Drone-View Building Identification by Cross-View Visual Learning and Relative Spatial Estimation

Chun-Wei Chen, Yin-Hsi Kuo, Tang Lee, Cheng-Han Lee, Winston Hsu
National Taiwan University, Taipei, Taiwan
jacky82226@gmail.com,kuonini@gmail.com,weitang114@gmail.com,r05922077@ntu.edu.tw,whsu@ntu.edu.tw

Abstract

Drones become popular recently and equip more sensors than traditional cameras, which bring emerging applications and research. To enable drone-based applications, providing related information (e.g., building) to understand the environment around the drone is essential. We frame this drone-view building identification as building retrieval problem: given a building (multimodal query) with its images, geolocation and drone's current location, we aim to retrieve the most likely proposal (building candidate) on a drone-view image. Despite few annotated drone-view images to date, there are many images of other views from the Web, like ground-level, street-view and aerial images. Thus, we propose a cross-view triplet neural network to learn visual similarity between drone-view and other views, further design relative spatial estimation of each proposal and the drone, and collect new drone-view datasets for the task. Our method outperforms triplet neural network by 0.12 mAP. (i.e., 22.9 to 35.0, +53% in a sub-dataset [LA])

1. Introduction

Recently, drones (known as Unmanned Aerial Vehicles [UAVs]) become more available for the masses of people and increasingly important in a wide range of security, surveillance and transportation applications. In 2017, as reported by Federal Aviation Administration, the number of registered drones is around 0.75 million in the U.S.[1], which enables a remarkably increase in emerging applications, like delivery drones, search and rescue (SAR) drones, and first-person view (FPV) drone racing (e.g., Amazon Prime Air, Matternet). Especially, Italdesign and Airbus unveil a new concept of Pop.Up (a ground-air vehicle) that also supports the need for drone-based visual understanding.

Compared with traditional vehicles (e.g., cars) and cameras, a wider area can be monitored by a drone, which means we can more easily utilize location-based properties as referring to the geographic map by drone's geolocation. It is vital to make a drone more deeply understand its environment for surveillance and manipulation. Thus, as illustrated in Figure 1, we design a **drone-view building identification** and utilize Computer Vision (CV) techniques to understand the environment automatically (e.g., image retrieval). However, image retrieval tasks are quite difficult in images with different views, like retrieval between street-view and aerial images [9, 26]. Fortunately, there are several sensors on drones, like camera, GPS tracker, compass and altimeter which we can utilize.

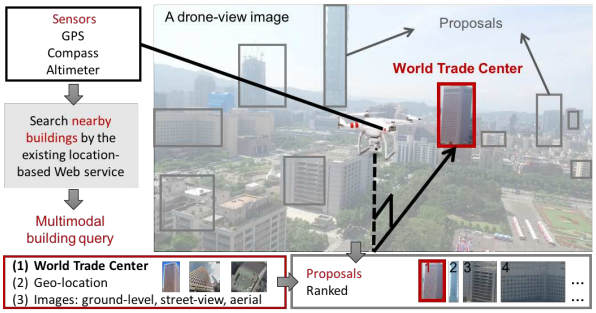

Figure 1. Drone-view building identification: given a drone-view image, we align nearby buildings with drone-view image proposals (building candidates). Each building is associated with its (1) name, (2) geolocation, and (3) images with different views from the Web. We take a building as the multimodal query to rank proposals (gray boxes) detected from the drone-view. The red box is ranked the highest by our method and tagged its name.

As shown in Figure 1, for the designed novel drone-view building identification task, we propose to leverage freely available resources from Web (e.g., Google Places API, Instagram, Dronestagram). Based on the location of drones, we collect all the nearby building (place) information and the corresponding images with different domains (e.g., street-view, aerial images). To associate them with drone-view images, these buildings are viewed as queries for retrieval. Due to the limitation of collecting data in big cities, we make an all-out effort to collect drone-view images in various locations and weather conditions. Since these annotated drone-view images are hard to get to date,

[1] http://twitter.com/FAANews/status/840273816072933376

Dataset	(1) Drone-BR (drone-view)	(2) Drone-BD (drone-view)	(3) IG-City8 (street-view)
Task	Building retrieval or identification	Building detection for proposals	Building matching in 8 cities
# Images	18 (1920x1080) + 62 (3840x2160)	18 (1920x1080) + 185 (1200x800)	4,409 (640x640)
# Data	585 queries (16,000 proposals)	2,334 bounding boxes	104,906 triplets (209,812 pairs)
Attribute	(a) Building queries from web (b) Matched bounding boxes (c) Drone's geolocation for images	Bounding boxes of buildings on drone-view images	(a) Matched building images (b) Date, check-ins, hashtags and geolocation for images

Table 1. Due to the lack of drone-view data, we collect 3 datasets for drone-view building identification. (1) Drone-BR consists of drone-view images and buildings (multimodal queries). (2) Drone-BD is used for acquiring proposals by training building detectors. (3) IG-City8 contains various building images on Instagram in 8 cities for cross-domain (transfer) learning.

this retrieval problem is a few-shot learning problem owing to the limited training data. Hence, we use distinct similarities for building identification due to the scalability to unknown environment. To conclude, we propose a drone-view retrieval system combined with different similarities by sensors on drones. Still, there are many difficulties in retrieval on drone-view images:

1. **Ultra-wide baseline matching.** Buildings which are far away from drones are low resolution and small size (e.g., 50x80). Besides, there are large distortions for building images between drone-view and other views, and many obstacles in front of buildings, like trees or other buildings.

2. **Uncertainty of sensors.** The sensors on drones have uncertainty and are not always precisely accurate in all conditions. GPS tracker and compass merely have a little noise, but altimeter is rather unstable with huge error (10-20m).

3. **Lack of drone-view data.** Since the drone is a new prevalent technical product, there are few annotated drone-view images to date. So far there is almost no drone-view data with sensor information.

To tackle difficulties above, here are our contributions:

- We collect and annotate new datasets for drone-view building detection and retrieval by our drones. Also, we gather a dataset of building images matching from Instagram for more diverse building images.
- We propose a novel drone-view building identification system with drones' various sensors and a cross-view drone-based triplet feature learning model to address this challenging cross-view retrieval problem between drone-view and other views.
- We further design relative spatial estimation for drone-view images, and combine with cross-view visual learning. Experiment results show that it is a challenging task (low baseline), but with our proposed method, we can achieve better performance than triplet neural network. (e.g., 24.07 to 34.97, +45% in LB)

2. Related work

Most of drone-based research focus on different kinds of applications, like real-time UAV path planning [17], tree counting [3], and UAV tracking [11, 21, 8]. Besides, many works focus on aerial imagery [1, 9, 26] instead of drone-view images, but it is fairly different from drone-view owing to inflexibility (fixed height and angle).

Content-based image retrieval is traditionally relied on handcrafted features, like SIFT [10] and HOG [5], but they are unable to handle ultra-wide baseline matching due to diverse views. Since convolutional neural networks (CNNs) have achieved state-of-the-art results in much CV research, prior works often use ImageNet-CNN features [7], Places205-CNN features [27], or further use a triplet learning to learn cross-view image features [24]. To perform better, we apply object proposal methods to drone-view images for acquiring building candidates. Popular methods include window-scoring-based (e.g., EdgeBoxes [28]), grouping-based (e.g., Selective Search [23], multiscale combinatorial grouping [12]) approaches and deep learning methods (e.g., Region Proposal Networks [RPN] [15]).

Wegner et al. [25] consider spatial information between a few street-view images to solve tree classification problem, but they have completely annotated tree dataset for robust evaluation. Wolff et al. [26] model spatial context and use traditional image features, like color and texture features [16] to solve the building facade matching problem between street-view and aerial images, while our problem focuses on drone-view to other views. Also, sensors including compass and GPS tracker are helpful for CV problems like image annotation by mobile sensors [4] and coronal plane estimation by UAV sensors [14].

3. Datasets

There are few labeled building images. Instead, we have tried to fine-tune on landmark datasets [2] which are relatively abundant. While it does not perform well since the appearance of landmark image are quite different from drone-view building images and some landmark images even do not contain buildings. We also try to find other aerial or street-view datasets [9]. Yet the authors cannot provide the images from Google API due to Google policy.

Consequently, we collect drone-view images with our drone (DJI 3) by designing and using an app for automat-

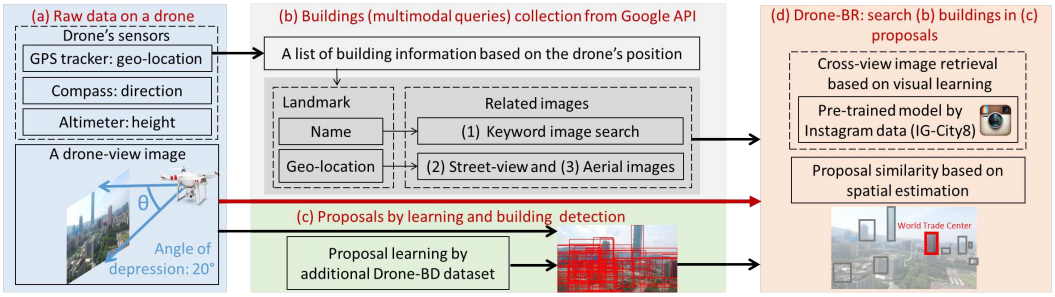

Figure 2. System diagram: (a) represents the raw data including drone-view images, drone's direction, geolocation and height from sensors. In (b), we collect these buildings' information based on the drone's geolocation and direction. For 3 cross-view images of building, we use Google text search to get ground-level images by the building name, Google Street View Image API to obtain street-view images by the geolocation, and Google Maps API to acquire aerial images by the geolocation. In (c), to identify buildings on drone-view images, we require to generate possible proposals by using object proposal methods (RPN) [15]. In (d), given a building (multimodal query) and proposals, we design cross-view visual learning and relative spatial estimation to compute similarity scores for our retrieval problem.

ically recording all drone sensors including GPS tracker, compass and altimeter while capturing images. We fix the angle of depression as 20°, raise the drone to 40m to 90m height and capture images with multiple buildings. Each image consists of 1-12 buildings as queries which have corresponding bounding boxes on drone-view images. We collect 80 images and annotate 108 buildings with bounding boxes on them at 3 locations under different weather. Besides, with Google Places API, we gather other information for these buildings including name, latitude and longitude.

As Figure 2 (b) depicts, we gather ground-level, street-view and aerial images for each building. Since the same buildings may appear in different images, there are 585 building queries in total presented in Table 1(1) which is **Drone-BR**. [2] Since buildings may be occluded or in low resolution, it requires great effort to annotate them with their corresponding high-resolution drone-view images (mostly 3840x2160). In our setting, we search each building query in 200 proposals on a drone-view image, which means the Drone-BR can be also described as 585 queries with 16,000 (200*80) sub-images for searching. Experiments in Sec. 7 show that our method can perform better with limited training data and it is scalable because the building identification for each drone-view image is performed independently.

On top of that, to the best of our knowledge, there are almost none drone-view images for building detection in any public datasets. In order to obtain building proposals with better quality, we also annotate 2,334 building bounding boxes on our 18 drone-view images and 185 images from Dronestagram website for training RPN [15]. This drone-view building detection dataset is called **Drone-BD**.

Due to the lack of drone-view data, we collect building images based on check-ins (locations) from Instagram and use the hashtag of #building and #buildings in 8 cities including New York, London, Paris, Hong Kong, Tokyo, Sydney, Berlin and San Francisco, to get building images. They are mostly ground-view or street-view images taken by users. We make buildings with the same location be a positive pair and with another random location be a negative pair to form a triplet. We manually remove some noisy images owing to not facing the correct building; eventually, there are 4,409 images, 848 buildings and 104,906 triplets in our Instagram building dataset called **IG-City8**.

[2]Datasets are available at https://jacky82226.github.io/DVBI

4. Building Identification System

Figure 2 provides an overview of our drone-view building identification system which is divided into four parts. For this task, because of the lack of drone-view data, few images of each building and too many various buildings to identify, it is a few-shot learning problem. Therefore, we frame it as building retrieval problem as follows and our retrieval model learns better features by similarity from building image matching instead of building classification. We are able to identify unknown buildings (unseen labels).

4.1. Problem Definition

Our drone-view building identification problem is defined as: given a nearby building B_i as query, we aim to retrieve a proper proposal P_i^* among all proposals. By referring to Google Places API based on drone's location for the top candidates, there are several buildings $B_1, B_2, ...B_n$ (multimodal queries) in a drone-view image and the number of them (n) is different in each image (i.e., we annotate 1-12 buildings per image). In addition, we utilize our building detector to acquire rough proposals $P_1, P_2, ...P_m$ (building candidates) and the number of them (m) depends on how many proposals we use in building detection (i.e., we use 200 proposals in our experiment). Thus, we can formulate this problem as maximizing our designed similarity (S):

$$P_i^* = \arg\max_j \ S(B_i, P_j), j = 1, 2, ...m. \quad (1)$$

4.2. Drone-View Building Detection

To retrieve buildings and further identify them on drone-view images, we have to create possible proposals (building candidates) first. We employ building detection to generate proposals on drone-view images. Since the ground truth (target buildings) may not actually exist in the detected proposals, the quality of building detection has the significant influence on identification results. To further improve the quality of proposals, we utilize RPN [15], a learning-based method. Accordingly, we train the drone-view building detector on Drone-BD and get proposals through it. Since it outperforms other object detection methods in our experiment, we utilize proposals from RPN as candidates (P_j).

4.3. Similarity Scores for Retrieval

We then compute similarity scores between the extracted proposals from the previous section and each building with visual learning and relative spatial estimation. Since each distance has different distributions without normalization, we convert distance into similarity by the inverse function. After trying borda count, linear normalization and sigmoid function, we find that sigmoid function is the best owing to more evenly representing the distribution. We aggregate them linearly to get an overall similarity score (S):

$$S(B_i, P_j) = \alpha \cdot \Psi(D_v) + \beta \cdot \Psi(D_a) + \gamma \cdot \Psi(D_x) + \delta \cdot \Psi(D_y), \quad (2)$$

where Ψ is a inverse-sigmoid function (inverse first then substitute to sigmoid function), $\Psi(x) = 1/(1 + e^{-1/x})$. We will introduce visual distance function (D_v) and relative spatial distance functions (D_a, D_x, D_y). $\alpha, \beta, \gamma, \delta$ are the weighted parameters of each similarity for linear combination according to the training data (alpha=0.4, beta=0.315, gamma=0.437, theta=0.437). These close weights indicate that our proposed methods are evenly essential.

5. Cross-View Visual Learning

To obtain the similarity scores between buildings (multi-modal query) and proposals, we frame this cross-view image retrieval as a cross-view matching problem. Given two images from different views, the model outputs a distance of this image pair. Thus, we propose (cross-view) drone-based triplet neural networks which make features in certain CNN layers close to positive pairs and far from negative pairs.

5.1. Transfer Learning: Pre-Trained on IG-City8

Based on the transfer learning, we propose leveraging external data (building images from other views) on Instagram to obtain a robust initial model. We train the initial CNN model with a triplet neural network on our collected IG-City8 to make our model focus on learning building features. We adopt the triplet loss [18] from a standard triplet network, and the triplets are formed with IG-City8. These user images taken under different weather also help our model to improve robustness in any illumination conditions. The results of different initial models show that matching between two ground-view building images helps to match between images from drone-view and other views. For better performance, we use a triplet neural network trained by IG-City8 as our initial model in the following sections.

5.2. Drone-based Triplet (DT)

We propose a drone-based triplet (DT) neural network for drone-view image matching. The architecture of DT illustrated in the first 3 CNN streams (anchor, positive and negative images) of Figure 3. For faster training and forward time, we use AlexNet [7] as our CNN streams and it can be changed to other networks (e.g., VGGNet [20]). We utilize a shared CNN architecture to learn cross-view features. Given features of anchor (a), positive (p) and negative (n) images, we attempt to make anchor-positive distances (d_{pos}) close to zero and anchor-negative distances (d_{neg}) far from a margin (m). This triplet loss is:

$$Loss(a, p, n) = max(0, d_{pos} - d_{neg} + m). \quad (3)$$

Empirically, we set the margin to 0.5 and use Euclidean distance as our distance function. For the input, we use single building image for a query as an anchor image, ground truth building image cropped from drone-view image as a positive image, and other proposals with Intersection-over Union (IoU) < 0.3 as negative images. After training, we can forward the DT and obtain features to compute the distance for the image similarity.

5.3. Cross-View Drone-based Triplet (CVDT)

Although we solely use an image as anchor image in our DT, images from other views contain different information to match drone-view images. To model all images simultaneously, we propose a cross-view drone-based triplet (CVDT) neural network. We use the same method as DT to sample positive and negative images from drone-view images (the 2nd and 3rd streams in Figure 3). The difference is CVDT uses 3 anchor images including ground-level, street-view and aerial images and goes through 3 CNN streams in anchor part. Hence, there are 5 CNN streams in CVDT with shared weights as shown in Figure 3(b).

To further integrate features with different views, we design a cross-view pooling (CVP) layer in Figure 4. The idea is that these convolutional or pooling features (e.g., pool5, before fc layers) contain spatial information that the upper part of ground-level images corresponds to aerial images (e.g., roofs) and the lower part corresponds to street-view images (e.g., windows, doors). Hence, to wisely fuse features from cross-view images, we split our ground-level

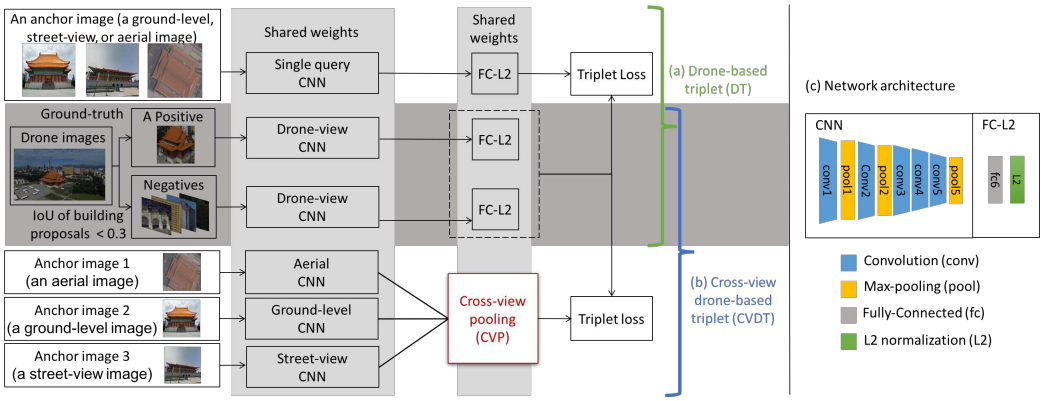

Figure 3. Drone-based triplet (DT) and cross-view drone-based triplet (CVDT): (a) The first 3 streams (green) are DT. An anchor image can be a ground-level, street-view, or aerial image from the same building. We then use our annotated ground truth on the drone-view image as positive image and sample negative images from proposals whose IoU < 0.3 for training. Accordingly, the triplet loss is computed by L2-normalized features. (b) The last 5 streams (blue) are our proposed CVDT. We use 3 cross-view images from the same building and design a cross-view pooling layer for learning cross-view visual features described in Figure 4. (c) With AlexNet, we utilize the same network architecture in our experiments and it can be changed to other networks (e.g., VGGNet or ResNet.)

image into the upper part and lower part and apply mean-pooling to the corresponding viewpoints. CVP enhances spatial features among cross-view images. The unmatched part in ground-level images is padding zero. After mean-pooling layer, the two features forward to a fc layer (shared weights), and eventually go through max-pooling to generate the final output features. It remains more shared information between 3 cross-view images by the combination of mean-pooling and max-pooling rather than merely applying max-pooling. [22] The same as DT, these features after max-pooling are normalized by Euclidean norm and compute triplet loss. For testing, CVDT goes through 3 query images, which are ground-level, street-view and aerial images, and attains a single feature to compute cross-view visual distance function shown below:

$$D_v = \|CNN(B_i) - CNN(P_j)\|_2^2, \quad (4)$$

where $CNN(B_i)$ and $CNN(P_j)$ are L2 normalized CVDT (or DT) features, and B_i and P_j are images of building from different views (or single image) and a proposal.

6. Relative Spatial Estimation for Proposals

Different from traditional image retrieval with GPS information, all the proposals share the same geolocation as the drone. It is hard to measure the spatial distance between buildings and proposals. Thus, we propose to estimate the distance by considering relative pixel positions on the drone image. The position of drone-view building can be more easily speculated than general ground-level imagery owing to the wide view (Bird's-eye view), and hence we can estimate spatial similarity for each proposal. Given a building (multimodal query), we measure the similarity between actual (geolocation) and relative estimated spatial relation.

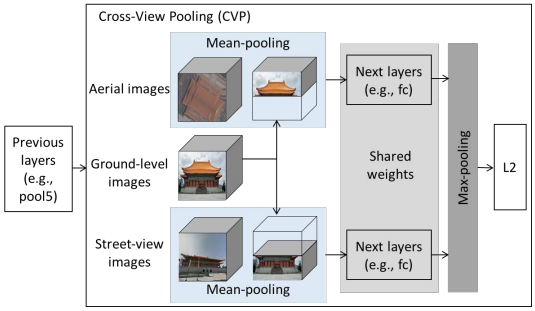

Figure 4. Cross-view pooling layer (CVP). We observe that ground-level images contain both aerial and street-view information. Based on the spatial information, we intend to integrate features (i.e., mean-pooling) in different views. We apply mean-pooling on aerial images with the upper part of ground-level images and street-view images with the lower part of ground-level images. Then these two features go through a fc layer (e.g., fc6) with shared weights and max-pooling to obtain the output features.

6.1. Drone-angle (D_a)

Without geolocation for proposals, we propose to estimate relative spatial relation for the drone and proposals based on their positions on the image. As shown in Figure 5(a), the drone's position is located in Pos_o (bottom center), and its heading is viewed as vertical direction (v_{om}). We can obtain the vector of certain proposal (v_{ob}) which is the vector from the drone's position (Pos_o) to a proposal (Pos_b). In addition, based on the collected geolocation, and drone's direction, we can compute angles between the heading vector (v_{df}) and building vectors (v_{db}). These vectors are shown in Figure 5(b). v_{db} is the vector from the drone's geolocation (GPS_d) to the building's geolocation (GPS_b). Considering the angle between drones and targets, we de-

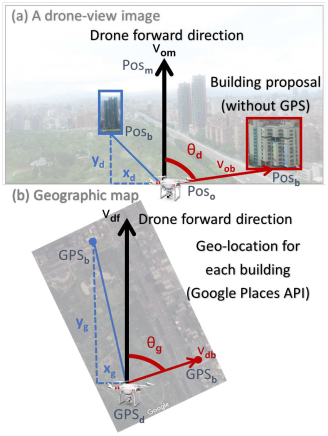

Figure 5. Relative spatial estimation for each proposal. Due to not knowing building proposals' geolocation on drone-view images, we propose estimate their distance and angle. (a) shows how to compute θ_d, x_d, y_d with each proposal. (b) reveals how to obtain θ_g, x_g, y_g by geolocation from Google Places API. Thus, we can compute their similarity scores between our estimation and the geolocation information (i.e., θ_d vs. θ_g, x_d vs. x_g and y_d vs. y_g).

sign the Drone-angle (D_a) to gain similarity between map angle and drone-view angle. The Drone-angle (D_a) is:

$$D_a = |\theta_d - \theta_g| / 180, \tag{5}$$

where $\theta_d = angle(v_{ob}, v_{om})$ is the estimated angle for a proposal, and $\theta_g = angle(v_{db}, v_{df})$ is the angle on geographic map. $angle(v_1, v_2) = \arccos((v_1 \cdot v_2)/(\|v_1\| \cdot \|v_2\|))$ returns the angle between two vectors.

6.2. Drone-distance (D_x and D_y)

Similar to D_a, we compute actual (geolocation) and estimated geographical distance for Drone-x-distance (D_x) and Drone-y-distance (D_y). For the estimated distance of a proposal, as shown in Figure 5(a), the estimated horizontal distance (x_d) and vertical distance (y_d) are calculated from $Pos_b : ((x_1 + x_2)/2, y_2)$ and $Pos_o : (w_I, h_I)$. We define $x_d = |(x_1 + x_2)/2 - w_I| / w_I$ and $y_d = |y_2 - h_I| / h_I$, where $\{x_1, y_1, x_2, y_2\}$ are the bounding box coordinates for a proposal Pos_b. While capturing drone-view images with different heights, the geographical distance (x_g, y_g) is always the same but the estimated distance (x_d, y_d) varies from camera settings (focal length and CCD width). Hence, we transform the estimated distance with t_x and t_y. Note that the vertical distance is additionally influenced by the drone's height. To mitigate the transformation error, we estimate t_x and t_y, respectively. Based on the drone's height (h_d) and settings including CCD width (c_d) and focal length (f_d), $t_x = c_d/(100 \cdot f_d)$ and $t_y = h_d/70$. The final proposed Drone-distance functions (D_x and D_y) are defined as:

$$D_x = |x_g - t_x \cdot x_d|, D_y = |y_g - t_y \cdot y_d|. \tag{6}$$

Finally, as mentioned in Sec. 4.2, we convert 3 distance functions (D_a, D_x and D_y) into similarity scores by the inverse function and then substitute them to sigmoid function.

7. Experiments

To obtain proposals for drone-view images, we apply the state-of-the-art methods like selective search, edgebox and RPN. We observe that RPN outperforms others. To further enhance the quality of proposals, we fine-tune RPN on Drone-DB. Due to the limited training data, we leave the improvement of proposals' quality in future work and focus on cross-view visual learning and relative spatial estimation. We experiment our proposed models on Drone-BR (80 drone-view images and 585 queries). For evaluation, we rank proposals for each query independently. If a retrieved bounding box's IoU overlaps with a ground truth bounding box $\geq$ threshold, then it hits. We evaluate the performance by mean Average Precision (mAP) and choose 0.3 as IoU threshold owing to the limitation of proposal quality. There are 56 queries which absolutely cannot retrieve the correct answer (cannot find IoU ≥ 0.3) in the proposals. It also increases the difficulties of image matching, since the ground truth proposals may only contain partial building features. The upper bound of the experiment is 0.923 (540/585). After experimenting on different number of proposals (i.e., 50-500), we find 200 proposals are good enough because the performance remains constant even if we use more proposals. For fixed database and fair comparison, we compare different models on the same proposal number (200) and only search the query in the same drone-view image.

Query Images	Aerial	Street-view	**Ground-level**
ImageNet-AlexNet	5.32	7.43	15.87
Places205-AlexNet	6.98	9.27	20.55
IG-City8-Triplet	7.38	13.06	**21.61**

Table 2. Effect of initial models. mAP (%) of various initial models on Drone-BR with 3 different types of 585 queries. It is critical to utilize related dataset (IG-City8) for getting better performance. Ground-level image is the most useful due to containing more visual information than other views.

7.1. Transfer Learning: The Effect of Initial Models

Research [19] suggests that CNNs outperform handcrafted features like SIFT. Thus, we use off-the-shelf deep features as baselines (i.e., fc6 in ImageNet [ImageNet-AlexNet] [7] and Places205 [Places205-AlexNet] [27]). In Table 2, existing pre-trained models do not perform well since they do not train on enough building images in general datasets. It also shows the difficulty in drone-view building identification due to the lack of drone-view datasets. Hence, leveraging external data and few annotated images for training is necessary. For ground-level images,

Figure 6. Drone-BR: we gather ground-level, street-view and aerial images of 108 building images for multimodal queries (buildings). Since we collect drone-view images at three locations under different weather, our Drone-BR can be split into five subsets according to diverse conditions. Hence, due to the variety of the collected cross-view and drone-view images, it is a challenging problem for our designed building identification. More data, some retrieval results and a video result are revealed in Supplementary. [Best viewed in color.]

Places205 (20.55%) is better than ImageNet (15.87%) because of seeing more location data which is relevant to buildings. We fine-tune the network with parameters from Places205 and train more building triplets with IG-City8-Triplet. The ground-level image achieves the best accuracy among all models. Besides, IG-City8-Triplet also improves significantly for street-view images (+41% over Places205) because photos taken by users in IG-City8 are also close to street-view images (look up from the ground). In summary, while testing 3 different views respectively, initial model trained by IG-City8 outperforms pre-trained deep features (+5% over Places205 in ground-level). This is because we train on a relevant domain (building images) from external IG-City8 dataset. We use this model as initial model for DT and CVDT to achieve better performance and robustness.

7.2. Cross-View Visual Learning

Compared with drone-view images, street-view images merely contain the facades of buildings, whereas aerial images simply contain the roofs of buildings. Thus, we use the ground-level image as the anchor image for DT to compare with other methods. Drone-BR can be categorized into 5 subsets according to its captured location and weather. To evaluate our method under diverse conditions, we respectively test on a subset by training on other subsets. Since early stopping is critical for limited training data, we adopt it and observe that the best performance usually appears in the first 5 epochs with learning rate=0.00001 and batch size=128. Our model outperforms pre-trained deep features, which shows the improvement for both unseen buildings (e.g., LC) and different weather (e.g., WB).

Siamese and triplet network are 2 state-of-the-art methods for cross-domain image matching problems [13, 18, 6, 9, 24]. As a result, we use triplet neural network (DT) as a strong baseline for our drone-view building identification. After applying DT on the initial model (IG-City8-Triplet), we can achieve further improvement owing to training with more drone-view building images. Since the query and database are in the different domains (ground-level and drone-view), the proposed DT can mitigate the gap between them. Besides, by considering cross-view visual learning (CVDT), we can achieve the best accuracy (+21% in LA) due to leveraging images in different views in the learning process. Compared with unseen buildings, different weather can achieve larger performance gains. Nevertheless, testing on unseen buildings is still better than IG-City8-Triplet (+31% in LB) owning to learning common building features to identify unseen buildings. If our models are trained with more training data, gains are significantly improved (+105% in WB). Moreover, Location B is the most challenging subset since we collect it under the extreme weather (i.e., almost rainy and very sunny days), which makes buildings blur and backlight. Still, CVDT+CVP can have 31% relative gains.

mAP (%)	LA	LB	LC	WA	WB
# Training	362	355	453	149	436
# Testing	223	230	132	436	149
ImageNet [7]	15.00	17.70	14.14	15.09	16.15
Places205 [27]	19.56	19.33	24.36	17.96	21.48
IG-City8	20.42	20.43	25.69	18.56	22.70
Baseline (DT)	22.89	24.07	26.72	32.19	43.79
CVDT (Max)	25.14	24.97	25.02	33.04	44.30
CVDT (CVP)	**27.75**	**26.81**	**28.87**	**34.47**	**46.45**
CVDT+Spatial	**35.00**	**34.97**	**35.70**	**42.60**	**50.15**

Table 3. Cross-view visual learning. mAP of different models on each subset. LA means that DT and CVDT are trained on Location B and Location C, and tested on Location A (i.e., buildings in Location A are unseen). WA and WB have seen all the buildings but trained and tested under different weather. # means the number of images. The results indicates our proposed CVDT+CVP outperforms others on all subsets. (IG-City8 follows the same training process as [18] and forms triplet by IG-City8. CVDT (CVP)+Spatial is the method combines with $D_a + Dx + Dy$.)

We further conduct experiments on two pooling methods with our CVDT. One is max-pooling [22] (i.e., max-pooling on different views), and the other is our proposed CVP. In Table 3, CVDT+CVP is better than CVDT+Max (Max-pooling) because max-pooling is easily influenced by noise. Aerial and street-view images are greatly differ-

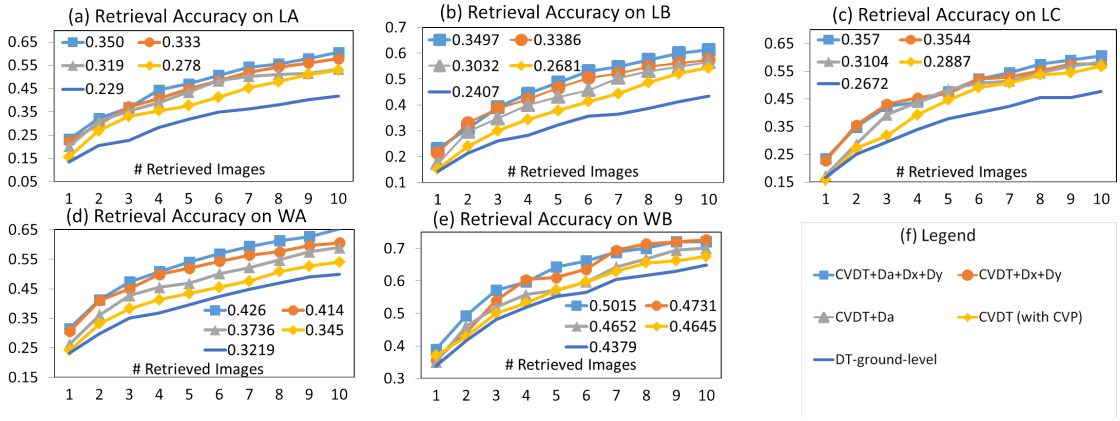

Figure 7. Retrieval with relative spatial estimation: The retrieval accuracy of line charts is drawn by different top-ranked images on five various subsets among different models. Each model is trained by other subsets. The value after line is mAP. The results show that our proposed method (CVDT+D_a+D_x+D_y) can achieve the best performance under various settings.

ent from drone-view and ground-level images. Since we use mean-pooling layer on 3 views in the CVP, the performance of CVDT is at least similar to DT with ground-level images in the worse case (e.g., DT-ground-view in Figure 7). However, we can obtain useful information from aerial and street-view images by our CVP. In conclusion, our CVDT+CVP outperforms other learning methods.

Methods	mAP (%) on 585 queries
Drone-angle (D_a)	8.91
Drone-distance (D_x, D_y)	12.29
Drone-angle + Drone-distance	16.23
IG-City8-Triplet (visual)	21.61

Table 4. The performance of relative spatial estimation is worse than visual features. Since there are multiple candidates with similar angle and distance, it is hard to distinguish them based on spatial estimation, which motivates us to leverage visual features.

7.3. Retrieval with Relative Spatial Estimation

Table 4 shows that simply considering relative spatial estimation performs worse than visual features. To leverage the sensor data, we integrate cross-view visual learning with relative spatial estimation. In the following experiments, we use CVDT+CVP in Figure 7 as visual features since it is the most powerful. The results on Drone-BR in 5 subsets are revealed in Figure 7. Drone-angle (+D_a) improves the retrieval accuracy (0.28 to 0.32 in LA) owing to filter out proposals which have large angle gaps. Drone-distance (+D_x+D_y) also has the better retrieval accuracy (0.28 to 0.33 in LA). It performs better than D_a since D_a has more serious projection errors from geographic map to drone-view. Finally, the model (CVDT+D_a+D_x+D_y) performs the best (0.35 in LA) on 5 diverse testing subsets. Our relative spatial estimation improves more in lower performance (0.27 to 0.35 in LB over 0.47 to 0.50 in WB). Thus, it is more competitive in challenging drone-view data, like unseen buildings or extreme weather.

On different ranking levels, our best model outperforms other methods stably. The accuracy@1 (only if the 1st retrieved result hits) is the most vital for real-world applications. Yet our best model achieves 0.23 in LB and 0.39 in WB which means this problem is still challenging. Besides, if we inspect accuracy@10, our best model performs well by achieving 0.61 in LB and 0.72 in WB which indicates retrieving more images improves more remarkably in harder group (+166% in LB). To conclude, our method achieves good retrieval accuracy in all conditions of limited dataset.

8. Conclusions

We collect drone-view datasets and design a brand-new and challenging drone-view building identification problem, which helps drones understand their environment more deeply. Due to the lack of annotated drone-view data, we utilize external building dataset for cross-view image matching and propose a cross-view visual learning model for drone-view image matching. Besides, for measuring the spatial relation between proposals (without geolocation) and the drone, we further integrate relative spatial estimation to improve retrieval performance. Experiments show that our methods (CVDT+D_a+D_x+D_y) significantly improve the retrieval accuracy over triplet neural network (e.g., 22.9 to 35.0, +53% in LA).

9. Acknowledgement

This work was supported in part by MediaTek Inc and the Ministry of Science and Technology, Taiwan, under Grant MOST 107-2634-F-002-007. We also benefit from the grants from NVIDIA and the NVIDIA DGX-1 AI Supercomputer.

References

[1] H. Altwaijry, E. Trulls, J. Hays, P. Fua, and S. Belongie. Learning to match aerial images with deep attentive architectures. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, number EPFL-CONF-217963, 2016.

[2] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. In *European conference on computer vision*, pages 584–599. Springer, 2014.

[3] Y. Bazi, S. Malek, N. Alajlan, and H. AlHichri. An automatic approach for palm tree counting in uav images. In *2014 IEEE Geoscience and Remote Sensing Symposium*, pages 537–540. IEEE, 2014.

[4] A.-J. Cheng, F.-E. Lin, Y.-H. Kuo, and W. H. Hsu. Gps, compass, or camera?: investigating effective mobile sensors for automatic search-based image annotation. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 815–818. ACM, 2010.

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.

[6] A. Gordo, J. Almazán, J. Revaud, and D. Larlus. Deep image retrieval: Learning global representations for image search. In *European Conference on Computer Vision*, pages 241–257. Springer, 2016.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[8] S. Li and D.-Y. Yeung. Visual object tracking for unmanned aerial vehicles: A benchmark and new motion models. 2017.

[9] T.-Y. Lin, Y. Cui, S. Belongie, and J. Hays. Learning deep representations for ground-to-aerial geolocalization. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5007–5015. IEEE, 2015.

[10] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[11] M. Mueller, N. Smith, and B. Ghanem. A benchmark and simulator for uav tracking. In *European Conference on Computer Vision*, pages 445–461. Springer, 2016.

[12] J. Pont-Tuset, P. Arbelaez, J. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping for image segmentation and object proposal generation. 2015.

[13] Y. Qi, Y.-Z. Song, H. Zhang, and J. Liu. Sketch-based image retrieval via siamese convolutional neural network. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 2460–2464. IEEE, 2016.

[14] A. M. Rahimi, R. Ruschel, and B. Manjunath. Uav sensor fusion with latent-dynamic conditional random fields in coronal plane estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4527–4534, 2016.

[15] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[16] F. Riaz, A. Hassan, S. Rehman, and U. Qamar. Texture classification using rotation-and scale-invariant gabor texture features. *IEEE Signal Processing Letters*, 20(6):607–610, 2013.

[17] V. Roberge, M. Tarbouchi, and G. Labonté. Comparison of parallel genetic algorithm and particle swarm optimization for real-time uav path planning. *IEEE Transactions on Industrial Informatics*, 9(1):132–141, 2013.

[18] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.

[19] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 118–126, 2015.

[20] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[21] J. Škoda and R. Barták. Camera-based localization and stabilization of a flying drone. 2015.

[22] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015.

[23] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[24] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu. Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1386–1393, 2014.

[25] J. D. Wegner, S. Branson, D. Hall, K. Schindler, and P. Perona. Cataloging public objects using aerial and street-level images-urban trees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6014–6023, 2016.

[26] M. Wolff, R. T. Collins, and Y. Liu. Regularity-driven facade matching between aerial and street views. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[27] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, pages 487–495, 2014.

[28] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *European Conference on Computer Vision*, pages 391–405. Springer, 2014.