Joint Vanishing Point Extraction and Tracking

Till Kroeger[1] Dengxin Dai[1] Luc Van Gool[1,2]
[1]Computer Vision Laboratory, D-ITET, ETH Zurich
[2]VISICS, ESAT/PSI, KU Leuven
{kroegert, dai, vangool}@vision.ee.ethz.ch

Abstract

We present a novel vanishing point (VP) detection and tracking algorithm for calibrated monocular image sequences. Previous VP detection and tracking methods usually assume known camera poses for all frames or detect and track separately. We advance the state-of-the-art by combining VP extraction on a Gaussian sphere with recent advances in multi-target tracking on probabilistic occupancy fields. The solution is obtained by solving a Linear Program (LP). This enables the joint detection and tracking of multiple VPs over sequences. Unlike existing works we do not need known camera poses, and at the same time avoid detecting and tracking in separate steps. We also propose an extension to enforce VP orthogonality. We augment an existing video dataset consisting of 48 monocular videos with multiple annotated VPs in 14448 frames for evaluation. Although the method is designed for unknown camera poses, it is also helpful in scenarios with known poses, since a multi-frame approach in VP detection helps to regularize in frames with weak VP line support.

1. Introduction

A vanishing point (VP) is the point of convergence of a set of parallel lines in the imaged scene under a projective transformation. Man-made structures often consist of geometric primitives, such as multiple sets of parallel or orthogonal planes and lines in the scene. Because of this, the detection of projected VPs in images provides strong cues for the extraction of knowledge about the unknown 3D world structure. VPs can often be further constrained to mutual orthogonality, due to the preference of right angles in man-made structures. Detected VPs have been used as a low-level input to many higher-level computer vision tasks, such as 3D reconstruction [9, 14], autonomous navigation [23], camera calibration [12, 33] and pose estimation [15, 22].

Many applications, which take video sequences or unordered image sets as input, require VP estimates in every frame and VP identities across views or frames. Usually, when this is needed, the camera pose is assumed to be known for every frame [1, 13], thereby rendering the VP association across images simple, or separate steps for VP detection and tracking (particle filters [23, 25], greedy assignment [11]) are used. Since pose knowledge can only be obtained through expensive odometry or external motion measurements, it will often not be available. Separate VP detection and tracking often results in missed detections or loss and re-initialization of VP tracks due to weak line support in some frames. Even in the case of known poses, joint detection over multiple frames benefits from integrating image evidence of many frames. Joint reasoning over sequences is particularly useful if long-term VP identities are required, and re-initializations are expensive.

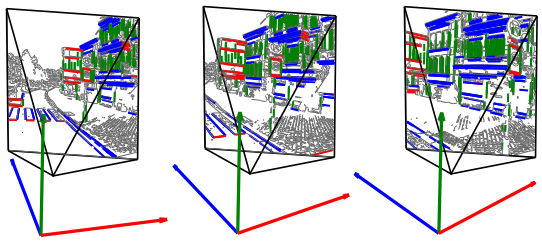

Figure 1: Tracked vanishing directions are shown together with associated imaged line segments in three frames of a sequence. This and the subsequent figures are best viewed in color.

To the best of our knowledge, no method exists that jointly extracts multiple VPs in all frames of a video with unknown camera motion. Our contributions are twofold:

1. Method. We propose the first algorithm for the joint VP extraction over all images of a sequence with unknown camera poses. We borrow from recent advances in multi-target tracking [6, 35] and model the problem as a variant of a network-flow tracking problem. We compute line segments in each frame and discretize the set of possible VPs on a *probabilistic spherical occupancy grid*. Line-VP association probabilities and VP transition probabilities are converted into an acyclic graph for joint VP extraction and tracking. VPs are extracted by Linear Programming (LP).

2. Dataset. As the field lacks a dataset for the evaluation

of VP extraction in videos, we augmented the Street-View video dataset of [17] with annotated VPs, which will be publicly available. We evaluate our approach on this dataset using established multi-object tracking metrics [7, 19] for unknown camera poses. We chose this dataset because camera poses are available for all frames, which enables one additional experiment: We evaluate the improvement of our algorithm when camera pose information is incorporated. Since our method also works for single frames and orthogonal VPs, we compare to a recent method[29] for VP detection in Manhattan Scenes on the York Urban Dataset [10].

The paper is organized as follows: §2 introduces our VP parametrization. §3 describes the method, with LP formulation in §3.1, and score modeling in §3.2. We evaluate in §4 and conclude in §5 with a discussion of future work.

Related Work

VP extraction has been studied extensively in Computer Vision. The most relevant recent works can be categorized according to several algorithmic design choices:

Input: While some approaches start directly from continuous image gradients or texture [23, 25, 27] and thresholded edges images [30], most works rely on short line segments [2, 3, 12, 13, 21, 26, 29, 33], or full lines [8, 11]. If the 3D geometry is known, surface normals can be used [28].

Accumulator Space: Intersections of imaged lines can be computed in the original (unbounded) image space [2, 11, 26, 27, 29, 34] or on a (bounded) Gaussian unit sphere, first proposed in [3] and used in [1, 5, 12, 13, 15, 20, 21, 22, 24, 28]. We use the latter approach, explained in §2. It allows for easy discretization [3, 20, 21]. [18] proposes a line parametrization in parallel coordinates to extract VPs.

Line-VP Consistency and VP Refinement: Consistency between an estimated VP and image lines can be computed directly by measuring line endpoint distances in the image [2, 5, 13, 18, 29], angular differences in the image [10, 26], with explicit probabilistic modeling of the line endpoint errors [34], or with angles between normals of interpretation planes in the Gaussian sphere, used by us and [20, 21, 24]. VP computation or refinement with given associated lines is done via Hough voting and non-maximum suppression [20, 21, 23, 31, 24], solving a quadratic program [2], implicitly in an EM setting [1, 27, 29, 34], or by linear least-squares, as in this paper and [15].

Solution: Several different methods exist for combining input, accumulator space and line-VP consistency measures into a final extraction solution. If no discretization of the accumulator space is attempted, solutions are found with efficient search [10, 26], direct clustering [18, 29], multi-line RANSAC [5, 33], EM procedures [1, 13, 15, 27, 34], or MCMC inference [28]. With a discretized accumulator space solutions are found by voting schemes [20, 21, 23, 24] or inference in graphical models [2, 30].

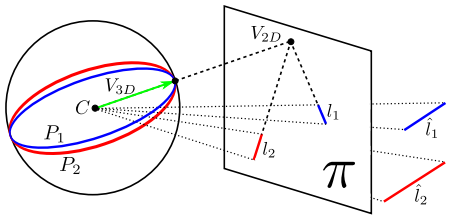

Figure 2: Imaged line segments l_1,l_2 of scene parallel lines $\hat{l}_1$, $\hat{l}_2$ form a VP V_{2D} on the image plane π. The same VP can be parametrized as the vector V_{3D} pointing towards the intersection of interpretation planes P_1, P_2 of lines l_1, l_2 and the unit sphere.

Camera Calibration and VP Orthogonality: Some VP extraction methods assume known internal camera calibration [11, 13, 20, 23, 24, 25]. Others do not need calibration [2, 15, 21, 26, 34]. Internal parameters can be estimated from extracted VPs [8, 12, 33]. VPs have also been used for estimation of external camera parameters: orientation of camera to scene [4, 15], 3D shape to camera [3], and as additional constraints for full camera poses [22]. Often, further scene-dependent VP constraints are included: mutual VP orthogonality (Manhattan World) [10, 11, 13, 33], sets of mutually orthogonal VPs [28, 16], with a shared vertical VP (Atlanta World) [2, 27].

Multi-View Extraction and VP Tracking: [1] solves multi-view VP extraction by Hough voting with EM refinement, but requires known camera poses. [13] extracts orthogonal VPs independently in multiple views, and integrates information across views by using SfM (Structure-from-Motion) camera pose estimates. [11, 16] explicitly track orthogonal VPs in videos. [11] extracts VPs separately in each frame, and greedily links VPs across frames. [16] uses a multi-target tracking approach to link multiple hypothesized sets of mutually orthogonal VPs, but, in contrast to this work, requires pre-processing to extract candidates. [23, 25] track VPs for road direction finding. One finite horizontal VP, corresponding to the heading direction, is extracted in each frame and tracked using particle filters.

In contrast to these approaches, in our method we find a jointly optimal solution without any knowledge about the camera pose, or number or relative directions of the VPs.

2. VP Representation

A 2D VP is the intersection of (the unbounded continuation of) two 2D line segments, imaged from two 3D scene-parallel lines. In Fig. 2 imaged line segments l_1, l_2 of scene parallel lines $\hat{l}_1$, $\hat{l}_2$ form a VP V_{2D} on the image plane π. The 2D VP may lie inside or outside the image frustum of π, or at infinity, in cases when imaged lines remain parallel.

An alternative parametrization, proposed in [3], models VP locations on the unit sphere. A point $\tilde{x}$ in homogeneous image coordinates is normalized by $x = K^{-1}\tilde{x}$, with

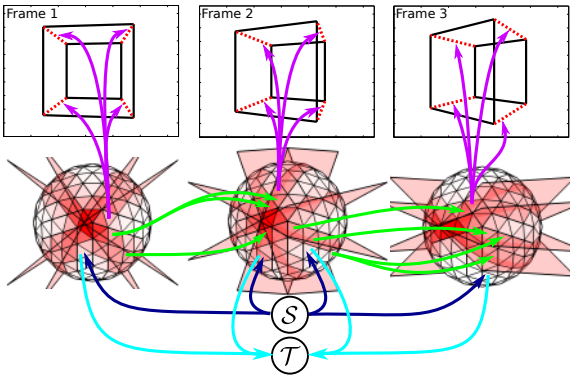

Figure 3: Combined figure for § 2.1 and § 3. For § 2.1: interpretation planes on a discretized sphere for a rotating cube. For visualization purposes only four planes belonging to one horizontal VP are drawn. For § 3: directed acyclic graph with, 1) arcs between VP bin and $\mathcal{S}$tart and $\mathcal{T}$erminal node (blue,cyan), 2) transition arcs from each VP bin to all bins in following frame (green), 3) line association arcs from each VP bin to all line segments at time t (magenta). For visualization only a subset of all arcs is drawn.

K the camera calibration matrix. A plane P is spanned by the center of projection at $[0,0,0]^T$, and the endpoints x_1, x_2 in normalized homogeneous image coordinates of an imaged line segment l. The plane is computed as $P = x_1 \times x_2 / (\|x_1\| \, \|x_2\|)$, and is called *interpretation plane*. For line segments l_1, l_2, interpretation planes P_1, P_2 are shown as intersection circles of the planes and the unit sphere in Fig. 2. The VP is given as their intersection on the sphere: $V_{3D} = \pm P_1 \times P_2$. In the following we use *vanishing direction* and *vanishing point* interchangeably. If more than two planes are detected, the VP is given by the least-squares solution to a system of incidence equations:

$$\sum\nolimits_i < P_i, V_{3D} >= 0 \; . \tag{1}$$

2.1. VP Discretization

In order to use the proposed parametrization for tracking we need to discretize the solution space of possible VPs. Fig. 3 illustrates the chosen discretization: a simple triangular tessellation of the sphere by iterative subdivisions of the faces of an icosahedron. Three frames of a sequence of a rotating cube are shown (top row). For visualization purposes four line segments belonging to one horizontal VP are drawn in red. With known internal camera calibration K, four interpretation planes are computed and plotted together with the discretized unit sphere (bottom row). The red color strength illustrates the likelihood of a VP in each cell of the spherical grid. This is determined by summing up all line-VP consistency scores as described later in Eq. (14). Since the cube is rotating around a vertical axis, the horizontal VP rotates accordingly, which can be seen by the change in VP likelihoods on the spherical grid. Note that the spherical grid itself does not rotate, only the planes and their intersections on the sphere follow the cube's rotation. While in this example correct VP-line associations have been selected manually, our proposed method will have to extract multiple VP tracks and VP-line associations jointly.

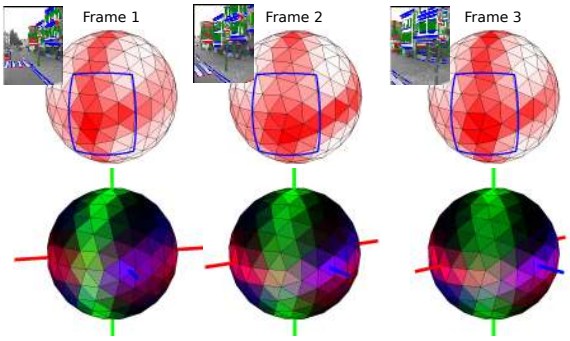

Figure 4: Illustration why aggregated occupancy probabilities are insufficient. Same example as in Fig. 1. Top: contribution to VP bins from all interpretation planes regardless of VP association. Color strength indicates increasing VP likelihood. Note the wide (but incorrect) peak within the image frustum outlined in blue. Bottom: after optimization we know the correct VPs, and color all plane contributions following their association to the red, green and blue VP. The misleading peak in the image frustum (top row) originates from overlapping interpretation planes of the true VPs.

3. Proposed Algorithm

The proposed VP representation and discretization can now be used in a directed acyclic graph, similar to flow networks in multi-target tracking-by-detection. Such graphs for probabilistic multi-target tracking were first proposed in [35], where object detections are linked across time through pairwise object transition arcs. Transition probabilities indicate which detections at different times capture the same object. This technique has been very successful in multi-target tracking, since it allows association of detection evidence jointly in time and space. [6] extends this by lifting the need for object hypotheses, and operates on a discretized ground plane with occupancy probabilities in each grid cell. Instead of object transitions, grid transitions are encoded in the graph. The task is solved by k-shortest path search through the graph with Dynamic or Linear Programming.

In order to avoid a separate VP detection step we follow the second approach, and consider the discretized sphere as an *occupancy sphere*, where probabilistic evidence in each bin indicates the likelihood for a VP. However, in contrast to [6], we cannot simply aggregate all occupancy evidence for a VP bin, since one interpretation plane could give its evidence to several, mutually contradictory bins along a great circle on the sphere. In general, there will usually be a greater number of line intersections within the image frustum than outside of it. Because of this VPs can be hallu-

cinated in the image frustum if each line is allowed to vote for all bins along the intersection of interpretation plane and sphere. This is illustrated in Fig. 4 (top row), where evidence (Eq. 14) from all interpretation planes for multiple VPs is aggregated, similar to Fig. 3 for one VP. Line segments to horizontal (red, blue) and vertical (green) VPs join in a wide, but incorrect, peak in the occupancy probability within the image frustum. Weaker, but sharper and temporally more stable peaks, corresponding to true VPs, are often lost in this noise. Fig. 4 (bottom row) visualizes this: we colored the interpretation plane contributions according to their (correct) VP association. It can be seen that the peak within the image frustum for aggregated occupancy probabilities does not correspond to any real VP.

Therefore, we do not want to aggregate occupancy probabilities, but need to enforce that each line segment and interpretation plane is assigned to maximally one VP bin. To achieve this, we keep the association between VP bins and interpretation planes as free variable in the joint VP extraction and tracking framework. This approach is related to the *Uncapacitated Facility Location* problem for VP extraction [2]. However, [2] focuses on single-frame extraction of multiple sets of orthogonal VPs, while we apply this idea to joint detection and tracking over time, and add optional constraints to enforce orthogonality in VP locations.

3.1. Linear Program Formulation

Overview: Solving this tasks requires deciding, firstly, which VP bins are active, secondly, which line segments are uniquely assigned to each VP, and, thirdly, where active VP bins are continued over time. Since these decisions are interdependent, we map this problem into a graph, a variant of flow-cost networks [35], as visualized in Fig. 3, to enable a joint solution. In this graph arcs between nodes define binary decision variables, which are activated or deactivated depending on whether associations between VP bins over time and VPs to line segments are made or not. The graph is structured such that each VP track starts at the starting node, traverses smoothly connected VP bins over time, collects uniquely assigned line evidence in each traversed frame, and ends at the terminal node. Tracks are constrained not to overlap, and to enclose a constant angle to each other active VP, with optional perpendicularity.

LP Formulation: For an image sequence of T frames, with I_t line segments at time t, we denote the interpretation plane for line segment i at time t as $P_{i,t}$. $V_{j,t}$ denotes the unit normal of VP bin j out of J possible bins at time t.

The graph is constructed as follows (See Fig. 3): for each bin $V_{j,t}$ we have an arc from the starting node $\mathcal{S}$ and an arc to the terminal node $\mathcal{T}$ with associated cost $C_s(j,t)$, $C_e(j,t)$, respectively. For each bin $V_{j,t}$ we have transition arcs to all bins $V_{j',t+1}, \forall j' \in J$ in the following frame, with associated cost $C_t(j,t,j')$. For each bin $V_{j,t}$ we define a unary cost $C_u(j,t)$. For each bin $V_{j,t}$ we have line association arcs to every interpretation plane $P_{i,t}$, with associated score $S_l(j,i,t)$. The solution to our problem is then given by the set of shortest paths (i.e. with lowest aggregate scores plus costs) from nodes $\mathcal{S}$ to $\mathcal{T}$. Inspired by [35], we model the problem such that only image evidence (i.e. line segments) encourages active VP tracks with scores S_l while all other terms C_s, C_e, C_u, C_t inhibit them as costs.

We sum all scores and costs for the objective function f:

$$f(\boldsymbol{\lambda}) = \sum_t^T \sum_j^J \Bigg[\left(\sum_i^{I_t} \lambda_l(j,i,t) \cdot S_l(j,i,t) \right) + \left(\sum_{j'}^{J} \lambda_t(j,t,j') \cdot C_t(j,t,j') \right) + \lambda_b(j,t) \cdot C_u(j,t) + \lambda_s(j,t) \cdot C_s(j,t) + \lambda_e(j,t) \cdot C_e(j,t) \Bigg] , \quad (2)$$

where $\boldsymbol{\lambda} = [\lambda_l, \lambda_t, \lambda_b, \lambda_s, \lambda_e]$ are binary variables, indicating active ([l]ine, [t]ransition, [b]in activation, [s]tart, [e]xit) arcs. Since a VP bin can only be active with an active outgoing transition or exit arc we replace $\lambda_b(j,t) = \lambda_e(j,t) + \sum_{j'} \lambda_t(j,t,j')$ in the optimization.

The LP solution is given by:

$$\boldsymbol{\lambda}^* = \text{argmin}_{\boldsymbol{\lambda}} \; f(\boldsymbol{\lambda}) \; , \quad (3)$$

subject to constraints enforcing the graph structure:

C1. Flow conservation. Every VP bin can maximally be traversed by one track:

$$\forall j,t: \quad \lambda_s(j,t) + \sum\nolimits_{j' \in J} \lambda_t(j',t-1,j) = \lambda_e(j,t) + \sum\nolimits_{j' \in J} \lambda_t(j,t,j') \leq 1 \quad . \quad (4)$$

C2. Line-VP association. Only active VP bins can support line-VP arcs. A line can at most be linked to one VP:

$$\forall j,t: \quad \lambda_b(j,t) \cdot I_t - \sum\nolimits_i^{I_t} \lambda_l(j,i,t) \geq 0 \quad , \quad (5)$$

$$\forall i,t: \quad \sum\nolimits_{j' \in J} \lambda_l(j,i,t) \leq 1 \quad . \quad (6)$$

Additionally, we discovered that it is helpful to directly constrain which VP bins can be active together.

C3. Non-Maximum Suppression. For an active VP bin, we suppress other active VP bins in neighborhood $\mathcal{N}_s$:

$$\forall j,t: \; \lambda_b(j,t) \cdot |\mathcal{N}_s| + \sum\nolimits_{j' \in \mathcal{N}_s(j)} \lambda_b(j',t) \leq |\mathcal{N}_s| \; . \quad (7)$$

C4. Angle preservation. For two active VP bins $V_{j,t}$,$V_{j',t}$ linked to active $V_{k,t+1}$,$V_{k',t+1}$, respectively, we require constancy of the enclosed angle. This follows from the fact that vanishing directions are constant over time.

$$|\arccos\langle V_{j,t}, V_{j',t}\rangle| - |\arccos\langle V_{k,t+1}, V_{k',t+1}\rangle| < \epsilon_a \quad (8)$$

C5. Orthogonality (optional). Similar to [11, 13, 33] we can optionally enforce that all tracked VPs have to be mutually orthogonal at all times:

$$\forall j, j', t | j \neq j : \lambda_b(j,t)\ \lambda_b(j',t)\ |\langle V_{j,t}, V_{j',t}\rangle| < \epsilon_a\ . \quad (9)$$

C1 and C2 are essential to enforce the graph structure as visualized in Fig. 3. We found experimentally that C3 is only needed for strong noise in line endpoints, and C4 for horizontal VPs near infinity. If a Manhattan world is assumed, C5 can be used. Because of a lack of an appropriate dataset, we only evaluate inclusion of C5 for single frames.

Integer Linear Programming is NP-hard in general. However, using branch-and-bound, implemented in many solvers such as CPLEX[1], our problem is optimally[2] solved.

The solution $\boldsymbol{\lambda}^*$ gives a set of n VP tracks $\mathcal{V} = \{\mathcal{V}_1, \ldots, \mathcal{V}_n\}$, where a VP track $\mathcal{V}_i = \{V_{k,t_a}, \ldots, V_{k',t_b}\}$ consists of list of VP bins from time t_a to t_b. Furthermore, for each active VP bin $V_{k,t}$ in track $\mathcal{V}_i$ at time t we obtain all interpretation planes assigned to that VP at that time.

3.2. Score and Cost Modeling

We derive the scores and costs probabilistically and convert them into values for the objective function f in (2). We set all unary, start and exit probabilities for all bins j at times t uniformly to $P_u = P_s = P_e = 1/J$. This leads to decreased sensitivity of the method for finer discretization (higher J), and offsets the effect of stronger influence of line segment noise. It is easy to add further domain knowledge at this point: non-uniform P_u over all spherical bins can encode *e.g.* higher probabilities for horizontal VPs, if the gravity direction is approximately known. Non-uniform P_s, P_e over time can give a bias for known start and end times of VP tracks, *e.g.* from initial labeling.

Between bins j and j' we assign a transition probability based on the enclosed angle $\alpha = |\arccos\langle V_{j,t}, V_{j',t+1}\rangle|$:

$$P_t(j,j') = (1 + e^{\ \gamma_1 \cdot (\alpha - \gamma_2)})^{-1} \quad . \quad (10)$$

This sigmoid function yields a smooth fall-off at $\alpha = \gamma_2$, with decay rate controlled by γ_1. Since bin locations are fixed, $P_t(j,j')$ is independent of time t.

Out of the many line-VP consistency measures used in related works, we select a simple angular distance between plane normal and VP bin. The plane $P_{i,t}$ exactly intersects the sphere on a greater circle through $V_{j,t}$ iff the angle $\beta = |\arcsin\langle V_{j,t},\ P_{i,t}\rangle| = 0$, i.e. iff plane normal and VP are exactly orthogonal. We set the linking probability:

$$P_l(j,i,t) = (1 + e^{\gamma_3 \cdot (\beta - \gamma_4)})^{-1} \quad . \quad (11)$$

[1] www.ibm.com/software/commerce/optimization/cplex-optimizer/

[2] In practice we terminate the search early for a small optimality gap for efficiency reasons with no measurable performance loss.

This sigmoid function yields a smooth fall-off at $\beta = \gamma_4$, with decay rate controlled by γ_3.

Probabilities P_u, P_s, P_e, P_t, P_l are converted into costs and scores as in [35]. Let costs for bins j, j' at time t be

$$C_u = C_s = C_e = -\log P_u = -\log P_s = -\log P_e\ , \quad (12)$$

$$C_t(j,t,j') = -\log P_t(j,j') \quad . \quad (13)$$

Scores for bin j at time t and line segment i are given as:

$$S_l(j,i,t) = \log \frac{1 - P_l(j,i,t)}{P_l(j,i,t)} \quad . \quad (14)$$

Scores can be negative and encourage active tracks, while costs are strictly positive and penalize active tracks.

4. Experiments

For all experiments we use the same implementation and parameters, detailed in § 4.1. We will evaluate our approach for three scenarios: joint VP detection and tracking on our new dataset in § 4.2, VP detection and tracking when camera poses are known § 4.3, and single-frame orthogonal VP detection on the York Urban Dataset (YUD) [10] in § 4.4.

4.1. Implementation Details

The neighborhood $\mathcal{N}_s$ for a VP bin j is selected such that no other VP is expected within $\mathcal{N}_s$. For our experiments we conservatively assume this to be the case for $\sigma = 5$ degrees: $\mathcal{N}_s(j) = \{V_{j',t} : |\arccos\langle V_{j',t}, V_{j,t}\rangle| < \sigma, j' \neq j\}$.

We set $\epsilon_a = \cos(1)$. Values $\gamma_1 = 10, \gamma_3 = 5, \gamma_2 = \gamma_4 = 2$ are set empirically after visual inspection of the dataset. Optimal $\gamma_{1,2}$ depend on the rate of change of orientation, $\gamma_{3,4}$ on line endpoint noise. While Fig. 3, Fig. 4 show a coarse 80-bin discretization, we chose a fine discretization of 5120 spherical bins for evaluation. This discretization is an empirically chosen trade-off between high VP accuracy (finer discretization) and short run-times (coarser discretization). Since solving large LPs may become computationally expensive, we propose five LP pruning strategies:

1. Grouping of Line Segments. We start by extracting LSD line segments [32], but reduce the number of lines, using the Hough transform, into maximally $I_t = 100$ lines.

2. Limiting Line-VP Association. We threshold and cut all line-VP bin associations with $P_l(j,i,t) < 0.5$.

3. Removal of VP Bins. We threshold and cut from the LP all VP bins j at time t for which $\sum_i^{I_t} S_l(j,i,t) > 0$. Furthermore, since all VP evidence is antipodal on the Gaussian sphere, we only use VP bins on one hemisphere.

4. Pruning of Transitions. We cut all transition arcs between $V_{j,t}$ and $V_{j',t+1}$, if $P_t(j,j') < 0.2$.

5. Batch Processing. We solve the LP in batches of maximally 30 frames using branch-and-bound in CPLEX. After the solution $\boldsymbol{\lambda}^*$ is found, we refine each VP in each

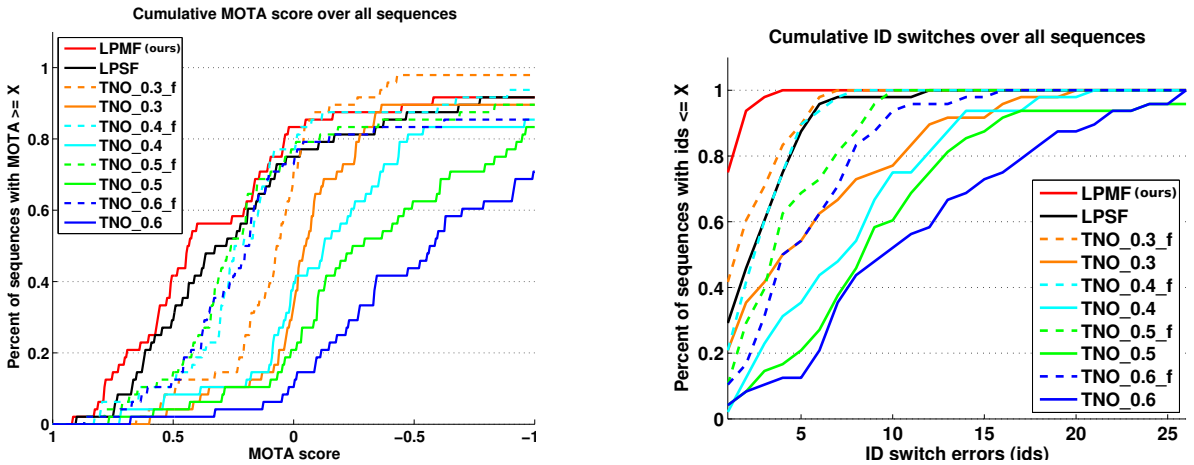

Figure 5: Cumulative MOTA and ID switches over 48 sequences for the Street-View evaluation in § 4.2. The result for the proposed method (LPMF) is colored in red. For MOTA our approach outperforms all baselines for 58 percent of the dataset, and is on par with the best baselines for the rest. For ID switches our method outperforms all baselines significantly over the whole dataset.

frame via least-squares optimization using Eq. (1). Batches are greedily merged, as explained for our baselines in § 4.2.

Experimentally we found that these pruning steps do not change the solution of the LP, but are helpful (esp. steps 1 and 3) to keep the computation to a few seconds per frame.

4.2. Evaluation on the Street-View Dataset

Dataset: As the field lacks a dataset for the evaluation of joint VP detection and tracking we augmented a dataset for video registration to SfM models [17] with our own VP annotations. Multiple vanishing directions and identities across frames were annotated semi-automatically in all frames, by using the known global camera pose and a supervised interpretation plane clustering. We provide more information about the annotation procedure in the supplementary material. The annotations will be publicly available. The dataset consists of 48 sequences of 301 frames (at 10 fps) of street-view video from van-mounted cameras, yielding a total of 14448 annotated frames with between zero and three VPs. Due to the non-orthogonal street-layout, the Manhattan world assumption is generally not valid for this dataset. The videos are of varying difficulty for VP extraction, and contain easy city scenes, as well as challenging scenes dominated by vegetation and street furniture.

Baselines: Since this is the first work for joint VP extraction and tracking, we construct our own baseline based on [29]. We compare our method to two types of baselines.

The first baseline, *LPSF* (*Linear Program, Single-Frame*), corresponds to VP detection with our proposed method, where every frame is treated separately. Following the frame-wise VP extraction, we greedily grow VP tracks. Initially, the set of VP tracks is empty. For a new frame we merge VPs to existing VP tracks if the angular difference is smaller then 5 degrees. The remaining VPs of this new frame start new tracks. Finally, we remove VP tracks shorter than 3 frames. The second baseline, in several variants, called *TNO*, for *Tardif, Non-Orthogonal*, corresponding to a recent single-frame VP extraction method [29], where we ignore the extension for orthogonal VPs. Since the VP detection sensitivity of [29] is strongly linked to the image resolution, we downscale the image by factors $\{0.3, 0.4, 0.5, 0.6\}$ to obtain better scores for this baseline. VP tracks are grown greedily as for *LPSF*. VP tracks shorter than 3 frames are removed for the variants marked with *f*.

Results: We evaluate with two metrics commonly used in tracking: *multi-object tracking accuracy* (MOTA, higher=better) [7], and *ID switches* (IDS,lower=better) [19]. The result is visualized in Fig. 5. The VP matching threshold was set to an angle of 2 degrees. Our method is denoted as *LPMF* (*Linear Program, Multi-Frame*). For MOTA the best baseline results are generally obtained with *LPSF* and *TNO 0.4 f*. The MOTA scores of *TNO 0.4 f* and *TNO 0.5 f* are very similar, but *TNO 0.4 f* has significantly fewer ID switches. Stronger downscaling (≤ 0.3) leads to an increase in missed detections, and weaker downscaling (≥ 0.6) to many false positive detections. Removing short (< 3 frames) VP tracks always increases MOTA and decreases ID switches. Stronger filtering leads to a loss in MOTA.

Greedy VP track linking in all baselines often fails, because of the lack of temporal smoothness constraints in the VP detection. This leads to frequent loss and reinitialization of VP tracks, yielding many ID switches. Our method significantly outperforms all baselines in ID switches: in 75 percent of the dataset no ground truth track is split. The best baseline on IDS, *TNO 0.3 f*, achieves this only for 41 percent of the dataset, with significantly worse MOTA. In MOTA our approach outperforms all baselines for 58 percent of the dataset, and is on par with the best baselines for MOTA > 0.

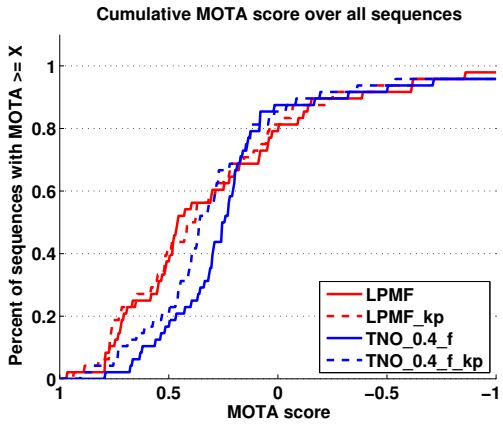

Figure 6: Cumulative MOTA for the evaluation in § 4.3. If pose knowledge is incorporated our method (LPMF kp) still outperforms the best baseline (TNO 0.4 f kp) for most sequences.

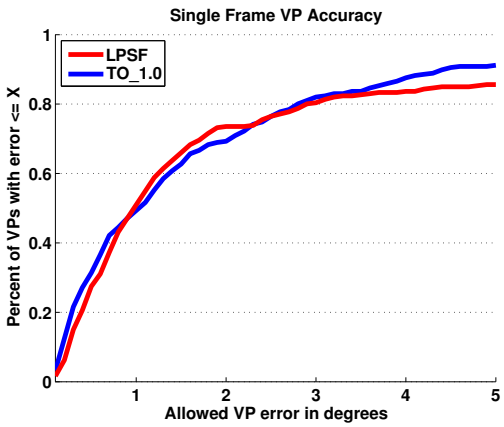

Figure 8: Single Frame VP Accuracy in § 4.4 for Manhattan scenes. Both methods were applied to extract orthogonal VPs only.

For a MOTA threshold of 0.5, our method, and the two best baselines *LPSF* and *TNO 0.4 f*, have 42, 29 and 13 percent, respectively, of all sequences above this score. MOTA in all methods drops significantly for the most difficult 20 percent of all sequences. For all methods the strongest negative influence on MOTA comes from missed VPs due to weak line support. In MOTP (*multi-object tracking precision*) all methods offer very similar performance. We provide a detailed quantitative evaluation (MOTA, MOTP, IDS) for all 48 videos in the supplementary material.

Our unoptimized single-core Matlab implementation (using CPLEX) requires 2.8 seconds per frame on average on a Intel Core i7 CPU. One second is needed for preprocessing (line extraction, Hough grouping) and the rest for solving the LP. The best baseline *TNO 0.4 f* runs for 0.7 seconds per frame including line extraction. Recent related works report runtimes from 'a few seconds' [34] to half a minute [30, 18] per frame for similar image resolutions, but without the need for finding temporal correspondences. The extra time required for our approach in comparison to the *TNO 0.4 f* baseline is spent well on joint temporal data association, since our method leads to significantly fewer failure cases. This is demonstrated by two important experimental results: Firstly, our approach creates significantly fewer false positive tracks, as reflected in the MOTA score. Secondly, our approach rarely splits a ground truth track: in 75 percent of the dataset our approach does not have any ID switch, while this is only the case for 21 percent of all sequences for *TNO 0.4 f*. Low IDS is especially crucial for long-term operations, in which continuous VP identity information is needed and re-initializations are costly.

Some qualitative results are shown in Fig. 7. Lines which were associated to the same VP track are drawn in the same color. Examples for challenging scenes with failure cases are shown as well. In these examples only short and noisy line segments are available, due to high-frequency texture on the ground (cobble stone) and dominance of vegetation.

4.3. Inclusion of Known Camera Orientation

Introduction. If the camera orientation is known for every frame (*e.g.* from odometry or 3D reconstruction) the problem of finding VPs over time can be simplified: 3D vanishing directions in the world reference frame are constant in time. Because of this, knowing the camera orientation in the world reference frame is equivalent to having the tracking component of our joint problem partially solved. We want to evaluate how much our method and the baselines improve when taking advantage of this knowledge.

Benchmark. In practice we can easily incorporate the known camera orientation for our method by rotating all VP evidence (i.e. 3D interpretation planes for all line segments) into the common world coordinate system for every frame. For our baselines we incorporate the known orientation after VP extraction, and apply the rotation to the extracted VPs. All other components of the methods remain unchanged. Since the Street-View dataset [17] provides precise camera poses for all frames, we can evaluate the behavior of our method when including pose knowledge on this dataset. The dataset generally does not have strong orientation changes. Because of this we subsampled the sequences to 1 fps, i.e. every 10th frame, for evaluation with strong frame-wise orientation changes. In Fig. 6 we show the cumulative MOTA score for our original method and the best performing baseline without (*LPMF*, *TNO 0.4 f*) and with (*LPMF kp*, *TNO 0.4 f kp*) inclusion of known poses.

Results. We observe that the separate VP extraction and greedy tracking in *TNO 0.4 f kp* improves strongly with known camera orientations. While our approach, *LPMF kp*, still outperforms both baselines the improvement is smaller

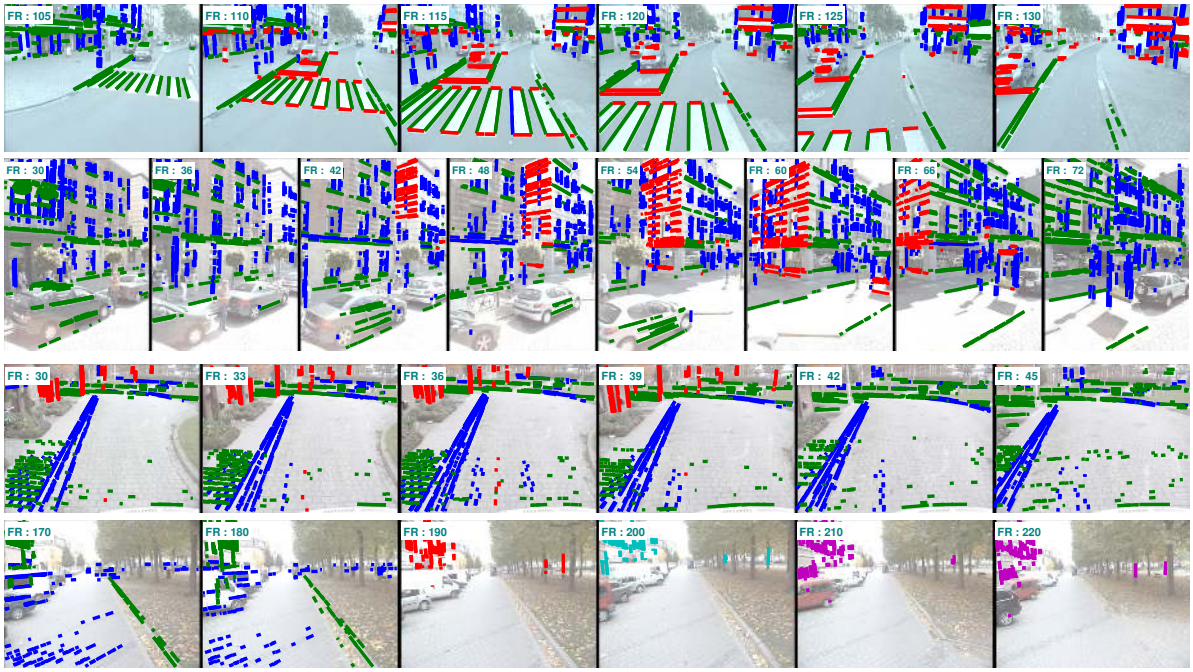

Figure 7: VP detection and tracking examples. Line segments are colored according to their association to a VP. **Top 2 rows**: in each example three VPs are visible, correctly extracted and tracked. **Bottom 2 rows**: two examples for challenging scenarios. Short line segments are a problem in both cases. 3rd row: The vertical VP (red) is only weakly supported by noisy vertical line segments on vegetation. Some line segments for a horizontal VP (blue) are incorrectly associated. 4th row: The vertical VP (green, red, cyan, magenta) is not reliably tracked and has multiple ID switches.

than for *TNO 0.4 f kp*. This is explained by the smooth transition probability (10) which recovers the VP motion quite well. However, using the pose we could achieve a speed-up by increasing the sensitivity of (10) (i.e. setting γ_2 small and γ_1 large), which reduces the search space for the LP solution. The remaining gap in MOTA between our method and *TNO 0.4 f kp* is explained by the fact that our method supports VP tracks through multiple frames where line support is weak, while frame-wise extraction and greedy tracking in *TNO 0.4 f kp* will often lose tracks in those cases.

4.4. Evaluation on the York Urban Dataset (YUD)

Because the proposed method can also be applied to single-frame VP extraction, we evaluated our approach on the established York Urban Dataset (YUD) [10] for Manhattan world VP extraction. We compare again to [29], labeled in the figure as *TO*, for *Tardif, Orthogonal*, where we include an EM step to refine the set of orthogonal VPs. We run our method on single frames, using constraint C5 (9) to enforce orthogonality. As can be seen in Fig. 8, our approach is on par with [29] for VP estimation accuracy.

5. Conclusion

Vanishing points encode low-level information of the scene structure and are used in many applications from scene understanding to 3D reconstruction. Many of these applications operate on video input and can benefit from knowledge about VP continuation over time. In this work we presented an approach for jointly extracting and tracking VPs from video with known internal camera calibration.

We are the first to propose a method for this problem and provide a new dataset for evaluation. We showed that our method significantly outperforms various iterative detection and tracking approaches. We showed that even in cases where poses are known, a multi-frame approach is helpful as a temporal regularizer.

We focused on scenarios where no prior knowledge about the scene is available, and tested with simple cost and score models. As extensions, the method can be easily combined with many more powerful subsystems, as mentioned in §1: line-VP consistency in image space, priors from partially or approximately known orientations, non-uniform spherical discretization, and VP refinement steps. Since the extracted VPs are constant in the world reference frame, our method can also be used to compute the change in camera orientation over time. Our method can also be adapted to other VP representations for which a non-parametric probability density over VP locations is available.

Acknowledgments: This work was supported by the European Research Council project *VarCity (#273940)*.

References

[1] M. E. Antone and S. Teller. Automatic Recovery of Relative Camera Rotations for Urban Scenes. In *CVPR*, 2000. 1, 2

[2] M. Antunes and J. a. P. Barreto. A Global Approach for the Detection of Vanishing Points and Mutually Orthogonal Vanishing Directions. In *CVPR*. Ieee, 2013. 2, 4

[3] S. T. Barnard. Interpreting Perspective Images. *Artificial Intelligence*, 1982. 2

[4] J.-C. Bazin, C. Demonceaux, P. Vasseur, and I. Kweon. Rotation estimation and vanishing point extraction by omnidirectional vision in urban environment. *International Journal of Robotics Research*, 31(1):63–81, 2012. 2

[5] J.-C. Bazin and M. Pollefeys. 3-line RANSAC for Orthogonal Vanishing Point Detection. *IROS*, 2012. 2

[6] J. Berclaz, F. Fleuret, E. Türetken, and P. Fua. Multiple Object Tracking using K-Shortest Paths Optimization. *PAMI*, 2011. 1, 3

[7] K. Bernardin, A. Elbs, and R. Stiefelhagen. Multiple Object Tracking Performance Metrics and Evaluation in a Smart Room Environment Performance Metrics for Multiple Object Tracking. *EURASIP*, 2008. 2, 6

[8] B. Caprile and V. Torre. Using Vanishing Points for Camera Calibration. *IJCV*, 1990. 2

[9] D. J. Crandall, A. Owens, N. Snavely, and D. P. Huttenlocher. SfM with MRFs: Discrete-Continuous Optimization for Large-Scale Structure from Motion. *PAMI*, 2012. 1

[10] P. Denis, J. H. Elder, and F. J. Estrada. Efficient Edge-Based Methods for Estimating Manhattan Frames in Urban Imagery. In *ECCV*, 2008. 2, 5, 8

[11] W. Elloumi, S. Treuillet, and R. Leconge. Tracking Orthogonal Vanishing Points in Video Sequences for a Reliable Camera Orientation in Manhattan World. In *CISP*, 2012. 1, 2, 5

[12] L. Grammatikopoulos, G. Karras, and E. Petsa. An Automatic Approach for Camera Calibration from Vanishing Points. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2007. 1, 2

[13] M. Hornáček and S. Maierhofer. Extracting Vanishing Points across Multiple Views. In *CVPR*, 2011. 1, 2, 5

[14] C. Kim and R. Manduchi. Planar Structures from Line Correspondences in a Manhattan World. In *ACCV*, 2014. 1

[15] J. Košecká and W. Zhang. Video Compass. In *ECCV*, 2002. 1, 2

[16] T. Kroeger, D. Dai, R. Timofte, and L. Van Gool. Discovery of Sets of Mutually Orthogonal Vanishing Points in Videos. In *BMTT Workshop at WACV*, 2015. 2

[17] T. Kroeger and L. Van Gool. Video Registration to SfM Models. In *ECCV*, 2014. 2, 6, 7

[18] J. Lezama, R. G. Von Gioi, G. Randall, and J.-m. Morel. Finding Vanishing Points via Point Alignments in Image Primal and Dual Domains. In *CVPR*, 2014. 2, 7

[19] Y. Li, C. Huang, and R. Nevatia. Learning to associate: HybridBoosted multi-target tracker for crowded scene. In *CVPR*, 2009. 2, 6

[20] E. Lutton, H. Maître, and J. Lodez-Krahe. Contribution to the Determination of Vanishing Points Using Hough Transform. *PAMI*, 1994. 2

[21] M. J. Magee and J. K. Aggarwal. Determining vanishing points from perspective images. *Computer Vision, Graphics, and Image Processing*, 1984. 2

[22] B. Micusik and H. Wildenauer. Minimal solution for uncalibrated absolute pose problem with a known vanishing point. In *3DVV*, 2013. 1, 2

[23] P. Moghadam and J. F. Dong. Road Direction Detection Based on Vanishing-Point Tracking. *IROS*, 2012. 1, 2

[24] L. Quan and R. Mohr. Determining perspective structures using hierarchical Hough transform. *Pattern Recognition Letters*, 1989. 2

[25] C. Rasmussen. RoadCompass: following rural roads with vision + ladar using vanishing point tracking. *Autonomous Robots*, 2008. 1, 2

[26] C. Rother. A new approach to vanishing point detection in architectural environments. *Image and Vision Computing*, 20, 2002. 2

[27] 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. In *CVPR*, 2004. 2

[28] J. Straub, G. Rosman, O. Freifeld, J. J. Leonard, and J. W. Fisher III. A Mixture of Manhattan Frames : Beyond the Manhattan World. In *CVPR*, 2014. 2

[29] J.-P. Tardif. Non-Iterative Approach for Fast and Accurate Vanishing Point Detection. In *ICCV*, 2009. 2, 6, 8

[30] E. Tretyak, O. Barinova, P. Kohli, and V. Lempitsky. Geometric Image Parsing in Man-Made Environments. *IJCV*, 2012. 2, 7

[31] T. Tuytelaars, M. Proesmans, and L. V. Gool. The Cascaded Hough Transform as Support for Grouping and Finding Vanishing Points and Lines. In *AFPAC*, 1997. 2

[32] R. G. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. LSD: a Line Segment Detector. *IPOL*, 2012. 5

[33] H. Wildenauer and A. Hanbury. Robust Camera Self-Calibration from Monocular Images of Manhattan Worlds. In *CVPR*, 2012. 1, 2, 5

[34] Y. Xu, S. Oh, and A. Hoogs. A Minimum Error Vanishing Point Detection Approach for Uncalibrated Monocular Images of Man-made Environments. In *CVPR*, 2013. 2, 7

[35] L. Zhang, Y. Li, and R. Nevatia. Global Data Association for Multi-Object Tracking Using Network Flows. In *CVPR*, 2008. 1, 3, 4, 5