MCMLSD: A Dynamic Programming Approach to Line Segment Detection

Emilio J. Almazàn
Nielsen
Madrid, Spain
ealmazanm@gmail.com

Ron Tal
Uber Technologies
San Francisco, USA
ron@uber.com

Yiming Qian
York University
Toronto, Canada
yimingq@cse.yorku.ca

James H. Elder
York University
Toronto, Canada
jelder@yorku.ca

Abstract

Prior approaches to line segment detection typically involve perceptual grouping in the image domain or global accumulation in the Hough domain. Here we propose a probabilistic algorithm that merges the advantages of both approaches. In a first stage lines are detected using a global probabilistic Hough approach. In the second stage each detected line is analyzed in the image domain to localize the line segments that generated the peak in the Hough map. By limiting search to a line, the distribution of segments over the sequence of points on the line can be modeled as a Markov chain, and a probabilistically optimal labelling can be computed exactly using a standard dynamic programming algorithm, in linear time. The Markov assumption also leads to an intuitive ranking method that uses the local marginal posterior probabilities to estimate the expected number of correctly labelled points on a segment. To assess the resulting Markov Chain Marginal Line Segment Detector (MCMLSD) we develop and apply a novel quantitative evaluation methodology that controls for under- and over-segmentation. Evaluation on the YorkUrbanDB dataset shows that the proposed MCMLSD method outperforms the state-of-the-art by a substantial margin.

1. Introduction

Much of our visual world can be approximated as piecewise planar, particularly in built environments. The boundaries and creases of these piecewise planar surfaces project to the image as line segments, and as a consequence the accurate detection of line segments continues to be one of the most important low-level problems in the field of computer vision. Line segments are important features for many tasks, including feature matching across views [23], vanishing point detection [15] and 3D reconstruction [22, 30, 12].

Two frameworks have been popular for line segment detection: perceptual grouping and global Hough analysis.

1.1. The perceptual grouping approach

In the perceptual grouping framework, a set of heuristics typically based upon geometric grouping cues (e.g., proximity, good continuation) is used to group roughly collinear local features (e.g., edges or vectors tangent to isophotes) into extended line segments, which are evaluated according to some quality of fit measure. An early example is the hierarchical heuristic framework developed by Boldt and colleagues [3]. More recent multi-stage grouping efforts include the SSWMS approach of Nieto et al. [21], which involves an iterative selection of image points with strongly oriented gradient structure, followed by an iterative growing process, the approach of Lu et al. [19], which involves both linking and splitting, and the biologically inspired approach of Liu et al. [18], which employs 'simple cell' filters to detect local oriented structure, 'complex cell' mechanisms that locally integrate these responses and 'hyper-complex' mechanisms to detect endpoints.

An alternative to this multi-stage grouping approach is to analyze the covariance matrix of image locations in a set of connected edges and label a set as a line segment if the smallest eigenvalue falls below a threshold [11, 17]. While beautifully simple, these methods are not robust to gaps or intersections in the edge map.

Another issue in this perceptual grouping framework is that some threshold on the quality of fit measure must be applied in order to discriminate 'true' line segments from false conjunctions that might arise by chance. This issue was addressed in the LSD framework introduced by von Gioi et al. [26] and based on earlier work by Desolneux et al. [7]. In this framework the so-called *a-contrario* approach is used to explicitly compute the probability that inferred line segments might have occurred by chance, given a maximum entropy model of the edge map. (This is related to the minimum reliable scale null hypothesis testing framework for edge detection developed by Elder & Zucker [8].) While this approach does not eliminate the need for a threshold, it transfers the threshold to a quantity (e.g., expected number of false positives per image) that is much easier to set rationally. A much faster version of this method dubbed

EDLines was later introduced by Akinlar & Topal [1].

Recent work in this area has focused on trying to discriminate salient or important line segments from less important 'background' segments. Kim et al. [14] used a combination of luminance and geometric features to select the most significant edges, reporting superior performance to LSD on two test images. Brown et al. [4] used a measure of divergence between colour statistics on either side of a hypothesized line segment to favour salient segments. The method outperformed LSD and Hough methods using quantitative measures of repeatability and registration accuracy on image pairs (see Section 3 below).

1.2. The Hough approach

A drawback of the perceptual grouping approach is that local decisions are made before potentially relevant global information can be brought to bear. The Hough approach avoids this problem by accumulating edges over the entire image into a histogram of potential line positions and orientations. Accuracy can be improved by modeling uncertainty in local edges and propagating that uncertainty to the Hough map [24].

While the Hough approach to line detection has the advantage of integrating information globally, identifying the endpoints that define the extent of the line segment in the image is not necessarily straightforward. A number of methods scan the detected lines in the image space looking for a maximal chain of connected or nearly-connected edges [10, 20]. Others have attempted to identify the endpoints of each line segment by analyzing the exact shape of a characteristic 'butterfly' pattern around the associated peak in the Hough map [13, 9, 28, 29, 27]. One major limitation of this approach is that only one segment can be found per line, whereas in built environments it is quite common to find multiple co-linear segments.

2. Our approach

The advantage of the Hough approach is that it can integrate all evidence for line hypotheses prior to inference. The perceptual grouping approach, on the other hand, allows endpoints to be detected more directly, and permits the identification of multiple segments per line.

Our two-stage method combines the advantages of these two approaches. In the first stage we employ the probabilistic Hough method of Tal & Elder [24] to identify globally optimal lines. In the second stage we search each of these lines in the image for the segment(s) that gave rise to it.

The key observation that recommends this approach is that narrowing the search for segments from the 2D image to 1D lines allows the problem to be modeled as the labelling of hidden states in a linear Markov chain model. The problem of determining the maximum probability (MAP) assignment of segments can then be shown to have an optimal substructure property that leads to an exact dynamic programming solution in linear time.

The benefits of this approach are several:

1. Each of the lines identified by a peak in the Hough map results from careful accumulation of the global evidence for the line, and thus will more accurately identify the position and orientation (ρ, θ) parameters of the line segments than will a few local edges.

2. The lines identified by the probabilistic Hough method have a natural order according to their significance in the Hough map, allowing the line segment search to be limited to the most significant lines.

3. In urban scenes, co-linear line segments are common, arising from architectural repetition seen in cladding, windows, etc. Unlike many Hough methods, our approach allows multiple segments to be recovered for each line.

4. Limiting search to a line allows the problem of determining maximum probability segments to be solved exactly, using dynamic programming, in linear time.

3. Prior Evaluation Methodology

Due in part to a lack of high quality labelled ground truth, most line segment detection methods are evaluated only qualitatively on real imagery [3, 9, 26, 1, 17]. More recently, quantitative evaluations have been conducted based on datasets consisting of pairs of images related by a known homography [4]. This is a promising method, but it does suffer from two potential drawbacks. First, it is restricted to an analysis of co-planar line segments. Second, the evaluation presupposes that the goal of line segment detection is for the association of these segments across images for the purposes of homography or disparity estimation. However there are many other possible applications - single view reconstrucion, for example.

While task-specific evaluation methodologies may be appropriate in some cases, it would be nice to have an evaluation method that is more general. In this work we present a new methodology for quantitative evaluation of line segment detectors on real images that does not assume a specific task, using images from the YorkUrbanDB dataset (`www.elderlab.yorku.ca/YorkUrbanDB/`).

4. Algorithm

4.1. Line Detection

One problem with traditional Houghing methods is that noise in the observations tends to cause each line

to generate multiple peaks in the Hough map. To address this issue we employ a probabilistic Hough transform method [24] (code available from elderlab.yorku.ca/resources), using the default Hough resolution: $\Delta\rho = 0.2$ pixels, $\Delta\theta = 0.1$ deg. The method uses edges detected by the multi-scale Elder & Zucker edge detector [8], models uncertainty in the location and orientation of the detected edges and propagates this uncertainty to the Hough map. This propagation of uncertainty produces a smooth Hough map that is roughly resolution invariant and greatly reduces the multiple response problem. The problem is mitigated further by a sequential line extraction step in which each peak in the Hough map is visited in descending order of significance, and edges contributing to the peak are subtracted from the Hough map when it is visited.

4.2. Line Segment Detection

Each selected peak in the Hough map identifies a line that extends from one of the image borders to another. In general, this line is only partially occupied by line segments in the image. The goal is now to find these segments, based on the location and orientation of nearby edges.

Prior work [6] suggests that most edges generated by a line and detected by the Elder & Zucker edge detector lie within one pixel of the line. To ensure we capture all edges related to a line we extend our search to all pixels within two pixels of the line (Fig. 1). The orthogonal projections of these pixel locations onto the line then define an ordinal sampling $i \in [1, \ldots, N]$ of the line. We let x_i represent the binary hidden segment state (ON or OFF) indicating whether a visible segment is present at position i on the line, d_i the distance from the line to the associated pixel and y_i the associated image observation at that pixel. Each observation y_i consists of 1-2 features:

1. A binary variable e_i indicating whether an edge exists at this pixel.

2. The angular deviation θ_i of the edge from the line, if the edge exists.

These features provide information about the probable state of the line at the associated position:
$p(y_i|x_i) \propto p(e_i = 1|x_i, d_i)p(\theta_i|x_i, e_i = 1)$ for edge pixels.
$p(y_i|x_i) \propto p(e_i = 0|x_i, d_i)$ for non-edge pixels.
(Note that we have assumed that the angular deviation θ_i is independent of the distance d_i of the pixel from the line.)

We learned these distributions from the 640×480 pixel images and hand-labelled ground truth lines of the YorkUrbanDB training dataset [6]. Figs. 2(a-b) show the likelihoods $p(e_i = 1|x_i = \text{ON}, d_i)$ and $p(e_i = 1|x_i = \text{OFF}, d_i)$ as functions of d_i for ON and OFF states respectively. We represent these distributions as histograms. (The likelihoods for non-edge observations $p(e_i = 0|x_i = \text{ON}, d_i)$ and $p(e_i = 0|x_i = \text{OFF}, d_i)$ are the complements of the edge likelihoods.)

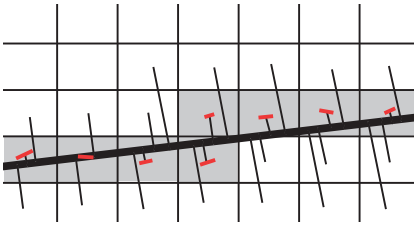

Figure 1: Orthogonal projections (thin black lines) of all pixels within two pixels of a detected line (thick black line) define an ordinal sampling of the line $i \in [1 \ldots N]$. Pixels within this band occupied by edges (shown red on grey) with orientations similar to the line support the assignment of the ON state for the associated segment variable x_i at sampled line locations.

Figs. 2(c-d) show the probability $p(\theta_i|x_i, e_i = 1)$ as a function of the angular deviation θ_i for ON ($x_i = 1$) and OFF ($x_i = 0$) states, respectively. For the ON state we approximate the heavy-tailed distribution as a mixture of a uniform and a Gaussian distribution (shown in red). For the OFF state we employ a histogram representation.

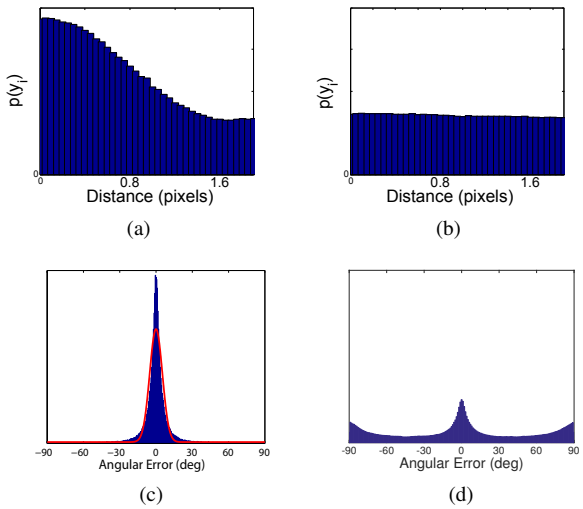

Figure 2: Likelihoods for line segment extraction, learned from the YorkUrbanDB training dataset [6]. (a-b) Likelihood $p\left(e_i|x_i, d_i\right)$ for distance d_i of observations from line for (a) ON ($x_i = 1$) and (b) OFF ($x_i = 0$) states. (c-d) Probability $p\left(\theta_i|x_i, e_i\right)$ for the angular deviation θ_i of observed edges from the line for (c) ON ($x_i = 1$) and (d) OFF ($x_i = 0$) states.

Given these observations, we wish to determine the se-

quence of hidden states $x_1, ..., x_N$ that maximizes

$$p(x_1, \ldots, x_N | y_1, \ldots, y_N) \propto p(y_1, \ldots, y_N | x_1, \ldots, x_N) p(x_1, \ldots, x_N) \quad (1)$$

We assume that, when conditioned on the hidden states x_i, the observations y_i are mutually independent and independent of all x_j, $j \neq i$. We further assume that the hidden states are first order Markov so that Eqn. 1 becomes

$$p(x_1, \ldots, x_N | y_1, \ldots, y_N) \propto p(y_1|x_1)p(x_1) \prod_{i=2}^{N} p(y_i|x_i)p(x_i|x_{i-1}) \quad (2)$$

The Markov assumption implies an exponential distribution of segment lengths; for the YorkUrbanDB training dataset we have verified that this distribution is indeed very close to exponential for segments down to ~ 15 pixels in length. (For smaller segments the density falls off, possibly due to difficulties in hand-labelling shorter segments.)

Table 1 shows values for the priors $p(x_1)$ and $p(x_i|x_{i-1})$, estimated from the 51 640×480 pixel images of the YorkUrbanDB labeled training dataset [6]. (Note that since the probabilities for ON and OFF states sum to 1 there are only 3 free parameters.) We make the approximation that $p(x_i|x_{i-1})$ is independent of the variation in spacing between points on the line; since the average segment in the YorkUrbanDB generates more than 500 point samples, errors due to this approximation tend to average out.

The standard errors for these parameter estimates are relatively small, and we have verified that variation within this range has negligible effect on results. While these parameters are specific to the YorkUrbanDB dataset and may therefore be sub-optimal for other kinds of imagery, they *can* be generalized to other image resolutions. Assuming that the number of segments per line and their relative length are functions of the scene and not the sensor, $p(OFF)$ and $p(ON)$ will be resolution-invariant and the probability of state changes will vary inversely with resolution. For example, doubling the resolution to 1280×960 pixels will halve the probability of transition from OFF to ON or ON to OFF.

The factoring of the global probability of the line segment configuration along the line confers an optimal substructure property that allows a dynamic programming solution to the problem of finding the maximum a posteriori configuration. In particular, let the cost function $C_i(j)$ represent the minimum negative log probability of all sequences $\{x_1, \ldots, x_i\}$ ending in state $x_i = j$. Then the maximum probability sequence of states over the whole line is the sequence that minimizes $\min_j C_N(j)$.

Defining the cost of transitioning from state j at location $i-1$ to state k at location i as

$$c_i(j,k) = -\log\left(p(y_i|x_i = k)p(x_i = k|x_{i-1} = j)\right), \quad i = 2, \ldots, N \quad (3)$$

we then have that

$$C_1(k) = -\log\left(p(y_1|x_1 = k)p(x_1 = k)\right) \quad (4)$$

$$C_i(k) = \min_j \left(C_{i-1}(j) + c_i(j,k)\right), i = 2, \ldots, N \quad (5)$$

Thus the cost function $C_i(k)$ can be computed sequentially from $i = 1$ to $i = N$ in $O(N)$ time (Fig. 3). In order to recover the maximum probability configuration, an auxiliary data structure containing

$$\hat{s}_i(k) = \arg\min_j (C_{i-1}(j) + c_i(j,k)) \quad (6)$$

is maintained, allowing the maximum probability configuration to be unwound from x_N back to x_1.

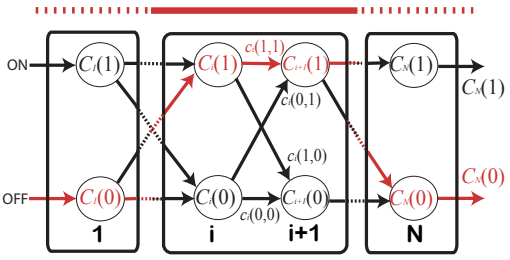

Figure 3: The sequence of segment state variables x_i are assumed to form a Markov chain. To compute the MAP solution we build a trellis table from the first line position $i = 1$ to the last line position $i = N$ that identifies the minimum cost (negative log probability) to reach either possible state (ON or OFF) at each position i. The selected MAP path is shown in red, and the resulting ON/OFF states are indicated by the solid/dashed line above the trellis.

Table 1: Prior marginal probabilities $p(x_i)$ and conditional transition probabilities $p(x_i|x_{i-1})$ for the hidden segment state x_i, derived from the YorkUrbanDB training dataset.

Parameter	Mean	Std. Err.
$p(OFF)$	0.75	0.0079
$p(ON)$	0.25	0.0079
$p(OFF\|OFF)$	0.9986	0.0001
$p(ON\|OFF)$	0.0014	0.0001
$p(ON\|ON)$	0.9949	0.0004
$p(OFF\|ON)$	0.0051	0.0004

Once a line segment is detected all associated edges (i.e., edges within two pixels of the segment) are removed from

the image. This serves to reduce the incidence of multiple detections for the same segment.

5. Ranking

Having extracted MAP segments for each line in the image, we would like to rank their significance. This will allow downstream applications to select only the number of segments needed to support their application, and can serve to eliminate low-ranked noise segments. Our Markov chain model allows us to approach the ranking problem from a probabilistic perspective. In particular, we evaluate the following four probabilistic methods for ranking a segment of length M extending from position i to position $i+M$:

Ranking Method 1. Posterior probability of line segment.

$$p(x_{i...i+M} = ON | y_{i...i+M})$$

This ranking criterion will maximize the expected number of segments with no false alarms.

Ranking Method 2. Posterior probability of line segment multiplied by length.

$$p(x_{i...i+M} = ON | y_{i...i+M}) * M$$

This criterion will maximize the expected total length of segments with no false alarms.

Ranking Method 3. Posterior odds for fully ON vs fully OFF configurations.

$$\frac{p(x_{i...i+M} = ON | y_{i...i+M})}{p(x_{i...i+M} = OFF | y_{i...i+M})}$$

Ranking Method 4. Sum of marginal posterior probabilities for ON states. The forward-backward algorithm is used to compute the posterior probability at each location.

$$\sum_{j=i}^{i+M} p(x_j = ON | y_{i:i+M})$$

This measure reflects the expected number of ON samples on the segment, and thus will maximize the expected number of correctly labelled locations within the segment.

6. Evaluation Methodology

It is important to evaluate line segment detection algorithms on real, complex images. Prior evaluations have generally been qualitative (i.e., visual). Recent efforts to quantify the evaluation require pairs of images related by a known homography, and are perhaps thus best suited for matching tasks [4]. Here we propose an alternative quantitative evaluation methodology that does not assume the existence of image pairs or known homographies and thus could be applicable for a broader range of tasks.

Our proposed evaluation method does require an image dataset in which important segments have been labelled. Here we employ the YorkUrbanDB dataset [6], which consists of 102 images of urban scenes, randomly divided into training and test subsets of 51 images each. In each image, major line segments that conform to one of the three so-called Manhattan directions [5] (i.e., vertical or horizontal and conforming to the main directions of orthogonal walls, streets, etc.) have been identified and labelled by hand. This database has been used widely to train and evaluate algorithms for vanishing point detection [25], line detection [2] and Manhattan frame estimation [6, 24].

We assume that the line segment detector under evaluation returns a list of line segments in ranked order. We sample each ground truth and detector segment uniformly with a sample spacing of one pixel and use these point samples to evaluate the detector as a function of the number k of top-ranked segments selected, varying k from 10 to 500.

For each value of k we first identify potential point matches as those (ground truth, detector) point pairs lying within a threshold distance of $2\sqrt{2}$ pixels of each other. This threshold was selected to associate any pair of lines that could potentially appear in the image with less than a one-pixel intervening gap. We then sort these candidate matches by Euclidean distance and accept matches in greedy fashion starting with the smallest distance, matching each point at most once, and thus arriving at a near-optimal bipartite match. Having associated ground truth and detector points, we employ the Hungarian algorithm [16] to identify the optimal bipartite match between ground truth and detector segments that maximizes the total number of points matched.

Now that we have a 1:1 association between ground truth and detector segments, it remains to evaluate the quality of this association. We propose three evaluative measures.

1. Recall as a function of the number of segments. We can compute a measure of recall as the number of ground truth point samples matched to detector samples, divided by the total number of ground truth point samples. This measure of recall is problematic if we allow matches without regard to the segments on which the points lie, as it does not penalize under-segmentation (joining multiple short segments into a single long segment) or over-segmentation (breaking up a long segment into multiple short segments).

However, constraining matches to lie on 1:1 associated segments solves both of these problems. In the case of under-segmentation, only one of the shorter ground truth segments is matched, leading to a high penalty. In the case of over-segmentation, only one of the detector segments is matched, again generating a high penalty.

Without additional constraints, using recall by itself is still problematic, as it is biased toward detectors that report a larger number of segments, thereby maximizing the prob-

ability of detecting ground truth points. We address this by comparing recall as a function of the same number k of segments reported.

2. Recall as a function of total segment length. There is still a potential bias in this recall-vs-k measure. Neglecting co-linear ground truth segments, the method can be biased toward detectors that report segments of maximal length (i.e., global lines) as this minimizes the risk of missing ground truth points. To address this potential bias, our second performance measure reports recall as a function of the sum L of the lengths of detected segments. This severely penalizes detectors that report over-long segments.

3. Precision-Recall. Our third and final performance measure is conventional precision-recall. We can take as a measure of precision the number of ground truth point samples matched to detector samples, divided by the total number of detector point samples. Again, by enforcing a 1:1 matching at the segment level, both under-segmentation and over-segmentation are penalized.

Since the YorkUrbanDB dataset does not provide a complete labelling of all segments in an image, detection of a segment that is not in the dataset does not necessarily represent an error. For this reason, the absolute precision values reported here should be interpreted with caution. Nevertheless, since the segments labelled in the YorkUrbanDB dataset are highly-visible features projecting from prominent structures in the scene, it is reasonable to expect a superior detector to rank these highly, and therefore attain higher *relative* precision values compared to inferior detectors.

To facilitate future comparisons, the code that performs these evaluations as well as the code for the MCMLSD algorithm is available online at `elderlab.yorku.ca/resources`.

7. Results

Our MCMLSD algorithm generates an average of 414 lines and 488 line segments for each 640×480-pixel image of the YorkUrbanDB training dataset. Note that not all lines generate a segment and some generate several segments. Fig. 4 shows the 10 top-ranked segments produced by each of our four ranking methods on an example image. We find that the multiplicative nature of the first criterion favours short high-confidence segments. This problem can be addressed by multiplying by segment length (Method 2), forming a contrast between purely ON and purely OFF configurations (Method 3), or summing the ON point marginals (Method 4) to estimate the number of correctly labeled points.

Figure 5 shows the recall for each of these ranking methods as a function of the number of segments returned, on the YorkUrbanDB training dataset. The bias toward shorter segments leads to poor recall for Method 1. Methods 2-4 yield much better results and in the sequel we adopt Method 4 as our ranking method of choice, given its superior performance and intuitive probabilistic interpretation. We call the resulting algorithm the Markov Chain Marginal Line Segment Detector (MCMLSD) to capture the importance of the Markov chain model of the line as well as the probabilistic ranking that maximizes the expected number of correctly labelled points on the segment.

(a) Method 1 (b) Method 2 (c) Method 3 (d) Method 4

Figure 4: 10 top-ranked segments for four ranking methods on example image.

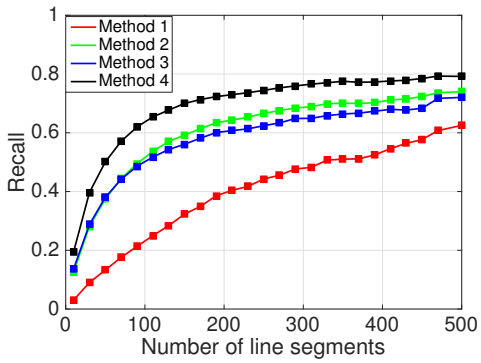

Figure 5: Performance of the four ranking methods described in section 5, as measured by recall vs number of segments returned, on the YorkUrbanDB training dataset.

Next, we compare the results of the proposed MCMLSD method against three other leading methods for which code is available online: The Progressive Probabilistic Hough Transform (PPHT) method of Matas et al. [20], the slice sampling weighted mean shift (SSWMS) method of Nieto et al. [21] and the widely-used line segment detector (LSD) method of Grompone von Gioi et al. [26].

We used the OpenCV implementation of the PPHT algorithm. This algorithm has five parameters - we set these to the values used by Nieto et al. [21] in their comparative evaluation: ρ resolution = 1 pixel, θ resolution = 1 deg, accumulator threshold = 80, minimum line segment length = 30 pixels, maximum gap between points in the same segment = 10 pixels. Matas et al. do not specify a ranking method, however the method is designed to detect the most salient segments first. We therefore use the order in which the algorithm reports the segments as the ranking order.

We obtained the code for the SSWMS method from `(sourceforge.net/projects/lswms`. (The authors renamed the method LSWMS there.) There are two

parameters - we used the author-recommended default values for both (orientation threshold $\Delta\theta = 22.5$ deg and mean shift bandwidth $= 3$ pixels). The SSWMS algorithm is designed to output segments roughly in descending order of salience - we therefore use this order to rank the segments.

We obtained the code for LSD from www.ipol.im/pub/art/2012/gjmr-lsd/. We rank segments using the criterion recommended by the authors and employed in later work [4], namely in increasing order of the number of expected false alarms, which is one of the outputs of the LSD detector.

Fig. 6 shows the top-ranked 90 segments returned by each algorithm on four example images from the YorkUrbanDB test dataset. It is clear that the proposed MCMLSD method tends to return more complete line segments, more evenly distributed over the major structures of the scene. Results for all images in the YorkUrbanDB test dataset can be found in the supplementary material.

Fig. 7 provides a quantitative comparison of the four methods on the YorkUrbanDB test set. On all three measures the proposed MCMLSD method outperforms the other three methods by a significant margin, particularly in the high-recall regime. For example, from Fig. 7(a) we can see that for $k = 90$ returned segments, our MCMLSD method attains a recall score of 0.62, 48% higher than the next-best method LSD. While the proposed MCMLSD method achieves a maximum recall rate of 0.80, the SSWMS and PPHT methods are unable to achieve recall rates higher than 0.5 and LSD tops out at 0.57. From Fig. 7(c) we can see that at its maximal recall rate, the LSD method has a precision of only 0.27, whereas the MCMLSD method has a precision of 0.40 for the same rate of recall, an improvement of 48%.

The MCMLSD algorithm adapts well to different image resolutions as long as the transition probabilities are adjusted correctly (Section 4.2). Fig. 8 shows the top 90 segments returned for an example image from the York UrbanDB dataset at normal (640×480 pixel) and high (1280×960 pixel) resolutions. Note that the algorithm is able to take advantage of the higher resolution to deliver more complete and accurate segments.

(a) 640×480 pixels (b) 1280×960 pixels

Figure 8: Top 90 segments for MCMLSD on an example image at low and high resolutions.

8. Run Time

The dynamic programming solution for line segment detection runs in $O(N) = O(\sqrt{n})$ time, where N is the number of point samples on the line and n is the number of pixels in the image. Given a set of m detected lines, the total time complexity of line segment extraction is $O(m\sqrt{n})$.

Table 2 shows the average run time for the four algorithms tested here on the 640×480 pixel images of the YorkUrbanDB training dataset, using a 3.4 GHz Intel Core i7 with 8GB RAM. The MATLAB implementation of our MCMLSD method has an average run time of 5.2 sec per image. This is slower than the other methods, which are optimized and implemented in C++, and return results within a few hundred milliseconds. About 63% of our run time is taken by the probabilistic Hough method for line extraction [24], which we believe could be sped up considerably with more efficient coding practices and implementation in C or C++. There are also many opportunities for mapping to parallel hardware, as edge detection is dominated by convolutions and in the dynamic programming line segment detection stage lines separated by more than 4 pixels are processed independently.

Table 2: Average number of segments returned and run time per image for the four systems evaluated.

Algorithm	# Segments	Run Time (sec)
PPHT	457	0.076
SSWMS	391	0.30
LSD	537	0.27
MCMLSD	488	5.2

9. Conclusions

We have developed and evaluated a novel method for line segment detection called MCMLSD that combines the advantages of global probabilistic Hough methods for line detection with spatial analysis in the image domain to identify segments. The key insight is that limiting segment search to Hough-detected lines leads naturally to a Markov chain formulation that allows maximum probability solutions to be computed exactly in linear time. Our method also has the advantage that it can detect multiple segments lying on the same line, a common scenario for images of the built environment. This formulation leads to a natural probabilistic measure for ranking segments based upon the sum over point marginals, which maximizes the expected number of correctly labelled points on detected lines.

A second contribution is our new methodology for evaluating line segment detectors on an incomplete labelled dataset. By constraining matches between ground truth and detector output to be 1:1 at the segment level, we show

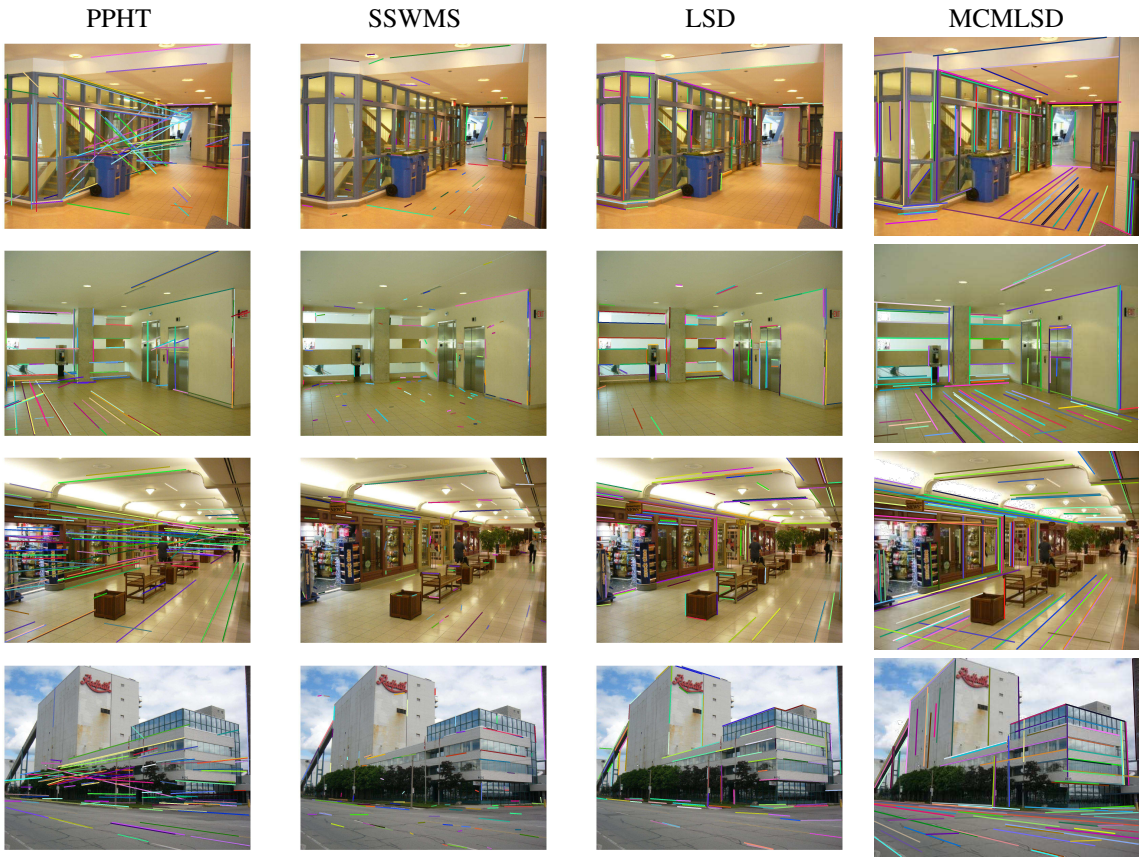

Figure 6: Top 90 segments returned by PPHT, SSWMS, LSD and the proposed MCMLSD method, for four example test images drawn from the YorkUrbanDB dataset.

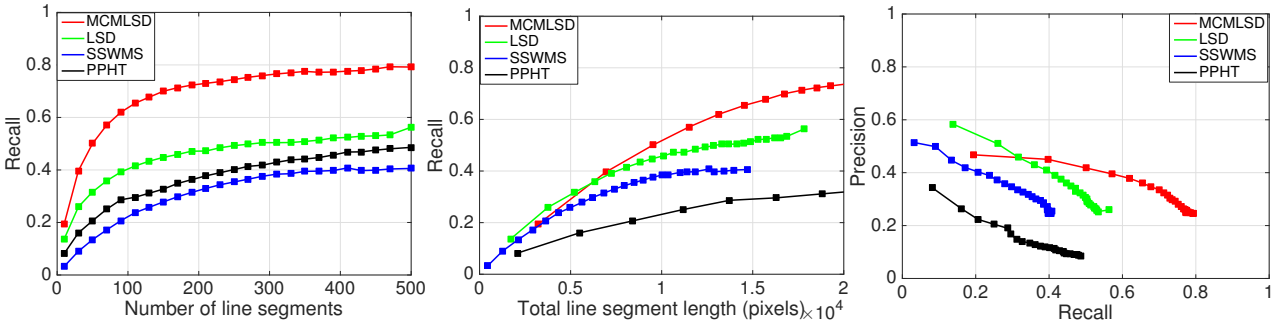

Figure 7: Performance of the proposed MCMLSD methods compared with the state of the art. (a) Recall as a function of number of segments returned. (b) Recall as a function of the total length of segments returned. (c) Precision-Recall.

that under- and over-segmentation are penalized appropriately. Using this new evaluation methodology we find that MCMLSD outperforms the state-of-the-art by a substantial margin. The code for MCMLSD and our evaluation method is available at www.elderlab.yorku.ca/resources.

Acknowledgements

This research was supported by an NSERC Discovery grant and by the NSERC CREATE Training Program in Data Analytics & Visualization.

References

[1] C. Akinlar and C. Topal. Edlines: A real-time line segment detector with a false detection control. *Pattern Recognition Letters*, 32(13):1633–1642, 2011. 2

[2] O. Barinova, V. Lempitsky, and P. Kholi. On detection of multiple object instances using Hough transforms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1773–1784, 2012. 5

[3] M. Boldt, R. Weiss, and E. Riseman. Token-based extraction of straight lines. *IEEE Transactions on Systems, Man and Cybernetics*, 19(6):1581–1594, 1989. 1, 2

[4] M. Brown, D. Windridge, and J. Guillemaut. A generalisable framework for saliency-based line segment detection. *Pattern Recognition*, 48:3993–4011, 2015. 2, 5, 7

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

[6] P. Denis, J. H. Elder, and F. J. Estrada. *Efficient edge-based methods for estimating Manhattan frames in urban imagery*. Springer, 2008. 3, 4, 5

[7] A. Desolneux, L. Moisan, and J.-M. Morel. Meaningful alignments. *International Journal of Computer Vision*, 40(1):7–23, 2000. 1

[8] J. H. Elder and S. W. Zucker. Local scale control for edge detection and blur estimation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(7):699–716, 1998. 1, 3

[9] Y. Furukawa and Y. Shinagawa. Accurate and robust line segment extraction by analyzing distribution around peaks in Hough space. *Computer Vision and Image Understanding*, 92:1–25, 2003. 2

[10] N. Guil, J. Villalba, and E. L. Zapata. A fast Hough transform for segment detection. *Image Processing, IEEE Transactions on*, 4(11):1541–1548, 1995. 2

[11] D. Guru, B. Shekar, and P. Nagabhushan. A simple and robust line detection algorithm based on small eigenvalue analysis. *Pattern Recognition Letters*, 25(1):1–13, 2004. 1

[12] M. Hofer, M. Maurer, and H. Bischof. Efficient 3D scene abstraction using line segments. *Computer Vision and Image Understanding*, 157:167–178, 2017. 1

[13] V. Kamat-Sadekar and S. Ganesan. Complete description of multiple line segments using the Hough transform. *Image and Vision Computing*, 16(9):597–613, 1998. 2

[14] J. Kim and S. Lee. Extracting major lines by recruiting zero-threshold Canny edge links along Sobel highlights. *IEEE Signal Processing Letters*, 22(10):1689–1692, 2015. 2

[15] J. Košecká and W. Zhang. Video compass. In *Computer Vision—ECCV 2002*, pages 476–490. Springer, 2002. 1

[16] H. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955. 5

[17] D. Liu, Y. Wang, Z. Tang, and X. Lu. A robust and fast line segment detector based on top-down smaller eigenvalue analysis. In *Fifth International Conference on Graphic and Image Processing*, pages 906916–906916. International Society for Optics and Photonics, 2014. 1, 2

[18] X. Liu, Z. Cao, N. Gu, S. Nahavandi, C. Zhou, and M. Tan. Intelligent line segment perception with cortex-like mechanisms. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(12):1522–1534, 2015. 1

[19] X. Lu, J. Yao, K. Li, and L. Li. Cannylines: A parameter-free line segment detector. In *Image Processing (ICIP), 2015 IEEE International Conference on*, pages 507–511. IEEE, 2015. 1

[20] J. Matas, C. Galambos, and J. Kittler. Robust detection of lines using the progressive probabilistic Hough transform. *Computer Vision and Image Understanding*, 78(1):119–137, 2000. 2, 6

[21] M. Nieto, C. Cuevas, L. Salgado, and N. García. Line segment detection using weighted mean shift procedures on a 2D slice sampling strategy. *Pattern Analysis and Applications*, 14(2):149–163, 2011. 1, 6

[22] P. Parodi and G. Piccioli. 3D shape reconstruction by using vanishing points. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 18(2):211–217, 1996. 1

[23] C. Schmid and A. Zisserman. Automatic line matching across views. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 666–671. IEEE, 1997. 1

[24] R. Tal and J. H. Elder. An accurate method for line detection and Manhattan frame estimation. In *Computer Vision-ACCV 2012 Workshops*, pages 580–593. Springer, 2013. 2, 3, 5, 7

[25] J.-P. Tardif. Non-iterative approach for fast and accurate vanishing point detection. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1250–1257. IEEE, 2009. 5

[26] R. G. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. LSD: A fast line segment detector with a false detection control. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (4):722–732, 2008. 1, 2, 6

[27] Z. Xu, B. Shin, and R. Klette. A statistical method for line segment detection. *Computer Vision and Image Understanding*, 138:61–73, 2015. 2

[28] Z. Xu, B.-S. Shin, and R. Klette. Accurate and robust line segment extraction using minimum entropy with Hough transform. *IEEE Transactions on Image Processing*, 24(3):813–822, 2015. 2

[29] Z. Xu, B.-S. Shin, and R. Klette. Closed form line-segment extraction using the Hough transform. *Pattern Recognition*, 48(12):4012–4023, 2015. 2

[30] L. Zhang and R. Koch. Structure and motion from line correspondences: representation, projection, initialization and sparse bundle adjustment. *Journal of Visual Communication and Image Representation*, 25(5):904–915, 2014. 1