Graph Based Texture Pattern Classification

Iyyakutti Iyappan Ganapathi *KUCARS*
Khalifa University
Abu Dhabi, UAE
iyyakutti.ganapathi@ku.ac.ae
Sajid Javed *C2PS*
Khalifa University
Abu Dhabi, UAE
sajid.javed@ku.ac.ae
Robert Bob Fisher *IPAB*
University of Edinburgh
Edinburgh, UK
rbf@inf.ed.ac.uk
Naoufel Werghi *C2PS & KUCARS*
Khalifa University
Abu Dhabi, UAE
naoufel.werghi@ku.ac.ae

Abstract

Textures in 3D meshes represent intrinsic surface properties and are essential for various applications, including retrieval, segmentation, and classification. However, it is distinct from other types of 3D object analysis. The primary objective is to capture the surface variations induced by multiple textures. While numerous classical approaches are published in the literature, only a few work directly on 3D meshes. Given the versatility of graph representations, we propose a graph learning- based approach for classifying the texture of each facet in a 3D mesh. First, a three-dimensional mesh is transformed into a graph structure in which every node is a facet of a given mesh. Further, each facet is described by a feature vector computed utilizing the neighboring facets within a radius and their geometric properties. The graph structure is then fed into a graph neural network, classifying each node as a texture or non-textured class. The proposed technique has been validated using texture patterns from SHREC'18 and demonstrated positive performance.

Index Terms

Relief pattern, 3D texture, Feature descriptor, Classification

I. Introduction

Texture analysis is a computer vision study that is utilized in two-dimensional and three-dimensional domains to categorize, segment, and retrieve distinct texture patterns [2], [3], [9]. On the other hand, texture analysis has been extensively explored in the 2D domain, and it is a relatively new area in the 3D domain. The main goal is to develop a method to extract texture patterns from 3D meshes and compare them to other 3D meshes to evaluate similarity. The relationship between texture and polygonal resolution, for example, is crucial in museums and cultural institutions for the preservation, research, and presentation of cultural heritage content [7]. Also, understanding the texture of damaged sculptures and artifacts might aid in precise reproduction in the 3D reconstruction of archaeological sites. There are a few ways to analyze texture patterns (Figure 1) in 3D meshes available in the literature [25]; these approaches effectively classify the texture patterns. Several existing algorithms convert 3D meshes to 2D images and employ texture detection techniques on the obtained 2D images [3], [6]. However, there are only a few methods for directly recognizing texture patterns on 3D meshes. These algorithms construct spatial and frequency domain global or local descriptors to recognize texture patterns. Geometrical parameters such as curvature, curvedness, and normals are used to build three-dimensional descriptors in the time domain, whereas the Laplace-Beltrami operator generates the most spectral descriptors. However, these techniques are inefficient in various situations, including surfaces with many textures and a significant degree of texture variation. Recently, introduced a 3D mesh convolutional algorithm for texture retrieval and classification. This technique employs a grid structure derived at each facet through an ordered ring and then performs 2D convolution on the computed grid [20].

Deep learning has gained in popularity due to its many applications, including handwritten character identification, image and video recognition, text analysis, and voice recognition. Several deep learning-based algorithms for 3D object recognition, classification, and segmentation are available in the 3D domain [16], [17]. However, it is unclear how these techniques may be extended to handle texture pattern recognition because texture patterns are distinguished by parts and local features that are independent of the overall shape. A CNN-based approach, in general, accepts as input a two-dimensional image with a regular

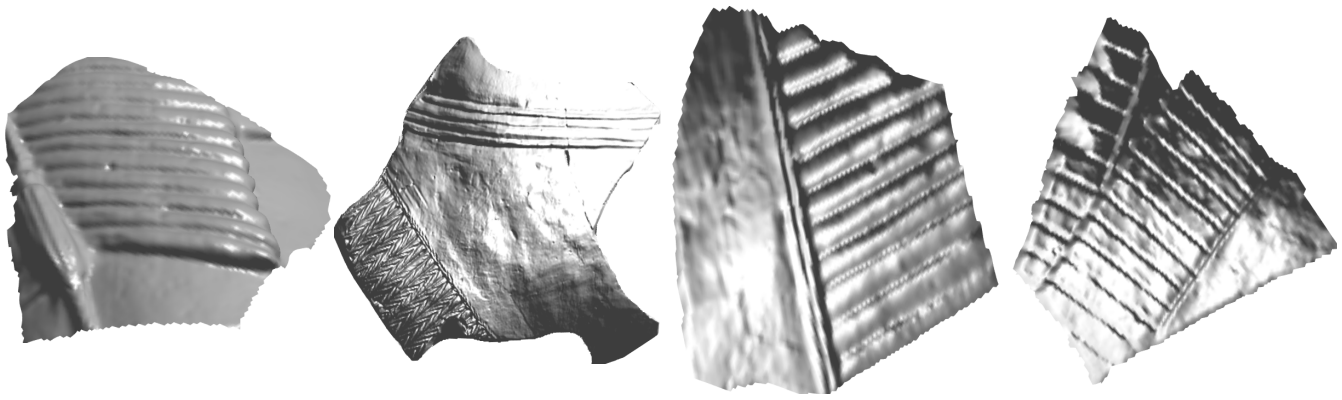


Fig. 1. A few example patches from the SHREC'18 dataset have texture and non-texture regions. The first patch has two textures and one non-texture, whereas the other three have one texture and one non-texture region. The first and third patches have a texture in common, and the remaining textures present in other patches are different.

grid structure. As a result, deep learning is being used to address texture recognition by transforming 3D models into depth images and 2D images. The transformation from one domain to another, on the other hand, loses information regarding surface variation, which is crucial for texture detection. 3D meshes, on the other hand, can be directly used by using networks such as MeshCNN [8]. However, the feature generated by this technique is a result of tessellation rather than geometric features. As a result, there is a huge demand for novel algorithms to generate efficient features to identify texture patterns.

To address the issues mentioned above, we present a deep graph neural network-based technique for 3D meshes. Though classical graph-based algorithms are known for texture classification [14], this is the first time a 3D mesh texture classification problem has been addressed using a graph neural network. The 3D meshes are converted to a graph structure, and each node in the graph is classified. In graph construction, we assume that each facet has three neighboring facets. Each face in a 3D mesh serves as a node, with edge connections created with adjacent facets. At each node, we compute a geometrical feature called local depth, and this feature is used for message passing and aggregation between nodes in the graph network. As a result, the features computed at each node are critical; therefore, the computed geometric surface features should be more efficient than curvature and curvedness features. The performance of the proposed technique is evaluated using the SHREC'18 dataset and is found to be promising.

The remainder of this article is organized as follows. In Section II, we review related work. Section III describes the proposed algorithm in detail, followed by the implementation. The experimental evaluations and discussions are presented in Section IV. Finally, the conclusions are drawn in Section V.

II. Related work

In the 2D image domain, numerous methods for defining texture patterns based on repeatability, randomness, and orientation have been developed [5], [12]. The most widely used techniques extracted this information by local descriptors derived from convolution-based filtering operations such as Gabor or Local Binary Pattern (LBP) [15], where LBP, in particular, is one of the most basic and widely used local texture descriptors. The texture analysis is a mature field in the 2D domain; however, in the 3D domain, it is in the early stage. Therefore, several techniques [3] in the literature extend the texture analysis used in 2D to 3D. For example, in [20], a 3D mesh convolution is introduced, an extended version of 2D. Likewise, we can find the extension of the most popular 2D techniques, HOG [5] and LBP extensions as MeshHOG [24], and MeshLBP [21]. Inspired by LBP in 2D, a mesh local binary pattern (meshLBP) is introduced in [21]–[23]. To have a regular grid-like structure similar to 2D images, this technique introduced ordered ring facets (ORF) to have an ordered structure of mesh support region for LBP computation. Depending on the requirement, this technique computes multi rings at each facet, and through subsampling, twelve facets are chosen at each ring. Further, the difference between central facets and the neighbouring facets is used in LBP computation using geometric properties obtained at facets. This technique faces challenges with boundary facets where adjacent facets are absent. A few other techniques use images obtained from 3D meshes [4] and then apply image processing techniques such as morphological operations to find the texture pattern. Though these techniques are simple and easy to handle, in SHREC'17, a track on relief patterns, these techniques performed better than many other techniques. In [3], a covariance descriptor is introduced using 2D images obtained from 3D mesh. These covariance descriptors are computed for patches and utilized to find the similarity of texture patterns. Similarly, Giachetti *et al.* use a 2D raster image of 3D meshes, and then IFV (improved fisher vector) is applied to the obtained 2D image to get the feature vectors [6]. Tatsuma and Aono obtained depth images from 3D meshes and then converted them into LBP images. Further, Kaze features are computed on the LBP image. A few statistical features were computed on the LBP image and concatenated as the final feature [3].

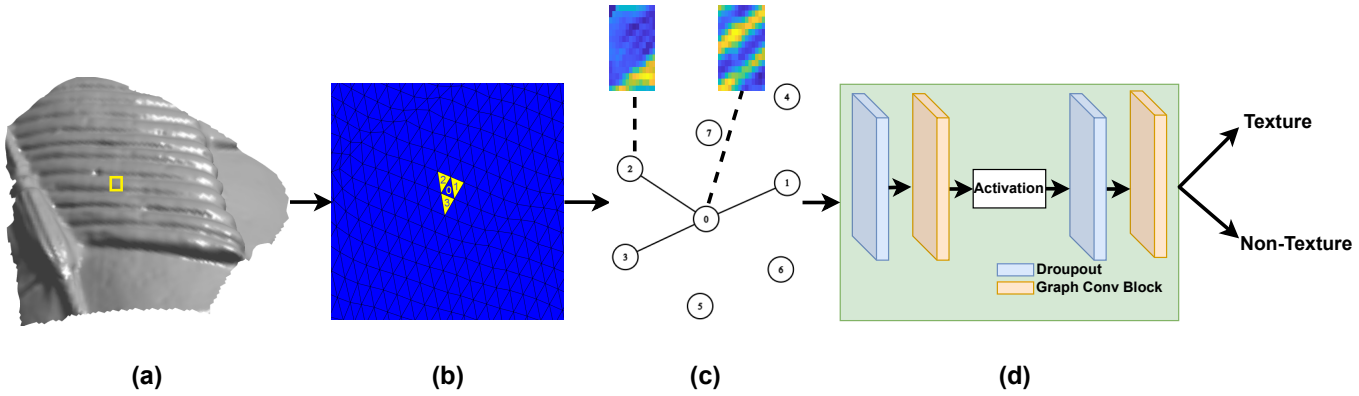


Fig. 2. Outline of the proposed method. (a) 3D mesh patch with a yellow box around a facet, (b) adjacent facets (1, 2, 3) of the chosen facet (0), (c) sample edge connections between the central facet and the adjacent facet, where the facets are nodes, and each node is represented by a feature descriptor (for node 0 and node 2 are shown), (d) constructed graph with its node feature fed to graph neural network for classification.

Like MeshLBP, an EdgeLBP on the mesh surface where concentric spheres with different radii are used at each facet. For each intersecting contour, a sequence of 12 equidistant points is generated. Further, these points are used in computing the local binary pattern [19]. A few works in mesh convolution can be applied directly on 3D meshes. In [8], a method is introduced where all types of common operations, such as convolution and pooling, similar to 2D images, are performed on 3D meshes. This technique has shown better performance in the benchmark dataset; however, the features constructed are not from geometric surface properties but from tessellation modification. Another technique in [20] introduced two mesh convolutions where one uses a 2D grid computed at each facet, and further, a 2D convolution is performed on the obtained grid. Another one uses ORF and polar coordinates similar to regular grids and Cartesian coordinates used in 2D. Similarly, a geodesic convolution neural network (GCNN) is introduced in [13], applicable in non-Euclidean space. However, the complexity of geodesic distance computation increases linearly with a number of points is a disadvantage. Sun *et al.* introduced a descriptor using the interior dihedral angle of mesh edges. The obtained angles are used to construct a histogram which is used to find similarities between texture patches [3]. Texture based applications are more popular with wavelet transform in the 2D domain. Inspired by this, Masoumi *et al.* introduced a signature using wavelet and geodesic distances [14].

There is another category called spectral descriptor, which is more robust to 3D object transformation. Limberger and Wilson [11] introduced a curvature-based Laplace Beltrami operator, which is decomposed to construct an improved wave kernel signature (IWKS). Further, the obtained kernel is encoded using the Fisher vector and Super vector. Though the method follows a standard approach, it is robust to non-rigid motion because of curvature, which down weights the articulated region. Signature Quadratic Form Distance (SQFD) [18] introduces a distance metric for matching the local features of 3D objects to evaluate their similarity. This technique generated the spectral descriptor using the Laplace-Beltrami operator and employed wave kernel descriptor variants to evaluate the performance of the proposed distance metric. The spectral descriptors, on the other hand, have their drawback. The complexity of these descriptors is directly proportional to the number of vertices in the input mesh since they rely on the computation of the Laplace-Beltrami operator and its eigendecomposition. As a result, the input 3D samples are downsampled to minimize complexity while simultaneously ensuring that the number of vertices is equal across all samples in a dataset. This results in a loss of fine details and reduced texture recognition performance.

III. Proposed Approach

We adopt deep learning-based algorithms for 3D mesh texture classification since they are successful in 2D computer vision applications such as recognition, classification, and segmentation. In 2D vision problems, CNN-based techniques generate the feature descriptor using the regular structure of a 2D image and convolution. However, these techniques are incompatible with 3D meshes due to the mesh structure. We approached the texture problem as a graph due to its versatile data structure. The graphs are fed into a graph neural network for node classification to predict the nodes' class in a graph with only partial node labeling. The following sections illustrate how to create a graph from a three-dimensional mesh and use geometrical features to describe each node in the graph for message passing and aggregation during network training. The outline of the proposed technique is illustrated in Figure 2.

A. Graph construction

A three-dimensional mesh is represented as a graph structure, with each facet representing a node in the graph. Each facet in the mesh has three adjacent facets, which are utilized to generate an adjacency matrix that creates the edge connection between the graph nodes. Further, each node is defined by a feature descriptor derived from the corresponding facet in the 3D mesh.

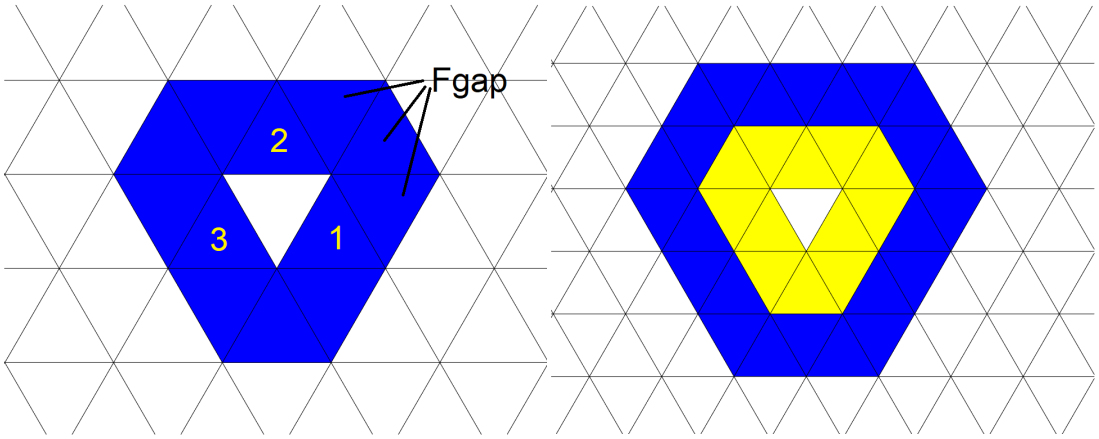


Fig. 3. Construction of an ordered ring of facets. (a) A single ring is created around the central facet using the adjacent facets (1, 2, 3) and the Fgap facets; (b) two rings are constructed similarly.

B. Node feature extraction

Two approaches are utilized to construct the descriptor to create an ordered ring around a facet and then a mesh grid based on it because the existing mesh structure is frequently uneven. The grid employed in this paper is inspired by [20], where it is constructed at each facet, f_c, using the ordered ring facet (ORF) developed by Werghi *et al.* [22]. Each facet in a mesh has three neighboring facets (1, 2, 3), and the gap between the adjacent facets to construct the ring is filled with three facets (Fgap), as illustrated in Figure 3(a). Similarly, multiple rings can be constructed at each facet, extending the coverage of the ordered region. Figure 3(b) depicts two rings created using the same process.

The indexes of the facets surrounding the central facet exhibit an arithmetic progression pattern that is used in grid construction. The grid is made up of four quadrants, each of which has its arithmetic pattern. We can extract the four quadrants that make up the Mesh-Grid by defining the quadrant size. The Mesh-Grid allows rotations based on the first neighboring facets to the center facets using the ORF ordered structure. Three different Mesh-Grid are thus feasible, one for each f_c neighboring facet. An example of three grids constructed around a central facet is shown in Figure 4. The node feature is constructed using these mesh grids computed from three orientations. After constructing the grid, each position in the grid is filled with

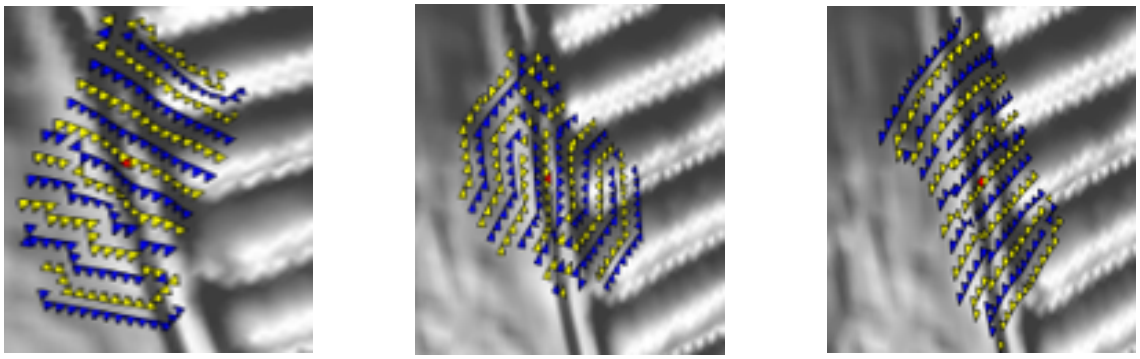


Fig. 4. Example of three grids extracted around a central facet (red).

a local depth (LD) value such that a two-dimension descriptor is obtained at each facet. The algebraic distance between a point on a manifold and a reference plane is defined as the LD value. The reference plane is determined by the vertices in the neighborhood that have the least eigenvector of the covariance matrix as normal. However, this feature captures local depth variation that differs from the curvature-based features H, C, and K2 in expressing the geometric texture.

C. Graph CNN

The graph network used consists of graph convolution, a linear layer, and a nonlinear activation. The data fed to the network is normalized to avoid varying degrees at each node. The node features are used as message passing between the nodes, and an average operation is used to aggregate the messages received at each node. Further, passing it to a dense layer and finally applying a nonlinear activation. Since the feature used is local depth at each node, it captures the geometric variations in the local vicinity. This helps the network to learn better as compared to other geometric features. The intuition here is that similar nodes are likely to share the same label where this has happened by passing the message, where each layer of the network

takes an aggregate of a neighbor node and passes it one "hop" away to the next node. So in our network with two-layer, we can convolve each node's 'second-order' neighborhood. Three networks are used in classifying the nodes, with the base model [10], and the other two models are the variants of the base model with residual layer and gated skip connected layer.

D. Training loss

A binary cross-entropy loss function is chosen and minimized during training to learn a set of parameters. Consider the parameter vector θ corresponding to the task of locating the texture region in a given mesh and is given as

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} -(y_i log(p_i) + (1-y_i) log(1-p_i)) \tag{1}$$

where p_i represents the probability of a textured region, $1-p_i$ represents the probability of a non-textured region, and $y_i = 1$ represents the label for a textured region and $y_i = 0$ otherwise. By passing training graphs, the optimal parameters are updated in each iteration. The optimal parameters learned are used to classify texture and non-texture regions in a test 3D mesh.

E. Implementation details

The proposed network is implemented using Tensorflow [1] on 2 RTX 2080 NVIDIA GPUs for 1000 epochs. The training is carried out using the dataset with pre-processing of 3D meshes to compute the node features. The binary cross-entropy loss function is minimized using the Adam optimizer with a learning rate of 0.001 and a dropout of 0.5. We used annotated label mask for the training graphs, where zero is assigned to the non-textured region and one is assigned to the textured regions.

IV. Experimental evaluation

The performance of the proposed approach is evaluated using the SHREC'18 dataset [4]. It consists of 10 patches, three of which include multiple texture patterns, while the remaining contain a single texture pattern and a non-textured region. There are 11 distinct texture patterns, excluding the non-textured region. Each patch contains various facets and is labeled as texture and non-textured using MeshLab. We conducted experiments and reported the results using three network architectures. The objective is to classify whether each facet in a patch corresponds to a node in a given graph that belongs to texture or non-texture. We evaluate our network's performance using accuracy, precision, recall, and F1-score. Precision is the proportion of correctly predicted positive instances, recall is the fraction of correctly predicted positive cases to all positive cases, and F1-score is a combination of precision and recall. Each patch is fed with 70% of the mesh facets as training and the rest of the facets as the test sample to train the network. An example training patch is shown visually in Figure 5(b). We used local depth features for each node, where one derived from a single orientation and the other derived using three orientations. These features are constructed as a grid with dimension 14×14 and later fattened to a single dimension. In the case of a single orientation, one grid is computed, and in the case of three orientations, three grids are computed. These grids are flattened and concatenated as the node feature. Hence, the dimension of the local depth feature from a single orientation is 1×196, and the dimension of the other is 1×588.

A. Quantitative results

To demonstrate the performance of the proposed technique, four metrics, accuracy, precision, recall, and F1-score, are computed on the SHREC'18 dataset. The performance of the proposed network is shown in Tables I and II. We measured performance using seven patches in which half of the nodes are not seen by the network. In patches containing multiple texture regions, we consider the texture regions with the highest percentage to be the texture and the others to be non-texture. The results obtained using single orientation local depth features demonstrated better performance than the other. In the case of a single orientation, the precision and recall values are less for a few patches compared to three orientations. However, a single orientation's accuracy is always high compared to a three orientation. Also, while training the network using three orientations, we found that the accuracy reduces after a certain epoch; however, there is a steady increase in accuracy and loss convergence in a single orientation.

B. Qualitative results

A few visual examples are shown to demonstrate the performance of the proposed technique. The mesh's texture and non-textured facets are used along with the ground truth labels. The results are obtained using local depth features derived in single and three orientations. Figure 5 show the results for one orientation, and Figure 6 shows the results for three orientation. The ground truth of a patch is shown in Figure 5 (a), where red represents the texture region and blue represents the non-textured region. A portion of nodes from this ground truth patch, as shown in Figure 5(b), is given as input to train the graph network. The predicted results for three models are shown in Figure 5(c-e), where the based model has produced better results as compared to the gated skip-connected network. However, the base model with residual layer has been predicted better than the

TABLE I

THE PERFORMANCE OF THE PROPOSED APPROACH IN TEXTURE CLASSIFICATION USING LOCAL DEPTH FEATURES DERIVED USING SINGLE ORIENTATION. THE RESULTS ARE SHOWN FOR THREE MODELS WHERE THE BASE MODEL WITH THE RESIDUAL LAYER HAS PRODUCED BETTER RESULTS THAN THE OTHER TWO. THE BEST RESULTS ARE HIGHLIGHTED IN BOLD.

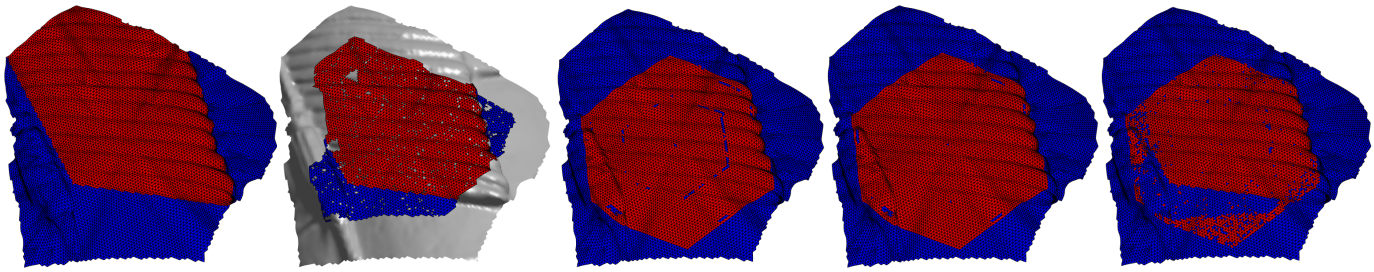
Patch	Base model				Base model with Residual layer				Base model with Gated Skip Connection			
	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score	Accuracy
1	0.64	0.71	0.67	0.63	0.64	0.72	0.68	0.64	**0.73**	**0.76**	**0.74**	**0.69**
2	0.58	0.61	0.60	0.59	**0.60**	**0.66**	**0.63**	**0.62**	0.56	0.61	0.59	0.58
3	0.62	0.65	0.63	0.61	**0.64**	**0.70**	**0.67**	**0.64**	0.59	0.58	0.58	0.59
4	**0.69**	**0.72**	**0.71**	**0.68**	0.65	0.68	0.67	0.65	0.61	0.67	0.64	0.62
5	0.59	0.61	0.60	0.60	**0.62**	**0.69**	**0.66**	**0.67**	0.56	0.59	0.58	0.57
6	0.57	0.59	0.58	0.56	**0.61**	**0.66**	**0.64**	**0.62**	0.54	0.57	0.55	0.56
7	**0.66**	**0.72**	**0.69**	**0.65**	0.63	0.67	0.65	0.64	0.60	0.63	0.62	0.61

TABLE II

THE PERFORMANCE OF THE PROPOSED APPROACH IN TEXTURE CLASSIFICATION USING LOCAL DEPTH FEATURES DERIVED USING THREE ORIENTATIONS. THE RESULTS ARE SHOWN FOR THREE MODELS WHERE THE BASE MODEL WITH THE RESIDUAL LAYER HAS PRODUCED BETTER RESULTS THAN THE OTHER TWO. THE BEST RESULTS ARE HIGHLIGHTED IN BOLD.

Patch	Base model				Base model with Residual layer				Base model with Gated Skip Connection			
	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score	Accuracy	Precision	Recall	F1-Score	Accuracy
1	0.64	0.71	0.67	0.63	0.60	0.66	0.63	0.57	**0.70**	**0.74**	**0.72**	**0.66**
2	0.58	0.61	0.59	0.54	**0.59**	**0.61**	**0.60**	**0.58**	0.51	0.57	0.54	0.52
3	0.54	0.60	0.57	0.53	**0.60**	**0.64**	**0.62**	**0.59**	0.56	0.58	0.57	0.53
4	0.59	0.60	0.60	0.56	**0.59**	**0.69**	**0.65**	**0.61**	0.60	0.62	0.61	0.59
5	0.54	0.61	0.58	0.56	**0.61**	**0.64**	**0.63**	**0.60**	0.58	0.61	0.60	0.57
6	0.51	0.55	0.53	0.52	**0.56**	**0.60**	**0.58**	**0.58**	0.52	0.58	0.55	0.55
7	**0.60**	**0.67**	**0.64**	**0.61**	0.58	0.60	0.59	0.60	0.62	0.65	0.63	0.58

other two networks. Similarly, Figure 6(c-e) shows the results for local depth derived from three orientations. The predicted results show less performance as compared to single-oriented features. This reduction is because the feature vector length produced from three orientations is three times that of a single orientation. Increasing the number of rings around a facet may also improve its performance. Nevertheless, determining the optimal number of rings is challenging. A large number results in a global feature representation instead of a local representation. This issue can be addressed by using adaptive rings around a facet, given that a facet in the non-textured region requires fewer rings than one in the texture region.

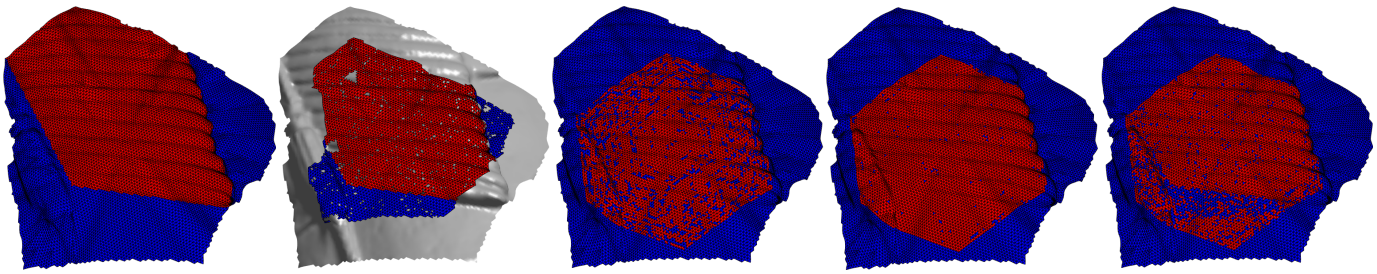


(a) (b) (c) (d) (e)

Fig. 5. The proposed approach achieved visual results for texture classification using local depth features derived using a single orientation. (a) A sample's ground truth, with red representing texture and blue representing non-texture, (b) a partial mesh facet used for training the network, (c) predicted results of the base model, (d) predicted results of base model with residual layer, and (e) predicted results of the base model with gated skip connection.

V. CONCLUSIONS

This article proposed a texture classification approach based on a graph deep neural network and a mesh grid local depth feature. We demonstrated categorizing texture at the facet level in a patch using a graph neural network. The three-dimensional



(a) (b) (c) (d) (e)

Fig. 6. The proposed approach achieved visual results for texture classification using local depth features derived using three orientations. (a) A sample's ground truth, with red representing texture and blue representing non-texture, (b) a partial mesh facet used for training the network, (c) predicted results of the base model, (d) predicted results of base model with residual layer, and (e) predicted results of the base model with gated skip connection.

mesh is first represented as a graph, and then each node in the graph is described by a geometric feature. We constructed local depth features in three orientations using a mesh grid formed by an ordered mesh ring to derive the feature descriptor. These features are used to categorize the nodes in a graph. The proposed network has been validated using data from the SHREC'18 dataset. Experimental results demonstrate an encouraging performance and effectiveness in classifying texture and non-textured nodes. We intend to expand on this work by developing custom aggregation for graph networks, allowing the aggregation to focus on nodes with similar textures during training for multi-class texture classification.

ACKNOWLEDGMENT

This work is supported by research funds from Khalifa University, Ref: CIRA-2019-047. The second author, Sajid Javed, is supported by the FSU-2022-003 project under award no. 000628-00001.

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous systems, 2015.

[2] Marco Andreetto, Nicola Brusco, and Guido Maria Cortelazzo. Automatic 3d modeling of textured cultural heritage objects. *IEEE Transactions on Image Processing*, 13(3):354–369, 2004.

[3] Silvia Biasotti, E Moscoso Thompson, Masaki Aono, A Ben Hamza, Benjamin Bustos, Shuilong Dong, Bowen Du, Amin Fehri, Haisheng Li, Frederico A Limberger, et al. Shrec'17 track: Retrieval of surfaces with similar relief patterns. In *10th Eurographics Workshop on 3D Object Retrieval*, 2017.

[4] Silvia Biasotti, E Moscoso Thompson, Loic Barthe, Stefano Berretti, A Giachetti, Thibault Lejemble, N Mellado, Konstantinos Moustakas, Iason Manolas, Dimitrios Dimou, et al. Shrec'18 track: Recognition of geometric patterns over 3d models. In *Eurographics workshop on 3D object retrieval*, volume 2, pages 71–77, 2018.

[5] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.

[6] Andrea Giachetti. Effective characterization of relief patterns. In *Computer Graphics Forum*, volume 37, pages 83–92. Wiley Online Library, 2018.

[7] David Gillespie and Kate Welham. Subjective and objective assessment of 3d textured and non-textured cultural heritage artefacts. *IEEE Computer Graphics and Applications*, 2020.

[8] Rana Hanocka, Amir Hertz, Noa Fish, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. Meshcnn: a network with an edge. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019.

[9] Shirui Hu, Zhiyuan Li, Shaohua Wang, Mingyao Ai, and Qingwu Hu. A texture selection approach for cultural artifact 3d reconstruction considering both geometry and radiation quality. *Remote Sensing*, 12(16):2521, 2020.

[10] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[11] Frederico A Limberger and Richard C Wilson. Feature encoding of spectral signatures for 3d non-rigid shape retrieval. In *BMVC*, pages 56–1, 2015.

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

[13] Jonathan Masci, Davide Boscaini, Michael Bronstein, and Pierre Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 37–45, 2015.

[14] Majid Masoumi, Chunyuan Li, and A Ben Hamza. A spectral graph wavelet approach for nonrigid 3d shape retrieval. *Pattern Recognition Letters*, 83:339–348, 2016.

[15] Timo Ojala, Matti Pietikainen, and David Harwood. Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In *Proceedings of 12th international conference on pattern recognition*, volume 1, pages 582–585. IEEE, 1994.

[16] Anshul Paigwar, Ozgur Erkent, Christian Wolf, and Christian Laugier. Attentional pointnet for 3d-object detection in point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[17] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[18] Ivan Sipiran, Jakub Lokoc, Benjamin Bustos, and Tomás Skopal. Scalable 3d shape retrieval using local features and the signature quadratic form distance. *The Visual Computer*, 33(12):1571–1585, 2017.

[19] Elia Moscoso Thompson and Silvia Biasotti. Description and retrieval of geometric patterns on surface meshes using an edge-based lbp approach. *Pattern Recognition*, 82:1–15, 2018.

[20] Claudio Tortorici, Stefano Berretti, Ahmad Obeid, and Naoufel Werghi. Convolution operations for relief-pattern retrieval, segmentation and classification on mesh manifolds. *Pattern Recognition Letters*, 142:32–38, 2021.

[21] Naoufel Werghi, Stefano Berretti, and Alberto Del Bimbo. The mesh-lbp: a framework for extracting local binary patterns from discrete manifolds. *IEEE Transactions on Image Processing*, 24(1):220–235, 2014.

[22] Naoufel Werghi, Claudio Tortorici, Stefano Berretti, and Alberto Del Bimbo. Local binary patterns on triangular meshes: Concept and applications. *Computer Vision and Image Understanding*, 139:161–177, 2015.

[23] Naoufel Werghi, Claudio Tortorici, Stefano Berretti, and Alberto Del Bimbo. Representing 3d texture on mesh manifolds for retrieval and recognition applications. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2521–2530, 2015.

[24] Andrei Zaharescu, Edmond Boyer, Kiran Varanasi, and Radu Horaud. Surface feature detection and description with applications to mesh matching. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 373–380. IEEE, 2009.

[25] Matthias Zeppelzauer, Georg Poier, Markus Seidl, Christian Reinbacher, Samuel Schulter, Christian Breiteneder, and Horst Bischof. Interactive 3d segmentation of rock-art by enhanced depth maps and gradient preserving regularization. *Journal on Computing and Cultural Heritage (JOCCH)*, 9(4):1–30, 2016.