Journal of Combinatorial Mathematics
and Combinatorial Computing
www.combinatorialpress.com/jcmcc

Combinatorial Press

Deep Convolutional Network Based 3D Target Modeling for Multi-view Video

Yijun Liu[1,✉], Junming Zuo[2]

[1] *Faculty of Humanities and Arts, Macau University of Science and Technology, Macau, 00853, China*
[2] *School of Digital Media and Design, Neusoft Institute Guangdong, Foushan, Guangdong, 528225, China*

ABSTRACT
This paper studies the 3D target modeling method under multi-view video based on deep convolutional network. Through the detailed exposition of the basic theory of 3D target modeling technology and the complete derivation of non-uniform rational B spline curve, this paper establishes technical support such as camera coordinate system for the generation of 3D target model. According to the basic structure of Deep Convolutional Network (DCNN), a DCNN network model suitable for the research scenario of this paper is established, and the model is utilized for feature extraction of images in multi-view videos. The softargmin algorithm is used to generate the parallax map for parallax estimation in the parallax calculation stage. According to the parallax map, voxel-based 3D reconstruction of the target in the multiview video is performed, and the surface reconstruction of the voxel model is performed using the Marching Cubes algorithm, and after obtaining the surface model of the target object, texture mapping is performed to enhance the realism of the model. The deep convolutional network based 3D building method in this paper can effectively realize the feature extraction of target objects in multi-view video. In 3D target modeling, the model in this paper achieves good results on both public and measured datasets, and has obvious performance superiority and generalization ability compared with other methods.

Keywords: deep convolutional network, 3D target modeling, feature extraction, texture mapping, multi-view video

1. Introduction

With the continuous development of network technology, people's audio-visual entertainment is also changing. As a new multimedia form, multi-view video has become a mainstream form of network video in recent years by virtue of its powerful interactivity and visual impact [1-2]. Multi-view video refers to the video data of the same event from different viewpoints and angles, as well as different

✉ Corresponding author.
E-mail address: liuyijun@dgcu.edu.cn (Y. Liu).
Received 25 December 2024; Revised 10 February 2025; Accepted 25 March 2025; Published Online 16 April 2025.
DOI: 10.61091/jcmcc127b-051

points in time, which are simultaneously captured by multiple cameras [3]. Nowadays, it has been widely used in the fields of sports, concerts, games, medical treatment, etc., and has become a highlight of people's audiovisual life [4-7]. A multi-view video database contains visual information captured by numerous cameras, which can be used to realize users' seamless roaming and interactive viewing of the scene by reconstructing the global scene. Compared with the traditional video captured by a single camera, multi-view video has several characteristics: first, wide coverage. Multi-view video can simultaneously record the scene of the same event from different angles and locations, thus covering all the details of the event scene [8]. The content of this kind of video is more complete, but also can provide more background information, so that the audience can more deeply understand the process of event development. Second, the visual impact is strong. Through multi-camera shooting, multi-view video can present an intense environment, the visual impact is far more than the traditional single-camera shooting video [9]. Utilizing multi-view video technology allows viewers to experience the real scene in a more immersive way, increasing the audience's sense of participation and entertainment. Third, scene reconstruction is required. Multiview video requires the construction of a complete scene from many different camera captures [10]. This process requires alignment, blending and fusion of images captured by different cameras to obtain a complete scene. This process requires high technical and algorithmic requirements.

While 3D target modeling refers to the use of computer technology to model and design the target object to form a realistic and three-dimensional image [11]. And 3D target modeling with the assistance of multi-view video has also been studied earlier, mostly for the detection and localization of 3D targets with high resolution and accurate perception [12-14].

Multi-view 3D target modeling is important to achieve complete information about a certain scene or space. For example, military exercises need to breach a certain site, but due to the front vision blockage is difficult to directly breach, you can use drones and other means to obtain multi-view video to determine the characteristics of the three-dimensional target to achieve rapid breach [15]. Life is mostly the use of self-driving vehicles, the literature [16] used multi-view three-dimensional space to obtain a complete point cloud data, making the car three-dimensional and bird's-eye view works well. On this basis, literature [17] uses multi-view encoding dispersed 3D point cloud to form a deeply fused network framework, which realizes accurate 3D target localization and detection in the self-driving environment. It can be seen that multiview supports 3D target modeling, which is basically based on acquiring 3D target panoramic features (point cloud) from multiview to form a high-resolution feature network to achieve fast target modeling [18]. Literature [19], on the other hand, did segmentation processing on the panorama and reconstructed the 3D geometric model with the target contour points as the reference. However, there are overlapping 3D target features in multiple views or irrational phenomena formed because of angles and other reasons, which is a disturbance to the modeling. For this reason, literature [20] utilizes multi-view to obtain complete 3D target point cloud data, and removes the intersecting and unreasonable data, and then further utilizes the local convex connectivity patch segmentation method to obtain the accurate target and realize the complete detection of 3D targets. In contrast, literature [21] adds deep learning techniques on the basis of multi-view to directly train a more advanced classification network to classify the 3D target point cloud data in order to recognize the objects comprehensively and accurately. Further, literature [22] utilizes the VEDet framework based on multi-view 3D geometry for viewpoint perception and isotropic improvement of 3D target localization with encoded reconstruction of the 3D scene by visual sensors. In addition, literature [23] utilized deep neural networks to learn the keypoints of 3D targets, which can accurately predict the movements of 3D targets through stereo inputs. While literature [24] combined multi-view with 3D convolutional neural network to form MV3D-CNN which can recognize moving targets, such as moving vehicles and sign language actions.

In this paper, the three-dimensional coordinate system of the camera is converted to the two-dimensional coordinate system of the image pixels to realize the transformation from multi-view video

line to multi-view image. A non-uniform rational B-spline curve is utilized to generate a curved target object, which improves the flexibility and accuracy when processing the shape of the target object. Based on the advantages of deep CNN over shallow CNN in complex data processing and the basic structure of DCNN, this paper constructs a deep convolutional neural network structure containing three convolutional layers, two fully connected layers, and a Softmax classifier, and uses it to realize feature extraction on the input images of the dataset. In the parallax dimension, the cost space is converted to a probability space, and the parallax value of each pixel is multiplied by the corresponding estimated probability to obtain the predicted parallax value of the pixel, which is combined with the loss function for the training of the network model to realize the parallax estimation of the input image. According to the calibration parameters of the camera and the parallax map information, the image is mapped to voxels in 3D space, and the voxel model is optimized using the spatial sculpting algorithm to obtain a more accurate target voxel. The Marching Cubes algorithm is used to determine the generation of triangular facets by traversing each voxel unit in the voxel model, and the corresponding triangular facets are generated according to the combination of vertex states to construct the surface model of the target object. The appropriate texture image is selected from the multi-view video image, and the texture mapping operation is performed according to the geometry of the surface model and the relationship between the camera viewpoints to obtain the 3D target model with texture. Comparative analysis is used to compare this paper's model with other models in terms of feature extraction and 3D object modeling results, highlighting the suitability of this paper's deep convolutional network-based 3D modeling approach to the task of modeling multi-view video.

2. Three-dimensional target modeling techniques

2.1. Point of view and coordinate transformation

2.1.1. World Coordinate System and Camera Coordinate System. It is known that a point in three-dimensional space is $P(P_x, P_y, P_z)$. The translation under three-dimensional space can be denoted as $P' = P + T$ or expressed as equation (1):

$$P' = \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \end{bmatrix} P \tag{1}$$

where P denotes the three-dimensional coordinates of the point P after translation; and T denotes $\begin{bmatrix} x & y & z \end{bmatrix}^T$, the amount of translation.

The reformulation of the corresponding position of point P under the camera coordinate system is equivalent to a rotation in three dimensions under the world coordinate system [25]. In general, a rotation in any dimension can be expressed as the product of a coordinate vector and a suitable square matrix. The angle under the camera coordinate system to the world coordinate system θ is equivalent to the point P rotated by the same angle around the world coordinate system in reverse θ. In three dimensions, the rotation can be decomposed into rotations around each of the three coordinate axes, and if the rotation is performed sequentially around X, Y, Z the three coordinate axes, then the total rotation matrix R is the product of these three matrices $R_x(\psi), R_y(\phi), R_z(\theta)$, i.e.:

$$R = R_x(\psi) R_y(\phi) R_z(\theta) \tag{2}$$

From Eq. (3), Eq. (4) and Eq. (5), we can see the calculation process of $R_x(\psi), R_y(\phi), R_z(\theta)$:

$$R_x(\psi)=\begin{bmatrix}1 & 0 & 0\\ 0 & \cos\psi & \sin\psi\\ 0 & -\sin\psi & \cos\psi\end{bmatrix} \tag{3}$$

$$R_y(\phi)=\begin{bmatrix}\cos\phi & 0 & -\sin\phi\\ 0 & 1 & 0\\ \sin\phi & 0 & \cos\phi\end{bmatrix} \tag{4}$$

$$R_z(\theta)=\begin{bmatrix}\cos\theta & \sin\theta & 0\\ -\sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix} \tag{5}$$

The transformation matrix from the world coordinate system to the camera coordinate system is then obtained, see equation (6):

$$P'=\begin{bmatrix}R & t\end{bmatrix}P \tag{6}$$

2.1.2. Image and Pixel Coordinate System. The transformation from the camera coordinate system to the image coordinate system is the process of converting a three-dimensional spatial coordinate system to a two-dimensional planar coordinate system (3D-2D), called a perspective projection transformation. In order to solve the relationship between them, the ordinary image coordinates (x, y) are expanded to chi-square coordinates (x, y). A point in space, projected onto the image plane is on a straight line with the center of light of the camera.

The conclusions that can be obtained based on the similar triangle relationship are given in Equation (7) and Equation (8):

$$x=f\cdot\frac{X_c}{Z_c} \tag{7}$$

$$y=f\cdot\frac{Y_c}{Z_c} \tag{8}$$

f is the camera focal length (the distance from the camera optical center to the imaging plane), expressed in matrix form as equation (9):

$$Z_c\begin{bmatrix}x\\ y\\ 1\end{bmatrix}=\begin{bmatrix}f & 0 & 0 & 0\\ 0 & f & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}X_c\\ Y_c\\ Z_c\\ 1\end{bmatrix} \tag{9}$$

Further, the relationship between the image coordinate system and the pixel coordinate system is shown in equation (10):

$$f(x)=\begin{cases}u=\dfrac{x}{dx}+u0\\ v=\dfrac{y}{dy}+v0\end{cases} \tag{10}$$

dx represents the width of a pixel (x direction), in the same unit as x, and indicates how many pixels are on the x axis, and similarly indicates the number of pixels on the y axis, and $(u0, v0)$ is the center of the image plane.

Converting the above relations into matrix form, see equation (11):

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{dx} & 0 & u0 \\ 0 & \frac{1}{dy} & v0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{11}$$

To summarize, the whole process of coordinate conversion from the world coordinate system to the pixel coordinate system, which is also known as the imaging plane, is expressed as a continuous left-multiplication next operation, as shown in Equation (12):

$$Z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{dx} & 0 & u0 \\ 0 & \frac{1}{dy} & v0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{12}$$

2.2. Non-uniform rational B-splines

2.2.1. B-spline basis functions. In Bessel curves, Bessel basis functions are used as weights. b-spline basis functions will be used in the same way. Unlike the Bessel basis functions, the B-spline basis functions have two properties [26], namely (1) the domain of action is subdivided by nodes, and (2) the basis functions are not nonzero over the entire interval. Each B-spline basis function is nonzero on several neighboring subintervals, and thus, the B-spline basis function scope is localized.

Let U be a set of $n+1$ non-decreasing numbers, i.e., $u_0 \le u_1 \le u_2 \le ... \le u_m$, where each u_i is called a knot and the whole sequence U is called a knot vector, see Eq. (13):

$$U = \{u_0, u_1, ..., u_m\} \tag{13}$$

The relationship between the number of degrees, the number of knots, and the number of control points is $m = n + k + 1$. For non-uniform and non-periodic B-splines, the knot vector adopts the form shown in Eq. (14):

$$U = \{0, 0, 0, 0, u_{k+1}, ..., u_{m+k+1}, 1, 1, ..., 1\} \tag{14}$$

where the 0's and 1's of the first and end nodes are repeated in k repetitions so that the endpoints of the curve coincide with the start and end control points.

Set U is the node vector and half-open interval $[u_i, u_{i+1})$ is the i rd node span. The nodes can be thought of as dividing points that subdivide the interval $[u_0, u_m]$ into node spans. All B-spline basis functions should have a domain of definition on $[u_0, u_m]$. Here, $u_0 = 0$ and $u_m = 1$ are usually used, so the domain of definition is the closed interval [0, 1]. In order to define the B-spline basis functions, let the order of the basis functions be d. The i th d th order B-spline basis function is denoted as $N_{l,d}(u)$ and its initial recursion is shown in Eq. (15) and Eq. (16):

$$N_{i,0}(u) = \begin{cases} 1 & \text{if } u_i \le u < u_{i+1} \\ 0 & \text{other} \end{cases} \tag{15}$$

$$N_{i,d}(u)=\frac{u-u_i}{u_{i+d}-u_i}N_{i,d-1}(u)+\frac{u_{i+d+1}-u}{u_{i+d+1}-u_{i+1}}N_{i+1,d-1}(u) \tag{16}$$

The above equation is often referred to as the Cox-deBoor recursion formula. These basis functions are all step functions if the order is 0, i.e., $d=0$. This is the first expression that indicates that u has a basis function $N_{i,0}(u)$ of 1 on the interval $[u_i,u_{i+1})$ at the i rd node.

2.2.2. Non-uniform rational B-spline curves. This section will extend the definition of B spline curves to nonuniform rational B spline curves by further derivations. First consider the method of introducing chi-square coordinates into the B-spline curve to derive the definition of nonuniform. The given $n+1$ control points are the $P_0,P_1,P_2,...,P_n$ and $m+1$ node vectors of the nodes $U=u_0,u_1,...,u_m$. The d th order B-spline curve defined by these parameters is shown in Eq. (17):

$$C(u)=\sum_0^n N_{i,d}(u)P_i \tag{17}$$

The control point P_i is rewritten as a column vector containing four components with the fourth component being 1, see equation (18):

$$P_i=(x_i,y_i,z_i,1)^T \tag{18}$$

One can think of this P_i as a chi-square coordinate. Since multiplying the coordinates of a point (in chi-square form) by a non-zero number does not change its position, multiplying the coordinates of P_i by a weight w_i yields a new form in chi-square coordinates, see equation (19):

$$P_i^{\omega}=w_i(C_i,1)=(w_ix_i,w_iy_i,w_iz_i,w_i)^T \tag{19}$$

Note that P_i^{ω} and P_i represent the same point in the chi-square coordinates. Substituting this new chi-square form into the above B-spline curve equation yields the result shown in Eq. (20):

$$\begin{aligned} C^w(u)&=\sum_0^n N_{i,d}(u)P_i^w \\ &=\sum_0^n N_{i,d}(u)w_i(C_i,1) \end{aligned} \tag{20}$$

$$C^w(u)=\begin{pmatrix} \sum_{i=0}^n N_{i,d}(u)w_ix_i \\ \sum_{i=0}^n N_{i,d}(u)w_iy_i \\ \sum_{i=0}^n N_{i,d}(u)w_iz_i \\ \sum_{i=0}^n N_{i,d}(u)w_i \end{pmatrix} \tag{21}$$

Thus, point $C^w(u)$ is the original B-spline curve in chi-square coordinate form. Now, converting it back to Cartesian coordinates by dividing $C^w(u)$ with the fourth coordinate, the form obtained is shown in equation (22):

$$C(u)=\sum_{i=0}^{n}\frac{N_{i,d}(u)w_i}{\sum_{j=0}^{n}N_{j,d}(u)w_j}P_i=\frac{1}{\sum_{i=0}^{n}N_{i,d}(u)w_i}\sum_{i=0}^{n}N_{i,d}(u)w_iP_i \tag{22}$$

This is the equation of the curve determined by control point $P_0,P_1,P_2,...,P_n$, knot vector $u_0,u_1,u_2,...,u_n$, and weights $w_0,w_1,w_2,...,w_n$. Note that since weights w_i are associated with control points P_i, the number of weights and the number of control points must be the same.

2.2.3. Non-uniform rational B-spline surfaces. A three-dimensional $d_0\times d_1$-degree tensor product NURBS surface is generated from a four-dimensional uniform B-spline surface. First, given the following information: (1) a set of $m+1$ rows of $n+1$ control points $P_{i,j}$ each, where $0\le i\le m, 0\le j\le n$. (2) the weights of the corresponding control points $w_{i,j}$. (3) a node vector of $h+1$ nodes in the U-direction $U=\{u_0,u_1,u_2,...,u_n\}$. (4) a node vector of $k+1$ nodes in the V-direction $V=\{v_0,v_1,v_2,...,v_k\}$. (5) an order p in the U-direction. (6) an order q in the V-direction. based on the formula of the nonuniform rational B-spline curve definition, the definition of the nonuniform rational B-spline surface formula is given in Eq. (23) and Eq. (24):

$$S(u,v)=\frac{\sum_{i=0}^{K}\sum_{j=0}^{L}N_{i,d_0}(u)N_{j,d_1}(v)w_{i,j}P_{i,j}}{\sum_{i=0}^{K}\sum_{j=0}^{L}N_{i,d_0}(u)N_{j,d_1}(v)w_{i,j}} \tag{23}$$

$$P_{i,j}=(C_{i,j},1) \tag{24}$$

where $C_{i,j}$ is the coordinate of each control point (i,j) in the grid array and $w_{i,j}$ is the weight of each control point (i,j). $N_{i,d_0}(u)$ and $N_{j,d_1}(v)$ are the d_0-degree basis functions in the u-direction and the d_1-degree basis functions in the v-direction, respectively. Control point (i,j) is a grid array which corresponds to the weight factor $w_{i,j}$. $w_{i,j}$ at the four corner points is greater than 0 and the rest are greater than or equal to 0. Note that a fundamental constancy, Eq. (25) and Eq. (26) must be satisfied in the u and v directions, respectively:

$$h=m+p+1 \tag{25}$$

$$k=n+q+1 \tag{26}$$

For convenience, the surface equation is defined in equation (27):

$$S(u,v)=\frac{A(u,v)}{w(u,v)} \tag{27}$$

Considering the requirement of differentiability with backpropagation, this section continues the discussion on differentiability of nonuniform rational B-splines by defining the differentiation method for NURBS surfaces in Eqs. (28) and (29):

$$A(u,v)=\sum_{i=0}^{K}\sum_{j=0}^{L}N_{i,d_0}(u)N_{j,d_1}(v)w_{i,j}P_{i,j} \tag{28}$$

$$w(u,v)=\sum_{i=0}^{K}\sum_{j=0}^{L}N_{i,d_0}(u)N_{j,d_1}(v)w_{i,j} \tag{29}$$

Then, according to $(\partial^k+l/\partial u^k\partial v^l)S(u,v)$, Equation (30) can be obtained:

$$\begin{aligned}\frac{\partial^{k+l}}{\partial u^k\partial v^l}S(u,v)&=\frac{\partial^k}{\partial u^k}\frac{\partial^l}{\partial v^l}(A(u,v),w_{(u,v)}^{-1})\\&=\frac{\partial^k}{\partial u^k}(\sum_{j=0}^{l}lC_j\frac{\partial^j}{\partial v^j}A(u,v)\frac{\partial^{l-j}}{\partial v^{l-j}}w(u,v)^{-1})\end{aligned} \tag{30}$$

If each term is finite, see equation (31):

$$\begin{aligned}\frac{\partial^{k+l}}{\partial u^k\partial v^l}S(u,v)&=\frac{\partial^k}{\partial u^k}(\sum_{j=0}^{l}lC_j\frac{\partial^j}{\partial v^j}A(u,v)\frac{\partial^{l-j}}{\partial v^{l-j}}w(u,v)^{-1})\\&=\sum_{j=0}^{l}lC_j\sum_{i=0}^{k}kC_i\frac{\partial^i}{\partial u^i}\frac{\partial^j}{\partial v^j}A(u,v)\frac{\partial^{k-i}}{\partial u^{k-i}}\frac{\partial^{l-j}}{\partial v^{l-j}}w(u,v)^{-1}\end{aligned} \tag{31}$$

3. 3D target modeling for multi-view video

3.1. Feature extraction and parallax estimation based on deep convolutional networks

CNNs, as excellent feature extractors, allow end-to-end classification learning of feature representations from raw image data, thus avoiding the process of human manual feature extraction. Deep CNNs usually have advantages over shallow CNNs when dealing with complex big data problems. Multiple layers of linear and nonlinear processing units stacked in a hierarchical manner provide the ability to learn complex representations at different levels of abstraction. As a result, deep convolutional networks (DCNNs) offer significant performance gains over traditional machine learning models in recognition tasks containing hundreds of categories [27]. The finding that deep architectures can improve the representation capabilities of CNNs improves the use of CNNs in machine learning tasks.

3.1.1. Structure of deep convolutional networks. The convolutional layer is the core of the DCNN and also the biggest differentiator between it and other neural networks. Figure 1 shows a basic DCNN structure.

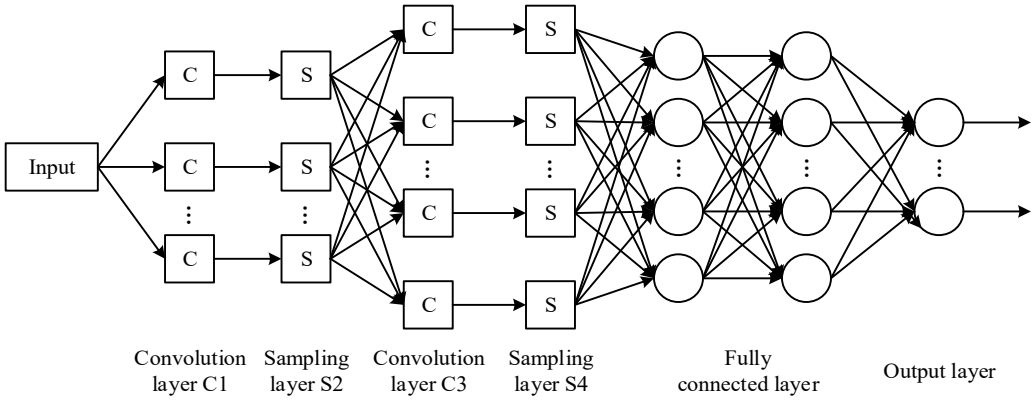


Fig. 1. Basic structure of deep convolutional neural network

As can be seen from Fig. 1, the whole deep convolutional network is mainly composed of input layer, convolutional layer, sampling layer, fully connected layer and output layer, and the convolutional layer and sampling layer are usually regarded as convolutional layer uniformly, and the role of each layer is as follows:

Input layer: input data such as pictures and videos. Take the picture as an example, the input layer is a 24*24*3 matrix, and 3 represents the RGB three channels. Usually before the input will be the original sample data preprocessing, such as normalization, de-mean, whitening and other operations.

Convolutional Layer: In a convolutional layer, each neuron is connected to only a small local subset of the neurons in the previous layer, which is a square region spanning the height and width dimensions. The part used to do the convolution operation is called the convolution kernel and needs to be specified in size. Data of the same size as the convolution kernel needs to be selected from the input matrix to perform the convolution operation, that is, to find the sum of the products of the corresponding positions as the output, when the output can be regarded as another form of the input. Usually, multiple convolution kernels can be designed in the network in order to extract multiple features.

Fully Connected Layer: The fully connected layer is mainly used at the end of the neural network for classification, and is usually added after the convolutional layer. Unlike pooling and convolutional layers, this is a global operation. It takes inputs from the feature extraction stage and globally analyzes the outputs of all the previous layers before nonlinearly combining the selected features and using these features for the classification task.

Fully Connected Layer: Fully connected layer is mainly used at the end of the neural network for classification and is usually added after the convolutional layer. Unlike pooling and convolutional layers, this is a global operation. It takes inputs from the feature extraction phase and globally analyzes the outputs of all the previous layers before nonlinearly combining selected features and using these features for classification tasks.

3.1.2. Feature extraction. In deep convolutional neural networks, the closer to the top layer of the network the more localized the extracted features are, the better the generalization of that part of the features, which in most cases applies to all images. The closer to the bottom layer the more globalized the features obtained from that part are, and the closer the relationship with the dataset used. Taking advantage of this property, in this paper, the convolutional module in the VGG16 model is selected and initialized using the weight parameters after sufficient training in the dataset as a preliminary feature extractor for generalized local feature extraction, after which a convolutional neural network consisting of three convolutional layers, a pooling layer, two fully connected layers, and a Softmax classifier is attached to target the multiview video dataset for training. viewpoint video dataset for training. The detailed design of the deep convolutional network structure used in this paper is shown in Fig. 2.

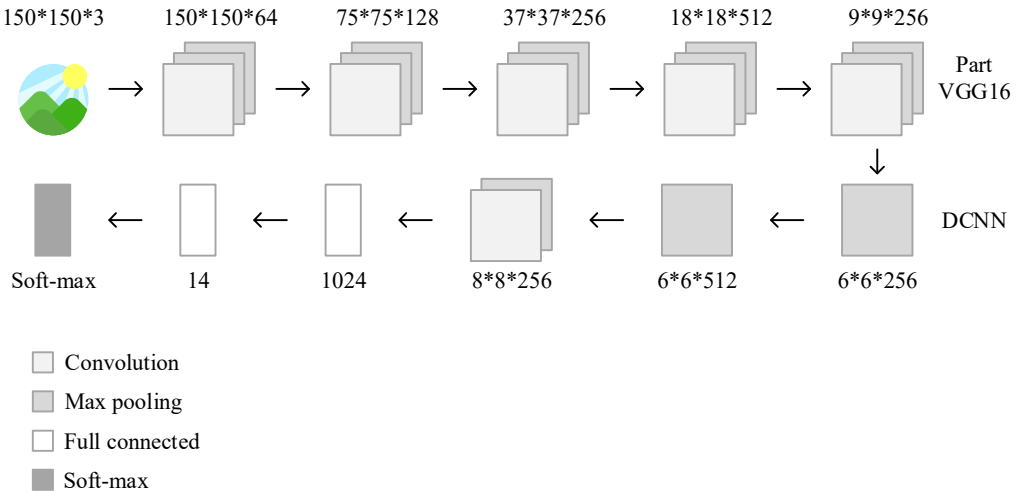


Fig. 2. Network structure of DCNN

First is the convolution module, where in each convolutional layer, the input x_i of the previous layer is convolved using convolution kernel $W_{i,j}$, and the result is processed using the ELU activation function to produce a new output y_j:

$$f(x)=\begin{cases}x, & x\geq 0\\ \alpha(e^{x}-1), & x<0\end{cases} \tag{32}$$

$$y_j^i=f(z^i)=f\left(\sum_{i=1}^{M}(x_i^{i-1}*W_{ij}^i+b_j^i)\right) \tag{33}$$

where l represents the l nd convolutional layer, * represents the convolutional operation, M represents the number of inputs to the layer, and b_j represents the bias of the layer. In the error back propagation stage, the error between the actual output and the desired output is calculated using equation (34):

$$E=\frac{1}{2m}\sum_{j=1}^{m}(y_j-o_j)^2 \tag{34}$$

where, y_j represents the actual output, o_j represents the desired output and m represents the number of samples. In this process, the cumulative error is minimized using Stochastic Gradient Descent (SGD).

In the selection of the activation function, compared to the commonly used ReLU function, the ELU function is selected for activation in this paper to avoid Dead ReLU, i.e., the use of the ReLU function may result in some neurons never being activated. Also, since the output mean of the ELU function is close to 0, the SGD converges faster when this function is used as the activation function.

For the setting of the convolution kernel in the convolution module, unlike the setting of the convolution layer in VGG16 where the size of the convolution kernel is all 3×3, this network chooses an alternating pattern of 1*1, 3*3, and 1*1, where the 1*1 convolution kernel serves to deform the inputs linearly while keeping the dimensionality unchanged, and then nonlinearize them afterwards by the ELU activation function, which reduces the increase of the number of layers tendency of overfitting brought about by the increase in the number of layers. A 2*2 pooling layer is set at the end of the convolutional layer, and the following effects are achieved by the maximum pooling operation:

1) Effectively reduce the number of parameters. The number of parameters of an m*m feature matrix will become (m/a)*(m/a) after pooling by a pooling region with area a*a.

2) Make the translation, rotation, and scaling of the extracted feature image have some invariance. This is because the pooling region, if large enough, ignores some subtle changes within the region, thus making the generalization of features stronger.

In the design of fully connected layers, this paper uses a 1024-dimensional fully connected layer paired with the last fully connected layer consistent with the total number of model categories as the fully connected module of the DCNN. Considering that line drawings contain less information than natural images, compared with the settings in the VGG16 structure, this paper reduces the parameters of the fully connected layer from 2048 to 1024, and reduces the three-layer fully connected layer structure to two layers, in order to reduce the number of neurons and improve the convergence speed.

3.1.3. Parallax estimation:

1) Parallax regression. At the end of the feature extraction stage, the model was designed with a total of four output modules, and for each output module, two 3D convolutions were used to generate a four-dimensional convolution with a channel number of 1, which was then sampled on the cost space. The softargmin algorithm is used in the parallax computation stage, which is done by converting the parallax dimension into a probability space using the softmax function. For each pixel, a vector of length $d_{\max}$ is used, where the elements are the probabilities of all parallax levels p_d. The specific formula is given in (35):

$$p_d = soft\max(x_i) = \frac{e^{x_i}}{\sum_{j=0}^{J} e^{x_j}} \tag{35}$$

The predicted parallax value of the pixel is obtained by multiplying and summing the parallax value of each pixel with the corresponding estimated probability as in Equation (36):

$$d_{pre} = \sum_{d=o}^{d_{\max}} d \cdot p_d \tag{36}$$

where d_{pre} is the predicted parallax value and $d_{\max}$ is the maximum parallax value.

2) Loss function. The parallax estimation in this chapter uses the L1 loss function to train the network.The $smooth_{L_1}$ loss function is a loss function commonly used in regression tasks, especially in cases where there is a large difference between the target values.If the absolute value of the difference between the predicted value and the target value is less than a threshold (often referred to as the smoothing parameter, delta), the loss is one half of the square of the difference. If the difference is greater than the threshold, the loss is the linear difference between the absolute value of the difference and delta L1 loss function as in equation (37):

$$L(d, d_{pre}) = \frac{1}{N}\sum_{i=1}^{N} smooth_{L_1}(d_i - d_{pre}) \tag{37}$$

where N denotes the total number of labeled pixels in the feature map, d_i is the true parallax with labels, and d_{pre} is the predicted parallax of the network. $smooth_L$ function as in equation (38):

$$smooth_{L_1}(x) = \begin{cases} 0.5x^2, & \text{if } |x| < \text{delta} \\ |x| - 0.5, & \text{otherwise} \end{cases} \tag{38}$$

where $x = d_i - d_{pre}$, delta is the threshold value.

3.2. Three-dimensional target modeling

3.2.1. Voxel-based 3D Reconstruction. Voxel-based 3D reconstruction is performed using parallax maps generated by parallax estimation. First, the image is mapped to voxels in 3D space based on the calibration parameters of the camera and the parallax map information. Let $P(x, y, z)$ be a voxel point in 3D space, u, v be pixel coordinates in the image plane, and the mapping relation can be expressed as:

$$z = \frac{f \times B}{D(u,v)} \tag{39}$$

$$x = \frac{(u - u_0) \times z}{f} \tag{40}$$

$$y = \frac{(v - v_0) \times z}{f} \tag{41}$$

where f is the camera focal length, B is the camera baseline length, $D(u,v)$ is the generated parallax map, and u_0, v_0 is the principal point coordinates. The voxel representation of the target object can be obtained by voxel mapping the multi-view image. The voxel model is optimized using a spatial sculpting algorithm to remove the background voxels and redundant voxels to obtain a more accurate voxel of the target object.

3.2.2. Surface reconstruction. Surface reconstruction based on the voxel model is performed using the Marching Cubes algorithm. The algorithm determines the generation of triangular facets by traversing each voxel cell in the voxel model, based on the state of the vertices in the voxel cell (whether they belong to the interior or exterior of the target object). Setting the eight vertices of a voxel cell as $V_0 \sim V_7$ and its state value as $S_i \in \{0,1\}$ (0 means external and 1 means internal), the surface model of the target object is constructed by searching the predefined triangular face sheet configuration table and generating the corresponding triangular face sheet according to the combination of vertex states.

3.2.3. Texture Mapping. After obtaining the surface model of the target object, the texture mapping operation is performed to enhance the realism of the model. First, a suitable texture image is selected from the multi-view video image, and the mapping coordinates of the texture image on the surface model are determined according to the geometry of the surface model and the relationship between the camera viewpoints. Let $T(u,v)$ be the coordinates on the texture image and $P(x,y,z)$ be the points on the surface model, whose mapping relationship can be determined by geometric projection and interpolation algorithms. Then, the color information of the texture image is mapped onto the surface model to obtain a 3D target model with texture.

4. Experimental results and analysis

4.1. Experimental data set

The experimental simulation takes the public dataset BrnoCompSpeed and the real scene measured dataset LzVideoSpeed as the research objects to validate the effectiveness of the deep convolutional network-based multi-view video 3D target modeling method proposed in this paper.

The publicly available dataset BrnoCompSpeed consists of 20 single full-HD videos recorded by traffic surveillance cameras with a duration of about 0.5 hours, which cover 11 angles, with a range of

viewpoints extending from 0° to 180°, and an 18° interval between neighboring viewpoints. Each vehicle consists of 6 Normal Movement (NM) sequences, 2 View Blocking (BG) sequences and 2 Overlapping with Other Vehicles (CL) sequences, for a total of 21,463 vehicles in the video. The speed measurement utilizes LiDAR LiDAR to achieve accurate determination of the optical gate, and the mainstream video speed measurement methods use this value as a benchmark to calculate the error rate. As can be seen from the GB5768.3-2009 specification, the length of the lane demarcation line is fixed in some countries, while the lane demarcation line standard is uneven in another part of the country, and the dataset needs to be calibrated to the datum according to the lane demarcation line related standards defined in the regulations. The subsequent experimental process can be completed under this datum setting for the verification of feature extraction, parallax estimation and other methods.

The measured dataset LzVideoSpeed by the relevant traffic control point road traffic monitoring video, the video covers seven cameras at intervals of 15 ° captured by the Department of the 0 reason to provide, including different roads recorded in 24 hours of heavy ° - 270 ° of the 14 different angles of the vehicle video images, and each viewpoint of each vehicle contains 2 to the normal driving sequence, respectively, the 00 and 01 sequences, the video vehicle The total number of vehicles in the video is 31,752. Subsequent experiments are based on the benchmarks defined by this relevant standard to complete the verification of the universality of the method proposed in this paper in the measured data of real scenarios.

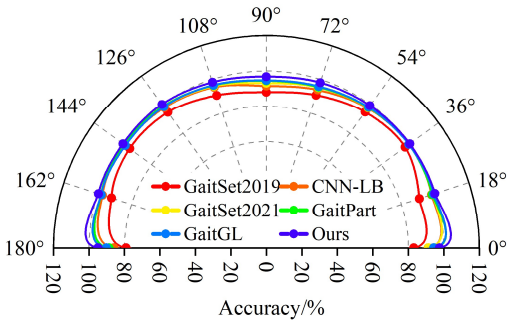
4.2. Feature extraction

In order to verify the effectiveness of the feature extraction module of this paper's algorithm, two datasets, BrnoCompSpeed and LzVideoSpeed, are selected for effective evaluation. The experiments were realized under Ubuntu18.04 system based on pytorch version 1.1.0 with CUDA10.0 and cudnn7.6.5, and the computer was configured with Intel Xeon(R) Sliver4112CPU2.6GHZ, NVIDAGeForce2080Ti graphics card. The profile size of the input network in all experiments is 128×128, the learning rate in model training is 10^{-4}, the batch size of the BrnoCompSpeed dataset is set to (5, 3), and the number of training times for the experiments with small, medium, and large sample sizes are 100K, 80K, and 60K, respectively. when training the LzVideoSpeed dataset, taking into account the large sample size, the batch size is set to (3), and the batch size is set to (3), and the batch size is set to (3). When training the LzVideoSpeed dataset considering its larger sample size, the batch size is set to (3, 3), and the first 100K of the model is trained with a learning rate of 10^{-4}, and the learning rate of 10^{-5} is used for the next 80K.

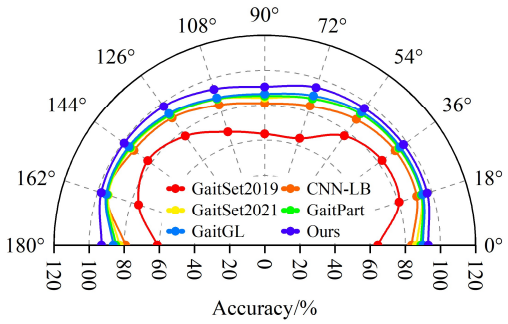
4.2.1. Analysis of results on the BrnoCompSpeed dataset. In order to compare the algorithm of this paper with other methods, the datasets were divided from different data volumes, which were denoted as LT, MT, and ST. The training set consisted of 6 NM sequences, 2 BG sequences, and 2 CL sequences for the first 74, 62, and 24 vehicles, respectively, and the remaining wearing of the gauntlet was divided in the test set. The first 4 NM sequences of each data were used as Gallery set and the remaining 2 NM sequences, BG sequences and CL sequences were used as Probe SET.

The feature extraction accuracy of this paper's algorithm with various other methods on LT, MT, and ST data volumes, respectively, is shown in Fig. 3. The figure represents the LT, MT, and ST experimental results from top to bottom, respectively, and all the results represent the average of the feature extraction accuracies of the data in the positive, side, and fuzzy views. The experimental data and parameter settings of the comparison methods CNN-LB, GaitSet2019, GaitSet2021, GaitPart, and GaitGL in the figure are consistent with the algorithm in this paper. From the experimental results, it can be seen that compared with CNN-LB, GaitSet2019 and other methods, the multi-view feature extraction effect of this paper's algorithm achieves a significant improvement in all angles, whether in a large sample size or in a small sample size. Since the contour appearance is in the side of the vehicle

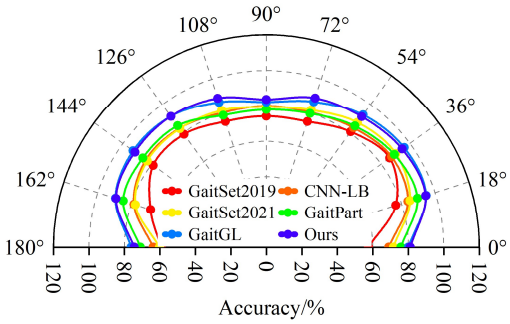
at 90°, there is a large difference between the contour appearance at this time and the other viewpoints, which leads to a decrease in the accuracy of feature extraction, but the accuracy of this paper's algorithm at 90° viewpoints is improved compared to other algorithms. In Figure 3(a), based on the test results in the LT sample size NM state, the accuracy of 90° is improved by 12.91% compared with GaitSet2019, and the accuracy of 0° and 180° is improved by 14.38% and 15.78%, respectively. These three angles also achieve similar or higher improvements in the rest of the graphs. Based on the accuracy rates in BG and CL states with different data sample sizes, it can be seen that the algorithm in this paper has a significant improvement for the phenomenon of feature extraction accuracy reduction due to occlusion of scenery and other vehicles in multiple viewpoints, and the deep convolutional network structure can substantially improve the feature extraction accuracy in BG and CL types in all three types. In the small sample size experiments, as shown in Fig. 3(g), (h), and (i) plots, the accuracies of all angles in the three states are improved compared with GaitSet2019. It proves that the algorithm in this paper remains robust with a small number of data samples.



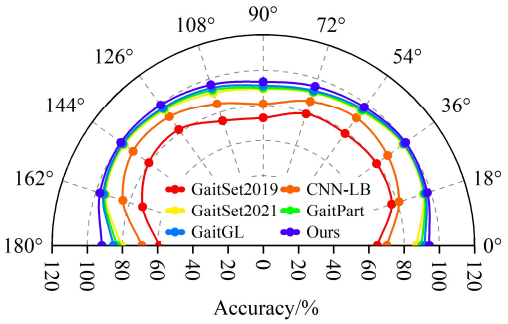
(a) LT test sequence: NM#5-6



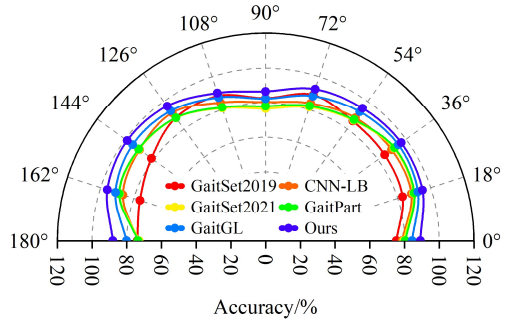
(b) LT test sequence: BG#1-2



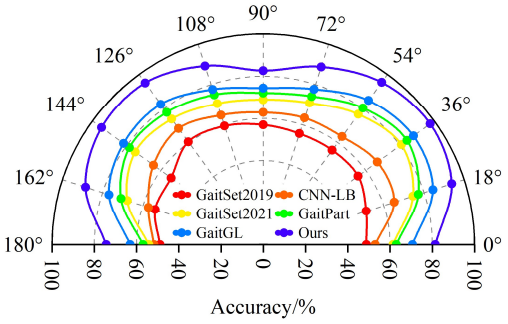
(c) LT test sequence: CL#1-2



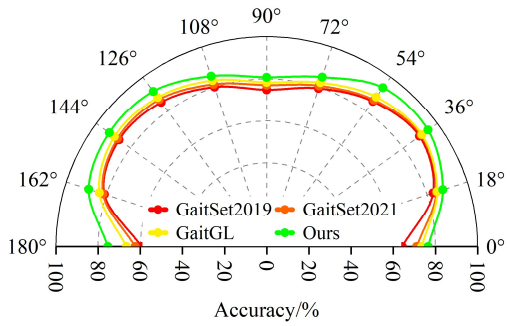
(d) MT test sequence: NM#5-6



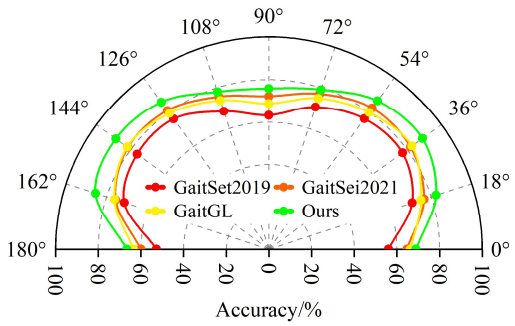
(e) MT test sequence: BG#1-2



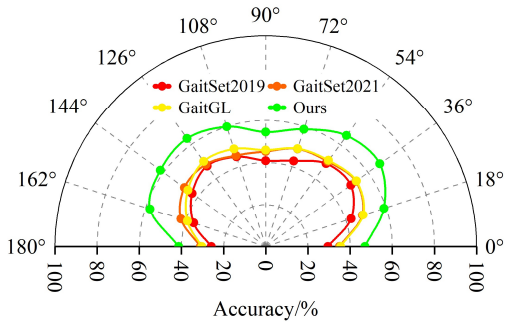
(f) MT test sequence: CL#1-2



(g) ST test sequence: NM#5-6



(h) ST test sequence: BG#1-2



(i) ST test sequence: CL#1-2

Fig. 3. Accuracy of BrnoCompSpeed under three different experimental settings

In the three driving states, the average of all angle accuracies in each state is compared with the typical method, and the results are shown in Table 1. From the data in the table, the feature extraction effect of this paper's algorithm in the three states is BG>CL>NM under LT sample size.Compared with GaitSet2019, the feature extraction accuracy in BG, CL and NM conditions is improved by 20.21%, 20.06% and 36.49%, respectively, and compared with CNN-LB, it is improved by 18.70%, 18.84% and 33.95%, the model performance improvement is obvious because the algorithm in this paper uses deep convolutional network to enhance the feature space richness and improve the representativeness of each feature, and thus achieves the best feature extraction accuracy.The accuracy difference between NM and BG conditions is 12.29% and 12.97% with GaitPart, respectively, and that of CL is as high as 21.37%. In the MT sample experiment, GaitSet2021 improves the accuracy of feature extraction on the original basis, while this paper's algorithm improves 35.27%, 35.71%, and 34.52% over GaitSet2019 in the NM, BG, and CL conditions, respectively, with an average improvement of 35.17%. The results illustrate that the performance of deep convolutional neural network for feature extraction is better than the effect of pixel-level attention and frame-level attention in medium sample size, and the algorithm is still stable with moderate adjustment of sample size. In the ST sample experiments, although the data sample size is not sufficiently large, the accuracy still maintains a respectable level compared with typical methods, and the average accuracy in the three states is improved by 25.94% compared with the GaitSet2019 method and 19.79% compared with the GaitSet2021 method. In all experiments, the feature extraction accuracy is much higher than methods such as CNN-LB, proving the effectiveness and accuracy of the algorithm in this paper.

Table 1. Average for accuracy rates of all angles

Gallery set		NM#1-4		
Probe set		NM#5-6	BG#1-2	CL#1-2
LT	GaitSet2019	72.36	70.83	53.24

	CNN-LB	73.87	72.05	55.78
	GaitSet2021	76.32	73.49	63.97
	GaitPart	80.28	77.92	68.36
	GaitGL	83.46	82.61	75.48
	Ours	**92.57**	**90.89**	**89.73**
MT	GaitSet2019	52.19	51.32	48.76
	GaitSet2021	67.42	64.85	54.37
	GaitGL	73.21	71.44	65.95
	Ours	**87.46**	**87.03**	**83.28**
ST	GaitSet2019	73.22	67.38	41.52
	GaitSet2021	75.47	72.43	52.68
	GaitGL	77.94	75.32	69.74
	Ours	**90.83**	**86.94**	**82.17**

4.2.2. Analysis of results on the LzVideoSpeed dataset. In order to verify the generalization performance of the algorithm, the measured data set LzVideoSpeed is used for the experiments, which is superior to the larger data volume of LzVideoSpeed, considering the larger model of the algorithm, the algorithm selects five angles of 0°, 45°, 90°, 135° and 180° from the data set for the experiments, and the experiments are conducted by using the first 15876 vehicles as the training samples, and the second 15876 vehicles as the test samples, of which the #01 sequence is used as the Gallery set. vehicles as test samples, where #01 sequence is used as Gallery set and #00 sequence is used as Probe set. the results are compared with DigGAN, Input/Output, GaitSet2019, GaitPart, and GaitGL methods. The feature extraction accuracies of all methods in LzVideoSpeed dataset are shown in Fig. 4. As can be seen from the figure, the accuracy of this paper's algorithm under five angles is 84.70%, 87.53%, 90.64%, 90.81%, 89.88%, respectively, which is an obvious advantage over other algorithms. It shows that the feature extraction performance of this paper's algorithm under the selected dataset has obvious superiority.

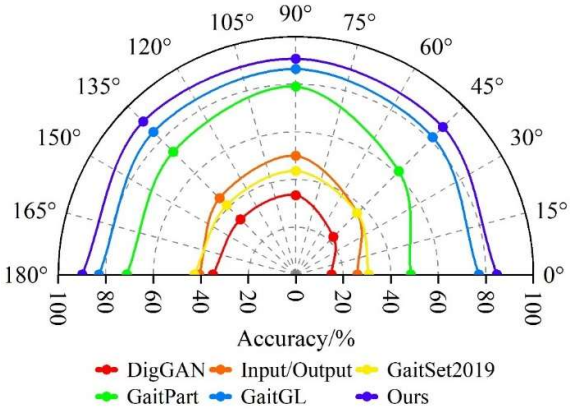


Fig. 4. Average accuracy for various methods under LzVideoSpeed

4.3. Three-dimensional target modeling

4.3.1. BrnoCompSpeed dataset modeling results. The 3D multi-view 3D modeling performance and modeling quality of the DCNN is first demonstrated on the BrnoCompSpeed dataset, evaluated on the basis of the accuracy and completeness of the generated 3D target models, as well as an overall measure (Overall) that combines the accuracy and completeness. The generation process of the multi-view 3D target model used for training is as follows:

1) In order to meet the input requirements of the DCNN, all input images are cropped in a center cropping manner, the resolution of the cropped images is $H = 1082, W = 1560$, and the camera model parameters are adjusted accordingly.

2) The parallax maps corresponding to all input images are estimated using the DCNN. When estimating the parallax maps, the number of sampled parallax planes is $Z = 128$, and the number of viewpoints of the source images used is 6. The parallax maps corresponding to all images are estimated by using all images as reference images in turn.

3) Filter the parallax maps based on the proposed voxel, surface, and texture reconstruction criteria, and then fuse all the parallax maps to construct an initial 3D model, and then filter the initial 3D model to obtain a 3D target model that characterizes the target state.

Comparison with DCNN is made with advanced traditional multi-view 3D target modeling algorithms such as Camp, Furu, Tola, Gipuma, and advanced multi-view 3D target modeling methods based on deep learning such as SurfaceNet, MVSNet, and so on. Table 2 shows the quantitative comparison results of the generated 3D target models, and from the comparison results, it can be seen that the deep convolutional network in this paper achieves the best degree of completeness while guaranteeing a high degree of accuracy, which in turn achieves the optimal overall 3D target modeling.

Table 2. Quantitative comparison results

Method	Accuracy	Completeness	Overall
Camp	0.236	0.462	0.429
Furu	0.348	0.528	0.497
Tola	0.359	0.634	0.536
Gipuma	0.473	0.667	0.604
SurfaceNet	0.532	0.703	0.679
MVSNet	0.718	0.738	0.703
DCNN	**0.892**	**0.877**	**0.847**

4.3.2. Analysis of Modeling Results for LzVideoSpeed Dataset. In order to show the generalization ability of the DCNN method, experiments are conducted on the real dataset LzVideoSpeed in this paper. The parallax maps of the LzVideoSpeed dataset are estimated without any kind of training or fine-tuning of the DCNN using the LzVideoSpeed dataset data. Similar to the BrnoCompSpeed dataset, the LzVideoSpeed dataset is also evaluated based on the generated multiview 3D target model, but uses $F - score$ as the primary evaluation basis.The process of generating the multiview 3D target model on the LzVideoSpeed dataset remains consistent with the BrnoCompSpeed dataset.

Table 3 shows the quantitative evaluation results of the DCNN-based multi-view 3D target modeling approach on the LzVideoSpeed dataset, and among all the state-of-the-art multi-view 3D target modeling approaches reviewed, such as Altizure, ACMH, R-MVSNet, D.R-MVSNet, MVSNet, Pix4D, COLMAP, MVE and others, the DCNN-based DCNN multi-view 3D target modeling approach achieves the most advanced model generation performance. Specifically, among the eight target perspectives, the DCNN-based 3D target model generation scored seven firsts and one second, which gives the DCNN-based 3D modeling method the most advanced performance compared to other methods.

Table 3. LzVideoSpeed data set evaluation results

Method	Mean	0°	20°	40°	60°	80°	100°	120°	140°
DCNN	59.32	**77.31**	55.10	**50.58**	**65.28**	**55.54**	**55.28**	**60.66**	**54.77**
Altizure	56.44	75.06	**61.31**	38.49	61.86	55.19	53.64	56.31	49.65
ACMH	55.10	70.47	49.75	45.28	59.43	53.02	52.41	58.56	51.84
D.R-MVSNet	50.84	73.34	54.67	43.56	44.14	46.88	47.11	51.27	45.73

R-MVSNet	48.67	70.20	47.00	32.95	43.30	52.00	49.24	52.27	42.42
MVSNet	43.64	56.06	28.59	25.16	50.96	54.32	51.00	48.02	34.99
Pix4D	43.49	64.70	32.10	26.73	54.71	51.03	35.48	48.13	35.07
COLMAP	42.48	50.70	22.57	25.91	56.93	45.04	47.44	48.98	42.26
MVE	25.60	49.07	23.96	13.03	5.24	39.84	38.53	5.84	29.32

5. Conclusion

In this paper, deep convolutional networks are used to achieve feature extraction and 3D modeling of target objects in multi-view videos. The model in this paper is compared with other models to illustrate the applicability of the deep neural network-based 3D modeling method in the research scenario of this paper.

1) The model in this paper performs well in feature extraction on the BrnoCompSpeed dataset, and shows the highest feature extraction accuracy when facing different data sizes. The model improves the feature extraction accuracy of vehicles in 90° normal driving condition by 12.91% over the GaitSet2019 model in the largest LT sample size, and the extraction accuracies of 0° and 180° are improved by 14.38% and 15.78%, respectively. Compared with the CNN-LB model, the feature extraction accuracy of this paper's algorithm in the three driving states is improved by 18.70%, 18.84% and 33.95%, respectively. Compared with the improved GaitSet2021 model, the accuracy of this paper's algorithm under the three driving states is improved by 16.25%, 17.40%, and 25.76%, respectively.The accuracy of the model's extraction under five angles, namely, 0°, 45°, 90°, 135°, and 180°, is 84.70%, 87.53%, and 90.64%, respectively, for the LzVideoSpeed real-world dataset, 87.53%, 90.64%, 90.81%, and 89.88% respectively, which is also an obvious advantage over other algorithms, and fully demonstrates the good performance of the model in the task of multi-view feature extraction.

2) The model's 3D target modeling in the BrnoCompSpeed dataset achieves the best completeness while guaranteeing high accuracy, with an overall measure of 0.847, which is a significant improvement over Camp and other 3D modeling methods.The F1 values of this paper in the BrnoCompSpeed dataset for the eight viewpoints are 77.31, 55.10, 50.58, 65.28, 55.54, 55.28, 60.66, 54.77, with an average F1 value of 59.32. Except for the F1 value of this paper's method at 20°, which ranks second, the F1 value of this paper's method at all other viewpoints is the first one among all the participating methods. It proves that the 3D target modeling method in this paper has advanced performance.

References

[1] Zhang, X., Toni, L., Frossard, P., Zhao, Y., & Lin, C. (2019). Adaptive Streaming in Interactive Multiview Video Systems. IEEE Transactions on Circuits and Systems for Video Technology, 29(4), 1130-1144.

[2] Xiao, Y., Chen, J., Wang, Y., Cao, Z., Zhou, J. T., & Bai, X. (2019). Action recognition for depth video using multi-view dynamic images. Information Sciences, 480, 287-304.

[3] Hussain, T., Muhammad, K., Ding, W., Lloret, J., Baik, S. W., & de Albuquerque, V. H. C. (2021). A comprehensive survey of multi-view video summarization. Pattern Recognition, 109, 1-15.

[4] Droste, M., Letellier, J., & Sieck, J. (2018, December). An interactive classical VR concert featuring multiple views. In Proceedings of the Second African Conference for Human Computer Interaction: Thriving Communities (pp. 1-4).

[5] Pan, X., Sun, Q., Wang, J., & Lim, E. G. (2024, August). Game Engine Based Multi-View Video Dataset Synthesis for Pedestrian Detection and Tracking. In 2024 IEEE International Conference on Metaverse Computing, Networking, and Applications (MetaCom) (pp. 259-264). IEEE.

[6] Shimizu, T., Oishi, K., Hachiuma, R., Kajita, H., Takatsume, Y., & Saito, H. (2020). Surgery recording without occlusions by multi-view surgical videos. In 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, VISIGRAPP 2020 (pp. 837-844). SciTePress.

[7] Bridgeman, L., Guillemaut, J. Y., & Hilton, A. (2019). Full-body performance capture of sports from multi-view video. Cvmp.

[8] Sener, F., Chatterjee, D., Shelepov, D., He, K., Singhania, D., Wang, R., & Yao, A. (2022, June). Assembly101: A Large-Scale Multi-View Video Dataset for Understanding Procedural Activities. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 21064-21074). IEEE Computer Society.

[9] Schops, T., Schonberger, J. L., Galliani, S., Sattler, T., Schindler, K., Pollefeys, M., & Geiger, A. (2017). A Multi-view Stereo Benchmark with High-Resolution Images and Multi-camera Videos. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE.

[10] Babaee, M., You, Y., & Rigoll, G. (2017). Combined segmentation, reconstruction, and tracking of multiple targets in multi-view video sequences. Computer Vision and Image Understanding, 100(154), 166-181.

[11] Macheret, P. D., & Savchuk, R. R. (2021). Automated Design Systems Based on the use of Three-dimensional Object Modeling Techniques. 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus). IEEE.

[12] Liao, Z., & Waslander, S. L. (2023). Multi-view 3d object reconstruction and uncertainty modelling with neural shape prior. IEEE.

[13] Aoi, D., Hasegawa, K., Li, L., Sakano, Y., Sakamoto, N., & Tanaka, S. (2024). Edge highlighting of laser-scanned point clouds improves the accuracy of perceived depth in transparent multi-view 3d visualizations. International Journal of Modeling, Simulation & Scientific Computing, 15(1).

[14] Jiaming, Qian, Shijie, Feng, Tianyang, & Tao, et al. (2019). High-resolution real-time 360° 3d model reconstruction of a handheld object with fringe projection profilometry. Optics letters, 44(23), 5751-5754.

[15] Fu, Y., Xiong, H., Dai, X., Nian, X., & Wang, H. (2023, September). Multi-UAV Target Localization Based on 3D Object Detection and Visual Fusion. In International Conference on Autonomous Unmanned Systems (pp. 226-235). Singapore: Springer Nature Singapore.

[16] Chen, S., & Wang, W. (2022). Multi-view 3D Object Detection Based on Point Cloud Enhancement. International Conference on Intelligent Transportation Engineering. Springer, Singapore.

[17] Chen, X., Ma, H., Wan, J., Li, B., & Xia, T. (2017). Multi-View 3D Object Detection Network for Autonomous Driving. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE.

[18] Peng, J., Fu, K., Wei, Q., Qin, Y., & He, Q. (2020). Improved multiview decomposition for single-image high-resolution 3d object reconstruction. Wireless Communications and Mobile Computing.

[19] Wei-Qing, G., Xiao-Gang, W. U., & Yi-Ping, T. (2019). Research of 3d reconstruction based on monocular multi-view panoramic visual perception. Journal of Chinese Computer Systems.

[20] Wang, L., Li, R., Sun, J., Liu, X., & Tandianus, B. (2019). Multi-view fusion-based 3d object detection for robot indoor scene perception. Sensors, 19(19), 4092.

[21] Qi, S., Ning, X., Yang, G., Zhang, L., & Li, W. (2021). Review of multi-view 3d object recognition methods based on deep learning. Displays, 102053.

[22] Chen, D., Li, J., Guizilini, V., Ambrus, R., & Gaidon, A. (2023, June). Viewpoint Equivariance for Multi-View 3D Object Detection. In 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 9213-9222). IEEE Computer Society.

[23] Angelova, A., Konolige, K., Jonschkowski, R., & Liu, X. (2019). KeyPose: Multi-View 3D Labeling and Keypoint Estimation for Transparent Objects. Computer Vision and Pattern Recognition.

[24] He, H. Y. Z. (2017). Moving object recognition using multi-view three-dimensional convolutional neural networks. Neural computing & applications, 28(12).

[25] Xu Guan,Zheng Anqi,Li Xiaotao & Su Jian. (2017). Global vision measurements of object position and orientation with non-coplanar feature points determined by point optimization in camera coordinate system. Applied Optics(1),105-105.

[26] Afzaal Mubashir Hayat,Muhammad Abbas,Farah Aini Abdullah,Tahir Nazir,Hamed Ould Sidi,Homan Emadifar & Amani Alruwaili. (2024). Numerical solutions of generalized Atangana–Baleanu time-fractional FitzHugh–Nagumo equation using cubic B-spline functions. Open Physics(1).

[27] Moon Inkyu & Jaferzadeh Keyvan. (2020). Automated digital holographic image reconstruction with deep convolutional neural networks. THREE-DIMENSIONAL IMAGING, VISUALIZATION, AND DISPLAY 2020.