Deep Vanishing Point Detection

Geometric priors make dataset variations vanish

Lin, Yancong; Wiersma, Ruben; Pintea, Silvia L.; Hildebrandt, Klaus; Eisemann, Elmar; van Gemert, Jan C.

DOI
10.1109/CVPR52688.2022.00601

Publication date
2022

Document Version
Final published version

Published in
Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)

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

2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) | 978-1-6654-6946-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/CVPR52688.2022.00601

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.

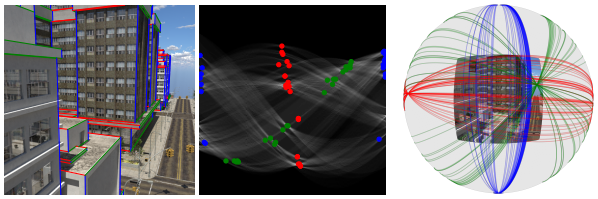


Figure 1. We add two geometric priors: (i) Hough Transform and (ii) Gaussian sphere mapping, for vanishing points detection. We transform learned image features to the Hough domain, where lines are mapped to individual bins. We further project the Hough bins to the Gaussian sphere, where lines become great circles and vanishing points are at the intersection of great circles. Each color represents a set of image lines related to a vanishing point. Adding geometric prior knowledge makes our model data-efficient, less dependent on domain-specifics, and easily adaptable to problem variations such as detecting a variable number of vanishing points.

We add two geometric priors, see Fig. 1: (i) the 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.

978-1-6654-6946-3/22/$31.00 ©2022 IEEE
DOI 10.1109/CVPR52688.2022.00601

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 ρ and discrete angles θ, defining a 2D discrete histogram. In practice, a set of pixels $(x(i), y(i))$ along a line indexed by i, vote for a line parameterization to

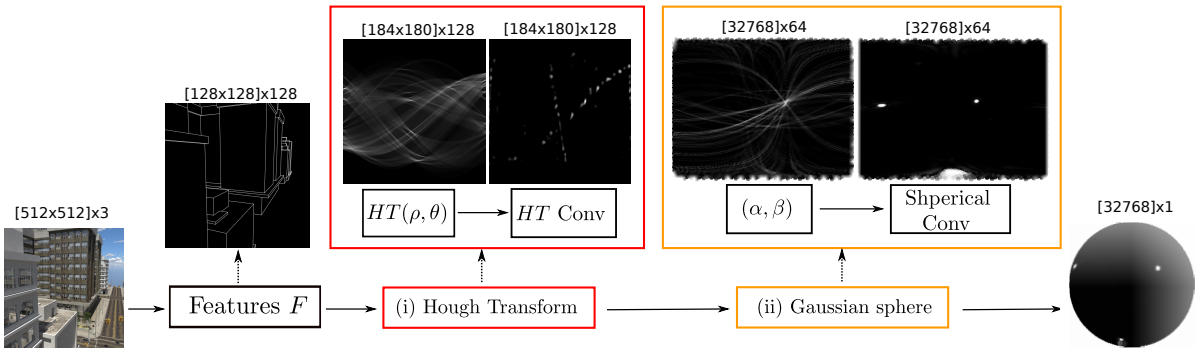


Figure 2. **Overview:** The model starts from in input image, and predicts vanishing points on the Gaussian hemisphere by relying on two geometric priors: (i) Hough Transform, and (ii) Gaussian sphere mapping. We use a convolutional network to learn features which are then mapped to Hough space, where each bin is a line. We filter the Hough space and project Hough bins to the Gaussian hemisphere and apply spherical convolutions to find vanishing points. We indicate the size of the learned features above, where the last dimension is the number of channels. We sample 32,768 points on the hemisphere using the Fibonacci lattice [20], resulting in features maps of size 32,768. Our model learns to classify spherical points as vanishing points or not using a binary cross-entropy loss. There is *no* intermediate supervision.

which they all belong:

$$HT(\rho,\theta) = \sum_i F(\rho\cos\theta - i\sin\theta, \rho\sin\theta + i\cos\theta) \quad (1)$$

The Hough Transform module starts from an $[H\times W]$ feature map F and outputs an $[N_\rho\times N_\theta]$ Hough histogram HT, where N_ρ and N_θ are the number of sampled offsets and angles in Hough Transform. We set H=128, W=128, N_ρ=184, and N_θ=180. This results in $[N_\rho\times N_\theta]$ possible line parameterizations. We find the local maxima in the Hough domain by performing a 1D convolutions 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 δ.

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

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

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

The spherical coordinates (α,β) describe a point on the Gaussian sphere, where α is the azimuth defined as the angle from the z-axis in the xz plane, and β 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 ψ and the Gaussian sphere is a great circle. This great circle represents the projection of the image line direction $\overrightarrow{AB}$ on the Gaussian sphere. Intersections of multiple great circles are potential vanishing points, see Fig. 3(b).

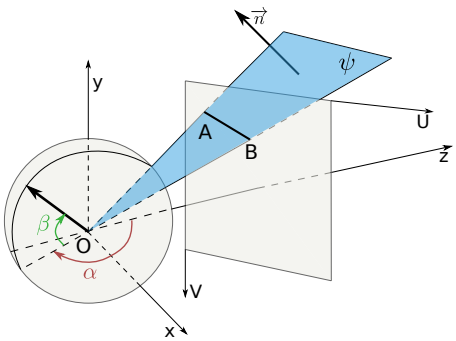
We compute the projection of the image line direction $\overrightarrow{AB}$, by estimating the elevation β as a function of the azimuth α and the normal vector $\overrightarrow{n}$ [51]:

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

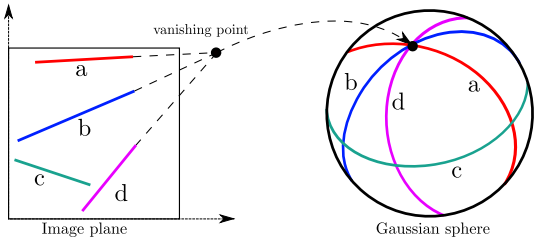
where we uniformly sample α in the range $[-\pi/2,\pi)$.

Because the Gaussian sphere is symmetric we only need a hemisphere. We sample N points on the Gaussian hemisphere using a 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 (ρ,θ), we first compute its normal vector $\overrightarrow{n}$. We, then, estimate its corresponding (α,β) spherical coordinates using Eq. (3). We subsequently assign each (α,β) 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 $[N_\rho\times N_\theta\times M]$ tensor, where $[N_\rho\times N_\theta]$ is the number of line parameterizations in Hough space and M is the number of sampled azimuth angles α. 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 [62] to convolve over the hemisphere. EdgeConv operates on a k-nearest neighbor graph on the



(a) The Gaussian sphere



(b) Vanishing points on the Gaussian sphere

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.

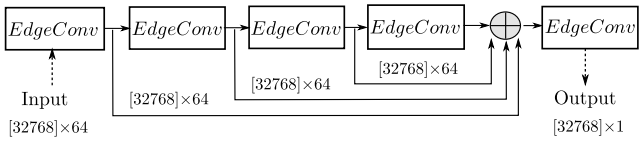


Figure 4. **Spherical convolutions on the hemisphere.** We use EdgeConv [62] for precise vanishing point localization on the Gaussian sphere. The concatenation of the previous feature maps is fed into the final layer to produce a prediction.

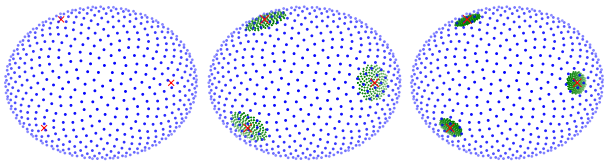


Figure 5. **Multi-scale sampling on the hemisphere.** We sample points at three scales for detecting vanishing points in the Manhattan world, as in [69]. Blue indicates sampling at the first scale, while green indicates fine-grained sampling at the following scales. The red crosses are the predictions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

points. It learns to represent local neighborhoods by applying a non-linear function to the neighbors' features, and then aggregates those features with a symmetric operator. The neighbors' features are localized by subtracting the features of the centroid. Like [62], we take the per-feature maximum over the neighbors to aggregate edge features.

As shown in Fig. 4, the spherical part of our model contains 5 EdgeConv modules [62]. Each EdgeConv module transforms neighboring features with a fully connected layer, a BatchNorm layer [22] and a LeakyReLU activation. We use N=32,768 nodes on the hemisphere and compute the 16 nearest neighbors for each node. We concatenate the features maps from previous layers and feed them into the last EdgeConv layer to produce the final prediction.

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 *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 δ 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,

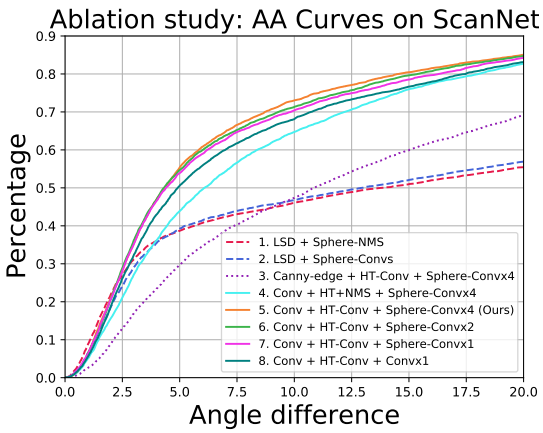


Figure 6. **Exp 1: Model choices.** We show the effects of the two geometric priors quantitatively on the ScanNet-1% subset. Adding HT layers and spherical convolutions outperforms the baselines, thus demonstrating the effectiveness of geometric priors.

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" VPs 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°.

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.

	HT (# of angles)		Sphere (# of points)		
	90	180	$8K$	$16K$	$32K$
AA@3°	77.1	79.3	73.3	77.4	79.3

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.

Implementation details. We implement our model in Pytorch [48], and provide the code online [1]. 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 4×10^{-4} and 1×10^{-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 datset 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.

[1] https://github.com/yanconglin/VanishingPoint_HoughTransform_GaussianSphere

Datasets			SU3 [70]		ScanNet [12]			YUD [14]		
Metrics	Params	FPS	AA@3°	AA@5°	AA@3°	AA@5°	AA@10°	AA@3°	AA@5°	AA@10°
J-Linkage† [17]	—	1.0	82.0	87.2	15.7	27.3	43.0	60.8	71.8	81.5
Contrario-VP† [55]	—	0.6	64.8	72.2	12.0	21.4	35.3	58.6	70.7	81.8
Quasi-VP [34]	—	**29.0**	75.9	80.7	14.7	25.3	39.4	58.6	61.0	74.0
CONSAC† [26]	0.2 M	3.0	86.3	90.3	15.8	24.6	36.0	**61.7**	73.6	84.4
NeurVPS [69]	22 M	0.5	**93.9**	**96.3**	24.0	41.8	**64.4**	52.4	64.0	77.8
Ours	7 M	5.5	84.0	90.2	**24.8**	**42.1**	63.7	60.7	**74.3**	**86.3**
*Ours**	5 M	23.0	84.8	90.7	22.9	39.8	62.4	59.5	72.6	85.4
Ours†	7 M	5.5	81.7	88.7	22.2	38.8	59.9	59.1	72.6	84.6

Table 2. **Exp 2: Manhattan world.** Angular accuracy on SU3, ScanNet and YUD datasets. *Ours* achieves the best results on the the YUD dataset, and is competitive on the larger ScanNet and SU3 datasets. *Ours** adopts the multi-scale sampling strategy, thus being significantly faster. † assumes unknown focal length, thus making the Manhattan assumption no longer applicable. *Ours*† shows a constant decrease 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.

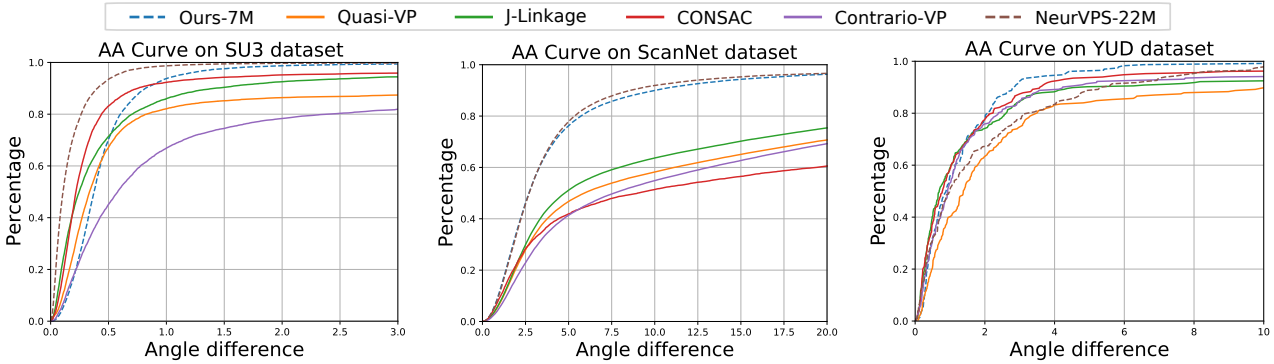
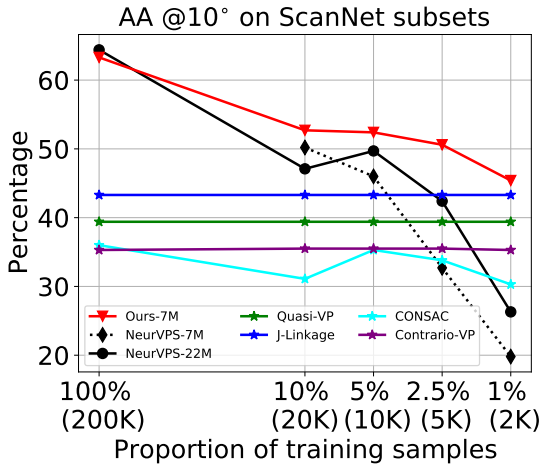


Figure 7. **Exp 2: Manhattan world.** AA curves on on the ScanNet, SU3 and YUD datasets containing 3 orthogonal vanishing points. Learning-based approaches outperform methods relying purely on line segments and grouping, validating the power of representation learning. Our model shows comparable results to the best performing NeurVPS on ScanNet, while using 3× 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.

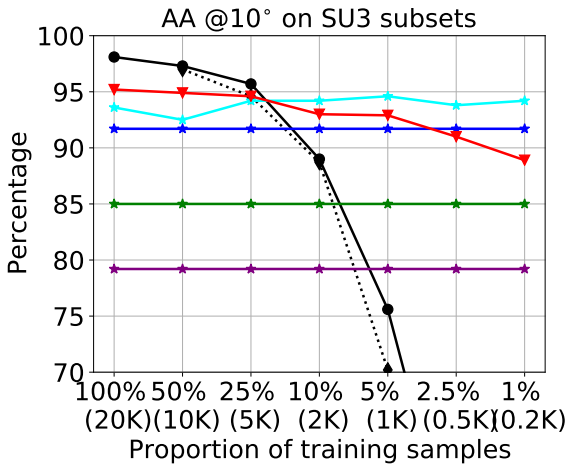
4.2. Exp 2: Validation on large datasets

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 pre-trained 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** 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 $0°$ to $1°$). J-Linkage, Quasi-VP and CONSAC have similar results, and all of them stabilize at $1°$. On SU3 our model is less accurate in $0°$-$1°$, yet it compensates at $\geq 1°$. Our inferior performance in $0°$-$1°$ 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 $\geq 2°$ 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.



(a) Data efficiency on ScanNet



(b) Data efficiency on SU3

Figure 8. **Exp 3.(a): Reduced data.** We report AA @10° on various subsets of the ScanNet and SU3 datasets, and indicate the number of parameters in the legend. On the ScanNet dataset, we outperform other methods on the 10%, 5% and 2.5% subsets. Our model degrades gracefully when reducing the training samples from 20K to 1K on the SU3 subset, while NeurVPS has a drastic drop in accuracy. CONSAC achieves top results on SU3 due to pre-extracted line segments, but fails on ScanNet because of inaccurate line detection. There is a similar trend for the baselines relying on line segment detection. Our model predictions are stable having small variances (± 0.50 and ± 0.43 on the 1% subsets of ScanNet and SU3 respectively) across 3 repetitions. This experiment validates the data efficiency of our model.

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.

Datasets	NYU Depth [45]			
	top-k = #gt		top-k = #pred	
AUC	@5°	@10°	@5°	@10°
J-Linkage [17]	49.30	61.28	54.48	68.34
T-Linkage [42]	43.38	58.05	47.48	64.59
CONSAC [26]	49.46	65.00	54.37	69.89
CONSAC [26]+ DLSD [37]	46.78	61.06	49.94	65.96
VaPiD [38]	-	69.10	-	-
Ours	**55.92**	**69.57**	**57.19**	**71.62**

Table 3. **Exp 3(b): Non-Manhattan scenario.** We report AUC scores on the NYU Depth dataset. Here "top-k = #gt" indicates the k most confident predictions where k is the number of annotated instances [26], while for "top-k =#pred" all predictions are used for evaluation. Our model exceeds state-of-the-art when detecting a varying number of vanishing points.

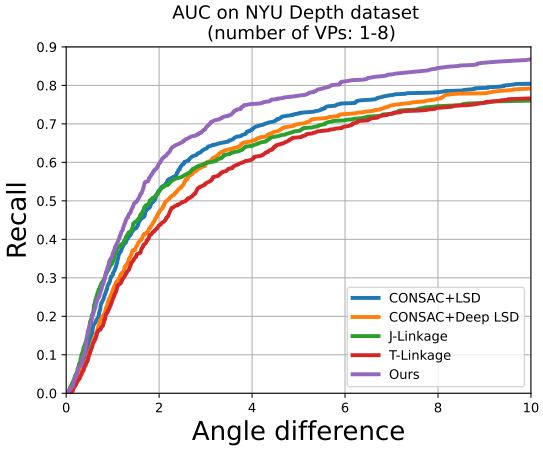


Figure 9. **Exp 3.(b): Non-Manhattan scenario.** We plot the recall curve on the non-Manhattan NYU dataset. Our method outperforms state-of-the-art, illustrating the ability of the model to detect a varying number of vanishing points. See supplementary material for qualitative visualizations.

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.

Synthetic - real-world data						
Train	SU3 [70]					
Test	ScanNet [12]			YUD [14]		
Models	*Ours*	NeurVPS	CONSAC	*Ours*	NeurVPS	CONSAC
AA@3°	**15.2**	11.1	10.1	60.7	53.8	**61.7**
AA@5°	**25.9**	20.3	17.3	**74.3**	65.6	73.6
AA@10°	**39.5**	35.5	27.2	**86.3**	79.7	84.4
Real-world data						
Train	NYU Depth [45]					
Test	ScanNet [12] (AA)		YUD [14] (AA)		YUD+ [14] (AUC)	
Models	*Ours*	CONSAC	*Ours*	CONSAC	*Ours*	CONSAC
@10°	**33.6**	30.3	**83.2**	82.7	71.4	**75.0**

Table 4. **Exp 3.(c) Cross-dataset domain switch.** "Train" and "Test" specify the training and test datasets. CONSAC uses pre-extracted lines, thus being accurate on YUD/YUD+. However, its accuracy is lower on ScanNet due to the lack of reliable lines. In comparison, *Ours* is more accurate on both ScanNet and YUD without tuning. Our geometric priors improve the transferability of the model across datasets.

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

[1] Matthew E Antone and Seth Teller. Automatic recovery of relative camera rotations for urban scenes. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, volume 2, pages 282–289. IEEE, 2000. 1, 2

[2] Michel Antunes and Joao P Barreto. A global approach for the detection of vanishing points and mutually orthogonal vanishing directions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1336–1343, 2013. 2

[3] Olga Barinova, Victor Lempitsky, Elena Tretiak, and Pushmeet Kohli. Geometric image parsing in man-made environments. In *European conference on computer vision*, pages 57–70. Springer, 2010. 2

[4] Stephen T Barnard. Interpreting perspective images. *Artificial intelligence*, 21(4):435–462, 1983. 1, 2, 4

[5] Jean-Charles Bazin, Yongduek Seo, Cédric Demonceaux, Pascal Vasseur, Katsushi Ikeuchi, Inso Kweon, and Marc Pollefeys. Globally optimal line clustering and vanishing point estimation in manhattan world. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 638–645. IEEE, 2012. 2

[6] Jean-Charles Bazin, Yongduek Seo, and Marc Pollefeys. Globally optimal consensus set maximization through rotation search. In *Asian Conference on Computer Vision*, pages 539–551. Springer, 2012. 2

[7] Ali Borji. Vanishing point detection with convolutional neural networks. *CVPR Scene Understanding Workshop*, 2016. 1, 2

[8] Chin-Kai Chang, Jiaping Zhao, and Laurent Itti. Deepvp: Deep learning for vanishing point detection on 1 million street view images. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8, 2018. 1, 2

[9] Roberto Cipolla, Tom Drummond, and Duncan P Robertson. Camera calibration from vanishing points in image of architectural scenes. In *BMVC*, volume 99, pages 382–391, 1999. 1

[10] Robert T Collins and Richard S Weiss. Vanishing point calculation as a statistical inference on the unit sphere. In *ICCV*, volume 90, pages 400–403, 1990. 2

[11] David F Crouse. On implementing 2d rectangular assignment algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 52(4):1679–1696, 2016. 5

[12] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5828–5839, 2017. 2, 4, 6, 8

[13] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1052–1067, 2007. 1

[14] Patrick Denis, James H Elder, and Francisco J Estrada. Efficient edge-based methods for estimating manhattan frames in urban imagery. In *European conference on computer vision*, pages 197–210. Springer, 2008. 2, 4, 6, 8

[15] Richard O Duda and Peter E Hart. Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, 1972. 1

[16] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996. 4

[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

[18] Alex Flint, David Murray, and Ian Reid. Manhattan scene understanding using monocular, stereo, and 3d features. In *2011 International Conference on Computer Vision*, pages 2228–2235. IEEE, 2011. 1

[19] Paolo Gamba, Alessandro Mecocci, and U Salvatore. Vanishing point detection by a voting scheme. In *Proceedings of 3rd IEEE International Conference on Image Processing*, volume 2, pages 301–304, 1996. 2

[20] Álvaro González. Measurement of areas on a sphere using fibonacci and latitude–longitude lattices. *Mathematical Geosciences*, 42(1):49, 2010. 3

[21] Lazaros Grammatikopoulos, George Karras, and Elli Petsa. An automatic approach for camera calibration from vanishing points. *ISPRS journal of photogrammetry and remote sensing*, 62(1):64–76, 2007. 1

[22] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015. 4

[23] Kyungdon Joo, Tae-Hyun Oh, Junsik Kim, and In So Kweon. Robust and globally optimal manhattan frame estimation in near real time. *IEEE transactions on pattern analysis and machine intelligence*, 41(3):682–696, 2018. 2

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[25] Florian Kluger, Hanno Ackermann, Michael Ying Yang, and Bodo Rosenhahn. Deep learning for vanishing point detection using an inverse gnomonic projection. In *German Conference on Pattern Recognition*, pages 17–28. Springer, 2017. 2

[26] Florian Kluger, Eric Brachmann, Hanno Ackermann, Carsten Rother, Michael Ying Yang, and Bodo Rosenhahn. Consac: Robust multi-model fitting by conditional sample consensus. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4634–4643, 2020. 2, 5, 6, 7, 8

[27] Jana Košecká and Wei Zhang. Video compass. In *European conference on computer vision*, pages 476–490, 2002. 2

[28] Seokju Lee, Junsik Kim, Jae Shin Yoon, Seunghak Shin, Oleksandr Bailo, Namil Kim, Tae-Hee Lee, Hyun Seok Hong, Seung-Hoon Han, and In So Kweon. Vpgnet: Vanishing point guided network for lane and road marking detection and recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 1947–1955, 2017. 1

[29] Attila Lengyel, Sourav Garg, Michael Milford, and Jan C van Gemert. Zero-shot day-night domain adaptation with a physics prior. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4399–4409, 2021. 2

[30] José Lezama, Rafael Grompone von Gioi, Gregory Randall, and Jean-Michel Morel. Finding vanishing points via point alignments in image primal and dual domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 509–515, 2014. 2

[31] Haoang Li, Kai Chen, Pyojin Kim, Kuk-Jin Yoon, Zhe Liu, Kyungdon Joo, and Yun-Hui Liu. Learning icosahedral spherical probability map based on bingham mixture model for vanishing point estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5661–5670, 2021. 2, 5

[32] Haoang Li, Pyojin Kim, Ji Zhao, Kyungdon Joo, Zhipeng Cai, Zhe Liu, and Yun-Hui Liu. Globally optimal and efficient vanishing point estimation in atlanta world. In *European Conference on Computer Vision*, pages 153–169. Springer, 2020. 2

[33] Haoang Li, Yazhou Xing, Ji Zhao, Jean-Charles Bazin, Zhe Liu, and Yun-Hui Liu. Leveraging structural regularity of atlanta world for monocular slam. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2412–2418. IEEE, 2019. 1

[34] Haoang Li, Ji Zhao, Jean-Charles Bazin, Wen Chen, Zhe Liu, and Yun-Hui Liu. Quasi-globally optimal and efficient vanishing point estimation in manhattan world. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1646–1654, 2019. 5, 6

[35] Haoang Li, Ji Zhao, Jean-Charles Bazin, and Yun-Hui Liu. Quasi-globally optimal and near/true real-time vanishing point estimation in manhattan world. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 2, 5

[36] Yunsheng Li and Nuno Vasconcelos. Efficient multi-domain learning by covariance normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5424–5433, 2019. 2

[37] Yancong Lin, Silvia L Pintea, and Jan C van Gemert. Deep hough-transform line priors. 2020. 2, 3, 7, 8

[38] Shichen Liu, Yichao Zhou, and Yajie Zhao. Vapid: A rapid vanishing point detector via learned optimizers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12859–12868, 2021. 2, 5, 7, 8

[39] Yawei Luo, Liang Zheng, Tao Guan, Junqing Yu, and Yi Yang. Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2507–2516, 2019. 2

[40] Evelyne Lutton, Henri Maitre, and Jaime Lopez-Krahe. Contribution to the determination of vanishing points using hough transform. *IEEE transactions on pattern analysis and machine intelligence*, 16(4):430–438, 1994. 2

[41] Michael J Magee and Jake K Aggarwal. Determining vanishing points from perspective images. *Computer Vision, Graphics, and Image Processing*, 26(2):256–267, 1984. 2

[42] Luca Magri and Andrea Fusiello. T-linkage: A continuous relaxation of j-linkage for multi-model fitting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3954–3961, 2014. 5, 7

[43] GF McLean and D Kotturi. Vanishing point detection by line clustering. *IEEE Transactions on pattern analysis and machine intelligence*, 17(11):1090–1095, 1995. 2

[44] Faraz M Mirzaei and Stergios I Roumeliotis. Optimal estimation of vanishing points in a manhattan world. In *2011 International Conference on Computer Vision*, pages 2454–2461, 2011. 2

[45] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 2, 4, 7, 8

[46] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499. Springer, 2016. 2

[47] Phil L Palmer and A Tai. An optimised vanishing point detector. In *BMVC*, pages 1–10, 1993. 2

[48] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 5

[49] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011. 4

[50] Kuan-Chuan Peng, Ziyan Wu, and Jan Ernst. Zero-shot deep domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 764–781, 2018. 2

[51] Long Quan and Roger Mohr. Determining perspective structures using hierarchical hough transform. *Pattern Recognition Letters*, 9(4):279–286, 1989. 2, 3, 4

[52] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Efficient parametrization of multi-domain deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8119–8127, 2018. 2

[53] Frederik Schaffalitzky and Andrew Zisserman. Planar grouping for automatic detection of vanishing lines and points. *Image and Vision Computing*, 18(9):647–658, 2000. 2

[54] Grant Schindler and Frank Dellaert. Atlanta world: An expectation maximization framework for simultaneous low-level edge grouping and camera calibration in complex man-made environments. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 1, pages I–I, 2004. 2

[55] Gilles Simon, Antoine Fond, and Marie-Odile Berger. A-contrario horizon-first vanishing point detection using second-order grouping laws. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 318–333, 2018. 2, 5, 6

[56] Marco Straforini, C Coelho, and Marco Campani. Extraction of vanishing points from images of indoor and outdoor scenes. *Image and Vision Computing*, 11(2):91–99, 1993. 2

[57] A Tai, Josef Kittler, Maria Petrou, and Terry Windeatt. Vanishing point detection. In *BMVC92*, pages 109–118. Springer, 1992. 2

[58] Jean-Philippe Tardif. Non-iterative approach for fast and accurate vanishing point detection. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1250–1257. IEEE, 2009. 2

[59] Roberto Toldo and Andrea Fusiello. Robust multiple structures estimation with j-linkage. In *European conference on computer vision*, pages 537–547. Springer, 2008. 2

[60] Rafael Grompone Von Gioi, Jeremie Jakubowicz, Jean-Michel Morel, and Gregory Randall. Lsd: A fast line segment detector with a false detection control. *IEEE transactions on pattern analysis and machine intelligence*, 32(4):722–732, 2008. 5, 8

[61] Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018. 2

[62] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. 3, 4

[63] Horst Wildenauer and Allan Hanbury. Robust camera self-calibration from monocular images of manhattan worlds. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2831–2838, 2012. 2

[64] Jianping Wu, Liang Zhang, Ye Liu, and Ke Chen. Real-time vanishing point detector integrating under-parameterized ransac and hough transform. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3732–3741, 2021. 2, 5

[65] Markus Wulfmeier, Alex Bewley, and Ingmar Posner. Addressing appearance change in outdoor robotics with adversarial domain adaptation. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1551–1558. IEEE, 2017. 2

[66] Menghua Zhai, Scott Workman, and Nathan Jacobs. Detecting vanishing points using global image context in a non-manhattan world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5657–5665, 2016. 2

[67] Xiaodan Zhang, Xinbo Gao, Wen Lu, Lihuo He, and Qi Liu. Dominant vanishing point detection in the wild with application in composition analysis. *Neurocomputing*, 311:260–269, 2018. 1, 2

[68] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization: A survey. *arXiv preprint arXiv:2103.02503*, 2021. 2

[69] Yichao Zhou, Haozhi Qi, Jingwei Huang, and Yi Ma. Neurvps: Neural vanishing point scanning via conic convolution. In *Advances in Neural Information Processing Systems*, pages 866–875, 2019. 1, 2, 4, 5, 6

[70] Yichao Zhou, Haozhi Qi, Yuexiang Zhai, Qi Sun, Zhili Chen, Li-Yi Wei, and Yi Ma. Learning to reconstruct 3d manhattan wireframes from a single image. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7698–7707, 2019. 2, 4, 6, 8

[71] Zihan Zhou, Farshid Farhat, and James Z Wang. Detecting dominant vanishing points in natural scenes with application to composition-sensitive image retrieval. *IEEE Transactions on Multimedia*, 19(12):2651–2665, 2017. 2