Generation of Upright Panoramic Image from Non-upright Panoramic Image

Jingguo Liu[1], Heyu Chen[1], Shigang Li[2]*, Jianfeng Li[1]*
[1]College of Electronic and Information Engineering, Southwest University, China
[2]Graduate School of Information sciences, Hiroshima City University, Japan
{liujingguo,cy10123}@email.swu.edu.cn shigangli@hiroshima-cu.ac.jp popqlee@swu.edu.cn

Abstract

The inclination of a spherical camera results in nonupright panoramic images. To carry out upright adjustment, traditional methods estimate camera inclination angles firstly, and then resample the image in terms of the estimated rotation to generate upright image. Since sampling an image is a time-consuming processing, a lookup table is usually used to achieve a high processing speed; however, the content of a lookup table depends on the rotational angles and needs extra memory to store also. In this paper we propose a new approach for panorama upright adjustment, which directly generates an upright panoramic image from an input nonupright one without rotation estimation and lookup tables as an intermediate processing. The proposed approach formulates panorama upright adjustment as a pixelwise image-to-image mapping problem, and the mapping is directly generated from an input nonupright panoramic image via an end-to-end neural network. As shown in the experiment of this paper, the proposed method results in a lightweight network, as less as 163MB, with high processing speed, as great as 9ms, for a 256×512 pixel panoramic image.

1. Introduction

Nonupright image is caused by the inclination of a camera. To carry out image upright adjustment, a natural and straight approach is to estimate the inclination firstly, and then compensate the estimated inclination and generate upright image by resampling the nonupright image, as shown in Figure 1(a). Until now, all the methods of image upright adjustment are based upon this approach, called traditional approach in this paper. That is, in the traditional approach, the task of image upright adjustment is divided into two subtasks: inclination estimation and image resampling. Since sampling an image is a time-consuming processing, a Look-Up Table(LUT) is usually used to achieve a high processing speed; however, the content of a LUT depends on the rotational angles and needs extra memory to store also.

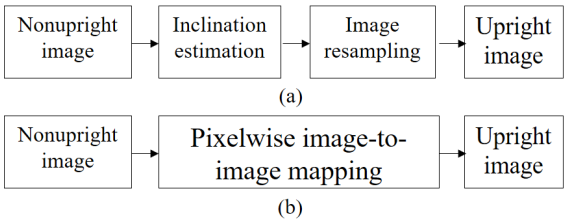

Figure 1. (a) Traditional approach for image upright adjustment, which consists of two subtasks: inclination estimation and image resampling. (b) Our proposed method which formulates image upright adjustment as a pixelwise image-to-image mapping problem.

If we focus on the input nonupright image and output upright images, this problem can be formulated as a pixelwise image-to-image mapping problem, as shown in Figure 1(b). That is, what we want to do is to generate an upright image for a given nonupright image. According to this point of view, we can attack this problem by a completely different approach, called direct method in this paper. As an example, we focus on panoramic image upright adjustment in this paper. If we treat the intermediate process as one block, the objective of panorama upright adjustment can be formulated as a generative task, that is, generating an upright panoramic image from an inclined one. More concretely, since rotating a spherical camera does not give rise of any change of image content, what we want to do is a pixelwise image-to-image mapping, that is, mapping a pixel of input image to a new position of output image. Therefore, generating an upright panoramic image can be seen as a problem of estimating pixel displacement between two images.

Estimating pixel displacement between the two images is a basic problem in computer vision, such as optical flow estimation for a motion camera [10,23,34] and disparity estimation for a stereo images [28]. Traditionally, image upright adjustment is attacked as a different problem because of different prerequisites; optical flow or stereo disparity is estimated from a pair of images while image upright adjust-

*Corresponding authors

ment is carried out given a single nonupright image under the condition that the output image must be upright. How to measure the inclination of a nonupright image accurately is not easy, especially for the image of cluttered indoor scenes or outdoor natural scenes. However, if we attack image upright adjustment problem using machine learning methods, the prerequisite change. In machine learning methods, a collected dataset also includes targets (ground truth) besides input. As for our image upright adjustment, every nonupright image has its corresponding upright image in the dataset. We train a model using nonupright input, and using upright images for the output loss. More concretely, in the training phase we can train a neural network to estimate the pixel displacement of a nonupright panoramic image from an upright one, while in the test phase an upright panoramic image is generated directly from a nonupright panoramic image using the trained model, which can extract inclination features from the nonupright panoramic image for estimating pixel displacement. The main contribution of this paper is as follows.

1. While the existing methods, either geometrical computation methods or neural network method of image upright adjustment are based on a two-subtask approach, we propose a new approach of carrying out panorama upright adjustment as a simple pixelwise image-to-image mapping processing. To the best of our knowledge, it is different from all the existing method.

2. We developed an end-to-end neural network to directly generate an upright panoramic image from a nonupright one. In contrast with the related research, which trains a neural work to estimate inclination angles of nonupright images, we argue that large dimension inclination features contain more information for the estimation of pixel displacement considering the difficulty of inclination angle estimation of clustered or natural outdoor scenes.

3. Since the trained model of our method generates an upright panoramic image directly from an inclined one, it results in a lightweight network with high processing speed. As shown in the experimental result, our trained model achieves as high as 9ms per frame processing speed for the generation of a 256×512 pixel upright panoramic image. It is suitable for real-time interactive tasks and the embedded employment of mobile devices.

2. Related Work

Here, we introduce the related research of this research. The related research is divided into two categories: deep-learning-based image upright adjustment and pixel displacement estimation

2.1. Deep-learning-based image upright adjustment

An upright image can be computed easily if the inclination angles of its corresponding nonupright image are known. The existing methods have focused on the estimation of inclination angles until now.

As for the upright adjustment of perspective images, Fischer *et al.* [9] proposed a method in which a convolutional network can learn subtle features to predict the canonical orientation of images. Olm-schenk *et al.* [22] proposed using convolutional neural networks (CNNs) to automatically determine the pitch and roll of a camera using a single, scene-agnostic, 2D image. Guerzhoy *et al.* [14] applied CNNs to the problem of image orientation detection in the context of determining the correct orientation (from 0, 90, 180, and 270 degrees). Shima *et al.* [26] proposed a novel orientation detection method for face images that relies on image category classification by deep learning.

As for the upright adjustment of panoramic images, Jeon *et al.* [12] proposed a novel upright adjustment framework based on a CNN. Jung *et al.* [16] proposed a deep learning-based approach that can automatically estimate the orientation of a VR image and return its upright version. Shan *et al.* [25] investigated the representation of spherical images by focusing on the inclination estimation of a spherical camera. Davidson *et al.* [6] investigated how to solve this problem by fusing purely geometric cues, such as apparent vanishing points, with learned semantic cues, such as the expectation that some visual elements have a natural upright position. Jung *et al.* [15] proposed a novel method for the upright adjustment of 360° images that consists of two modules: a CNN and a graph convolutional network (GCN).

The common point of the above research is that upright image adjustment is solved as a two-subtask problem, and different from our method of generating upright image directly by computing pixelwise mapping.

2.2. Deep-learning-based pixel displacement estimation

Object or camera motion results in pixel displacement of scene points in images. Estimating pixel displacement between two images is a basic problem in computer vision, such as optical flow estimation for a motion camera [10, 23, 34] and disparity estimation for stereo images [28].

As for the estimation of optical flow, Liu *et al.* [21] used optical flow to reduce the influence of noise induced by head movements. Zhi *et al.* [32] used optical flow to describe the change in pixels between two images. In face tracking, Decarlo *et al.* [7] used optical flow to greatly improve the estimation of qm, the motion parameters of the deformable model. In mobile vehicle detection, Aslani *et al.* [2] used optical flow to replace the combination of image segmentation regions for object detection. In video estimates, Wang *et al.* [29] used optical flow to perform motion compensation to encode the time correlation, which greatly improved the SR performance. In object detection, Al-Battal *et al.* [1]

proposed a framework that uses a segmentation-based CNN to detect and localize the target anatomical structure within a scan. Concurrently, it uses an optical flow CNN to track the movement of this structure across frames to accurately guide therapeutic procedures.

As for the estimation of disparity of stereo images, Kordelas *et al.* [17] presents a novel stereo disparity estimation method, which combines three different cost metrics, defined using RGB information, the CENSUS transform, as well as Scale-Invariant feature transform coefficients. Zhou *et al.* [33] proposed a new method that the disparity estimation tasks can be accomplished using a single input image. Zhang *et al.* [31] proposed to initial disparity estimates are refined with an embedding learned from the semantic segmentation branch of the network. Du *et al.* [8] proposed a new deep learning architecture for stereo disparity estimation: atrous multiscale network (AMNet), AMNet adopts an efficient feature extractor with depthwise-separable convolutions and an extended cost volume that deploys novel stereo matching costs on the deep features. While two images are used as input to estimate output (target) in the methods of optical flow or disparity estimation for both training and testing phases, in our method a nonupright image (input) and an upright image (target) is used to extract inclination features to estimate pixel displacement in the training phase, and an upright image is generated directly from a nonupright image in the testing phase. Therefore, the neural network of our method has a different architecture in comparison with optical flow or disparity estimation methods, as shown in Figure 2.

3. Proposed Methods

3.1. Mathematical computation of panorama upright adjustment

Here, we give the detailed mathematical computation of panorama upright adjustment in terms of the traditional approach (see Figure 3). A panoramic image can be captured by a spherical camera with full field of view, and usually is represented as an equirectangular image. Assume that we have a pair of nonupright equirectangular image, $I_i(u, v)$, and its upright one, $I_i'(u', v')$. The coordinates of a scene point at $I_i(u, v)$ and $I_i'(u', v')$ are $p(u_p, v_p)$ and $p'(u_p', v_p')$. The azimuth angle φ_i and polar angle θ_i of a pixel pin the nonupright equirectangular image can be computed as:

$$\varphi_i = 2\pi \frac{u_p}{W}, \theta_i = \pi \frac{v_p}{H} \tag{1}$$

where W and H are the image width and height, respectively. Then, we have the orthogonal coordinate (x_p, y_p, z_p) of $p(u_p, v_p)$ on a unit sphere as:

$$x_p = \cos\varphi_i \sin\theta_i,\ y_p = \sin\varphi_i \sin\theta_i,\ z_p = \cos\theta_i \tag{2}$$

Suppose the camera inclination is represented by a roll angle, α, and pitch angle, β, which are estimated from the nonupright panoramic image. Then, the image upright adjustment can be carried out by rotating the nonupright image with a rotation matrix $R(\alpha, \beta)$.

$$\begin{bmatrix} x_p' \\ y_p' \\ z_p' \end{bmatrix} = R(\alpha, \beta) \begin{bmatrix} x_p \\ y_p \\ z_p \end{bmatrix} \tag{3}$$

where (x_p', y_p', z_p') is the orthogonal coordinates on a unit sphere. Finally, using the inverse functions of Equation 1 and Equation 2 we can obtain the upright equirectangular image as follows.

$$\varphi_i' = arctan2\left(y_p', x_p'\right), \theta_i' = arccos(z_p') \tag{4}$$

$$u_p' = \frac{W}{2\pi}\varphi_i', v_p' = \frac{H}{\pi}\theta_i' \tag{5}$$

where φ_i' and θ_i' are the azimuth angle and polar angle at the unit sphere rectified upright, respectively; and u_p' and v_p' are the coordinates of point $p\prime$ at the upright rectangular image.

Summarizing the above mathematical computation, the problem of panorama upright adjustment is to compute point $p\prime(u_p', v_p')$ of an upright panoramic image from point $p(u_p, v_p)$ of a given nonupright panoramic image by using estimated inclination angles (α, β).

Since Equation 1, Equation 2, Equation 3, Equation 4 and Equation 5 are deterministic, the traditional methods of panorama upright adjustment focus on the estimation of camera inclination angles (α, β) from a nonupright image [6, 12, 15, 16, 25]. In addition, since mapping a point $p(u_p, v_p)$ of nonupright image to the corresponding one $p'(u_p', v_p')$ of upright image involves nonlinear computation, as shown in Equation 2, Equation 3 and Equation 4, LUTs are usually used to speed up the processing for real-time interactive tasks. However, since a LUT depends on the estimated inclination angles, theoretically, it means that infinitive LUTs are needed to cope with arbitrary inclination angles. That is, a large amount of memory storage is needed to achieve high speed processing performance.

We formulate upright adjustment task as a pixelwise displacement estimation task. We represent the nonupright image pixel as $p(u_p, v_p)$, and upright image pixel as $p'(u_p', v_p')$, where u_p and v_p are the coordinate of the pixel in the image. Our purpose is to estimate a pixel displacement model to map the pixel from $p(u_p, v_p)$ to $p'(u_p', v_p')$ directly.Let the displacement of $p(u_p, v_p)$ to $p\prime(u_p', v_p')$ be $(\Delta u_p, \Delta v_p)$. Then, we have:

$$u_p' = u_p + \Delta u_p, v_p' = v_p + \Delta v_p \tag{6}$$

In this paper, we do not predict the tilted angle of the camera as traditional methods. In our network, we extract the inclination features through the encoder, and then the decoder

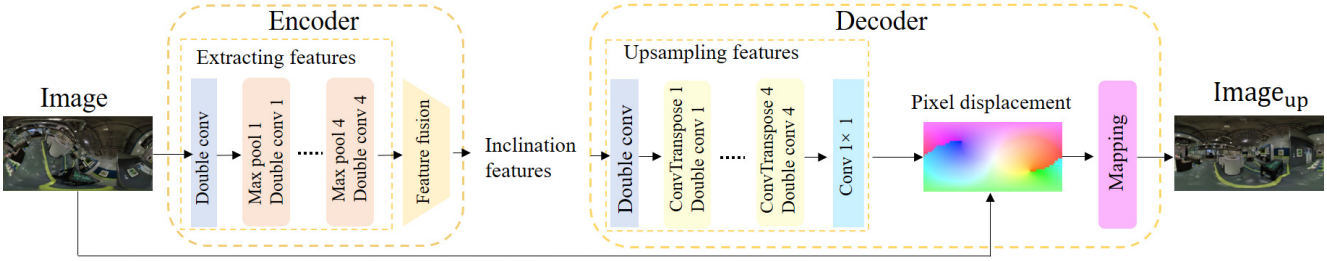

Figure 2. Network architecture: Extract inclination features using the encoder, estimate pixel displacement and generate an upright image using the decoder

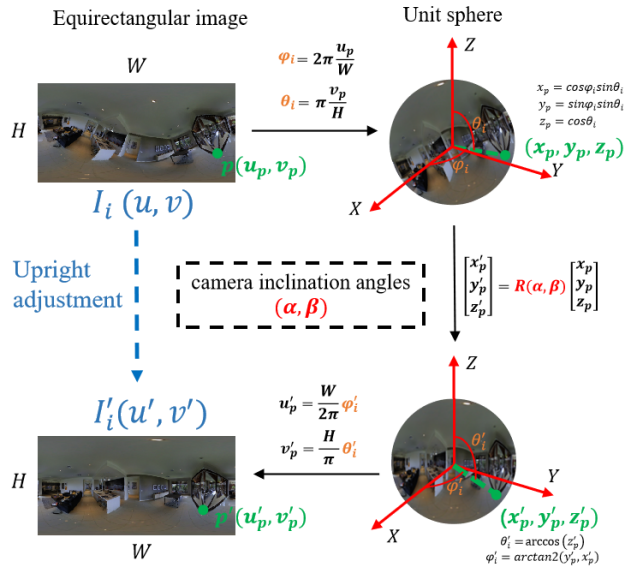

Figure 3. Mathematical computation of panorama upright adjustment.

estimate the pixel displacement. Therefore, the upright adjustment in our view can be represented as follows:

$$p'(u'_p, v'_p) = p(u_p, v_p) \oplus p(\Delta u_p, \Delta v_p) \tag{7}$$

Where $p(\Delta u_p, \Delta v_p)$ represents the pixel displacement. $\oplus$ represents the mapping operation which will fill the values of the input to a specified position according to the coordinate information provided by the pixel displacement, our method use grid_sample functions to realize it. We will introduce our network in later sections.

3.2. Network Architecture

As mentioned in subsection 2.2, estimating pixel displacement is a basic problem in computer vision, such as optical flow estimation and disparity estimation. Until now, estimate pixel displacement by neural networks has been a common method in many computer vision tasks. For example, Sun *et al.* [27] proposed a method to extract optical flow using a deep learning network. Yan *et al.* [30] proposed a deep learning network to learn the inherent pixel correspondence between stereo views and restores stereo image with the cross-view information at image and feature level.

These methods prove that compared with traditional methods, using a deep learning network to extract pixel displacement can have good performance. Thus, our method designs a deep learning network for estimating the pixel displacement from the nonupright image. Concretely, we utilize a downsampling encoder network to obtain the inclination features and a decoder to estimate the pixel displacement. As Figure 2 shows, our network can adjust the image in one step, while current upright adjustment methods are only focusing on the rotation estimation, and generate upright image by resampling the nonupright image (mostly by LUT). Moreover, compared with traditional LUT-based methods, our method cost smaller space and have a better performance. Until now, obtaining optical flow by deep learning networks has been a common method in many computer vision tasks. For example, Sun *et al.* [27] proposed a method to extract optical flow using a deep learning network. Chang *et al.* [4] used an optical flow generation method to conduct unsupervised periodic consistent learning for facial representation. They used convolution to extract the features in the downsampling network and estimate the optical flow in the upsampling network. Through the optical flow generated by their network, they modeled facial expressions and proposed a new framework for unsupervised learning facial representations from a single facial image. These methods prove that compared with traditional methods, using a deep learning network to extract optical flow can have good performance. Different with previous work, our method is based on the motion of a 360 panoramic scene. For motion, we extract a motion feature from the image to express the motion. We design a downsampling encoder network to obtain the motion and a decoder to estimate optical flow. Our method is the first to adjust the panoranic without angle prediction.

As Fig. 2 shows, our network is divided into an encoder and a decoder. The encoder extracts the high-level motion features from the nonupright image. The decoder estimates

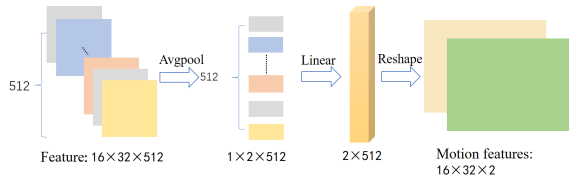

Figure 4. Feature fusion process

the dense optical flow according to the high-level motion features. The decoder also needs to map the nonupright image with the optical flow to generate an upright image. Our network can adjust the image in one step, while traditional methods are divided into two steps. Moreover, our method cost more smaller space and have a better performance than traditional LUT-based methods.

3.2.1 Encoder

In this section, we will introduce the downsampling encoder network. To fully extract the inclination features, we use a double convolution first to fully extract the inclination features on the input size and then downsampling with a double convolution at every size. We attempted to use a single convolution for each downsampling layer, but the effect was poor, which proves that the shallow network cannot extract features well. Ablation analysis can be seen in the experiments. The size of the convolution kernel for all layers is 3, and the stride is 1 for convolution and 2 for max-pooling. Usually, the pixel displacement network does not downsample the feature size to very small [20]; therefore, only five layers are included in our design, and the final feature size of the convolutional layers is 16×32×512. Similar to some common networks, such as ResNet, average pooling is followed by convolutional layers. Different with traditional methods, we set the kernel of average pooling to (1, 2) since the height-width ratio of 360 panoramic images is 1:2. Next, essentially, the convolutional layers provide a meaningful, low-dimensional, and somewhat invariant feature space, and the fully connected layer learns a (possibly nonlinear) function in that space. A fully connected layer is added to integrate the features. Considering that the following decoder is designed for generating a 2D dense pixel displacement by upsampling, 2D feature maps are needed as the input. Additionally, the displacement of pixels on images is separated into horizontal and vertical directions, and we reshape the features to 2-channel 2D feature maps. The whole process after convolutional layers is shown in Figure 4.

3.2.2 Decoder

In this section, we introduce the key to our task: obtain the pixel displacement.

We design an upsampling network to estimate the pixel displacement from inclination features. Since the size of inclination features is 16×32, while the pixel displacement should be the same size as the 256×512 input size. Thus, the structure of the decoder is consistent with that of the encoder, with 5 layers included, a 2D transposed convolution operator for upsampling and double convolution on each size for feature integration. The size of the convolution kernel for all layers is 3, and the size of the transposed convolution kernel is 2. Finally, a pointwise convolution with Tanh activation is applied to obtain a 2-channel 256×512 pixel displacement. Compared with other activation functions, Tanh is quite common in current generation networks [19] [24]. Tanh has the merit that the gradient is relatively large during backpropagation, reducing the possibility of gradient dispersion. After the pixel displacement is generated through upsampling, a simple mapping function is conducted to generate the upright image: grid_sample function, which will fill the values of the input to a specified position according to the coordinate information provided by the pixel displacement.

3.3. Loss Function

In image generation networks, most of the generate networks are regularized by $L1$ loss. regularizing the networks by $L1$ loss can greatly regularize the layout of the generated images, making the generated images generally similar to the ground truth and reducing blurring. For example, the well-known networks, PasteGAN [18] and pix2pix [11] both use $L1$ loss to achieve good performance. Therefore, in this paper, to make the images generated by our network achieve a good effect, we use $L1$ loss to regularize the network. In this paper, L1 loss is denoted as:

$$L1 = \frac{\sum_i^n \left| X_i - \hat{X}_i \right|}{n} \tag{8}$$

where X_i denotes the pixel truth value from the upright dataset, and $\hat{X}_i$ denotes the pixel value from the generated image. However, the image generated using only $L1$ loss has flaws in high-frequency details, which leads to the generated image not being perfect in high-frequency details.

In this paper, to improve our image details and quality and reduce the pixel loss between images, shorten the Euclidean distance between the generated image and the target image, we use perceptual loss [13] to regularize our network. We use VGG19 to compute the features of the image, in which image perceptual loss is denoted as:

$$L_{Perceptual} = \text{MSE}(\varphi(\hat{X}), \varphi(X)) \tag{9}$$

where X denotes the upright panorama, $\hat{X}$ denotes the generated image from our network, and φ denotes the VGG19 feature extraction module, MSE denotes the mean squared

Table 1. Ablation analysis with different options

single convolution	double convolution	perceptual loss	λ=1	λ=0.1	λ=0.01	FID↓
√		√		√		50.25
	√					35.30
	√	√	√			35.99
	√	√			√	32.58
	√	√		√		**31.21**

all trained with the same hyper-parameters

error. In this paper, we introduce these two loss functions to enrich our generated image quality. The total loss is denoted as:

$$L_{total} = L1 + \lambda * L_{Perceptual} \tag{10}$$

where $L1$ is the $L1$ loss, $L_{Perceptual}$ is the perceptual loss, and λ = 0.1. By regularize the network, the quality of the image generated by our method greatly improved. Compared with the traditional methods, our performance is better.

4. Experiments

4.1. Dataset and Training Details

Dataset: In this paper, we use Matterport3D [3] to correct panoramic images and to evaluate our method. Due to hardware limitations, we used low solutions panoramic images with 256*512 solutions and we only trained and designed a network apply to 256*512 panoramic images (the same size as current tile estimation methods in the field), but the work can be extended to higher resolutions in theory. We believe our novel idea could enlighten the future works.At present, researchers use this dataset for various tasks, such as depth estimation. However, these tasks are all conducted on upright panoramic images. Therefore, it is appropriate to conduct our upright adjustment on this dataset. Since Matterport3D only provides upright images, we rotate the source images with a random angle in the range [-90°, 90°] for both pitch and roll to achieve a nonupright panorama dataset. We take 70% for training, 15% for evaluation, and the final 15% for testing in original dataset. To cover all the angles in the range of [-90°, 90°], five different random rotations of each image were used to expand the datasets after the dataset is divided.

Training details: Our entire network was trained using the Adam optimizer, a batch size of 8, and a learning rate of 2×10-4 on a TITAN RTX 24G.

4.2. Ablation Analysis

In 3.2.1, we mentioned that our network needs to obtain the inclination features through the encoder, and we propose to use double-convolution layers instead of single-convolution layers to extract inclination features in the encoder. To illustrate the merits of using double-convolution layers, a comparison is conducted. In addition, ablation analyses using perceptual loss and the value of λ are also conducted. We introduce FID (Fréchet Inception Distance) to evaluate the whole test set under different situations since it is difficult to distinguish the difference in view. As Table 1 shows, the single convolution performs worst. The perceptual loss did help to upgrade the image quality. Finally, we take the best options when doing the following experiments.

Figure 5. The left is the input, the second column is the result of our method, the third column is the result of the LUT, and the 4th column is the result of Chen *et al.* [5]. *: using LUT to adjust the image with a known tilt angle. This method is used by Shan *et al.* [25], Jung *et al.* [16]

4.3. Upright Evaluation with state of the art

Since our method focuses on pixel displacement, it cannot have an intuitive angle output as previous works have done. To make a reference standard, we rotate the original upright images in the test set by a unified angle error and then calculate the Normalized Root Mean Square Error (NRMSE) and Normalized Mean Absolute Error (NMAE) between the ground truth and the rotated ground truth. As Table 2 shows, we achieve the references with 1° and 2° rotation in pitch and roll, respectively. Next, we calculate the NRMSE and NMAE between our upright generations and the ground truth with the whole test set. We can see that the values of our method are between 1° and 2°, which means our method almost make all the nonupright images horizontal. We also list the result of Chen *et al.* [5] and Shan

Table 2. Upright evaluation

Evaluation	Ours	Chen *et al.* [5]	Shan *et al.* [25]	Pitch 1°	Pitch 2°	Roll 1°	Roll 2°
NRMSE↓	0.2268	0.3225	0.5452	0.1935	0.2520	0.1920	0.2497
NMAE↓	0.1363	0.2180	0.4843	0.1039	0.1447	0.1034	0.1433

Table 3. Image quality, time, space comparison

Evaluation	Ours	LUT*	Chen *et al.* [5]
FID↓	**31.21**	41.7467	59.97
Time↓	**0.009s**	Could be real time on GPUs (on-site LUT calculation)	0.012s
space↓	**163MB**	4.5GB (LUTs offline)	429MB

*: using LUT to adjust the image with a known tilt angle. This method is used by Shan *et al.* [25], Jung *et al.* [16]

et al. [25], and our performance is better than them. There are also two related papers in the field, Jung *et al.* [16] and Davidson *et al.* [6]. Since they did not generate the upright images and are not making their code public available, we compute their average predict angle's deviation in 5° from their paper. Jung *et al.* [16] is 1.977° and Davidson *et al.* [6] is 1.807°. Because the values of the NRMSE and NMAE of our method are both between 1° and 2°, we estimate our average angle roughly from the range of NRMSE and NMAE, our method is 1.825°. The experiment proves that our method is competitive with existing methods in the accuracy of upright adjustment. And it shows that our method is acceptable for usage.

4.4. Image quality, time, space comparison

There are many researchers performing upright adjustment at present. However, all of them focus on angle estimation [6, 12, 15, 16, 25], then adjust the nonupright images with tilt angles referring to the spherical model-based projection. Notably, their methods just predict the tilt angle, the output of their method is angle. Moreover, to trade space for time, they usually build LUTs offline to save the mapping relation in place of on-site calculation. In theory, on-site calculation and LUTs offline have the same effect. Chen *et al.* [5] is the first to propose to adjust the image by an end-to-end network without LUT. Thus, we have a comparison among our method, the LUT-based method, and Chen *et al.* [5] with official settings.

Image quality: In Figure 5, the first column shows the nonupright images, the second column is directly output by our end-to-end network, the third column is rotated by LUT, and the last column is Chen *et al.* [5] method. Visually, our method is nearly the same as the LUT-based method. However, the FID on the whole testset of our generations is 31.21, while the LUT generation is 41.75, Chen *et al.* [5] is 59.97. This proves our method would have a better effect. We also test our method on Stanford2D3D [35] while trained on Matterport3D [3]. The quality is quite good, especially perform better on edges than LUT-based method.

Space and time cost: We test our network running on PyTorch with GPUs, and the average time for a sample is 0.009s, which we think is faster than traditional pipeline. Here are the reasons: 1. the traditional pipeline needs to estimate the tile angle first, the recent accurate methods are all deep learning based methods with 256*512 size input, and the time cost of their network may not be faster than ours, since our five basic CNN layers are very simple. Let alone they need extra time on upright adjustment based on the spherical model. 2. As the resolution be higher, the traditional pipeline would grow exponentially, Take 1024*2048 as an example, the rotation computation based on the spherical model would be 16 times than 256*512, since the computation is pixel wise. While our method does not need the spherical based rotation. Certainly, our model on higher resolution would cost more time than current, since the parameters in pretrained model increasing, but their tile estimation methods would also cost more time. They have the same problems as we have, because we are all deep learning based methods. A pure deep learning based method could be real-time always. Meanwhile, the running time of the traditional remapping process by the spherical model is truly different under different platforms and optimization levels. For example, we have used Python and MATLAB to realize the traditional process, and we found that the running time is quite slow (e.g., 0.5s for a sample in MATLAB on GPUS, much slower using Python), which is not fair to list as the reference. We think the spherical-based projection can reach real time on GPUs with a very high-level optimization strategy (Table 3). However, considering the additional time cost of preposition angle estimation methods, we believe our 0.009s is quite competitive. The main space cost of our network is the pretrained model when using it. Since we only use a lightweight encoder-decoder network, it only takes 163 MB of storage, which is quite small in deep learning fields. It can be shown that our network is an efficient way to balance space and time.

4.5. Application Test

4.5.1 Pedestrian Detection on Outdoor Images

To show the viability of our approach, we use the YOLOv5 pretrained model, which trains on perspective images, to detect pedestrians in panoramic images. We collect some out-

Table 4. Depth estimation results

	Abs Rel↓	Sq Rel↓	RMS↓	RMSlog↓	$\delta < 1.25 \uparrow$	$\delta < 1.2^2 \uparrow$	$\delta < 1.25^3 \uparrow$
Non-upright images in range [-90°,90°]	0.4492	0.7937	1.2815	0.5467	0.4042	0.6461	0.7982
Chen *et al.* [5]	0.1408	0.1175	0.5506	0.2253	0.8247	0.9436	0.9771
Upright adjustment(our method)	0.0905	0.0488	0.3580	0.1433	0.9189	0.9850	0.9946
Ground truth	0.0548	0.0165	0.2241	0.0865	0.9759	0.9965	0.9991

Figure 6. Pedestrian detection results

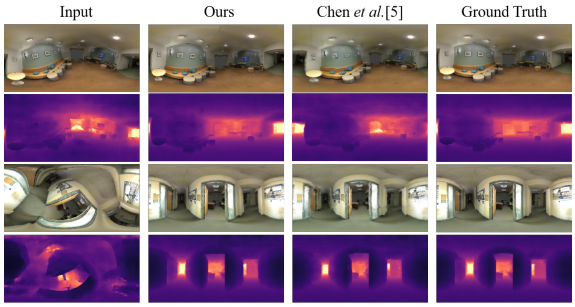

Figure 7. Depth estimation results

door images and rotate them at random angles. As Figure 6 shows, if we use the nonupright images, the model fails to detect since the pretrained model is not training for pedestrians with distortion. Only the image in the last row detects one pedestrian. Additionally, we use the model training on indoor images directly on outdoor images, which shows that our method works well even in outdoor situations. Moreover, after processing by our end-to-end network, all pedestrians are detected in the images, and the distortion is adjusted. The experimental results prove that our method does not have any influence on such detection tasks.

4.5.2 Depth Estimation

In this section, we estimate the depth by the Zioulis *et al.* [35] method and use samples from Stanford2D3D dataset to estimate the depth as in Zioulis *et al.* [35]. First, we rotate the image with random angles in the range [-90°,90°] to achieve nonupright images. Then, both the nonupright images and images adjusted by our method are input to the depth estimation network. The qualitative evaluation results are shown in Table 4. Compared with the ground truth, the nonupright images perform poorly, since the existing depth estimation models are all trained on upright images, while the generation by our method performs close to the ground truth. Moreover, our method is better than Chen *et al.* [5]. This experiment proves that our network can output qualified upright images for depth estimation. Figure 7 shows the visual depth results of our generation, which are nearly the same as the ground truth.

5. Conclusions

In this paper, we propose a new method of directly generating an upright panoramic image from an input nonupright one without rotation estimation and lookup tables as an intermediate processing. Since our method estimates a pixelwise image-to-image mapping between nonupright and upright images, it is straight forward in contrast with the existing methods. Consequently, this approach results in a lightweight end-to-end neural network. The effectiveness of this paper is shown in the experimental results in comparison with the state-of-the-art methods. The improvement of the quality of generated images is our future work.

References

[1] Abdullah F Al-Battal, Imanuel R Lerman, and Truong Q Nguyen. Object detection and tracking in ultrasound scans using an optical flow and semantic segmentation framework based on convolutional neural networks. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech*

and Signal Processing (ICASSP), pages 1096–1100. IEEE, 2022. 2

[2] Sepehr Aslani and Homayoun Mahdavi-Nasab. Optical flow based moving object detection and tracking for traffic surveillance. *International Journal of Electrical, Computer, Energetic, Electronic and Communication Engineering*, 7(9):1252–1256, 2013. 2

[3] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017. 6, 7

[4] Jia-Ren Chang, Yong-Sheng Chen, and Wei-Chen Chiu. Learning facial representations from the cycle-consistency of face. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9680–9689, 2021. 4

[5] Heyu Chen, Shigang Li, and Jianfeng Li. An end-to-end network for upright adjustment of panoramic images. *Procedia Computer Science*, 222:435–447, 2023. 6, 7, 8

[6] Benjamin Davidson, Mohsan S Alvi, and João F Henriques. 360°camera alignment via segmentation. In *European Conference on Computer Vision*, pages 579–595. Springer, 2020. 2, 3, 7

[7] Douglas Decarlo and Dimitris Metaxas. Optical flow constraints on deformable models with applications to face tracking. *International Journal of Computer Vision*, 38(2):99–127, 2000. 2

[8] Xianzhi Du, Mostafa El-Khamy, and Jungwon Lee. Amnet: Deep atrous multiscale stereo disparity estimation networks. *arXiv*, abs/1904.09099, 2019. 3

[9] Philipp Fischer, Alexey Dosovitskiy, and Thomas Brox. Image orientation estimation with convolutional networks. In *German conference on pattern recognition*, pages 368–378. Springer, 2015. 2

[10] Liwen Hu, Rui Zhao, Ziluo Ding, Lei Ma, Boxin Shi, Ruiqin Xiong, and Tiejun Huang. Optical flow estimation for spiking camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17844–17853, 2022. 1, 2

[11] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 5

[12] Junho Jeon, Jinwoong Jung, and Seungyong Lee. Deep upright adjustment of 360 panoramas using multiple roll estimations. In *Asian Conference on Computer Vision*, pages 199–214. Springer, 2018. 2, 3, 7

[13] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016. 5

[14] Ujash Joshi and Michael Guerzhoy. Automatic photo orientation detection with convolutional neural networks. In *2017 14th Conference on Computer and Robot Vision (CRV)*, pages 103–108. IEEE, 2017. 2

[15] Raehyuk Jung, Sungmin Cho, and Junseok Kwon. Upright adjustment with graph convolutional networks. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 1058–1062. IEEE, 2020. 2, 3, 7

[16] Raehyuk Jung, Aiden Seung Joon Lee, Amirsaman Ashtari, and Jean-Charles Bazin. Deep360up: A deep learning-based approach for automatic vr image upright adjustment. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 1–8. IEEE, 2019. 2, 3, 6, 7

[17] Georgios A Kordelas, Dimitrios S Alexiadis, Petros Daras, and Ebroul Izquierdo. Enhanced disparity estimation in stereo images. *Image and Vision Computing*, 35:31–49, 2015. 3

[18] Yikang Li, Tao Ma, Yeqi Bai, Nan Duan, Sining Wei, and Xiaogang Wang. Pastegan: A semi-parametric method to generate image from scene graph. *Advances in Neural Information Processing Systems*, 32, 2019. 5

[19] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. *Advances in neural information processing systems*, 30, 2017. 5

[20] Pengpeng Liu, Michael R. Lyu, Irwin King, and Jia Xu. Selflow: Self-supervised learning of optical flow. In *CVPR*, 2019. 5

[21] Yong-Jin Liu, Jin-Kai Zhang, Wen-Jing Yan, Su-Jing Wang, Guoying Zhao, and Xiaolan Fu. A main directional mean optical flow feature for spontaneous micro-expression recognition. *IEEE Transactions on Affective Computing*, 7(4):299–310, 2015. 2

[22] Greg Olmschenk, Hao Tang, and Zhigang Zhu. Pitch and roll camera orientation from a single 2d image using convolutional neural networks. In *2017 14th Conference on Computer and Robot Vision (CRV)*, pages 261–268. IEEE, 2017. 2

[23] Liyuan Pan, Miaomiao Liu, and Richard Hartley. Single image optical flow estimation with an event camera. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1669–1678. IEEE, 2020. 1, 2

[24] Tingting Qiao, Jing Zhang, Duanqing Xu, and Dacheng Tao. Mirrorgan: Learning text-to-image generation by redescription. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1505–1514, 2019. 5

[25] Yuhao Shan and Shigang Li. Discrete spherical image representation for cnn-based inclination estimation. *IEEE Access*, 8:2008–2022, 2019. 2, 3, 6, 7

[26] Yoshihiro Shima, Yumi Nakashima, and Michio Yasuda. Detecting orientation of in-plain rotated face images based on category classification by deep learning. In *TENCON 2017-2017 IEEE Region 10 Conference*, pages 127–132. IEEE, 2017. 2

[27] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943, 2018. 4

[28] Chen Wang, Xiao Bai, Xiang Wang, Xianglong Liu, Jun Zhou, Xinyu Wu, Hongdong Li, and Dacheng Tao. Self-supervised multiscale adversarial regression network for

stereo disparity estimation. *IEEE Transactions on Cybernetics*, 51(10):4770–4783, 2020. 1, 2

[29] Longguang Wang, Yulan Guo, Li Liu, Zaiping Lin, Xinpu Deng, and Wei An. Deep video super-resolution using hr optical flow estimation. *IEEE Transactions on Image Processing*, 29:4323–4336, 2020. 2

[30] Bo Yan, Chenxi Ma, Bahetiyaer Bare, Weimin Tan, and Steven C. H. Hoi. Disparity-aware domain adaptation in stereo image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 4

[31] Junming Zhang, Katherine A. Skinner, Ram Vasudevan, and Matthew Johnson-Roberson. Dispsegnet: Leveraging semantics for end-to-end learning of disparity estimation from stereo imagery. *IEEE Robotics and Automation Letters*, 4(2):1162–1169, 2019. 3

[32] Ruicong Zhi, Jing Hu, and Fei Wan. Micro-expression recognition with supervised contrastive learning. *Pattern Recognition Letters*, 163:25–31, 2022. 2

[33] Changxin Zhou, Yazhou Liu, Quansen Sun, and Pongsak Lasang. Vehicle detection and disparity estimation using blended stereo images. *IEEE Transactions on Intelligent Vehicles*, 6(4):690–698, 2021. 3

[34] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Ev-flownet: Self-supervised optical flow estimation for event-based cameras. *arXiv preprint arXiv:1802.06898*, 2018. 1, 2

[35] Nikolaos Zioulis, Antonis Karakottas, Dimitrios Zarpalas, and Petros Daras. Omnidepth: Dense depth estimation for indoors spherical panoramas. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 448–465, 2018. 7, 8