

This is a postprint version of the following published document:

García-Faura, Álvaro, Fernández-Martínez, Fernando, Kleinlein, Ricardo, San-Segundo, Rubén, Díaz-de-María, Fernando (2019). A multi-threshold approach and a realistic error measure for vanishing point detection in natural landscapes. *Engineering Applications of Artificial Intelligence*, (2019), 85, pp.: 713-726.

DOI: <https://doi.org/10.1016/j.engappai.2019.08.001>

© 2019 Elsevier Ltd. All rights reserved.



This work is licensed under a [Creative Commons Attribution-NonCommercialNoDerivatives 4.0 International License](https://creativecommons.org/licenses/by-nc-nd/4.0/).

Manuscript Number: EAAI-18-2497R1

Title: A Multi-Threshold Approach and a Realistic Error Measure for
Vanishing Point Detection in Natural Landscapes

Article Type: Research paper

Keywords: Vanishing Point; Multi-Threshold; Perspective; Landscapes

Corresponding Author: Mr. Álvaro García-Faura,

Corresponding Author's Institution: E.T.S.I. de Telecomunicación

First Author: Álvaro García-Faura

Order of Authors: Álvaro García-Faura; Fernando Fernández-Martínez;
Ricardo Kleinlein; Rubén San-Segundo; Fernando Díaz-de-María

Abstract: Vanishing Point (VP) detection is a computer vision task that can be useful in many different fields of application. In this work, we present a VP detection algorithm for natural landscape images based on a multi-threshold edge extraction process that combines several representations of an image, and on novel clustering and cluster refinement procedures. Our algorithm identifies a VP candidate in images with single-point perspective and improves detection results on two datasets that have already been tested for this task. Furthermore, we study how VP detection results have been reported in literature, pointing out the main drawbacks of previous approaches. To overcome these drawbacks, we present a novel error measure that is based on a probabilistic consistency measure between edges and VP hypothesis, and that can be tuned to vary the strictness on the results. Our reasoning on how our measure is more correct is supported by an intuitive analysis, simulations and an experimental validation.

A Multi-Threshold Approach and a Realistic Error Measure for Vanishing Point Detection in Natural Landscapes

Álvaro García-Faura^a, Fernando Fernández-Martínez^a, Ricardo Kleinlein^a,
Rubén San-Segundo^a, Fernando Díaz-de-María^b

^a*Information Processing and Telecommunications Center, E.T.S.I. de Telecomunicación,
Universidad Politécnica de Madrid, Avda. Complutense, 30, 28040, Madrid, Spain*

^b*Signal Theory and Communications Department, School of Engineering, Universidad
Carlos III de Madrid, Av. de la Universidad, 30, 28911, Leganés, Spain*

Abstract

Vanishing Point (VP) detection is a computer vision task that can be useful in many different fields of application. In this work, we present a VP detection algorithm for natural landscape images based on an multi-threshold edge extraction process that combines several representations of an image, and on novel clustering and cluster refinement procedures. Our algorithm identifies a VP candidate in images with single-point perspective and improves detection results on two datasets that have already been tested for this task. Furthermore, we study how VP detection results have been reported in literature, pointing out the main drawbacks of previous approaches. To overcome these drawbacks, we present a novel error measure that is based on a probabilistic consistency measure between edges and VP hypothesis, and that can be tuned to vary the strictness on the results. Our reasoning on how our measure is more correct is supported by an intuitive analysis, simulations and an experimental validation.

Keywords: Vanishing Point, Multi-Threshold, Perspective, Landscapes

Email addresses: agfaura@die.upm.es (Álvaro García-Faura),
fernando.fernandezm@upm.es (Fernando Fernández-Martínez), ricardo.kleinlein@upm.es
(Ricardo Kleinlein), ruben.sansegundo@upm.es (Rubén San-Segundo), fdiaz@tsc.uc3m.es
(Fernando Díaz-de-María)

1. Introduction

In recent years, Computer Vision has been gaining increasing interest within the huge field of Artificial Intelligence, since its applications in systems used by the general public continue to grow. The research presented here intends to provide new tools for this trend to advance in the same direction.

When projecting lines that are parallel in 3D space onto an image plane, which is two-dimensional, they appear to converge to a point. This point on the image is referred to as *Vanishing Point* (VP). Many Computer Vision problems could benefit from an automatic and accurate VP detector, not only when its application is obvious, such as in camera calibration, but also in other fields. For instance, the presence of VPs in an image determine its perspective, which in turn characterizes it in a wider sense. Perspective is valuable, among other things, for studying photography composition and aesthetics.

The detection of VPs in an image is a challenging problem that has been repeatedly tackled, as it has proven to be useful in a wide variety of applications ranging from 3D reconstruction to road detection. Here, we present a method for automatically detecting VPs on landscape images. Some natural scenery images featuring a VP, and taken from the datasets we will later use, can be seen in Figure 1.

Accurate VP detection requires understanding of how things are arranged in the image. To do so, different approaches can be undertaken, being the most typical one the detection of edges in the image. However, methods usually present limitations that makes them unsuitable for certain type of images. An example to this are images presenting curves or successions of objects that, although creating VPs that would be clear for humans, make edge-based algorithms fail. By combining several representations of the same image, obtained using an scale-invariant contour splitting procedure, we think to have taken one step forward in solving this issue. This way, our detection algorithm makes less assumptions on the structure of the image and, additionally, performs a more relaxed edge grouping.

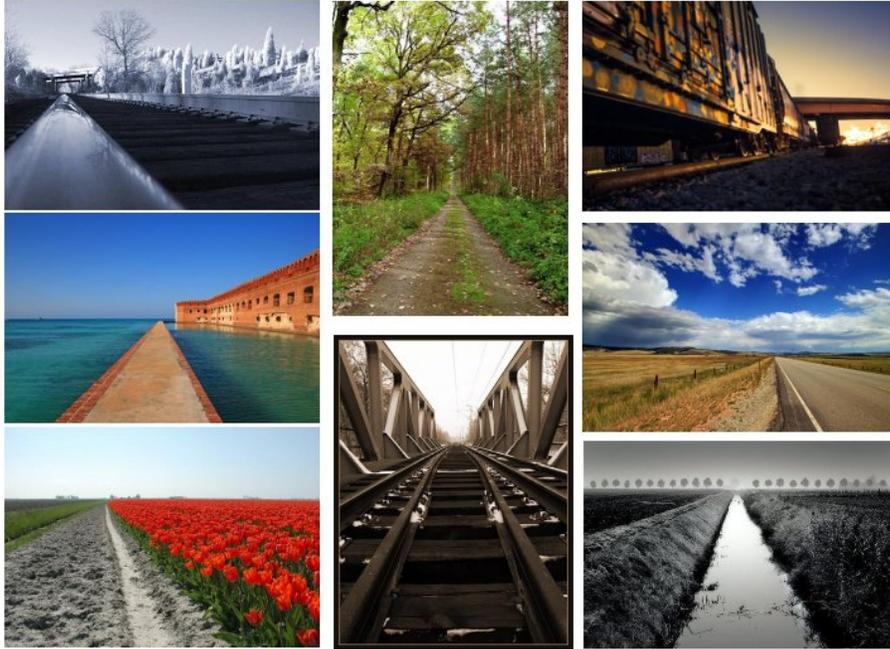


Figure 1: Single-point perspective images of natural scenes taken from the databases that are used in this work.

In literature, VP detection has usually been closely tied to specific applications. Because of that, previous research tend to report detection results depending on the application and also on how the dataset is labeled: there is no consensus on the way of measuring the accuracy of a VP detection algorithm. Here, we analyze weaknesses in previously proposed methods and present a novel tunable technique that, in order to ensure a correct modeling of detection errors, is less permissive than others.

The two main contributions presented in this paper are:

- A novel VP detection algorithm that provides better detection results on existing datasets by improving edge extraction, edge clustering, and cluster refinement techniques. It is presented in Section 3.
- A novel method for measuring the accuracy of VP detection results that encodes better the actual performance of algorithms. It is presented in

Section 4 .

2. Related Work

The Vanishing Point detection problem has been tackled many times in the past, featuring different methods for different applications. Accordingly, VP detection techniques can be classified into several categories depending on the family of algorithms they use, the assumptions on the image characteristics they make, or the camera parameters they assume to know. In this section, we review some of the most recent works and classify them depending on whether they consider to know or not any intrinsic camera parameter (calibrated vs. uncalibrated case). Most of the solutions reviewed here, as ours, rely on detected line segments in the image, which is the most common practice for VP detection. However, we will also make some comments on novel techniques that make use of Deep Learning to solve the VP detection problem.

2.1. Calibrated VP Detection

Firstly, VP detection techniques were based on the knowledge of camera intrinsic parameters. The primary work on VP detection is considered to be [3], in which edges are mapped onto the Gaussian Sphere, being the camera focal length known. Image edges are represented by circles on the sphere whose intersection point denotes the VP location. However, as it was pointed out in a more recent work [20], this method lacked robustness against texture edges and weak perspective effects. Since then, researchers have proposed different solutions for reducing the number of spurious results.

One possibility that leads to better detection results consists in forcing orthogonality between the three main real-world vanishing directions. This is known as the Manhattan-world assumption [7], that limits detection to two horizontal and one vertical VP. This restriction is used in several publications in combination with the knowledge of camera parameters [10, 16, 4]. In [16], they estimate the three VPs by formulating a constrained least-squared opti-

mization problem and solving it analytically in combination with RANSAC [8], which is a clustering algorithm.

Another category of estimators use Expectation-Maximization (EM) algorithms for iteratively refining detection results. This was first proposed for the calibrated case in [1].

2.2. Uncalibrated VP Detection

With respect to the uncalibrated case, certain techniques make some assumptions on the camera parameters —as in [14, 24], where a value for the focal length is manually selected—, while most of them considers the parameters completely unknown.

Similarly to [1], [13] used EM techniques for the case without knowing the internal camera parameters.

More Recent works rely on algorithms similar to RANSAC for grouping edges into clusters, some of them also defining a way of measuring consistency between edges and VP candidates [21, 26, 24, 23]. In [21], line segment clusters are created using J-Linkage [22], a model fitting algorithm based on RANSAC. Similar processes are carried out in [26], but applied to natural scenes and using a novel contour-based edge extraction method. Also based on J-Linkage there is [24], where a new consistency error measure and a minimum error VP estimator are defined. Our work falls into the same category of the last mentioned ones, although we use a novel RANSAC-based model fitting algorithm and define our own cluster refinement technique.

Notice that the Manhattan-world assumption can be also applied for the uncalibrated case. For instance, it is used in [21] for the estimation of the focal length and in [24, 14] for the estimation of the horizon line, which is a common application of VP detection.

We conclude mentioning novel methods that perform uncalibrated VP detection that are based on Convolutional Neural Networks, such as [19], in which they use a regression CNN based on AlexNet; or [11], in which the CNN is fed with a Gaussian sphere representation of the edges, instead of the image

itself. State-of-the-art performance on well-known datasets of man-made environments was achieved in [25], in which they use a CNN to consider the global image context when estimating the horizon line and the zenith VP.

3. VP Detection Algorithm

In this section, we explain in detail the complete procedure that allows us obtaining VP hypothesis from any regular image. First, we detect straight edges on an image and filter out those which are more likely to be noisy. Then, we group edges using a clustering method that considers where they point to in the image. Finally, we refine the obtained clusters, removing outliers and regrouping the edges that remain, and select the hypothesis which is considered to be the dominant VP in the image. An illustrative diagram of the whole process can be seen in Figure 2.

3.1. Edge Detection

As previously stated, VPs are the consequence of the presence of converging lines in an image. Accordingly, the first logical step will be to obtain these lines from the image, what can be done employing several different techniques.

Many previous works in VP detection make use of the classical approach for edges extraction, the Canny Edge Detection algorithm [6], while others use the more recent detection technique, Line Segment Detector (LSD) [9]. However, as it was previously pointed out in [26], edge detectors that only consider local information are not able to distinguish the relevance of edges with respect to each other and therefore to the whole image. In order to overcome this issue, in that paper they used the contour detector described in [2], which combines local and global information to detect contours and generates a hierarchical representation of the regions within an image. This representation, which is referred to as *Ultrametric Contour Map* (UCM), is the one we use for edge detection.

Having computed the UCM of the image, we proceed to find the connected pixels presenting the same UCM level. We obtain a set of contours, each one

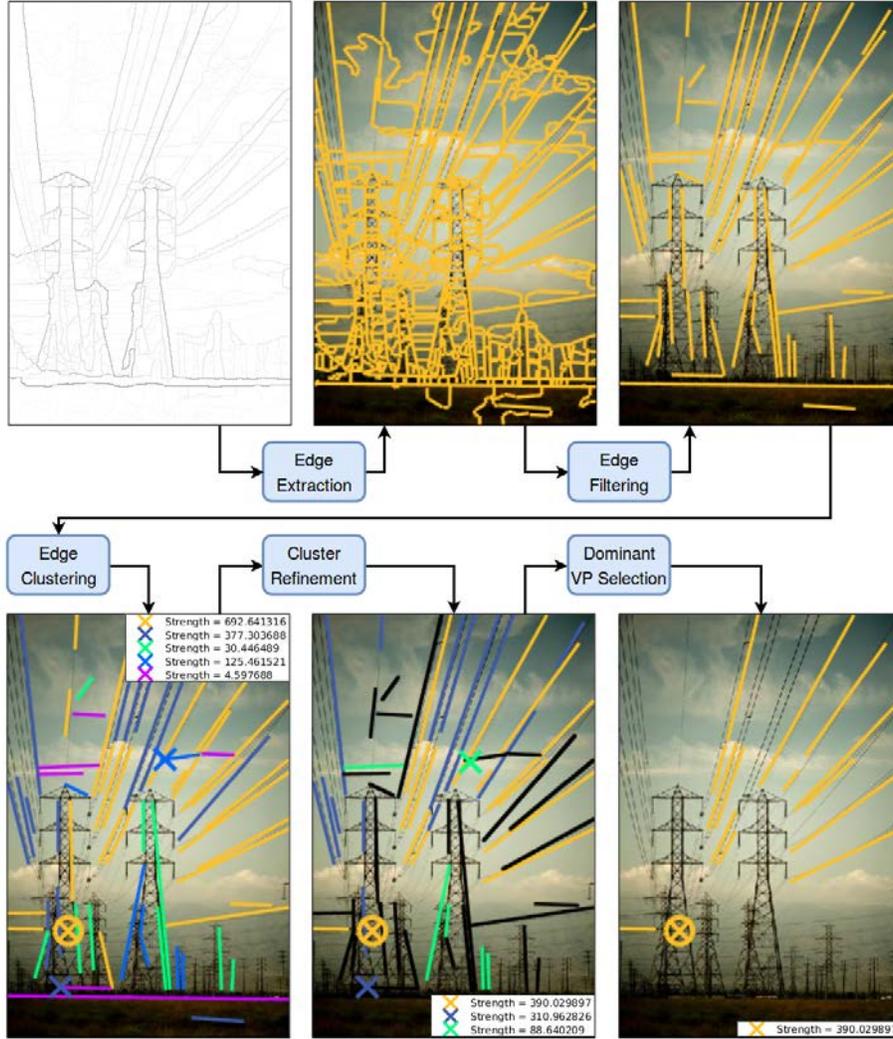


Figure 2: Block diagram of the whole VP detection process for an example image, going from the Ultrametric Contour Map (UCM) to the detected dominant VP.

of them labeled with a probability value. In order to generate straight edges out of the set of contours, we apply a scale-invariant splitting procedure similar to that presented in [26]. First, for every contour C_j , and being \mathbf{c}_j^1 and \mathbf{c}_j^2 its endpoints, we get the point $\hat{\mathbf{p}}$ that belongs to the contour and whose distance

to the straight line connecting the endpoints, $\overline{\mathbf{c}_j^1 \mathbf{c}_j^2}$, is maximal:

$$\hat{\mathbf{p}} = \arg \max_{\mathbf{p} \in C_j} (\text{dist}(\mathbf{p}, \overline{\mathbf{c}_j^1 \mathbf{c}_j^2}))$$

where dist is the euclidean distance. Then, C_j will be split at $\hat{\mathbf{p}}$ if that maximal distance is greater than a certain fraction, α , of the length of $\overline{\mathbf{c}_j^1 \mathbf{c}_j^2}$:

$$\text{dist}(\hat{\mathbf{p}}, \overline{\mathbf{c}_j^1 \mathbf{c}_j^2}) > \alpha \cdot \text{dist}(\mathbf{c}_j^1, \mathbf{c}_j^2)$$

While [26] uses a fixed value for α , we consider that different images would need different values for it depending on the characteristics of its contours. For instance, there may be images with some kind of building in it, which mostly produces well-defined, straight contours. On the contrary, other images may contain big areas of vegetation, which usually lead to shorter and more irregular contours. In other words, the optimal value for α is likely to have a strong dependency upon the image content. One of the objectives of this work is to make the edge extraction process as independent as possible from the image. For such purpose, we will apply the previously explained method for every image using a set of values for α and, afterwards, we group together the resulting edges for all of them. If two or more α values generate the same exact edge, it is included only once. The detection results obtained with this combination of edges will be later compared to those obtained with a single value for α .

3.2. Edge Filtering

Before getting into edges grouping, we filter some of them out to keep only those that are more likely to be valuable. Edges shorter than certain length l are removed. Short edges are usually the result of textures, objects, or simply image noise, thus not giving information about perspective.

We also filter out noisy edges which are likely to be a consequence of a framed image. In such a case, these edges—which are straight, long, and sharp, therefore not being split nor filtered out—happen to be among the longer ones. This would lead to them acquiring an undeserved high relevance,

as we will see later on. To avoid so, edges whose two endpoints are closer than b pixels to the same image border are suppressed. Even though this filtering could remove legitimate relevant edges in unframed images, it is very unlikely that it affects significantly the detection results, since it is reasonable to assume that the location of the VP will be defined by several other edges not so close and parallel to the image border. The presence of a few artificial, long, sharp, and straight edges would lead to worse results than the absence of some legitimate, regular edges since, as mentioned, the former would acquire high importance in our algorithm.

Finally, horizontal edges are also filtered out as, in the 3D scene, they correspond to lines which are parallel to the camera projection plane (image plane) and, when projected, will not denote the presence of a VP in the image. An exception to this is the horizon line, which is not a *real* line in the 3D world but can appear as an horizontal line in an image. However, as it is not necessarily present on all images, we find more effective to filter out all horizontal lines, thus removing many irrelevant edges. We define a parameter ϕ that establishes an edge’s minimum angle with respect to the horizontal axis for it to be kept.

3.3. Edge Clustering

After edges extraction and filtering, every image will be represented by a set of straight edges $\mathcal{E} = \{E_1, \dots, E_n\}$. Having several edges pointing towards the same area in the image will potentially denote the presence of a VP. Consequently, our goal is now to identify groups of edges fulfilling this requirement. To do so, many previous approaches [26, 24, 21] have used the multi-model fitting algorithm J-Linkage [22]. However, we have decided to use T-Linkage [15], which is a continuous relaxation of J-Linkage that we think could better fit our problem conditions, as it avoids the need to determine a threshold in order to convert a continuous problem into a binary one. The rest of this section is organized as follows: first, we define a way of measuring consistency between an edge and a VP hypothesis that we will need afterwards to group edges consistent with the same hypothesis; then, we proceed to explain the proper clustering

algorithm.

3.3.1. Probabilistic Consistency Measure

The consistency measure we use is the one defined in [24], where the authors state the main drawbacks of other measures and prove theirs to reflect better the concept of consistency between an edge and a VP. The function considers both the degree of misalignment, modeled probabilistically, and the distance between the edge and the VP. To the best of our knowledge, this is the first time that this consistency measure is used in conjunction with T-Linkage.

For an edge E_j , the *true* location, \hat{e}_j^1 and \hat{e}_j^2 , of its endpoints e_j^1 and e_j^2 , respectively, is modeled by a Gaussian distribution. By transforming the coordinates, the endpoints of the *true* edge can be seen as $\hat{e}_j^1 = [0, \hat{y}_j^1]$ and $\hat{e}_j^2 = [L, \hat{y}_j^2]$, where L is the edge length, $\hat{y}_j^1 \sim \mathcal{N}(0, \sigma)$, and $\hat{y}_j^2 \sim \mathcal{N}(0, \sigma)$. Here, σ is the standard deviation and encodes the edges extraction error. The VP hypothesis is $\mathbf{v}_i = [x_i, y_i]$. Being $x_i > L$ and \hat{E}_j (the *true* edge) collinear with \mathbf{v}_i , then $\hat{y}_j^2 = \frac{x_i - L}{x_i} \hat{y}_j^1 + \frac{y_i L}{x_i}$. Finally, after certain parametrization —not included here for brevity— using an auxiliary variable t , the consistency measure is:

$$c(E_j, \mathbf{v}_i, \sigma) = \int_{-\infty}^{\infty} f(\hat{y}_j^1(t); 0, \sigma^2) f(\hat{y}_j^2(t); 0, \sigma^2) dt = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{y_i^2 L^2}{2\sigma^2(x_i^2 + (x_i - L)^2)}} \quad (1)$$

where $f(\cdot; \mu, \sigma^2)$ is a Gaussian PDF.

In Figure 3 we can appreciate the influence of the edge length, L , and the edge extraction error, σ , on the consistency measure. For a given σ , longer edges produce narrower ridges, while for a given L , smaller edge extraction error also produces narrower ridges and a higher maximum possible value. This means that the shorter the edge and the greater the edge extraction error, the higher the uncertainty we have about the true VP location.

3.3.2. T-Linkage Clustering

In T-Linkage, as in J-Linkage, a *Minimal Sample Set* (MSS) is defined as the minimal set of data from which a model hypothesis can be generated. In the case of Vanishing Point estimation, a MSS is formed by a set of two different

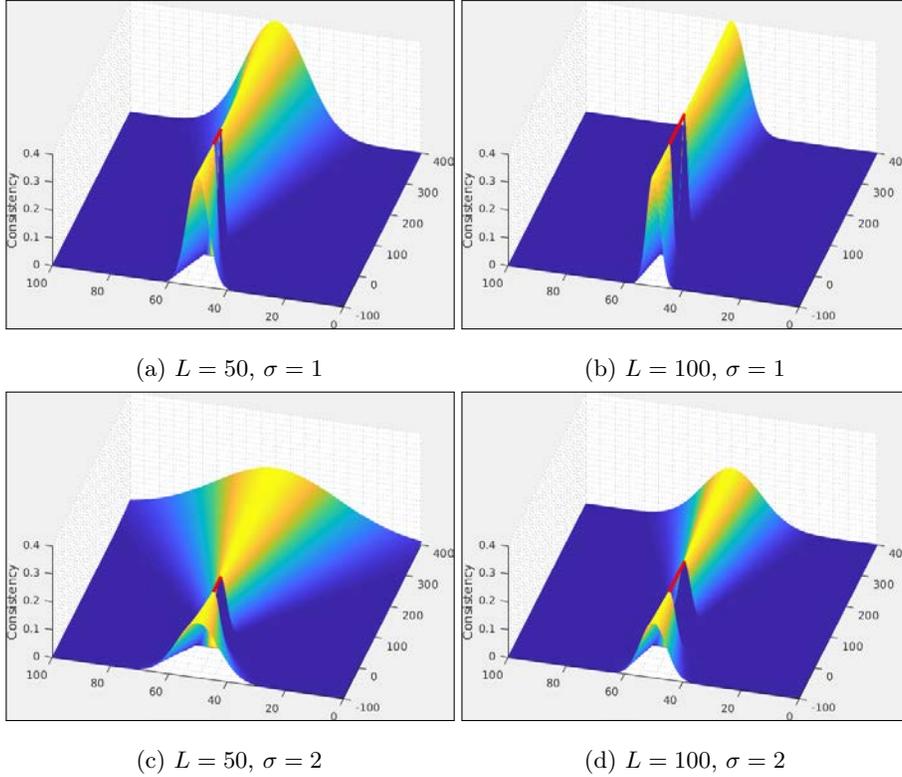


Figure 3: Representation of the probabilistic consistency measure (Eq. 1) for different values of the edge extraction error, σ , and the edge length, L .

straight edges $\varepsilon_i = \{E_{i_1}, E_{i_2}\} \subseteq \mathcal{E}$, which are randomly sampled from the set of all the edges in an image. Both algorithms start with the generation of m random model hypothesis from m randomly chosen MSSs. The model hypothesis they produce is given by $\mathbf{v}_i = \mathbf{l}_{i_1} \times \mathbf{l}_{i_2}$, where \mathbf{l}_{i_1} and \mathbf{l}_{i_2} are the lines (in homogeneous coordinates) corresponding to E_{i_1} and E_{i_2} respectively. The resulting VP hypothesis, \mathbf{v}_i , is also represented in homogeneous coordinates.

Having computed the set of m VP hypothesis, T-Linkage will define a space in which each data element, in our case each edge E_j , is represented by the set of models (VP hypothesis) it matches, which will be referred to as the Preference Function (PF) of E_j . Thus, we need a way of measuring the degree of fitting between an edge and a VP hypothesis. For this purpose, we use the previously

defined function $c(E_j, \mathbf{v}_i, \sigma)$, which measures the consistency between an edge, E_j , and a point, \mathbf{v}_i , considering certain edge extraction error encoded by σ .

With this information, we can now build a $n \times m$ matrix, being n the number of edges and m the number of VP hypotheses, in which each row is an edge's PF, that is, it represents an edge by the consistency it presents with respect to every hypothesis. The consistency values within the matrix are normalized by the maximum so that the greater value is always equal to 1. T-Linkage will then measure the *Tanimoto distance*, as it is defined in [15], between every row (PF), and will merge those two rows presenting the lowest distance into the same cluster, following an agglomerative clustering procedure. The PF of the union will have, for each VP hypothesis, the minimum value of consistency between that of all the edges within the cluster. This means that when a cluster is formed, its consistency with certain VP candidate will be the minimum consistency with that candidate of the edges that form the cluster.

Mathematically, suppose that $\mathbf{p}_i = c(E_j, \mathbf{v}_i, \sigma)$ and $\mathbf{q}_i = c(E_k, \mathbf{v}_i, \sigma)$, with $i = [1, m]$, are the respective PFs of two edges E_j and E_k . Now, suppose that no cluster has been formed yet, and that the Tanimoto distance between E_j and E_k is minimal among those of all possible pairs of edges. This would make them to be joined together in a cluster, and their PF would be:

$$\mathbf{r}_i = \min(\mathbf{p}_i, \mathbf{q}_i), \quad i = [1, m]$$

Afterwards, this clustering process will continue for the rest of edges, with \mathbf{p}_i and \mathbf{q}_i being replaced by \mathbf{r}_i , until all PFs are orthogonal among them.

Some interesting properties of the resulting clusters, which apply both for J-Linkage and T-Linkage, are that all VP hypothesis are fit by certain cluster and that there will not be two clusters consistent with the same VP hypothesis (otherwise they would have been merged). Edges within the same cluster would have expressed consistency for at least one common hypothesis. As a consequence of T-Linkage fitting all the data, there would be clusters consistent with bad hypothesis that must be treated afterwards. We refer to this process as Clusters Refinement, which we proceed to describe.

3.4. Cluster Refinement

Now, we need to determine which clusters are more relevant than others, what can be conditioned by the presence of outlier edges that do not show relevant consistency with any of the VP candidates. Similarly to previous works [24, 21], we will adopt a EM-like algorithm which will be applied iteratively for refinement. To be able to carry it out, there is a need that arises: clusters must be compared somehow and ranked accordingly, as eventually we will have to choose a dominant VP. For such purpose, we first present a measure that will be then used in our refinement algorithm.

3.4.1. Cluster Strength Measure

Having obtained a certain number of clusters, each with its associated VP hypothesis, we need a way to decide which are better candidates than others. Our assumption is that a cluster formed by more edges than other will probably be more relevant, and the longer the edges the stronger sense of depth they convey. Therefore, cluster sorting could be done by simply counting the number of edges within each cluster, or by adding their lengths. Consequently, the cluster containing the highest number of edges, or with highest aggregated edge length, would be chosen as the dominant cluster.

These two mentioned ways of ranking clusters are perfectly valid and work reasonably well. However, in [26], they came up with a function to measure the *strength* of a cluster that was intuitive and wisely defined. It takes into account both the number of edges in a cluster and their length, but also their distance to the candidate VP. For a VP hypothesis \mathbf{v}_i , its strength is defined as:

$$S(\mathbf{v}_i) = \sum_{E \in \mathcal{E}_i} \sum_{\mathbf{q} \in E} \frac{1}{l_{\mathbf{q}} + \tau} \quad (2)$$

where \mathcal{E}_i is the set of edges belonging to a certain cluster k_i , $l_{\mathbf{q}}$ is the distance in the image from a pixel \mathbf{q} to \mathbf{v}_i , and τ is a constant for enhancing detection robustness that determines the importance given to the edge's length versus its distance to the candidate. This strength measure was proven to perform slightly better than the others [26], so we have adopted it in our approach.

3.4.2. Refinement Algorithm

As already mentioned, once we have the output from T-Linkage, we perform an iterative process for refining the obtained clusters. Throughout an iteration, we will first compute the corresponding VP \mathbf{v}_i for each cluster K_i using the vanishing point estimation function, $V(\varepsilon_i)$, defined in [21]:

$$\mathbf{v}_i = V(\varepsilon_i) = \arg \min_{\mathbf{v}} \sum_{E_j \in \varepsilon_i} \text{dist}^2(\bar{\mathbf{e}}_j \times \mathbf{v}, \mathbf{e}_j^1) \quad (3)$$

where ε_i is the set of edges assigned to a cluster k_i , \mathbf{e}_j is the centroid of the edge E_j , \mathbf{e}_j^1 is one of the endpoints of E_j , and dist is the orthogonal distance from a line to a point. Note that $\bar{\mathbf{e}}_j \times \mathbf{v}$ is the line that passes through the centroid of the edge and the VP. Therefore, the resulting VP, \mathbf{v}_i , will be the one that minimizes the sum, for all edges in the cluster, of the squared distance from the edge’s endpoint to the line formed by itself and the centroid of that edge.

Then, we sort the clusters with respect to their strength $S(\mathbf{v}_i)$ and remove the VP hypothesis with minimal strength. This is done in every iteration until no more than three candidates are left. Besides, if the strength of the weakest candidate is lower than the 20% of that of the strongest, it will be removed too, no matter how many candidates are left. We establish the maximum VP hypothesis number to three for generalization, as it would be the number of VPs in case the previously explained Manhattan assumption (the image has three mutually orthogonal VPs) was fulfilled, though we know that in the datasets we use there is only one VP per image.

Note that we remove a VP hypothesis, but all the edges in its associated cluster are kept. Hence, the next step is to measure the consistency of every edge in the image with respect to every remaining VP candidate. Edges will be reassigned to the cluster whose VP hypothesis they present maximal consistency to or, alternatively, to a cluster of outliers in case their consistency is below a

certain threshold η :

$$E_j \in \begin{cases} k_i, & \text{if } \max(c(E_j, \mathbf{v}_i, \sigma)) > \eta \\ \text{outliers}, & \text{otherwise} \end{cases}$$

where k_i is the cluster whose VP candidate is \mathbf{v}_i . Finally, we merge clusters whose hypothesis are likely to be the same (less than 2 pixels far from each other) and start over until all edges remain in the cluster they were assigned to in the previous iteration. A summary of this refinement algorithm is presented in Algorithm 1.

3.5. Dominant Vanishing Point Selection

Finally, we have to select, among the clusters resulting from the refinement procedure, the one we consider to be dominant. It is worth reminding that in the datasets we use, only images with a single dominant vanishing point have been included, and that images with two or more vanishing points with the same visual relevance were excluded. Consequently, the VP hypothesis that will be selected to be the dominant VP in the image, $\hat{\mathbf{v}}$, will simply be the one that, after refinement, has maximum strength, $S(\hat{\mathbf{v}})$.

4. Vanishing Point Estimation Error Measure

Typically, the metric or metrics used for reporting an algorithm’s performance become a key issue in order to capture its actual behavior, as different ways of evaluating results may lead to different perceived performance when comparing algorithms. Thus, a metric must reflect to which extent our problem is solved as well as clearly indicate possible cases of failure.

Regarding Vanishing Point estimation, given that we want to compare two points in the image, the detected and the ground truth VP, the most straightforward error function could be measuring the euclidean distance between them. This measure has been used before, and may be suitable for specific cases in which the images we are benchmarking against present similar perspective, probably due to the peculiarities of that concrete problem. An example to this is VP

Algorithm 1 Cluster Refinement

Require: T-Linkage outputs K clusters, being ε_i the set of edges assigned to cluster $k_i, i \in [1, K]$

Reassignments = true

repeat

for all $k_i \in K$ **do**

$\mathbf{v}_i = V(\varepsilon_i)$ ▷ VP estimation function (Eq. 3)

$s_i = S(\mathbf{v}_i)$ ▷ Strength Measure (Eq. 2)

if $(\min(s) < 0.2 \times \max(s))$ or $(K > 3)$ **then**

 Remove $\hat{\mathbf{v}}_i = \underset{\mathbf{v}_i}{\arg \min}(S(\mathbf{v}_i))$

$K = K - 1$

for all $E_j \in \mathcal{E}$ **do**

if $\max_{i \in [1, K]} (c(E_j, \mathbf{v}_i, \sigma)) > \eta$ **then**

$i^* = \underset{i \in [1, K]}{\arg \max}(c(E_j, \mathbf{v}_i, \sigma))$

 Assign E_j to k_{i^*}

else

 Assign E_j to *outliers*

 Merge clusters whose hypothesis are closer than 2 pixels

if Every $E_j \in \mathcal{E}$ belongs to the same cluster than in previous iteration

then

 Reassignments = false

until Reassignments = false

detection for road following [12, 17]. In this case, images will always feature a road—or at least some kind of unstructured path—and will have always been taken from a vehicle front-camera, what makes them have similar perspective properties. However, in more general cases, the distance on the image will not give us a reliable metric on how well our algorithm is performing, as an image is a projection of a real world 3D scene. All Vanishing Points are caused by lines which are parallel on the real world, but the camera position with respect to

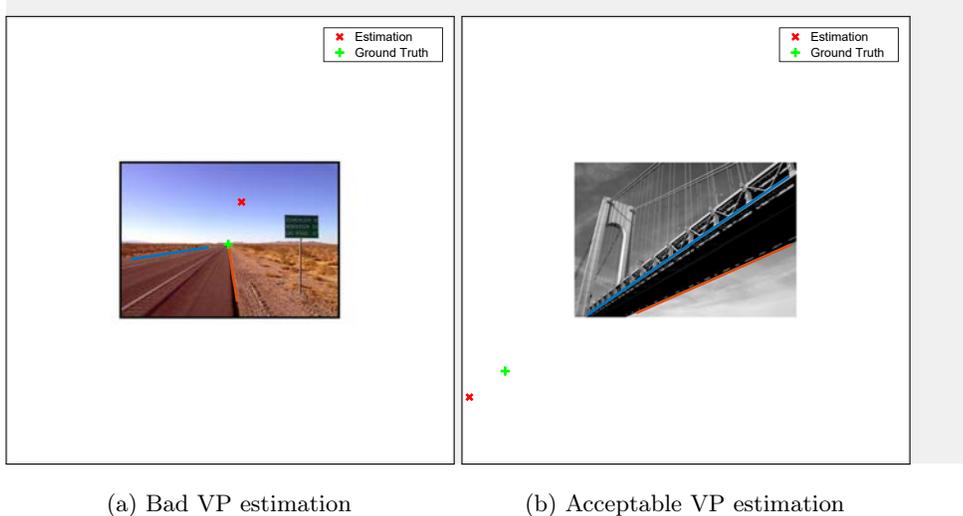


Figure 4: Comparison of VP estimations whose distance to the ground truth VP is equal to 100 pixels when the VP is: (a) close to or (b) far from the image center

those lines will make them to be projected differently on the image.

In this regard, two extreme opposite situations can be used to clarify the idea. In one case, when the image plane is parallel to those lines that are parallel among them in the real world, the VP will be at infinity, as the projected lines will also be parallel on the image plane. Contrarily, if the image plane is completely perpendicular to the real-world parallel lines, the VP will appear at the center of the image (the center of projection). Thus, intuitively, if we measure the error as a distance in the image, the actually perceived error will be much greater when estimating a VP which is close to image center compared to when it is very far away from the center. An example to this can be seen in Figure 4.

A possible solution to this problem may be mapping the vanishing directions to a point on the Gaussian Sphere, and then measuring the distance on it between the points originated from the estimated direction and the ground truth one [3]. However, for this mapping to be accurate the focal length of the lens has to be known, so we will not consider this method for solving our problem.

As seen in previous works, a reasonable solution is to define an error metric

based on the consistency measure between a VP and an edge. Both in [21] and [26], this kind of solution is used but, although quite similar, their consistency measures are not exactly the same. First, [21] defined its consistency measure as the orthogonal distance between an edge’s endpoint and the line passing through the VP and minimizing the maximal distance to that edge’s endpoints, which is proven to intersect the centroid of the edge. Note that, using this measure, the higher the misalignment between the VP and the edge, the higher values we obtain, so it could be argued that this measure actually encodes the inconsistency between a point and an edge. However, we will keep referring to it as "consistency measure," as this is how it is defined by the corresponding authors.

Afterwards, the consistency measure was defined in [26] as the root mean square distance of all the points on the edge E_i to a line \hat{l} that passes through the VP v_j and minimizes the function

$$D_{RMS}(E_i, v_j) = \min_{l: l \times v_j = 0} \left(\frac{1}{N} \sum_{p \in E_i} dist(p, l)^2 \right)^{\frac{1}{2}}$$

where N is the number of points on E_i , and $dist(p, l)$ is the perpendicular distance from a point p to a line l . The same reasoning we did for the previous measure can be applied here, as again higher measured values means lower actual consistency between the edge and the VP. Thus, provided that the optimal value for this consistency measure is 0, the consistency error of an estimated VP with respect to the ground truth edges, $\mathcal{G} = \{E_1, E_2\}$, is defined as the mean of their consistency measure with that VP:

$$err(v_j) = \frac{1}{2} \sum_{E_i \in \mathcal{G}} D_{RMS}(E_i, v_j) \quad (4)$$

However, in our opinion this measure does not accurately reflect the performance of the algorithm. VP hypothesis contained in one of the Ground Truth (GT) edges, or even consistent with only one of them, might obtain little error if GT edges have different lengths or are placed in certain manners. To help to

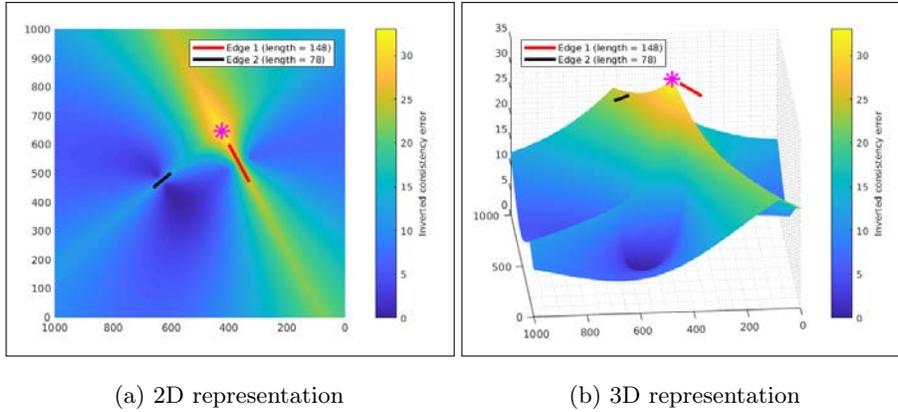


Figure 5: Representation for the error measure in [26]. Consistency error for all possible locations in a 1000×1000 grid has been computed and minimum error VP has been marked. For illustrative purposes, we have inverted results so that the minimum error correspond to the maximum value and the maximum error to 0. Therefore, the plot actually shows $\max(err(\mathbf{v}_j)) - err(\mathbf{v}_j)$.

understand this fact, in Figure 5 we plot the consistency error for all locations within a 1000×1000 frame given two ground truth edges. As we can see, the error along the longest edge is relatively small, which is undoubtedly a deficiency. Besides, it is common that, in the image, one the edges labeled as GT is clearer, longer or sharper than the other, thus gaining relevance during the dominant cluster selection process and leading to situations similar to the one included here.

To overcome this issue, our error measure is based on the probabilistic consistency measure we have used throughout the algorithm. First, we compute the Ground Truth VP using the labeled edges and, then, we measure its consistency with respect to each one of them. These are the maximum consistency values our VP candidate could get, since it would mean that the detected VP is the same as the one generated by the GT edges. Afterwards, we compute the consistency of our candidate with respect to the GT edges. Finally, the error value will be the greatest difference between the consistency values of the GT and those of our hypothesis. Note that the consistency measures for the two

GT edges are treated separately: instead of computing the mean consistency difference, we take the worst case, as we observed that it often happened that hypothesis were totally consistent with one the the edges and almost nothing with the other. The difference is normalized by the GT value for it to range between 0 and 1. Consequently, our estimation error for a candidate VP, \mathbf{v} , and a set of GT edges, $\mathcal{G} = \{E_1, E_2\}$, can be written like:

$$\xi = \max_{E_i \in \mathcal{G}} \frac{c(\hat{\mathbf{v}}, E_i, \sigma) - c(\mathbf{v}, E_i, \sigma)}{c(\hat{\mathbf{v}}, E_i, \sigma)} \quad (5)$$

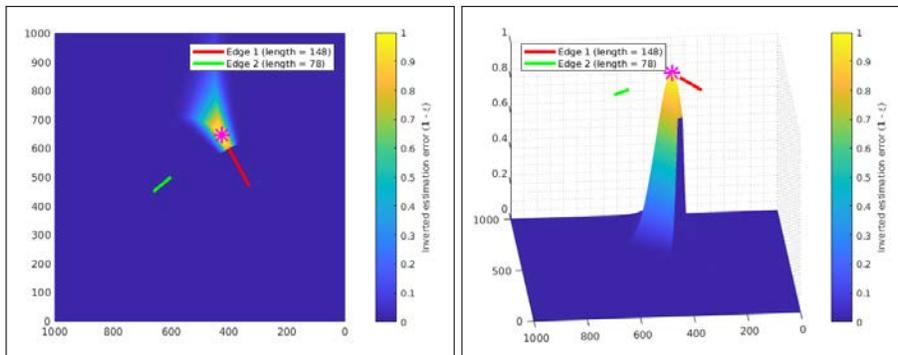
where $\hat{\mathbf{v}}$ is the Ground Truth VP, generated out of \mathcal{G} . Here, σ encodes our tolerance to a deviation from the GT position. Lower values of σ cause that only a small area around the optimal location is considered as accurate, while most of the points in the space have an error value of 1. On the contrary, higher values of σ produce a softer decay of the function when moving away from the GT position.

In order to compare both error measures, we include in Figure 6 the exact same case as for the previously introduced measure. We have selected $\sigma = 15$, which, as it can be seen in the plot, we think takes as accurate a reasonable area around the GT position, given the considered image size. The comparison between measures shows that ours (Eq. 5) is more realistic and less permissive than the former (Eq. 4).

4.1. Experimental Validation

The simulations that we have included in Figures 5 and 6 clearly illustrate the behavior of both measures. However, we have carried out an analysis on the images of the dataset so as to demonstrate that the differences between these measures are actually relevant.

VP detection error has been estimated for every image, first by applying the error measure from [26] (Eq. 4), and then by applying our proposal, thus obtaining two different rankings for the same set of images sorted by the corresponding error from the lowest to the highest. The idea here is to identify those



(a) 2D representation

(b) 3D representation

Figure 6: Representation for the error measure described by Eq. 5 with $\sigma = 15$. Consistency error for all possible locations in a 1000×1000 grid has been computed and minimum error VP has been marked. For illustrative purposes, we have inverted results, so $1 - \xi$ is actually plotted.

images for which both evaluation methods particularly differ thus yielding significantly different interpretations (i.e. different positions in both rankings) for the correctness of the detected VP (e.g. the estimated VP for a particular image could be ranked top in one case but the opposite in the other). The detailed analysis of such cases will help us to clarify which error measure models VP detection correctness the best. To derive meaning out of this analysis there is an effect that needs to be considered: our estimation error (Eq. 5) has an upper bound of 1, while the measure described in Eq. 4 has no upper bound. This will cause that the mean difference in the rankings is higher than expected, since the order of all the bad predictions could vary randomly for our probabilistic error measure. In this analysis, we have used the VP detected by our best experimental setup, which we will see in next section. However, it is important to keep in mind that the position of the VP itself is not important here, but the error value each measure assigns to it.

In Figure 7, we include a histogram and a box-plot for the rank difference. There is a mean difference of 958 positions. Having a total of 2245 images for which we have estimated the VP 10 times corresponding to 10 independent

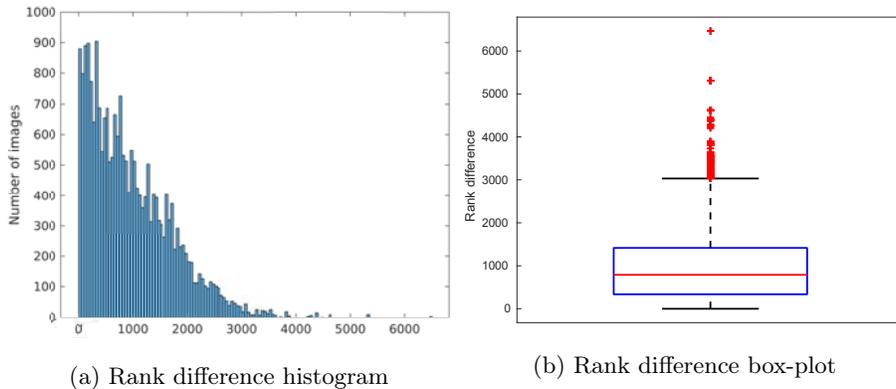


Figure 7: Rank difference when sorting images with respect to our estimation error (Eq. 5) and to the consistency error in [26] (Eq. 4).

trials (average consistency error is computed over them), we can see that there are images for which the rank difference is over 6000 positions. We include in Figure 8 the four images for which the rank difference has been the highest.

In all these cases, for which the estimated VP are objectively wrong, our estimation error is 1 (i.e. the prediction is considered to be completely wrong), while the error measure used in [26] was relatively low, thus not representing accurately the actual performance of the algorithm.

5. Experiments

In this section, we present the experimental setup we have used and we compare our results to those from previous work. We first present the dataset we have used for evaluation and then discuss about how different parameters affect our algorithm’s performance.

5.1. Ground Truth Dataset

The images we use to evaluate our algorithm’s performance belong to two different datasets, which have been manually labeled and made public in [26].

All the images have been labeled with two Ground Truth lines that can be used to determine the location of the VP. Only images where these lines are



(a) $err = 1.8410$ (Eq. 4); $\xi = 1$ (Eq. 5) (b) $err = 2.5655$ (Eq. 4); $\xi = 1$ (Eq. 5)



(c) $err = 4.5347$ (Eq. 4); $\xi = 1$ (Eq. 5) (d) $err = 5.2124$ (Eq. 4); $\xi = 1$ (Eq. 5)

Figure 8: Four example images for which the rank difference was particularly large when using the two different error measures.

visible and with a single dominant VP have been included. This means that images with VPs formed by succession of objects (not proper lines) or by parallel curves are excluded, as well as images with two or more vanishing directions. Images were resized so that their longer side's length is 500 pixels, and only images whose VP lie within a 1000×1000 area frame (with the image centered on it) are finally considered.

This process has been done for two sets of images: the first is a subset of

the landscape category in AVA dataset [18], while the second consists in images retrieved from Flickr website.

Once the images that did not fulfill the requirements were excluded, we end up with 1316 images in the AVA dataset and 959¹ images in the Flickr dataset. As in [26], we present our results for both datasets separately.

5.2. Experimental Setup

On trying to find the best working point for our algorithm, we have tested it with different values for the parameters that may influence performance. The default setup for our experiments is:

- Minimum edge length l is set to 40 pixels. Performance with respect to variations in this parameter has not been tested, as [26] showed this to be the optimal value for both databases.
- Minimum distance from both endpoints of an edge to image borders is $b = 20$.
- Minimum edge angle with respect to the horizontal axis, in degrees, is selected to be $\phi = 0.5$, which we find small enough not to lose any valuable edge.
- T-Linkage clustering is carried out with $m = 10000$ VP hypothesis.
- For the strength measure (see Eq. 2), we use $\tau = 1$.
- When refining clusters, the outliers consistency threshold is set to $\eta = \frac{1}{\sqrt{2\pi}\sigma}e^{-0.5}$, which is the consistency at one standard deviation away from the mean (see Eq. 1).
- The edge extraction error for measuring consistency in Eq. 1 is $\sigma = 3$.

The reported performance for every experiment is the result of taking the average consistency error over 10 independent trials.

¹Only 929 images were available online when this evaluation was done.

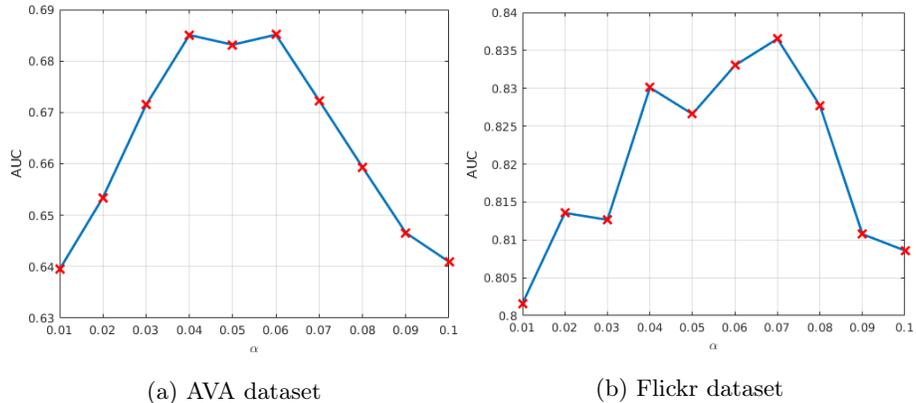


Figure 9: AUC for the CDF obtained our error measure (Eq. 5) for different α values. Please note that the y-axis is not the same in both plots.

5.2.1. Single-Threshold Contour Splitting

First, we compare results when using different single values for the contour splitting parameter α . This parameter implies a trade-off between the length of the extracted edges and their *veracity*—their similarity with respect to the actual contours present in the image—: lower values will generate shorter and more “realistic” edges, as deviation from straight contours will be less tolerated and will result in more splits. On the contrary, higher α values will create longer edges that are less representative of the actual contours in the image, as curvy contours are more likely to be mapped to straight lines. Additionally, note that short edges are filtered out *a posteriori* for noise reduction.

In Figure 9, we plot the Area Under the Curve (AUC) for the CDF obtained with different α values. We can observe that the maximum AUC for AVA dataset is 68.51 %, while for Flickr dataset we obtain an AUC of 83.65 %. This shows that, in general, VPs are harder to detect for images in AVA dataset. Also, the optimal value is $\alpha_{opt} = 0.06$ for AVA dataset and $\alpha_{opt} = 0.07$ for Flickr dataset. It is worth mentioning that the optimal value for both datasets in [26] was $\alpha = 0.05$, which, though different, is quite close to ours. In the next section, we use this information to propose different combinations of α values.

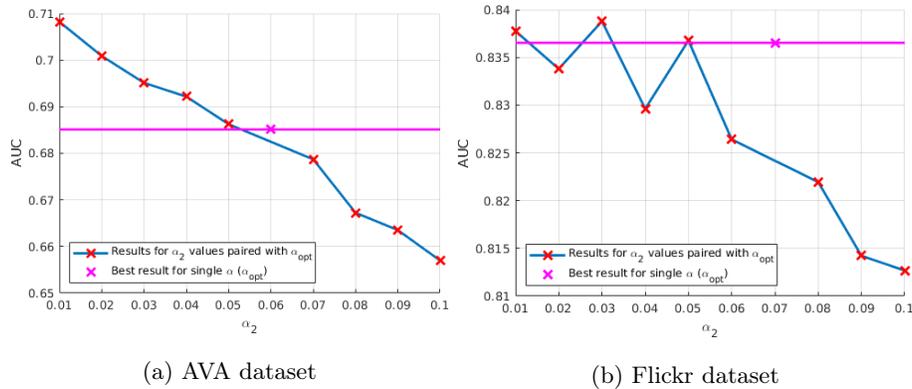


Figure 10: AUC for the CDF obtained using our error measure (Eq. 5) for different combinations of α values which include the optimal single value α_{opt} and a secondary α_2 . Please note that the y-axis is not the same in both plots.

5.2.2. Multi-Threshold Contour Splitting

Now, we compare the results when using our multi-threshold approach, which combines the edges resulting from the contour splitting procedure when adopting different α values. The combinations that we have tested are based on the optimal values α_{opt} obtained for the single threshold strategy. Specifically, for each dataset, we have paired the optimal value α_{opt} with other secondary values α_2 in the range $[0.01, 0.1]$, with a resolution of 0.01.

Results for all these experiments can be seen in Figure 10. For comparison purposes, we also include the optimal AUC value obtained when relying on a single threshold (i.e. α_{opt}) for each dataset. It is noticeable that, in both cases, the best results are obtained for a pair of values formed by the optimal single value and a lower one. When adding edges extracted with lower α values, we are including edges that were not previously considered—it is worth reminding that if the exact edge appears for several α 's we include it only once—but whose correspondence with the actual contours in the image is higher. Contrarily, as previously explained, higher α values may tend to produce oversized edges that are somehow more “artificial” and that tend to oversimplify the actual contour structure present in the images, what may explain the obtained worse results

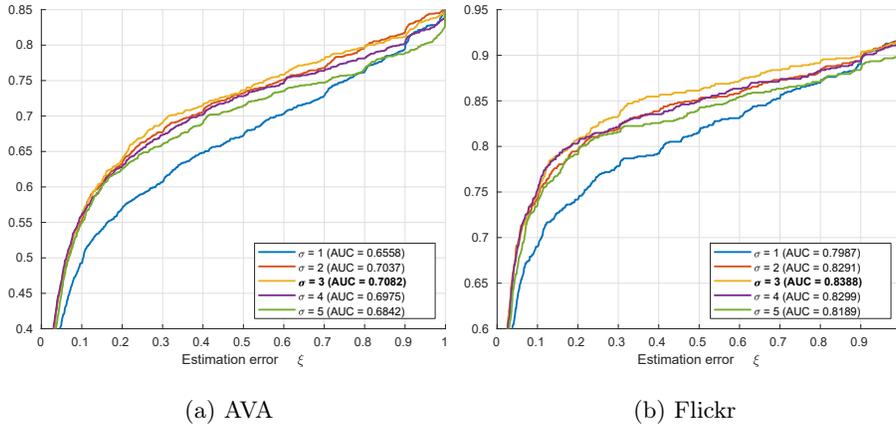


Figure 11: CDFs obtained using our error measure (Eq. 5) for different values of the edge extraction error σ .

with respect to the single optimal α .

Using values below the optimal single one results in the inclusion of many contours that were present in the image, but whose conversion into edges did not represent them in the best way, i.e. edges that should have been split but were not. Furthermore, as we filter edges by length after extraction, we should not be afraid of oversplitting contours when using α 's that may be too low. The better results obtained when combining the optimal with lower α 's prove that the additional edges included, which match the original contours better, are complementary to those from the single optimal splitting threshold, thus providing relevant information about the dominant VP location.

5.2.3. Edge extraction error

Here, we compare results with respect to σ , which encodes the uncertainty of edge extraction in Eq. 1.

For this experiment edges are extracted by adopting the optimal multi-threshold contour splitting strategy previously identified for each dataset (i.e. using the optimal pair of α values). Performance results are included in Figure 11. We can see that the optimal value for both datasets is the default one, $\sigma = 3$ pixels. In [24], where they defined the probabilistic consistency measure,

	AVA dataset	Flickr dataset
Zhou et al. [26]	0.655062 \pm 0.025683	0.812737 \pm 0.025087
Ours	0.708231 \pm 0.024560	0.838788 \pm 0.023647

Table 1: AUC for the CDF (with a 95 % confidence interval) obtained using our error measure (Eq. 5) for the method in [26] and ours.

	AVA dataset	Flickr dataset
Zhou et al. [26]	0.345021 \pm 0.022162	0.187300 \pm 0.021993
Ours	0.291836 \pm 0.020709	0.161252 \pm 0.020075

Table 2: Average estimation error (with a 95 % confidence interval) obtained using our error measure (Eq. 5) for the method in [26] and ours.

they used a value $\sigma = 1$ pixel for their experiments. The need for a greater σ here is a possible consequence of the differences in the content of the photos: while ours feature natural landscapes, theirs were taken in urban environments.

5.3. Results

In this section, we report our final results for both databases and compare them with the ones obtained using the method in the already mentioned paper [26]. To the best of our knowledge, it is the only one where the VP detection problem for natural landscape images was dealt with before. Some examples of correctly and incorrectly detected Vanishing Points, generated using our algorithm, can be seen in Figure 13.

In Tables 1 and 2, we include the results that we have obtained using the best setup for our technique. Our method outperforms, for both datasets, the previously proposed one in [26]. Regarding the AUC for the CDF, presented in Table 1, there is an improvement of 5.32 % for AVA dataset and of 2.61 % for Flickr dataset. The increase is greater for the images in AVA dataset, for which the detection results are generally worse, so we could conclude that our algorithm deals better with hard examples. The results for the mean estimation error, contained in Table 2, confirm the improvement that our method provides.

It can be observed that the differences between the results from both methods are statistically significant for the case of AVA dataset, while there exist some overlap on the confidence intervals for Flickr dataset. From our point of view, this is probably a consequence of the fewer number of images included in Flickr dataset (929 images) with respect to those in AVA (1316 images), since the width of the confidence intervals is inversely proportional to the square root of the number of images in the sample.

For completeness, we include CDFs for our algorithm in Figure 12. It is worth noting that, for instance, there are almost 70 % and 85 % of images in AVA and Flickr, respectively, for which our method present an estimation error below 0.3. Easier examples are for sure included in that percentiles and detected accurately by both methods, but it is the correct detection of harder examples what enables improvement above the level of the previously proposed technique. Also, in Figure 9, we can see that when using our system with a single α threshold equal to that found optimal in [26] ($\alpha = 0.05$), our AUC results for AVA and Flickr datasets are 68.32 % and 82.66 %, respectively, showing that even in such situation our method outperforms the reference one.

5.4. Ablation Analysis and Computation Time Measures

Finally, we present an ablation analysis in order to show the influence of each one of our added modules in the final accuracy improvement. We refer to our detection method without carrying out the cluster refinement procedure as *Ablated System 1*. Furthermore, we have also measured how detection results are affected by the elimination of our additional edge filtering, i.e. the filtering of horizontal edges as well as those close to image borders, in addition to the removal of the cluster refinement procedure. We refer to this second version of our system as *Ablated System 2*, which still introduces a novel and improved clustering procedure (i.e. T-Linkage) compared to the baseline. Results for AVA and Flickr datasets are included in Figure 12.

In summary, our main contributions to the VP detection algorithm are: the use of T-Linkage as clustering algorithm, filtering out more edges which are

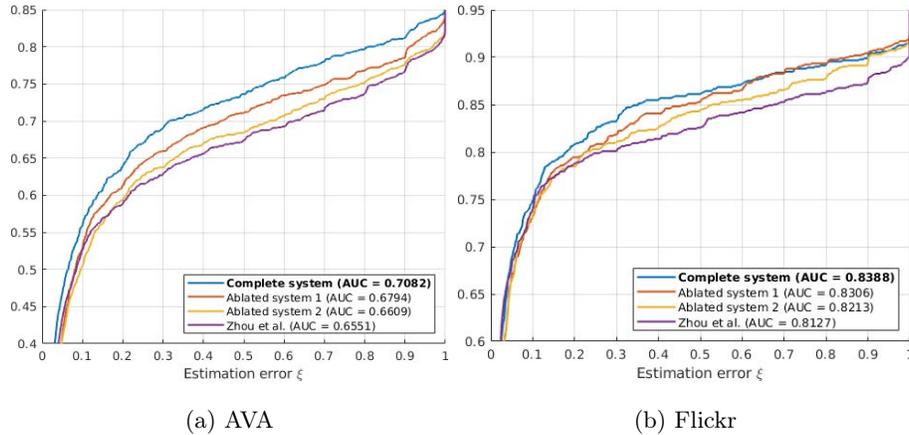


Figure 12: AUC for the CDF obtained with our complete VP detection system, the two ablated versions, and the baseline one.

Stage	Mean	Standard deviation
1. UCM computation	148.26	26.17
2. Edge extraction and filtering	2.84	1.00
3. Clustering and refinement	11.05	4.19
Total	162.14	28.55

Table 3: Computation time (in seconds) for our VP detection algorithm.

likely to be noisy, and carrying out a cluster refinement procedure. This ablation analysis proves that all of them result in an improvement of the detection results with respect to the previous reference algorithm.

Additionally, we have measured the computation time ² of our VP detection algorithm. Results are detailed in Table 3. It can be observed that the computation of the Ultrametric Contour Maps takes most of the time (i.e. 91.44% of the total computation time). For this reason, although our algorithm may be slightly more complex than others, we consider that the associated increase in

²Reported measures have been obtained when executing all the algorithms in a computer equipped with an Intel i7 3770K processor and a 32 GB RAM.

computation time is negligible.

6. Conclusions

In this paper, we present an improved algorithm for Vanishing Point detection in natural landscape images. Our approach is based on the multi-threshold extraction of edges in images, combining several representations of the same image, and the grouping of edges using the RANSAC-based clustering algorithm T-Linkage. Then, a refinement procedure is carried out and a final VP candidate is chosen to be the dominant one. Our technique has been proven to perform better than others that previously tried to solve the VP detection problem on the datasets we use, being all our contributions to the algorithm fruitful.

Additionally, we perform an analysis on the difficulties that arise when reporting VP detection results as well as the limitations of some error measures that have been used for this purpose before. To overcome these limitations, we present a novel error measure, which is based on a probabilistic consistency measure, and that can be tuned depending on how strict we want to be when evaluating our VP detection results. Simulations and experimental validations are included to support our proposal.

In this work, we have only tested our algorithm on images including a single VP. In the future, we plan to adapt it to be able to detect whether linear perspective is present or not in the image, and if so, how many vanishing directions are there in the image. As an additional line of research, we are also exploring the application of VP detection to aesthetics assessment, where our algorithm could be used to detect VPs and to enable the estimation of different aesthetics related descriptors based on their location. Furthermore, we are currently experimenting with new detection solutions based on deep learning, which are undoubtedly promising and challenging. One of the main aspects to be considered regarding this approach is the identification of the optimal loss function for the detection problem. Some previous proposals apply a grid to the image and try to solve a classification problem [5], while others employ a two-dimensional

regression approach [19]. However, in our work we show why the correctness of a detection result is not accurately represented by the Euclidean distance, so the loss function should reflect this complex aspect regarding the evaluation of the results, which is our main goal nowadays.

7. Acknowledgements

The work leading to these results has been supported by the Spanish Ministry of Economy, Industry and Competitiveness through the ESITUR (MINECO, RTC-2016-5305- 7), CAVIAR (MINECO, TEC2017-84593-C2-1-R), and AMIC (MINECO, TIN2017-85854-C4-4-R) projects (AEI/FEDER, UE). We also gratefully acknowledge the support of NVIDIA Corporation.

References

- [1] Antone, M.E., Teller, S., 2000. Automatic recovery of relative camera rotations for urban scenes, in: 2000 IEEE Conference on Computer Vision and Pattern Recognition, pp. 282–289. doi:[10.1109/CVPR.2000.854809](https://doi.org/10.1109/CVPR.2000.854809).
- [2] Arbelaez, P., Maire, M., Fowlkes, C., Malik, J., 2011. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33, 898–916. doi:[10.1109/TPAMI.2010.161](https://doi.org/10.1109/TPAMI.2010.161).
- [3] Barnard, S.T., 1983. Interpreting perspective images. *Artificial intelligence* 21, 435–462. doi:[10.1016/S0004-3702\(83\)80021-6](https://doi.org/10.1016/S0004-3702(83)80021-6).
- [4] Bazin, J.C., Seo, Y., Demonceaux, C., Vasseur, P., Ikeuchi, K., Kweon, I., Pollefeys, M., 2012. Globally optimal line clustering and vanishing point estimation in manhattan world, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 638–645. doi:[10.1109/CVPR.2012.6247731](https://doi.org/10.1109/CVPR.2012.6247731).
- [5] Borji, A., 2016. Vanishing point detection with convolutional neural networks. SUNw workshop, CVPR 2016 .

- [6] Canny, J., 1986. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8, 679–698. doi:[10.1109/TPAMI.1986.4767851](https://doi.org/10.1109/TPAMI.1986.4767851).
- [7] Coughlan, J.M., Yuille, A.L., 1999. Manhattan world: compass direction from a single image by bayesian inference, in: *Proceedings of the Seventh IEEE International Conference on Computer Vision*, pp. 941–947. doi:[10.1109/ICCV.1999.790349](https://doi.org/10.1109/ICCV.1999.790349).
- [8] Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* 24, 381–395. doi:[10.1145/358669.358692](https://doi.org/10.1145/358669.358692).
- [9] von Gioi, R.G., Jakubowicz, J., Morel, J.M., Randall, G., 2010. LSD: A fast line segment detector with a false detection control. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32, 722–732. doi:[10.1109/TPAMI.2008.300](https://doi.org/10.1109/TPAMI.2008.300).
- [10] Heuvel, F.A.V.D., 1998. Vanishing point detection for architectural photogrammetry. *International archives of photogrammetry and remote sensing* 32, 652–659.
- [11] Kluger, F., Ackermann, H., Yang, M.Y., Rosenhahn, B., 2017. Deep learning for vanishing point detection using an inverse gnomonic projection, in: *Pattern recognition: Proceedings of the 39th German Conference, GCPR 2017*, pp. 17–28. doi:[10.1007/978-3-319-66709-6_2](https://doi.org/10.1007/978-3-319-66709-6_2).
- [12] Kong, H., Audibert, J.Y., Ponce, J., 2009. Vanishing point detection for road detection, in: *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 96–103. doi:[10.1109/CVPR.2009.5206787](https://doi.org/10.1109/CVPR.2009.5206787).
- [13] Košecká, J., Zhang, W., 2002. Video compass, in: *Proceedings of the 7th European Conference on Computer Vision-Part IV*, pp. 476–490. doi:[10.1007/3-540-47979-1_32](https://doi.org/10.1007/3-540-47979-1_32).

- [14] Lezama, J., v. Gioi, R.G., Randall, G., Morel, J.M., 2014. Finding vanishing points via point alignments in image primal and dual domains, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition, pp. 509–515. doi:[10.1109/CVPR.2014.72](https://doi.org/10.1109/CVPR.2014.72).
- [15] Magri, L., Fusiello, A., 2014. T-linkage: A continuous relaxation of j-linkage for multi-model fitting, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3954–3961. doi:[10.1109/CVPR.2014.505](https://doi.org/10.1109/CVPR.2014.505).
- [16] Mirzaei, F.M., Roumeliotis, S.I., 2011. Optimal estimation of vanishing points in a manhattan world, in: 2011 International Conference on Computer Vision, pp. 2454–2461. doi:[10.1109/ICCV.2011.6126530](https://doi.org/10.1109/ICCV.2011.6126530).
- [17] Moghadam, P., Starzyk, J.A., Wijesoma, W.S., 2012. Fast vanishing-point detection in unstructured environments. *IEEE Transactions on Image Processing* 21, 425–430. doi:[10.1109/TIP.2011.2162422](https://doi.org/10.1109/TIP.2011.2162422).
- [18] Murray, N., Marchesotti, L., Perronnin, F., 2012. Ava: A large-scale database for aesthetic visual analysis, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2408–2415. doi:[10.1109/CVPR.2012.6247954](https://doi.org/10.1109/CVPR.2012.6247954).
- [19] Shuai, Y., Tiantian, Y., Guodong, Y., Zize, L., 2017. Regression convolutional network for vanishing point detection, in: 2017 32nd Youth Academic Annual Conference of Chinese Association of Automation (YAC), pp. 634–638. doi:[10.1109/YAC.2017.7967487](https://doi.org/10.1109/YAC.2017.7967487).
- [20] Shufelt, J.A., 1999. Performance evaluation and analysis of vanishing point detection techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21, 282–288. doi:[10.1109/34.754631](https://doi.org/10.1109/34.754631).
- [21] Tardif, J.P., 2009. Non-iterative approach for fast and accurate vanishing point detection, in: 2009 IEEE 12th International Conference on Computer Vision, pp. 1250–1257. doi:[10.1109/ICCV.2009.5459328](https://doi.org/10.1109/ICCV.2009.5459328).

- [22] Toldo, R., Fusiello, A., 2008. Robust multiple structures estimation with J-Linkage, in: Computer Vision - ECCV 2008, pp. 537–547. doi:[10.1007/978-3-540-88682-2_41](https://doi.org/10.1007/978-3-540-88682-2_41).
- [23] Wildenauer, H., Hanbury, A., 2012. Robust camera self-calibration from monocular images of manhattan worlds, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2831–2838. doi:[10.1109/CVPR.2012.6248008](https://doi.org/10.1109/CVPR.2012.6248008).
- [24] Xu, Y., Oh, S., Hoogs, A., 2013. A minimum error vanishing point detection approach for uncalibrated monocular images of man-made environments, in: 2013 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1376–1383. doi:[10.1109/CVPR.2013.181](https://doi.org/10.1109/CVPR.2013.181).
- [25] Zhai, M., Workman, S., Jacobs, N., 2016. Detecting vanishing points using global image context in a non-manhattan world, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5657–5665.
- [26] Zhou, Z., Farhat, F., Wang, J.Z., 2017. Detecting dominant vanishing points in natural scenes with application to composition-sensitive image retrieval. IEEE Transactions on Multimedia 19, 2651–2665. doi:[10.1109/TMM.2017.2703954](https://doi.org/10.1109/TMM.2017.2703954).



Figure 13: First row: examples of correct VP detection. Second row: examples of wrong VP detection.

A Multi-Threshold Approach and a Realistic Error Measure for Vanishing Point Detection in Natural Landscapes

Álvaro García-Faura, Fernando Fernández-Martínez, Ricardo Kleinlein, Rubén San-Segundo and Fernando Díaz-de-María

Highlights:

- Novel methodology for improved vanishing point detection in landscape images
- Improved edge extraction process by combining different representations of an image
- New cluster refinement method for discarding weak VP hypothesis
- New probabilistic error measure for more robust vanishing point detection
- Qualitative and quantitative analysis validating the reliability of the new measure

