Superpixel-based Convolutional Neural Network for Georeferencing the Drone Images

Superpixel-based Convolutional Neural Network for Georeferencing the Drone Images

Shihang Feng, Luca Passone and Gerard T. Schuster

***Abstract*—Information extracted from aerial photographs has been used for many practical applications such as urban planning, forest management, disaster relief, and climate modeling. In many cases labeling of information in the photo is still performed by human experts, making the process slow, costly, and error-prone. This paper shows how a convolutional neural network can be used to determine the location of GCPs in aerial photos, which significantly reduces the amount of human labor in identifying GCP locations. Two CNN methods, sliding-window CNN with superpixel-level majority voting and superpixel-based CNN, are evaluated and analyzed. The results of the classification and segmentation show that both of these methods can quickly extract the locations of objects from aerial photographs, but only superpixel-based CNN can unambiguously locate the GCPs.**

***Index Terms*—Superpixel, convolutional neural network (CNN), georeferencing**

I. Introduction

AERIAL image interpretation finds applications in many diverse areas including urban planning, forest management, vegetation monitoring, and climate modeling [1, 2]. Standard GPS accuracy used to geotag image is usually in the order of few meters. This error causes a systematic shifting in the processed outputs far beyond the acceptable bound, which typically ranges between 1 to 5 cm. To improve the accuracy of the results, Ground Control Points (GCPs) are evenly distributed across the area of interest and measured using high accuracy methods such as Real Time Kinematic (RTK) GPS. A fundamental step in aerial image georeferencing consists of marking the location of GCPs on the ground in the aerial photos as shown in Figure 1. However, much of the work in identifying the locations of these GCPs in the photographs is still performed by human experts, which can be labor intensive with the explosive growth in data volumes.

Recently, several deep learning algorithms have emerged to automate the process of information retrieval from aerial images. These algorithms have been used successfully for object detection [3, 4, 5] and scene classification [6]. The impressive accuracy produced by these algorithms suggest that automated aerial image interpretation systems may be within reach [7, 8].

A convolution neural network (CNN) is a special kind of multi-layer neural network developed to classify high-dimensional patterns [9, 10]. It does not require feature extraction, thus resulting in higher generalization capabilities.

Shihang Feng and Gerard T. Schuster are with King Abdullah University of Science and Technology, Thuwal, 23955, Saudi Arabia (e-mail: shihang.feng@kaust.edu.sa, gerard.schuster@kaust.edu.sa),

Luca Passone is with FalconViz, Thuwal, 23955, Saudi Arabia (e-mail: luca.passone@falconviz.com).

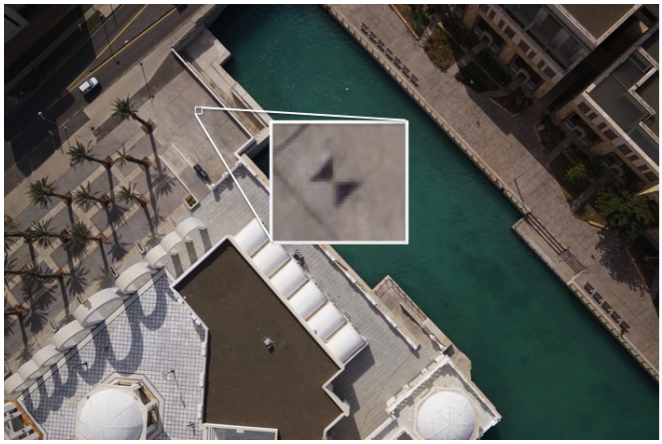


Fig. 1: A example of the 0.5 $\times$ 0.5 m^2 GCP on the ground. GCP has a unique hourglass shape and its color is dark

To improve the performance and accuracy of CNN, different structures have been developed. AlexNet has two parallel CNN lines trained on two GPUs with cross-connections. It has been shown that this replaces the sigmoid activation with a rectified linear unit (ReLu) activation [4]. Other architectures such as GoogLeNet introduces inception modules and increases the depth of the network [11]. To avoid zero gradients, ResNet presents a residual learning framework to ease the training of networks that are substantially deeper than those used previously [12].

The goal of the CNN method in this paper is to detect the locations of GCPs in the drone photos. These GCPs are only about $0.5 \times 0.5\ m^2$ in size, and are difficult to detect manually. The drone takes thousands of images per flight, so it takes days of manual labor to locate the GCPs in the photographs. These GCPs georeference the model at cm level accuracy, allowing geomatics engineer to take accurate measurements of roads, buildings and any other visible features. If these GCPs can be automatically detected and labeled then this would result in a significant savings in human labor. The aerial image interpretation is usually formulated as a pixel labeling problem. Given an aerial image, the pixel labeling approaches classify each pixel to get a binary classification of the image for a single object class [13, 14] or a multi-class classification to get a complete semantic segmentation of the image into classes such as roads, grasses, and cars [15]. In our paper, both cases are studied for detecting the objects in the aerial image.

The pixel labeling classification method alone is known to produce salt-and-pepper errors since it only includes sparse

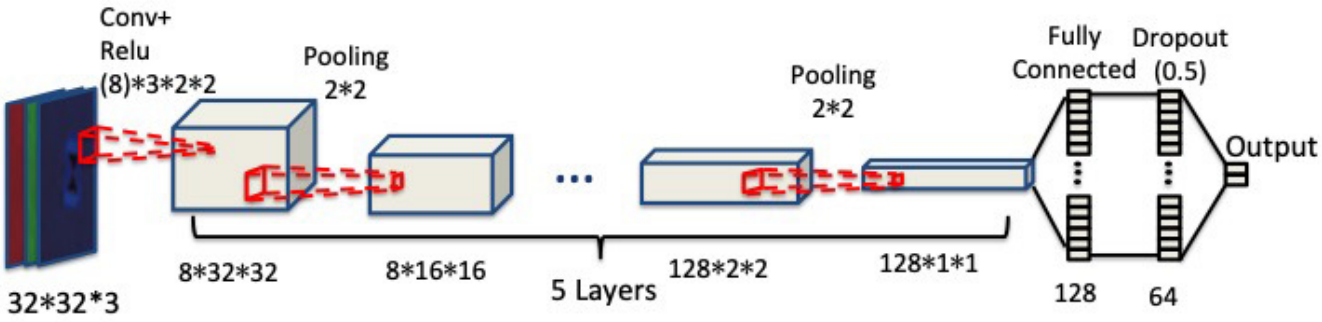


Fig. 2: Illustration of the binary CNN architecture with five convolution layers and two fully-connected layers.

spatial information within the neighborhood of the pixel in the classification [2]. To overcome this problem, we propose sliding-window CNN with superpixel-level majority voting, which apply a simple linear iterative clustering (SLIC) and the density-based spatial clustering of applications with noise (DBSCAN) for image segmentation after classification.

The other method we propose is superpixel-based CNN. The GCP has a unique hourglass shape and its color is dark (see Figure 1 as an example). To take advantage of these features, a superpixel-based classification method is applied to the aerial images. Compared with the pixel labeling classification, it can locate the GCP accurately. However, it doesn't perform well in classifying other objects.

This paper is divided into four sections. After the introduction, the second section presents the theory of image classification and segmentation. The CNN architecture and the SLIC method are described. In the third section, the CNN methods used in the classification are introduced. The fourth section presents numerical test results in classifying objects in aerial images. The images are taken by a drone flying at an average elevation of 80 m and equipped with a high-resolution camera. The conclusions are given in the last section.

II. Theory

A. CNN Architecture

The aerial image classification problem is an image classification task, which classifies objects in image patches that are used as input data. An image patch which includes a GCP is labeled as a positive patch. Otherwise, it is labeled as a negative patch. We extract image patches containing a GCP with dimension 65×65 and downsample them into 32×32 patches.

The network takes these 32×32 patches as input and the classification label of the center point is the output. The network (see Figure 2) is composed of five convolution (Conv) layers and two fully-connected (FC) layers followed by a softmax classifier. There are eight convolution filters with size 2×2 in the first Conv layer. The number of channels is doubled in the latter convolution layers. Rectified linear activation (ReLu) and Max-pooling with size 2×2 are applied after each Conv layer. There are 128 feature maps in the first fully-connected (FC) layer and we apply dropout to 50% of them in the second FC layer. The softmax classifier gives the probability of the center point being a reflection event or not. The probability of two classifiers are compared and the one with higher probability is selected as the prediction label.

All convolutional filter kernel elements are trained from the data in a supervised fashion by learning from the labeled set of training examples. A training set $N = \{x^i, y^i\}$ is given which contains n image patches, where x^i is the i-th image patch and $y^i \in \{0, 1\}$ is the corresponding class label. Then the corresponding cross-entropy cost function is given by

$$L = -\frac{1}{n}\sum_{i}^{n} \log p(Y = y^i|x^i), \tag{1}$$

where $p(Y = y^i|x^i)$ is the probability that the label of x^i is y^i.

After training, the network is tested with a validation set to check the success of training. If the network is trained properly, it will recognize and correctly classify only those cases included in the training set. For new conditions not included in the training set, they will be misclassified or not recognized.

B. Simple linear iterative clustering (SLIC)

A simple linear iterative clustering (SLIC) algorithm is used to reduce the classification noise within segments. To expedite the classification of GCPs we define a superpixel as a group of connected pixels with similar colors and gray levels. SLIC adapts a k-means clustering approach to efficiently generate superpixels in the CIELAB color space [16].

CIELAB is a three-dimensional integer space $(L^\star, a^\star, b^\star)$ for digital representation of colors shown in Figure 3, where $L^\star$ represents the lightness component. The lightness value $L^\star$ ranges from 0 to 100, where the darkest black at $L^\star = 0$ and the brightest white at $L^\star = 100$. $a^\star$ represents the green-red component, with green in the negative direction and red in the positive direction. $b^\star$ represents the blue-yellow color component, with blue in the negative direction and yellow in the positive direction. The $a^\star$ and $b^\star$ axes both range from -128 to 127. The clustering procedure of SLIC requires the following steps shown in Figure 4:

1. In the first step, the input image is divided into K approximately equally-sized superpixels. For an input image

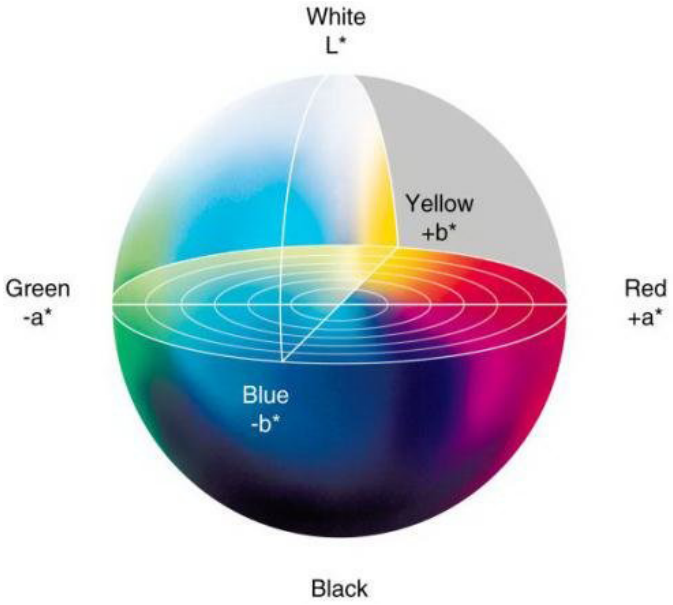


Fig. 3: CIELab Color Space (adapted from Tang et al, 2015 [17]). L^* for the lightness from black to white, a^* from green to red, and b^* from blue to yellow

with N pixels, the approximate size of each superpixel is N/K. Hence, there is a superpixel center at every grid interval $S = \sqrt{N/K}$ as shown in Figure 4a. The coordinates for the initial superpixel centers are $[L_k, a_k, b_k, x_k, y_k]^T$ with $k = \{1, 2, ..., K\}$ at regular grid intervals S.

2. If the initial center of a superpixel is placed on an edge or a noisy pixel, no pixels in the neighborhood belong to the same cluster with the center. To prevent the situation, the magnitude of the image gradient is computed as:

$$G_{x,y} = ||\mathbf{I}_{x+1,y} - \mathbf{I}_{x-1,y}||^2 + ||\mathbf{I}_{x,y+1} - \mathbf{I}_{x,y-1}||^2, \quad (2)$$

where $\mathbf{I}_{x,y}$ is the CIELAB vector $[L_{x,y}, a_{x,y}, b_{x,y}, x, y]^T$ corresponding to the pixel at position (x,y), and $||.||$ is the L_2 norm. The calculation of the gradients takes into account both color and intensity information. If the center is located on the edge or a noisy pixel, the magnitude of the image gradient will be large. To move the center point away from the edge and the noise pixel, the magnitudes in the 3×3 neighborhood around the initial superpixel centers are calculated as shown in Figure 4b. Then the centers are moved to the locations corresponding to the points with the lowest magnitude.

3. The search region for each superpixel center has the area $2S \times 2S$ shown in Figure 4c, where S is the grid interval. Each pixel i would be in the search regions of its surrounding superpixel centers and it is associated with the nearest one. The dark green area in Figure 4c is composed of the pixels associated with the yellow superpixel center. Compared with the conventional k-means clustering that compares each pixel with all the superpixel centers, this method greatly speeds up the algorithm by limiting the size of the search region.

The distance measure D_s between the pixel and superpixel center is defined as follows:

$$\begin{aligned} d_{lab} &= \sqrt{(L_k - L_i)^2 + (a_k - a_i)^2 + (b_k - b_i)^2}, \\ d_{xy} &= \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2}, \\ D_s &= d_{lab} + \frac{m}{S} d_{xy}, \end{aligned} \quad (3)$$

where D_s is the sum of the CIELAB distance d_{lab} and the xy plane distance is normalized by the grid interval S. A variable m is introduced for controlling the compactness of a superpixel.

4. Once each pixel has been associated the nearest superpixel center, an update step adjusts the superpixel centers to be the mean $[L, a, b, x, y]^T$ vector of all the pixels belonging to the superpixel shown in Figure 4d. The L_1 norm is used to compute a residual E corresponding to the new superpixel center locations $[L_{k_{new}}, a_{k_{new}}, b_{k_{new}}, x_{k_{new}}, y_{k_{new}}]^T$ and previous superpixel center locations $[L_k, a_k, b_k, x_k, y_k]^T$:

$$\begin{aligned} E &= \sum_k (|L_k - L_{k_{new}}| + |a_k - a_{k_{new}}| + |b_k - b_{k_{new}}|) \\ &+ \sum_k (|x_k - x_{k_{new}}| + |y_k - y_{k_{new}}|). \end{aligned} \quad (4)$$

5. Repeat steps 3 and 4 until the residual E is less than the specified threshold.

III. CNN Methods

Two approaches are used to solve the georeferencing problem. In the first method, SLIC is used as post-processing with majority voting on CNN labels. In the second method, SLIC is used as pre-processing to extract the superpixels from the images for CNN classification.

A. Sliding-window CNN with Superpixel-level Majority Voting

The sliding-window CNN is a traditional neural network for solving certain classification problems [18]. It classifies an image by taking a pixel and a border padding of some size around it as input and classifies the center pixel by running the patch through a neural network. Then the next pixel is labeled by shifting the window patch by one pixel, and classifying the neighboring pixel of the first one.

The main disadvantage of this approach is the impractical runtime. A 3500×5000 pixel^2 aerial image requires 1.75×10^7 classifications, which may cost hours for the computation. In order to reduce the computational time, a classification stride value of s is used. A stride of s pixels results in the center pixels having a distance of s to the next patch's center pixel. This increases the computation speed by a factor of s^2. However, it will reduce the resolution of the classification results and amplify some salt-and-pepper errors.

To mitigate the salt-and-pepper problem, a region merging process is performed as a post-processing method by segmenting the aerial image into superpixels using the SLIC algorithm. These superpixels are then processed using the density-based spatial clustering of appications with noise (DBSCAN) algorithm to form clusters of superpixels to generate the final segmentation shown in Figure 5. The details are

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3065398, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

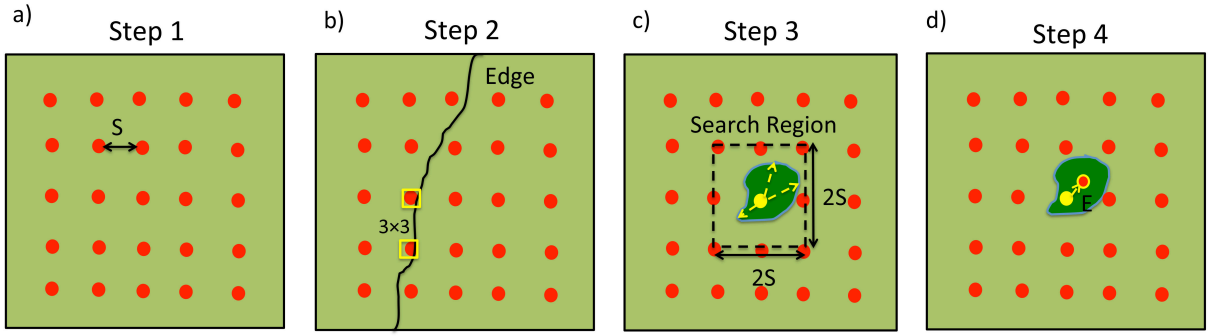


Fig. 4: The workflow of SLIC. Step 1: initialize superpixel center with grid interval S. Step 2: move the centers away from edge and noise. Step 3: cluster the pixels with search region $2S \times 2S$. Step 4: Update the location of centers.

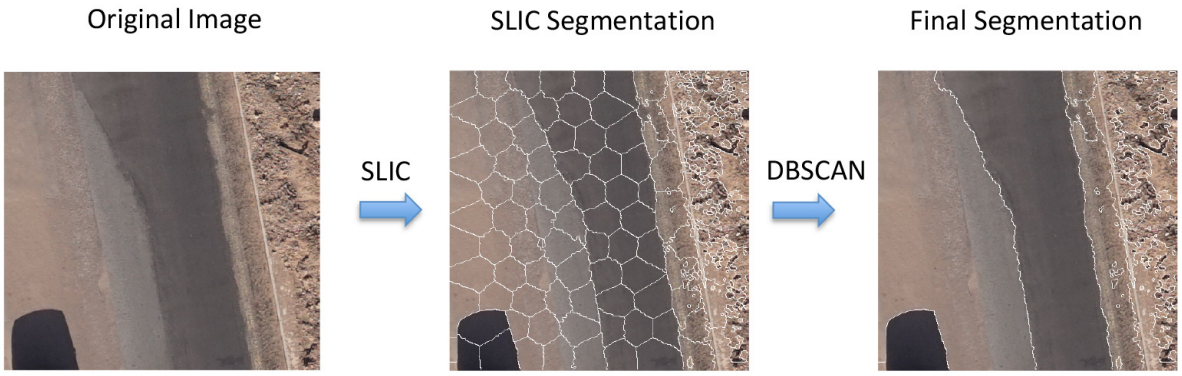


Fig. 5: The image segmentation using SLIC and DBSCAN. The image is firstly segmented by SLIC then the neighboring superpixels with similar color are merged into one superpixel by DBSCAN

Fig. 6: a) The superpixel, b) CNN classification result in a superpixel and c) classification result after majority voting. Different colors represent the label of pixels.

shown in Appendix A. The CNN labels belonging to the same superpixels should have the same labels, so the label of each superpixel is decided by majority voting of the pixel labels that belong to it as shown in Figure 6. For every superpixel, the number of pixels belong to each class is counted. The class that has the highest number would be the dominate class in this superpixel. Then the labels of all the pixels within this superpixel will be changed to the dominate class.

B. Superpixel-based CNN

The GCP has a unique hourglass shape and its color is dark. To take advantage of these features in the classification, a superpixel-based method is proposed. The workflow of the superpixel-based CNN is as below:

1. The input image is firstly partitioned into superpixels using SLIC as shown in Figure 7. The initial size of the superpixel should be larger than the GCP so that the pixels of the GCP belong to the same superpixel.

2. The superpixels are interpolated to be the same size as

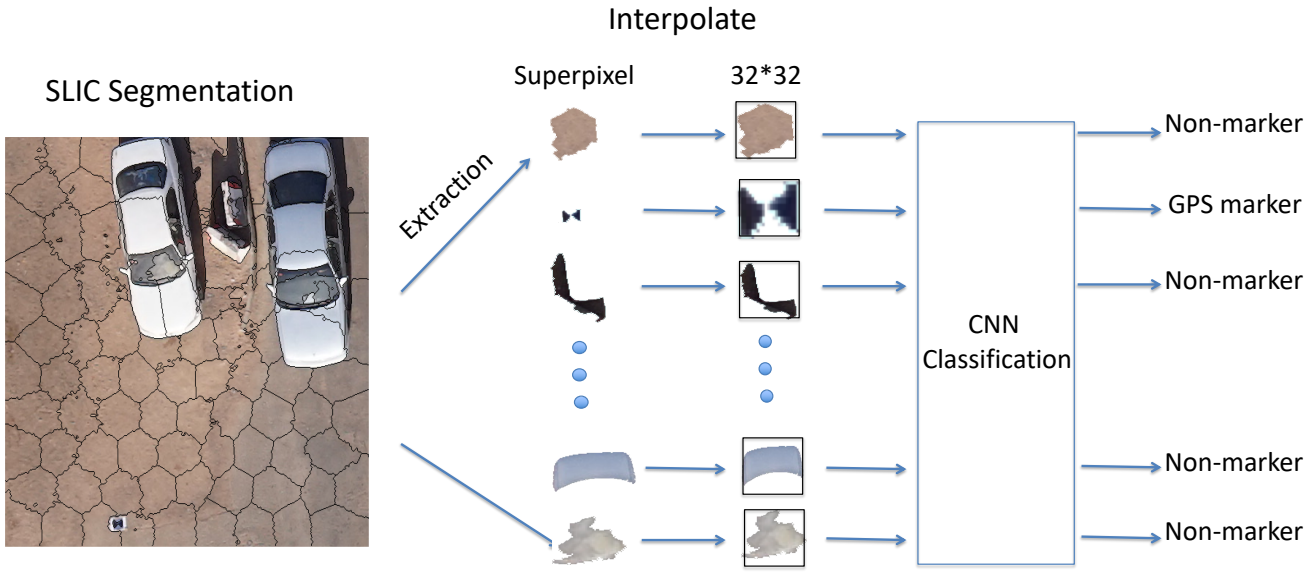


Fig. 7: The workflow of superpixel-based CNN. The superpixels are extracted from the images using SLIC and the superpixels are interpolated to be the same size for CNN classification. Then the labels of the superpixels are given by the classification.

one another and placed on a homogeneous background. The color of the background should be different from the color of the target, otherwise the superpixel will merge into the background and the shape of the superpixel can't be preserved in the patch. In this paper, white is chosen as the color of the background.

3. The patches with superpixels are used as the input to CNN and the label of the superpixel is the output.

A 3500×5000 pixel^2 aerial image can be partitioned into 30000 superpixels, so that only 30000 classifications are required for the classification of the whole image. The computational performance of this method is 10^4 faster than the sliding-window method without losing any resolution.

Different from the sliding-window CNN, DBSCAN is not used in this method although it may decrease the size of the input data. After applying SLIC to the image, most of the superpixels from non-GCPs have a hexagon-like shape as shown in Figure 7. This is useful for distinguishing them from the hourglass shapes of GCPs. If the DBSCAN is applied to the superpixels, the complexity of the superpixel shape will greatly increase. Then much more data are needed to make sure the training data contain all the different shapes of the superpixels.

IV. Numerical Tests

The CNN methods discussed above are applied to a 2D aerial image shown in Figure 8 to identify the GCP. The full image size is 3500 × 5000, the size of the GCP is about 25×30. The number of image patches with and without the GCPs in this image is extremely unbalanced.

A. Binary Classification

To detect the locations of GCPs in the drone photos can be interpreted as a binary classification problem. The architecture

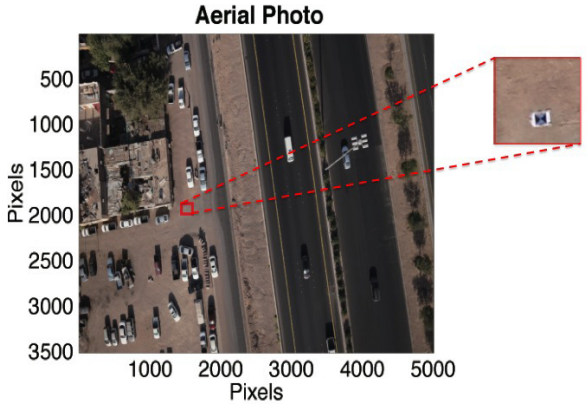


Fig. 8: The input aerial photo. Red box indicates the location of the GCP.

of CNN is illustrated in Figure 2, and the aerial image patch is used as the input layer. The CNN parameters for classification are listed in Table I.

1) Sliding-window CNN with Superpixel-level Voting: Sliding-window CNN with superpixel-level voting is firstly applied to the image. To build a balanced training set, 18000 negative patches are randomly picked in this image and 18000 positive patches are collected from other aerial images. 80% of the patches are used as the training set, and the others are used as the validation set.

After 50 epochs, both the accuracy of the training set and the validation set are around 99%. The CNN labels in the aerial image are shown in Figure 9a. The points in yellow and in blue denote the labels for the GPS GCPs and the non-GCPs. There are about 2.5×10^6 misclassifications in the result, especially from the pixels of sands, cars, and houses. After post-processing by superpixel-level voting, the number

TABLE I: CNN parameters for binary classification

Image Patch	Channels	Convolution Filter	Convolution Layers	Max Pooling	Mini-batch	Stride
65×65	8	2×2	5	2×2	100	9

of false positives is greatly reduced to 3.6 $\times 10^4$ false positives as shown in Figure 9b. The labels of the GCP would be preserved after the voting shown in Figure 10. However, it is still impossible to find the location of the GCP.

There are many types of objects in the aerial image, such as cars, houses, sand, roads and so on. A proper training set should include most types of objects in the image. Although the training set we used contains patches with the GCP, it does not include enough samples with other objects. To improve the performance of the sliding-window CNN, more data should be included in the training set while the number of data examples for each class should be balanced.

2) Superpixel-based CNN: Different from the sliding-window CNN, superpixel-based CNN segments all the objects into superpixels so that the superpixels from all the non-GCP objects have similar shape and color while the GCP has a unique hourglass shape and dark color. Then the dataset for the non-GCPs and GCPs can be easily balanced.

The whole image is segmented into approximately 30000 superpixels using the SLIC algorithm and superpixel-based CNN is applied to the SLIC-processed data. Since the shape and the color of the GCP is known, there is no need to extract positive patches from the aerial images. 3000 synthetic positive training examples are easily built by rotating, shrinking and expanding a dark-color hourglass shape superpixel as shown in Figure 11. 3000 negative superpixels are picked randomly from the superpixels in the image. Only 10 epochs are needed to achieve better than a 99% accuracy for the training and validation examples. The trained CNN network is then applied to all of the superpixels in the image. Only 5 superpixels are classified as GCPs as shown in Figure 12 and all of them have similar hourglass shapes and dark colors. The number of false positive is only 4, which is much less than 3.6 $\times 10^4$ in the sliding-window CNN.

To remove the false positives, the pre-trained sliding-window CNN can be applied to these 5 positive patches. All the false positive patches are then removed and the GCPs are correctly located. The same workflow is applied to the two new images shown in Figure 12b and 12c. The correct locations of GCPs in these two images can be easily obtained from the final positive patches, respectively.

B. Multi-class Classification

To test if these methods can be extended to the detection of other objects, multi-class classification CNN is computed with both methods. The architecture of multi-class CNN is similar to binary CNN, expect the output layer should be multi-class instead of only two labels. The loss function for multi-class is defined as:

$$L = -\sum_{c}^{M} b_{o,c} log(p_{o,c}) \tag{5}$$

where M is the number of classes, p is the predicted probability that the observation o is of class c while b is the binary indicator (0 or 1) if class label c is the correct classification for observation o. The probability p is computed by applying a softmax operation to the output at each node of the CNN.

To obtain the training set for CNN, the aerial image is labeled manually as shown in Figure 13. There are six classes in the aerial image: background, road, car, sand, house and GCP. The CNN parameters for multi-class classification are the same as in Table I.

1) Sliding-window CNN with Superpixel-level Voting: In the sliding-window CNN with superpixel-level voting, 6000 samples are picked randomly from each class in the manually labeled image. However, only 750 pixels belong to the GCP in this image, so the training data for the GCP class is collected from other aerial images. 80% of the patches are used for the training set and the others are used for the validation set.

After 200 epochs, the accuracy of the training set and the validation set are 98.1% and 98.0%, respectively. The CNN labels for multiple classes are shown in Figure 14a.

In Figure 14a , there are many salt-and-pepper errors, such as small cars labeled as houses and small houses labeled as the background. To remove these misclassifications, the superpixel-based majority voting is performed by segmenting the image into segments using the SLIC and DBSCAN algorithm.

The effect of this voting process on the entire image can be seen in Figure 14. Figure 15a and 15d show the selected regions with CNN labels. Figure 15b and 15e show the segmentation of the selected region using SLIC. It can be seen that individual objects, such as cars and trees are segmented into multiple superpixels. The CNN labels belonging to the same superpixels should have the same labels, so the labels in each superpixel are merged into one label that represents the majority class in the superpixel. Figure 15e and 15f show the CNN labels after merging, where it can be seen that the CNN labels after voting clearly represent the shapes of objects. The confusion matrices before and after voting for the entire image are shown in Tables II and III. The salt-and-pepper errors have been greatly removed and the accuracy for each class is improved. For example, the accuracy of the house labeling with CNN without voting was 72.05% compared to 85.09% with voting.

2) Superpixel-based CNN: The superpixel-based CNN is also used for the multi-class classification. 3000 GCP superpixel training examples are created in the same way that was used for the binary classification and 3000 superpixels are picked randomly for each class from the superpixels in the image. 200 epochs are preformed to achieve an accuracy greater than 98% for the training and validation sets. The CNN labels and the confusion matrices are shown in Figure 16a and Table IV. There are many background superpixels labeled as sands since both the the sand and background superpixels have

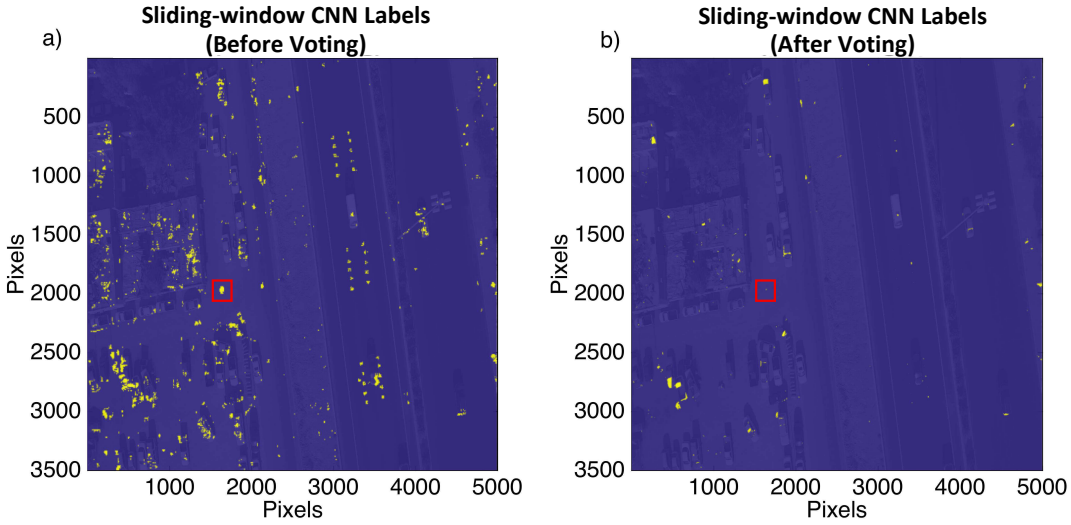


Fig. 9: Sliding-window CNN binary labeled image a) before and b) after majority voting. Red box indicates the location of the GCP and the points in yellow and blue denote the labels for the GCP and the non-GCP, respectively.

TABLE II: Confusion matrix for sliding-window CNN multi-class labels before majority voting

		Predicted Labels					
	%	Background	Road	Car	Sand	House	GCP
	Background	69.43	2.69	11.17	1.04	13.70	1.97
	Road	7.56	84.77	4.83	0.04	2.53	0.25
True Labels	Car	9.41	0.74	65.47	0.33	15.18	8.88
	Sand	4.30	0	0.73	67.20	25.44	2.33
	House	11.47	0.38	8.02	2.59	72.05	5.52
	GCP	0	0	0	0	0	100

TABLE III: Confusion matrix for sliding-window CNN multi-class labels after majority voting

		Predicted Labels					
	%	Background	Road	Car	Sand	House	GCP
	Background	77.59	2.34	8.90	0.28	10.34	0.55
	Road	1.62	94.69	2.44	0	1.13	0.12
True Labels	Car	11.00	2.42	69.03	0.06	12.47	4.11
	Sand	0.64	0	0.23	87.78	10.95	0.40
	House	7.38	0	4.21	1.05	85.09	2.26
	GCP	0	0	0	0	0	100

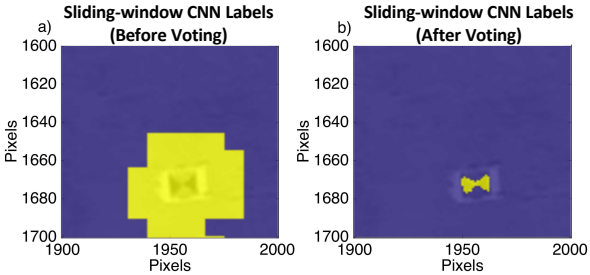


Fig. 10: Sliding-window CNN binary labeled image a) before and b) after majority voting in the red box in Figure 9.

a similar dark yellow color and hexagonal-like shape. Sliding-window CNN can recognize the differences between the sand and background since it takes the surrounding information into account.

To refine the superpixel-based CNN classification result, the majority voting in the sliding-window CNN can also be applied since DBSCAN will include some surrounding information. The majority voting removes many misclassifications, especially the background pixels labeled as sand, as shown in Figure 16b and Table V.

V. Discussion

Both the sliding-window CNN with majority voting and superpixel-based CNN highly depend on the successfully application of SLIC. All the pixels in the GCP must be partitioned into the same superpixel. However, there are factors which can make the performance of SLIC unstable. The first factor is the shadow, the color contrast of the GCP is not obvious when the GCP is located in the shadow shown in

TABLE IV: Confusion matrix for superpixel-based CNN multi-class labels before majority voting

		Predicted Labels					
	%	Background	Road	Car	Sand	House	GCP
	Background	49.87	9.08	6.27	17.23	17.47	0.08
	Road	2.5	96.55	0.43	0.04	0.44	0.03
True Labels	Car	3.36	2.32	90.08	0.67	3.56	0
	Sand	3.77	0.11	0.52	78.85	16.74	0
	House	6.44	2.06	2.18	11.24	78.00	0.09
	GCP	0	0	0	0	0	100

TABLE V: Confusion matrix for superpixel-based CNN multi-class labels after majority voting

		Predicted Labels					
	%	Background	Road	Car	Sand	House	GCP
	Background	75.86	8.93	3.66	2.14	9.35	0.04
	Road	1.19	98.59	0.17	0	0.04	0.01
True Labels	Car	7.93	3.00	85.59	0.32	3.16	0.00
	Sand	0.97	0.07	0.12	91.68	7.15	0
	House	6.08	1.00	1.78	12.22	78.9	0.01
	GCP	0	0	0	0	0	100

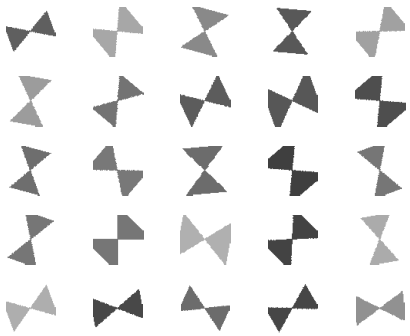


Fig. 11: The synthetic positive patches created for superpixel-based CNN by rotating, shrinking and expanding a dark-color hourglass shape superpixel.

Figure 17a and 17b. The compactness variable m in equation 3 should be small so that SLIC can detect small color contrasts. The second factor is the number k of superpixels in SLIC, where small values of k can lead to inaccuracies. The GCP may be partitioned into a large superpixel and the superpixel doesn't have a hourglass shape. If k is too big, the GCP may be separated into several small superpixels as shown in Figure 17c and 17d. To mitigate this problem, DBSCAN can merge these superpixel into a hourglass-shape superpixel. However, it will increase the diversity of the superpixel and decrease the accuracy of the CNN classification.

Another limitation of this method is the high computational cost of SLIC for large images. To mitigate this problem, selective search [19] or a region proposal network [20] can be used to replace SLIC.

VI. Conclusions

We used both sliding-window CNN and superpixel-based CNN for automatically extracting the locations of objects from aerial photographs.

In the sliding-window CNN, the simple linear iterative clustering (SLIC) and majority voting are applied in the post-processing to remove the misclassifications from the CNN results. This method accurately detects the locations of various objects and clearly delineates their boundaries. However, it cannot unambiguously locate the GCP due to the creation of many false positives.

In the superpixel-based CNN, SLIC is applied in the pre-processing to extract the unique shape and color of the GCP. This method can quickly narrow the scope to less than 10 superpixels labeled as GCPs. An additional sliding-window CNN can further eliminate false positives. The numerical tests indicate the efficiency of the method in locating the GCP, but it doesn't perform well in locating other objects since the superpixels don't contain the background information.

VII. Acknowledgements

The research reported in this publication was supported by the King Abdullah University of Science and Technology (KAUST) in Thuwal, Saudi Arabia. We are grateful to the sponsors of the Center for Subsurface Imaging and Modeling Consortium for their financial support. For computer time, this research used the resources of the Supercomputing Laboratory at KAUST and the IT Research Computing Group. We thank them for providing the computational resources required for carrying out this work. We must thank FalconViz for providing the dataset.

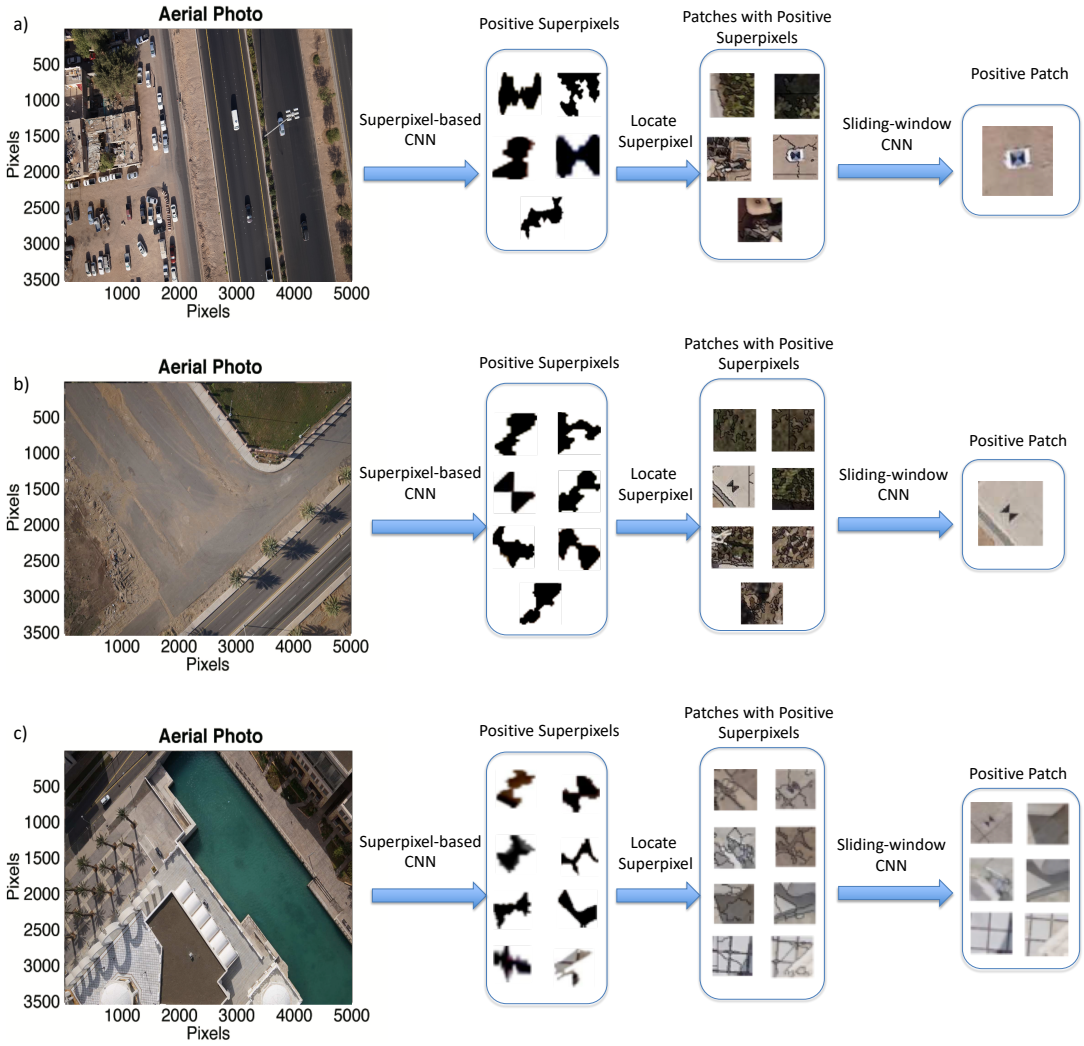


Fig. 12: Superpixel-based and sliding-window binary classifications for a) Figure 8, b) test image 1 and c) test image 2. The positive superpixels are firstly found by superpixel-based CNN classification and the false positives among the positive superpixels are then removed by sliding-window CNN classification.

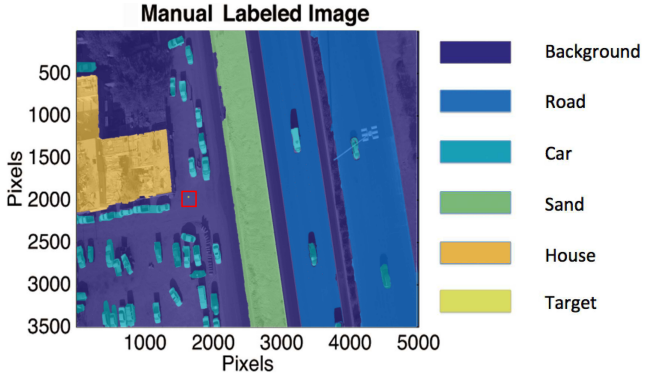


Fig. 13: Manually labeled image with Figure 8. Red box indicates the location of the GCP.

APPENDIX A
DENSITY-BASED SPATIAL CLUSTERING OF APPICATIONS WITH NOISE

Density-based spatial clustering of applications with noise (DBSCAN) is a density-based data clustering algorithm [21, 22]. Given a set of points in some space, it groups together points that are closely packed together (points with many nearby neighbors), and marks the outlier points that lie alone in low-density regions (whose nearest neighbors are too far away). There are two parameters need to be specified in DBSCAN: a distance threshold, ϵ and minimum number of points, $MinPts$.

DBSCAN starts with an arbitrary seed point which has at least $MinPts$ points nearby within the distance of ϵ. Then we search each of these nearby points. For a given nearby point, if there are fewer than $MinPts$ points within its radius ϵ, this point is called a $leaf$. The search would stop at the $leaf$ point. If there are at least $MinPts$ points within its radius ϵ, this point is called a $branch$ point, the search would continue to the nearby points around the $branch$ point until all the nearby points are $leaf$ point. The first round of search is finished when no $branch$ points appear. All the points that have been searched in the first round belong to the same cluster

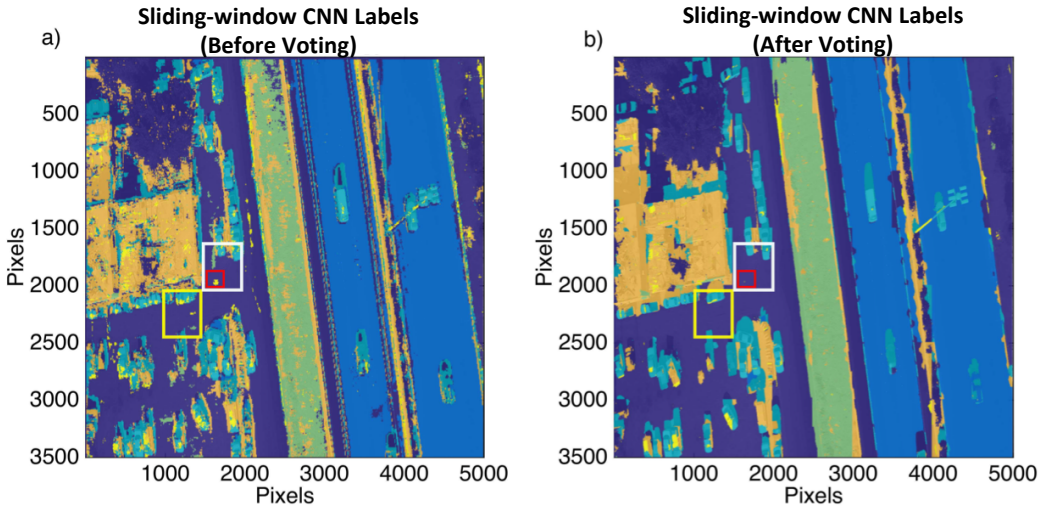


Fig. 14: Sliding-window CNN multi-class labeled image a) before majority voting and b) after majority voting. Red box indicates the location of the GCP.

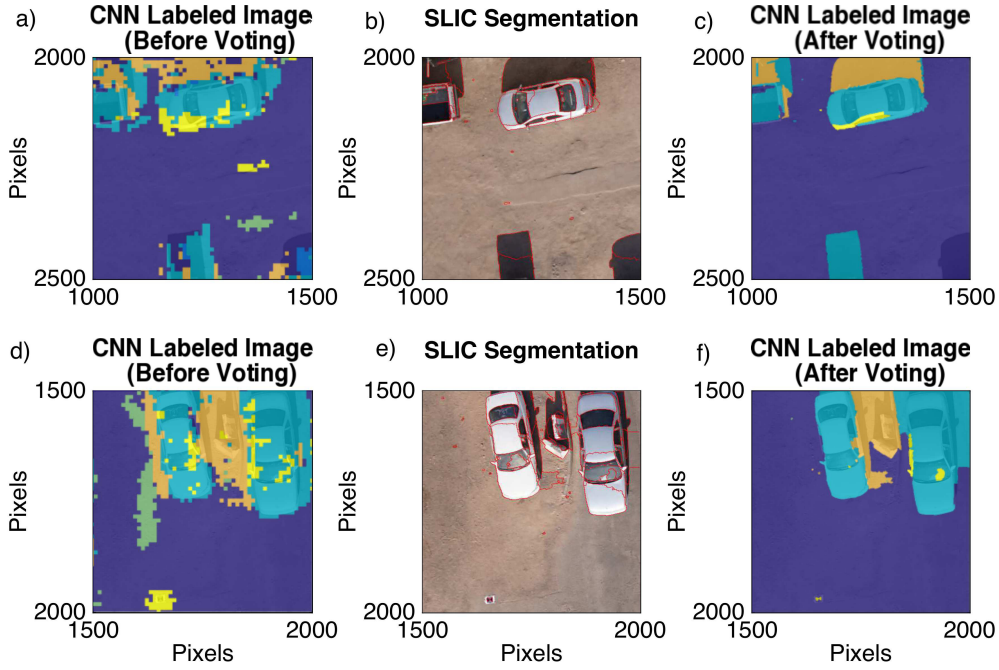


Fig. 15: a) Sliding-window CNN multi-class labeled image before majority voting, b) segmentation and c) CNN multi-class labeled image after majority voting for the yellow box in Figure 14. d) Sliding-window CNN multi-class labeled image before majority voting, e) segmentation and f) CNN multi-class labeled image after majority voting for the white box in Figure 14.

and we never revisit them later. Then a new arbitrary point is picked and another round search start. This continues until all of points in the images are assigned. If a point has fewer than $MinPts$ points with its radius ϵ, and it is not a leaf node of another cluster. It's labeled as a $noise$ point. An example of the DBSCAN is shown in Figure 18.

In this paper, we apply DBSCAN to all the updated superpixel centers $[L_k, a_k, b_k, x_k, y_k]^T$. The distance between two centers is calculated using equation 3. If the superpixel centers belong to the same cluster, then the superpixels associated with these centers would be merged in the final segmentation.

REFERENCES

[1] V. Mnih and G. E. Hinton, "Learning to label aerial images from noisy data," pp. 567–574, 2012.

[2] M. Längkvist, A. Kiselev, M. Alirezaie, and A. Loutfi, "Classification and segmentation of satellite orthoimagery using convolutional neural networks," *Remote Sensing*, vol. 8, no. 4, p. 329, 2016.

[3] M. Quigley, S. Batra, S. Gould, E. Klingbeil, Q. Le, A. Wellman, and A. Y. Ng, "High-accuracy 3D sensing for mobile manipulation: Improving object detection and

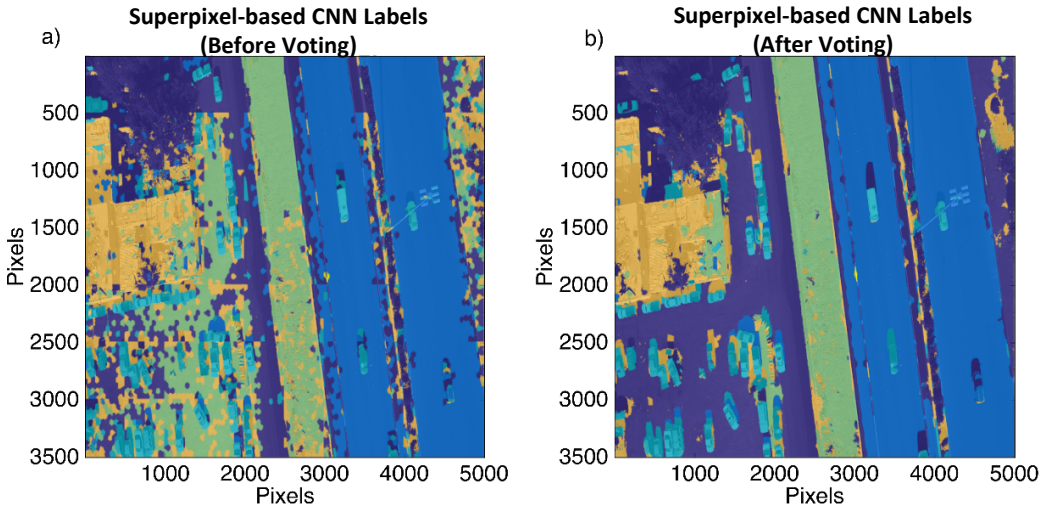


Fig. 16: a) Superpixel-based CNN multi-class labeled image before majority voting and b) CNN multi-class labeled image after majority voting.

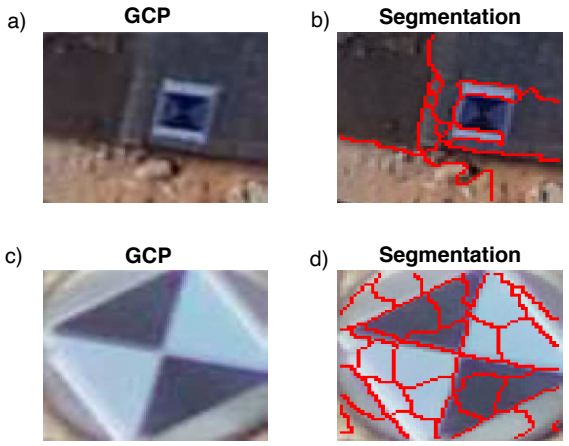


Fig. 17: a) A GCP in the shadow and b) its segmentation. c) A big GCP and d) its segmentation.

door opening," pp. 2816–2822, 2009.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," pp. 1097–1105, 2012.

[5] Y. Zhang, K. Sohn, R. Villegas, G. Pan, and H. Lee, "Improving object detection with deep convolutional networks via Bayesian optimization and structured prediction," pp. 249–258, 2015.

[6] M. Castelluccio, G. Poggi, C. Sansone, and L. Verdoliva, "Land use classification in remote sensing images by convolutional neural networks," *arXiv preprint arXiv:1508.00092*, 2015.

[7] A. Albert, J. Kaur, and M. C. Gonzalez, "Using convolutional networks and satellite imagery to identify patterns in urban environments at a large scale," pp. 1357–1366, 2017.

[8] V. Khryashchev, V. Pavlov, A. Priorov, and E. Kazina, "Convolutional neural network for satellite imagery," pp. 344–347, 2018.

[9] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[10] S. Feng, Y. Lin, and B. Wohlberg, "Physically realistic training data construction for data-driven full-waveform inversion and traveltime tomography," in *SEG Technical Program Expanded Abstracts 2020*. Society of Exploration Geophysicists, 2020, pp. 3472–3476.

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," pp. 1–9, 2015.

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," pp. 770–778, 2016.

[13] V. Mnih and G. E. Hinton, "Learning to detect roads in high-resolution aerial images," pp. 210–223, 2010.

[14] J. Li, Y. Chen, and G. T. Schuster, "Separation of multimode surface waves by supervised machine learning methods," *Geophysical Prospecting*, vol. 68, no. 4, pp. 1270–1280, 2020.

[15] S. Kluckner, T. Mauthner, P. M. Roth, and H. Bischof, "Semantic classification in aerial imagery by integrating appearance and height information," pp. 477–488, 2009.

[16] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Süsstrunk *et al.*, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012.

[17] Y. Tang, W. Cai, and B. Xu, "Profiles of phenolics, carotenoids and antioxidative capacities of thermal processed white, yellow, orange and purple sweet potatoes grown in guilin, china," *Food Science and Human Wellness*, vol. 4, no. 3, pp. 123–132, 2015.

[18] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE transactions*

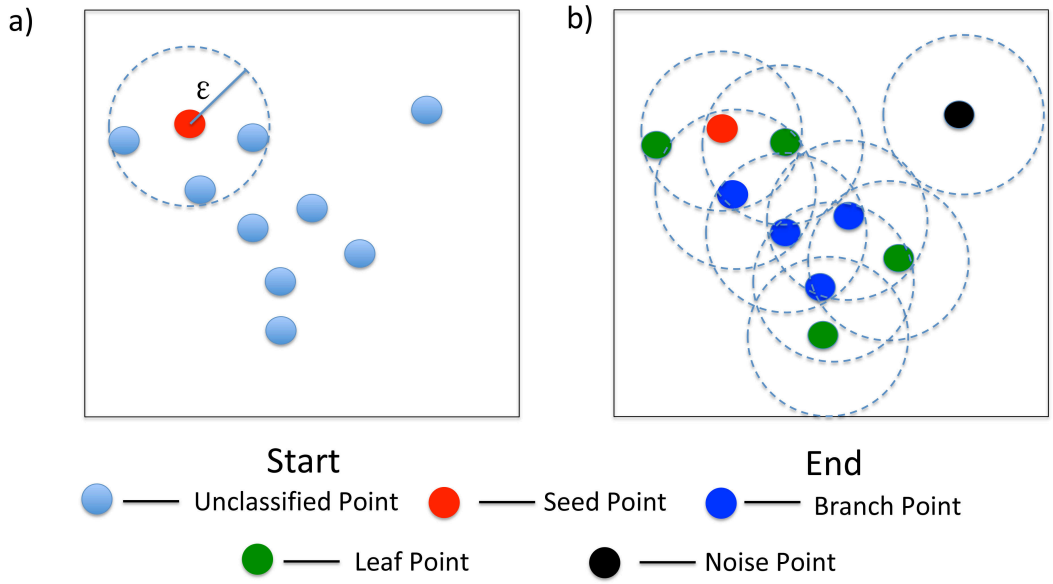


Fig. 18: A example of DBSCAN with $MinPts = 3$. a) The start of the search. b) The end of the search.

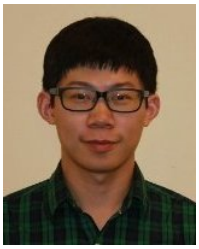
on neural networks and learning systems, vol. 30, no. 11, pp. 3212–3232, 2019.

[19] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

[20] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," pp. 91–99, 2015.

[21] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." vol. 96, no. 34, pp. 226–231, 1996.

[22] K. Lu and S. Feng, "Auto-windowed super-virtual interferometry via machine learning: A strategy of first-arrival traveltime automatic picking for noisy seismic data," pp. 10–14, 2018.



Shihang Feng received B.S degree in Geophysics from China University of Petroleum-Beijing in 2012, M.S degree in Geophysics from Univerisity of Utah in 2014 and Ph.D. degree in Earth Science and Engineering from King Abdullah University of Science and Technology in 2019.

Currently, he is a Postdoc in Los Alamos National Laboratory. His main research interests include machine learning, seismic imaging and remote sensing.

Luca Passone earned his PhD in Earth Science and Engineering from King Abdullah University of Science and Technology in 2018 for his work in ground motion simulation for seismic hazard.

In his current role of General Manager and CTO at FalconViz, he is solving technological challenges in the acquisition and processing of geospatial data, with emphasis on photogrammetry and UAVs.

Gerard T. Schuster has an MS (1982) and a PhD (1984) from Columbia University and was a postdoctoral researcher there from 1984-1985. From 1985 to 2009 he was a professor of Geophysics at University of Utah. He left Utah to start his current position as Professor of Geophysics at KAUST in 2009. He received a number of teaching and research awards while at University of Utah. He was editor of Geophysics from 2004-2005 and was awarded SEG's Virgil Kauffman gold medal in 2010 for his work in seismic interferometry.

He is currently a Professor of Geophysics at King Abdullah University Science and Technology (KAUST) and an adjunct Professor of Geophysics at University of Utah. He was the founder and director of the Utah Tomography and Modeling/Migration consortium from 1987 to 2009, and is now the co-director and founder of the Center for Fluid Modeling and Seismic Imaging at KAUST.