SCIENCE CHINA
Information Sciences

• POSITION PAPER •

February 2018, Vol. 61 023101:1–023101:16
https://doi.org/10.1007/s11432-017-9234-7

Learning stratified 3D reconstruction

Qiulei DONG[1,3,4], Mao SHU[1], Hainan CUI[1], Huarong XU[1,2] & Zhanyi HU[1,3,4*]

[1]*Institute of Automation, Chinese Academy of Sciences, Beijing* 100190*, China;*
[2]*Department of Computer Science and Technology, Xiamen Institute of Technology, Xiamen* 361024*, China;*
[3]*University of Chinese Academy of Sciences, Beijing* 100049*, China;*
[4]*Center for Excellence in Brain Science and Intelligence Technology, Chinese Academy of Sciences, Beijing* 100190*, China*

Received 22 May 2017/Accepted 8 August 2017/Published online 26 December 2017

Abstract Stratified 3D reconstruction, or a layer-by-layer 3D reconstruction upgraded from projective to affine, then to the final metric reconstruction, is a well-known 3D reconstruction method in computer vision. It is also a key supporting technology for various well-known applications, such as streetview, smart3D, oblique photogrammetry. Generally speaking, the existing computer vision methods in the literature can be roughly classified into either the geometry-based approaches for spatial vision or the learning-based approaches for object vision. Although deep learning has demonstrated tremendous success in object vision in recent years, learning 3D scene reconstruction from multiple images is still rare, even not existent, except for those on depth learning from single images. This study is to explore the feasibility of learning the stratified 3D reconstruction from putative point correspondences across images, and to assess whether it could also be as robust to matching outliers as the traditional geometry-based methods do. In this study, a special parsimonious neural network is designed for the learning. Our results show that it is indeed possible to learn a stratified 3D reconstruction from noisy image point correspondences, and the learnt reconstruction results appear satisfactory although they are still not on a par with the state-of-the-arts in the structure-from-motion community due to largely its lack of an explicit robust outlier detector such as random sample consensus (RANSAC). To the best of our knowledge, our study is the first attempt in the literature to learn 3D scene reconstruction from multiple images. Our results also show that how to implicitly or explicitly integrate an outlier detector in learning methods is a key problem to solve in order to learn comparable 3D scene structures to those by the current geometry-based state-of-the-arts. Otherwise any significant advancement of learning 3D structures from multiple images seems difficult, if not impossible. Besides, we even speculate that deep learning might be, in nature, not suitable for learning 3D structure from multiple images, or more generally, for solving spatial vision problems.

Keywords stratified 3D reconstruction, learning, deep neural networks, outlier detector, spatial vision

Citation Dong Q L, Shu M, Cui H N, et al. Learning stratified 3D reconstruction. Sci China Inf Sci, 2018, 61(2): 023101, https://doi.org/10.1007/s11432-017-9234-7

1 Introduction

Nowadays, computer vision research follows broadly two major directions: the geometry-based direction for spatial vision, exemplified by structure-from-motion (SfM) [1] and visual simultaneous localization and mapping (SLAM) [2], and the learning-based direction for object vision, exemplified by deep neural networks (DNNs) [3]. The goals of these two directions seem to have largely been pursued rather independently.

* Corresponding author (email: huzy@nlpr.ia.ac.cn)

In the last two decades, stratified 3D scene reconstruction from multiple images has been intensively studied, and several state-of-the-art methods have been reported [4]. Owing to its step-by-step upgrading nature — that is, from initial images to intermediate projective reconstruction and affine reconstruction and then to the final metric reconstruction — the robustness and accuracy of the reconstruction results are both enhanced because each step involves a lower number of free parameters to be estimated and the corresponding complicated optimization problem is substantially simplified accordingly. Currently, several well-known scene-reconstruction applications are invariantly geometry-based, for example, streetview, smart3D, and oblique photogrammetry.

In recently years, convolutional neural networks and deep learning have revolutionized object vision, and impressive results have been obtained for object recognition [5], object detection, image matching, image segmentation. However, to the best of our knowledge, learning-based 3D computer vision, in particular, learning-based spatial vision, is still lacking. Therefore, in this study, we explore the feasibility of learning 3D stratified reconstruction from noisy corresponding image points and assess its robustness to matching outliers in an attempt to bridge the geometry with learning.

To the best of our knowledge, no similar study has been reported in the extant literature. Perhaps the studies most closely related to ours include depth learning from single images and camera-pose learning from images. Indeed, there is a large body of literatures on depth learning using single images [6]. In such studies, the main underlying principle is the seemingly successful depth perception by a human vision system using single images. However, the human vision system is a product of long evolution, and its underlying neural mechanism is still far from clear and cannot be reliably used as a computing principle. In addition, such methods suffer from inherent ambiguities. Moreover, depth learning using single images is fundamentally different from our current study on learning stratified 3D reconstruction from corresponding image points.

Another related study is the so-called camera-pose learning from images [7]. In such studies, the camera parameters should remain unchanged. This condition is also fundamentally different from that in our learning method. In our study, we are required to simultaneously learn both the camera parameters and the stratified 3D scene reconstruction.

In recent years, a new learning framework in the deep-learning community, called inverse graphic [8], has emerged and has been used to transform the invariant object-recognition problem into an equivariant one by learning the imaging parameters such as illumination, pose, shape, and texture from the input image. In principle, this framework can be used for our current stratified reconstruction learning problem. However, recovering the imaging parameters is a very complicated inverse problem, and a rigorous theoretical basis still seems to be lacking. We think like the aforementioned depth learning and visual SLAM by learning, the imaging parameters recovered using the inverse graphic can, at best, be "statistically true" in terms of the input with respect to the statistics of the training data but not "absolutely true" in terms of the optimal satisfaction of the inherent geometric constraints as in the geometry-based methods. Furthermore, by "absoluteness", we merely mean that the results do not contain gross errors or apparently false results.

In this study, our objectives are two-fold. Firstly, we explore the feasibility of learning the stratified scene reconstruction from putative image point correspondences by using a specially designed parsimonious neural network. Secondly, we assess whether the learnt reconstruction could be as robust to outliers and random noise as the state-of-the-art SfM methods.

Our preliminary results show that learning stratified 3D reconstruction is feasible, and it is also resistant to some types of outliers and noise, but its robustness is not as good as the state-or-the-art SfM counterparts. We speculate that this is largely owing to the lack of an explicit and integrated outlier-detector that is invariantly embedded in SfM methods. This is also possibly the main reason why the spatial vision in the literature is still geometry-based.

2 Stratified 3D reconstruction

Stratified 3D reconstruction is a major concept in multiple-view geometry [4]. The stratified reconstruction is, to begin with, a projective reconstruction that is then upgraded progressively to an affine reconstruction and finally to a metric reconstruction. The stratified 3D reconstruction can be mathematically described as follows. Given image point correspondences $\boldsymbol{u}_{ij}$ $(i = 1, 2, \ldots, N,\ j = 1, 2, \ldots, M)$ across N images and M space points, we must determine the corresponding N camera projection matrices $\boldsymbol{P}_i (i = 1, 2, \ldots, N)$ and M space points $\boldsymbol{X}_j (j = 1, 2, \ldots, M)$ in homogeneous coordinates such that

$$\boldsymbol{u}_{ij} \approx \boldsymbol{P}_i \cdot \boldsymbol{X}_j, \quad i = 1, 2, \ldots, N, \quad j = 1, 2, \ldots, M, \tag{1}$$

where "$\approx$" indicates the equality up to a scale. $\boldsymbol{P}_i \in \mathbb{R}^{3\times 4}$ is the i^{th} camera projection matrix. $\boldsymbol{X}_j \in \mathbb{R}^{4\times 1}$ are the homogeneous coordinates of the j^{th} space point. Evidently, if (1) holds, for any non-singular matrix $\boldsymbol{Q} \in \mathbb{R}^{4\times 4}$, we have

$$\boldsymbol{u}_{ij} \approx (\boldsymbol{P}_i \boldsymbol{Q}) \cdot (\boldsymbol{Q}^{-1} \boldsymbol{X}_j), \quad i = 1, 2, \ldots, N, \quad j = 1, 2, \ldots, M. \tag{2}$$

Hence, the stratified 3D reconstruction involves reconstructing the scene points and projection matrices using a three-step approach. At first, a projective reconstruction is obtained from image correspondences $\boldsymbol{u}_{ij}$, which differ from the true Euclidean reconstruction by a projective transformation, or $\boldsymbol{Q}$ is a projective transformation matrix. The projective reconstruction is then upgraded to an affine reconstruction in the second step, which differs from the true Euclidean reconstruction by an affine transformation, or $\boldsymbol{Q}$ is an affine transformation matrix. Finally, the affine reconstruction is upgraded to a metric reconstruction, which differs by a similarity transformation from the true Euclidean reconstruction, or $\boldsymbol{Q}$ is a similarity transformation matrix.

Let $\boldsymbol{P}_i^{\text{P}} (i = 1, 2, \ldots, N)$ be the N reconstructed projection matrices in the projective reconstruction of the following form:

$$\boldsymbol{P}_1^{\text{P}} \approx \begin{pmatrix} \boldsymbol{I} & \boldsymbol{0} \end{pmatrix}, \quad \boldsymbol{P}_2^{\text{P}} \approx \begin{pmatrix} \boldsymbol{H}_1 & \boldsymbol{e}_1 \end{pmatrix}, \quad \ldots, \quad \boldsymbol{P}_N^{\text{P}} \approx \begin{pmatrix} \boldsymbol{H}_{N-1} & \boldsymbol{e}_{N-1} \end{pmatrix}, \tag{3}$$

where $\boldsymbol{I}$ is a 3×3 identity matrix. $\boldsymbol{H}_i$ is the homography of a space plane between the $(i+1)^{\text{th}}$ image and the first image, and $\boldsymbol{e}_i$ is the epipole of the optical center of the first camera in the $(i+1)^{\text{th}}$ image.

The projection matrices in the affine reconstruction can then be calculated from the projection matrices in the projective reconstruction by using

$$\boldsymbol{P}_i^{\text{A}} = \boldsymbol{P}_i^{\text{P}} \begin{pmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{a}^{\text{T}} & 1 \end{pmatrix}, \quad i = 1, 2, \ldots, N, \tag{4}$$

where $\boldsymbol{P}_i^{\text{A}}$ is the i^{th} projection matrix in the affine reconstruction, $\boldsymbol{I}$ is a 3×3 identity matrix, and $\boldsymbol{0}$ is a 3D vector, where $\boldsymbol{0} = (0,\ 0,\ 0)^{\text{T}}$. $\boldsymbol{a}$ is a 3D vector that represents the infinite plane in the coordinate system of the projective reconstruction.

Finally, the projection matrices in the affine reconstruction are upgraded to the projection matrices in the metric reconstruction using

$$\boldsymbol{P}_i^{\text{M}} = \boldsymbol{P}_i^{\text{A}} \begin{pmatrix} \boldsymbol{K}_1 & \boldsymbol{0} \\ \boldsymbol{0}^{\text{T}} & 1 \end{pmatrix}, \quad i = 1, 2, \ldots, N, \tag{5}$$

where $\boldsymbol{P}_i^{\text{M}}$ is the i^{th} projection matrix in the metric reconstruction, and $\boldsymbol{K}_1$ is the intrinsic matrix of the first camera.

In summation, after the projective reconstruction is performed using known correspondences of image points, the stratified reconstruction is to determine the infinite plane vector $\boldsymbol{a}$ and the first camera's intrinsic parameter matrix $\boldsymbol{K}_1$ of 5 degrees of freedom under the pinhole camera model.

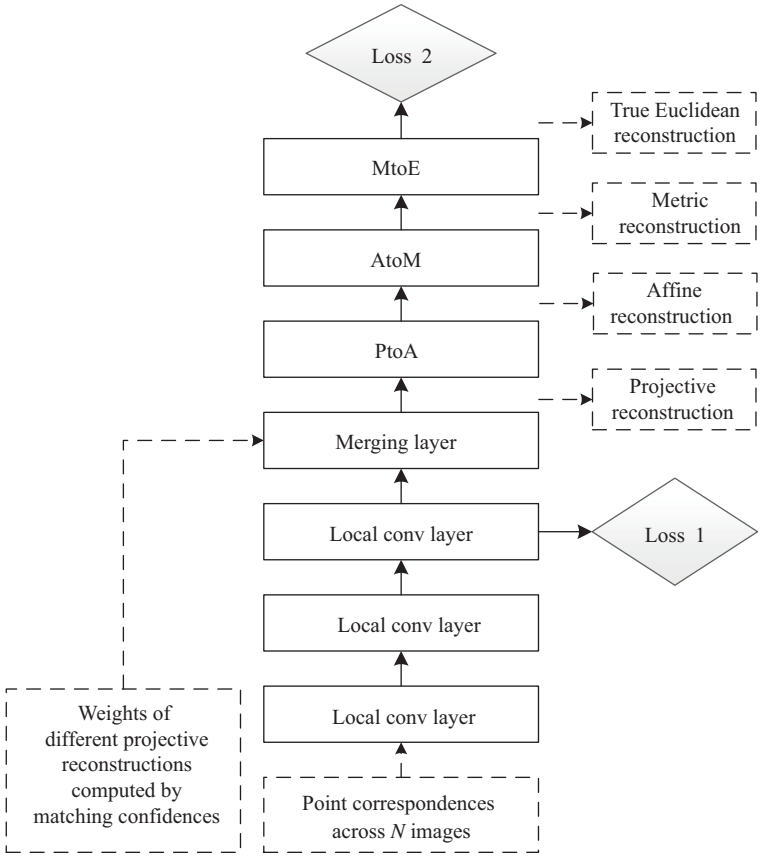


Figure 1 Architecture of our proposed parsimonious network. The inputs are the 2D coordinates of point correspondences across N images, and matching confidences are used for the weight computation of various projective reconstructions. The outputs are 3D stratified reconstructions, including projective reconstruction, affine reconstruction, metric reconstruction, and the final Euclidean reconstruction.

3 Learning 3D stratified reconstruction with a specially designed parsimonious network

In this section, we propose a specially designed parsimonious network architecture for learning the stratified 3D reconstruction. As mentioned above, an efficient reconstruction method should have some resistance to various errors that are mainly caused by point mismatching, missing matching points, and inaccurate feature-point locations. As missing matching points can be considered as a type of mismatch, only two types of errors are considered in this study. The first type of error is point mismatching, which includes gross errors. The second type of error comprises feature-location errors that are introduced by defocusing, edge detection. Feature-location errors are usually small random errors with a magnitude of 1 or 2 pixels; this is because these location errors can be reasonably controlled within a small range owing to the current imaging technology and edge detection techniques.

3.1 Network architecture

As shown in Figure 1, we propose a convolutional neural network for learning 3D stratified reconstruction from noisy point correspondences across images. Our network comprises three local convolution layers, a merging layer, and three sub-networks called PtoA, AtoM, and MtoE. The first three local convolution layers and the merging layer are used for projective reconstruction. The three sub-networks are designed here in a cascaded form, and each sub-network is aimed at learning a particular type of reconstruction. PtoA upgrades a projective reconstruction to an affine reconstruction. AtoM in turn upgrades the affine reconstruction to a metric reconstruction. Finally, MtoE upgrades the metric reconstruction to a

true Euclidean reconstruction. Rectified linear units are adopted as the activation functions of hidden layers. The real 3D points corresponding to the image point correspondences at the training stage are used as supervised information. The inputs to our network are point correspondences across N images and weights of various projective reconstructions computed by matching confidences. The outputs of our network are stratified 3D reconstructions including projective reconstruction, affine reconstruction, metric reconstruction, and the final Euclidean reconstruction. The detailed description of the network used in this study is presented in the following subsections.

The data of each layer in our network is a 3D array of size $h \times w \times d$, where h and w denote spatial dimensions, and d is the number of channels. For example, the input point correspondences across N images is of size $N \times 2 \times 1$.

3.2 Projective reconstruction

In Figure 1, the three local convolution layers and the merging layer that performs a weighted linear summation operation are used for projective reconstruction. The filters in the local convolution layers do not share weights. The number of channels of the three local convolution layers are 64, 64, and 4 respectively. As missing matching points and mismatching (hereinafter referred to as outliers) are inevitable in practice, it is necessary for our network to possess some type of outlier resistance. To tackle this problem, we adopt a strategy of multiple reconstructions, each of which is achieved with the help of a subset of the input matching points, and hopefully at least one of the reconstructions is obtained from pure inliers. In theory, a projective reconstruction requires only a pair of images. In order to balance the desire for robustness with the network complexity, in this study, each projective reconstruction is computed from the matching points in K consecutive images. In other words, we assume that for a set of N putative input matching points, there exists at least one subset of K ($K \ll N$) consecutive matching points that consists of only inliers. Hence, given N input matching points, we have

$$M = N - K + 1, \tag{6}$$

where M is the number of different projective reconstructions. This multi-reconstruction scheme is implemented by setting the receptive field of the third local convolution layer as $K \times 2$. The reception field of a layer is determined from the kernel sizes and the strides of the previous layers

$$\mathbf{RF}_i = \begin{cases} (\mathbf{RF}_{i-1} - [1,1]) \times s_i + \boldsymbol{k}_i, & i > 1, \\ \boldsymbol{k}_i, & i = 1, \end{cases} \tag{7}$$

where $\mathbf{RF}_i$ is a 2D vector that represents the reception field of the i^{th} layer, s_i is the stride of the i^{th} layer, and $\boldsymbol{k}_i$ is a 2D vector that represents the kernel size of the i^{th} layer.

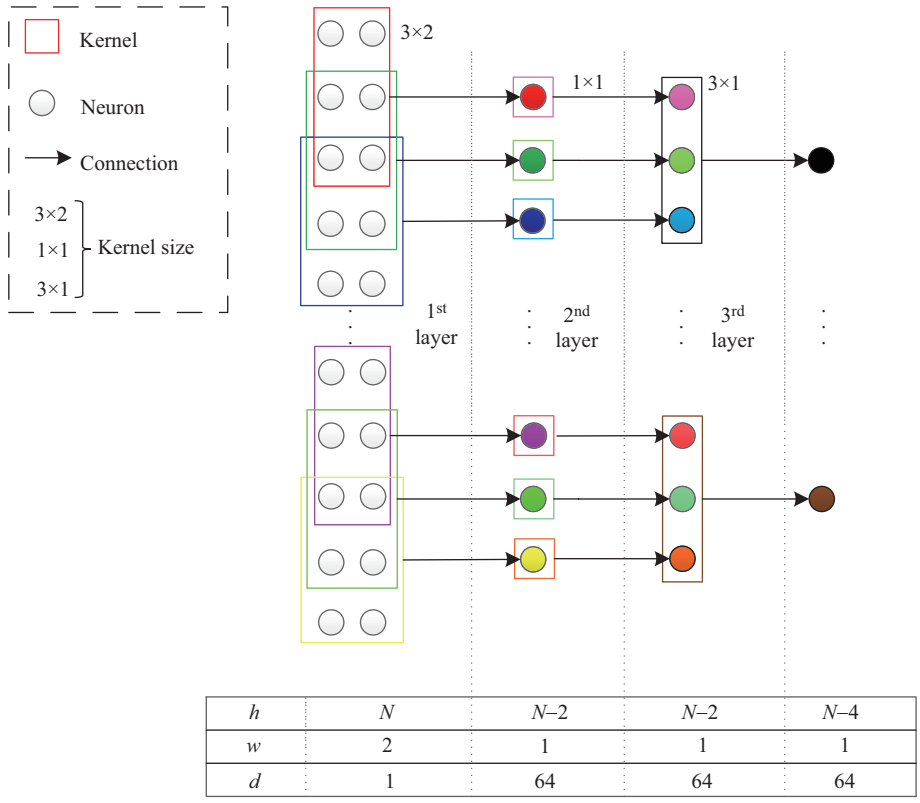
The following is an illustrative example. If we set the kernel sizes of the three local convolution layers as 3×2, 1×1, and 3×1 respectively, and set the strides of the three local convolution layers to 1, then the receptive field of the third local convolution layer is 5×2, i.e., $K = 5$, and $M = N - 4$. The details of the implementation of the three local convolution layers are shown in Figure 2.

It should be noted that the M projective reconstructions are not necessarily performed using a same coordinate system, but they should be aligned under a same coordinate system for the subsequent affine reconstruction. The following structural loss function, which constrains each pair of projective reconstructions $\{\boldsymbol{X}_i^{\text{P}}, \boldsymbol{X}_j^{\text{P}}\}$ to be as close as possible to each other, is adopted to align the M projective reconstructions under a same coordinate system during learning

$$\text{Loss1} = \frac{1}{2} \sum_{i=1}^{M} \sum_{j=i+1}^{M} w_i w_j (\boldsymbol{X}_i^{\text{P}} - \boldsymbol{X}_j^{\text{P}})^2, \tag{8}$$

where $\boldsymbol{X}_i^{\text{P}}$ is the estimated i^{th} projective reconstruction, and w_i is the weight of the i^{th} projective reconstruction that is defined as follows:

$$w_i = \frac{\hat{w}_i}{\sum_{i=1}^{M} \hat{w}_i}, \quad \hat{w}_i = \prod_{j=i}^{i+K-1} c_j, \quad i = 1, 2, \ldots, M, \tag{9}$$



h	N	N–2	N–2	N–4
w	2	1	1	1
d	1	64	64	64

Figure 2 (Color online) Implementation of the three local convolution layers in the proposed network. The inputs to the first layer are point correspondences of size $N \times 2 \times 1$. The kernel sizes of the three layers are 3×2, 1×1, and 3×1, respectively. The channel numbers of the three layers are 64, 64, and 4, respectively. The dimensions of outputs at the three layers are $(N-2) \times 1 \times 64$, $(N-2) \times 1 \times 64$, and $(N-4) \times 1 \times 4$, respectively. Kernels presented in different colors have different weights.

where c_j is the confidence of the j^{th} point correspondence. The confidences are assumed to be known and determined during the point-matching stage. In this study, we merely assume that, if the corresponding point is missing or is an outlier, the associated confidence is 0; else, the otherwise associated confidence is set as 1.

The structural loss in (8) ensures that all the M projective reconstructions are performed under the same coordinate system such that the different projective reconstructions can be merged into the final projective reconstruction. In our current implementation, the merging layer is obtained from a weighted linear summation. The final projective reconstruction $\boldsymbol{X}^{\text{P}}$ is then merged from the M projective reconstructions by using the merging layer as follows:

$$\boldsymbol{X}^{\text{P}} = \sum_{i=1}^{M} w_i \boldsymbol{X}_i^{\text{P}}. \tag{10}$$

3.3 Affine reconstruction

The sub-network PtoA in Figure 3 is used to upgrade the projective reconstruction $\boldsymbol{X}^{\text{P}}$ to an affine reconstruction $\boldsymbol{X}^{\text{A}}$. According to (4), in order to upgrade the projective reconstruction to an affine one, we are only required to learn the parameter vector $\boldsymbol{a}$ of the infinite plane.

Figure 3 shows the architecture of PtoA, which is composed of a slice layer, a fully connected (FC) layer, and a concat layer. The slice layer is a utility layer that slices an input into multiple outputs along a given dimension. Here, the slice layer slices the input 4D homogeneous coordinates into two parts. One part contains the first three elements of the input projective reconstruction, which are directly passed to the concat layer. The other part contains only the fourth element of the input projective reconstruction, which is discarded. The FC layer connects the input 4D homogeneous coordinates to one single neuron; it

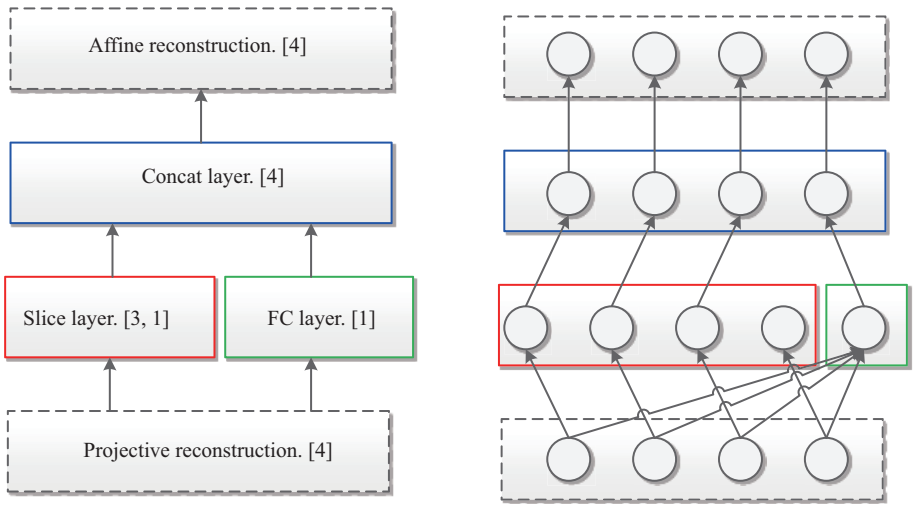


Figure 3 (Color online) Architecture of PtoA, which upgrades the projective reconstruction to the affine reconstruction. The numbers in brackets are the number of neurons in each layer. The infinite plane vector is learned implicitly in the FC layer.

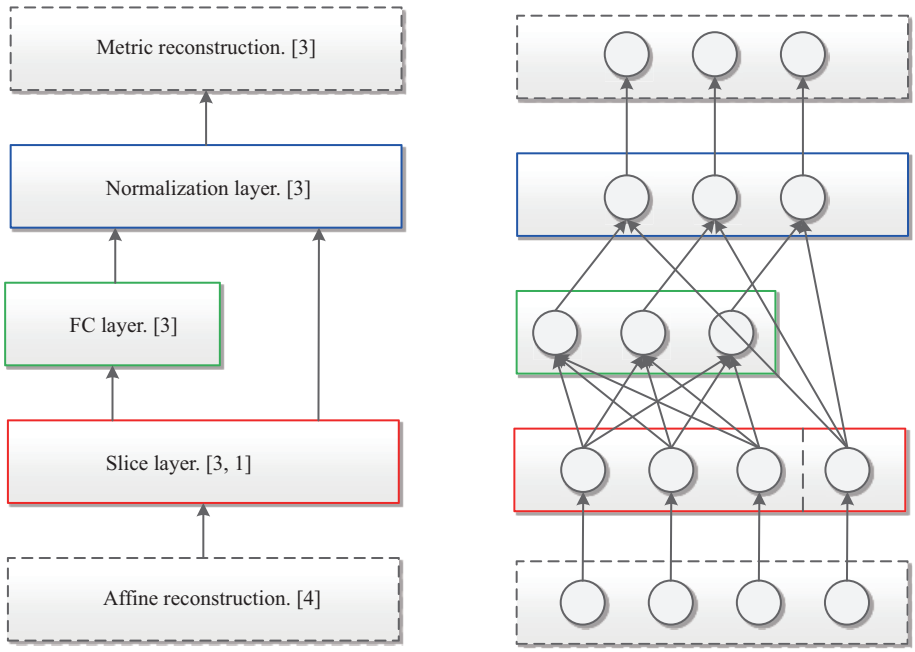


Figure 4 (Color online) Architecture of AtoM, which upgrades the affine reconstruction to the metric reconstruction. The numbers in brackets are the number of neurons in each layer. The intrinsic matrix $\boldsymbol{K}_1$ is learned implicitly in the FC layer.

learns the infinite plane vector $\boldsymbol{a}$ implicitly by learning the fourth coordinate of the affine reconstruction. The concat layer concatenates the three elements from the slice layer and the output from the FC layer as the affine reconstruction.

3.4 Metric reconstruction

In order to upgrade the affine reconstruction $\boldsymbol{X}^{\mathrm{A}}$ to a metric reconstruction $\boldsymbol{X}^{\mathrm{M}}$, sub-network AtoM in Figure 4 is used to learn the intrinsic matrix $\boldsymbol{K}_1$ in (5), which is functionally equivalent to the implementation of the affine transformation.

AtoM is composed of a slice layer, an FC layer, and a normalization layer. The slice layer slices the input 4D homogeneous coordinates into two parts. One part contains the first three elements of the input

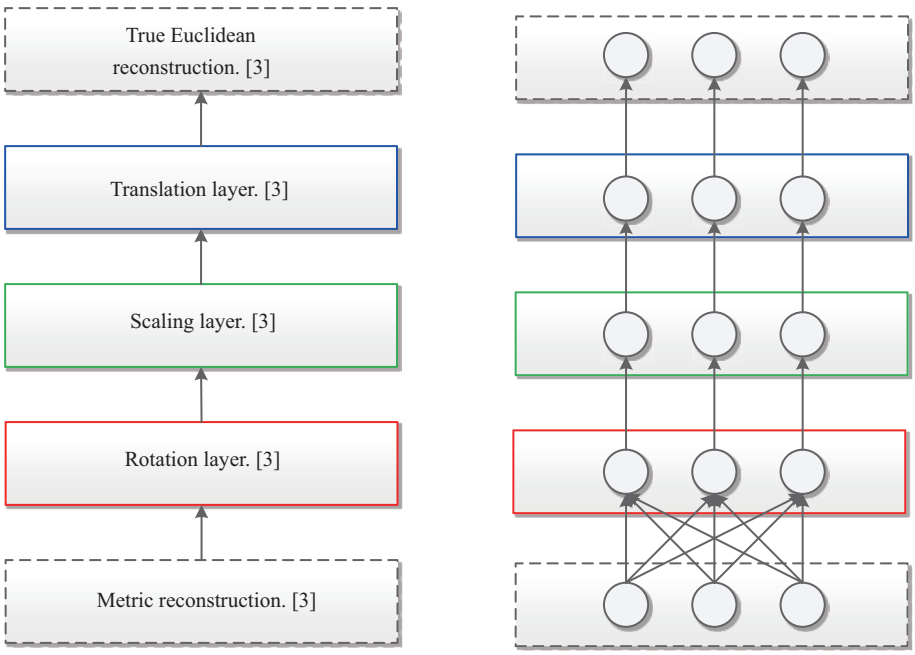


Figure 5 (Color online) Architecture of MtoE. The network transforms the metric reconstruction to the true Euclidean reconstruction. The numbers in brackets are the number of neurons in each layer. MtoE learns a similarity transformation that includes three operations: rotation, scaling, and translation.

affine reconstruction, which are passed to the FC layer. The other part contains only the fourth element of the input affine reconstruction, which is passed to the normalization layer. The FC layer has three neurons, each of which connects to all the first three elements of the input affine reconstruction. The FC layer implicitly learns the intrinsic matrix by learning the first three elements of the metric reconstruction. In the normalization layer, the three elements from the FC layer are divided by the fourth element from the slice layer. Finally, the three inhomogeneous coordinates of the metric reconstruction are obtained from the output of the normalization layer.

3.5 Final Euclidean reconstruction

The metric reconstruction $\boldsymbol{X}^{\mathrm{M}}$ can be transformed into the true Euclidean reconstruction $\boldsymbol{X}^{\mathrm{E}}$ through a similarity transformation

$$\boldsymbol{X}^{\mathrm{E}} = s\boldsymbol{R}(\boldsymbol{X}^{\mathrm{M}}) + \boldsymbol{t}, \tag{11}$$

where s is a non-zero scalar, $\boldsymbol{R}$ is a 3×3 rotation matrix, and $\boldsymbol{t}$ is a 3D translation vector. The rotation is defined using the three Euler angles α, β, and γ. As shown in Figure 5, the rotation, scaling, and translation are learnt in a layer-by-layer manner.

The rotation layer is an FC layer. It has three neurons each of which connects to all the neurons in the input layer. The rotation layer is to implicitly remove the rotation transformation (determined using Euler angles α, β, and γ) between the estimated metric reconstruction and the true Euclidean reconstruction. The rotation layer is implemented as follows. Eq. (12) specifies the forward propagation process, and (13), (14) specify the backward propagation process. In the forward propagation process, the output $\boldsymbol{X}^{\mathrm{R}}$ is calculated using the input $\boldsymbol{X}^{\mathrm{M}}$ and the parameters α, β, and γ. In the backward propagation process, errors $\delta\boldsymbol{X}^{\mathrm{M}}$, $\delta\alpha$, $\delta\beta$, and $\delta\gamma$ are calculated using the known $\delta\boldsymbol{X}^{\mathrm{R}}$.

$$\boldsymbol{X}^{\mathrm{R}} = \boldsymbol{R}_x(\alpha)\boldsymbol{R}_y(\beta)\boldsymbol{R}_z(\gamma)\boldsymbol{X}^{\mathrm{M}}, \tag{12}$$

$$\delta\boldsymbol{X}^{\mathrm{M}} = [\boldsymbol{R}_x(\alpha)\boldsymbol{R}_y(\beta)\boldsymbol{R}_z(\gamma)]^{\mathrm{T}}\delta\boldsymbol{X}^{\mathrm{R}}, \quad \delta\alpha = \left[\frac{\partial\boldsymbol{R}_x(\alpha)}{\partial\alpha}\boldsymbol{R}_y(\beta)\boldsymbol{R}_z(\gamma)\boldsymbol{X}^{\mathrm{M}}\right]^{\mathrm{T}}\delta\boldsymbol{X}^{\mathrm{R}}, \tag{13}$$

$$\delta\beta = \left[\boldsymbol{R}_x(\alpha)\frac{\partial\boldsymbol{R}_y(\beta)}{\partial\beta}\boldsymbol{R}_z(\gamma)\boldsymbol{X}^{\mathrm{M}}\right]^{\mathrm{T}}\delta\boldsymbol{X}^{\mathrm{R}}, \quad \delta\gamma = \left[\boldsymbol{R}_x(\alpha)\boldsymbol{R}_y(\beta)\frac{\partial\boldsymbol{R}_z(\gamma)}{\partial\gamma}\boldsymbol{X}^{\mathrm{M}}\right]^{\mathrm{T}}\delta\boldsymbol{X}^{\mathrm{R}}. \tag{14}$$

The scaling layer is a locally connected layer with shared weights. It has three neurons representing three coordinates $\boldsymbol{X}^s$, which are defined as

$$\boldsymbol{X}^s = s\boldsymbol{X}^{\mathrm{R}}. \tag{15}$$

Each neuron is connected to one neuron of the rotation layer, and the weights of all the three connections are shared. The scaling layer learns the scaling parameter s implicitly.

The translation layer is also a locally connected layer. It has three neurons that represent three coordinates $\boldsymbol{X}^{\mathrm{E}}$, which are defined as

$$\boldsymbol{X}^{\mathrm{E}} = \boldsymbol{X}^s + \boldsymbol{t}. \tag{16}$$

Each neuron connects to one neuron of the scaling layer. The translation layer learns the 3D translation vector $\boldsymbol{t}$ implicitly.

Finally, the following loss function is used to measure the discrepancy between the learnt reconstructed points $\boldsymbol{X}^{\mathrm{E}}$ with the ground truth $\bar{\boldsymbol{X}}$:

$$\mathrm{Loss2} = \begin{cases} 0.5(y)^2, & \text{if } |y| < 1, \\ |y| - 0.5, & \text{otherwise,} \end{cases} \qquad y = \boldsymbol{X}^{\mathrm{E}} - \bar{\boldsymbol{X}}. \tag{17}$$

Our method is implemented using Caffe. We train our models using a stochastic gradient descent with a momentum of 0.9 and weight decay of 0.005. The weights of local convolution layers and FC layers are initialized from a zero-mean Gaussian distribution with a standard deviation 0.01. The learning rate is initialized as 0.0001.

4 Experiments

In this section, we evaluate the performance of our trained model for simulation datasets and real-world datasets respectively. For simulation datasets, our experiments are divided into two parts. First, we explore the feasibility of learning the stratified 3D reconstruction from known point correspondences across images. We then add two types of noise into the test data in order to simulate feature location errors and matching outliers, and assess whether our learnt model could be as robust as OpenMVG, which is a state-of-the-art incremental SfM approach. For real-world datasets, we use three benchmark datasets Herz-Jesu-P8, Fountain-P11, and Castle-P30 to evaluate our model and compare it with 10 traditional geometry-based methods: FUR, ST6, ZAH, ST4, VU, TYL, SAL, TYL09, JAN, and JAN09.

4.1 Simulation experiment

The used simulation dataset consists of $25K$ 3D space points distributed uniformly in a cube of volume 10 m $\times$ 10 m $\times$ 10 m. $15K$ of the data points are used for the training, and the rest are used for the test. Thirty virtual cameras with different extrinsic parameters are used. We learn the corresponding 3D points from the 2D image points projected from the simulated 3D points by using

$$\boldsymbol{P}_i = \boldsymbol{K}_i\boldsymbol{R}_i\left[\boldsymbol{I}|-\boldsymbol{C}_i\right], \tag{18}$$

where $\boldsymbol{K}_i$ is the i^{th} camera intrinsic matrix, $\boldsymbol{R}_i$ is the rotation matrix that represents the orientation of the i^{th} camera, $\boldsymbol{C}_i$ represents the coordinates of the i^{th} camera center in the world coordinate system, and $\boldsymbol{P}_i$ is the i^{th} camera projection matrix. The focal lengths of all the 30 cameras are 1500, and the image resolution of each camera is of 1000 pixels $\times$ 1000 pixels. Thus, the intrinsic matrix $\boldsymbol{K}_i$ of each camera is given by

$$\boldsymbol{K}_i = \begin{bmatrix} 1500 & 0 & 500 \\ 0 & 1500 & 500 \\ 0 & 0 & 1 \end{bmatrix}, \quad i = 1, 2, \ldots, 30. \tag{19}$$

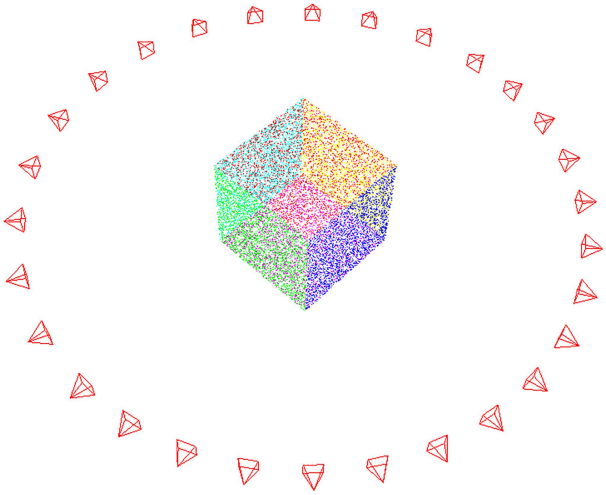


Figure 6 (Color online) 3D points and cameras locations in the simulation experiment. There are 30 cameras; the optical center of each camera is evenly distributed along the circumference of a circle, and each optical axis is oriented towards the center of the cube.

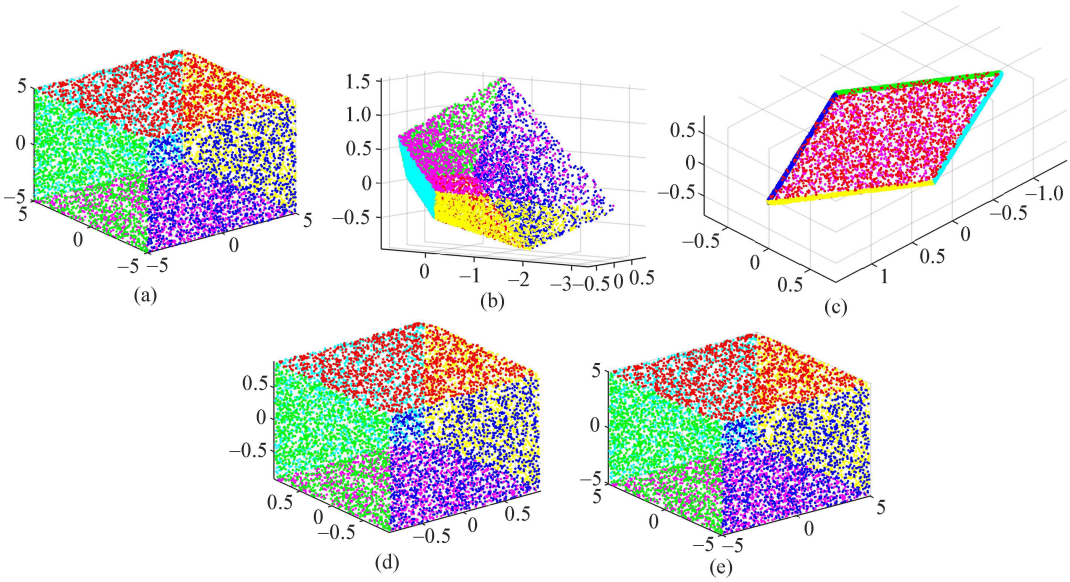


Figure 7 (Color online) Stratified 3D reconstruction result of the simulation test data. (a) Ground truth; (b) projective reconstruction; (c) affine reconstruction; (d) metric reconstruction; (e) true Euclidean reconstruction.

The optical center of each camera is evenly distributed along the circumference of a circle with a radius of 24 m. Each optical axis is oriented towards the center of the cube, as shown in Figure 6.

First, we investigate the feasibility of learning the stratified 3D reconstruction from the above simulated image point correspondences. For the projective reconstruction, we set the receptive field of the third local convolution layer as 5×2, i.e., $K = 5$, as in 6. Hence, the kernel sizes of the three local convolution layers are set as 3×2, 1×1, and 3×1 accordingly, and the strides of the three local convolution layers are set to 1. In other words, we assume that for each space point, out of its 30 projected image points, there is at least one subset of 5 consecutive image points that comprises only inliers. The projective reconstruction, affine reconstruction, metric reconstruction, and true Euclidean reconstruction are shown in Figure 7.

In order to evaluate the performance of our trained model, the root mean square error (RMSE) is

Table 1 Distribution parameters of uniform noise

Uniform noise	Distribution of parameter
Feature location errors	±U(1,2)
Outliers	±U(100,500)

Table 2 RMSE of the true Euclidean reconstruction for the following three cases: noise-free data, data with feature location errors, and data with outliers

	Proportion	RMSE (m)
Noise-free data	-	0.008
Feature location errors	30/30	0.028
Outliers	1/30	0.008
	2/30	0.009
	4/30	0.010
	6/30	0.012
	8/30	0.016
	10/30	0.019

adopted as follows:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} \left(\boldsymbol{X}_i^{\mathrm{E}} - \bar{\boldsymbol{X}}_i\right)^2}{n}}, \tag{20}$$

where $\boldsymbol{X}_i^{\mathrm{E}}$ is the learnt true Euclidean reconstruction, $\bar{\boldsymbol{X}}_i$ is the ground truth, and n is the number of 3D points.

Second, we assess whether our learnt model could also be as robust to noises and outliers as the traditional geometry-based methods. We include uniform random noises into the test data in order to simulate feature location errors and outliers. The noise distribution parameters are shown in Table 1. Feature location errors are simulated using random noise distributed uniformly within $[-2, -1]$ and $[1, 2]$ intervals. The outliers are simulated by using uniform random noise distributed within $[-500, -100]$ and $[100, 500]$ intervals. In the simulation, the confidence for points with location errors is set to 1, and the confidence for an outlier is set to 0.

In order to measure the robustness and accuracy of our model, we evaluate the learnt true Euclidean reconstruction against the ground truth using RMSE measurements. Table 2 lists the RMSE results of three types of test data: noise-free data, data with only small location errors, and data containing outliers. The "proportion" represents the number of noisy point correspondences out of 30, and the following six proportions are assessed: 1/30, 2/30, 4/30, 6/30, 8/30, and 10/30. As can be observed, the RMSE of our model in the case of the noise-free data is 0.008 m; the small reconstruction error indicates that the learnt reconstruction results are highly accurate. Moreover, the RMSE obtained for our model with feature location errors is 0.028 m, and that with outliers increases smoothly with the increase in the proportion of outliers. The RMSE increases from 0.008 to 0.019 m when the proportion of outliers increases from 1/30 to 10/30. These results show that learning stratified 3D reconstruction from noisy image point correspondences is feasible, and our model exhibits some outlier and noise resistance.

In order to compare our model's robustness with that of traditional SfM methods, we use a state-of-the-art SfM method, OpenMVG, for comparison. We use the reprojection error as the measurement criterion, the results of which are listed in Table 3. As can be observed, in the case of feature location errors, the reprojection error of our model is 1.88 pixels, while the reprojection error for OpenMVG is 1.64 pixels. In the case of outliers, the reprojection error of our model increases smoothly with the increase in the proportion of outliers. The reprojection error increases from 0.85 to 1.12 pixels when the proportion of outliers increases from 1/30 to 10/30. The reprojection error for OpenMVG shows no obvious variation under various proportions of outliers and remains less than 0.6 pixels.

Therefore, our model exhibits some outlier and noise resistance, but the robustness of our model is not as good as that of OpenMVG owing largely to its lack of an explicit and integrated outlier-detector, which is invariantly embedded in SfM methods.

Table 3 Comparison of reprojection error (pixels) for three cases: noise-free data, data with feature location errors, and data with outliers

	Proportion	Our model	OpenMVG
Noise-free data	-	0.85	0.55
Feature location errors	30/30	1.88	1.64
Outliers	1/30	0.85	0.50
	2/30	0.86	0.55
	4/30	0.88	0.55
	6/30	0.93	0.57
	8/30	1.02	0.58
	10/30	1.12	0.59

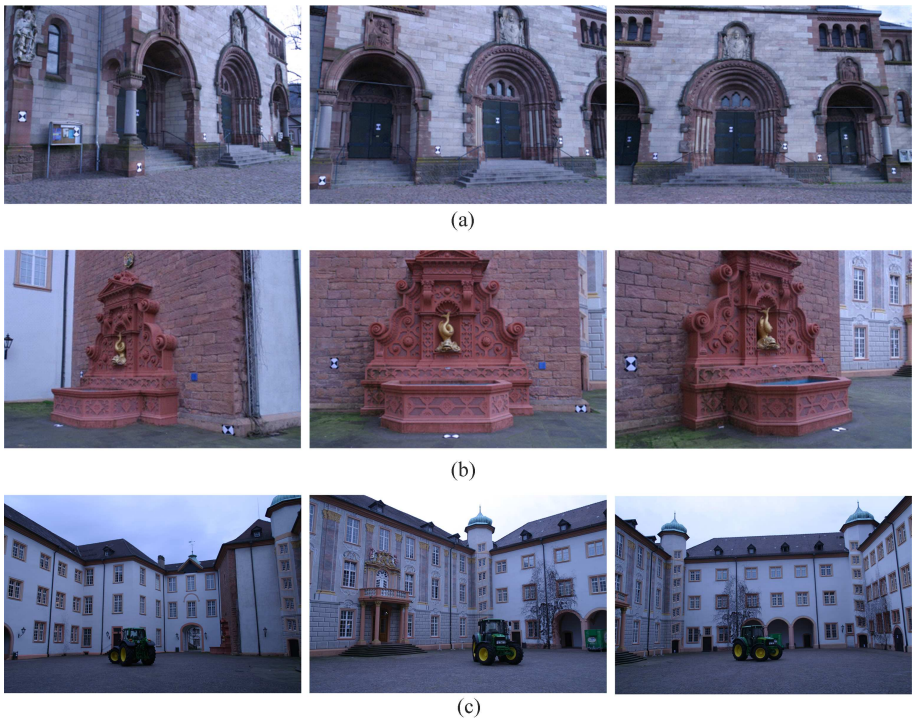


Figure 8 (Color online) Images in the multi-view-stereo datasets. (a) Three images from Herz-Jesu-P8; (b) three images from Fountain-P11; (c) three images from Castle-P30.

4.2 Real experiments

In this subsection, we evaluate our model using real scenes with respect to 10 traditional geometry-based methods: FUR, ST6, ZAH, ST4, VU, TYL, SAL, TYL09, JAN, and JAN09.

In order to evaluate our model using real scenes, we use multi-view-stereo benchmark datasets[1)], including Herz-Jesu-P8, Fountain-P11, and Castle-P30, as shown in Figure 8. In the case of Herz-Jesu-P8, eight images are used, $10K+$ 3D points are used for the training, and $10K$ 3D points are used for the test. In the case of Fountaion-P11, 11 images are used, and the number of training and test datasets are $10K+$ and $9K$ respectively. In the case of Castle-P30, 30 images are used, and the number of training and test datasets are $10K+$ and $8K$ respectively. As the number of cameras and the intrinsic and extrinsic parameters of the three scenes are different, we trained our model using three datasets separately.

In the case of Herz-Jesu-P8 and Fountain-P11, the corresponding image points are generated from the known 3D points with their provided camera projection matrices in our experiments. In the case of Castle-P30, the ground-truth 3D points are the reconstructed points obtained using OpenMVG. However, in a strict sense, such points lack ground truth. Hence, for this dataset, our aim is to assess whether

1) https://icwww.epfl.ch/ marquez/multiview/denseMVS.html.

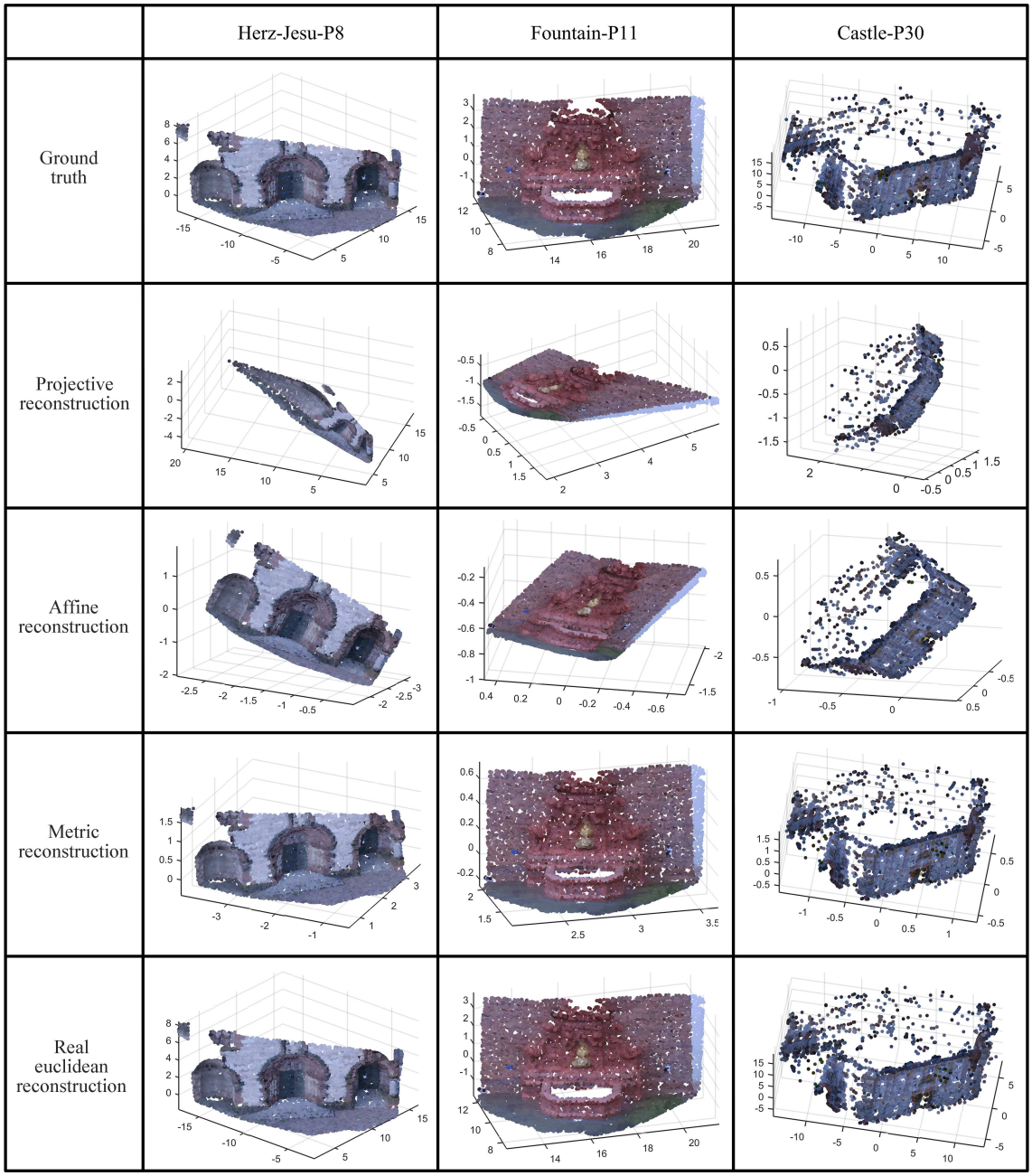


Figure 9 (Color online) Stratified 3D reconstruction results of multi-view-stereo benchmark datasets. The results, from top to bottom, are the ground truth, projective reconstruction, affine reconstruction, metric reconstruction, and true Euclidean reconstruction. The scenes, from left to right, are that of Herz-Jesu-P8, Fountain-P11, and Castle-P30.

our method is also workable for such long-feature-track datasets, the accuracy comparison is not as meaningful as for the other two datasets. Furthermore, in the case of Herz-Jesu-P8, the lengths of the image point correspondences are shorter than 8. Therefore, we set the receptive field of the third local convolution layer as 3×2, i.e., $K = 3$ in (6). In other words, we use 3D points that have at least three consecutive corresponding image points, and hence, the kernel sizes of the three local convolution layers in the proposed network are set as 3×2, 1×1, and 1×1 respectively, and the strides of the three local convolution layers are set as 1. In the case of Fountain-P11 and Castle-P30, we set the receptive field of the third local convolution layer as 5×2, i.e., $K = 5$ in (6). Hence, the kernel sizes of the three local convolution layers are set as 3×2, 1×1, and 3×1 respectively, and the strides of the three local convolution layers are set as 1. In these real experiments, as we cannot determine which pixels are

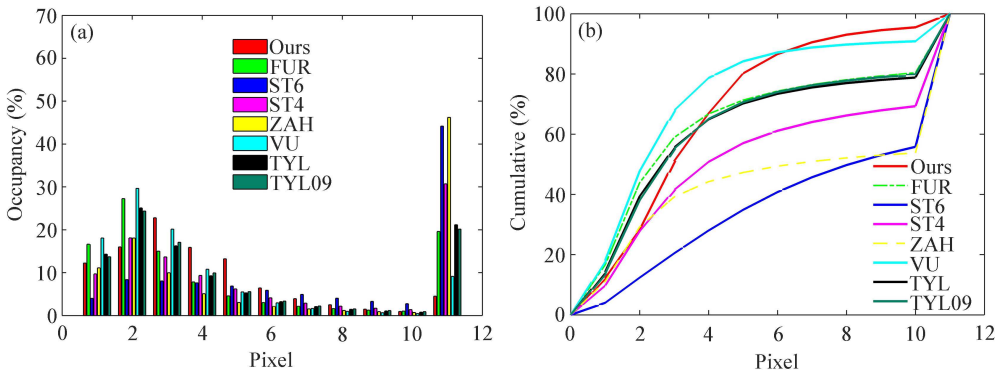


Figure 10 (Color online) Reprojection errors of Hers-Jesu-P8 3D reconstruction results. (a) Histograms of the reprojection errors occurrences: percentage of data with an error of n pixels; (b) cumulative reprojection errors distribution percentage of pixels with an error smaller than n pixels.

outliers, we set all the confidences as 1.

Figure 9 shows the stratified 3D reconstruction results for our trained model. As can be observed, our projective reconstruction preserves the collinearity, intersection, and tangency. The affine reconstruction preserves the parallelism, and the metric reconstruction preserves the relative lengths and orthogonality.

In order to quantitatively evaluate our real scenes' reconstruction results against the ground truth, we compute the RMSE of our learnt true Euclidean reconstruction with respect to the ground truth. The RMSEs of Herz-Jesu-P8, Fountain-P11, and Castle-P30 are 0.027, 0.029, and 0.165 m respectively. The RMSE of Castle-P30 is greater, which is expected because the scene to be reconstructed is relatively large (buildings) and the amount of the training data is relatively small. In order to evaluate the true Euclidean reconstruction more objectively, we use the relative reconstruction error ratio (RRER) of each of the scenes. RRER is defined as the ratio of the RMSE to the depth range of the ground truth. The RRERs of Herz-Jesu-P8, Fountain-P11, and Castle-P30 are 0.18%, 0.32%, and 0.53% respectively.

We compare our model with several traditional geometry-based methods, for which the reconstruction results using these benchmark datasets are available online. In the case of Herz-Jesu-P8, FUR, ST6, ST4, ZAH, VU, TYL, SAL, and TYL09 are used for comparison. In the case of Fountain-P11, FUR, ST6, ST4, ZAH, VU, TYL, SAL, TYL09, JAN, and JAN09 are used for comparison. In the case of Castle-P30, VU and JAN09 are used for comparison, as only these two methods provide results for this dataset. The reprojection error is used as the evaluation criterion for comparison.

First, we present the reprojection errors of our model and FUR, ST6, ZAH, ST4, VU, TYL, SAL, and TYL09 for Herz-Jesu-P8, as shown in Figure 10. Figure 10(a) shows the histograms of the reprojection error occurrences. As can be observed, the reprojection errors obtained with our method that are greater than 10 pixels are evidently fewer than those of the traditional geometry-based methods, which indicates that our model is more accurate in this case. Figure 10(b) shows that 95.5% of the reprojection errors of our method are smaller than 10 pixels while the best performance among those of the traditional methods (VU) is 90.8%, and the worst (ST6) is 53.7%.

Next, we present a comparison of the performance of our model for Fountain-P11 with FUR, ST6, ST4, ZAH, VU, TYL, SAL, TYL09, JAN, and JAN09. Figure 11(a) and (b) shows that the reprojection errors of our model are greater than those of most of the traditional methods, but smaller than those of JAN09. Figure 11(c) and (d) shows that 76.4% of the reprojection errors of our method are smaller than 10 pixels while the best performance of the traditional methods (VU) is 90.8%, and the worst (JAN09) is 67.1%.

Finally, we present a comparison of the performance of our model for Castle-P30 with VU and JAN09. It should be noted that, in this experiment, no ground truth is available and our image points are projected from the reconstructed 3D points obtained from OpenMVG, which contain a sizeable number of outliers. Figure 12(a) shows that the reprojection error of our model is greater than those of both VU and JAN09. Figure 12(b) shows that 65.4% of the reprojection errors of our method are smaller than 10

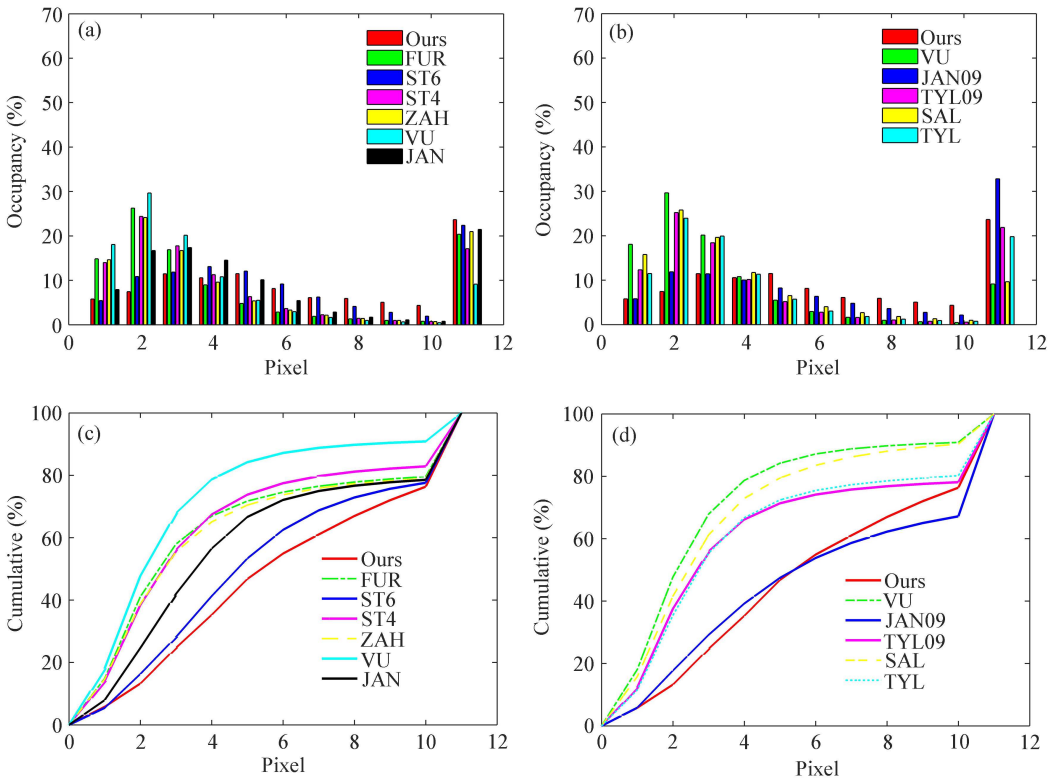


Figure 11 (Color online) Reprojection errors of Fountain-P11 3D reconstruction results. (a) and (b) Histograms of the reprojection errors occurrences: percentage of data with an error of n pixels; (c) and (d) cumulative reprojection errors distribution percentage of pixels with an error smaller than n pixels.

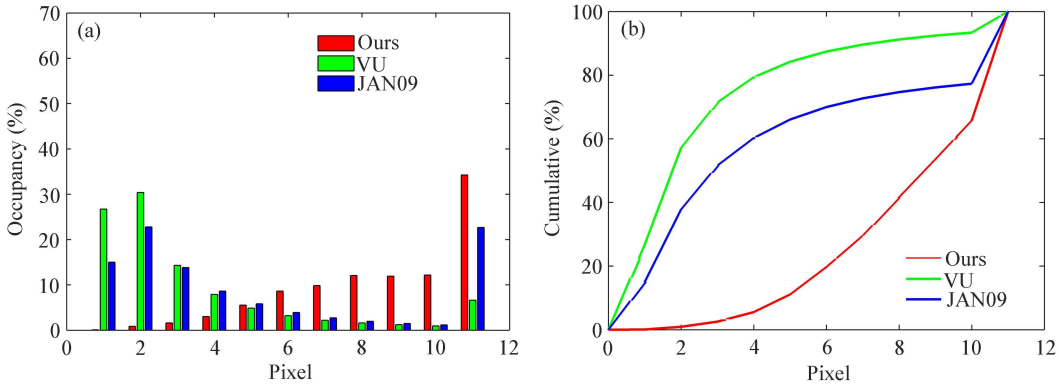


Figure 12 (Color online) Reprojection errors of Castle-P30 3D reconstruction results. (a) Histograms of the reprojection errors occurrences: percentage of data with an error of n pixels; (b) cumulative reprojection errors distribution percentage of pixels with an error smaller than n pixels.

pixels while 93.3% and 77.3% of those of VU and JAN09 respectively are smaller than 10 pixels.

In summation, our model exhibits good performances for the benchmark datasets, and it exhibits a certain robustness to noises and outliers. However, the performance of our method is not as good as that of traditional geometry-based methods, which was largely anticipated because our method does not possess a robust outlier-removal model.

5 Conclusion and future work

In this paper, we investigate the feasibility of learning the stratified 3D scene reconstruction from putative image point correspondences and determine whether the learnt reconstruction could be as robust to outliers and random noises as the state-of-the-art SfM methods. A specially designed parsimonious neural network is proposed for the learning. Both the simulation and real experiments are conducted for evaluating the true Euclidean reconstruction. The results show that learning stratified 3D reconstruction is feasible, and it also exhibits some outlier and noise resistance. However, its robustness and accuracy are not as good as those of the state-or-art SfM counterparts owing largely to its lack of an explicit and integrated outlier-detector, which is invariantly embedded in SfM methods. To the best of our knowledge, our study is the first attempt in the extant literature to bridge the geometry with learning.

In this study, we used a parsimonious network rather than a general CNN; this is because, by enforcing inherent intermediate features as demonstrated in several literatures, the performance can generally be improved in addition to the demand for fewer resources in its implementation in terms of network complexity, training time, and the size of the training datasets. In our future study, we intend to use a general CNN in order to learn our stratified 3D reconstruction to determine whether it could outperform our parsimonious network.

Finally, we do not claim that our learning-based stratified 3D reconstruction could provide a better performance than the state-of-the-art SfM methods, i.e., at least, not at the current stage. Our main objective is to explore the feasibility of learning the stratified 3D scene reconstruction and to assess its robustness to noise and outliers. We found that making learning-based 3D reconstruction robust to outliers is a key issue that must be addressed in the future in order for leaning-based approaches to compete with SfM methods. In SfM methods, outlier removal relies on an epipolar constraint and repeated optimization. In learning-based methods, epipolar constraints cannot be explicitly enforced, and once learnt, the networks become fixed; it is then impossible to detect outliers on the fly. In addition, it is difficult to embed an outlier-removal model in DNNs. This is possibly why space vision is still dominated by geometry-based approaches nowadays although object vision has been revolutionized by DNNs.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. 61333015, 61375042, 61421004, 61573359, 61772444).

References

1 Roberts R, Sinha S N, Szeliski R, et al. Structure from motion for scenes with large duplicate structures. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Colorado, 2011. 3137–3144
2 Kerl C, Sturm J, Cremers D. Dense visual slam for rgb-d cameras. In: Proceedings of 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, 2013. 2100–2106
3 Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks. Adv Neural Inf Process Syst, 2012, 25: 1097–1105
4 Hartley R, Zisserman A. Multiple View Geometry in Computer Vision. New York: Cambridge University Press, 2003
5 He K, Zhang X, Ren S, et al. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, 2016
6 Eigen D, Fergus R. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In: Proceedings of the IEEE International Conference on Computer Vision, Santiago, 2015. 2650–2658
7 Kendall A, Grimes M, Cipolla R. Posenet: a convolutional network for real-time 6-dof camera relocalization. In: Proceedings of the IEEE International Conference on Computer Vision, Santiago, 2015. 2938–2946
8 Kulkarni T D, Whitney W F, Kohli P, et al. Deep convolutional inverse graphics network. In: Proceedings of International Conference on Neural Information Processing Systems. Cambridge: MIT Press, 2015. 2539–2547