Integral spin images usage in deep learning algorithms for 3D model classification

A I Denisenko, A A Krylovetsky and I S Chernikov

Computer Sciences Department, Voronezh State University, 1, University Square, Voronezh, 394018, Russia

E-mail: androwd@gmail.com

Abstract. We are investigating a problem of 3D model classification using deep learning algorithms. We propose integral spin images usage as 3D model representation. A number of computational experiments were made to build spin images for 3D models of Princeton Shape Benchmark and use them to train LeNet-5, AlexNet and ResNet deep neural networks. The results showed that integral spin images can be used in conjunction with deep learning algorithms in 3D model classification problem. However, with greater number of classes classification accuracy tends to decrease. It is expected that designing a more complex neural network architecture and expanding number of data characteristics can increase accuracy.

1. Introduction

Nowadays computer vision systems are widely used in different areas of technology. Autonomous vehicle control systems and medical systems require accurate identification of images, therefore classification of real-world objects with high precision is getting more and more important. Modern computer vision systems usually have a bunch of sensors working in visible spectrum accompanied by different infrared and ultrasonic sensors. Such systems are capable of constructing a high-quality monochrome 3D models of real-life objects.

In radiology there is a variety of imaging techniques such as magnetic resonance imaging, ultrasound and computed tomography. They form pictures of the anatomy and the physiological processes of the body in the form of monochrome 3D models. These models are then used in clinical analysis to diagnose and treat diseases including different kinds of tumors [1]. In technology 3D models are being used in quality control processes that include testing of units and determining if they are within the specifications for the final product [2]. Lidar scanners are frequently used in autonomous vehicle control systems [3]. They construct monochrome 3D models using distance measuring by illuminating the target with laser light and measuring the reflection with a sensor. It allows autonomous vehicles to recognize obstacles, traffic lights, pedestrians, etc.

Analyzing these different areas of application, we can come up with requirements for 3D model classification algorithms. First of all, such an algorithm should have high operational speed as models are usually processed in real-time. Secondly, it should be robust that includes minimization of type I and type II classification errors. Additionally, 3D model classification algorithm should be scale, rotation and translation invariant.

Existing approaches for 3D model classification algorithms can be divided in several groups based on used model representation approach. The first group of methods [4, 5] uses raw data which is a 3D
model represented by point cloud. This data is directly gathered by optical sensors without any further processing. But its usage in 3D model classification algorithms is quite complex as it can have noises and points are not connected to each other. Another approach includes 3D model representation as voxels or octrees [6, 7]. Generally, for each point of 3D space these data structures contain information whether this point is present in model or not. Disadvantage of this approach is high operational memory usage which makes it harder to work with the model. One more group of methods uses polygon mesh of 3D model surface [8]. This representation demands less computational resources and its execution time is more efficient. But this approach also makes it harder to work with real-life object models as they are usually complex and uneven. Because of that classification becomes more complex and 3D models should be additionally processed before learning.

There is one more approach that is becoming more popular nowadays. It is based on idea of using 3D model descriptors instead of 3D model itself. One of the frequently used local model surface descriptors is spin image descriptor [9]. Work [10] introduces integral spin images that can be used as global model surface descriptor and shows how they can be used in the automatic surface matching [11] and 3D model search [12] problems.

This work examines integral spin images applicability for the problem of 3D model classification using deep learning neural networks. The 3D model classification system can be divided in two phases: online and offline (figure 1). The offline phase is the neural network training using training dataset. The online phase is the direct usage of the system. During this phase request model is normalized and integral spin image is constructed. Then it is used as an input for the trained in offline phase classifier. The output result of the classifier is an array containing probabilities of class membership.

![Figure 1. 3D model classification system.](image)

The paper is organized as follows. In section 2 we recall the concept of spin image and the process of integral spin image construction for 3D model. Section 3 describes some neural network architectures that are successfully used for 2D image classification. In section 4 we propose computational experiment with classification of 3D models using integral spin images as their representation. We conclude in section 5.
2. Global surface descriptor calculation

On both online and offline phases input 3D model is processed and global surface descriptor is constructed to be later used in neural network. In general, global descriptors are the expansion of the concept of local surface descriptors that take only neighborhood points of the selected point on surface. This section deep dives into one of the proven local descriptors called spin image and describes global descriptor based on it called integral spin image.

2.1. Spin images

To construct spin images the cylindrical coordinate system without polar angle is introduced. Its origin is associated with one of the surface points that we call base point $O$. Longitudinal axis of the coordinate system corresponds to surface normal $n$ at the base point $O$. For each point $A$ from a locality of base point relative cylindrical coordinates $(\alpha, \beta)$ can be calculated using the following formula:

$$
\alpha = \sqrt{\|r_A - r_O\|^2 - (n \cdot (r_A - r_O))}, \quad \beta = n \cdot (r_A - r_O).
$$

where $r_O$ is a position vector of the base point in Cartesian coordinate system, $r_A$ is a position vector of point $A$.

![Relative cylindrical coordinates](image)

Figure 2. Relative cylindrical coordinates $(\alpha, \beta)$ of point $A$ from a locality of the base point $O$.

Point $A(\alpha, \beta)$ lies on a circle with radius $\alpha$ and the center in the base point $O$ on the distance of $\beta$ from the point $O$ along the normal vector $n$. The locality of the base point $O$ is divided into bins by the values of $\alpha$ and $\beta$. The resulting beans have a structure of a 3D ring as shown on the figure 3. In the result points that have close relative cylindrical coordinates $(\alpha, \beta)$ are placed at the same bin. Number of points lying in each bin can be represented as a matrix. This matrix is a spin image of the base point $O$ locality.

Spin image accuracy depends on the 3D model resolution. Resolution of a 3D model is defined as a mean value of all its edges’ lengths. If base point locality has an uneven resolution, spin image matrix has spread values and therefore its accuracy is low. Due to this a 3D model resolution should be normalized before a spin image construction [9]. For that purpose different algorithms can be used. Generally such an algorithm is based on edge split and edge collapse operations. The resulting model describes the same object as the original one but its edges’ lengths are within the given range.

2.2. Integral spin images

The idea of spin images can be extended to calculate global model surface descriptors. To do so all model points should be used instead of base point locality points.
In order to construct ordinary spin image, we should select one base point and calculate surface normal vector at this point. For integral spin images we also need base point and base vector. They should be calculated in such a manner so that integral spin images calculated for the identical models would be the same. Moreover, integral spin image should be scale invariable therefore a 3D model should be normalized first.

3D model normalization process includes optimal scaling and edges’ lengths normalization. Length normalization is performed in the same manner as for ordinary spin images.

Two 3D models have the same scale factor if standard deviation of their points is equal to 1. Therefore, the optimal scale factor can be calculated using the following formula:

$$w = \frac{1}{\sqrt{D}},$$  \hspace{1cm} (2)

where $D$ is a standard deviation of model’s points. Model scaled with this factor will have standard deviation of its points equal to 1.

Base point for each model should be selected in the same manner. For this reason, we can take its center of mass that is calculated as a mean of model surface’s points. Similar models will have their centers of mass near each other, therefore if we choose the right base vectors spin images in these points will be close to each other. After we have calculated the base point it makes sense to translate model in such a way so that the base point would become the coordinate system origin point.

Base vector can be calculated using principal component analysis (PCA). First, we need to calculate eigenvalues and eigenvectors for the covariance matrix of model points. For the 3D point cloud covariance matrix can be calculated in the following way:

$$C_{ij} = \frac{1}{n-1} \sum_{k=1}^{n} p^{(i)}_k p^{(j)}_k.$$  \hspace{1cm} (3)

Figure 3. Bins of a spin image.
where \( p^{(i)}_k \) and \( p^{(j)}_k \) are the \( i \)-th and \( j \)-th coordinates of point \( p_k \). The dimension of the matrix \( C \) is \( 3 \times 3 \), therefore it has three eigenvectors. We can select one of them that has the largest corresponding eigenvalue as the base vector for integral spin image.

3. Neural networks architectures analysis

In this section we will discuss some neural networks that already perform well in 2D images classification tasks. They include LeNet-5, AlexNet and ResNet architectures. All of them belong to the same class of deep neural networks called convolutional neural network (CNN).

CNNs usually consist of multiple convolutional, pooling (or subsampling) and fully-connected layers going after each other. Network architectures usually differ at number of layers and their parameters.

Convolutional layer is the main part of a CNN. In a large input image, a small section called kernel is taken and passed through all points. While passing kernel we convolve it into a single value.

Pooling (or subsampling) layer takes a small region of the convolutional layer output and sub-samples it to produce a single value. Different pooling techniques can be used such as maximum pooling, average pooling, mean pooling, etc. Pooling layer reduces the number of parameters to compute but it makes the network invariant to translations.

Fully-connected layer performs operation for each neuron in the current layer using data from all neurons in the previous layer to generate output.

The considered neural network architectures were originally designed to classify 2D images with fixed image size that differ for each architecture. Because of that in our research we “fit” these architectures to one size equal to \( 512 \times 512 \).

3.1. LeNet-5

LeNet-5 [13] is a CNN that consists of 7 training layers. They include 3 convolutional layers and 2 average pooling layers followed by 2 fully-connected layers (figure 4). Sigmoid function was used to include non-linearity before a pooling operation.

| Layer                        | Output data dimension | Number of training parameters |
|------------------------------|-----------------------|------------------------------|
| Input image                  | (512, 512, 1)         |                              |
| Convolutional Layer (5 x 5, step 2) | (508, 508, 6)         | 156                          |
| Average Pooling Layer (2 x 2, step 2) | (254, 254, 6)         | 0                            |
| Convolutional Layer (5 x 5, step 2) | (250, 250, 16)        | 2416                         |
| Average Pooling Layer (2 x 2, step 2) | (125, 125, 16)        | 0                            |
| Convolutional Layer (5 x 5, step 2) | (121, 121, 16)        | 6416                         |
| Fully-Connected Layer (84)   | 84                    | 19677988                     |
| Fully-Connected Layer (35)   | 35                    | 2975                         |

Figure 4. LeNet-5 architecture that can be used to classify monochrome 2D images with size \( 512 \times 512 \) by 35 classes.

3.2. AlexNet

AlexNet was designed by Krizhevsky et al. in 2012 [14]. Its architecture (figure 5) consists of 8 layers which include 5 convolutional layers and 3 fully-connected layers. After the first, second and fifth
convolutional layer max pooling layer is used. One of the main features of this architecture is usage of non-saturating rectified linear unit (ReLU) as an activation function after convolutional and fully-connected layers which showed better training performance over previously used sigmoid and tanh functions.

| Layer                          | Output data dimension | Number of training parameters |
|-------------------------------|-----------------------|-------------------------------|
| Input Image                   | (512, 512, 1)         |                               |
| Convolutional Layer (11 x 11, step 4) | (126, 126, 96)       | 11712                         |
| Max Pooling Layer (3 x 3, step 2) | (62, 62, 96)         | 0                             |
| Convolutional Layer (5 x 5)   | (62, 62, 256)        | 614656                        |
| Max Pooling Layer (3 x 3, step 2) | (30, 30, 256)       | 0                             |
| Convolutional Layer (3 x 3)   | (30, 30, 384)        | 885120                        |
| Convolutional Layer (3 x 3)   | (30, 30, 384)        | 1327488                       |
| Convolutional Layer (3 x 3)   | (30, 30, 256)        | 884992                        |
| Max Pooling Layer (3 x 3, step 2) | (14, 14, 256)       | 0                             |
| Fully-Connected Layer (4096)  | 4096                 | 205524992                     |
| Fully-Connected Layer (4096)  | 4096                 | 16781312                      |
| Fully-Connected Layer (35)    | 35                   | 35935                         |

Figure 5. AlexNet architecture that can be used to classify monochrome 2D images with size 512×512 by 35 classes.

3.3. ResNet
A residual neural network (ResNet) [15] was developed by Microsoft Research under supervision of K. He to solve a degradation problem when accuracy gets saturated with neural network depth increasing. ResNet introduces residual layers with shortcut connections (figure 6, a) for feedforward neural networks. Shortcut connections are those skipping one or more layers. They perform identity mapping, and their outputs are added to the outputs of the stacked layers. Shortcut connections are considered as building blocks of ResNet architecture.

ResNet architecture (figure 6, b) is largely based on VGG net architecture. It consists of multiple residual layers one after another. The convolutional layers in them mostly have 3×3 kernels. The network ends with a global average pooling layer and a fully-connected layer with softmax.

4. Computational experiment
We took Princeton Shape Benchmark as a dataset for training neural networks. It contains 1814 3D models that can be used as both train and validation datasets. Every model in dataset is represented by surface description in OFF format. In order to use this dataset for training we would need to construct integral spin images. For that reason, we did the following:

1. First of all, for each model we found its center of mass and then translated model by its position vector. In the result we got models that have center of mass at the coordinate system origin.
2. All models were optimally scaled with factor calculated using formula (2).
3. Then edge resolution normalization was performed for each model. We used the resolution value equal to 0.01. In the result we got models with bigger number of vertices and at the
same time each edge length was not greater than 0.01. That allows us to construct an accurate spin image for the model.

4. Finally, for each model we calculated base vector as described in subsection 2.2 and constructed an integral spin image with size 512×512.

![Residual Layer Architecture](image1)

![ResNet-18 architecture](image2)

Figure 6. a) Residual Layer Architecture. b) ResNet-18 architecture that can be used to classify monochrome 2D images with size 512×512 by 35 classes.

By doing these steps we got the new dataset with 1814 matrices that correspond to 3D models’ integral spin images. Then we used this dataset to train neural networks LeNet, AlexNet and ResNet-18 that were described in section 3.

Figure 7 shows the classification accuracy chart for the neural networks trained for 20 epochs to classify integral spin images by 2 classes. We can see that the best result has AlexNet, ResNet is slightly worse and LeNet’s accuracy does not exceed 80%. It means that results can noticeably differ from each other for different deep neural network architectures but still all of them can be used effectively to classify integral spin images by 2 classes.
Figure 7. Integral spin image classification accuracy for 2 classes.

Figure 8 shows that classification accuracy chart for the same neural networks trained to classify integral spin images by 6 classes. In this case, ResNet shows the best results while the other two networks have significantly worse results.

Analyzing these results, we can conclude that classification accuracy depends on the neural network architecture and the number of classes. The greater number of classes is used, the less accuracy we get. Apparently, the main reason for that is the usage of neural network architectures that were designed to solve different problems. Therefore, one of the ways to increase classification accuracy for bigger number of classes is to design a new neural network architecture specifically for integral spin images. Furthermore, we can enrich input data with other characteristics such as global surface descriptors calculated for different 3D model points or color information for each model point.

Figure 8. Integral spin image classification accuracy for 6 classes.

5. Conclusion
In this work we examined integral spin images applicability for the problem of 3D model classification using existing deep learning neural networks architectures. We conducted a number of computational experiments that included training of neural networks previously efficiently used with
2D images to classify 3D models. The results of these experiments show that integral spin images can be used with existing architectures to classify a small (2-3) number of classes. With greater number of classes an accuracy is starting to decrease. That means that integral spin images can be used as additional characteristics to improve existing results. In later researches we are planning to improve classification accuracy by extending the number of characteristics used and design a new neural network architecture so that integral spin images can be used for classifying greater number of classes with decent accuracy.

References

[1] Svensson C M, Krusekopf S, J. Lücke J and Figge M T 2014 Automated detection of circulating tumor cells with naive Bayesian classifiers Cytometry Part A 85 pp 501–511
[2] Asoudegi E and Pan Z 1991 Computer vision for quality control in automated manufacturing systems Computers & Industrial Engineering 21 pp 141-145
[3] Lu Y, Xue Z, Xia G S and Zhang L 2018 A survey on vision-based UAV navigation Geo-spatial Information Science 21 num 1 pp 21–32
[4] Charles R Q, Su H, Kaichun M and Guibas L J 2017 PointNet: Deep learning on point sets for 3D classification and segmentation Proc. IEEE Conf. Comput. Vis. Pattern Recognit (CVPR) 1 pp 77–85
[5] Erdogmus N and Marcel S 2013 Spoofing in 2D face recognition with 3D masks and anti-spoofing with Kinect Proc. IEEE 6th Int. Conf. Biometrics, Theory, Appl. Syst. (BTAS) 1 pp 1–6.
[6] Wang P S, Liu Y, Guo Y X, Sun C Y and Tong X 2017 Octreebased convolutional neural networks for 3D shape analysis ACM Trans. Graph 36 no 4 pp 1–11
[7] Brock A, Lim T, Ritchie J and Weston N 2016 Generative and discriminative voxel modeling with convolutional neural networks Proc. 3D Deep Learn. Workshop (NI PS) pp 1–9
[8] Hanocka R, Hertz A, Fish N, Giryes R, Fleishman S and Cohen-Or D 2019 MeshCNN ACM Trans. Graph. 38 no 4 pp 1–12
[9] Johnson A E 1997 Spin-Images: A Representation for 3-D Surface Matching (Pittsburgh: Carnegie Mellon University)
[10] Krylovetsky A A and Chernikov I S 2012 3D-model normalization for integral spin images calculation Izvestiya SFEDU. Engineering Sciences 6 pp 135-139 (in Russian)
[11] Krylovetsky A A and Chernikov I S 2010 Automatic surface registration: improved spin images Proc. Cybernetics and High Technologies of XXI Century 2 pp 781-790 (in Russian)
[12] Krylovetsky A A and Chernikov I S 2012 Integral spin images in 3D model search systems Proc. GraphCon 22th Int. Conf. On Computer Graphics and Machine Vision (Moscow) pp 214-217 (in Russian)
[13] Lecun Y, Bottou L, Bengio Y and Haffner P 1998 Gradient-based learning applied to document recognition Proceedings of the IEEE 86 no 11 pp 2278–2324
[14] Krizhevsky A, Sutskever I, Hinton G E 2012 Imagenet classification with deep convolutional neural networks Advances in Neural Information Processing Systems 25 pp 1097–1105
[15] He K, Zhang X, Ren S and Sun J 2016 Deep residual learning for image recognition The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 1 pp 770–778