


# A Reliable Online Method for Joint Estimation of Focal Length and Camera Rotation

Yiming Qian<sup>1</sup>  and James H. Elder<sup>2</sup>

<sup>1</sup> A\*star, Institute of High Performance Computing, Singapore  
qian.yiming@ihpc.a-star.edu.sg

<sup>2</sup> York University, Centre for Vision Research, Canada  
jelder@yorku.ca

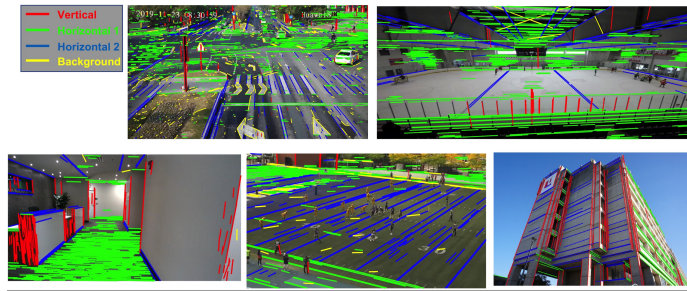
**Abstract.** Linear perspective cues deriving from regularities of the built environment can be used to recalibrate both intrinsic and extrinsic camera parameters online, but these estimates can be unreliable due to irregularities in the scene, uncertainties in line segment estimation and background clutter. Here we address this challenge through four initiatives. First, we use the PanoContext panoramic image dataset [27] to curate a novel and realistic dataset of planar projections over a broad range of scenes, focal lengths and camera poses. Second, we use this novel dataset and the YorkUrbanDB [4] to systematically evaluate the linear perspective deviation measures frequently found in the literature and show that the choice of deviation measure and likelihood model has a huge impact on reliability. Third, we use these findings to create a novel system for online camera calibration we call  $f\mathbf{R}$ , and show that it outperforms the prior state of the art, substantially reducing error in estimated camera rotation and focal length. Our fourth contribution is a novel and efficient approach to estimating uncertainty that can dramatically improve online reliability for performance-critical applications by strategically selecting which frames to use for recalibration.

## 1 Introduction

Online camera calibration is an essential task for applications such as traffic analytics, mobile robotics, architectural metrology and sports videography. While intrinsic parameters can be estimated in the lab, these parameters drift due to mechanical fluctuations. Online estimation of extrinsic camera parameters is crucial for translating observations made in the image to quantitative inferences about the 3D scene. If a fixed camera is employed, extrinsics can potentially be estimated manually at deployment, but again there will be drift due to mechanical and thermal variations, vibrations and wind, for outdoor applications. With longer viewing distances, rotational drift can lead to major errors that may be devastating for performance-critical tasks, such as judging the 3D distance between a pedestrian and a car. For PTZ cameras and mobile applications, camera rotation varies over time. PTZ encoder readings are not always available and become less reliable over time [22], and for mobile applications IMU data

are subject to drift. For all of these reasons, reliable visual methods for online geometric recalibration of camera intrinsics and extrinsics are important.

Coughlan and Yuille [3] introduced an approach to online estimation of 3D camera rotation based on a “Manhattan World” (3-point perspective) assumption, i.e., that a substantial portion of the linear structure in the image projects from three mutually orthogonal directions: one vertical and two horizontal. This assumption can apply quite generally in the built environment and is relevant for diverse application domains including traffic analytics, mobile robotics, architecture and sports videography (Fig. 1).



**Fig. 1.** Example application domains of online camera calibration. We show here labelling of line segments according to Manhattan direction, as determined by our novel  $f\mathbf{R}$  system that simultaneously estimates focal length and camera orientation.

Subsequent work, reviewed below, has built on this idea to improve accuracy and incorporate simultaneous estimation of focal length. However, we have found that these methods vary a lot in accuracy depending upon the scene and whether focal length is known or estimated. This motivates our attempt to better understand how these methods generalize to diverse scenes and knowledge of intrinsics, and to develop methods to estimate uncertainty of individual estimates.

We address these limitations through four contributions: **1)** From the PanoContext dataset [27] we curate a novel and realistic PanoContext- $f\mathbf{R}$  dataset of planar projections over a broad range of scenes, focal lengths and camera poses that can be used to evaluate systems for simultaneous estimation of focal length and camera rotation (tilt/roll). **2)** We use this novel dataset and the YorkUrbanDB [4] to evaluate the linear perspective deviation measures found in the literature and show that a) the choice of deviation measure has a huge impact on reliability, and b) it is critical to also employ the correct likelihood models. **3)** We use these findings to create a novel system for online camera calibration we call  $f\mathbf{R}$ , and show that it outperforms the state of the art. **4)** We develop a novel approach to estimating uncertainty in parameter estimates. This is important for two reasons. First, knowledge of uncertainty can be factored into risk models employed in engineering applications, co-determining actions designed to mitigate this risk. Second, especially for cases where parameters are

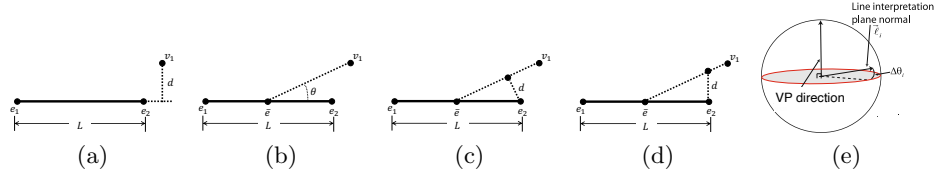
expected to be slowly varying, systems can sparsely select frames that are more likely to generate trustable estimates of camera parameters. The source code and PanoContext- $\mathbf{R}$  dataset are available on GitHub<sup>3</sup>.

## 2 Prior work

### 2.1 Image features and deviation measures

To apply linear perspective cues to camera calibration, two key design choices are 1) what features of the image to use and 2) how to measure the agreement of these features with hypothesized vanishing points. Coughlan and Yuille’s original approach [3] employed a likelihood measure on the angular deviation between the line connecting a pixel to the hypothesized vanishing point (which we will refer to in the following as the *vanishing line* and the ray passing through the pixel in the direction orthogonal to the local luminance gradient.

There have since been diverse attempts to improve on this approach, in large part by using different features and different measures of agreement. Deutscher et al. [5] and Košeckà and Zhang [9] first generalized this approach to allow simultaneous estimation of focal length. While Deutscher et al. retained the gradient field representation of Coughlan and Yuille, in their Video Compass system, Košeckà and Zhang switched to using sparser line segments, and proposed a Gaussian likelihood model based upon the projection distance from the line passing through a line segment to its associated vanishing point (Fig. 2 (a)).



**Fig. 2.** Five deviation measures evaluated (see also [25]): (a) distance from vanishing point to line passing through line segment. (b) angular deviation between line segment and the vanishing line through the line segment midpoint. (c) distance from line segment midpoint to vanishing line (d) distance from line segment endpoint to vanishing line, measured orthogonal to line segment. (e) angle between the interpretation plane normal and the plane orthogonal to the vanishing direction [19].

Rother [16] used line segments as well, employing a heuristic version of the deviation measure similar to Coughlan and Yuille’s (Fig. 2 (b)). Lee et al. [10] followed a similar approach to simultaneous estimation of focal length and camera rotation, and have open-sourced their code, allowing us to evaluate their system below. Denis et al. [4] employed the same angular deviation measure

<sup>3</sup> <https://github.com/ElderLab-York-University/OnlinefR>

for edges, but returned to Coughlan and Yuille’s probabilistic framework, using a learned generalized exponential distribution to model these deviations.

Tardif [20] continued the practice of using line segments, but proposed an alternative deviation measure based on the squared distance between segment endpoints and the vanishing line (Fig. 2 (c)). Wildenauer [21] later adopted the same measure but, similar to Denis et al. [4], returned to Coughlan and Yuille’s probabilistic framework, using a Cauchy distribution to model the signed distance of segment endpoints to the vanishing line.

These three deviation measures were reviewed by Xu et al. [25], who used intuitive arguments to suggest that each of these measures is flawed. Based upon their analysis, they proposed a novel deviation measure that also uses the distance of the segment endpoints to the vanishing line, but measures this distance orthogonal to the *segment*, instead of orthogonal to the vanishing line (Fig. 2 (d)). Their full probabilistic model also favours vanishing points that lie far from the segment midpoint, along the segment direction.

Hough maps can be used to accurately detect lines in an image. Motivated by the early work of Collins & Weiss [2], Tal & Elder [19] proposed a probabilistic Hough method to estimate lines and then estimated camera rotation based on a probabilistic model of deviation in the Gauss sphere, specifically, the angular deviation between the interpretation plane normal and the plane orthogonal to the vanishing direction (Fig. 2(e)).

## 2.2 Benchmarks

The most prevalent dataset used for Manhattan scene analysis is the YorkUrbanDB dataset [4], which does sample a broad range of scenes but is limited in that focal length is fixed and roll and tilt vary over a relatively modest range. Also, the 101 images in the dataset are too few to train a deep network. More recently there have been efforts [8,12] to create datasets large enough to train deep networks by curating random planar projections from existing panoramic image datasets such as SUN360 [24] and Google StreetView. This makes it easy to generate enough images to train a deep network, and also ensures that we have exact ground truth for camera focal length, tilt and roll angle. Unfortunately, there is no ground truth for camera pan, and so networks trained on these datasets are unable to recover the full camera rotation  $\mathbf{R}$ .

## 2.3 State-of-the-Art Systems

In this paper we are focused on the problem of online estimation of focal length  $f$  and camera rotation  $\mathbf{R}$ . Unfortunately, many of the systems reviewed above assumed known focal length, and/or did not open-source their code to allow comparison. The exception, as mentioned, is the line segment method of Lee et al. [10] (Fig. 2 (b)) which we compare against below.

More recently, Simon et al. [18] use the angular deviation measure in Fig. 2(b) but introduce a preprocessing step to estimate the horizon as a guide to estimating Manhattan vanishing points. They have open-sourced their code.

Hold-Geoffroy et al. [8] kicked off the use of deep learning for online camera calibration, using the Sun360 panoramic image dataset [24] to generate planar projections based on random samplings of focal length, horizon, roll and camera aspect ratio. They used this dataset to train, validate and test a network based on a denseNet backbone and three separate heads to estimate the horizon height and angle and the vertical FOV, from which camera focal length, roll and tilt can be derived. While the Hold-Geoffroy et al. [8] dataset has not been released and the method has not been open-sourced, they do provide a web interface for testing their method on user-provided images, allowing us to evaluate and compare their approach on public datasets.

Lee et al. [12] followed a similar approach to generating large datasets (see above) to train a *neural geometric parser*. Their system takes line segments detected by LSD [7] as input, first estimating the zenith vanishing point, selecting segments that are close to vertical, generating vanishing point hypotheses from pairs of these segments, and then feeding the segments and the hypotheses into a deep network to score the hypotheses. High-scoring zenith candidates are then used to generate camera tilt/roll and focal length hypotheses that are combined with the image and Manhattan line maps as input to a second network to generate final camera tilt/roll and focal length estimates. In a more recent paper, Lee et al. [11] have introduced an updated system called CTRL-C that continues to employ LSD line segments but shifts to a transformer architecture for estimating the parameters. While the neural geometric parser has not been open-sourced, CTRL-C has, and we compare with their approach below.

There are other recent deep learning approaches to estimation of vanishing points and/or camera rotation [23,14,13], but these do not estimate focal length.

### 3 *fR* Method

To develop our *fR* method for online camera calibration, we will assume a camera-centred world frame aligned with the Manhattan structure of the scene and employ a standard projection model  $\tilde{\mathbf{x}} = \mathbf{K}\mathbf{R}\tilde{\mathbf{X}}$  where  $\tilde{\mathbf{x}}$  is an image point in homogenous coordinates,  $\mathbf{K}$  is the camera’s intrinsic matrix,  $\mathbf{R}$  is the camera rotation matrix and  $\tilde{\mathbf{X}}$  is a 3D world point in augmented coordinates. Unless otherwise noted, we will assume that any nonlinear distortions in the camera have been calibrated out in the lab, and that after laboratory calibration the camera has a central principal point, square pixels and zero skew, leaving a single intrinsic unknown, the focal length  $f$ . While these assumptions will not be met exactly by real cameras in the field, they are the standard assumptions employed by all of the methods reviewed above and the methods we compare against below, and in our experience are reasonable approximations for most higher-quality modern cameras.

The goal of *fR* then is to estimate the unknown focal length  $f$ , together with the 3D rotation  $\mathbf{R}$  of the camera relative to the Manhattan frame of the scene. Again, we note that this goes beyond the capacity of recent deep learning approaches, which are only able to estimate two of the three rotational degrees

of freedom (roll and tilt). Standard formulae can be used to convert between camera Euler angles (roll, tilt, pan) and the camera rotation matrix  $\mathbf{R}$  [6].

We collect these unknowns into the parameter set  $\Psi = \{f, \mathbf{R}\}$ . Note that  $\Psi$  completely determines the locations of the three Manhattan vanishing points: The  $i$ th vanishing point is given by  $\tilde{\mathbf{x}}_i = \mathbf{K}\mathbf{R}_i$ , where  $\mathbf{R}_i$  is the  $i$ th column of  $\mathbf{R}$ .

In order to better understand the relative advantages of the various deviation measures proposed in the literature, and the role of probabilistic modeling, we will adopt the original mixture model framework first proposed by Coughlan & Yuille [3] for luminance gradients and later extended to line segments [9], edges [4] and lines [19]. We will focus on the use of line segments as features, as there are numerous high-quality open-source line segment detectors now available [7, 1, 15] and line segments have been shown to be effective features for Manhattan scene analysis. We will use the MCMLSD line segment detector [1] for most of the experiments below, but will also evaluate the advantages and disadvantages of alternative detectors.

### 3.1 Probabilistic model

$f\mathbf{R}$  will assume that each line segment  $\mathbf{l}_i$  in the image is generated by a model  $m_i \in M$  corresponding to one of four possible processes: the three Manhattan families of parallel lines or a background process assumed to generate a uniform distribution of lines. Given camera parameters  $\Psi$ , the probability of observing the segment  $\mathbf{l}_i$  is then given by:

$$p(\mathbf{l}_i|\Psi) = \sum_{m_i \in M} p(\mathbf{l}_i|\Psi, m_i)p(m_i) \quad (1)$$

where  $p(m_i)$  is the prior probability that a segment is generated by process  $m_i$  and  $p(\mathbf{l}_i|\Psi, m_i)$  is the likelihood of an observed line generated by  $m_i$  under camera parameters  $\Psi$ .

Since, as noted above, the parameters  $\Psi$  completely determine the vanishing points, the likelihoods  $p(\mathbf{l}_i|\Psi, m_i)$  for Manhattan processes  $m_i$  can be parameterized by one of the non-negative measures of deviation between the line  $\mathbf{l}_i$  and the corresponding Manhattan vanishing point (deviation measures **a-e** reviewed above and summarized in Fig. 2). We will generally use the YorkUrbanDB training partition to fit exponential models  $p(x) = \frac{1}{\lambda} \exp(-x/\lambda)$  for these deviations, and assume that the background process is uniform on this measure - see supplemental material for the fits. The one exception is for deviation measure d, where we employ the central Gaussian model proposed by Xu et al. [25], which also favours vanishing points that lie far from the segment midpoint, along the segment direction. Here the dispersion parameter  $\sigma$  does not have a clear interpretation in terms of the YorkUrbanDB so we use the value of  $\sigma = 1$  pixel recommended by Xu et al. Table 1 summarizes these parameters.

The priors  $p(m_i)$  are also estimated from the YorkUrbanDB training set: 45% of lines are expected to arise from the vertical process, 26% from each of the horizontal processes and 3% from the background process.

**Table 1.** Dispersion parameters for likelihood models.

Deviation measure	Model	Param	Horiz	Vert
a	Exponential	$\lambda$	94.46 pix	17.26 pix
b	Exponential	$\lambda$	1.46 deg	0.57 deg
c	Exponential	$\lambda$	0.39 pix	0.53 pix
d	Gaussian	$\sigma$	1.00 pix	1.00 pix
e	Exponential	$\lambda$	0.80 deg	0.57 deg

Assuming conditional independence over  $n$  observed line segments  $\mathbf{l}_i$  and a flat prior over the camera parameters  $\Psi$ , the optimal solution should be found by maximizing the sum of log likelihoods, however we find empirically that we obtain slightly better results if we weight by line segment length  $|\mathbf{l}_i|$ , solving for

$$\Psi^* = \arg \max_{\Psi} \sum_{i=1, \dots, n} |\mathbf{l}_i| \log p(\mathbf{l}_i | \Psi) \quad (2)$$

We conjecture that the small improvement achieved by weighting by line length may derive from a statistical dependence between line length and the likelihood model: Longer lines may be more accurate.

### 3.2 Parameter search

The  $f\mathbf{R}$  objective function (Eqn. 2) is generally non-convex. We opt for a simple search method that can be used to compare the deviation measures **a-e** summarized in Fig. 2 and to assess the role of probabilistic modeling. The method proceeds in two stages. In Stage 1, we do a coarse  $k \times k \times k \times k$  grid search over the four-dimensional parameter space of  $\Psi$ , evaluating Eqn. 2 at each of the  $k^4$  parameter proposals. Specifically, we sample uniformly over the range  $[-45, +45]$  deg for pan,  $[-15, +15]$  deg for roll,  $[-35, +35]$  deg for tilt and  $[50, 130]$  deg for horizontal FOV. (We also evaluated a RANSAC approach [21] for Stage 1 but found it be less accurate - see supplementary material for results.) We then deploy a nonlinear iterative search (MATLAB fmincon) initialized at each of the the top  $l$  of these  $k^4$  proposals and constrained to solutions within the ranges above. In the following we will use  $k = 8, l = 4$ .

Note that focal length  $f$  and FOV are monotonically related through  $\text{FOV} = 2 \arctan\left(\frac{w}{2f}\right)$ , where  $w$  is the image width. While we search over FOV we will generally convert to focal length when reporting results.

### 3.3 Error prediction

One advantage of a probabilistic approach to online camera calibration is the opportunity to predict when estimates can be trusted. Since we explicitly model the distribution of deviations between line segments and hypothesized vanishing points, we might hope to estimate uncertainty in camera parameters using standard error propagation methods. Unfortunately, we find empirically that this



local method does not lead to very accurate estimates of uncertainty, perhaps because the derivative of the camera parameters with respect to the segment orientations at the estimated parameters  $\hat{\psi}$  is not very predictive of the derivative at the true camera parameters  $\psi^*$  (see supplementary material for details). We therefore propose instead three easily computable global predictors to inform the level of trust that should be invested in estimated camera parameters:

1) **Number of line segments.** We assign each detected segment  $l_i$  to the most likely generating process  $m_i$  under our mixture model (Eqn. 1), count those assigned to each Manhattan direction, and take the minimum, to reflect the intuition that this will be the weakest link in the global parameter estimation process. 2) **Entropy.** This cue takes advantage of the first stage of our parameter search. We conjecture that more reliable parameter estimation will be reflected in a more peaked distribution of likelihoods over the  $k^4$  parameter proposals evaluated. To capture this intuition, we normalize the likelihoods into a discrete probability distribution and compute its entropy. Low entropy distributions are expected to be more reliable. 3) **Likelihood.** We compute the mean log likelihood of the final estimate output from the second nonlinear iterative stage of our search, normalized by the number of line segments.

To predict camera parameter MAE from these cues we fit a KNN regression model on the PanoContext- $fR$  training partition - see Section 5.4 for details.

## 4 Datasets

We evaluate  $fR$  and competing methods on two datasets: the YorkUrbanDB [4] test partition, and our novel PanoContext- $fR$  dataset, curated from [27].

The YorkUrbanDB test partition consists of 51 indoor and outdoor images. Camera focal length  $f$  was fixed and estimated in the lab. Manhattan line segments were hand-labelled to allow estimation of the camera rotation  $R$  [4]. While the camera was handheld, the variation in camera roll and tilt is fairly modest: standard deviation of 0.83 deg for roll and 4.96 deg for tilt.

To allow evaluation over a wider range of camera parameters and to assess how well recent deep networks generalize compared to geometry-driven approaches, we introduce PanoContext- $fR$ , a curation of planar projections from the PanoContext dataset [27], which contains 706 indoor panoramic scenes. We randomly divide this dataset into two equal training and test partitions of 353 scenes each. We will use the test partition to evaluate and compare deviation measures, probabilistic models and state-of-the-art systems for camera parameter estimation. While our  $fR$  system is not a machine learning method and does not require the training partition, we will explore methods for learning to predict estimation error in Section 3.3 and will use the training dataset there.

For each of these scenes we generated 15 planar projections, using a standard  $640 \times 480$  pixel resolution. Three images were sampled for each of 5 fixed horizontal FOVs from 60 to 120 deg in steps of 15 deg. Pan, roll and tilt were randomly and uniformly sampled over  $[-180, +180]$ ,  $[-10, +10]$  and  $[-30, +30]$  ranges respectively (Fig. 3). This generated 5,295 images for each of the training and



test partitions. This dataset is not intended for network training, but rather for evaluating generalization of pre-trained and geometry-driven models.

We sampled FOVs discretely for PanoContext- $f\mathbf{R}$  so we could clearly see whether algorithms are able to estimate focal length (see below). However, we have also created and will make publicly available an alternative PanoContext- $u\mathbf{R}$  that samples horizontal FOV randomly and uniformly over  $[60,120]$  deg. (See supplementary material.) Since the the YorkUrbanDB provides the ground truth rotation matrix, we can evaluate the frame angle error, i.e., the magnitude of the rotation required to align the estimated camera frame with the ground truth frame. Since for PanoContext- $f\mathbf{R}$  we do not have ground truth pan, we report mean absolute roll and tilt errors.



**Fig. 3.** Three example planar projections from our new PanoContext- $f\mathbf{R}$  dataset.

## 5 Experiments

### 5.1 Evaluating deviation measures

We begin by applying the  $f\mathbf{R}$  search method (Section 3.2) to identify the parameters  $\Psi$  maximizing agreement between line segments and vanishing points, based on the five deviation measures **a-e** summarized in Fig. 2. Without probabilistic modeling, we employed the objective function  $\sum_i \log \delta_i$ , where  $\delta_i = d_i, \theta_i, \Delta\theta_i$  depending on which of the five deviation measures is employed. (We found empirically that logging the deviations before summing improved results.). We found that none of the deviation measures yielded good results: Mean focal length error remained above 11% on the York Urban DB and above 34% on the PanoContext- $f\mathbf{R}$  dataset (See Supplementary Material for detailed results.)

Probabilistic modeling of the deviation measure greatly improves results (see Table 2 for results on the PanoContext- $f\mathbf{R}$  dataset and supplementary material for results on the YorkUrbanDB). We note that this represents the learning of only 4 parameters (horizontal and vertical dispersions and priors). Moreover this learning seems to generalize well, as the parameters used to evaluate on the PanoContext- $f\mathbf{R}$  dataset were learned on the YorkUrbanDB training set.

Our second observation is that performance also depends strongly on the deviation measure, even when carefully modeling the likelihood. The large errors produced by measure **a** likely reflect the fact that vanishing points are often

**Table 2.** Evaluation of deviation measures on the PanoContext- $f\mathbf{R}$  training set. Numbers are mean $\pm$ standard error.

Dev. measure	Roll MAE (deg)	Tilt MAE (deg)	Focal length MAE (%)
a	4.53 $\pm$ 0.045	14.2 $\pm$ 0.14	35.6 $\pm$ 0.35
<b>b (<math>f\mathbf{R}</math>)</b>	<b>0.78<math>\pm</math>0.02</b>	<b>1.59<math>\pm</math>0.06</b>	<b>8.4<math>\pm</math>0.26</b>
c	0.90 $\pm$ 0.009	2.26 $\pm$ 0.022	14.1 $\pm$ 0.14
d	0.81 $\pm$ 0.008	1.67 $\pm$ 0.015	10.1 $\pm$ 0.08
e	0.89 $\pm$ 0.009	2.13 $\pm$ 0.021	19.5 $\pm$ 0.19

far from the principal point, resulting generally in very unpredictable deviations from the line passing through an associated line segment. The other measures yield better performance, but we note that deviation measure **e** tends to have higher errors for focal length. We believe this is because the Gauss Sphere measure of deviation suffers from a degeneracy: As focal length tends to infinity, the interpretation plane normals collapse to a great circle parallel to the image plane. Thus a hypothesized vanishing point near the principal point will always generate small deviations, leading to a large likelihood.

This leaves deviation measures **b-d**. Method **d** performs less well on the YorkUrbanDB dataset and method **c** has higher errors on the PanoContext- $f\mathbf{R}$  dataset. Overall, method **b**, the method originally used by Coughlan & Yuille for isophotes [3], Rother for line segments [16] and Denis et al. for edges [4], appears to generate the most consistent performance, and so we adopt this method as our standard  $f\mathbf{R}$  deviation measure for the remainder of the paper.

## 5.2 Evaluating line segment detectors

All of the experiments above employed the MCMLSD line segment detector [1]. Table 3 assesses the sensitivity of our system to this choice by substituting two alternative detectors: The well-known LSD detector [7] and a more recent deep learning detector called HT-LCNN [15]. Performance on the PanoContext- $f\mathbf{R}$  dataset is significantly better using MCMLSD than the more recent HT-LCNN, despite the fact that HT-LCNN is reported [15] to have much better precision-recall performance on the YorkUrban DB and Wireframe [26] datasets.

We believe this discrepancy stems largely from the incompleteness of these datasets, which do not claim to label *all* Manhattan line segments. Methods like MCMLSD, which attempt to detect all line segments (Manhattan and non-Manhattan), thus tend to achieve lower precision scores on these datasets, while deep learning methods like HT-LCNN learn to detect only the labelled segments and thus achieve higher precision scores. Our experiment reveals that this higher precision does not translate into better performance but rather *worse* performance, presumably because HT-LCNN fails to generate Manhattan segments that may be unlabelled but are nevertheless useful for estimating the camera parameters. We proceed with the MCMLSD line segment detector but will return to this choice when considering run-time efficiency (Section 5.5).

**Table 3.** Evaluating the choice of line segment detector on the PanoContext- $\mathcal{f}\mathbf{R}$  training set. Numbers are mean  $\pm$  standard error.

Methods	Run time (sec)	Roll MAE (deg)	Tilt MAE (deg)	Focal length MAE (%)
MCMLSD [1]	4.23	<b><math>0.78 \pm 0.02</math></b>	<b><math>1.59 \pm 0.06</math></b>	<b><math>8.4 \pm 0.26</math></b>
LSD [7]	0.40	$1.23 \pm 0.012$	$3.35 \pm 0.03$	$8.6 \pm 0.11$
HT-LCNN [15]	2.91	$0.93 \pm 0.009$	$1.91 \pm 0.02$	$10.1 \pm 0.14$

### 5.3 Comparison with state of the art

We compare our  $\mathcal{f}\mathbf{R}$  system against the four state-of-the-art systems [10,18,8,11] reviewed in Sec. 2. Three of these [10,18,11] are open-sourced and we downloaded the code. The fourth (Hold-Geoffroy) [8] provides a web interface. We test two versions of the CTRL-C [11] network: CTRL-C S360 was trained on a dataset curated from the SUN360 dataset [24]. CTRL-C GSV was trained on a dataset curated from the Google Street View dataset [12].

Table 4(top) shows results of this comparison on the York UrbanDB test set. We report roll and tilt error instead of frame error, since the deep learning methods (Hold-Geoffroy [8] and CTRL-C [11]) do not estimate pan angle. (See supplemental material for comparison of pan angle error with the methods of Lee et al. [10] and Simon et al. [18]). (The Simon et al. method returned a valid result for only 69% of the York UrbanDB test images and 46% of the PanoContext- $\mathcal{f}\mathbf{R}$  test images; we therefore report their average performance only for these.)

**Table 4.** Performance comparison with SOA on the York UrbanDB (top) and PanoContext- $\mathcal{f}\mathbf{R}$  (bottom) test sets. Numbers are mean  $\pm$  standard error.

YorkUrbanDB	Run time (sec)	Roll (MAE deg)	Tilt MAE (deg)	FOV MAE (%)	Focal len. MAE (%)
Lee[10]	1.13	$0.80 \pm 0.2$	$1.33 \pm 0.2$	$9.2 \pm 1.15$	$11.5 \pm 1.62$
Simon[18]	0.44	$2.63 \pm 0.6$	$11.0 \pm 1$	$11.4 \pm 2.56$	$15.1 \pm 2.08$
Hold-Geoffroy[8]	n/a	$1.47 \pm 0.2$	$7.57 \pm 0.6$	$48.7 \pm 2.11$	$37.7 \pm 7.02$
CTRL-C[11] S360	<b>0.32</b>	$1.38 \pm 0.2$	<b><math>1.09 \pm 0.1</math></b>	$29.0 \pm 2.50$	$24.7 \pm 1.80$
CTRL-C [11] GSV	<b>0.32</b>	$1.37 \pm 0.2$	$3.90 \pm 0.3$	$69.4 \pm 1.66$	$48.8 \pm 0.77$
$\mathcal{f}\mathbf{R}$	6.40	<b><math>0.50 \pm 0.1</math></b>	$1.16 \pm 0.1$	<b><math>3.8 \pm 0.61</math></b>	<b><math>4.6 \pm 0.78</math></b>
Fast- $\mathcal{f}\mathbf{R}$	0.89	$1.13 \pm 0.4$	$1.62 \pm 0.3$	$5.1 \pm 1.04$	$5.7 \pm 1.21$

PanoContext- $\mathcal{f}\mathbf{R}$	Roll MAE (deg)	Tilt MAE (deg)	FOV MAE (%)	Focal len. (MAE %)
Lee[10]	$2.02 \pm 0.02$	$3.35 \pm 0.03$	$13.4 \pm 0.22$	$22.3 \pm 0.22$
Simon[18]	$1.50 \pm 0.02$	$3.18 \pm 0.05$	$32.8 \pm 0.46$	$54.0 \pm 0.53$
Hold-Geoffroy[8]	$1.58 \pm 0.02$	$3.69 \pm 0.04$	$16.1 \pm 0.12$	$20.5 \pm 0.20$
CTRL-C[11] S360	$1.03 \pm 0.01$	$2.19 \pm 0.02$	$8.1 \pm 0.07$	$13.1 \pm 0.13$
CTRL-C [11] GSV	$2.24 \pm 0.02$	$8.98 \pm 0.09$	$17.3 \pm 0.11$	$21.9 \pm 0.31$
$\mathcal{f}\mathbf{R}$	<b><math>0.78 \pm 0.02</math></b>	<b><math>1.59 \pm 0.06</math></b>	<b><math>5.3 \pm 0.15</math></b>	$8.4 \pm 0.26$
Fast- $\mathcal{f}\mathbf{R}$	$0.89 \pm 0.02$	$1.90 \pm 0.07$	$5.4 \pm 0.16$	<b><math>7.6 \pm 0.23</math></b>

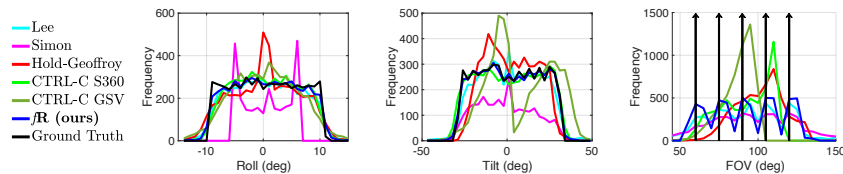
All methods have fairly low error in roll, but remember that in the York UrbanDB, variation in camera roll is limited (standard deviation of 0.83 deg). Our  $\mathbf{fR}$  system improves on the next best method (Lee et al. [10]) by 38%. For tilt estimation, variability in performance across systems is more pronounced, but three systems (our  $\mathbf{fR}$  system, Lee et al. [10] and CTRL-C S360 [11]) all perform well. Focal length estimation is the big differentiator. Our  $\mathbf{fR}$  system performs much better than all of the other systems, beating the next best system (Lee et al. [10]) by 57% (reducing MAE from 11.5% to 4.9%). Notice that the deep learning methods (Hold-Geoffroy [8] and CTRL-C [11]) do not perform well.

Table 4(bottom) compares these systems on our novel PanoContext- $\mathbf{fR}$  dataset. (While FOV and focal length are monotonically related, we show MAE for both for the reader’s convenience.) We find that our  $\mathbf{fR}$  system performs significantly better than all of the other systems on all three benchmarks (roll, tilt and focal length), improving on the next best system (CTRL-C S360 [11]) by 23%, 25% and 33%, respectively. (The relatively strong performance of the CTRL-C S360 system here may derive from the fact that the PanoContext dataset was original drawn from the SUN360 dataset on which CTRL-C S360 is trained.)

Fig. 4 compares distributions of estimated and ground truth parameters for the PanoContext- $\mathbf{fR}$  dataset. The distributions for camera rotation are generally reasonable, although the method of Simon et al. [18] exhibits odd preferences for specific roll and tilt angles, the Hold-Geoffroy system [8] has an overly strong bias to zero roll, and when trained on GSV, the CTRL-C system [11] develops a bias against small upward tilts.

The sampling of five discrete FOVs in the PanoContext- $\mathbf{fR}$  dataset allows us to visualize the sensitivity of the algorithms to focal length. While the geometry-based algorithms all show modulation with the sampled FOVs (albeit weak for Simon et al [18]), the deep learning methods do not, each forming a single broad peak. Our  $\mathbf{fR}$  system appears to be best able to pick up the FOV signal from the data without a strong prior bias.

The ability of the deep learning methods to estimate FOV/focal length might in part be due to limitations in the range of FOVs in their training datasets. However, in the supplementary material we show that this does not fully account for the superiority of our  $\mathbf{fR}$  system, which we believe derives from more direct geometry and probabilistic modeling.



**Fig.4.** Distribution of ground truth and estimated camera parameters for the PanoContext- $\mathbf{fR}$  test sets.

#### 5.4 Predicting Reliability

We employ three global cues to predict camera parameter estimation error: 1) The minimum number of segments over the three Manhattan directions, 2) Entropy over our parameter grid search and 3) mean log likelihood of the final parameter estimate: See supplementary material for a visualization of how parameter error varies as a function of these cues. To estimate the reliability of  $\mathbf{fR}$  estimates at inference, we use the PanoContext- $\mathbf{fR}$  training set to fit a regression model that uses these three cues jointly to predict the absolute error in each of the camera parameters: We collect these cues into a 3-vector, use the training data to compute a whitening transform and then use KNN regression in the 3D whitened space to estimate error, again selecting K by 5-fold cross-validation.

Table 5 assesses the ability of this model to predict error on the held-out PanoContext- $\mathbf{fR}$  test set. The table shows the mean error obtained if we accept only the top 25%, 50%, 75% or 100% of estimates predicted to have least error. The model is remarkably effective. For example, by accepting only the 25% of estimates predicted to be most reliable, mean error declines by 24% for roll, 52% for tilt and 66% for focal length. This ability to predict estimation error can be extremely useful in applications where the camera can be recalibrated on sparse frames, or when conservative actions can be taken to mitigate risks.

**Table 5.** Parameter MAE for 25%, 50%, 75% and 100% of held-out PanoContext- $\mathbf{fR}$  test images predicted to be most reliable.

% of images	Roll MAE (deg)	Tilt MAE (deg)	Focal length MAE (%)
25%	0.56	0.68	2.8%
50%	0.59	0.75	3.9%
75%	0.63	0.86	5.2%
100%	0.74	1.42	8.3%

#### 5.5 Run time

We ran our experiments on a 2.3GHz 8-Core Intel i9 CPU with 32 GB RAM and an NVIDIA Tesla V100-32GB GPU. Our  $\mathbf{fR}$  system as implemented is slower than competing systems (Table 4), primarily due to our line segment detection system MCMLSD, which consumes on average 4.23 sec per image (Table 3).

To address this issue, we have created a more efficient version of our system we call Fast- $\mathbf{fR}$ , through two innovations. First, we replace MCMLSD with LSD, which consumes only 40 msec per image on average. Second, we limit the number of iterations in our second search stage to 10. This reduces average run time from 6.4 sec to 0.83 sec with only a modest decline in accuracy (Table 4).

### 6 Limitations

$\mathbf{fR}$  relies on the Manhattan World assumption, i.e., that images contain aligned rectilinear structure. The superior performance of  $\mathbf{fR}$  on the York Urban DB

and PanoContext- $\mathcal{R}$  datasets, which are representative of common outdoor and indoor built environments, attests to the real-world applicability of this assumption. Nevertheless, the method will typically fail on scenes that do not conform to the Manhattan assumption. This may include images with insufficient structure (see Supplementary Material for examples), nature scenes and non-Manhattan built environments (e.g., the Guggenheim museums in New York or Bilbao). Generalizing the  $\mathcal{R}$  method to exploit additional scene regularities, e.g., Atlanta World [17] or quadrics, is an interesting direction for future research.

Like all SOA methods reviewed and evaluated here, our  $\mathcal{R}$  system assumes no non-linear distortions, a central principal point, square pixels and zero skew. These are reasonable approximations, since deviations from these assumptions can be calibrated out in the factory or lab, and subsequent drift of these parameters is typically minor relative to changes in focal length and rotation.

While the Manhattan constraint can also be used to estimate the principal point, we find that more accurate estimates of focal length and rotation are obtained by assuming a central principal point. Simultaneous online estimation of focal length, principal point and camera rotation, as well as radial distortion, thus remains an interesting and challenging direction for future research.

## 7 Conclusions

We have employed existing and novel datasets to assess the reliability of online systems for joint estimation of focal length and camera rotation. We show that the reliability of geometry-driven systems depends profoundly on the deviation measure employed and accurate probabilistic modeling. Based on these findings, we have proposed a novel probabilistic geometry-driven approach called  $\mathcal{R}$  that outperforms four state-of-the-art competitors, including two geometry-based systems and two deep learning systems. The  $\mathcal{R}$  advantage is most pronounced for estimation of focal length, where the deep learning systems seem to struggle. We note that the deep learning systems also do not predict pan angle, and so do not fully solve for the camera rotation. We further demonstrate an ability to estimate the reliability of specific  $\mathcal{R}$  predictions, leading to substantial gains in accuracy when systems can choose to reject parameter estimates judged to be unreliable. We believe this will be useful in many real-world applications, particularly in mobile robotics and autonomous vehicles.

## 8 Acknowledgement

This work was supported by the University of Toronto, NSERC, the Ontario Research Fund, the York University VISTA and Research Chair programs (Canada), and the Agency for Science, Technology and Research (A\*STAR) under its RIE2020 Health and Biomedical Sciences (HBMS) Industry Alignment Fund Pre-Positioning (IAF-PP) Grant No. H20c6a0031 and AI3 HTP0 Seed Fund (C211118014) (Singapore).

## References

1. Almazan, E.J., Tal, R., Qian, Y., Elder, J.H.: MCMLSD: A dynamic programming approach to line segment detection. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 2031–2039 (2017) [6](#), [10](#), [11](#)
2. Collins, R.T., Weiss, R.S.: Vanishing point calculation as a statistical inference on the unit sphere. In: International Conference on Computer Vision. vol. 90, pp. 400–403. Citeseer (1990) [4](#)
3. Coughlan, J.M., Yuille, A.L.: Manhattan world: Compass direction from a single image by Bayesian inference. In: International Conference on Computer Vision. vol. 2, pp. 941–947. IEEE (1999) [2](#), [3](#), [6](#), [10](#)
4. Denis, P., Elder, J.H., Estrada, F.J.: Efficient edge-based methods for estimating manhattan frames in urban imagery. In: European Conference on Computer Vision. pp. 197–210. Springer (2008) [1](#), [2](#), [3](#), [4](#), [6](#), [8](#), [10](#)
5. Deutscher, J., Isard, M., MacCormick, J.: Automatic camera calibration from a single Manhattan image. In: European Conference on Computer Vision. pp. 175–188. Springer (2002) [3](#)
6. Faugeras, O.: Three-Dimensional Computer Vision: A Geometric Viewpoint. MIT Press, Cambridge, MA (1993) [6](#)
7. von Gioi, R.G., Jakubowicz, J., Morel, J.M., Randall, G.: Lsd: A fast line segment detector with a false detection control. IEEE Transactions on Pattern Analysis & Machine Intelligence (4), 722–732 (2008) [5](#), [6](#), [10](#), [11](#)
8. Hold-Geoffroy, Y., Sunkavalli, K., Eisenmann, J., Fisher, M., Gambaretto, E., Hadap, S., Lalonde, J.F.: A perceptual measure for deep single image camera calibration. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 2354–2363 (2018) [4](#), [5](#), [11](#), [12](#)
9. Košecká, J., Zhang, W.: Video compass. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) European Conference on Computer Vision. pp. 476–490. Springer Berlin Heidelberg, Berlin, Heidelberg (2002) [3](#), [6](#)
10. Lee, D.C., Hebert, M., Kanade, T.: Geometric reasoning for single image structure recovery. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 2136–2143. IEEE (2009) [3](#), [4](#), [11](#), [12](#)
11. Lee, J., Go, H., Lee, H., Cho, S., Sung, M., Kim, J.: CTRL-C: Camera calibration transformer with line-classification. In: International Conference on Computer Vision. pp. 16228–16237 (2021) [5](#), [11](#), [12](#)
12. Lee, J., Sung, M., Lee, H., Kim, J.: Neural geometric parser for single image camera calibration. In: European Conference on Computer Vision. pp. 541–557. Springer (2020) [4](#), [5](#), [11](#)
13. Li, H., Chen, K., Kim, P., Yoon, K.J., Liu, Z., Joo, K., Liu, Y.H.: Learning icosahedral spherical probability map based on bingham mixture model for vanishing point estimation. In: International Conference on Computer Vision. pp. 5661–5670 (2021) [5](#)
14. Li, H., Zhao, J., Bazin, J.C., Chen, W., Liu, Z., Liu, Y.H.: Quasi-globally optimal and efficient vanishing point estimation in Manhattan world. In: International Conference on Computer Vision. pp. 1646–1654 (2019) [5](#)
15. Lin, Y., Pintea, S.L., van Gemert, J.C.: Deep hough-transform line priors. In: European Conference on Computer Vision. pp. 323–340. Springer (2020) [6](#), [10](#), [11](#)
16. Rother, C.: A new approach to vanishing point detection in architectural environments. Image and Vision Computing **20**(9-10), 647–655 (2002) [3](#), [10](#)



17. Schindler, G., Dellaert, F.: Atlanta world: An expectation maximization framework for simultaneous low-level edge grouping and camera calibration in complex man-made environments. In: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004. vol. 1, pp. I–I. IEEE (2004) [14](#)
18. Simon, G., Fond, A., Berger, M.O.: A Simple and Effective Method to Detect Orthogonal Vanishing Points in Uncalibrated Images of Man-Made Environments. In: Bashford-Rogers, T., Santos, L.P. (eds.) EuroGraphics - Short Papers. The Eurographics Association (2016). <https://doi.org/10.2312/egsh.20161008> [4](#), [11](#), [12](#)
19. Tal, R., Elder, J.H.: An accurate method for line detection and Manhattan frame estimation. In: Park, J.I., Kim, J. (eds.) Asian Conference on Computer Vision-Workshops. pp. 580–593. Springer Berlin Heidelberg, Berlin, Heidelberg (2012) [3](#), [4](#), [6](#)
20. Tardif, J.P.: Non-iterative approach for fast and accurate vanishing point detection. In: IEEE International Conference on Computer Vision. pp. 1250–1257. IEEE (2009) [4](#)
21. Wildenauer, H., Hanbury, A.: Robust camera self-calibration from monocular images of Manhattan worlds. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 2831–2838. IEEE (2012) [4](#), [7](#)
22. Wu, Z., Radke, R.: Keeping a pan-tilt-zoom camera calibrated. Pattern Analysis and Machine Intelligence, IEEE Transactions on **35**(8), 1994–2007 (Aug 2013). <https://doi.org/10.1109/TPAMI.2012.250> [1](#)
23. Xian, W., Li, Z., Fisher, M., Eisenmann, J., Shechtman, E., Snavely, N.: Upright-Net: geometry-aware camera orientation estimation from single images. In: International Conference on Computer Vision. pp. 9974–9983 (2019) [5](#)
24. Xiao, J., Ehinger, K.A., Oliva, A., Torralba, A.: Recognizing scene viewpoint using panoramic place representation. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 2695–2702. IEEE (2012) [4](#), [5](#), [11](#)
25. Xu, Y., Oh, S., Hoogs, A.: A minimum error vanishing point detection approach for uncalibrated monocular images of man-made environments. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 1376–1383 (2013) [3](#), [4](#), [6](#)
26. Xu, Z., Shin, B., Klette, R.: A statistical method for line segment detection. Computer Vision and Image Understanding **138**, 61–73 (2015) [10](#)
27. Zhang, Y., Song, S., Tan, P., Xiao, J.: PanoContext: A whole-room 3D context model for panoramic scene understanding pp. 668–686 (2014) [1](#), [2](#), [8](#)