Finding Vanishing Points via Point Alignments in Image Primal and Dual Domains

José Lezama[*†], Rafael Grompone von Gioi[*], Gregory Randall[†], Jean-Michel Morel[*]
[*]CMLA, ENS Cachan, France, [†]IIE, Universidad de la República, Uruguay
{lezama, grompone, morel}@cmla.ens-cachan.fr, randall@fing.edu.uy

Abstract

We present a novel method for automatic vanishing point detection based on primal and dual point alignment detection. The very same point alignment detection algorithm is used twice: First in the image domain to group line segment endpoints into more precise lines. Second, it is used in the dual domain where converging lines become aligned points. The use of the recently introduced PClines dual spaces and a robust point alignment detector leads to a very accurate algorithm. Experimental results on two public standard datasets show that our method significantly advances the state-of-the-art in the Manhattan world scenario, while producing state-of-the-art performances in non-Manhattan scenes.

1. Introduction

Under the pinhole camera model, 3D lines are transformed into 2D lines. Moreover, parallel lines in 3D are projected into lines that converge on a point (perhaps at infinity) known as a *vanishing point* (VP). In the presence of parallel lines, as is common in human-made environments, VPs provide crucial information about the 3D structure of the scene and have applications in camera calibration, single-view 3D scene reconstruction, autonomous navigation, and semantic scene parsing, to mention a few.

There is a vast literature on the problem of VP detection, starting with the seminal work by Barnard [5] and leading to high-precision algorithms in recent works [20, 21]. Typically, a method starts by the identification of oriented elements, which are then clustered into groups of concurrent directions, and finally refined to get the corresponding VP. Most proposed methods use image line segments as oriented elements [5, 7, 1, 21], but oriented edge points are also used [8, 9, 4], or even the alignment of similar structures [19]. The clustering is often performed in the image plane, by identifying points or zones of concurrence of the elements [16, 1]; other methods, however, rely on the Gaussian

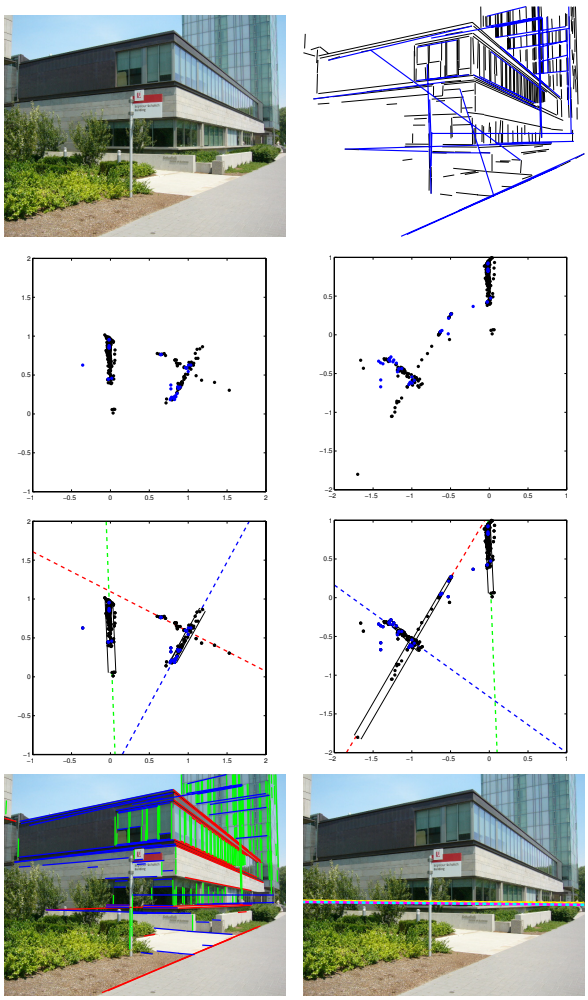


Figure 1. Main steps of our method. **Top-Left:** Input image. **Top-Right:** Line segments (black) and alignments of line segments endpoints (blue). **2nd Row:** Lines as points in *straight* (left) and *twisted* (right) PClines spaces. **3rd Row:** Aligned points detections (parallel black lines) and the ground truth (dashed lines). Some alignments are more visible in one of the spaces than in the other. **Bottom-Left:** Final VP associations by enforcing orthogonality. **Bottom-Right:** Horizon line estimation (yellow-orange: ground truth, cyan-magenta: ours).

sphere of world directions [5, 7], which requires camera calibration information. Less common is the use of a dual space in which points become lines and converging lines become aligned points [3, 22]. Various validation methods are used, from simple thresholds to statistical methods [7, 1] or Bayesian inference [8, 9]. Early methods used the Hough transform in the Gaussian sphere [5, 15]. More recent methods rely on variations of RANSAC [14, 20] or EM [9], which was successfully used to solve the uncalibrated case in [12]. Some algorithms start with a clustering step that is obtained non-iteratively, and perform iterations to improve the result [18, 21]. To simplify the often ill-posed problem, some methods make assumptions about the scene contents. The most common one is the so-called "Manhattan world", implying the existence of only three orthogonal VPs [8], which is sometimes inherently enforced during clustering [14, 6, 20]. When valid, this assumption helps improving the results. Barinova *et al*. [4], however, claim that a better balance between generality and robustness is provided by a relaxed assumption where one vertical VP and multiple horizontal ones (not necessarily orthogonal) are considered [17, 4]. Some methods make no assumption at all [1]. Recent works made progress by defining various consistency measures between vanishing points and line segments [20, 2, 21], while [4] performs a joint optimization of line, VP and camera parameters.

In this paper we build on the advances of various previous works to obtain a novel and more accurate VP detection algorithm. The oriented elements are the line segments detected with the LSD algorithm [11]. The clustering step is done in the dual space [22], but takes advantage of the PClines point-to-line mappings described recently by Dubská *et al*. [10]. An unsupervised point alignment detector [13] is used to compute sets of collinear points, which correspond to converging lines and thus to VPs. The very same point alignment detector is used to group aligned line segments into longer and more precise ones. After the clustering is done and the candidate vanishing points are obtained, the measure of statistical significance [13] of the alignments is used to find the final triplet of VPs, when the Manhattan world assumption is applicable, or the horizon line when it is not. The method is deterministic and non-iterative and has an accuracy comparable or better than state-of-the-art algorithms.

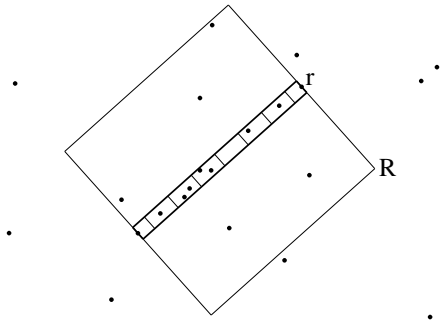


Figure 2. Representation of the candidate rectangle from [13]. In this case, the candidate rectangle r is divided into $c = 6$ boxes, 5 of which are occupied. The local density estimation window is R.

2. Algorithm

Our algorithm works in six steps:

1. Detect line segments in the image.
2. Find and join aligned line segments.
3. Compute the correspondence of line segments in the PClines *straight* and *twisted* dual spaces.
4. Detect point alignments in both dual spaces.
5. Refine the obtained VPs.
6. Estimate horizon line based on one of two hypothesis: Manhattan or non-Manhattan world.

These steps are detailed in the rest of this section. Sect. 3 analyses the experimental results.

2.1. Point alignment detector

The proposed method uses the point alignment detector introduced in [13]. Given a set of 2D points, this detector defines point alignments as rectangular clusters. Candidate rectangles are obtained by considering each possible pair of points and a set of possible widths. Then, each candidate rectangle is divided into boxes and the boxes occupied by at least one point are counted. Based on the *a contrario* methodology, the idea behind the algorithm is to measure the expected number of occurrences of such an event under a null hypothesis H_0 of independent and uniformly distributed random points. When this expectation is small, the event is termed non accidental and detected.

Let $\mathbf{x}$ be a set of N points. Let r be a candidate rectangle divided into c equal boxes, and R a rectangle surrounding r, used for local point density estimation, see Fig. 2. If $b(r, c, \mathbf{x})$ is the observed number of occupied boxes, the expectation of occurrences of such an event under H_0 is approximated by the associated Number of False Alarms (NFA) value [13],

$$\begin{aligned}\mathrm{NFA}(r, R, c, \mathbf{x}) &= \\ &= \frac{N(N-1)}{2} WLC \cdot \mathcal{B}\big(c, b(r, c, \mathbf{x}), p(R, c)\big), \quad (1)\end{aligned}$$

where W, L and C are the number of different rectangle widths, local windows widths and number of boxes tested

for each rectangle r, $\mathcal{B}$ is the tail of the binomial distribution, and $p(R, c)$ is the probability for a box of being occupied taking into account the point density in the local window R. When the NFA of an observed configuration is large, this means that such an event was to be expected under the *a contrario* model and therefore it is irrelevant. On the other hand, when the NFA is small, the event is rare and probably meaningful. A threshold ε is fixed for the NFA. Rectangles with $\mathrm{NFA}(r, R, c, \mathbf{x}) \leq \varepsilon$ are considered ε-meaningful and constitute the detection result of the algorithm (previous to a redundancy reduction step). In [13] it is proved that this threshold effectively bounds the expected number of times that such an event would occur under the *a contrario* hypothesis H_0.

Once all the ε-meaningful alignments are obtained, the algorithm applies a *masking principle* to resolve the redundancy of detections. One detection B is said to be "masked" by another detection A if, when the elements (points) of A are taken out from B, B is no longer ε-meaningful. In this step, the method keeps the most meaningful detections in terms of the NFA, and discards those that are "masked" by them. Fig. 1(e) & (f), 3 (center) and 5 (4^{th} row) show alignments detected by this algorithm (parallel black lines).

2.2. Segment endpoint alignments

Similarly to [19], our method exploits the alignment of features that a line segment detector alone does not capture. To achieve this, the point alignment detector of Sect. 2.1 is used to find alignments among the endpoints of the segments detected by LSD [11]. This step increases the accuracy of short segments by grouping them and brings an improvement to the algorithm performance[1].

First, line segments are grouped by length and orientation. A single threshold τ is used on the length, and angular steps of 30 degrees are used to group by orientation. The objective is to connect segments that share the same orientation – e.g. the borders of the windows of a building – or endpoints of parallel segments – e.g. an array of vertical structures. In each subset, the point alignment detector is run over the segment endpoints. This produces a new set of line segments. At the end of this stage, short (and therefore inaccurate) line segments are discarded. The final list of segments is composed of the long segments and the segments from the alignment of endpoints found among both the short and long ones. Fig. 3 shows some example detections.

2.3. PClines parameterization

The problem of finding converging lines can be cast in a point alignment detection problem by parameterizing the lines as points in a suitable dual space. Our method uses the PClines parameterization [10], see Fig. 4. The *straight*

[1] On average 2% with the metric used in Sect. 3

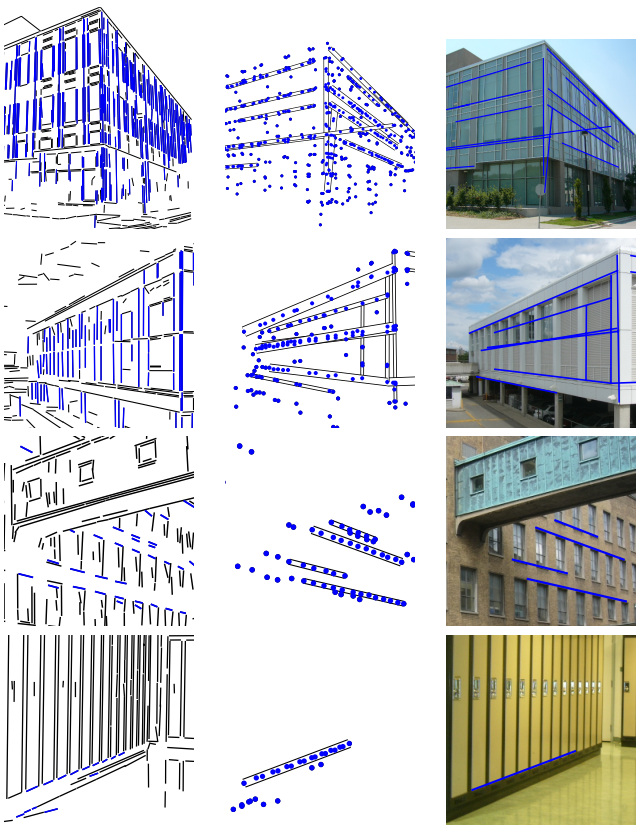


Figure 3. Examples of detected alignments of line segment endpoints. **Left**: original line segments, highlighting in blue one group of length and orientation. **Center**: line segment endpoints and the detected alignments for that group (parallel black lines). **Right**: additional oriented features (blue lines). In the top two rows, parallel line segment endpoints are connected to recover structural directions that were not captured by the line segment detector alone. In the bottom rows, short, inaccurate segments are joined into larger, more accurate ones.

version uses a parallel coordinate system, where the horizontal x axis in the image is represented as the vertical v axis in the dual space, and the vertical y axis in the image is represented as a vertical line passing through $u = d$ in the dual space. Thus, a point $A = (A_x, A_y)$ in the image is represented in the dual space by a line passing by $(0, A_x)$ and (d, A_y). A line joining multiple points in the image is represented in the dual space as a point that lies at the intersection of the lines representing those points. Note that points in dual space can be arbitrarily far away from the origin [10]. To overcome this unboundedness problem, the *twisted* version of the PClines transform is also used, where the x axis stays in the ordinates axis but the $-y$ axis is transformed into a vertical line at $u = -d$. Fig. 1 (c) & (d) show example results of the transformation for real data.

By using the *straight* and *twisted* transforms, it is guaranteed that all directions in the image are represented as

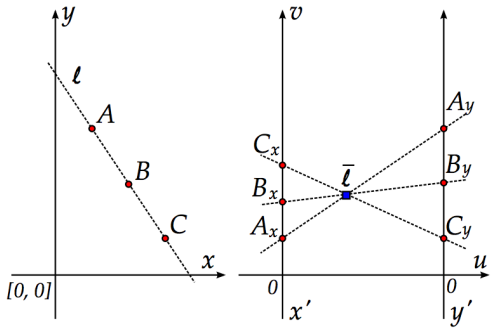


Figure 4. Schematic of the *straight* PClines transform. **Left**: three points and a line in the image $(x,\ y)$ space. **Right**: their representation in the PClines $(u,\ v)$ *straight* dual space. This figure was taken from [10].

a bounded set of points in at least one of the two spaces. The value of d is set to 1 and the limits of the dual domains where the point alignment detector will be executed are set to $[-1,\ 2] \times [-1,\ 2]$ for the *straight* transform and $[-2,\ 1] \times [-2,\ 1]$ for the *twisted* transform. Points that fall outside each domain are discarded. Once the alignment detector is run in each dual space, each alignment found determines a candidate vanishing point $\mathbf{v}_i$ in the image, with an associated meaningfulness value NFA_i (Sect. 2.1). VPs that were detected in both spaces are identified with a simple distance threshold δ and only the one with the best NFA value is kept.

2.4. Refinement

Each rectangle candidate of the point alignment detector (Sect. 2.1) is defined by a pair of points. Thus, the direction it defines represents in the image the intersection of two lines, which as a candidate VP, needs to be refined. This is done by using a consistency measure existing in the literature [16, 9] – the more elegant version of [21] can also be used. Given a line segment $\mathbf{l}$ and a VP candidate $\mathbf{v}_i$, the consistency measure is the angle between the direction of $\mathbf{l}$ and the ideal line passing through $\mathbf{v}_i$ and the centroid of $\mathbf{l}$. The line segments consistent with $\mathbf{v}_i$ are obtained by setting a threshold θ on this angle. Finally, we use the function for updating the VP estimate of [2], which minimizes the weighted sum of the square of perpendicular distances from the VPs to the lines defined by the line segments. The weights are given by the segments lengths. For clusters of parallel line segments, this problem is undetermined, and the refinement is not performed. Due to the quality of the obtained clusters, a single refinement iteration is enough.

2.5. Solving redundancy

Once the refined candidate VPs are obtained, relevant detections must be discriminated from spurious ones. Two possible hypothesis can be introduced: a) Manhattan world images with a calibrated camera. b) Non-Manhattan world images (multiple horizontal VPs) without a calibrated camera. We shall treat each scenario differently. Note however that the previous stages of the algorithm are the same in both cases, and there is no need for parameter tuning in the alignment detection stage.

2.5.1 Manhattan world, calibrated camera

When the camera parameters are known, the candidate VPs can be represented in the Gaussian sphere. The statistical meaningfulness given by the NFA is exploited by selecting the orthogonal triplet (up to a tolerance γ) with the lowest combined NFA value as the final VP triplet. If no orthogonal triplet is found, the best orthogonal pair is selected – again, in terms of its combined NFA – and the third VP is estimated by the cross product.

2.5.2 Non-Manhattan world, uncalibrated camera

In this case, similarly to [21], the algorithm starts by identifying the vertical VP: The vertical distance from a VP to the center of the image is used to form a subset of possible vertical VPs, and the most meaningful one (with lowest NFA) is kept as the zenith[2] $\mathbf{z}$. Here it is assumed that the principal point lies at the center of the image, and by enforcing orthogonality, the direction of the horizon line is obtained, which is perpendicular to the line connecting the principal point and $\mathbf{z}$. Therefore, the problem of estimating the horizon line is one-dimensional. The candidate VPs are filtered out based on two criteria: First, and similar to [21], the focal length f is estimated, $f \in [0.28W,\ 3.8W]$ and the VPs that are far from being orthogonal to $\mathbf{z}$ are discarded based on the orthogonality threshold γ of Sect. 2.5.1 and a distance threshold η. Note that the focal length f is used only to discard non-horizontal VPs, so an approximate value is generally enough. The results suggest that an appropriate estimation is obtained. Second, the horizontal VPs that are obtained from clusters of parallel lines – for which the problem of setting the horizon line height is undetermined – are also discarded, based on a threshold λ. Finally, the horizon line is obtained by a voting scheme [21, 4, 19]. Whilst [21] considers the inverse of the trace of each VP's covariance matrix, and [4, 19] consider the number of lines corresponding to each VP, we use weights based on the NFA (Sect. 2.1) of each VP detection. The weight w_i for the VP $\mathbf{v}_i$ is:

$$w_i = \left(\frac{(-\log_{10} \mathrm{NFA}_i)}{\sum_j (-\log_{10} \mathrm{NFA}_j)} \right)^2 . \qquad (2)$$

[2] Technically it should be referred to as nadir when it is below the horizon line, but we shall call it zenith without loss of generality.

Once we obtain a first candidate for the horizon line, we filter out VP candidates that are far away from it with a threshold κ and re-estimate. The reason for this re-estimation is that good candidates should be close to the real horizon line.

2.6. Parameters

The alignment detector is always used as described in [13], without modification or adjustment. There are eight parameters in the VP detection method that are estimated in the training sets of the evaluation databases using grid search. Table 1 provides the values. The only sensitive parameter is the focal length f. The other parameters can be fixed for both datasets (under the non-Manhattan assumption), still producing state-of-the-art results (see Sect. 3.1).

Name	Sect.	Value
τ	2.2	25 pixels
δ	2.3	$\frac{\|\mathbf{v}_1-\mathbf{v}_2\|}{\max(\|\mathbf{v}_1\|,\|\mathbf{v}_2\|)} < 0.02$
θ	2.4	2 degrees
γ	2.5.1 & 2.5.2	$\|\mathbf{v}_{1_u} \cdot \mathbf{v}_{2_u}\| > 0.6$
f	2.5.2	$3W$
η	2.5.2	$\mathbf{v}_y > 2.5H$
λ	2.5.2	$\mathbf{v}_x > 4W$
κ	2.5.2	$dist > 0.2H$

Table 1. Parameters of the proposed method. W and H are the image width and height. The subscripts x and y indicate the horizontal and vertical coordinates in the image and the subscript u indicates a vector in the Gaussian sphere (unit norm).

3. Experiments

We used the C code of [13] for alignment detection and the C code of [11] for line segment detection. The rest of the algorithm was implemented in MATLAB. Processing a 640x480 image takes an average of 30 seconds in a 2 Ghz Intel Core i7 laptop with 4 GB of RAM. The bottleneck of the method is the computation of point alignments in *straight* and *twisted* dual domains, which accounts for more than 90% of the processing time. The current alignment detector performs an expensive exhaustive search, which could be improved using heuristics (*e.g.* RANSAC). Furthermore, the code is near-linear parallel and can be sped-up by running on a multiprocessor platform.

As in previous works [4, 19, 20, 21], the horizon detection error metric is used to evaluate the performance of our algorithm. This measure is defined as the maximum distance in the image domain between the estimated horizon line and the ground truth, divided by the image height.

3.1. York Urban Dataset

The York Urban Dataset (YUD) [9] is widely used for VP detection quantitative evaluation. It includes 102 images of outdoor and indoor scenes, the camera parameters, and the ground truth VP triplets. Since all the images satisfy the Manhattan world assumption, we are in the case described in Sect. 2.5.1. Following the protocol of [4], the first 25 images were used to adjust our method's parameters (except for the alignment detector, always used as described in [13]) and the evaluation was performed on the remaining 77. The first three columns of Fig. 5 show qualitative results for images from YUD.

Fig. 6(a) shows a quantitative comparison of our results to those of [12], [18], [4], [19], [20] and [21]. The overall score is measured as the area under the curve (AUC) of the cumulative histogram of the horizon line detection error. We used the values published in [21]. In terms of the AUC score, our algorithm achieves a performance improvement of nearly 2%. It obtains a maximum horizon error of 0.052 against 0.078 in [21]. It should be noted that [21], [4] and [19] do not impose the Manhattan world hypothesis, whilst [18] and [20] do. For a fairer comparison, our method for the non-Manhattan scenario, as described in Sect. 2.5.2, was also evaluated on this dataset. The same parameters as for the Eurasian Cities Dataset were used (see Sect. 3.2), except for the provided focal length f. Our method still obtains a 1% AUC performance gain.

3.2. Eurasian Cities Dataset

The Eurasian Cities Dataset (ECD) [4] presents a much more challenging dataset of 103 urban scenes that do not satisfy the Manhattan world assumption in general. They depict different architectural and urban styles and are taken by different cameras. Under these conditions, the method described in Sect. 2.5.2 is used. Again, the first 25 images are used to train the parameters of the algorithm [4]. The three rightmost columns of Fig. 5 show results for images from ECD. Note that line segments that correspond to horizontal, non-orthogonal VPs are correctly grouped.

Fig. 6(b) shows the cumulative histogram of horizon line errors for various methods [12], [18], [4], [19], [20] and [21]. These values are the same published in [21] and, as before, represent the horizon line error metric. To the best of our knowledge, the state-of-the-art performance on this dataset is obtained by Xu *et al.* [21]. Our method performs slightly better than theirs in terms of the AUC score, but has a worse maximum horizon error of 0.117 against 0.081. It should be noted here that there is always an intrinsic error in this kind of ground truth. This error may be estimated by observing that images 43 and 51 are the same, but their ground truth horizons differ by 2 pixels. Assuming the same error is present in the rest of the dataset, it accounts for 1 % of the AUC score. With this consideration, one may say that our performance and Xu *et al.*'s are at the same level.

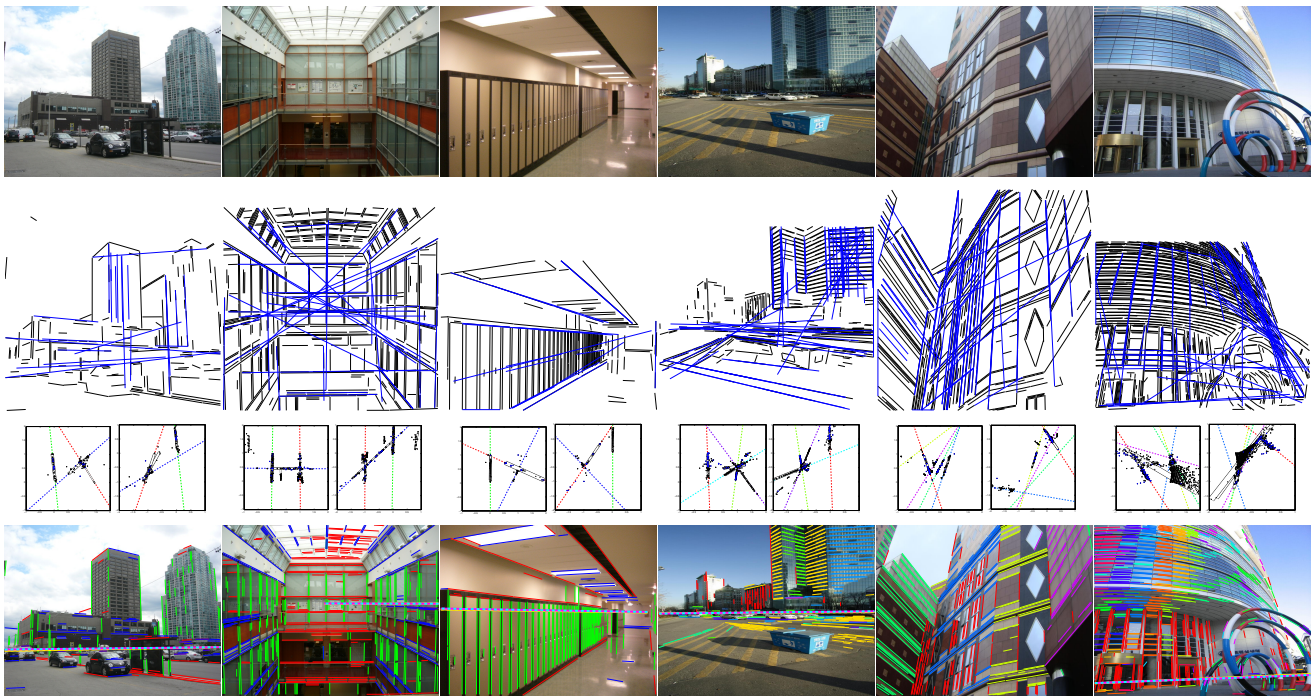
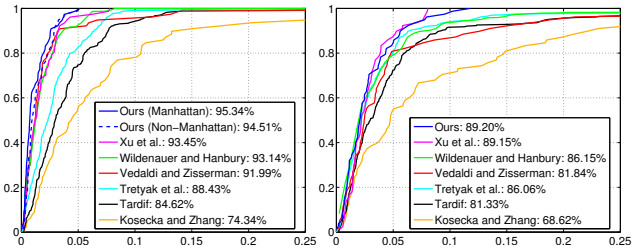


Figure 5. Some results of our method. The first three columns correspond to images from the York Urban Dataset, while the last three columns are from the Eurasian Cities Dataset. **Top Row:** Original image. **2nd Row:** Line segments (black) and alignments of line segments endpoints (blue). **3rd Row:** PClines *straight* and *twisted* dual spaces and point alignment detections (parallel black lines). The ground truth is represented with colored dashed lines. **Bottom Row:** Line segments corresponding to each final VP detection and horizon line (yellow-orange: ground truth, magenta-cyan: ours). In the last row, line segments from endpoint alignments have been removed for clarity. Note that the refinement and redundancy steps are not represented in this figure.



(a) York Urban Dataset (b) Eurasian Cities Dataset

Figure 6. Cumulative histograms of the horizon detection error. The horizontal axis represents the horizon line error [4]. The vertical axis represents the ratio of images with horizon line error lower than the corresponding abscissa.

4. Conclusion

We have introduced a method for vanishing point detection based on four key ideas: First, the use of a robust point alignment detector, used in the image domain as well as in dual space, without any modification or parameter tuning. Second, finding alignments of line segment endpoints, which enhances the accuracy of short line segments and produces new oriented elements as extra cues to the vanishing directions. Third, the use of the PClines parameterization in its two variants, *straight* and *twisted*, to improve the discriminability of vanishing points. Finally, exploiting the measure of meaningfulness provided by the alignment detector to estimate the horizon line. Our experimental results show that our method performs, in general, as well as state-of-the-art methods, and achieves significantly better accuracy when the Manhattan world assumption is applicable. The main drawback of our method is a high computational cost; future work will concentrate on improving speed.

Acknowledgments

We would like to thank E. Meinhardt-Llopis for his most valuable suggestions and Y. Xu and H. Wildenauer for their validation data and helpful comments. This work was partially supported by the Centre National d'Etudes Spatiales (CNES, MISS Project), the European Research Council (Advanced Grant Twelve Labours), the Office of Naval Research (Grant N00014-97-1-0839), Direction Générale de l'Armement (DGA), Fondation Mathématique Jacques Hadamard and Agence Nationale de la Recherche (Stereo project).

References

[1] A. Almansa, A. Desolneux, and S. Vamech. Vanishing point detection without any a priori information. *TPAMI*, 25(4):502–507, 2003. 1, 2

[2] M. Antunes and J. P. Barreto. A global approach for the detection of vanishing points and mutually orthogonal vanishing directions. In *CVPR*, 2013. 2, 4

[3] D. H. Ballard and C. M. Brown. *Computer Vision*. Prentice-Hall, 1982. 2

[4] O. Barinova, V. Lempitsky, E. Tretiak, and P. Kohli. Geometric image parsing in man-made environments. In *ECCV*, 2010. 1, 2, 4, 5, 6

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

[6] J.-C. Bazin, Y. Seo, C. Demonceaux, P. Vasseur, K. Ikeuchi, I. Kweon, and M. Pollefeys. Globally optimal line clustering and vanishing point estimation in Manhattan world. In *CVPR*, 2012. 2

[7] R. T. Collins and R. S. Weiss. Vanishing point calculation as a statistical inference on the unit sphere. In *ICCV*, 1990. 1, 2

[8] J. M. Coughlan and A. L. Yuille. Manhattan world: Orientation and outlier detection by Bayesian inference. *Neural Computation*, 15(5):1063–1088, 2003. 1, 2

[9] P. Denis, J. H. Elder, and F. J. Estrada. Efficient edge-based methods for estimating Manhattan frames in urban imagery. In *ECCV*, 2008. 1, 2, 4, 5

[10] M. Dubská, A. Herout, and J. Havel. PClines - line detection using parallel coordinates. In *CVPR*, 2011. 2, 3, 4

[11] R. Grompone von Gioi, J. Jakubowicz, J. M. Morel, and G. Randall. LSD: a line segment detector. *Image Processing On Line*, 2012. 2, 3, 5

[12] J. Kosecká and W. Zhang. Video compass. In *ECCV*, 2002. 2, 5

[13] J. Lezama, R. Grompone von Gioi, J.-M. Morel, and G. Randall. A Contrario 2D Point Alignment Detection. Submitted to TPAMI. Preprint: http://hal.inria.fr/hal-00956596. 2, 3, 5

[14] F. M. Mirzaei and S. I. Roumeliotis. Optimal estimation of vanishing points in a Manhattan world. In *ICCV*, 2011. 2

[15] L. Quan and R. Mohr. Determining perspective structures using hierarchical Hough transform. *Pattern Recognition Letters*, 9:279–286, 1989. 2

[16] C. Rother. A new approach to vanishing point detection in architectural environments. *Image and Vision Computing*, 20(9):647–655, 2002. 1, 4

[17] G. Schindler and F. Dellaert. Atlanta world: An expectation maximization framework for simultaneous low-level edge grouping and camera calibration in complex man-made environments. *CVPR*, 2004. 2

[18] J.-P. Tardif. Non-iterative approach for fast and accurate vanishing point detection. In *ICCV*, 2009. 2, 5

[19] A. Vedaldi and A. Zisserman. Self-similar sketch. In *ECCV*, 2012. 1, 3, 4, 5

[20] H. Wildenauer and A. Hanbury. Robust camera self-calibration from monocular images of Manhattan worlds. In *CVPR*, 2012. 1, 2, 5

[21] Y. Xu, S. Oh, and A. Hoogs. A minimum error vanishing point detection approach for uncalibrated monocular images of man-made environments. *CVPR*, 2013. 1, 2, 4, 5

[22] Y.-G. Zhao, X. Wang, L.-B. Feng, G. Chen, T.-P. Wu, and C.-K. Tang. Calculating vanishing points in dual space. In *Intelligent Science and Intelligent Data Engineering*, volume 7751 of *Lecture Notes in Computer Science*, pages 522–530. Springer Berlin Heidelberg, 2013. 2