

## FEATURE MATCHING ENHANCEMENT USING THE GRAPH NEURAL NETWORK (GNN-RANSAC)

Elashry A., C. Toth

Dept. of Civil, Environmental and Geodetic Engineering, The Ohio State University, 470 Hitchcock Hall, 2070 Neil Avenue  
Columbus, OH 43210, USA - (Elashry.1, toth.2)@osu.edu

**KEY WORDS:** Feature matching, Outlier rejection, Graph Neural Network, RANSAC.

### ABSTRACT:

Improving the performance of feature matching plays a key role in computer vision and photogrammetry applications, such as fast image recognition, Structure from Motion (SfM), aerial triangulation, Visual Simultaneous Localization and Mapping (VSLAM), etc., where the RANSAC algorithm is frequently used for outlier detection; note that RANSAC is the most widely used robust approach in photogrammetry and computer vision for outlier detection. It is known that the outlier ratio used in RANSAC primarily determines the number of trial runs needed, which eventually, determines the computation time. Over time, different methods have been proposed to reject the false-positive correspondences and improve RANSAC, such as GR\_RANSAC, SuperGlue, and LP-RANSAC. The specific objective of this study is to propose a filtering algorithm based on Graph Neural Networks (GNN), as a pre-processing step before RANSAC, which can result in improvements for rejecting the outliers. The research is based on the idea that descriptors of corresponding points, as well as their spatial relationship, should be similar in image sequences. In graph representation, built by the adjacency matrix of data (nodes features), there should be similarity for corresponding points that are close to each other in the image domain. From the many GNNs techniques, Graph Attention Networks (GATs) were selected for this study as they assign different importance to each neighbour's contribution as anisotropic operations, so the features of neighbour nodes are not considered in the same way, unlike other GNNs techniques. In our approach, we build a graph in each image, because the similarity of the two-dimensional spatial relationships between points in the image domain of consecutive images should be similar. Then during processing, points with any significantly different neighbours are considered as outliers. Next, the points can be updated in the GNN layer. GNN-RANSAC is tested experimentally on real image pairs. Clearly, the proposed pre-filtering increases the inlier ratio and results in faster convergence compared to ordinary RANSAC, making it attractive for real-time applications. Furthermore, there is no need to learn the features.

### 1. INTRODUCTION

Feature matching is considered as a primary step in all computer vision and photogrammetry applications, such as fast image recognition, 3D reconstruction, image georeferencing, motion/object tracking, and navigation since all the subsequent processing steps depend on the correctness of the correspondences. There are several studies that have attempted to detect and match the point pairs flawlessly, such as Scale-Invariant Feature Transform (SIFT) (Lowe., 2004), Speeded-Up Robust Features (SURF) (Bay et al., 2008), ORiented FAST and rotated BRIEF (ORB) (Rublee et al., 2011). The feature descriptors can be seen as the signature of the point, based on, we want to compare the points across different images and find those whose signature is similar and thus are likely correct matches. Brute force matching, nearest-neighbour ratio, and local sensitivity hashing (Li et al., 2015) are widely used as matching methods, yet there is still uncertainty in feature matching. Therefore, outlier rejection is an essential step in any application; in other words, refining the correspondences by removing the false positive matches is of great interest. RANSAC algorithm (Brown et al., 2005) used for outlier detection is the most widely used robust approach in photogrammetry and computer vision for outlier detection, which is typically used to estimate the Fundamental, Essential, and Homography matrices. Removing the outliers early on leads to enhancing all the subsequent steps in photogrammetry and computer vision, such as triangulation, bundle adjustment and correctly estimating the Fundamental, Essential, and Homography matrices (Yang and Li, 2013). The traditional way to reject

the false positive matches relies on RANSAC (Brown et al., 2005; Turcot et al., 2009; Zhang et al., 2011). When there is a higher number of the sample points that don't fit the model, the RANSAC method may perform poorly and number of iterations could dramatically increase. An image pair with a high outlier's ratio, processed by using RANSAC can lead to a bad hypothesis and poor results even after many iterations (Bhattacharya et al., 2012). Before RANSAC was introduced, various methods have been developed and proposed in the statistics field, such as L-estimator, M-estimator, and least median of squares (LMedS) (Fotouhi et al., 2019). RANSAC simply iterates two steps and is not as complex as an M-estimator which uses sophisticated optimization or needs huge memory, such as the Hough transform. LMedS needs a numerical optimization algorithm to solve such a nonlinear minimization problem. Different methods have been proposed to modify the original RANSAC and can be categorized into three main types of research objectives: being accurate, being fast, and being robust as shown in Figure 1 (Choi et al., 2009). In addition, there are some techniques that used different kinds of optimization techniques, such as particle swarm optimization (PSO) into RANSAC (PSOSAC) (Wu et al., 2018), which is less sensitive to the correct rate than RANSAC or Genetic Algorithm Sample Consensus (GASAC) (Rodehorst and Hellwich, 2006; Shojaedini et al., 2019).

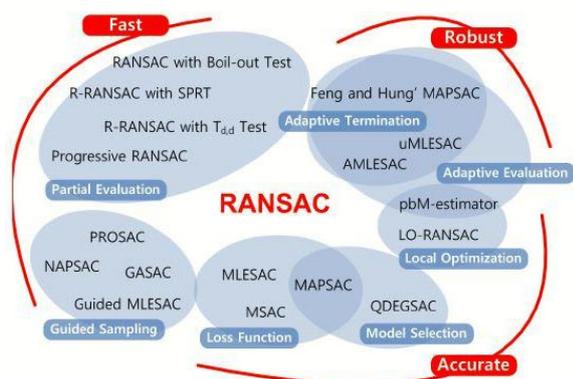


Figure 1. RANSAC family

Over time, different methods have been proposed to reject the false-positive correspondences and improve the RANSAC, such as GR RANSAC (Elashry et al., 2021) which depends on the geometric relation between the features but requires adjustable thresholds based on the images' relative orientation, SuperGlue (Sarlin et al., 2020) which learns feature matching based on Graph Neural Networks but needs to learn each feature based on all the features in the same image and the other image, and thus, it consumes more time, and LP-RANSAC (Wang et al., 2020) which uses RANSAC with locality preserving constraint. The specific objective of this study is to propose a filtering algorithm based on the Graph Networks, as a pre-processing step before RANSAC, which can result in improvements for rejecting the outliers and needs no variable threshold or to learn features, etc. The research here is based on the idea that descriptors of corresponding points, as well as their spatial relationship, should be similar in image sequences. After feature matching between two images, the points are scattered randomly in each image. Using Delaunay triangulation (Simon et al., 2005), a triangular mesh of the points in each image can be obtained from which we extract the graph information, such as the direct neighbours of each node and then build the adjacency matrix. Under the assumption that the similarity of the two-dimensional spatial relationships between points in the image domain of consecutive images should be similar, the neighbours of each point should be generally the same with the corresponding point in the other image. Otherwise, this point pair is a likely outlier. Similarly, the keypoints can be updated by their direct neighbours and compared with their corresponding ones using GNN, and if there is a significant difference, they will be outliers.

## 2. GRAPH NEURAL NETWORK

Over the past years, there has been a dramatic increase in interest in Graph Neural Networks (GNN) and rapid acceptance of GNN in many fields, such as Social Networks (Facebook, etc.), recommending/advising systems, medicine (classifying diseases) and pharmacy (learning molecular fingerprints, etc.) (Hamilton et al., 2018). The structure of the Graph (G) is defined by the nodes or vertices and connections between these nodes which are called edges,

formally expressed as  $G = (N, E)$ , which can be represented in an adjacency matrix (A), see Figure 2.

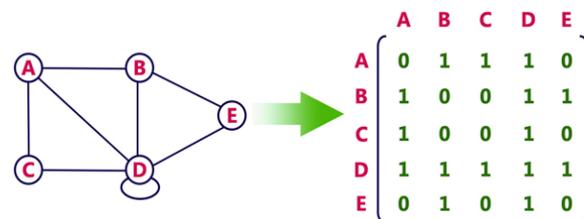


Figure 2. Building the adjacency matrix

The nodes or edges can have further properties which are called node features or edge features. In our case, the nodes are keypoints and their features are the descriptors. There are different types of operations/tasks, shown in Figure (3), that can be performed on graphs. First, node-level predictions or node classification simply means that if there is a graph with unlabeled nodes and we want to predict attributes about these nodes and classify them, then GNN will use the information from the other nodes in the graph to infer these unlabeled nodes. Another possibility is called link prediction or edge level prediction which predicts the connection between two nodes in the graph. Finally, we can use the whole graph as input and classify it or predict an attribute of interest.

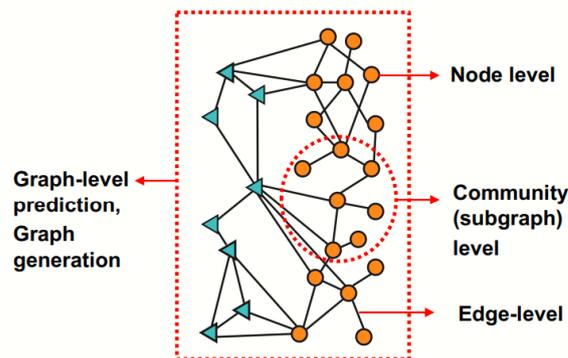


Figure 3: Different GNN operations

There are modern deep learning toolboxes designed for simple sequences and grids as shown in Figure 4.

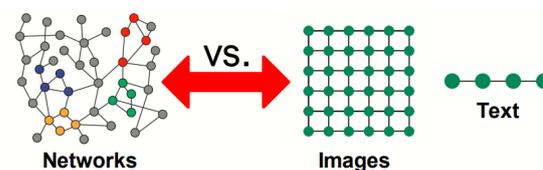


Figure 4: Network structure vs image structure

While graphs are generic, yet not everything can be represented as a sequence or a grid. For example, the networks or graphs have an arbitrary size and complex topological structure (i.e., no spatial locality like grids) and no fixed node ordering or reference point. The fundamental idea of the GNN is to train neural networks to be suitable for representation of graph data; this is called representation learning. Using all information about the graph, including

the node features and the connections stored in the adjacency matrix, the GNN outputs new representations which are also called embedding nodes as shown in Figure 5. These embedding nodes contain information from the other nodes in the graph. Then, the embedding can be used to perform predictions. Similar nodes meaning nodes with similar features will lead to similar node embedding, same way similar graph will lead to similar graph embedding by using GNN. Message passing layers are the core building blocks of the graph neural networks, they are responsible for combining the node and edge information into the node embedding.

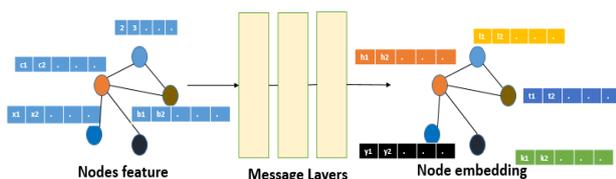


Figure 5: GNN structure

The basic idea of GNNs is to learn the embedding nodes by iteratively combining the node information in a local neighbourhood; in other words, the nodes learn something about the direct neighbours then the neighbours' neighbour and so on.

The message passing layers consist of update and aggregation functions:

$$h_u^{(k+1)} = UPDATE^{(k)}\left(h_u^{(k)}, AGGREGATE^{(k)}\left(\{h_v^{(k)}, \forall v \in N(u)\}\right)\right)$$

Aggregation uses the information of the direct neighbours of a node  $u$  and aggregate them in a specific way and then update the current state in step  $k$  and combine them with the aggregated neighbour states. In previous studies, several researchers have developed different methods of aggregation and update functions in the message layers. Different types of GNN layers perform diverse aggregation. The simplest formulations of the GNN layer, such as Graph Convolutional Networks (GCNs) (Kipf and Welling, 2017) or GraphSage (Hamilton et al., 2017.) execute an isotropic aggregation, where each neighbour contributes equally to updating the representation of the central node. Graph Attention Networks (GATs) (Veličković et al., 2018) was selected for this study which assigns different importance to each neighbour's contribution as anisotropic operations, so the features of neighbour nodes are not considered in the same way, unlike other GNNs techniques.

### 3. GRAPH ATTENTION NETWORKS (GAT).

The GAT depends on the attention-based architecture which assigns different importance to each edge through the attention coefficients as shown in Equations 1-4.

$$z_i^{(1)} = W^{(1)}h_i^{(1)} \quad (1)$$

$$e_{ij}^{(1)} = \text{LeakyReLU}\left(\bar{a}^{(1)T}\left(z_i^{(1)} \parallel z_j^{(1)}\right)\right) \quad (2)$$

$$\alpha_{ij}^{(1)} = \frac{\exp\left(e_{ij}^{(1)}\right)}{\sum_{k \in N^{(1)}(i)} \exp\left(e_{ik}^{(1)}\right)} \quad (3)$$

$$h_i^{(1+1)} = \sigma\left(\sum_{j \in N^{(1)}(i)} \alpha_{ij}^{(1)} z_j^{(1)}\right) \quad (4)$$

where  $\sigma$  is an activation function, which introduces non-linearity in the transformation, and  $W$  is the weight matrix of learnable parameters adopted for feature transformation. The processing steps are:

- **Equation (1)** is a linear transformation of the lower layer embedding  $h_i$ .
- **Equation (2)** determines a pair-wise attention score between two neighbours, where  $\parallel$  denotes concatenation.
- **Equation (3)** applies a SoftMax to normalize the attention scores on each node's incoming edges.
- **Equation (4)** is GCN aggregation, the embedding from neighbours are aggregated together, scaled by the attention scores.

### 4. RANSAC ALGORITHM

The RANSAC algorithm used for outlier detection (Yang and Li, 2013) is known that the outlier ratio primarily determines the number of trial runs needed, which eventually, determines the computation time. RANSAC simply iterates two steps: pick minimum random samples that fit the model and verify it to the data. The points that are less than the threshold will be classified as inlier points and otherwise they will be counted as outliers. These steps are repeated till a specified iteration number is reached. Knowing statistical parameters, the minimum iteration number can be estimated by the following equation:

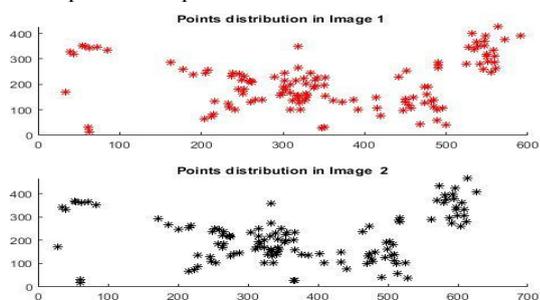
$$M = \frac{\log(1-p)}{\log(1-(1-e)^s)}$$

- p.... indicates the probability that all the points in the sample are inliers,
- s.... number of the random sample points,
- e..... the outlier ratio,
- M... the number of iterations.

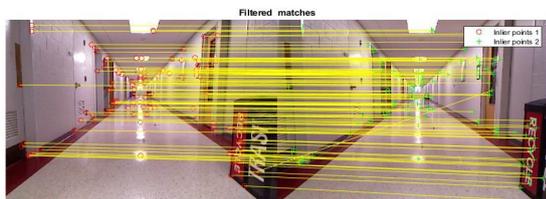
Clearly, the number of iterations heavily depends on the outliers' ratio in the dataset and the number of the sample points. Note that the second parameter is less controllable, as it primarily depends on the image texture content. With a lot of false positive matches (large outlier ratio) in the matched list, many iterations may be required before RANSAC can find the correct hypothesis.

### 5. THE PROPOSED ALGORITHM

Our former work (Elashry et al., 2021) used the geometric relation between points based on their spatial relationship that should be similar in image sequences in the image domain. For example, the distances and the angles between the points should be similar in the image sequences where the difference between images is not big (high overlap). A point with a high ratio test score is a reference point in each image which measures the distance and angles between this reference point and all the points, and then, comparing these values, if there is a significant difference between the point pairs, they will be outliers. Note that this method forced us to set up a threshold depending on the relative closeness of orientation between the images. In this study, we propose two algorithms based on the spatial relationship that should be similar in image sequences in the image domain. So, the direct neighbours of each point should be the same as its correspondence in the other image. Otherwise, these two-point pairs will be outliers, as shown in Figure 6 in addition to the point descriptors.



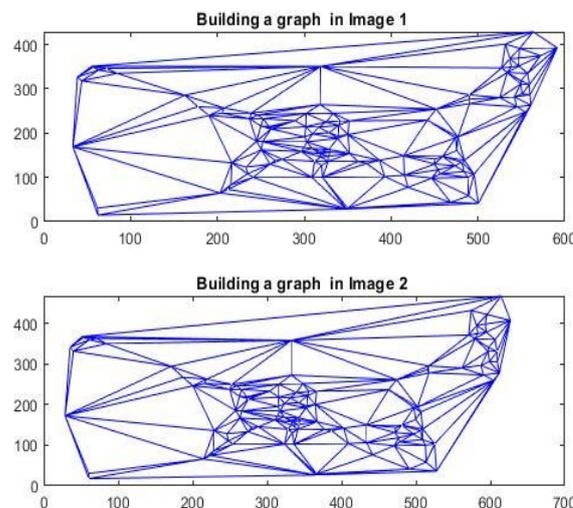
(a) Point distribution in two images



(b) Feature Matching

**Figure 6:** The distribution of the points in two consecutive images and their feature matching

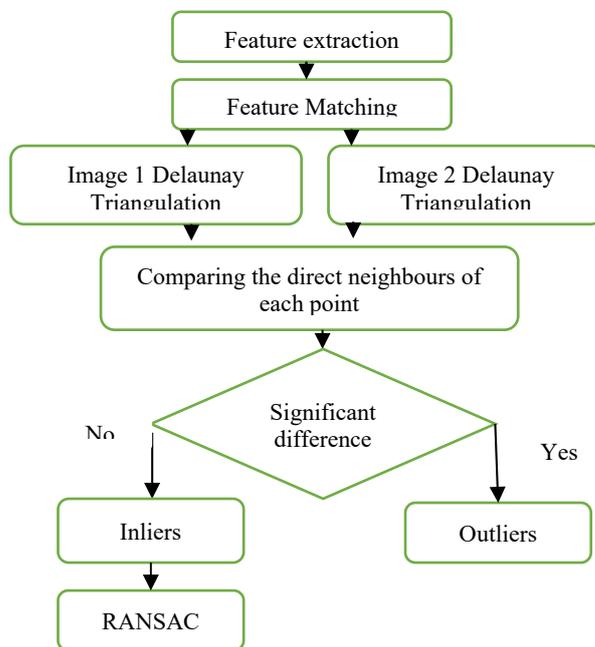
After feature matching between the two images, Delaunay triangulation is used to create a triangular mesh of the points in each image, i.e., the direct neighbours of each node to build the adjacency matrix as shown in Figure 7.



**Figure 7:** Building a graph based on the adjacency matrix

*Algorithm 1:*

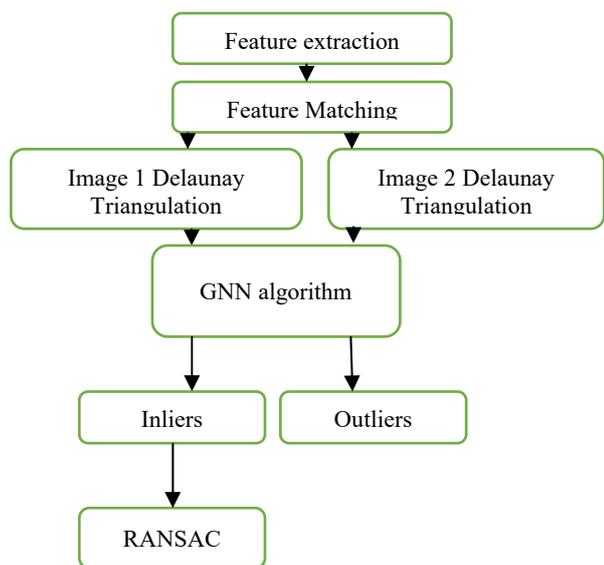
After building the graphs in the image sequences. The direct neighbours of each point should be the same as the direct neighbours with their corresponding point in the 2<sup>nd</sup> image. So, if there are any significant changes in the direct neighbours the two-point pairs will be considered as outliers and rejected from the dataset; the algorithm shown in Figure 8.



**Figure 8:** Algorithm 1 graph network RANSAC (GN\_RANSAC)

*Algorithm II:*

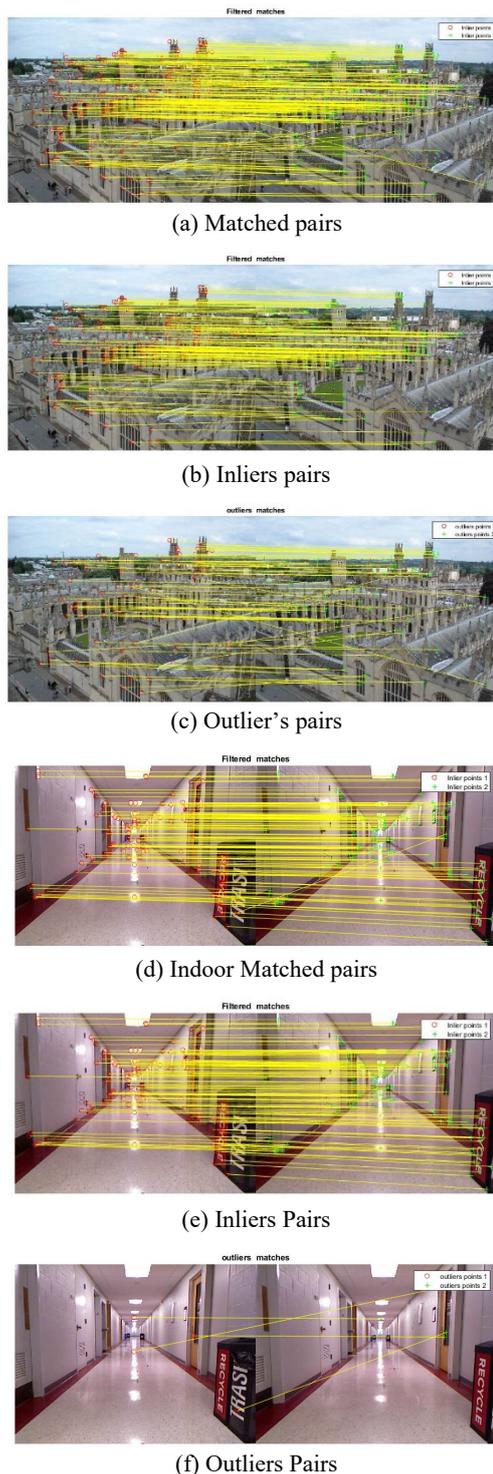
We propose an algorithm (GNN-RANSAC) as shown in Figure 9 to reject any false positive correspondences that might be found after performing Algorithm I. Note that this algorithm can be used separately, as it only depends on updating the point or the node feature with the direct neighbours' features using the GAT Networks that gives the node neighbours different importance based on the Euclidian distance between them. The GNN outputs are nodes embedding which contain information from its neighbours. Finally, the embedding can be used to perform predictions, such as similar nodes (Keypoints) meaning nodes with similar features will lead to similar node embedding. So, the correspondences should have the same embedding nodes and become inliers otherwise they are outliers.



**Figure 9:** The 2nd proposed algorithm graph neural network RANSAC (GNN\_RANSAC)

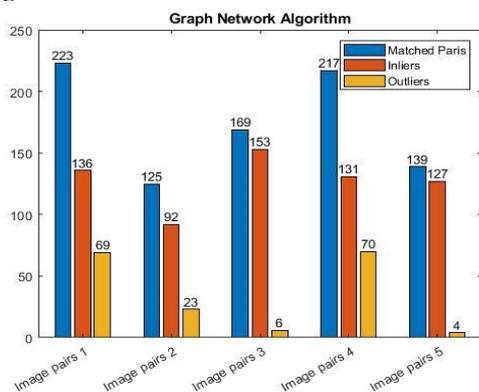
**6. RESULTS**

The purpose of our proposed algorithms is to reject the false-positive matches to reduce the outlier ratio and make RANSAC execute faster; the method exploits the geometric relation between the features and how the features get updated from their neighbours. For testing, we used two datasets, the Oxford landmark dataset which has images with different orientations and then the SPIN lab dataset which contains image sequences; both datasets were used to evaluate the performance of our algorithms. The algorithms were applied to several image pairs to remove the outliers, as shown in Figure 9. If the point in the 1<sup>st</sup> image has more than two neighbours different from the neighbours of its corresponding one, these pairs will be outliers.

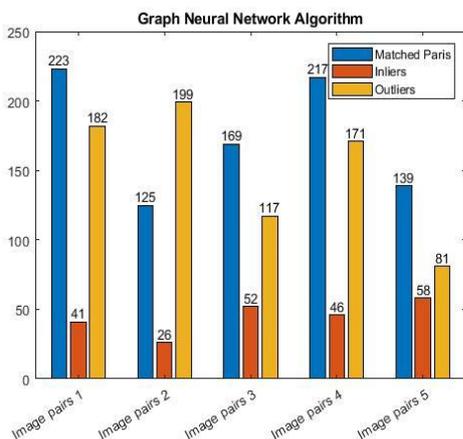


**Figure 10:** Executing the algorithms on the image pairs Oxford dataset (a-c) and SPIN lab dataset (d-f)

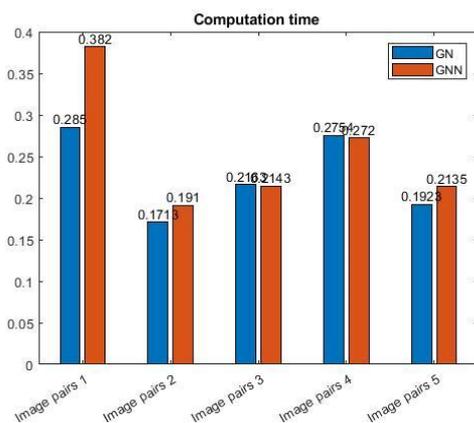
Applying the algorithms to several image pairs in a sequence or with different orientations allows to judge the performance of rejecting the false-positive matches from the image pairs. Removing the outlier pairs leads to reducing the outlier ratio and increasing the probability that the samples are inliers which makes RANSAC execute faster, as shown in Figure 10.



a. Graph Network Algorithm applied to reject the outliers



b. Graph Neural Network Algorithm applied to reject the outliers



c. Computation time of the two algorithms

Figure 10: Applying the algorithms to image pairs of different orientation

As can be seen from Figure 10, the two algorithms are applied to different image pairs. Looking at Figures 10a-b, it is apparent that when we used the Graph Neural Network algorithm standalone, it reduces the number of outliers and thus the dataset must help the RANSAC execute faster; especially, in image sequences. The GN algorithm experimental results indicate that it works well not only for image sequences but also on image pairs with different orientation. Figure 10c compares the computation time which is a relatively small difference between the two algorithms.

## 7. CONCLUSION

In conclusion, the study contributes to our understanding of the importance of rejecting false matches from the matched point pairs set based on exploiting the two-dimensional relationships between keypoints in the image domain. We tested two different techniques, GN and GNN and both give us good results that make the RANSAC algorithm on different datasets work faster than before. The GNN algorithm removes more mismatching pairs in image sequences than Algorithm I as shown in Figure 10, and thus makes the remaining pairs are more likely to be considered as inliers. When the dataset with significant orientation differences such as the Oxford dataset, Algorithm I is more reliable than Algorithm II, due to the larger distance between the nodes, so the importance of the neighbours will be different in both images. After removing the outliers, the outlier ratio will decrease, and consequently, the number of iterations decreases dramatically which will be reflected in the computational time. The computer vision and photogrammetry applications can benefit from the proposed algorithms due to the importance of the execution time that is critical to many computer vision applications.

## REFERENCES

- Brown, R. Szeliski, and S. Winder., 2005. Multi-image matching using multi-scale-oriented patches. CVPR.
- Choi, S., Kim, T., Yu, W., 2009. Performance Evaluation of RANSAC Family, in: Proceedings of the British Machine Vision Conference 2009. Presented at the British Machine Vision Conference 2009, British Machine Vision Association, London, p. 81.1-81.12. <https://doi.org/10.5244/C.23.81>.
- D. G. Lowe, 2004. "Distinctive image features from scale-invariant keypoints," International journal of computer vision, vol. 60, no. 2, pp. 91-110.
- Elashry, A., Sluis, B., Toth, C., 2021. IMPROVING RANSAC FEATURE MATCHING BASED ON GEOMETRIC RELATION. Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci. XLIII-B2-2021, 321-327. <https://doi.org/10.5194/isprs-archives-XLIII-B2-2021-321-2021>.

E. Rublee, V. Rabaud, K. Konolige and G. Bradski, 2011. "ORB: An efficient alternative to SIFT or SURF," in International conference on computer vision.

Fotouhi, M., Hekmatian, H., Kashani-Nezhad, M.A., Kasaei, S., 2019. SC-RANSAC: Spatial consistency on RANSAC. *Multimed Tools Appl* 78, 9429–9461. <https://doi.org/10.1007/s11042-018-6475-6>

Hamilton, W.L., Ying, R., Leskovec, J., 2018. Representation Learning on Graphs: Methods and Applications. arXiv:1709.05584 [cs].

Hamilton, W., Ying, Z., Leskovec, J., 2017. Inductive Representation Learning on Large Graphs 11.

P. Turcot and D. G. Lowe., 2009. Better matching with fewer features: The selection of useful features in large database recognition problems. ICCV Workshop on Emergent Issues in Large Amounts of Visual Data (WS-LAVD).

Rodehorst V, Hellwich O., 2006. Genetic algorithm sample consensus (gasac)-a parallel strategy for robust parameter estimation. In: Conference on computer vision and pattern recognition workshop, 2006. CVPRW'06. IEEE, pp 103–103

Sarlin, P.-E., DeTone, D., Malisiewicz, T., Rabinovich, A., 2020. SuperGlue: Learning Feature Matching With Graph Neural Networks. Presented at the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4938–4947.

Shojaedini, E., Majd, M., Safabakhsh, R., 2019. Novel adaptive genetic algorithm sample consensus. *Applied Soft Computing* 77, 635–642. <https://doi.org/10.1016/j.asoc.2019.01.052> Simon, G., Steiner, M., Biersack, E., 2005. Distributed Dynamic Delaunay Triangulation in d-Dimensional Spaces 12.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y., 2018. Graph Attention Networks. arXiv:1710.10903 [cs, stat].

Wang, G., Sun, X., Shang, Y., Wang, Z., Shi, Z., Yu, Q., 2020. Two-view Geometry Estimation Using RANSAC With Locality Preserving Constraint. *IEEE Access* 8, 726–7279. <https://doi.org/10.1109/ACCESS.2020.2964425>

Wu, Y., Miao, Q., Ma, W., Gong, M., Wang, S., 2018. PSOSAC: Particle Swarm Optimization Sample Consensus Algorithm for Remote Sensing Image Registration. *IEEE Geosci. Remote Sensing Lett.* 15, 242–246. <https://doi.org/10.1109/LGRS.2017.2783879>

Yang, S., & Li, B., 2013. Outliers Elimination Based Ransac for Fundamental Matrix Estimation. In 2013 International Conference on Virtual Reality and Visualization (pp. 321–324). <https://doi.org/10.1109/ICVRV.2013.63>

Y. Zhang, Z. Jia, and T. Chen., 2011. Image retrieval with geometry preserving visual phrases. CVPR, pages 809–816.

H. Bay, T. Tuytelaars and L. Van Gool, 2008. "SURF: Speeded up robust features," in European conference on computer vision.

J.-y. Li and J.-h. Li, 2015. "Fast Image Search with Locality-Sensitive Hashing and Homogeneous Kernels Map," *The Scientific World Journal*.

P. Bhattacharya, M. Gavrilova., 2012. Improving RANSAC feature matching with local topological information Voronoi diagrams in science and engineering (ISVD), ninth international symposium pp. 17-23.