Deep vanishing point detection: Geometric priors make dataset variations vanish

Yancong Lin, Ruben Wiersma, Silvia L. Pintea, Klaus Hildebrandt, Elmar Eisemann, and Jan C. van Gemert
Delft University of Technology, The Netherlands

Abstract

Deep learning has improved vanishing point detection in images. Yet, deep networks require expensive annotated datasets trained on costly hardware and do not generalize to even slightly different domains, and minor problem variants. Here, we address these issues by injecting deep vanishing point detection networks with prior knowledge. This prior knowledge no longer needs to be learned from data, saving valuable annotation efforts and compute, unlocking realistic few-sample scenarios, and reducing the impact of domain changes. Moreover, the interpretability of the priors allows to adapt deep networks to minor problem variations such as switching between Manhattan and non-Manhattan worlds. We seamlessly incorporate two geometric priors: (i) Hough Transform – mapping image pixels to straight lines, and (ii) Gaussian sphere – mapping lines to great circles whose intersections denote vanishing points. Experimentally, we ablate our choices and show comparable accuracy to existing models in the large-data setting. We validate our model’s improved data efficiency, robustness to domain changes, adaptability to non-Manhattan settings.

1. Introduction

Vanishing point detection in images has non-vanishing real-world returns: camera calibration [1, 9, 21], scene understanding [18], visual SLAM [13, 33], or even autonomous driving [28]. Deep learning is an excellent approach to vanishing point detection [7, 8, 67, 69], where all geometric knowledge is learned from large annotated data sets. Yet, in the real-world, there are several factors that complicate deep learning solutions: (1) Manually annotating large training sets is expensive and error prone; (2) Training models on large data sets require costly computational resources; (3) Practical changes to data collection cause domain shifts, hampering deep network generalization; (4) Slight changes in the problem setting require a complete change in deep network architectures. Thus, there is a need to make deep learning less reliant on data, and its architectures more robust to variants of the same problem.

In this paper, we add geometric priors to deep vanishing point detection. Using geometric priors is data-efficient as this knowledge no longer needs to be learned from data. Thus, fewer annotations and compute resources are needed. Moreover, by relying on priors, the model is less sensitive to particular idiosyncrasies in the training data and generalizes better to domains with slightly different data distributions. Another advantage of a knowledge-based approach is that it is interpretable, and thus the architecture is easy to adapt to a slightly different problem formulation.

We add two geometric priors, see Fig. 1: (i) Hough Transform and (ii) a Gaussian sphere mapping. Our trainable Hough Transform module represents each line as an (offset, angle) pair in line polar coordinates, allowing us to identify individual lines in Hough space [15]. We subsequently map these lines from Hough space to the Gaussian sphere, where lines become great circles and vanishing points are located at the intersection of great circles [4]. The benefit of using great circles is that lines are mapped from the unbounded image plane to a bounded unit sphere, facilitating vanishing point detection outside the image view. Both the Hough Transform and the Gaussian sphere mapping are end-to-end trainable, taking advantage of learned representations, while adding knowledge priors.
This paper makes the following contributions: (1) we add two geometric priors for vanishing point detection by mapping CNN features to the Hough Transform, and mapping Hough bins to the Gaussian sphere; (2) we validate our choices and demonstrate similar accuracy as existing models on the large ScanNet [12] and SceneCity Urban 3D [70]; (3) we show that adding prior knowledge increases data-efficiency, improving accuracy for smaller datasets; (4) we demonstrate our ability to tackle a different problem variant: detecting a varying numbers of vanishing points on the NYU Depth [45] dataset, where the number of vanishing points varies drastically from 1 to 8; (5) we show that adding prior knowledge reduces domain shift sensitivity, which we validate by cross-dataset testing.

2. Related work

Geometry-based vanishing point detection. Vanishing points occur at intersections of straight lines. Lines can be found by contour detection [71] or a dual point-to-line mapping [30]. The common approach, however, is using an explicit straight-line parameterization in the Hough Transform [40,47,51]. We exploit this straight-line parameterization as prior knowledge in a Hough Transform module.

Combining lines to vanishing points can be done by measuring the probability of a group of lines passing through the same point [57], voting schemes [19,64], or hypothesis testing by counting the number of inlier lines such as J-Linkage [59] used in [17,58]. Other approaches see vanishing point detection as a grouping problem by applying line clustering [3,43,53], expectation-maximization [1,14,27,54], or branch-and-bound [5,6,23]. While these methods work well, they do not exploit prior knowledge of the 3D world.

A strong geometric prior for vanishing points is modeled by the Gaussian-sphere [4,10]. A line in an image represents a great circle on the Gaussian-sphere and the intersections of great circles on the sphere denote vanishing points, detected as local maxima [4,51]. Mapping lines to the Gaussian sphere shifts the problem from the unbounded image plane to the constrained parameter space defined by the Gaussian sphere [10,41,56]. Constraining the search space is a form of regularization, and is particularly beneficial for limited-data deep learning. We exploit this prior knowledge by incorporating the Gaussian sphere mapping.

Learning-based vanishing point detection. Vanishing point detection can be learned from large annotated datasets [7,8,66,67]. It is effective to split the problem in separate stages: line detection, inverse gnomonic projection, network training and post-processing, as in Kluger et al. [25]. Conic convolutions on hemisphere points, provided further improvements, in Zhou et al. [69]. In contrast, rather than focusing on accurate large-scale deep models, we consider challenging real-world scenarios, such as: limited training samples, cross-dataset domain switch, and non-Manhattan world.

Robustness to domain shifts. Classical solutions to vanishing point detection [17,35,55] are built exclusively on prior-knowledge. Such methods are data-free, and thus designed to work on any domain. Yet, they cannot take advantage of expressive deep-feature learning for vanishing point detection [38,69]. On the other hand, deep models are notoriously sensitive to distribution shifts between training and test [29,39]. Active research on this includes: domain adaptation [50,61,65], domain generalization [68], multi-domain learning [36,52], etc. Such solutions entail significant changes to the deep network model, adding complexity for practical real-world applications. Hence, we focus on a single method which does not requiring large model changes for robustness to minor domain shifts. Our goal is to combine the robustness of knowledge-based methods, with the power of deep representation learning.

Manhattan versus non-Manhattan world. The Manhattan world assumes exactly 3 vanishing points. This assumption has been proven useful for orthogonal vanishing point detection [2,5,44,63]. However, the Manhattan assumption does not hold in several real-world scenarios such as non-orthogonal walls and wireframes in man-made structures. Vanishing point detection in non-Manhattan world is done by robust multi-model fitting [26], horizon line detection [66], branch-and-bound with a novel mine-and-stab strategy [32], Bingham mixture model fitting [31] or non-maximum suppression on the Gaussian sphere [38]. In our work, we refrain from adding explicit orthogonality constraints, which makes our method applicable to non-Manhattan scenarios as well. We rely on the Hough Transform and the Gaussian sphere to map pixel-wise representations to the entire hemisphere. And using a clustering algorithm we detect multiple vanishing points simultaneously.

3. Geometric priors for VP detection

General outline of our approach. Fig. 2 depicts the overall structure of our model. We build on two geometric priors: (i) Hough Transform, and (ii) Gaussian sphere mapping. A CNN learns image features, which are then mapped to a line parameterization via Hough Transform. We project the features of parameterized lines to the Gaussian sphere where spherical convolutions precisely localize vanishing points.

(i) Hough Transform. Similar to [69], we use a single-stack hourglass network [46] to extract image features, $F$ to be mapped into Hough space [37], $HT$. The $HT$ space parameterizes image lines in polar coordinates using a set of discrete offsets $\rho$ and discrete angles $\theta$, defining a 2D discrete histogram. In practice, a set of pixels $(x(t), y(t))$ along a line indexed by $t$, vote for a line parameterization to
The Hough Transform module starts from an $[H \times W]$ feature map $F$ and outputs an $[N_p \times N_{\theta}]$ Hough histogram $HT$, where $N_p$ and $N_{\theta}$ are the number of sampled offsets and angles in Hough Transform. We set $H=128$, $W=128$, $N_p=184$, and $N_{\theta}=180$. This results in $[N_p \times N_{\theta}]$ possible line parameterizations. We find the local maxima in the Hough domain by performing a 1D convolution over the offsets. This removes the noisy responses in the Hough space, as in Fig. 2. We refer the readers to [37] for details.

(ii.1) Gaussian sphere mapping. The Gaussian sphere is a unit sphere centered at the camera origin, $O$. Vanishing points on the sphere are represented as normalized 3D line directions $\delta$.

Starting from a bin in the Hough domain $(\rho_{AB}, \theta_{AB})$, corresponding to a line direction in the image plane $AB$, we want to map this to the Gaussian sphere. Two image points $A$ and $B$ sampled from a line represented by its $HT$ bin $(\rho_{AB}, \theta_{AB})$, together with the camera center $O$, form a plane $\psi$ as depicted in Fig. 3(a). The plane $\psi$ is described by its normal vector:

$$\mathbf{n} = (n_x, n_y, n_z) = \frac{\mathbf{OA} \times \mathbf{OB}}{\| \mathbf{OA} \times \mathbf{OB} \|}. \quad (2)$$

This normal vector $\mathbf{n}$ is the only information we need to map the image line direction $AB$ to the Gaussian sphere.

The spherical coordinates $(\alpha, \beta)$ describe a point on the Gaussian sphere, where $\alpha$ is the azimuth defined as the angle from the $z$-axis in the $xz$ plane, and $\beta$ is the elevation representing the angle measured from the $xz$ plane towards the $y$-axis, as shown in Fig. 3(a). The intersection between the plane $\psi$ and the Gaussian sphere is a great circle. This great circle represents the projection of the image line direction $AB$ on the Gaussian sphere. Intersections of multiple great circles are potential vanishing points, see Fig. 3(b).

We compute the projection of the image line direction $AB$, by estimating the elevation $\beta$ as a function of the azimuth $\alpha$ and the normal vector $\mathbf{n}$ [51]:

$$\beta(\alpha, \mathbf{n}) = \tan^{-1} \frac{n_x \sin \alpha - n_z \cos \alpha}{n_y}, \quad (3)$$

where we uniformly sample $\alpha$ in the range $[-\pi/2, \pi]$.

Because the Gaussian sphere is symmetric we only need a hemisphere. We sample $N$ points on the Gaussian hemisphere using the Fibonacci lattice [20], and then project lines, corresponding to bins in the Hough space, to these $N$ sampled sphere points. For each line parameterization in Hough space $(\rho, \theta)$, we first compute its normal vector $\mathbf{n}$. We, then, estimate its corresponding $(\alpha, \beta)$ spherical coordinates using Eq. (3). We subsequently assign each $(\alpha, \beta)$ pair to its nearest neighbor in the sampled points from the Fibonacci lattice, by computing their cosine distance. To parallelize this process, we precompute the projection of all Hough line parameterizations onto the sampled sphere locations. This mapping is stored in an $[N_p \times N_{\theta} \times M]$ tensor, where $[N_p \times N_{\theta}]$ is the number of line parameterizations in Hough space and $M$ is the number of sampled azimuth angles $\alpha$. We set $N=32,768$ and $M=1,024$.

(ii.2) Spherical convolutions on the hemisphere. We employ spherical convolutions to predict vanishing points. We treat the points sampled on the hemisphere as a point cloud and use EdgeConv [52] to convolve over the hemisphere. EdgeConv operates on a k-nearest neighbor graph on the
Figure 3. Gaussian sphere representations for vanishing points [4, 51]: (a) The Gaussian sphere is a unit sphere located at the camera center, O. Points on the sphere are encoded by two angles: (α, β) the azimuth and the elevation, respectively. A line segment AB in the image plane together with the camera center O forms a plane ψ, highlighted in blue. To define the mapping from the image to the sphere, we only need to know the normal $\vec{n}$ to the plane ψ. (b) Image lines are projected as great circles on the sphere. The intersection of multiple great circles on the sphere represents a vanishing point.

Figure 4. Spherical convolutions on the hemisphere. We use EdgeConv [62] for precise vanishing point localization on the Gaussian sphere. The concatenation of the previous feature maps is fed into the final layer to produce a prediction.

Figure 5. Multi-scale sampling on the hemisphere. We sample points at three scales for detecting vanishing points in the Manhattan world, as in [69]. Blue indicates sampling at the first scale, while green indicates fine-grained sampling at the following scales. The red crosses are the predictions at each scale.

Model training and inference. We train the model using the binary cross-entropy loss. For each annotated vanishing point, we label its nearest neighbor in the sampled points as +1 and the others as 0. Because the number of positive samples is considerably lower than the negative samples, we compute two separate average losses over the positive and the negative samples, and then sum these. There is no intermediate supervision or guidance.

During inference, we use DBSCAN [16, 49] to cluster all points on the Gaussian sphere based on the cosine distance. The $\text{eps}$ parameter of DBSCAN [49] is set to be 0.005. The point with the highest confidence in each cluster is the prediction. We rank all predictions by confidence.

Multi-scale sampling with the Manhattan assumption. For the Manhattan world, we know beforehand that there are only 3 orthogonal vanishing points, therefore in this scenario, we can use a multi-scale sampling strategy to reduce computation, as in [69]. Here, we sample points and apply spherical convolutions at 3 scales: $\delta \approx \{90°, 13°, 4°\}$ and $N = \{512, 128, 128\}$, where $\delta$ controls the sampling radius and $N$ indicates the number of sampled points respectively. Fig. 5 displays the multi-scale sampling. The spherical convolution networks share the same architecture while processing different number of samples. We provide details in the supplementary material.

4. Experiments

Datasets. We evaluate on three datasets following the Manhattan world assumption: SU3 (SceneCity Urban 3D) [70], ScanNet [12], YUD [14], as well as the NYU Depth [45] dataset which does not follow the Manhattan world assumption. The SU3 dataset contains 23K synthetic images,
2. LSD + Sphere-Convs
3. Canny-edge + HT-Conv + Sphere-Convo4
4. Conv + HT-NMS + Sphere-Convo1
5. Conv + HT-Conv + Sphere-Convo4 (Ours)
6. Conv + HT-Conv + Sphere-Convo2
7. Conv + HT-Conv + Sphere-Convo1
8. Conv + HT-Conv + Convx4

<table>
<thead>
<tr>
<th>HT (# of angles)</th>
<th>Sphere (# of points)</th>
</tr>
</thead>
<tbody>
<tr>
<td>90</td>
<td>77.1</td>
</tr>
<tr>
<td>180</td>
<td>79.3</td>
</tr>
<tr>
<td>8K</td>
<td>73.3</td>
</tr>
<tr>
<td>16K</td>
<td>77.4</td>
</tr>
<tr>
<td>32K</td>
<td>79.3</td>
</tr>
</tbody>
</table>

Table 1. **Quantization analysis on SU3-10% subset.** Denser samplings improve performance. In practice, we uniformly sample 180 angles from $[0, \pi)$ for HT, and 32K points on the sphere.

Figure 6. **Exp 1: Model choices.** We show the effects of the two geometric priors quantitatively on the ScanNet-1% subset. Adding HT layers and spherical convolutions outperforms the baselines, thus demonstrating the effectiveness of geometric priors.

which are split into 80%, 10% and 10% for training, validation and testing respectively. The ScanNet has more than 200K real-world images, among which 189,916 examples are used for training. The “ground truth” VP s are estimated from surface normals as in [69], thus being less precise than other datasets. In the NYU Depth dataset, the number of vanishing points varies from 1 to 8 across images, making it more challenging. The NYU Depth dataset has 1,449 images, approximately $\times 200$ and $\times 20$ smaller than the ScanNet dataset and the SU3 dataset, respectively, further increasing the difficulty of training CNN models. We additionally demonstrate the effect of geometric priors on the small-scale YUD dataset with only 102 images. Detailed comparisons are in the supplementary material. Unless specified otherwise, we use the ground-truth focal length on SU3, ScanNet and YUD for the Manhattan assumption.

**Evaluation.** On the SU3, ScanNet and YUD datasets (Manhattan assumption), we evaluate the angle difference between the predicted and the ground-truth vanishing points in the camera space, as in [26, 38, 69]. We then estimate the percentage of the predictions that have a smaller angle difference than a given threshold and compare the angle accuracy (AA) under different thresholds, as in [38, 69]. We use the ground-truth focal length to exploit the orthogonal constraint. On the NYU Depth dataset we follow [26] and first rank detected vanishing points by confidence, and then use the bipartite matching [11] to calculate the angular errors for the top $k$ predictions. After matching, we generate the recall curve and measure the area under the curve (AUC) up to a threshold, e.g. $10^\circ$.

**Baselines.** We compare our model with J-Linkage [17], Contrario-VP [55], Quasi-VP [34, 35], NeurVPS [69] and CONSAC [26] on SU3, ScanNet and YUD. On the non-Manhattan NYU Depth dataset we only compare with J-Linkage, T-Linkage [42], CONSAC and VaPid [38], as the other models rely on the Manhattan assumption. J/T-Linkage, Contrario-VP and Quasi-VP are non-learning methods, employing line segment detection [60]. NeurVPS and our model are end-to-end trainable, while CONSAC needs line segments as inputs. We follow the official implementations and use the default hyperparameters to reproduce all results. We do not consider the baselines [31,38,64] due the lack of code/results on certain datasets.

**Implementation details.** We implement our model in PyTorch [48], and provide the code online. Our models are trained from scratch on Nvidia RTX2080Ti GPUs with the Adam optimizer [24]. The learning rate and weight decay are set to be $4 \times 10^{-5}$ and $1 \times 10^{-5}$, respectively. To maximize GPU usage, we set the batch size to 4 and 16 when using multi-scale sampling. On the SU3 and NYU Depth datasets, we train the model for a maximum of 36 epochs, with the learning rate decreases by 10 after 24 training epochs. On the ScanNet dataset, we train for 10 epochs and decay the learning rate by 10 after 4 epochs. On the YUD dataset we use pre-trained models on SU3.

4.1. Exp 1: Evaluating model choices

We evaluate on a subset of ScanNet containing 1% of the data, and provide the results in Fig. 6. Model (1) is a non-learning baseline using a classic line segment detector (LSD) [60] and non-maximum suppression (NMS) on the sphere. Model (2) replaces the NMS with spherical convolutions, but still shows inferior result as LSD fails to detect reliable line segments. Model (3) combines a Canny-edge detector, Hough Transform and spherical convolutions. Comparing (3-5) indicates the added value of learning semantics from images, rather than using classic edge detectors. Comparing (4-5) shows the effectiveness of backpropagating through Hough Transform. Comparing (5-8) exemplifies the added value of spherical convolutions. Our method combines both classical and deep learning approaches into an end-to-end trainable model.

We also evaluate the impact of quantizations numerically on the synthetic SU3-10% subset, which contains precise VP annotations, thus making quantization a crucial factor. As shown in Tab. 1, fine-grained sampling is essential for a better result.
Table 2. Exp 2: Manhattan world. Angular accuracy on SU3, ScanNet and YUD datasets. Ours achieves the best results on the YUD dataset, and is competitive on the larger ScanNet and SU3 datasets. Ours\(^\dagger\) adopts the multi-scale sampling strategy, thus being significantly faster. \(^\dagger\) assumes unknown focal length, thus making the Manhattan assumption no longer applicable. Ours\(^\dagger\) shows a constant increase over Ours across datasets, indicating the usefulness of the orthogonal constraint. Supplementary material provides qualitative visualizations. We conclude that adding priors does not reduce accuracy in the large scale setting.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Metrics</th>
<th>Params</th>
<th>FPS</th>
<th>AA(^\circ)@3(^\circ)</th>
<th>AA(^\circ)@5(^\circ)</th>
<th>AA(^\circ)@10(^\circ)</th>
<th>SU3 [70]</th>
<th>ScanNet [12]</th>
<th>YUD [14]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>J-Linkage(^\dagger) [17]</td>
<td>—</td>
<td>1.0</td>
<td>82.0</td>
<td>87.2</td>
<td>15.7</td>
<td>60.8</td>
<td>71.8</td>
<td>81.5</td>
</tr>
<tr>
<td></td>
<td>Contrario-VP(^\dagger) [55]</td>
<td>—</td>
<td>0.6</td>
<td>64.8</td>
<td>72.2</td>
<td>12.0</td>
<td>58.6</td>
<td>70.7</td>
<td>81.8</td>
</tr>
<tr>
<td></td>
<td>Quasi-VP [34]</td>
<td>—</td>
<td>29.0</td>
<td>75.9</td>
<td>80.7</td>
<td>14.7</td>
<td>58.6</td>
<td>61.0</td>
<td>74.0</td>
</tr>
<tr>
<td></td>
<td>CONSAC(^\dagger) [26]</td>
<td>0.2 M</td>
<td>3.0</td>
<td>86.3</td>
<td>90.3</td>
<td>15.8</td>
<td>61.7</td>
<td>73.6</td>
<td>84.4</td>
</tr>
<tr>
<td></td>
<td>NeurVPS [69]</td>
<td>22 M</td>
<td>0.5</td>
<td>93.9</td>
<td>96.3</td>
<td>24.0</td>
<td>52.4</td>
<td>64.0</td>
<td>77.8</td>
</tr>
<tr>
<td>Ours</td>
<td>7 M</td>
<td>5.5</td>
<td>84.0</td>
<td>90.2</td>
<td>24.8</td>
<td>42.1</td>
<td>60.7</td>
<td>74.3</td>
<td>86.3</td>
</tr>
<tr>
<td>Ours(^\dagger)</td>
<td>5 M</td>
<td>23.0</td>
<td>84.8</td>
<td>90.7</td>
<td>22.9</td>
<td>39.8</td>
<td>59.5</td>
<td>72.6</td>
<td>85.4</td>
</tr>
<tr>
<td>Ours(^\dagger)</td>
<td>7 M</td>
<td>5.5</td>
<td>81.7</td>
<td>88.7</td>
<td>22.2</td>
<td>38.8</td>
<td>59.1</td>
<td>72.6</td>
<td>84.6</td>
</tr>
</tbody>
</table>

Figure 7. Exp 2: Manhattan world. AA curves on the ScanNet, SU3 and YUD datasets containing 3 orthogonal vanishing points. Learning-based approaches outperform methods relying purely on line segments and grouping, validating the power of representation learning. Our model shows comparable results to the best performing NeurVPS on ScanNet, while using \(3\times\) less parameters. On the smaller YUD dataset, our model slightly exceeds state-of-the-art. Generally, with ample data, our approach is comparable to others.

4.2. Exp 2: Validation on large datasets

We validate that adding prior knowledge does not deteriorate accuracy when there is plenty of data. We compare to five state-of-the-art baselines [17, 26, 34, 55, 69] on the ScanNet, SU3 and YUD datasets. On the ScanNet and SU3 datasets, we train all learning models from scratch on the full training split. On the YUD dataset, we use the pretrained models on SU3 without fine-tuning. For CONSAC and J-Linkage, we select top-3 predictions. We also measure the inference speed on a single RTX2080 GPU. Multi-scale sampling Ours\(^\dagger\) achieves 23 FPS, a large speedup over the vanilla design, as we utilize the orthogonality for efficient sampling.

Tab. 2 shows the AA scores on the ScanNet, SU3 and YUD datasets, while Fig. 7 depicts AA curves for varying angle differences. The SU3 dataset is easier as most images contain strong geometric cues (e.g. sharp edges and contours); this is no longer the case in the ScanNet dataset. The prediction error on the more realistic ScanNet dataset is significantly larger for all methods. On the ScanNet dataset, NeurVPS and our model are visibly better than methods relying on predefined line segments as inputs. The main advantage of NeurVPS and our model is their ability to learn useful feature representations directly from images. On the SU3 dataset, NeurVPS exceeds the other methods in the low-error region (from 0\(^\circ\) to 1\(^\circ\)). J-Linkage, Quasi-VP and CONSAC have similar results, and all of them stabilize at 1\(^\circ\). On SU3 our model is less accurate in 0\(^\circ\)-1\(^\circ\) range, yet it compensates at ≥ 1\(^\circ\). Our inferior performance in 0\(^\circ\)-1\(^\circ\) range results from the quantization errors in Hough Transform and the Gaussian sphere mapping. On the small-scale YUD dataset [14], our model achieves comparable accuracy without fine-tuning, and exceeds the other methods in the ≥ 2\(^\circ\) area, indicating the generalization ability of our model in the small data regime. We conclude that our model using prior knowledge performs similar to existing solutions.
4.3. Exp 3: Challenging scenarios

4.3.1 Exp 3(a): Reduced data

We evaluate data efficiency by reducing the number of training samples to \{10\%, 5\%, 2.5\%, 1\%\} on the ScanNet dataset, resulting in approximately 20K, 10K, 5K and 2K training images. Similarly, we also sample the SU3 dataset into \{50\%, 25\%, 10\%, 5\%, 2.5\%, 1\%\} subsets. We train all learning models from scratch using the default hyperparameters on each subset.

In Fig. 8 we compare the AA scores at 10\° with state-of-the-art methods. We use our vanilla design without the multi-scale sampling speedup. The first thing to notice is that non-learning methods are robust to data reduction. Yet, non-learning methods cannot take any training data into account, and thus they do not perform as well when more data is available, as we validated in the previous experiment. On the ScanNet dataset, our model visibly exceeds the other methods on the 10\%, 5\% and 2.5\% subsets. In comparison, NeurVPS suffers from large accuracy decreases on small training data subsets. When decreasing the number of samples to 2K (1\% subset), we still achieve competitive accuracy when compared to the non-learning methods, while NeurVPS fails to make reasonable predictions due to the lack of data. This shows the capability of our model to learn from limited data, thanks to the added geometric priors.

The NeurVPS model has \times 3 more parameters than our model due to its fully-connected layer with 16M parameters. For fairness, we also consider ‘NeurVPS-7M’ with reduced fully-connected layers, having a similar number of parameters with our model. Both NeurVPS variants perform similar on various subsets. On the SU3 dataset the accuracy of NeurVPS decreases significantly when reducing the training dataset size, despite its superiority on the large training subsets. In comparison, our model degrades gracefully when training data decreases from 20K to 1K. Notably, on the 1\% subset, with only 200 images for training, we are still able to achieve comparable performance with non-learning methods.
4.3.2  Exp 3.(b): Non-Manhattan scenario

We compare with state-of-art methods in a more realistic non-Manhattan scenario with limited annotated data. CONSAC [26] uses the line segment detection in [60]. We also consider a variant of CONSAC with the more recent line segment detector in [37].

Fig. 9 and Tab. 3 display the recall curve and the AUC values on the NYU Depth dataset, respectively. Our model consistently outperforms state-of-the-art baselines, and the improvement is more pronounced for larger angular differences. Although achieving the second-best result, VaPiD [38] assumes a constant number of instances and requires non-maximum suppression, which often results in over- and under-prediction. Our model outperforms existing methods by exploiting geometric priors, while not limiting the number of vanishing points detected.

4.4.  Exp 3.(c): Cross-dataset domain switch

We conduct cross-dataset test on multiple datasets, as displayed in Tab. 4. We compare with NeurVPS and CONSAC, which achieve top accuracy on individual datasets. When generalizing from synthetic dataset to the real-world (e.g. from SU3 to YUD), our model shows comparative results to CONSAC, which relies on prior line segment detection, making it robust to domain shifts. We observe a similar trend on real-world datasets (e.g. from NYU to YUD). However, on the challenging ScanNet dataset, Ours exceeds CONSAC, indicating the advantage of learning semantics over using pre-extracted lines. In contrast, NeurVPS does not transfer well to another dataset. This validates the robustness of the two priors in tackling domain shifts.

5. Conclusion and limitations

This paper focuses on vanishing point detection relying on well-founded geometric priors. We add two geometric priors as building blocks in deep neural networks for vanishing point detection: Hough Transform and Gaussian sphere mapping. We validate experimentally the added value of our geometric priors when compared to state-of-the-art Manhattan methods, and show their usefulness on realistic/challenging scenarios: with reduced samples, in the non-Manhattan world where the challenge is to predict a varying number of vanishing points without the orthogonality assumption, and across datasets.

Limitations. Despite of these improvements, our model also has several limitations. We pre-compute offline the mapping from images to the Hough bins and to the Gaussian sphere by fixing the size of the Hough histogram, as well as the Fibonacci sampling. However, these samplings introduce quantization errors which set an upper bound on accuracy. This is the primary reason for the limited accuracy on the SU3 dataset in the low-error region. A future research avenue is exploring an analytical mapping from image pixels to the Gaussian sphere. In addition, our model still relies on hundreds of fully labeled samples for training. One might consider testing the added geometric priors in an unsupervised or weakly-supervised setting.
References


[17] Chen Feng, Fei Deng, and Vineet R Kamat. Semi-automatic 3d reconstruction of piecewise planar building models from single image. CONVR (Sendai:), 2010. 2, 5, 6, 7


