Additional Implementation Details for Single Image 3D Without a Single 3D Image

David F. Fouhey[1], Wajahat Hussain[2], Abhinav Gupta[1], Martial Hebert[1]
[1] Robotics Institute, Carnegie Mellon University, USA
[2] University of Zaragoza, Spain

Purpose: This supplement aims to offer additional details on the technique. In turn, we cover: (1) Rectification (2) The rectification method (3) Bookkeeping issues (4) The box prior (5) Inference.

Metric Rectification Overview

The procedure heavily relies on *rectified* images, or images that have been warped so that they appear as if viewed head-on from a particular direction. These rectified images are related to the original image by a homography $\mathbf{H}$, and thus both pixels as well as detections can be put in correspondence with one another.

Metric Rectification Method

We produce metric rectifications of the scene facing a particular vanishing point using the method of Zaheer, Rashid, and Khan to generate the homographies. The method gives a homography for metric rectification under the assumption of square pixels and known principal point but leaves behind an ambiguity consisting of a similarity transform (i.e., in-plane rotation, isotropic scale, translation). In order to use it, we need to be able to determine three parameters – one for focal length, and two corresponding to rotation angles.

Similarity Ambiguity: We handle the remaining similarity in a brute-force manner by multi-scale detection (handling translation and isotropic scaling) and by rotating scenes consistently (i.e., ensuring the vertical vanishing point is above/below the image).

Parameters: We determine the free parameters of transformation using the mutual-orthogonality of the vanishing points' corresponding directions by ensuring: (1) that the surface normals corresponding are orthogonal as possible given the assumed principal point and (2) that rays drawn from the vanishing points not being rectified (i.e., corresponding to orthogonal normals) form right angles in the rectified image. Essentially, this is the technique of Zaheer et al., but using virtual lines as opposed to detected lines.

Metric Rectification Bookkeeping Issues

Consider looking down a corridor that is 300m long and 3m high. If we obtained a metric rectification of the corridor wall, it would have a 100:1 aspect ratio and most of the rectified image would correspond to a handful of pixels corresponding to the end of the corridor. In fact, the vast majority of the pixels would be compressed to the small part of the rectified wall corresponding to the first few meters of the corridor.

This is an issue because if we are not careful, most of the rectified image will be the upsampling of a handful of pixels, and the place where we have good image data will be squashed. We handle this as follows. This can be observed in Fig. 1, where the rectified fragments further from the camera have much less detailed compared to the ones closer.

- *Training time.* Since we are interested in learning good visual elements and uninterested in bookkeeping, we use one large fragment thus removing parts of the image that are close to the vanishing line of the homography, whch have too little high-frequency texture.
- *Test time.* Our inference technique is simply the accumulation of detections. Thus, we can break the image into chunks. Intuitively, we split the corridor into the part from 0-15m, the part from 10m-25m, the part from 20m-35m, etc. and rectify each individually. This produces a set of overlapping fragments. We can then run detectors on these fragments, and max-pool the detection scores over them when we warp back. We use 5 fragments extending across the image

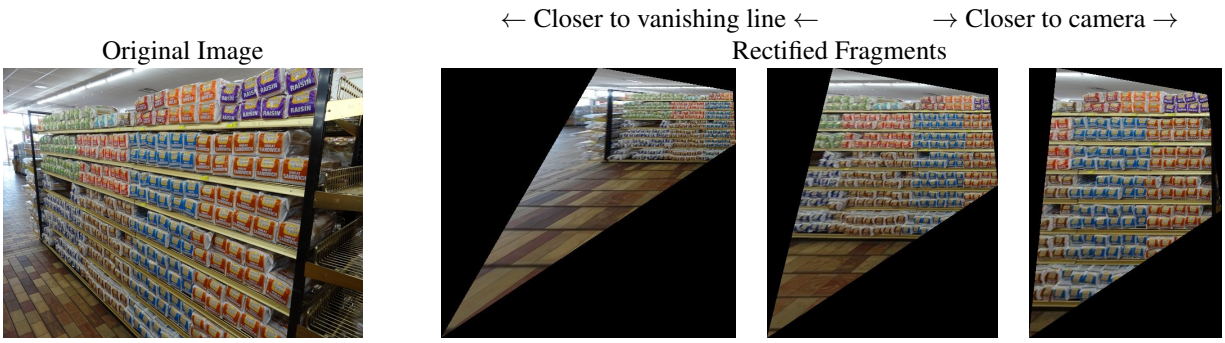
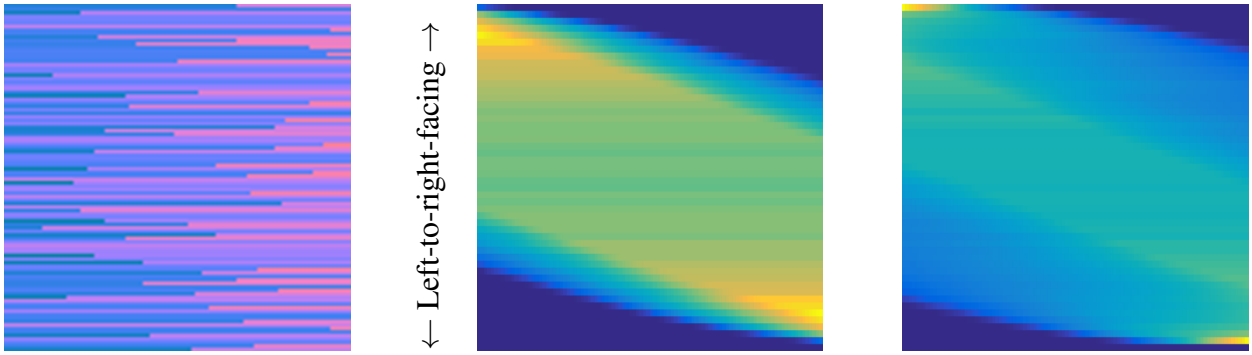


Figure 1. Sample rectifications – it may help to imagine if the shelf extended all the way to the bottom (in which case there would be a great deal of shelving below visible at this angle). Each fragment is related to the original image by a homography and thus detections can be easily accumulated in the rectified space.



(a) Samples (b) Histograms

Figure 2. An illustration of the prior. (a) 100 samples from the distribution. Each sample is a scanline, and we simply display the stacked scanlines. Some scanlines are all one color, when the camera looks at the a wall; some scanlines have a discontinuity, which corresponds to the corner. (b) Two joint histograms of the distribution (left-to-right normal) as a function of the image position (left-to-right image location) for two possible aspect ratios. The y axis represents 50 bins of the x surface normal component; the x axis represents the x-location in the image.

for vertical directions and 1 for horizontal ones because we find texture structure tends to be absent in far horizontal regions.

The more involved bookkeeping gave a modest boost compared to just cutting out the far parts of the image ($\approx 1\%$ good pixels for each threshold), but was not crucial to the method.

Box Prior

Generating the vertical prior: As described in the paper, we determine this box prior via a Monte-Carlo method.

We first describe how we generate a sample. We sample an aspect ratio $\sim$ Uniform$[1, 2]$ and a rotation $\theta \sim [0°, 360°)$. We can then render what vertical directions the camera would see if placed in a rectangular box with the given aspect ratio and rotation. This is done by pretending the camera has a finite pixel array, shooting rays out of the center, and checking which bounding wall segment the rays hit. Including blank lines, comments, and code to plot the sample, this procedure is < 100 lines of unoptimized MATLAB.

Given a large number of samples, we can compute a histogram of the x direction of surface normals as a function of of the u location in the image. We use the histogram at inference time for our prior, and we can compute the expected direction from this histogram as well.

We illustrate these qualitatively in Fig. 2. In Fig. 2(a), we show 100 samples from the prior, represented as scanlines. In Fig. 2(b) we show two settings for the aspect ratio value and how it affects the joint histogram of the surface normal

(left-to-right) and the position in the image. Assuming a more square room precludes extremely-left-facing surfaces towards the middle of the room.

Misc details: Here, we report miscellaneous implementation details. In practice, we did not find the particulars of any of these details to be important.

- *Detections:* We cutoff detections at an ELDA threshold of 0.4 as suggested in the ELDA code we used. We auto-reject detections that are 5% or more outside the image (i.e., the black bits of the rectified images) to prevent the detector from picking up on the strong borders where known image data transitions to fill-in black data.
- *Vertical elements:* We use an array of pixels of size 100 and assume that the camera has a viewing angle of 44°, as was done in Photo Pop-Up of Hoiem et al. Simply averaging would suffer from data scarcity issues (on average 10 detections per pixel), so we take a Nadarya-Watson estimator instead. We used a squared exponential kernel and set the bandwidth to 20% of the image width.
- *Horizontal elements:* Our top detections used for purity-testing are either the top 10, or the detections with ELDA score greater than 0.8, whichever is largest. For correlation between x and orientation, to get a continuous variable, we use the y coordinate as a proxy (i.e., upwards facing at bottom). We fuse these criteria by converting them to ranks across all elements and summing them to eliminate any scaling or non-linearity issues.

Inference

We help illustrate what goes on inside the method during inference. We show probability maps per direction (right-facing: blue; left-facing: pink) for both the *likelihood* and the *likelihood plus the prior* in Fig. 3

1. The likelihood comes from detections in the rectified images. Areas with zeros exist because we clip detections that straddle the border when the image is rectified.
2. The likelihood plus prior also incorporates the box prior. The results are very similar. Two differences are that the results are: (a) smoother because many of the strong predictions are based off of small amounts of evidence (i.e., one or two detections); and (b) tend to match the box model more closely.

Misc-details:

- *Prior strength:* Of course, you have to set a trade-off between the prior and the likelihood. If we interpret this as a multinomial over the directions, the conjugate prior is equivalent to prior observations. The question is how many "observations" you get. We set this to 10 ELDA detection scores for the vertical task since the likelihood you get is quite strong, and 50 ELDA detections for the horizontal-vertical task since the evidence is weak (recall: floors tend to be very low-texture). *However, our analysis in the supplement shows that performance doesn't vary much if you wiggle these parameters.* In fact, there are better parameters.
- *ELDA calibration:* In typical detection tasks, you need to calibrate ELDA models. In practice, however, we found this to be unnecessary for our problem. This is likely due to the fact that we're voting and not doing NMS across elements: thus, a miscalibrated element will generally be overruled by the votes of its peers.

Once we get these probability maps, we simply take argmax to get the label. We can also use the fraction of the probability mass as a sense of confidence; we found this to be well-correlated with performance.

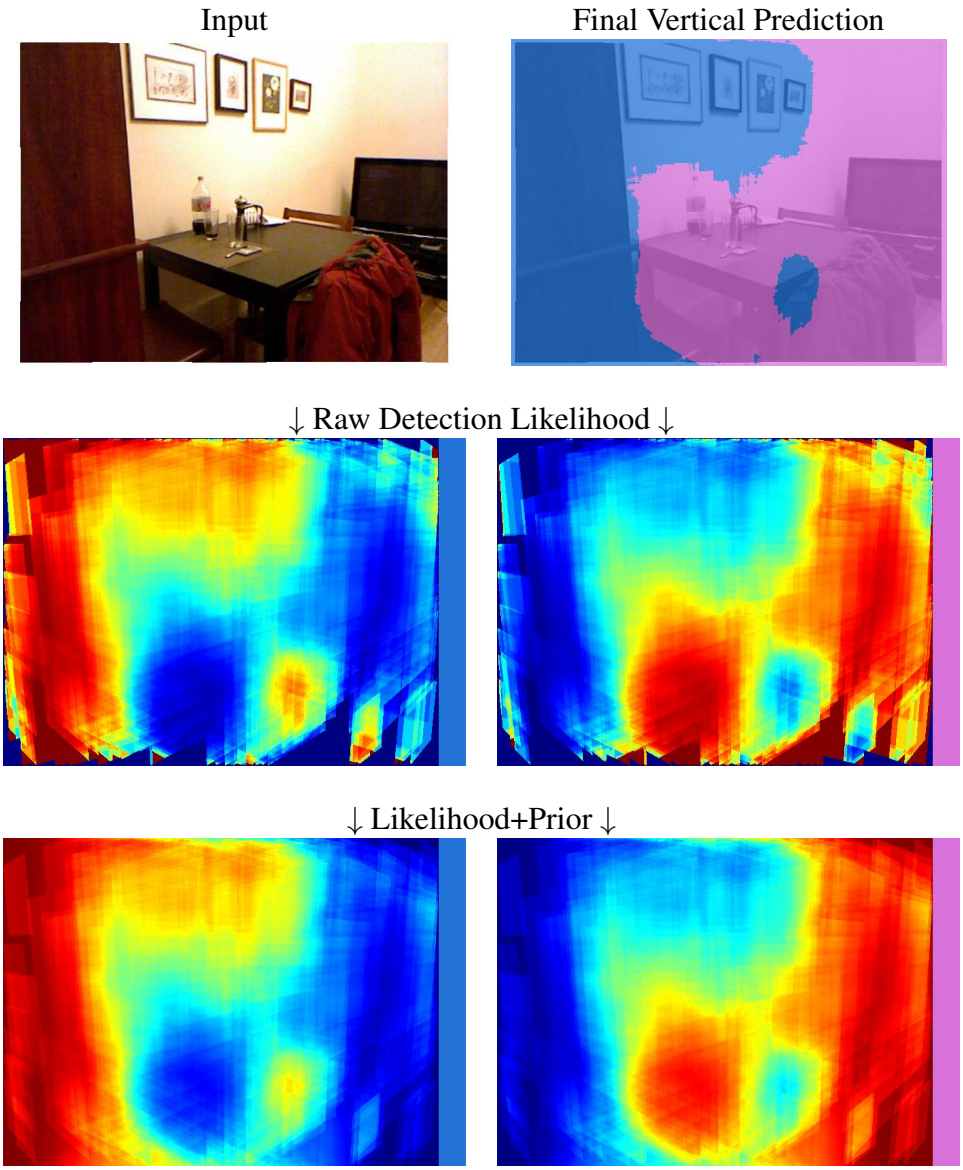


Figure 3. Probability maps over the image per direction