Perspective Fields for Single Image Camera Calibration

Linyi Jin\textsuperscript{1*}, Jianming Zhang\textsuperscript{2}, Yannick Hold-Geoffroy\textsuperscript{2}, Oliver Wang\textsuperscript{2}, Kevin Blackburn-Matzen\textsuperscript{2}, Matthew Sticha\textsuperscript{1}, David F. Fouhey\textsuperscript{1}

University of Michigan\textsuperscript{1}, Adobe Research\textsuperscript{2}

\textsuperscript{1}\{jinlinyi,msticha,fouhey\}@umich.edu
\textsuperscript{2}\{jianmzha,holdgeof,owang,matzen\}@adobe.com

Abstract

Geometric camera calibration is often required for applications that understand the perspective of the image. We propose Perspective Fields as a representation that models the local perspective properties of an image. Perspective Fields contain per-pixel information about the camera view, parameterized as an Up-vector and a Latitude value. This representation has a number of advantages; it makes minimal assumptions about the camera model and is invariant or equivariant to common image editing operations like cropping, warping, and rotation. It is also more interpretable and aligned with human perception. We train a neural network to predict Perspective Fields and the predicted Perspective Fields can be converted to calibration parameters easily. We demonstrate the robustness of our approach under various scenarios compared with camera calibration-based methods and show example applications in image compositing. Project page: https://jinlinyi.github.io/PerspectiveFields/

1. Introduction

Take a look at the left-most photo in the teaser (Fig. 1-A). Can you tell if the photo is captured from an everyday camera and if it has been geometrically edited? The horizon location at the bottom of the image and the parallel vertical lines of the buildings do not follow a typical camera model: The horizon at the bottom of the image indicates the camera was tilted up (pitch $\neq 0$), but this would instead produce converging vertical lines in the image due to perspective projection (Fig. 1-B). Alternatively suppose the camera has 0 pitch, preserving the vertical lines of the buildings, the horizon line would instead be in the middle of the image (Fig. 1-C). This contradiction is explained by the shift of the photo, yielding a non-center principal point, and breaking a usual assumption of many camera calibration systems.

Many single-image camera calibration works make use of a simplified pinhole camera model [22] that assumes a centered principal point [10,24,29] and is parameterized by extrinsic properties such as roll, pitch, and intrinsic properties such as field of view. However, estimating the calibration of a camera is challenging for images in the wild since they are captured by various types of cameras and lenses. Moreover, like the example in Fig. 1, the images are often...
cropped [15] or warped for aesthetic composition, which may shift the image center.

In this work, we propose Perspective Fields, an over-parameterized per-pixel image-based camera representation. Perspective Fields consist of per-pixel Up-vectors and Latitude values that are useful on their own for alignment and can be converted to calibration parameters easily by solving a simple inverse problem. The Up-vector gives the world-coordinate up direction at each pixel, which equals the inverse gravity direction of the 3D scene projected onto the image. The Latitude is the angle between the incoming light ray and the horizontal plane (see Fig. 1-D). This enables our method to be robust to cropping and we show results on multiple camera projection models.

Perspective Fields have a strong correlation with local image features. For example, the Up-vectors can be inferred by vertical edges in the image, and the Latitude is 0 at the horizon, positive above, and negative below. Since Perspective Fields have this translation-equivariance property, they are especially well suited to prediction by convolutional neural networks. We train a neural network to predict Perspective Fields from a single image by extracting crops from 360° panoramas where ground truth supervision can be easily obtained (see Fig. 2). We also use a teacher-student distillation method to transfer Perspective Fields to object-cutouts, which lets us train models to predict Perspective Fields for object-centric images.

For applications that require traditional camera parameters (e.g., roll, pitch, field of view and principal point), we propose ParamNet to efficiently derive camera parameters from Perspective Fields. Our method works on image crops and outperforms existing methods in single image camera parameter estimation. In addition, Perspective Fields can be used in image compositing to align the camera view between a foreground object and the background based on a local Perspective Field matching metric. We show with a user study that this metric for view alignment more closely matches human perspective than existing approaches. On cropped images, we reduce the pitch error by 40% over [29].

We propose a metric of Perspective Fields to estimate the low-level perspective consistency between two images. We show that this consistency measure is stronger in correlation with human perception of perspective mismatch than previous metrics such as Horizon line [24,47].

2. Related Work

Calibration for perspective images. Most calibration methods aimed at consumer cameras assume a pinhole camera model [22] to estimate both its intrinsics and extrinsics. Traditional camera calibration processes require a reference object like chessboards or planar grids [5, 13, 14, 20, 21, 23, 36, 41, 43, 52], or multiple images [18, 22, 42]. Other methods strongly rely on the Manhattan world assumption to estimate camera parameters via vanishing points [8, 9, 12, 22, 28, 37, 40]. Recently, deep learning methods directly predict camera parameters from single images, including horizon line [47] and focal length [46]. Hold-Geoffroy et al. [24] further extend a CNN to simultaneously predict camera roll, pitch, and FoV. UprightNet [48] predicts 3D surface geometry to optimize for camera rotation. A few works [29, 30, 50] combine learned features with detected vanishing points to improve performance. However, these methods are limited to perspective images with a centered principal point and often do not work on images in the wild where the centered pinhole assumption does not hold due to cropping, warping, or other similar edits.

Calibration for non-pinhole camera models. Besides the common pinhole camera model, prior works have proposed different non-linear models such as Brown-Conrady for small distortions [16], the division model [17] for fisheye cameras, and the unified spherical model [7,19,35]. Assuming certain distortion models, learning-based methods can recover focal length and distortion parameter [3,10,31,34]. With a known 3D shape and its correspondences, [11, 38] can recover lens distortions. Instead of relying on a specific lens model, we propose a generic representation that stores the up and latitude information for each pixel. This local representation encompasses multiple camera projection models. Our versatile Perspective Field can be used to recover the parameters of a specific model, if desired.

Perspective aware object placement. Many works aim to automate the image compositing process by directly learning to match lighting, scale, etc. [27,44,51,53]. To plausibly composite an object in a background image, one can match their camera parameters. One way to achieve this is to match the horizon lines between two images [24,26]. All these methods share the same limitations as the perspective image calibration methods due to their assumptions.

3. Method

We first define Perspective Fields and show some examples on various images. Then we show how we train a network to recover Perspective Fields from a single image. Finally, we demonstrate some downstream applications that
Perspective Fields enable, including camera parameter recovery, image compositing, and object cutout calibration.

### 3.1. Definition of Perspective Fields

Each pixel $x \in \mathbb{R}^2$ on the image frame is originated from a light ray $R \in \mathbb{R}^3$ emitted from a 3D point in the world frame $X \in \mathbb{R}^3$. When the ray travels through the camera, it is bent by the lens and projected onto the image frame. We assume an arbitrary projection function $x = \mathcal{P}(X)$ that maps a point in the world to the image plane. We denote the gravity direction in the world frame to be a unit vector $g$. For each pixel location $x$, a Perspective Field consists of a unit Up-vector $u_x$ and Latitude $\varphi_x$. The Up-vector $u_x$ is the projection of the up direction of $X$, or

$$u_x = \lim_{c \to 0} \frac{\mathcal{P}(X - cg) - \mathcal{P}(X)}{||\mathcal{P}(X - cg) - \mathcal{P}(X)||_2} \tag{1}$$

The limit is not required for perspective projection since it preserves straight lines. The Latitude $\varphi_x$ of this pixel is defined as the angle between the ray $R$ and the horizontal plane, or

$$\varphi_x = \arcsin \left( \frac{R \cdot g}{||R||_2} \right) \tag{2}$$

This representation is applicable to arbitrary camera models. In Fig. 2, we illustrate the Perspective Field representation of images captured from commonly used cameras extracted from a 360° panorama. Although our representation is general, we mainly focus on perspective projection to compare with existing works and leave extensive applications to other camera models for future work.

### 3.2. Estimating Perspective Fields

Our goal is to train a neural network to estimate Perspective Fields from a single image. To do this, we introduce PerspectiveNet (Fig. 3 left), an end-to-end network that takes a single RGB image as input and outputs a per-pixel value for Up-vector and Latitude. Unlike previous camera calibration works where the network outputs a single vector of camera parameters [24, 29], the output of our system has the same dimension as the input, making it amenable to pixel-to-pixel architectures [6, 39, 49]. We train our PerspectiveNet on crops from 360° panoramas with cross entropy loss $L_{\text{pers}}$ (see Sec. 3.3.)

#### Camera parameters from Perspective Fields.

When camera parameters from specific models are needed, we can recover the camera parameters from Perspective Fields. For instance, if we parameterize perspective projection by roll, pitch, and field of view following [24, 29], and optionally the principal point location, we can represent these with a vector $\theta$. As extracting these parameters requires combining potentially noisy Perspective Field estimates, we extract them by training a neural network named ParamNet that maps the Perspective Fields to the camera parameters, as shown in Fig. 3. This network is trained directly with a sum of $\ell_1$ losses $L_{\text{param}} = \Sigma \|\theta_i - \hat{\theta}_i\|_1$.

#### Perspective Fields as a metric for perspective mismatch.

Our representation is easy to interpret: the Up-vectors align with structures that are upright, such as trees and vertical lines on buildings; the Latitude values align with viewpoint direction: if the top of an upright object is visible. Therefore, we propose to use Perspective Fields agreement as a measurement for the image compositing quality between a foreground object and background scene. We propose Perspective Field Discrepancy (PFD), which is defined as the sum of the difference between the Up-vectors and the Latitude values, or

$$\mathcal{E}_{\text{PFD}} = \lambda \arccos(u_1 \cdot u_2) + (1 - \lambda)\|\ell_1 - \ell_2\|_1, \tag{3}$$

where $u_i$ is the Up-vector and $\ell_i$ is the Latitude value. The weight $\lambda = 0.5$ is used in our experiments. Both the Up-vector and the Latitude are in an angular space, so we can take a weighted sum of their angular differences. We aggregate the metric by averaging the PFD over all the pixels.
denoted as APFD. The experiment in Sec. 4.3 shows that the proposed metric strongly correlates with human perception.

**Object cutout calibration.** Image composition often involves compositing a segmented object with a scene. As foreground objects contain little to no background information, camera calibration methods, including our scene level Perspective Field prediction network, fail on such images due to the domain gap between the panorama training data and the real object images (see Table 2).

We can easily train Perspective Fields on objects by taking COCO [32] and doing distillation training using our scene level model as a teacher. Since the Perspective Fields are stored per-pixel, we can crop out an object in the image and its corresponding pseudo ground truth Perspective Field to form a new training pair.

### 3.3. Implementation details

To learn Perspective Fields from single images, we use the architecture of SegFormer [49] with Mix Transformer-B3 encoder which was originally used for semantic segmentation tasks. The transformer based encoder is effective to enforce global consistency in the Perspective Fields. We use two decoder heads to output a per-pixel probability over discretized Latitude and Up-vector bins. We use cross-entropy loss \( \ell_{\text{pers}} = \ell_{\text{CE}} \), which we empirically found better than regression. The ParamNet uses ConvNet-tiny [33] to predict a vector of camera parameters trained with \( \ell_1 \) loss.

### 4. Experiments

**Overview.** In the following experiments, we study three questions. First, (Sec. 4.1), can methods that recover a global set of camera parameters (e.g. pitch) produce accurate Perspective Fields. We verify that directly producing Perspective Fields produces more accurate camera calibrations, especially on cropped images. We then ask in Sec. 4.2 the reverse statement: whether our Perspective Field method can be used to recover global camera parameters well. We find that our method matches and often outperforms previous methods on images with a centered principal point and substantially outperforms these methods on cropped images. Next, we ask whether errors in Perspective Field error is meaningful. We conduct a user study in Sec. 4.3 to evaluate our proposed metric with human perception and show that humans are more sensitive to the Perspective Fields discrepancy than other existing measurements on image perspective. We finally show image editing applications from Perspective Fields in Sec. 4.4.

#### 4.1. Predicting Perspective Fields

We first evaluate our PerspectiveNet on both natural scenes and object-centric images.

**Training data and training details.** We train our network on a diverse dataset of panorama scenes which includes 30,534 indoor, 51,157 natural and 110,879 street views from 360Cities.† Although we can generate arbitrary types of image projections from the panoramas, we choose to train on perspective images for a fair comparison with previous methods. To do this, we uniformly sample crops from the panoramas with camera roll in \([-45^\circ, 45^\circ]\), pitch in \([-90^\circ, 90^\circ]\) and FoV in \([30^\circ, 120^\circ]\). Our training and validation set consist of 190,830/1,740 panorama images respectively. We augment training data with random color jittering, blurring, horizontal flipping, rotation and cropping. We later show results on other camera models such as fisheye images.

**Ours-distill:** We distill our network on COCO [32] images by using pseudo ground truth predicted by our scene level network. We crop out the foreground object and the pseudo ground truth to generate the training pairs, and randomly (70% of the time) remove the background of the image using segmentation masks as data augmentation to generalize to object cutouts.

**Test data.** We test generalization of different methods on publicly available datasets including Stanford2D3D [4] and TartanAir [45] where ground truth camera parameters are available. None of the methods compared were trained on the test set. Stanford2D3D is an indoor panorama dataset where arbitrary camera views can be extracted. TartanAir is a photo-realistic dataset captured by drones with extreme viewpoint and diverse scenes (indoor, outdoor, natural, and man-made structures) rendered with different lighting and weather conditions. Assuming perspective projec-

---

† https://www.360cities.net/
Table 1. Quantitative evaluation for scene-level Perspective Field prediction. Perturb: None on centered principal point images; Crop on uncentered principal point images. We re-implement Percep. [24] using the same backbone and training data as ours. None of the methods have been trained on Stanford2D3D [4] or TartanAir [45]. Results on warped test data and qualitative results are in the supp.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Perturb</td>
<td>Mean ↓</td>
<td>Median ↓</td>
<td>% &lt; 5° ↑</td>
<td>Mean ↓</td>
<td>Median ↓</td>
<td>% &lt; 5° ↑</td>
<td>Mean ↓</td>
<td>Median ↓</td>
<td>% &lt; 5° ↑</td>
</tr>
<tr>
<td>Upright [28]</td>
<td>None</td>
<td>3.63</td>
<td>3.28</td>
<td>64.97</td>
<td>7.03</td>
<td>7.03</td>
<td>41.12</td>
<td>3.53</td>
<td>3.19</td>
<td>65.36</td>
</tr>
<tr>
<td>Percep. [24]</td>
<td>None</td>
<td>3.58</td>
<td>3.32</td>
<td>64.19</td>
<td>6.27</td>
<td>6.07</td>
<td>42.36</td>
<td>7.30</td>
<td>6.86</td>
<td>47.04</td>
</tr>
<tr>
<td>CTRL-C [29]</td>
<td>None</td>
<td>7.39</td>
<td>6.87</td>
<td>42.49</td>
<td>10.21</td>
<td>9.96</td>
<td>28.67</td>
<td>8.64</td>
<td>7.67</td>
<td>41.21</td>
</tr>
<tr>
<td>Ours</td>
<td>None</td>
<td>2.18</td>
<td>1.88</td>
<td>82.83</td>
<td>3.40</td>
<td>3.06</td>
<td>68.27</td>
<td>3.47</td>
<td>2.86</td>
<td>67.45</td>
</tr>
</tbody>
</table>

Figure 4. Qualitative results on Objectron [1]. The top two rows show the results on the original image crops. The bottom row shows the results on isolated object images. Upright [28] and Perceptual measure [24] often fail dramatically on these images. Up-vectors are shown in the green vectors. Latitude are visualized by colormap: −π/2 to π/2.

For object-centric test images, we randomly sample 2,415 views from Stanford2D3D with camera roll in [−45°, 45°], pitch in [−50°, 50°] and FoV in [30°, 120°]. For TartanAir, we randomly sample 2,000 images from its test sequences with roll ranging in [−20°, 20°], pitch in [−45°, 30°], and fixed FoV (74°). To test the robustness of methods, we add image crop perturbation to the test image, details in supp.

Eq. 2. We compare our method with the following baselines: Upright [28], Perceptual measure [24] and CTRL-C [29], among which Upright is the only non-learning based method. They all predict camera roll, pitch, and FoV from a single RGB image and assume that the principal point is at the image center. From the predicted camera parameters, we calculate their Perspective Fields for evaluation. We re-implement [24] using the same backbone and train it on our data. For Upright and CTRL-C, we use the official model and code for evaluation. None of these methods have seen any training data from the test datasets.

Metrics. We calculate the angular error of Up-vector and Latitude fields and report three metrics: the mean error (Mean), median error (Med), and fraction of pixels with error less than a threshold (in our case 5°). For methods that output camera parameters, we convert the predicted parameters to Perspective Fields.

Results on scene images. We show the results on Stanford2D3D and TartanAir in Table 1. Predicting the Perspective Fields is more effective than recovering camera parameters from previous methods. On centered principal point images (Perturb: None), our method outperforms the second best by a large margin. On shifted principal point images (Perturb: Crop), (simulating images found in the wild that have undergone cropping), our method has less degradation than previous baselines. Our performance on Up-vector prediction is robust to cropping, with comparable numbers (4% drop in % < 5° of Up on Stanford2D3D). Other methods, have large performance drop in both Up-vector and Latitude prediction in this setting. Nevertheless, our method outperforms the competing methods on Latitude. Visual results can be found for qualitative evaluation in our supplementary material.

Results on object-centric images. The results on the Objectron dataset [1] are shown in Table 2. Our model trained on COCO (Ours-distill) using the proposed distillation method significantly improves over its teacher model, especially for isolated object images. Both our teacher model trained on panorama scene images and [24] have a
big performance drop on isolated object images due to a major data domain gap. In contrast, Upright and Ours-distill only have a mild performance drop. It is quite surprising that CTRL-C’s accuracy on Latitude field improves on the isolated object images. We suspect that the structure in the background might contradict CTRL-C’s assumption of a centered principal point, as the the cropping will often shift the principal point. Our student model (Ours-distill) achieves overall better accuracy than the baselines. Some visual results are shown in Fig. 4.

We also test each method on some challenging in-the-wild web images in Fig. 5. These web images may have been cropped or warped for aesthetic composition. Camera calibration methods with rigid scene and camera assumptions cannot robustly handle these images. Our method tends to provide better estimations. More results can be found in our supplementary material.

**Generalization to non-perspective projections.** In this section, we ask whether we can recover the Perspective Field for images with non perspective properties without explicitly training on them. We take advantage of the local representation and use a sliding window inference technique for images that are out of our training distribution. We inference on small crops and aggregate the prediction for each pixel from overlapping windows. Using this technique, we show in Fig. 6 - Sliding Win. that, without fine-tuning, the recovered Up-vectors are already tangential to the upward curves and the horizon line is curved, which is close to the ground truth. In Fine-tune, we show results after fine-tuning on distorted images, e.g. Fig. 2-(E), which has comparable predictions in Up-vectors and slightly better predictions in Latitude. In Fig. 6 row 2, we show results on a challenging multiperspective image from the Inception movie, using the same sliding window technique. The network is able to pick up the negative Latitude on top of the building and the Up-vector distortion at the top right corner. In row 3 and 4, we show more results from the PerspectiveNet on art works with non physically plausible cameras.

### 4.2. Camera parameter estimation

We have shown in Sec. 4.1 that Perspective Fields from predicted camera parameters are less effective than directly predicting them; can camera parameters be effectively recovered from Perspective Fields? In this section, we use the ParamNet in Sec. 3.2 Fig. 3 to recover camera parameters from Perspective Field predictions and compare with methods that directly predict them [24, 28, 29].

**Setup.** We test on perspective images and recover roll, pitch, and FoV for uncropped images as well as principal point for cropped images. All methods are trained and evaluated on Google Street View (GSV) [2] for fair comparison, following [29]. Besides the test images used in [29], we generate a more general set of uncentered principal-point images by cropping. See Supp for detailed dataset settings.

**Metrics.** Since FoV is undefined for cropped images, we define it (FoV*) as follows: Denote camera pinhole as $O$...
and the middle points of the top and bottom edges of the image as \( M_1, M_2 \). FoV is the angle between \( OM_1 \) and \( OM_2 \). The principal point location (\( cx \) and \( cy \)) is relative to the image size. We use \( \epsilon_1 \) error between the prediction and ground truth as our metric for all camera parameters (Roll, Pitch, FoV, \( cx, cy \)). We also measure the Up and Latitude errors of the Perspective Fields recovered from the predicted camera parameters using Eq. 1 and 2 to show the impact of errors from camera parameters on Up and Latitude.

**Results.** We start with the principal-centered test set that previous methods use, which is a highly constrained setting. We report the angle differences of roll, pitch, and FoV in Table 4. Our method gets comparable performance to the baselines. We then test on a more general cropped dataset, where images have an uncentered principal point. As shown in Table 3, our method outperforms all the other baselines by 40% on roll and pitch errors and 40% on pitch error. Our Perspective Field representation is a dense local representation, which leads to a robust way to estimate the global camera parameters. The Up-vector field and the Latitude field provide interpretable cues for the estimation of camera roll and pitch respectively.

We then test on a more general cropped dataset, where images have an uncentered principal point. As shown in Table 3, our method outperforms all the other baselines by a large margin on all metrics. Compared to CTRL-C, we reduce the error on roll (19%), pitch (40%), and FoV (31%), which reflects a large increase in Latitude accuracy by over 13% and Up-vector accuracy by 2%. None of the baselines above handles principal point by design, therefore, we perform two ablations: 1) No \( L_{\text{pers}} \): we train without \( L_{\text{pers}} \) but with \( L_{\text{param}} \). This is a network that predicts parameters end-to-end. Compared to Perceptual [24], it has the same backbone followed by the ParamNet while additionally predicting the principal point. It differs from our method by lacking the \( L_{\text{pers}} \). This is to test whether using Perspective Fields as the intermediate representation is helpful. Results show that it works better than existing methods but is still worse than Ours. 2) No Center Shift: our ParamNet in Table 4 assumes a central principal point and only predicts roll, pitch and FoV. We improve the relative principal point shift accuracy by over 36%. By recovering the principal point, we improve Up-vector accuracy (\( \% < 5^\circ \)) by >4% and Latitude accuracy by >6%.

Although our method does not directly learn to predict the camera parameters, our results show that they can be recovered from the Perspective Fields alone without RGB data and still outperform SOTA methods.

### 4.3. User study for perspective matching metrics

To validate our proposed APFD metric, we conduct a user study to analyze its correlation with human perception of perspective consistency for image compositing.

Given a background image with known camera settings, we render a 3D object with 10 randomly perturbed cameras and composite it to the background. Example images are in the supp material. These 10 images are ranked by participants using a two alternative forced-choice (2AFC) test. In this test, two composites are displayed side by side, and a user picks the one that looks better in term of perspective consistency. We compute the Pearson’s correlation coefficient of APFD w.r.t. the human ranking scores. We receive 18 votes for each image pair and we repeat the experiment on 8 scenes (background-object pairs).

Table 5 shows the median correlation scores on the 8 scenes for different metrics. See the supplementary material for full statistics and more details of the test. Our proposed APFD metric has the highest correlation with human perception. The Up-vector field error captures the local perspective distortion well, thus providing good performance among individual metrics. The APFD metric which combines both Up-vector and Latitude gives a slightly higher correlation score. Single camera parameter metrics have widely varying correlations. Among them, deviation in FoV is a poor indicator of human perception, also shown in [24]. The change in pitch is a dominant factor in perspective mis-

---

**Table 3. GSV uncentered principal-point results.** Our method recovers the principal point and outperforms the baselines on all metrics.

<table>
<thead>
<tr>
<th>Method</th>
<th>Roll (°) ↓</th>
<th>Pitch (°) ↓</th>
<th>FoV (°) ↓</th>
<th>Roll (°) ↓</th>
<th>Pitch (°) ↓</th>
<th>FoV (°) ↓</th>
<th>cx ↓</th>
<th>cy ↓</th>
<th>Up(°) ↓</th>
<th>Latitude(°) ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upright [28]</td>
<td>2.73</td>
<td>1.55</td>
<td>6.42</td>
<td>10.50</td>
<td>7.67</td>
<td>3.59</td>
<td>3.59</td>
<td>2.72</td>
<td>1.83</td>
<td>74.57</td>
</tr>
<tr>
<td>Perceptual [24]</td>
<td>2.39</td>
<td>1.45</td>
<td>5.26</td>
<td>8.47</td>
<td>7.22</td>
<td>3.56</td>
<td>-</td>
<td>-</td>
<td>1.70</td>
<td>93.15</td>
</tr>
<tr>
<td>CTRL-C [29]</td>
<td>1.92</td>
<td>1.21</td>
<td>4.51</td>
<td>5.57</td>
<td>4.66</td>
<td>3.07</td>
<td>-</td>
<td>-</td>
<td>1.29</td>
<td>96.84</td>
</tr>
<tr>
<td>Ours</td>
<td>1.37</td>
<td>0.97</td>
<td>2.60</td>
<td>3.75</td>
<td>3.19</td>
<td>0.09</td>
<td>0.09</td>
<td>0.08</td>
<td>1.05</td>
<td>98.95</td>
</tr>
</tbody>
</table>

**Table 4. Results on GSV centered principal-point images.** Our method has comparable performance to the baselines.

<table>
<thead>
<tr>
<th>Method</th>
<th>Roll (°) ↓</th>
<th>Pitch (°) ↓</th>
<th>FoV (°) ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upright [28]</td>
<td>6.19</td>
<td>0.53</td>
<td>1.36</td>
</tr>
<tr>
<td>Perceptual [24]</td>
<td>0.94</td>
<td>0.53</td>
<td>1.36</td>
</tr>
<tr>
<td>CTRL-C [29]</td>
<td>0.66</td>
<td>0.53</td>
<td>1.36</td>
</tr>
<tr>
<td>Ours</td>
<td>0.66</td>
<td>0.52</td>
<td>1.36</td>
</tr>
</tbody>
</table>

---

**Table 5. Pearson’s correlation for different metrics w.r.t. human perception.** Our APFD metric has strongest correlation with human perception. More statistics and visual results are in the supp.
match. Summing the parameter difference (Camera-All) does not improve correlation scores, which shows the difficulty of using camera parameters to measure perceived perspective consistency. The horizon line used in [24] performs comparably with our Latitude field metric, since they measure similar quantities, however horizon lines are not always visible in images.

4.4. Image editing applications

We conclude with applications of Perspective Fields.

Perspective-aware image recommendation. Perspective Fields can guide the retrieval of images from a database of 2D images. Our method works on images with extreme viewpoints, while horizon-based perspective matching methods like [24] fail on this type of image because the horizon is far outside the image. We demonstrate this in Fig. 7, where we estimate Perspective Fields on 10 images of hot air balloons with diverse view angles. We use our Perspective Field metric to retrieve the best balloon sprite based on the bounding box given by the user (yellow and green boxes). The system calculates the APFD error (Eq. 3) between the background and foreground fields, and adjusts the best candidate with a similarity transformation to better align the Perspective Fields with the background. The left columns rank the balloon sprites by error from low to high.

AR effect. Our Perspective Fields can be used in AR effect applications related to gravity e.g. simulating snows, hanging a chandelier to the ceiling, etc. In Fig. 8, we demonstrate rain effect using the Up-vector prediction. Our compositing w/ Up which considers the Up-vector prediction looks more natural than vertical raindrops in the image frame w/o Up.

3D object insertion. Our Perspective Fields can be used to achieve better compositing. In Fig. 9, we render 3D mod-

Figure 7. Given selected locations on a skyscraper image, our system computes the local Perspective Field for the background and retrieves foreground objects from a set of air balloons that best match the predicted fields. Left: ranking of air balloon images for two insertion locations. APFD error shown at the bottom. Middle: background image with two boxes as the insertion locations (top) and predicted Perspective Fields (bottom). Right: image composition with 2D rotation adjustment of foreground sprites (top) and visualization of Up-vector fields after compositing (bottom).

Figure 8. Rain effect based on Perspective Fields. See supp video for a dynamic composite. Our compositing w/ Up which considers the Up-vector prediction looks more natural than vertical raindrops in the image frame w/o Up.

Figure 9. On the left half: we estimate camera by [24] and insert a 3D lamp and a Doctor Who Police Box. One may notice that it looks a bit off, and especially that the tip of the lamp is tilted. This is because [24] predicts pitch $\approx 7^\circ$ which matches the horizon line, but causes distortion in the up direction in that region. In contrast, on the right half, using our Perspective Fields to estimate the camera view, we can correctly maintain the perspective consistency for the temple on the right. The white dashed lines intersect at the horizon location, the green dashed lines are up direction.

5. Conclusion

We propose Perspective Fields, an over-complete representation capturing the local perspective properties of an image. We introduce a neural network model to predict Perspective Fields for various image types, and ParamNet which recovers camera parameters directly from Perspective Fields. Perspective Fields can serve as a metric to quantify perspective matching quality in image compositing. As a local representation, it is robust to different camera models and lens types, and several image editing operations, a complete study of which we leave to future work.

Acknowledgements. This work was partially funded by the DARPA Machine Common Sense Program. We thank Geoffrey Oxholm for the help with Upright, and Aaron Hertzmann, Scott Cohen, Ang Cao, Dandan Shan, Mohamed El Banani, Sarah Jabbour, Shengyi Qian for discussions.
References

[31] Xiaoyu Li, Bo Zhang, Pedro V Sander, and Jing Liao. Blind geometric distortion correction on images through deep learning. In CVPR, 2019. 2
[33] Zhuang Liu, Hanzhi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. CVPR, 2022. 4


[38] Linfei Pan, Marc Pollefeys, and Viktor Larsson. Camera pose estimation using implicit distortion models. In *CVPR*, 2022. 2


[40] Carsten Rother. A new approach to vanishing point detection in architectural environments. *BMVC*, 2002. 2

[41] Chester Slama, C Theurer, and SW Henriksen. The manual of photogrammetry the american society of photogrammetry. *Falls Church, VA*, 1980. 2


[47] Scott Workman, Menghua Zhai, and Nathan Jacobs. Horizon lines in the wild. *BMVC*, 2016. 2


[50] Lingzhi Zhang, Tarmily Wen, Jie Min, Jiancong Wang, David Han, and Jianbo Shi. Learning object placement by inpainting for compositional data augmentation. In *ECCV*, 2020. 2
