.5	2.1	3.9	0.1	1.0
Max. Err. (m) [3]	10.1	9.6	9.7	7.4	5.1
MAE (m) [3]	6.0	6.3	7.1	4.0	3.4
Std (m) [4]	1.74	1.74	1.07	1.54	1.07
Min. Err. (m) [5]	0.0	0.0	2.0	0.0	0.0
Max. Err. (m) [5]	12.3	6.4	6.1	5.30	4.1
MAE (m) [5]	2.9	1.8	4.5	1.8	1.2
Std (m) [6]	3.1	1.53	0.70	1.25	1.05

[1] UAS initial clockwise orientation in reference to North direction. [2] UAS altitude fluctuation during the whole testing flight. [3] UAS horizontal geopositioning (GPS level) accuracy in meters. [4] UAS horizontal geopositioning (GPS level) precision. [5] UAS vertical geopositioning (Altitude level) accuracy in meters. [6] UAS vertical geopositioning (Altitude level) precision.

4.3. GIS Filtering and Its Effectiveness

During the generation of the landmark dataset, detected landmarks in the geo-rectified orthophoto are kept or removed from the dataset depending on the type of object to which they belong. To facilitate this filtering, as discussed in [1], the geopositions of the landmarks are checked against the OpenStreetMap GIS information, and the landmarks that belong to buildings are removed. The resulting set of points generally lay on the roads and greenery. Noting that this filtering is offline, it does not increase the processing time of the platform geopositioning.

The effect of GIS filtering on geopositioning can be analyzed in horizontal plane coordinates and platform attitude. The quantitative comparison for geopositioning accuracy with and without GIS filtering is shown in Tables 5 and 6, respectively, in horizontal and vertical directions. The table lists minimum, maximum, and mean (MAE) errors and the standard deviation of error. We can see from the results that pseudo-perspective and affine transformations provide better results than the homography transformation, especially when GIS filtering is enabled. We should note, however, that the horizontal accuracy does not change significantly with or without the GIS filter for the affine and pseudo-perspective transformations.

Table 5. Horizontal UAS geopositioning accuracy tabulated by minimum, maximum, mean absolute error, and standard deviation (in meters) under three transformation variants and activating or deactivating the GIS filter.

GIS Filter	**Transformation**	**Residential Area**				**University Campus**				**Parking**				**Farm1**			
		Min.	**Max.**	**MAE**	**Std.**	**Min.**	**Max.**	**MAE**	**Std.**	**Min.**	**Max.**	**MAE**	**Std.**	**Min.**	**Max.**	**MAE**	**Std.**
Activated	Pseudo-perspective	1.5	**10.1**	**6.0**	1.74	2.1	**9.6**	6.3	1.74	3.9	9.7	**7.1**	1.07	**0.1**	**7.4**	**4.0**	**1.54**
	Affine	1.6	10.4	**6.0**	**1.69**	2.4	9.7	6.3	1.71	3.9	**9.5**	**7.1**	**1.06**	**0.1**	7.6	4.1	1.55
	Homography	0.9	16.0	7.3	2.68	1.6	18.5	7.6	3.15	4.4	16.0	8.0	1.74	0.3	11.1	5.0	2.26
Deactivated	Pseudo-perspective	1.5	**10.1**	**6.0**	1.71	1.9	9.8	**5.7**	1.61	3.9	9.7	**7.1**	1.08	**0.1**	7.7	**4.0**	**1.54**
	Affine	1.7	10.3	**6.0**	1.70	2.4	9.8	**5.7**	**1.57**	3.7	**9.5**	**7.1**	1.08	**0.1**	7.8	4.1	**1.54**
	Homography	**0.8**	15.7	7.4	2.80	**1.0**	26.9	8.6	4.60	**3.2**	27.1	8.3	2.40	0.3	33.7	5.1	2.96

Table 6. Vertical UAS geopositioning accuracy reported in minimum, maximum, mean absolute error and standard deviation (in meters) under pseudo-perspective and affine transformations and activating or deactivating GIS filter.

GIS Filter	**Transformation**	**Residential Area**				**University Campus**				**Parking**				**Farm1**			
		Min.	**Max.**	**MAE**	**Std.**	**Min.**	**Max.**	**MAE**	**Std.**	**Min.**	**Max.**	**MAE**	**Std.**	**Min.**	**Max.**	**MAE**	**Std.**
Activated	Pseudo-perspective	0.0	12.3	**2.9**	**3.10**	0.0	**6.4**	**1.8**	**1.53**	2.0	6.1	4.5	0.70	0.0	**5.3**	**1.8**	**1.25**
	Affine	0.0	12.1	3.1	3.24	0.0	6.5	**1.8**	1.56	1.8	6.2	4.5	0.67	0.0	6.1	1.9	1.35
Deactivated	Pseudo-perspective	0.0	**12.0**	3.4	3.40	0.0	29.5	6.2	8.28	2.1	**6.0**	4.5	**0.66**	0.0	5.5	**1.8**	**1.25**
	Affine	0.0	12.2	3.4	3.38	0.0	33.7	6.6	8.59	**0.6**	6.1	**4.4**	0.68	0.0	5.7	1.9	1.30

In contrast to the more minor horizontal errors in Table 5, the attitude errors in Table 6 are affected significantly by the application of the GIS filter, especially for the university campus area (see Figure 11), where features from building tops and the ground have different heights. The predicted altitude and reference altimeter readings are plotted as a function of time shown in Figure 12. When the platform flies over high buildings, most matching features are from the building tops. Without a GIS filter, those building features are mistakenly assumed on the ground, thus increasing the positioning error. On average, without the GIS filter, the MSE and maximum errors of altitude estimation are, respectively, 6.22 and 29.50 m. When the GIS filter is activated, the MSE decreases to 1.89 m with a 69.6% improvement, and the maximum error decreases to 6.40 m with a 78.3% improvement. This is visible in regions 1, 2, and 3, shown in Figure 13, where the UAS flies over tall buildings such as the Thompson Library or the RPAC building at the Ohio State University Campus. In multiple experiments at the UAS flying altitude of 110 m, we observed that buildings below 15 m do not violate the constraints for the pseudo-perspective and affine transformations, and the GIS filter is unnecessary. Our experiment results from the *parking* and *Farm*1 columns in Table 6 validate this.

4.4. SuperGlue vs. ORB

Compared to ORB and earlier approaches, we consider the SuperGlue to be arguably the next-generation detection and matching approach using neural networks. From this perspective, the ORB and SuperGlue tests result in two outcomes. The first one is about the detection and matching success rate, especially in the presence of a historical landmark dataset. SuperGlue is tested superior to ORB when the reference landmark features are generated from infrequently updated Google Maps aerial orthophotos. The higher number of matching features from SuperGlue results in better estimating the transformation matrix. This advantage vanishes when the reference landmarks are generated from a recent aerial mapping. Figure 14 shows the qualitative matching performance of SuperGlue vs. ORB. Quantitative results for more recently acquired landmark datasets are shown in Table 7. It can be observed that both SuperGlue and ORB work well and achieve similar accuracy in UAS geopositioning where the speed of error on average is 3.4 m/s. The predicted trajectories are visualized in Figure 15 and are compared with the reference trajectory (plotted in red). Yellow and blue trajectories almost overlap, which agrees with the quantitative analysis in Table 7.

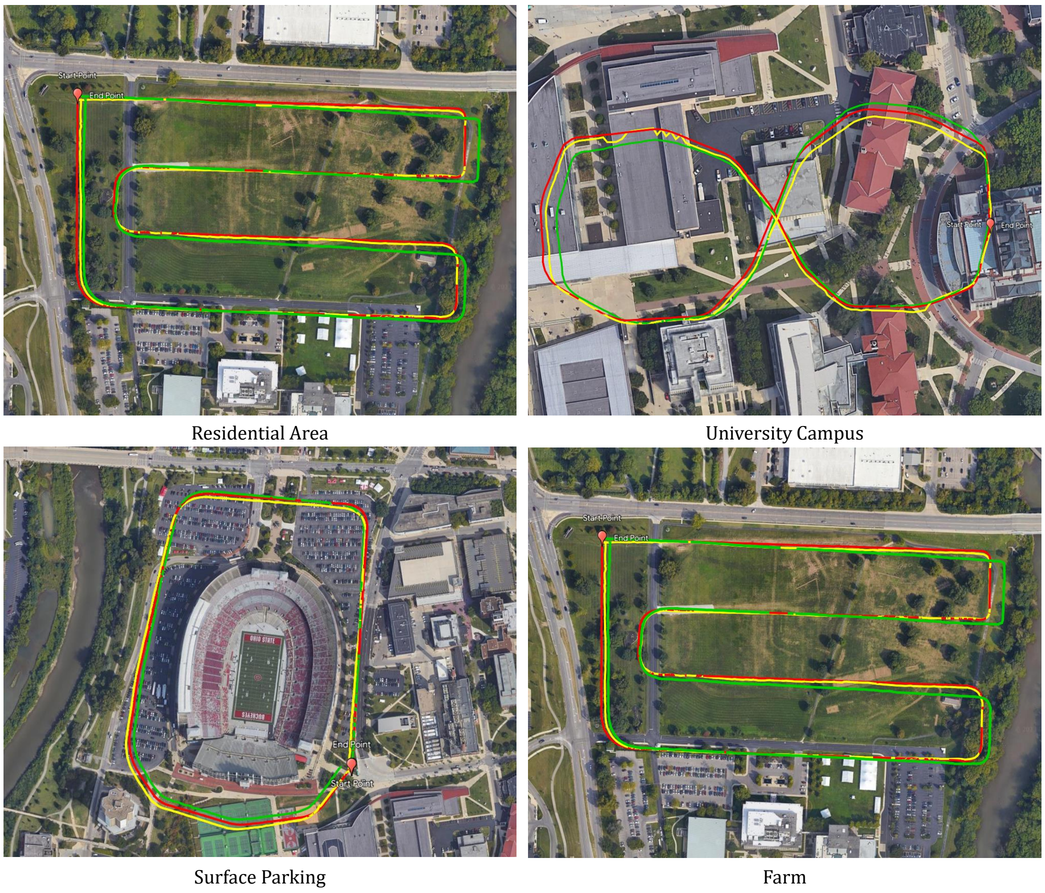

Figure 11. In horizontal space, four scenarios superimposed with GPS sensor readings during the flight are shown in red, the predicted geopositions using our approach in yellow and ORB-SLAM3 in monocular mode predicted geopositions in green. The reference and predicted start and end positions are the red pin icons. Vertical results can be seen in Figure 16.

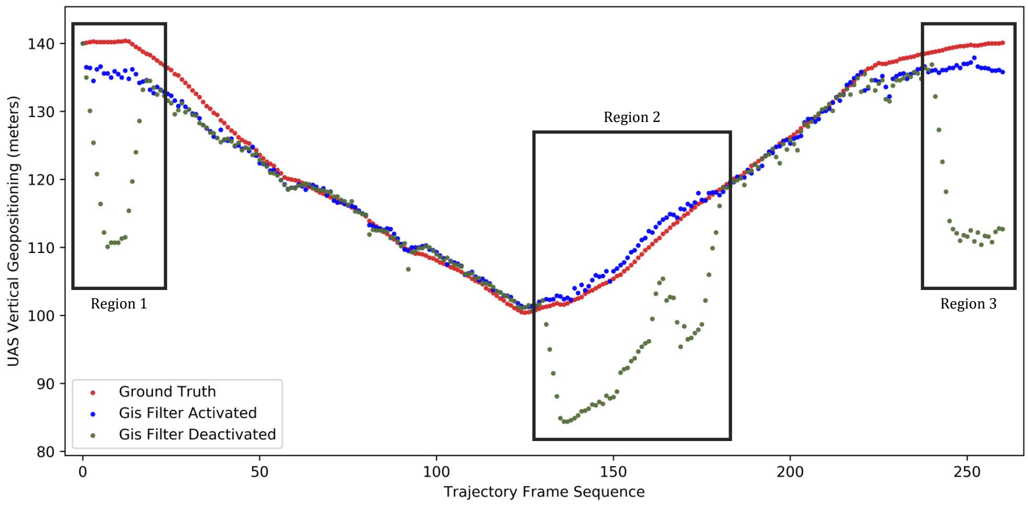

Figure 12. UAS reference geoposition (red) and estimated UAS geopositioning. The blue trajectory shows the GIS filter-activated trajectory, and the green shows the deactivated trajectory for pseudo-perspective transformation. The GIS filter significantly improves the predicted altitude in regions 1, 2 and 3.

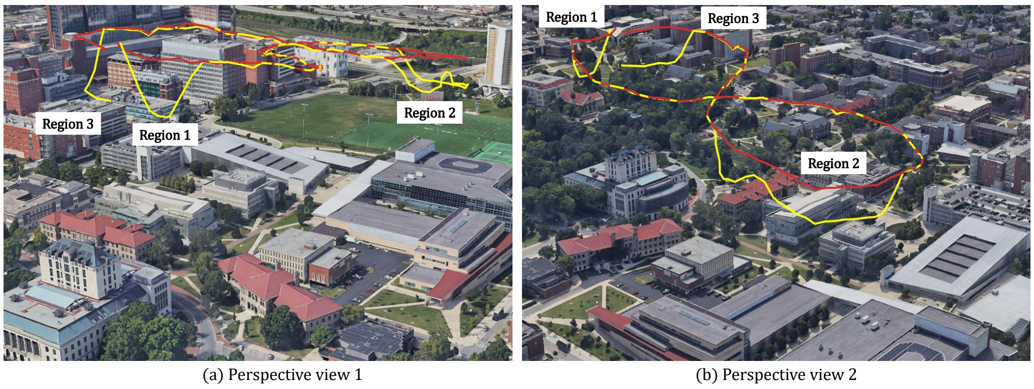

Figure 13. Visualization of the UAS vertical geopositioning from two perspective views on Google Earth, activating the GIS filter (red) vs. deactivating it (yellow). The regions 1∼3 correspond to the same regions in Figure 12.

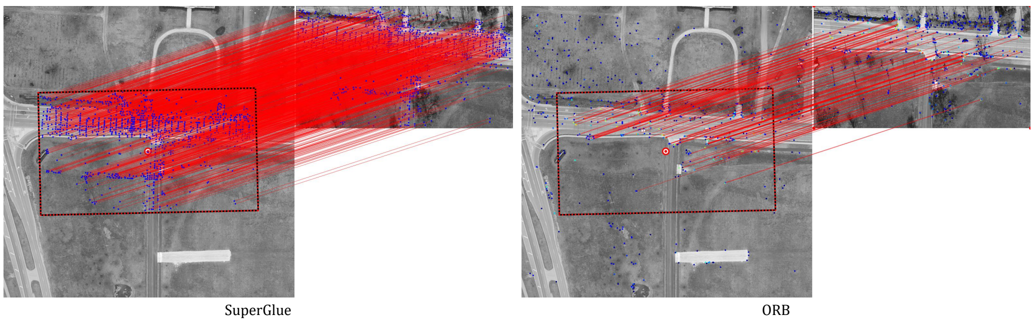

Figure 14. Landmark matching performance of SuperGlue (**left**) vs. ORB (**right**). The aerial orthophoto used for the comparison is generated by aerial mapping.

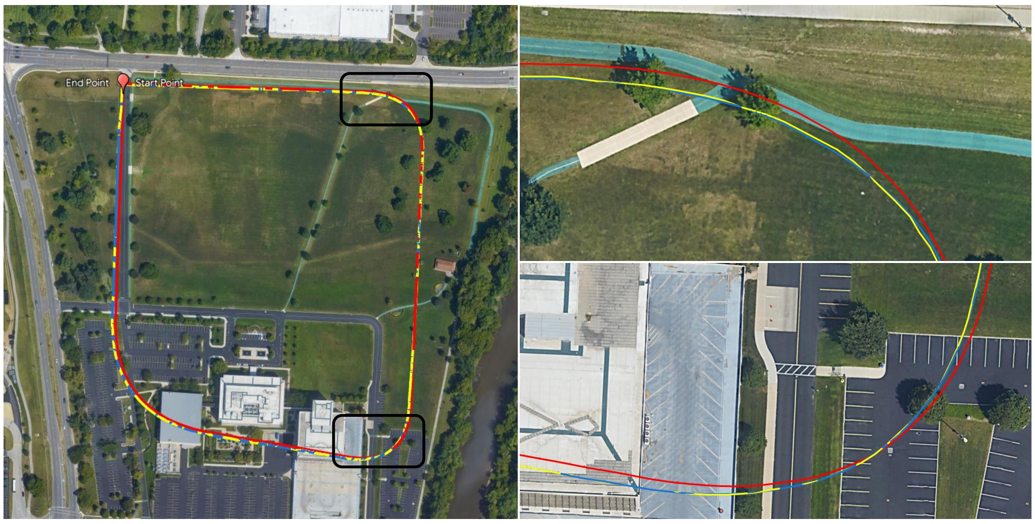

Figure 15. Geopositioning test at a farm site (referred to *Farm2* in Table 4), SuperGlue (yellow) vs. ORB (blue). The red is the reference trajectory from the platform GPS sensor. The right figures show zoomed-in trajectories of the black boxes in the left image.

Another outcome of this experiment is the practical use of the detection and matching scheme for real-time edge computing applications. The ORB-based pipeline has a higher processing speed averaging 7 fps which is twice the speed of the SuperGlue-based pipeline. During the flight, the faster processing allows faster flight and a larger lever arm between consecutive image acquisitions (see Table 7 for details).

Table 7. UAS geopositioning pipeline landmark matching scheme (SuperGlue vs. ORB) comparison on horizontal geopositioning accuracy, pipeline inference speed, and allowed UAS maximum speed for high-end GPUs.

	Min. [1]	**Max.** [1]	**MAE** [1]	**Std** [1]	**fps** [2]	**dist** [3]	**Max. Speed** [4]
SG [5]	1.0	5.1	3.4	1.08	3.40	33	112
ORB	1.0	5.1	3.4	1.06	6.58	90	592

[1] Measures of UAS geopositioning accuracy, meter(s). [2] UAS Geopositioning pipeline average running speed, f/s. [3] Allowed maximum distance between two consecutive frames, m/f. [4] Max. speed = fps × gap, m/s. Theoretically, this is the allowed maximum speed under the promise of ensuring the pipeline run in real time. [5] SuperGlue.

4.5. Comparison with SLAM

SLAM-based approaches rely on only visual odometry or its combination with inertial measurements. For a fair comparison, we chose a recent monocular SLAM approach. Considering that SLAM solutions are relative, we initialized the camera pose with the altitude and attitude of the platform. For comparison, we use ORB-SLAM3, the most recent version of the ORB-SLAM series [11–13].

In Figures 11 and 16, we show horizontal and vertical geopositioning results generated from our approach (blue trajectory) and ORB-SLAM3 (green trajectory), respectively, compared with reference onboard GPS sensor and altimeter readings (red trajectory). We can see from the figures that the relative solution of SLAM results in drift in the both horizontal plane and altitude without any external correction. The error for the altitude estimation is much higher than the horizontal error due to the better spatial distribution of landmarks in the horizontal plane.

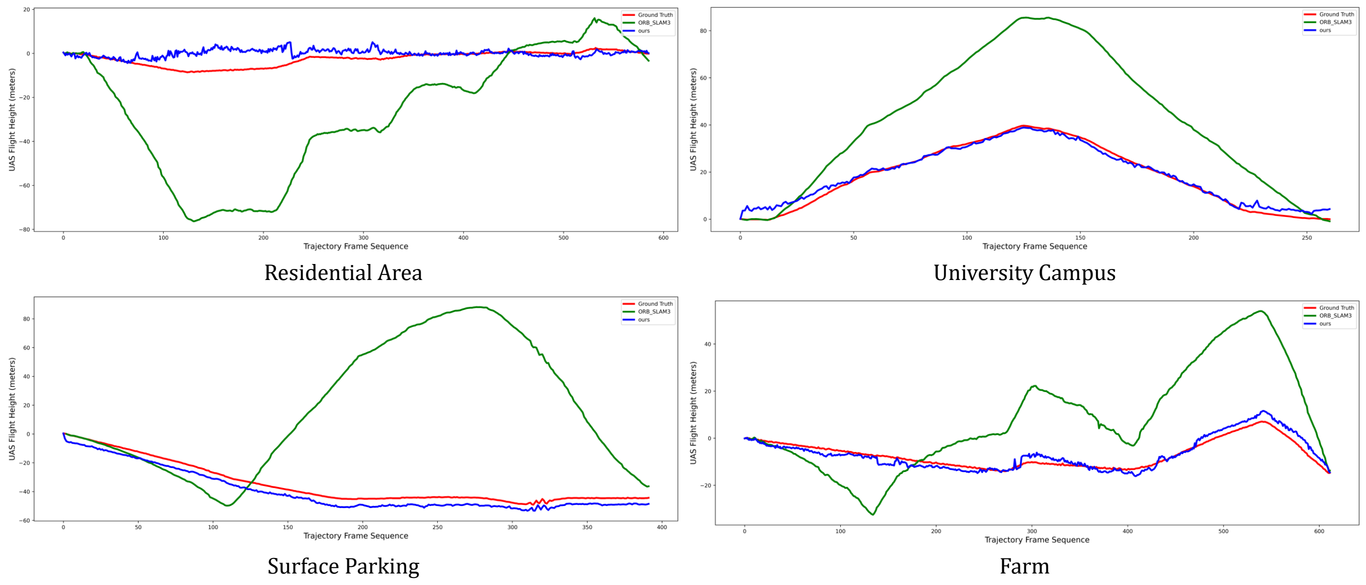

Figure 16. Horizontally, we compare our approach (blue trajectory) with monocular ORB-SLAM3 (green trajectory) and the reference altimeter readings (red trajectory). Under each scenario, the horizontal comparisons are shown in Figure 16.

5. Conclusions

In this work, we have introduced a real-time positioning pipeline that can provide absolute 3D space geopositioning and works better than state-of-the-art approaches. In this approach, the UAS can fly in extensive geospatial coverage and change attitude and altitude without losing vertical and horizontal accuracy. To achieve this performance, we introduce a fast geospatial database containing geo-tagged and labeled landmarks and an attitude control mechanism to mitigate matching problems due to platform motion.

The pipeline can be extendable and contain other modules. We included a GIS filtering module that improves vertical errors to demonstrate this. Compared with recent SLAM methods, our approach provides real-time absolute geoposition and does not suffer from drift problems.

Author Contributions: Conceptualization, J.W. and A.Y.; methodology, J.W. and A.Y.; software, J.W.; data curation, J.W.; writing—original draft preparation, J.W.; writing—review and editing, J.W. and A.Y.; visualization, J.W.; supervision, A.Y.; project administration, A.Y.; funding acquisition, J.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The source code and data presented in our experiments are available on request from the corresponding author. Source code is available at https://github.com/JianliWei1995/UAS-Visual-Geopositioning.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CNN	Convolutional Neural Network
GCP	Ground Control Point
GIS	Geographic Information System
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
IMU	Inertial Measurement Unit
ORB	Oriented FAST and Rotated BRIEF
SLAM	Simultaneous Localization and Mapping
UAS	Unmanned Aerial System

References

1. Wei, J.; Karakay, D.; Yilmaz, A. A Gis Aided Approach for Geolocalizing an Unmanned Aerial System Using Deep Learning. In Proceedings of the 2022 IEEE Sensors, Dallas, TX, USA, 30 October–2 November 2022; pp. 1–4.
2. Remondino, F.; Morelli, L.; Stathopoulou, E.; Elhashash, M.; Qin, R. Aerial triangulation with learning-based tie points. *Int. Arch. Photogramm. Remote. Sens. Spat. Inf. Sci.* **2022**, *43*, 77–84. [CrossRef]
3. Sarlin, P.E.; DeTone, D.; Malisiewicz, T.; Rabinovich, A. Superglue: Learning feature matching with graph neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 19–13 June 2020; pp. 4938–4947.
4. Ahmad, N.; Ghazilla, R.A.R.; Khairi, N.M.; Kasi, V. Reviews on various inertial measurement unit (IMU) sensor applications. *Int. J. Signal Process. Syst.* **2013**, *1*, 256–262. [CrossRef]
5. Davison, A.J.; Reid, I.D.; Molton, N.D.; Stasse, O. MonoSLAM: Real-time single camera SLAM. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 1052–1067. [CrossRef] [PubMed]
6. Qin, T.; Li, P.; Shen, S. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Trans. Robot.* **2018**, *34*, 1004–1020. [CrossRef]
7. Hu, S.; Feng, M.; Nguyen, R.M.; Lee, G.H. CVM-net: Cross-view matching network for image-based ground-to-aerial geo-localization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7258–7267.
8. Zhuang, J.; Dai, M.; Chen, X.; Zheng, E. A Faster and More Effective Cross-View Matching Method of UAV and Satellite Images for UAV Geolocalization. *Remote Sens.* **2021**, *13*, 3979. [CrossRef]
9. Macario Barros, A.; Michel, M.; Moline, Y.; Corre, G.; Carrel, F. A comprehensive survey of visual slam algorithms. *Robotics* **2022**, *11*, 24. [CrossRef]
10. Engel, J.; Schöps, T.; Cremers, D. LSD-SLAM: Large-scale direct monocular SLAM. In Proceedings of the Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014; Springer: Berlin/Heidelberg, Germany, 2014; Volume 13, pp. 834–849.

11. Mur-Artal, R.; Montiel, J.M.M.; Tardos, J.D. ORB-SLAM: A versatile and accurate monocular SLAM system. *IEEE Trans. Robot.* **2015**, *31*, 1147–1163. [CrossRef]
12. Mur-Artal, R.; Tardós, J.D. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Trans. Robot.* **2017**, *33*, 1255–1262. [CrossRef]
13. Campos, C.; Elvira, R.; Rodríguez, J.J.G.; Montiel, J.M.; Tardós, J.D. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Trans. Robot.* **2021**, *37*, 1874–1890. [CrossRef]
14. Shetty, A.; Gao, G.X. UAV pose estimation using cross-view geolocalization with satellite imagery. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 1827–1833.
15. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 187.
16. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16 $\times$ 16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929 .
17. Samet, H. The quadtree and related hierarchical data structures. *ACM Comput. Surv. (CSUR)* **1984**, *16*, 187–260. [CrossRef]
18. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G. ORB: An efficient alternative to SIFT or SURF. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–11 November 2011; pp. 2564–2571.
19. Hartley, R.; Zisserman, A. *Multiple View Geometry in Computer Vision*; Cambridge University Press: Cambridge, UK, 2003.
20. Chang, K.T. Geographic information system. In *International Encyclopedia of Geography: People, the Earth, Environment and Technology*; Wiley-Blackwell, Hoboken, NJ, USA, 2016; pp. 1–10.
21. OpenStreetMap Contributors. Planet Dump. 2017. Available online: https://www.openstreetmap.org (accessed on 9 August 2004).