Synthetic-Aperture Radar image based positioning in GPS-denied environments using Deep Cosine Similarity Neural Networks

Article *in* Inverse Problems and Imaging · January 2020

DOI: 10.3934/ipi.2021013

Seonho Park
Georgia Institute of Technology
24 PUBLICATIONS **164** CITATIONS

Panos Pardalos
University of Florida
1,728 PUBLICATIONS **48,337** CITATIONS

Inverse Problems and Imaging doi:10.3934/ipi.2021013
Volume 15, No. 4, 2021, 763–785

SYNTHETIC-APERTURE RADAR IMAGE BASED POSITIONING IN GPS-DENIED ENVIRONMENTS USING DEEP COSINE SIMILARITY NEURAL NETWORKS

Seonho Park

Department of Industrial and Systems Engineering
University of Florida, Gainesville, FL 32611, USA

Maciej Rysz*

Department of Information Systems and Analytics
Miami University, Oxford, OH 45056, USA

Kaitlin L. Fair

Air Force Research Laboratory (AFRL/RWWI)
Eglin Air Force Base, FL 32542, USA

Panos M. Pardalos

Department of Industrial and Systems Engineering
University of Florida, Gainesville, FL 32611, USA

(Communicated by Margaret Cheney)

Abstract. Navigating unmanned aerial vehicles in precarious environments is of great importance. It is necessary to rely on alternative information processing techniques to attain spatial information that is required for navigation in such settings. This paper introduces a novel deep learning-based approach for navigating that exclusively relies on synthetic aperture radar (SAR) images. The proposed method utilizes deep neural networks (DNNs) for image matching, retrieval, and registration. To this end, we introduce Deep Cosine Similarity Neural Networks (DCSNNs) for mapping SAR images to a global descriptive feature vector. We also introduce a fine-tuning algorithm for DCSNNs, and DCSNNs are used to generate a database of feature vectors for SAR images that span a geographic area of interest, which, in turn, are compared against a feature vector of an inquiry image. Images similar to the inquiry are retrieved from the database by using a scalable distance measure between the feature vector outputs of DCSNN. Methods for reranking the retrieved SAR images that are used to update position coordinates of an inquiry SAR image by estimating from the best retrieved SAR image are also introduced. Numerical experiments comparing with baselines on the Polarimetric SAR (PolSAR) images are presented.

1. Introduction.

Unmanned aerial vehicles (UAVs) have become an integral part of reconnaissance and defense applications, and routinely operate in hostile and uncertain environments. UAV flight maneuvering and navigation can be autonomously

2020 *Mathematics Subject Classification.* Primary: 68T07, 68U10; Secondary: 65D18.

Key words and phrases. Synthetic Aperture Radar (SAR), polarimetric SAR (PolSAR), Deep Cosine Similarity Neural Networks, image graphs, image retrieval, image registration, scale-invariant feature transform (SIFT), SAR-SIFT, remote sensing.

Distribution A: Approved for public release, distribution is unlimited, 96TW-2020-0170.

* Corresponding author: Maciej Rysz.

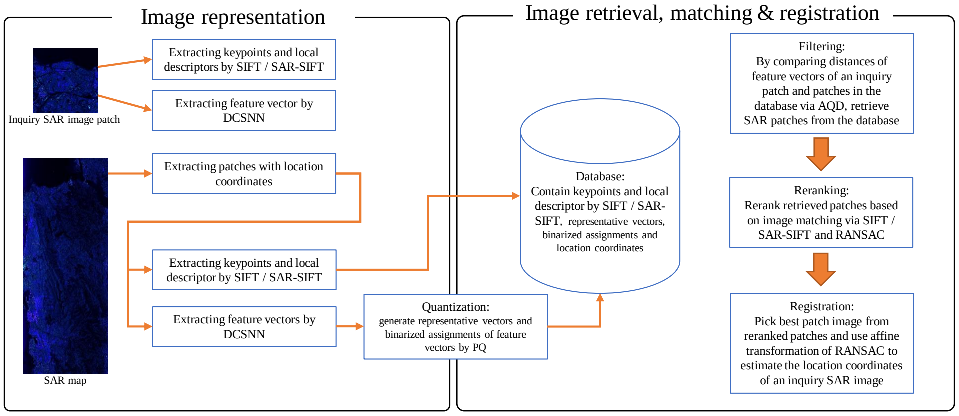


Image representation
Inquiry SAR image patch
Extracting keypoints and local descriptors by SIFT / SAR-SIFT
Extracting feature vector by DCSNN
SAR map
Extracting patches with location coordinates
Extracting keypoints and local descriptor by SIFT / SAR-SIFT
Extracting feature vectors by DCSNN
Quantization: generate representative vectors and binarized assignments of feature vectors by PQ
Image retrieval, matching & registration
Database: Contain keypoints and local descriptor by SIFT / SAR-SIFT, representative vectors, binarized assignments and location coordinates
Filtering: By comparing distances of feature vectors of an inquiry patch and patches in the database via AQD, retrieve SAR patches from the database
Reranking: Rerank retrieved patches based on image matching via SIFT / SAR-SIFT and RANSAC
Registration: Pick best patch image from reranked patches and use affine transformation of RANSAC to estimate the location coordinates of an inquiry SAR image

Figure 1. Overview of the system for SAR aided navigating by SAR image representation, matching and registration

controlled and often relies on an embedded global positioning system (GPS). Although the GPS provides position, navigation and timing (PNT) information, in the GPS-denied settings it is necessary to obtain analogous spatial and temporal awareness via other mechanisms. Such settings may include, but are not limited to, operations in areas with GPS jamming devices, interference and outages. Indeed, experiments have demonstrated that even a low-power jamming device can interfere GPS signals, resulting in possible denial of GPS service over large areas [16]. Additionally, attackers can control a maritime surface vessel by broadcasting counterfeit GPS signals in order to manipulate a target receiver's position, velocity, or time [4].

Numerous studies in the literature have considered aided navigation in the GPS-denied environments. A classical approach relies on Dead Reckoning (DR), which involves estimating position based on a previously determined position integrated with velocity or acceleration. Although it has been shown to work effectively in the absence of GPS signal, a major drawback stems from the fact that it accumulates position errors over time.

Vision-aided navigation may be a promising alternative that has been widely studied for a decade [39, 48, 31, 37, 7, 8, 3, 5, 18]. Visual Odometry (VO), which was termed by Nister *et al.* [31], estimates position and orientation by analyzing the sequence of images. Namely, VO estimates the UAV's current position with respect to a previously acquired position by accumulating inter-frame translation and rotation. VO can also be combined with the Simultaneous Localization and Mapping (SLAM) technique [45, 5] as well as several other fusing methods including filtering methods such as extended Kalman filter [3, 48] and State-Dependent Riccati Equation nonlinear filter [30]. Furthermore, there exist previous works that fused measured information from inertial measurement unit (IMU) [48, 30, 3], and onboard cameras [39, 5]. Another subject of emphasis within the scope considered image registration aided navigation [37, 32]. Mo Shan *et al.* [37] proposed a method that used image feature extraction via Histograms of Oriented Gradients (HOG) [9], which demonstrated promising results on image registration on Google Maps.

Nitti *et al.* [32] explored the use of interferometric synthetic aperture radar (InSAR) images for image registration to aid navigation in GPS-denied environments.

Their proposed approaches are classified into two categories: cases when both SAR amplitude and phase images are available; and cases when SAR amplitude images are not available. In the former, their method is based on a comparison between SAR images acquired from on-board equipment and a terrain landmark database that contains geographic information and ground landmarks. In the latter, they additionally exploit terrain elevation references acquired from digital terrain model (DTM) as well as SAR phase data. The image coordinates of the expected landmarks of an inquiry SAR image were obtained by an Automatic Target Detection and Recognition (ATR) algorithm [34] and correlated with coordinates stored in the database.

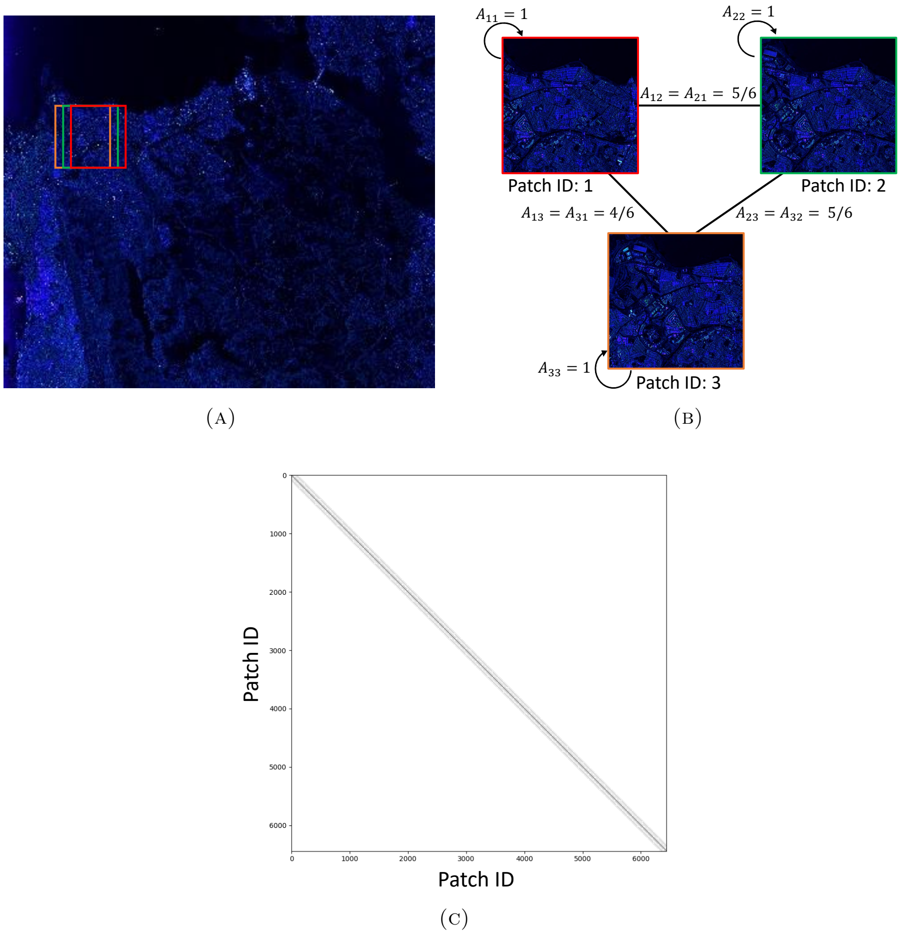
More recent image matching and registration techniques have exploited capabilities of deep neural networks (DNNs) [2, 42, 47, 49, 15, 33]. Convolutional neural networks (CNNs) have been widely used for mapping complicated images to "simpler" feature vectors. The feature vectors, which are also called *global descriptors* of images, are used to compare and retrieve similar images from a database. Also, faster and more scalable CNN inference allow for large-scale image retrieval tasks [33] that would otherwise be difficult with conventional image descriptors such as scale-invariant feature transform (SIFT) [27, 28] or LIFT [46].

In this work, we introduce a novel approach to aid navigation in GPS-denied environments by using a CNN-based SAR image descriptor. The general procedure and corresponding article sections are as follows. Section 2 furnishes the procedure for representing SAR images using the CNN-based descriptor. This involves a method to generate a SAR image search database, which also serves as a training dataset for the CNN-based descriptor. We introduce a novel CNN-based image descriptor, the *Deep Cosine Similarity Neural Network* (DCSNN), and its training procedure. The DCSNN is designed to generate a SAR image descriptor, a simple feature vector that contains salient information of the image, that can be compared via cosine similarity. In Section 3, we present a process to estimate the positioning of an inquiry SAR image, using the DCSNN and scalable feature vector representations via product quantization (PQ) [17]. Given an inquiry SAR image, we infer a feature vector by the DCSNN, compare cosine similarity distances between the feature vector and feature vectors in the database, and retrieve SAR images whose feature vectors are similar to inquiry image's. After retrieving SAR images, the conventional image feature description methods such as SAR-SIFT [10] or SIFT [28], and the image matching method, RANdom SAmple Consensus (RANSAC) [14], are used to rerank the retrieved images. Finally, an affine transformation associated with RANSAC is utilized to estimate the position of an inquiry image by adjusting the coordinates of the best retrieved SAR image. A flowchart depicting the general procedure in this article is provided in Fig. 1. Numerical experiments demonstrating the performance of our approach against several baseline methods that employ DNN-based binary hashing methods are provided in Section 4.

The primary contributions of this work are the following:

- A novel CNN-based image descriptor, DCSNN, that utilizes a graph-based representation of SAR image patches for training.
- The use of cosine similarity, induced by the DCSNN and equipped with PQ, to effectively measure the distances between feature vectors of SAR image patches in a scalable manner.

- A navigation procedure using SAR image matching, retrieval and registration that obtains and correlates current coordinates of a vehicle with coordinates retrieved from a database.
- The methodology is validated and shown to be effective for polarimetric SAR (PolSAR) image data from UAVSAR [1].



(a)

(b)

(c)

Figure 2. Example of a patch graph. (a) a SAR map from the UAVSAR dataset (b) patches extracted from the SAR map and its associated graph (c) visualization of the adjacency matrix. White cells show zero edge values whereas black cells show nonzero values.

2. Deep cosine similarity neural networks for image matching.

In this section we first describe the procedure for extracting SAR image patches from a SAR map and generating a corresponding graph that represents the SAR map, which are utilized as the training dataset for the DCSNN. Thereafter, we introduce the DCSNN as a means of efficiently mapping a given SAR patch image to a single descriptive vector. An integrated fine-tuning procedure for the DCSNN is also described. After fine-tuning, the output vector of the DCSNN serves as a feature vector that is compared against feature vectors of images in a database. Namely, given an inquiry SAR patch, images for the database with similar attributes are matched and retrieved.

2.1. Generating SAR patch graph.

To guarantee a well-posed modeling framework we assume availability of sufficient SAR image map data to cover the region of navigation. SAR maps in their entirety are usually too large to be utilized as inputs for DNNs. To develop computationally tractable process, we therefore extract image *patches* from a given SAR map. Fig. 2 presents a PolSAR map obtained from the UAVSAR dataset [1] that is used throughout this study, where the red, green, and orange rectangles represent patches as shown in Fig. 2a and 2b. Specifically, patches of size 600 × 600 pixels with a stride of 100 pixels are extracted and used for subsequent analysis.

A key concept in our methodology involves a connectivity mapping of the image patches, which is represented as an undirected *graph* consisting of *nodes* corresponding to different patch images, and *edges* representing the similarities between two patches. Nodes are labeled according to the assigned unique patch IDs. A graph's characteristics are usually represented by an *adjacency matrix*, $\mathbf{A} \in \mathbb{R}^{N\times N}$, where N is the number of nodes (patches). Given patches $i, j \in \{1, \dots, N\}$, let element A_{ij} of matrix $\mathbf{A}$ corresponds to the edge value defined as the common pixel area ratio between patches i and j. Formally, the edge values, A_{ij}, $\forall i, j \in \{1, \dots, N\}$, measure the similarities between patches as,

$$A_{ij} = \max(0, \frac{2 \cdot \mathrm{Area}_{ij}}{\mathrm{Area}_i + \mathrm{Area}_j}), \ \forall i, j \in \{1, \dots, N\}, \tag{1}$$

where Area_i and Area_j are pixel areas of patch i and j, respectively; and Area_{ij} is the intersection of pixel area between patches i and j. Observe that the nonzero term in (1) corresponds to the *Dice score* [12], a commonly used performance measure for image segmentation tasks. Clearly, matrix $\mathbf{A}$ is symmetric and sparse since any given patch tends to be similar to its neighboring patches with overlapping pixel areas. Fig. 2c illustrates the adjacency matrix visualization corresponding to the PolSAR map used in our experiments.

Note that the constructed adjacency matrix $\mathbf{A}$ represents the similarity between SAR patches, and that its elements can be used as labels between any pair of patches. In other words, matrix $\mathbf{A}$ readily furnishes labels that would otherwise be extremely labor intensive to obtain for large-scale data in an ad-hoc manner by a human. Consequently, given the low computational expense for obtaining the SAR patches and corresponding matrix $\mathbf{A}$, a neural network can be trained efficiently in a supervised way.

Next, a SAR image search database consisting of information derived from the image patches and their corresponding adjacency matrix $\mathbf{A}$ is constructed (see Fig. 1). It is assumed that location coordinates of all image patches are also known and stored. Denote the set of image patches in the database by $\mathbf{X} = \left\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(N)}\right\}$, where $\mathbf{x}^{(i)}$ represents an image patch $i \in \{1, \dots, N\}$. The patches in $\mathbf{X}$ and matrix $\mathbf{A}$ are used as the training dataset. Details about training the DCSNN model to generate global descriptive feature vector are described in the next subsections.

2.2. Deep cosine similarity neural network.

CNN-based descriptors have been effectively used in numerous image analysis and retrieval applications. Structuring our methodology accordingly, the primary goal of the DCSNN model is to efficiently construct a "simple" descriptive vector of a given SAR image. The descriptive feature vector of an inquiry patch image can then be compared against feature vectors of patch images stored in a database, thereby, retrieving similar patches along with their location coordinates (e.g., latitude and longitude).

Previous efforts demonstrate that the pretrained CNN-based descriptors, which is trained on general image datasets such as ImageNet [11], work well for such downstream tasks [2, 42, 49]. To increase the performance of tasks that use SAR data, a new CNN-based descriptor can be trained on the SAR data by using the pretrained CNN-based descriptor as an initialization for the new descriptor, which is also known as *fine-tuning*. To "fine-tune" the neural network model in a supervised manner, we use the adjacency matrix $\mathbf{A}$ described in Subsection 2.1 as labels of the SAR images. We first define a loss function to train the DCSNN and present the stochastic algorithm to minimize the loss function in the following subsection.

Define the DCSNN model as a CNN-based mapping $f_{\boldsymbol{\theta}}$ parameterized by $\boldsymbol{\theta}$, where the parameters $\boldsymbol{\theta}$ are learned during fine-tuning. Let $\mathbf{d} \in \mathbb{R}^l$ be a feature vector of feature length l that is obtained from the DCSNN as $\mathbf{d} = f_{\boldsymbol{\theta}}(\mathbf{x})$. Clearly, to exclusively use the feature vector $\mathbf{d}$ for comparing patch images during the retrieval process, it is required that it be sufficiently "compact" yet descriptive of the image $\mathbf{x}$. For image patches stored in the database, we additionally construct an *anchor matrix* $\mathbf{D} \in \mathbb{R}^{l\times N}$ such that the ith column corresponds to the feature vector $\mathbf{d}^{(i)}$ of the patch $\mathbf{x}^{(i)} \in \mathbf{X}$.

We define the loss function, $L^{(i)}$, of the DCSNN as follows. For each patch $\mathbf{x}^{(i)}$ in the database, $L^{(i)}$ consists of a *cross-entropy loss* $L_{ce}^{(i)}$ and a *regularization loss* $L_{reg}^{(i)}$:

$$L^{(i)} = L_{ce}^{(i)} + \lambda L_{reg}^{(i)}, \tag{2}$$

where $\lambda > 0$ is a regularization factor. Let $\mathbf{A}_i$ be the ith row vector of the adjacency matrix $\mathbf{A}$ (i.e., $\mathbf{A}_i$ is associated with the patch i), then the cross-entropy loss takes the form,

$$L_{ce}^{(i)} = -\left(\mathbf{A}_i^T \log \Omega^{(i)} + (\mathbf{1} - \mathbf{A}_i^T) \log (\mathbf{1} - \Omega^{(i)})\right) \tag{3}$$

$$\text{where } \Omega^{(i)} = \sigma\left(\frac{\mathbf{D}^T \mathbf{d}^{(i)}}{s}\right), \tag{4}$$

and $\sigma(\cdot)$ is an element-wise sigmoid function, i.e., $\sigma(x) = \frac{1}{1+\exp^{-x}}$, that forces the dot product of feature vectors to range between 0 and 1. Since the gradient of the sigmoid function $\sigma(\cdot)$ approaches 0 as its value nears $\pm\infty$, it results in longer training times. To mitigate this, in (4) we impose a *similarity factor* $s > 0$, which was introduced in previous literature [24]. Observing that an element A_{ij} of the adjacency matrix $\mathbf{A}$ must be in the range [0,1], A_{ij} can be interpreted as a probability that the patch i is equivalent to the patch j. Thus, $\mathbf{A}_i$ in (3) can serve as "ground truth" probabilities, whereas $\Omega^{(i)}$ is the predicted probability that patch i is similar to other patches.

Observe that the jth element of $\Omega^{(i)}$, denoted by $\Omega_j^{(i)}$, corresponds to the estimated probability that patch i is similar to patch j. Thus, the higher the value of an element A_{ij} is, the higher the value of $\Omega_j^{(i)}$ that the DCSNN is expected to generate. Accordingly, we apply a regularization loss, $L_{reg}^{(i)}$, such that the norm of the feature vector produced by the DCSNN is expected to be approximately 1

Algorithm 1 Fine-tuning DCSNN

```
1: Input: learning rate $\eta$, minibatch size $N_{\mathcal{B}}$, stochastic anchor matrix size $N_{\mathcal{D}}$.
2: Initialize $\mathbf{D}$ by computing $\mathbf{d}^{(i)} = f_{\boldsymbol{\theta}}(\mathbf{x}^{(i)})$, $\forall \mathbf{x}^{(i)} \in \mathbf{X}$.
3: repeat
4:     Select a random subset $\mathcal{B}$ of $\{1, \cdots, N\}$ of size $N_{\mathcal{B}}$.
5:     Select a random subset $\mathcal{D}$ of $\{1, \cdots, N\}$ of size $N_{\mathcal{D}}$.
6:     Define $\tilde{\mathbf{D}}$ and $\tilde{\mathbf{A}}$ whose columns correspond to $\mathbf{d}^{(i)}$ and $\mathbf{A}_i$ $\forall i \in \mathcal{D}$, respectively
7:     $\triangle\boldsymbol{\theta} \leftarrow \frac{1}{N_B} \sum_{i\in\mathcal{B}} \frac{\partial}{\partial\boldsymbol{\theta}} \left(\tilde{L}_{ce}^{(i)} + \lambda L_{reg}^{(i)}\right)$
8:     $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \triangle\boldsymbol{\theta}$
9:     Update $\mathbf{D}$ with $\mathbf{d}^{(i)}$, $\forall i \in \mathcal{B}$
10: until $\boldsymbol{\theta}$ has converged
11: Output: learned parameters $\boldsymbol{\theta}$ of DCSNN.
```

upon successful fine-tuning. To this effect, the regularization loss considered in (2) is defined as,

$$L_{reg}^{(i)} = (\|\mathbf{d}^{(i)}\|_2 - 1)^2. \tag{5}$$

After training, by using “regularized” feature vectors from the DCSNN, it is expected that the dot products of the feature vectors are consistent with their cosine similarities. Thus, the cosine similarity between feature vectors generated by the model measures the extent of adjacency of the image patches. This fact motivated the adopted name “DCSNN”.

Finally, by averaging over the N patches, the total loss function takes the form

$$L = L_{ce} + \lambda L_{reg} = \frac{1}{N} \sum_{i=1}^{N} \left(L_{ce}^{(i)} + \lambda L_{reg}^{(i)}\right). \tag{6}$$

2.3. A stochastic gradient descent algorithm for fine-tuning the DCSNN.

This subsection describes a procedure for fine-tuning the DCSNN. The most common training approach for cases with a large number of images are stochastic optimization algorithms. Indeed, typical first and second order stochastic optimization algorithms [20, 13, 35] can efficiently minimize the loss function (6). In this article, we utilize *stochastic gradient descent* (SGD), which is the most prototypical first-order stochastic optimization algorithm. Our procedure is illustrated in Algorithm 1 and described next.

The algorithm is initialized with a fixed learning rate η, minibatch size $N_{\mathcal{B}}$, stochastic anchor matrix size $N_{\mathcal{D}}$, and anchor matrix $\mathbf{D}$ (lines 1-2). To manage a large number of image patches in the database, we use subsets of patches to estimate the value of the loss function and its derivative stochastically. During each iteration, a subset of randomly selected patches, $\mathcal{B} \subseteq \{1, \dots, N\}$, of size $N_{\mathcal{B}}$ is created. The images $\mathbf{x}^{(i)}$, $\forall i \in \mathcal{B}$, are used to estimate the gradient of the loss function with respect to the parameters of the DCSNN model (line 7) and the feature vectors, $\mathbf{d}^{(i)}$, $\forall i \in \mathcal{B}$, are used to update the anchor matrix, $\mathbf{D}$ (line 9). Similarly, a subset of randomly selected patches, $\mathcal{D} \subseteq \{1, \dots, N\}$, of size $N_{\mathcal{D}}$ is generated and used to construct the approximated anchor matrix, $\tilde{\mathbf{D}} \in \mathbb{R}^{l \times N_{\mathcal{D}}}$, such that its columns correspond to the feature vectors $\mathbf{d}^{(i)}$ where $i \in \mathcal{D}$ (lines 5 and 6). Further, a submatrix $\tilde{\mathbf{A}}$ is constructed from the columns of $\mathbf{A}$ that correspond to the patches in $\mathcal{D}$ (line 6). The cross-entropy loss of each patch $i \in \mathcal{B}$ in (3) can then be estimated by the following loss function:

$$\tilde{L}_{ce}^{(i)} = -\left(\tilde{\mathbf{A}}_i^T \log \tilde{\Omega}^{(i)} + (\mathbf{1} - \tilde{\mathbf{A}}_i^T) \log (\mathbf{1} - \tilde{\Omega}^{(i)})\right) \tag{7}$$
$$\text{where } \tilde{\Omega}^{(i)} = \sigma\left(\frac{\tilde{\mathbf{D}}^T \mathbf{d}^{(i)}}{s}\right)$$

and $\tilde{\mathbf{A}}_i$ is the ith row vector of $\tilde{\mathbf{A}}$. The above approximate cross-entropy loss and the regularization loss is evaluated at each iteration to update the parameters $\boldsymbol{\theta}$ of the DCSNN (lines 7-8). We utilize backpropagation to determine the first gradients of the approximate loss function. The anchor matrix $\mathbf{D}$ is updated and the process is repeated until $\boldsymbol{\theta}$ converges. Note that we assume that the anchor matrix, $\mathbf{D}$, is detached from the gradient calculation, i.e., $\frac{\partial}{\partial\theta}\mathbf{D}^T d = \mathbf{D}^T \frac{\partial d}{\partial \theta}$.

2.4. **A comparison between DCSNN and binary hashing methods.** Binary representations of images can significantly decrease the required storage memory, which is especially meaningful when managing a large image search database. They can, however, result in substantial information loss when the real-valued outputs of a CNN-based descriptor are approximated to binary values. The DCSNN utilizes a "loose" regularization term, thus would be expected not to lose much descriptive information of the SAR images. Nevertheless, it lacks the capability of using binary hashing to retrieve similar patches directly. For comparison with the DCSNN, we revisit existing binary hashing methods via DNNs for image retrieval (also referred to as *deep hashing neural networks*), and presents a technique to address this drawback in Section 3.1.

We consider three recent methods including the DPSH [23], DHN [50], and DHNN-L2 [24]. Firstly, the loss function of the DPSH [23] is defined as,

$$L = -\frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{A}_i^T \log \sigma(\mathbf{D}^T \mathbf{d}^{(i)}) + \lambda(\mathbf{b}^{(i)} - \mathbf{d}^{(i)})^2\right), \tag{8}$$

where $\mathbf{b}^{(i)}$ is the binary representation of $\mathbf{d}^{(i)}$. For the elements of the vectors, $\mathbf{b}_j^{(i)} = \text{sign}(\mathbf{d}_j^{(i)})$, $\forall j \in \{1, \dots, l\}$, where the sign$(\cdot)$ function maps an element of the feature vector to -1 or 1. A distinguishing factor of our approach relative to the DPSH is that the loss function of the DCSNN uses the similarity factor s and the loose regularization term (5).

The loss function of the DHN [50] is defined as,

$$L = -\frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{A}_i^T \log \sigma(\mathbf{D}^T \mathbf{d}^{(i)}) + \lambda\||\mathbf{d}^{(i)}| - \mathbf{1}\|_1\right) \tag{9}$$

where $\mathbf{1}$ is a vector of ones and $|\cdot|$ represents an element-wise absolute function. The authors also introduced a smooth surrogate of the regularization of (9):

$$L = -\frac{1}{N}\sum_{i=1}^{N}\mathbf{A}_i^T \log \sigma(\mathbf{D}^T \mathbf{d}^{(i)}) \tag{10}$$
$$+\lambda\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{l}\log\cosh(|\mathbf{d}_j^{(i)}| - 1).$$

Lastly, the DHNN-L2 [24] was particularly proposed for remote sensing image retrieval, with the loss function defined by

$$L = -\frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{A}_i^T \log\sigma\left(\frac{\mathbf{D}^T\mathbf{d}^{(i)}}{s}\right) + \lambda\|\mathbf{b}^{(i)} - \mathbf{d}^{(i)}\|_2^2\right). \tag{11}$$

The sole difference between the loss function of the DPSH (8) and that of the DHNN-L2 is the use of the similarity factor s. Comparative studies between the above binary hashing methods and the DCSNN are furnished in Section 4.

3. SAR image retrieval and registration with DCSNN for Positioning.

This section introduces a *product quantization* (PQ) technique and *asymmetric quantizer distance* (AQD) for fast and scalable retrieval of SAR image patches from the database. A reranking method that enhances the performance of image retrieval and registration is also described.

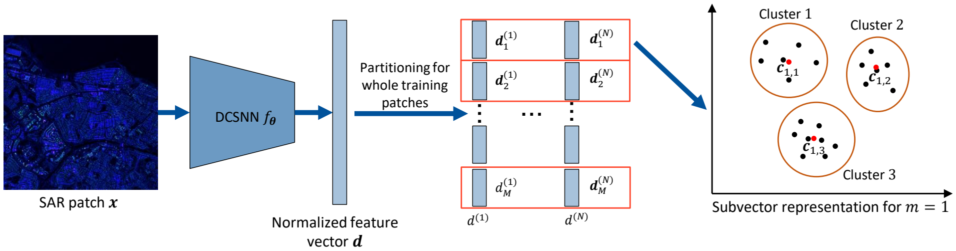


Figure 3. Overview of Product Quantization (PQ) with DCSNN

3.1. Approximate SAR retrieval via product quantization.

After the fine-tuned DCSNN generates feature vectors, their relative distances are computed and used to retrieve images from the database that are similar to a given inquiry image. Since the regularization loss term in (5) results in feature vectors whose norm values are approximately 1, we employ cosine similarity as a distance metric between the vectors. In particular, given an inquiry SAR image's feature vector, images stored in the database whose feature vectors produce high cosine similarity values are retrieved. Suppose $\mathbf{d}^{(i)}$ and $\mathbf{d}^{(j)}$ are feature vectors for images i and j, respectively, then the cosine similarity CS is defined as

$$CS(\mathbf{d}^{(i)}, \mathbf{d}^{(j)}) = \frac{\mathbf{d}^{(i)T}\mathbf{d}^{(j)}}{\|\mathbf{d}^{(i)}\|_2\|\mathbf{d}^{(j)}\|_2}. \tag{12}$$

Note that cosine similarity and Euclidean distance are proportional when the norms of the feature vectors are 1. For simplicity, below we assume that a given feature vector is already normalized, i.e., $\|\mathbf{d}^{(i)}\|_2 = \|\mathbf{d}^{(j)}\|_2 = 1$ and $CS(\mathbf{d}^{(i)}, \mathbf{d}^{(j)}) = \mathbf{d}^{(i)T}\mathbf{d}^{(j)}$.

Clearly, the computational cost of calculating distances is proportional to the number of images N and can be prohibitive when $N \gg 1$. To make real-time application of the proposed method feasible, we employ the PQ method in [17] to increase the scalability of computing (12). For each feature vector $\mathbf{d} \in \mathbb{R}^D$, take $M \in \mathbb{Z}_+$ subvectors of size $T \in \mathbb{Z}_+$ such that $l = MT$, i.e., $\mathbf{d} = [\mathbf{d}_1, \ldots, \mathbf{d}_M]$ and $\mathbf{d}_i \in \mathbb{R}^T$, $\forall i \in \{1, \ldots, M\}$. For example, if $\mathbf{d} = [1.1, 1.2, 1.3, 1.4]^T$ and $M = 2$, the subvectors are $\mathbf{d}_1 = [1.1, 1.2]^T$ and $\mathbf{d}_2 = [1.3, 1.4]$. In this case, the size of subvector $T = 2$. Then, for each subvector, K clusters with corresponding representative vectors are generated using the k-means algorithm [29]. Given the mth subvectors, $\mathbf{d}_m^{(i)}$, $i \in \{1, \dots, N\}$, denote the representative vectors by $\mathbf{c}_{m,k} \in \mathbb{R}^T$, $\forall k = \{1, \dots, K\}$, which are given by the mean of the subvectors in their respective clusters. Next, each subvector is assigned to the cluster whose representative vector is closest. To represent this, define a binary assignment vector, $\mathbf{b}_m^{(i)} \in \{0,1\}^K$, corresponding to feature vector $\mathbf{d}^{(i)}$, such that its jth element, $\mathbf{b}_{m,j}^{(i)}$, is given by

$$\mathbf{b}_{m,j}^{(i)} = \begin{cases} 1, & \text{if } j = \text{argmax}_k CS(\mathbf{d}_m^{(i)}, \mathbf{c}_{m,k}), \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

for all $j \in \{1, \dots, K\}$. As a result, instead of $\mathbf{d}^{(i)}$, the binary assignment $\mathbf{b}_m^{(i)}$ and the representative vectors $\mathbf{c}_{m,k}$ for all m and k are stored in the database. The sequence of the described PQ procedure is depicted in Fig. 3.

Feature vector $\mathbf{d}^{(i)}$ is then estimated using the binary assignment and its clusters' representative vectors as follows. Define a matrix whose columns are given by the representative vectors, i.e., $\mathbf{C}_m \in \mathbb{R}^{T \times K} = [\mathbf{c}_{m,1}, \dots, \mathbf{c}_{m,K}]$. For a SAR image patch $i \in \{1, \dots, N\}$ in the database, the subvector $\mathbf{d}_m^{(i)}$ of $\mathbf{d}^{(i)}$ can therefore be approximated by $\mathbf{C}_m \mathbf{b}_m^{(i)}$. Consequently, feature vector $\mathbf{d}^{(i)}$ can be estimated by concatenating the approximated subvectors:

$$\mathbf{d}^{(i)} \approx \hat{\mathbf{d}}^{(i)} = [\mathbf{C}_1 \mathbf{b}_1^{(i)}, \dots, \mathbf{C}_M \mathbf{b}_M^{(i)}]. \tag{14}$$

Finally, given an inquiry image with a feature vector $\mathbf{d}$, we employ AQD [6] to calculate the approximate cosine similarity distances $CS(\mathbf{d}, \hat{\mathbf{d}}^{(i)})$, $\forall i \in \{1, \dots, N\}$, where

$$CS(\mathbf{d}, \mathbf{d}^{(i)}) \approx CS(\mathbf{d}, \hat{\mathbf{d}}^{(i)}) = \sum_{m=1}^{M} \mathbf{d}_m^T \mathbf{C}_m \mathbf{b}_m^{(i)}. \tag{15}$$

It is important to emphasize that the binary assignment vectors and representative vectors can be calculated offline and stored in the database. Moreover, due to the fact that the computational cost of AQD depends on T and K, but not N, the distance calculations become much more scalable. Note that in AQD the distance measure used is asymmetric, i.e., $CS(\mathbf{d}, \hat{\mathbf{d}}^{(i)}) \neq CS(\hat{\mathbf{d}}^{(i)}, \mathbf{d})$ [6]. It is not necessary to compute the binary assignment vector of an inquiry image's feature vector, resulting in reduced computational costs.

3.2. **Reranking retrieved SAR images and positioning based on image registration.** For a given inquiry image, patches with high AQD distances are retrieved from the database. To further enhance the accuracy of retrieval, they are reranked using conventional image feature detection methods and image matching techniques. The process for reranking is two-fold. First, *points of interest* in SAR images are identified by using the feature detecting methods SIFT [27, 28] and SAR-SIFT [10]. Second, we perform *image matching* by comparing the inquiry image with retrieved images based on location differences of the points of interest and their descriptions.

A popular feature detecting method for extracting points of interest, also called *keypoints*, is SIFT [27, 28]. The underlying procedure of SIFT relies on using the Difference of Gaussian (DoG) as an approximation of the Laplacian of Gaussian (LoG) to construct an image "pyramid". This is then used to extract scale invariant characteristics of the image. The resulting output consists of keypoints and their

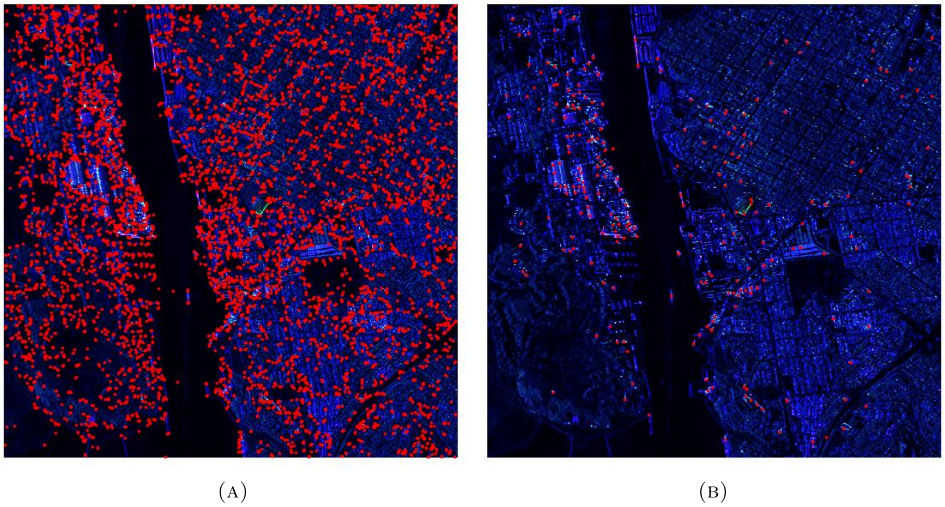


Figure 4. Comparison between keypoints generated by (a) SAR-SIFT and (b) SIFT

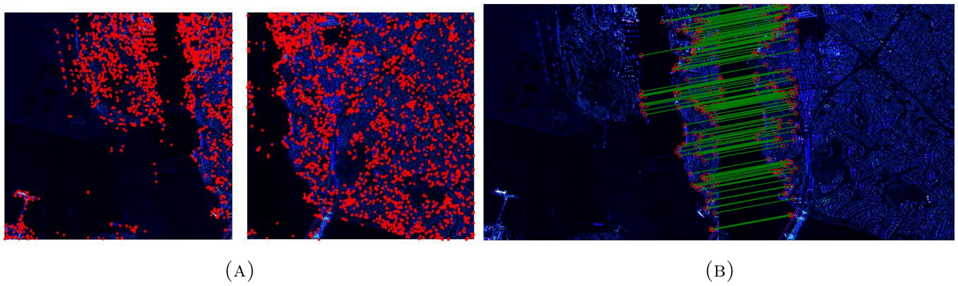


Figure 5. (a) SAR-SIFT based keypoints on two adjacent SAR patches (b) image matching of the keypoints via RANSAC

local descriptors. A keypoint provides location information about the point of interest, while the local descriptor uses a square neighborhood/region to define histograms of gradient orientations weighted by gradient magnitudes. The local descriptors are obtained by normalizing and concatenating the histograms for each scale (see [27, 28] for details).

Although SIFT has been widely used in numerous applications including image indexing, image retrieval and video tracking, it does not work well on SAR images. Because it relies on square neighborhoods of pixels to calculate the gradients, and SAR images are prone to contain high levels of speckle noise, it prevents SIFT from measuring the gradients accurately. There have been numerous attempts to modify SIFT for processing SAR and remote sensing optical images [26, 41, 44, 10, 43]. These improvements include adapting a prefilter [26] or denoising of images [41]. Additional information such as digital elevation model (DEM) or orbit information can be used to complement SIFT-based local descriptors on SAR images [44]. A

Table 1. Descriptions of the PolSAR map from UAVSAR [1]

Region (Dataset Name)	Usage	Acquired Date	Pixel Size (Height×Width)	#Patches
Hayward Fault Zone	Training	Oct 9, 2018	23506 × 3300	6440
	Test	May 30, 2019	23476 × 3300	6412
Yukon–Kuskokwim Delta	Training	Aug 28, 2018	19066 × 3300	5180
	Test	Sep 17, 2019	19148 × 3300	5208

popular method that mitigates this drawback is known as SAR-SIFT [10]. To detect keypoints in SAR images, SAR-SIFT employs a gradient definition based on a multiscale Harris function and *gradient by ratio*. The authors suggested that using a circular descriptor (rather than a square) to generate histograms is a better suited to obtain robust local descriptors in SAR images. By comparison, Fig. 4 shows that SAR-SIFT and SIFT generate vastly different keypoints when applied to the same SAR image. Observe that keypoints from SAR-SIFT better distinguish boundaries of the city terrain of a SAR image. It is important to note that the keypoints and local descriptors can be precomputed using SAR-SIFT or SIFT for all images in the database. In Section 4, we compare SAR and SAR-SIFT for reranking retrieved SAR image patches.

Next, RANSAC [14] is utilized for matching retrieved images to an inquiry image. RANSAC determines whether keypoints are inliers or outliers (for its consensus) in order to find the best possible affine (or other) transformation for global deformations between two images. Past research has shown that global relationships between two SAR images can be successfully described by affine transformation because SAR images are roughly “flat scenes” [51, 10]. For the SAR images in Fig. 5a, the red dots in Fig. 5b represent locations of inliers, whereas green lines indicate the affine transformations between images. The *score* for each retrieved image is then defined as the number of inlier keypoints. The retrieved SAR images are reranked according to their scores.

After reranking, a retrieved image with the highest score is selected for registration and positioning. As previously indicted, it is assumed that the location coordinates of images in the database are stored and can be used for navigation. Given the location coordinates of the top reranked image, say x_1 and y_1, we estimate the coordinates of an inquiry image, x_2 and y_2, as,

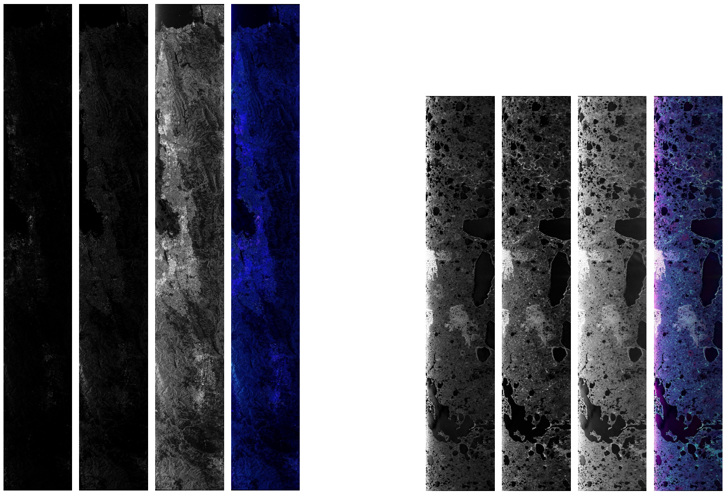
$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} + \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}. \tag{16}$$

The coefficients a and b are obtained after completing image matching via RANSAC. Experimental results for the proposed methodology are presented next.

4. Computational experiment.

4.1. Experiment settings.

The UAVSAR dataset [1] was used to validate the performance of the DCSNN and positioning estimations. UAVSAR comprises the PolSAR and InSAR dataset that are used for studying dynamic changes on the Earth’s surface. The datasets provide location coordinates acquired from real-time GPS. We use L-band PolSAR data because it is more conducive to navigation tasks



(a) (b)

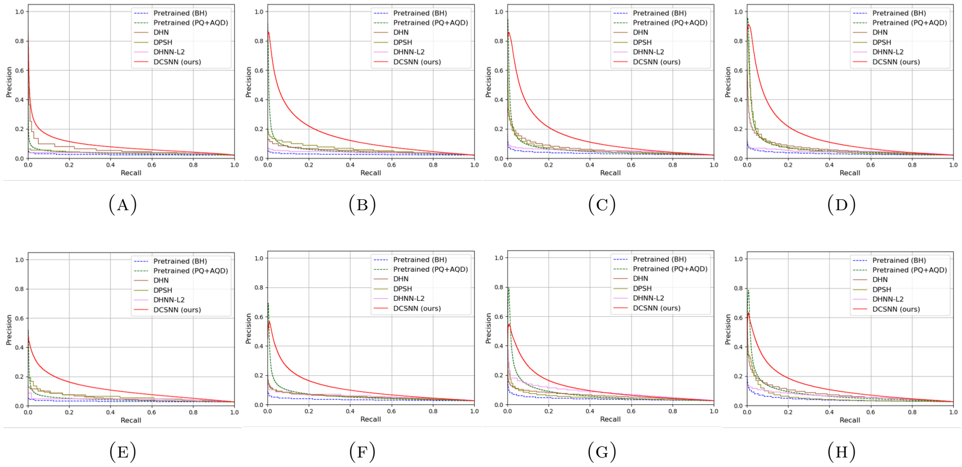
Figure 6. PolSAR data from UAVSAR dataset for our experiments. *(From left)* VVVV(R), HVHV(G), HHHH(B) channels, and total RGB image. Best viewed in color. (a) PolSAR map for Hayward Fault Zone in California, US. (b) PolSAR map for Yukon–Kuskokwim Delta in Alaska, US.

because PolSAR consists of the amplitude and/or phase of backscattered signals that can be collected during a single flight. Multi-look cross (MLC) products of PolSAR were considered, and VVVV, HVHV, and HHHH SAR images of MLC were used. These correspond to red, green, and blue channels of the total SAR map image, respectively, as depicted in Fig. 6.

We examined two geographically distinct regions on the Earth's surface: the Hayward Fault Zone in California, US, shown in Fig. 6a; and the Yukon–Kuskokwim Delta in Alaska, US, shown in Fig. 6b. The former contains many man-made structures; whereas the latter consists of only natural formations. Two SAR maps were prepared for each region. The first map was for training and was preprocessed to extract image patches and to construct the associated graph and adjacency matrix $\mathbf{A}$. After fine-tuning the DCSNN on the training dataset, binary assignment vectors and representative vectors of PQ were generated and stored in an image search database. The second map served for testing purpose and was therefore used to extract inquiry SAR patches. Details of the maps used in our experiments are illustrated in Table 1. Both SIFT and SAR-SIFT procedures were implemented on the image patches in the database and the resulting output data was stored.

All experiments were conducted in Python on a Linux Ubuntu 16.04 operating system. The DCSNN models were implemented based on PyTorch [36]. The DCSNN models were run on GeForce GTX 1080Ti with 11GB RAM, while implementations of PQ, AQD, SIFT/SAR-SIFT, and RANSAC were run on Intel I7-7700K CPU with 16GB RAM.

4.2. **DCSNN configuration.** Two types of CNN-based backbone architectures were used to construct the DCSNNs. The first was AlexNet [21] – considered a



(A) (B) (C) (D)

(E) (F) (G) (H)

Figure 7. Precision recall curves on Hayward Fault Zone (*Top rows*) and Yukon–Kuskokwim Delta (*Bottom rows*) PolSAR maps. From left column, the feature length is 24, 48, 96, and 120. AlexNet is used as a backbone.

simpler model – which has five convolutional layers followed by two fully connected (FC) layers. It contains approximately 61 millions trainable parameters. The second was VGG-11 [40], which has 8 convolutional layers followed by 3 FC layers. It contains approximately 134 million trainable parameters and, consequently, requires more time to train and make predictions. For both architectures we used transfer learning from the pretrained convolutional layers on ImageNet dataset [11]. The ImageNet dataset contains 1000 classes and was used for classification purpose, thus only the convolutional layers from both pretrained architectures were utilized. Two FC layers that output $l/2$ and l activations, respectively, were added after the last convolutional layer output.

The performances of the pretrained models with and without fine-tuning were also examined. We employed SGD with a weight decay of 1E$-$4 for fine-tuning the DCSNN. The learning rate was initialized at 0.01 and reduced by a factor of 10 at 150th epoch for both SAR datasets and both architectures. The total number of epochs was set to 200. The minibatch size, $N_{\mathcal{B}}$, was set to 128, and the stochastic anchor matrix size, $N_{\mathcal{D}}$, was set to 512 for all experiments. All generated image patches had a size of 600×600 pixels and were resized to 224×224. Each patch covers a regional area of approximately 16km^2.

Data augmentation was used on the training patches to mitigate the effects of speckles noise and prevent overfitting. Gaussian blur was applied on 80% of the training patches (randomly selected) with radius ranged from 0.5 to 2.0. The regularization coefficient λ was set to 0.1 after hyperparameter tuning considering $\lambda \in \{100, 10, 1, 0.1, 0.01, 0.001\}$. Further, for the DCSNNs, the similarity factor, s, was set to 0.5 after hyperparameter tuning considering $s \in \{0.01, 0.1, 0.25, 0.5, 1.0, 10.0\}$. For other binary hashing baselines such as the DPSH, DHN, DHNN-L2, coefficients recommended in the previous studies [24, 23, 50] were used.

Table 2. Mean average precision (mAP) results of DCSNN and binary hashing methods before reranking on Hayward Fault Zone PolSAR map.

Methods	Feature length l	AlexNet [21]	VGG-11 [40]
Pretrained (BH)	$l = 24$	0.0478	0.0313
	$l = 48$	0.0761	0.0522
	$l = 96$	0.1547	0.1200
	$l = 120$	0.1566	0.1219
Pretrained (PQ+AQD)	$l = 24$	0.2332	0.1706
	$l = 48$	0.3208	0.2055
	$l = 96$	0.3231	0.2676
	$l = 120$	0.3856	0.3161
DHN [50]	$l = 24$	0.1182	0.1255
	$l = 48$	0.1642	0.1693
	$l = 96$	0.2274	0.2153
	$l = 120$	0.3202	0.2449
DPSH [23]	$l = 24$	0.0895	0.1220
	$l = 48$	0.2825	0.2632
	$l = 96$	0.4545	0.4005
	$l = 120$	0.5213	0.4334
DHNN-L2 [24]	$l = 24$	0.0451	0.1147
	$l = 48$	0.0683	0.1291
	$l = 96$	0.2044	0.1329
	$l = 120$	0.2190	0.1304
DCSNN (ours)	$l = 24$	**0.2519**	**0.4889**
	$l = 48$	**0.6145**	**0.6301**
	$l = 96$	**0.6481**	**0.5783**
	$l = 120$	$\underline{\mathbf{0.6813}}$	**0.5819**

The DCSNN was compared against the baselines using the mean average precision (mAP) as well as the precision-recall curve. The mAP metric, which was widely adopted in past studies [22, 50, 25, 38, 19], is formally defined as,

$$\text{mAP} = \frac{1}{|\mathcal{Q}|}\sum_{i=1}^{|\mathcal{Q}|}\frac{1}{R}\sum_{j=1}^{R}\text{precision}(\mathcal{R}_i^j), \tag{17}$$

where $\mathcal{Q}$ is inquiry image set and R is the number of retrieved image patches from the database for $q_i \in \mathcal{Q}$. $\mathcal{R}_i^j$ is a ranked patch set containing the top j ranked retrieved patches for q_i, which is determined using AQD. The term precision$(\mathcal{R}_i^j)$ is the precision value representing the ratio of relevant image patches among the j retrieved patches. Note that $\mathcal{R}_i^j$ can be re-ordered after the reranking procedure introduced in Section 3.2 is applied.

Table 3. mAP results of the DCSNN and binary hashing methods before reranking on Yukon–Kuskokwim Delta PolSAR map.

Methods	Feature length l	AlexNet [21]	VGG-11 [40]
Pretrained (BH)	$l = 24$	0.0566	0.0396
	$l = 48$	0.1048	0.0550
	$l = 96$	0.1927	0.1227
	$l = 120$	0.2196	0.1174
Pretrained (PQ+AQD)	$l = 24$	0.2684	0.1736
	$l = 48$	0.3580	0.2091
	$l = 96$	0.3718	0.2579
	$l = 120$	0.4184	0.2690
DHN [50]	$l = 24$	0.1219	0.1610
	$l = 48$	0.2072	0.1910
	$l = 96$	0.2715	0.2447
	$l = 120$	0.3627	0.2239
DPSH [23]	$l = 24$	0.1347	0.1170
	$l = 48$	0.2371	0.2516
	$l = 96$	0.3649	0.3281
	$l = 120$	0.4098	0.3331
DHNN-L2 [24]	$l = 24$	0.0621	0.1132
	$l = 48$	0.1556	0.1812
	$l = 96$	0.2916	0.3166
	$l = 120$	0.3227	0.2822
DCSNN (ours)	$l = 24$	**0.4393**	**0.4324**
	$l = 48$	**0.5424**	**0.5196**
	$l = 96$	**0.5734**	**0.4913**
	$l = 120$	**0.5996**	**0.4831**

Table 4. mAP values before and after reranking with SAR-SIFT or SIFT on Hayward Fault Zone PolSAR map.

CNN backbone	Feature length l	Before reranking	After reranking (SAR-SIFT/SIFT)
AlexNet [21]	$l = 24$	0.2519	**0.4074**/0.3533
	$l = 48$	0.6145	**0.7394**/0.6850
	$l = 96$	0.6481	**0.7548**/0.6998
	$l = 120$	0.6813	**0.7760**/0.7252
VGG-11 [40]	$l = 24$	0.4889	**0.6512**/0.5813
	$l = 48$	0.6301	**0.7540**/0.6923
	$l = 96$	0.5783	**0.6799**/0.6231
	$l = 120$	0.5819	**0.6787**/0.6216

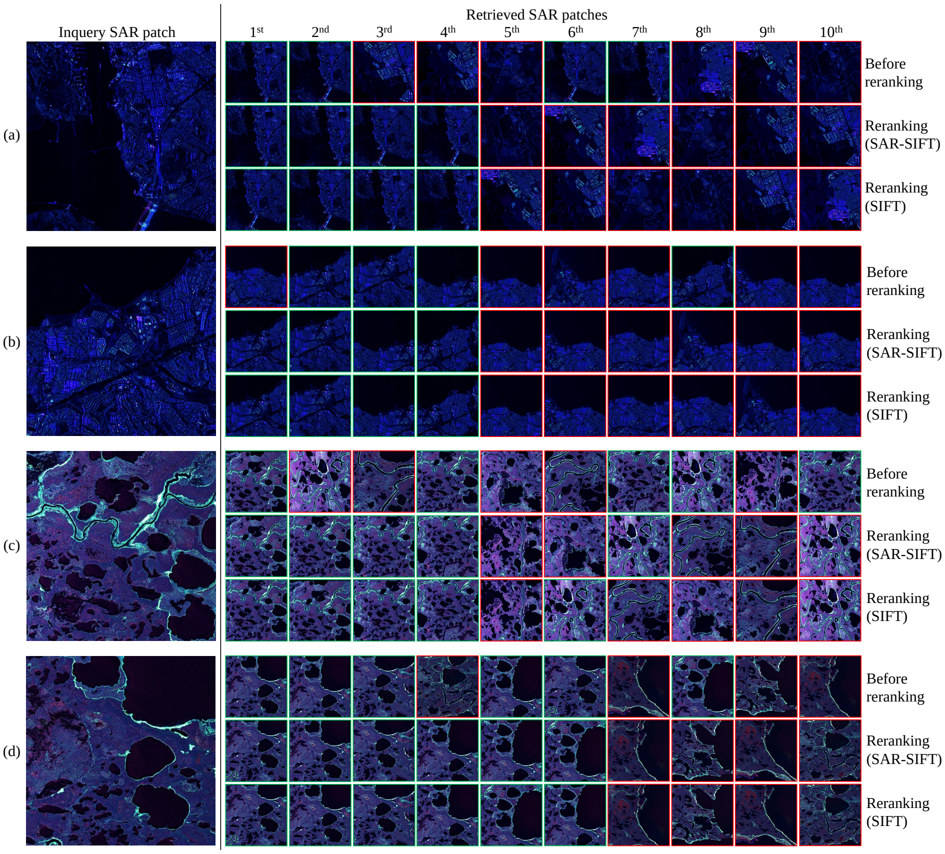


FIGURE 8. Examples of the retrieved SAR patches before and after reranking processes. First column represents examples of inquiry SAR patches. The first two rows ((a) and (b)) are from Hayward Fault Zone PolSAR and the later two rows ((c) and (d)) are from Yukon-Kuskokwim Delta PolSAR data. Having green box indicates it is correctly retrieved, whereas having red box indicates that it is incorrectly retrieved.

4.3. **Performance results.** Performances of the DCSNN with the two mentioned CNN architectures were investigated. We considered the DPSH, DHN and DHNN-L2 described in Subsection 2.4 as baselines. The baselines used the Hamming distance (instead of AQD) as a measure for comparison and retrieval because they generate binary descriptor vectors. To compare computational performances, we varied the feature length l to be $24, 48, 96, 120$, and measured the corresponding mAP values. The number of retrieved images from the database, R, was set to 10 for all experiments. That is, for each inquiry image from the test SAR map we retrieve the top 10 images. Although our choice of R is relatively smaller than those in previous studies [33], it is more conducive to simulating real-time applications when computational time is limited. Also, we used $M = 12$ and $K = 2^7$ for PQ.

Tables 2 and 3 illustrate the mAP values obtained from the Hayward Fault Zone and Yukon-Kuskokwim Delta PolSAR datasets, respectively. The values in bold

Table 5. mAP values of the DCSNN before and after reranking with SAR-SIFT or SIFT on Yukon–Kuskokwim Delta PolSAR map.

CNN backbone	Feature length l	Before reranking	After reranking (SAR-SIFT/SIFT)
AlexNet [21]	$l = 24$	0.4393	0.5831/**0.5965**
	$l = 48$	0.5424	0.6591/**0.6693**
	$l = 96$	0.5734	0.6782/**0.6888**
	$l = 120$	0.5996	0.7021/**<u>0.7123</u>**
VGG-11 [40]	$l = 24$	0.4324	0.5909/**0.6030**
	$l = 48$	0.5196	0.6418/**0.6521**
	$l = 96$	0.4913	0.5939/**0.6036**
	$l = 120$	0.4831	0.5840/**0.5940**

Table 6. Positioning accuracy examples.

Inquiry SAR Patch	Actual Coordinates [deg]	Estimated Coordinates [deg]	Error [m]
Fig.8(a)	38.0625, -122.2733	38.0625, -122.2734	5.7288
Fig.8(b)	37.9836, -122.3599	37.9836, -122.3600	5.7347
Fig.8(c)	61.0926, -164.1878	61.0926, -164.1879	4.2529
Fig.8(d)	61.0808, -164.1208	61.0808, -164.1208	5.0970

Table 7. Mean and standard deviation of positioning error results.

Data Name	Success Cases Ratio [%]	Distance Error [m]
Hayward Fault Zone	98.50	4.9635±0.1755
Yukon–Kuskokwim Delta	97.70	4.9522±0.4038

indicate the best results for each feature length, and the underlined value indicates the best result for each SAR dataset. “Pretrained (BH)” in the tables represents the results from the pretrained model on ImageNet dataset without performing fine-tuning and using hamming distance (H) with binary representations (B) of the output features. Similarly, “Pretrained (PQ+AQD)” represents the results obtained from the pretrained model that uses PQ and AQD to retrieve patches from the database. By comparison, the pretrained (PQ+AQD) produced better results than pretrained (BH) over all the configurations. This can be attributed to the fact that using a binary representation of a real-valued feature vector loses a significant portion of descriptive information about a SAR image. Among binary hashing methods, fine-tuning with the DPSH produced superior results on both datasets. Additionally, all binary hashing methods are better than the pretrained model, Pretrained (BH), which suggests that the fine-tuning methods with binary hashing can improve the performance of SAR image retrieval tasks. Nevertheless, the proposed fine-tuned DCSNN outperformed all the binary baselines by a significant margin. We also note that the mAP values of the DCSNN were significantly larger than those of Pretrained (PQ+AQD).

It can also be seen that performances generally improved as the feature length l increased. Despite the fact that the VGG-11 architecture is more advanced and contains more parameters than AlexNet, observe that the DCSNN model with AlexNet gave superior results as l increased. This suggests that the more sophisticated learning scheme embedded in VGG-11 was outweighed by the increased computational expense. Therefore, the DCSNN with AlexNet was better suited in this setting.

Precision-recall curves of the AlexNet backbone in Fig. 7 clearly demonstrate that the accuracy of the DCSNN is superior to the other baselines. This is predominantly attributed to the regularization term of the DCSNN, which does not sacrifice descriptive information of SAR images and enables efficient and concise feature vectors representations.

Computational times for a single forward pass for prediction of the DCSNN with the AlexNet and VGG-11 architectures on the Hayward Fault zone dataset with 6412 patch images were on average 0.6407ms and 2.1705ms, respectively. We note that single forward passes can be done in parallel for multiple inquiry patches by utilizing a GPU.

4.4. **Performance results on reranking.** We next examine performance enhancements from the reranking procedure by comparing mAP values before and after its implementation. Both SIFT and SAR-SIFT methods were used for reranking. The mAP results are illustrated in Tables 4 and 5 for Hayward Fault Zone PolSAR and Yukon-Kuskokwim Delta PolSAR data, respectively. As shown, reranking via either SIFT or SAR-SIFT improved the outcomes overall. When compared side-by-side, the SAR-SIFT-based reranking outperformed the SIFT-based reranking for the Hayward Fault Zone dataset, whereas the SIFT-based reranking gave slightly better mAP values than SAR-SIFT-based reranking for the Yukon-Kuskokwim Delta dataset. This stems from the fact that SAR-SIFT reduces the effects of speckle noise on small local features in the city-region SAR images. Fig. 8 shows several examples of inquiry images and corresponding retrieved images. It can be seen that the order of retrieved patches tends to be corrected after reranking is employed, thereby leading to higher mAP values.

4.5. **Performance results on location positioning.** The performance of the developed approach was explored in the context of navigation. Per Section 3.2, the affine transformation associated with RANSAC (see (16)) was used to estimate the coordinates of a given inquiry SAR image. Table 6 furnishes the estimated coordinates produced by our approach for the inquiry SAR image examples shown in Figure 8. A DCSNN with AlexNet architecture, feature length of $l = 120$, and the SAR-SIFT methods for generating keypoints, were used for this experiment. The "Actual Coordinates" and "Estimated Coordinates" in Table 6 represent the latitude and longitude pairs of ground truth and prediction, respectively. Errors were computed using the geodesic distance between actual and estimate coordinates. Observe that the errors are quite small, the largest of which was only $5.7m$. This suggests that the proposed technique can be effective for navigational tasks.

Table 7 shows the average and standard deviation of distance errors over all the SAR inquiry images. The "Success Cases Ratio" furnishes the ratio of the

SAR image patches that were processed to estimate the coordinates successfully, which were 97.7% and 98.5% for the Yukon-Kiskokwim Delta and Hayward Fault Zone, respectively. Average distance errors and standard deviations (Std. Dev.) of distance errors were measured only on success cases. We note that the SAR images for which coordinates were not estimated successfully failed to generate keypoints via SAR-SIFT, which is attributed to the images not containing meaningful pixel variations for calculating gradients of pixels. For example, if a SAR image only contains a sea or lake region, there may be no keypoints other than small tides. Consequently, the affine transformation of RANSAC cannot be applied effectively for such images. To address this, it may be possible to consider majority voting or consensus of coordinate estimations and incidence angles of patches across the swath direction of the SAR map. For example, the SAR maps in our experiments had the swath width of 3300 pixels and patches of 600 × 600 pixels were extracted with the stride of 100 pixels. Thus, for each swath, we had 28 patches that could have possibly been used to attain better coordinate estimates – a procedure that we reserve for future work. Nevertheless, based on the small average distance errors and standard deviations shown in Table 7, it can be concluded that the proposed approach for position identification was successful on the adopted datasets.

5. Conclusion.

This work introduced a navigational approach that relies on SAR image processing via deep neural networks to enable effective image matching, retrieval, and registration. We developed a deep neural network-based SAR descriptor, the DCSNN, and a fine-tuning procedure that was used to describe a SAR image with a "simple" feature vector. By using asymmetric quadratic distance as a scalable metric for comparing feature vectors, images from a database that are similar to an inquiry image can be efficiently retrieved. It was also demonstrated that reranking via SIFT or SAR-SIFT can increase the performance of the image retrieval process. Affine transformations used for keypoints matching via RANSAC, which were obtained as a biproduct, were used for image registration to estimate location coordinates of inquiry SAR images. Finally, we demonstrated that our approach was highly effective on PolSAR datasets.

Image matching techniques such as SIFT and SAR-SIFT can fail to generate keypoints in cases when, for example, a retrieved SAR image patch does not contain a meaningful scene, thus lacking coordinates information. In future studies, we will investigate navigation methodologies that overcome this drawback. These may include using deep neural network-based approach for reranking rather than partially relying on the classical image matching and registration techniques that can be learned in an end-to-end manner, and using multiple patches in the same swath direction to generate consensus on estimations. Moreover, we will consider different scales of SAR patches by constructing image pyramid inputs to the deep neural network.

Acknowledgments. This work was supported by the U.S. Air Force grant FA8651-08-D-0108/048, and by U.S. Air Force grant FA9453-20-1-0007.

REFERENCES

[1] Dataset: UAVSAR POLSAR, NASA 2020. Retrieved from ASF DAAC, 2020.

[2] A. Babenko, A. Slesarev, A. Chigorin and V. Lempitsky, Neural codes for image retrieval, in *European Conference on Computer Vision*, Springer, **2014** (2014), 584–599.

[3] G. Balamurugan, J. Valarmathi and V. Naidu, Survey on UAV navigation in GPS denied environments, in *2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES)*, IEEE, 2016, 198–204.

[4] J. Bhatti and T. E. Humphreys, Hostile control of ships via false GPS signals: Demonstration and detection, *NAVIGATION: Journal of the Institute of Navigation*, **64** (2017), 51–66.

[5] F. Caballero, L. Merino, J. Ferruz and A. Ollero, Vision-based odometry and SLAM for medium and high altitude flying UAVs, *Journal of Intelligent and Robotic Systems*, **54** (2009), 137–161.

[6] Y. Cao, M. Long, J. Wang, H. Zhu and Q. Wen, *Deep Quantization Network for Efficient Image Retrieval*, in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[7] A. Cesetti, E. Frontoni, A. Mancini, P. Zingaretti and S. Longhi, A vision-based guidance system for UAV navigation and safe landing using natural landmarks, *Journal of Intelligent and Robotic Systems*, **57** (2010), 233–257.

[8] G. Conte and P. Doherty, Vision-based unmanned aerial vehicle navigation using georeferenced information, *EURASIP Journal on Advances in Signal Processing*, **2009** (2009), Article number: 387308.

[9] 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)*, vol. 1, IEEE, 2005, 886–893.

[10] F. Dellinger, J. Delon, Y. Gousseau, J. Michel and F. Tupin, SAR-SIFT: A SIFT-like algorithm for SAR images, *IEEE Transactions on Geoscience and Remote Sensing*, **53** (2015), 453–466.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Ieee, 2009, 248–255.

[12] L. R. Dice, Measures of the amount of ecologic association between species, *Ecology*, **26** (1945), 297–302.

[13] J. Duchi, E. Hazan and Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization, *Journal of Machine Learning Research*, **12** (2011), 2121–2159.

[14] M. A. Fischler and R. C. Bolles, Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, *Communications of the ACM*, **24** (1981), 381–395.

[15] A. Gordo, J. Almazan, J. Revaud and D. Larlus, End-to-end learning of deep visual representations for image retrieval, *International Journal of Computer Vision*, **124** (2017), 237–254.

[16] A. Grant, P. Williams, N. Ward and S. Basker, GPS jamming and the impact on maritime navigation, *The Journal of Navigation*, **62** (2009), 173–187.

[17] H. Jegou, M. Douze and C. Schmid, Product quantization for nearest neighbor search, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **33** (2011), 117–128.

[18] M. Kaiser, N. Gans and W. Dixon, Vision-based estimation for guidance, navigation, and control of an aerial vehicle, *IEEE Transactions on Aerospace and Electronic Systems*, **46** (2010), 1064–1077.

[19] W.-C. Kang, W.-J. Li and Z.-H. Zhou, *Column Sampling Based Discrete Supervised Hashing*, Thirtieth AAAI Conference on Artificial Intelligence, 2016.

[20] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv preprint, `arXiv:1412.6980`.

[21] A. Krizhevsky, I. Sutskever and G. E. Hinton, Imagenet classification with deep convolutional neural networks, in *Communications of the ACM*, **60** (2017).

[22] P. Li and P. Ren, Partial randomness hashing for large-scale remote sensing image retrieval, *IEEE Geoscience and Remote Sensing Letters*, **14** (2017), 464–468.

[23] W.-J. Li, S. Wang and W.-C. Kang, Feature learning based deep supervised hashing with pairwise labels, arXiv preprint, `arXiv:1511.03855`.

[24] Y. Li, Y. Zhang, X. Huang, H. Zhu and J. Ma, Large-scale remote sensing image retrieval by deep hashing neural networks, *IEEE Transactions on Geoscience and Remote Sensing*, **56** (2017), 950–965.

[25] H. Liu, R. Wang, S. Shan and X. Chen, Deep supervised hashing for fast image retrieval, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, 2064–2072.

[26] J.-Z. Liu and X.-C. Yu, Research on SAR image matching technology based on SIFT, ISPRS08, B1.

[27] D. G. Lowe, Object recognition from local scale-invariant features, in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, Ieee, 1999, 1150–1157.

[28] D. G. Lowe, Distinctive image features from scale-invariant keypoints, *International Journal of Computer Vision*, **60** (2004), 91–110.

[29] J. MacQueen et al., Some methods for classification and analysis of multivariate observations, in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, Oakland, CA, USA, 1967, 281–297.

[30] A. Nemra and N. Aouf, Robust INS/GPS sensor fusion for UAV localization using SDRE nonlinear filtering, *IEEE Sensors Journal*, **10** (2010), 789–798.

[31] D. Nistér, O. Naroditsky and J. Bergen, Visual odometry, in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 1, Ieee, 2004, I–I.

[32] D. O. Nitti, F. Bovenga, M. T. Chiaradia, M. Greco and G. Pinelli, Feasibility of using synthetic aperture radar to aid UAV navigation, *Sensors*, **15** (2015), 18334–18359.

[33] H. Noh, A. Araujo, J. Sim, T. Weyand and B. Han, Large-scale image retrieval with attentive deep local features, in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, 3456–3465.

[34] C. Oliver and S. Quegan, *Understanding Synthetic Aperture Radar Images*, SciTech Publishing, 2004.

[35] S. Park, S. H. Jung and P. M. Pardalos, Combining stochastic adaptive cubic regularization with negative curvature for nonconvex optimization, *Journal of Optimization Theory and Applications*, **184** (2020), 953–971.

[36] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga et al., Pytorch: An imperative style, high-performance deep learning library, in *Advances in Neural Information Processing Systems*, 2019, 8024–8035.

[37] M. Shan, F. Wang, F. Lin, Z. Gao, Y. Z. Tang and B. M. Chen, Google map aided visual navigation for UAVs in GPS-denied environment, in *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, IEEE, 2015, 114–119.

[38] F. Shen, C. Shen, W. Liu and H. Tao Shen, Supervised discrete hashing, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, 37–45.

[39] D.-G. Sim, R.-H. Park, R.-C. Kim, S. U. Lee and I.-C. Kim, Integrated position estimation using aerial image sequences, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24** (2002), 1–18.

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

[41] S. Suri, P. Schwind, P. Reinartz and J. Uhl, Combining mutual information and scale invariant feature transform for fast and robust multisensor SAR image registration, in *75th Annual ASPRS Conference*, 2009.

[42] G. Tolias, R. Sicre and H. Jégou, Particular object retrieval with integral max-pooling of CNN activations, arXiv preprint, `arXiv:1511.05879`.

[43] B. Wang, J. Zhang, L. Lu, G. Huang and Z. Zhao, A uniform SIFT-like algorithm for SAR image registration, *IEEE Geoscience and Remote Sensing Letters*, **12** (2015), 1426–1430.

[44] B. Wessel, M. Huber and A. Roth, Registration of near real-time SAR images by image-to-image matching, in *Proc. Photogramm. Image Anal.*, 2007, 179.

[45] P. Williams and M. Crump, All-source navigation for enhancing UAV operations in GPS-denied environments, in *Proceedings of the 28th International Congress of the Aeronautical Sciences*, 2012.

[46] K. M. Yi, E. Trulls, V. Lepetit and P. Fua, LIFT: Learned invariant feature transform, in *European Conference on Computer Vision*, Springer, **2016** (2016), 467–483.

[47] J. Yue-Hei Ng, F. Yang and L. S. Davis, Exploiting local features from deep networks for image retrieval, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2015, 53–61.

[48] S. Zhao, F. Lin, K. Peng, B. Chen and T. Lee, Homography-based vision-aided inertial navigation of UAVs in unknown environments, in *AIAA Guidance, Navigation, and Control Conference*, 2012, 5033.

[49] L. Zheng, Y. Yang and Q. Tian, SIFT meets CNN: A decade survey of instance retrieval, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **40** (2018), 1224–1244.

[50] H. Zhu, M. Long, J. Wang and Y. Cao, Deep hashing network for efficient similarity retrieval, in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[51] B. Zitova and J. Flusser, Image registration methods: A survey, *Image and Vision Computing*, **21** (2003), 977–1000.

Received September 2020; revised November 2020.

E-mail address: seonhopark@ufl.edu
E-mail address: ryszmw@miamioh.edu
E-mail address: kaitlin.fair@us.af.mil
E-mail address: pardalos@ufl.edu