Deep Single Image Camera Calibration with Radial Distortion

Supplementary material

A. Reprojection loss

Initially, we experimented with another loss, which we call reprojection loss. This type of loss was directly inspired by the reprojection error which is common in many 3D computer vision problems.

First, a regular grid of points in pixel coordinates $\boldsymbol{x}_1 \dots \boldsymbol{x}_n$ spanning the extent of the image is projected onto the unit sphere using the ground truth parameters $\boldsymbol{\Omega}$, resulting in a set of 3D points which we refer to as *bearings* $\boldsymbol{p}_1 \dots \boldsymbol{p}_n$. Then, the predicted parameters $\boldsymbol{\Omega}'$ are used to project the bearings back to the image plane, yielding points $\boldsymbol{x}'_1 \dots \boldsymbol{x}'_n$. The distance between the original grid and the reprojected points is the error to be minimized. This error is directly measured in pixel coordinates.

The reprojection step from $\boldsymbol{p}_1 \dots \boldsymbol{p}_n$ to $\boldsymbol{x}'_1 \dots \boldsymbol{x}'_n$ is differentiable and can be optimized using gradient-based techniques. An illustration of a simplified version of this process using only two parameters f and θ is depicted in Fig. 1.

Unlike the bearing loss presented in the main paper, there is no need for differentiable undistortion, since only the ground truth grid needs undistortion. However, the reprojection loss presents a behavior that is highly problematic for training: Depending on the combination of $\boldsymbol{\Omega}$ and $\boldsymbol{\Omega}'$, points $\boldsymbol{x}'_1 \dots \boldsymbol{x}'_n$ may be projected very far away from the original grid $\boldsymbol{x}_1 \dots \boldsymbol{x}_n$ as shown in Figure 2, creating outliers with large gradients that destabilize the learning process. We tried to solve this problem by replacing the squared distance metric by several robust metrics, as well as clipping of the distances. Such a strategy complicates matters as at least one new hyperparameter is required (i.e. a threshold for clipping or a scaling factor for robust functions). Ultimately, even with these measures, the bearing loss presented in the main paper converged faster than the reprojection loss.

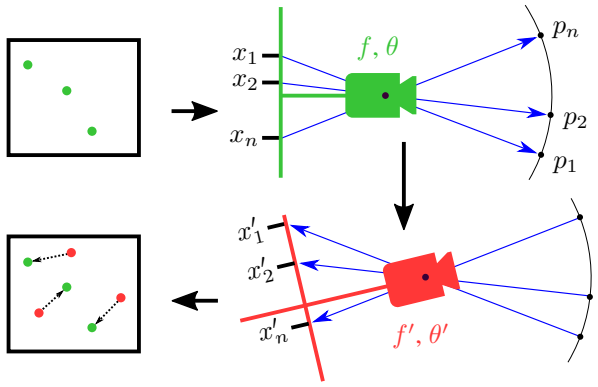

Figure 1. A simplified depiction of the reprojection loss using only two parameters (tilt θ and focal length f). Points $\boldsymbol{x}_1 \dots \boldsymbol{x}_n$ in 2D image coordinates are projected onto 3D points $\boldsymbol{p}_1 \dots \boldsymbol{p}_n$ on a unit sphere using the ground truth camera parameters f, θ. The points are then reprojected onto the image plane using the parameters f', θ' predicted by the model. The average squared distance from each point x_i to its reprojection x'_i is the reprojection loss.

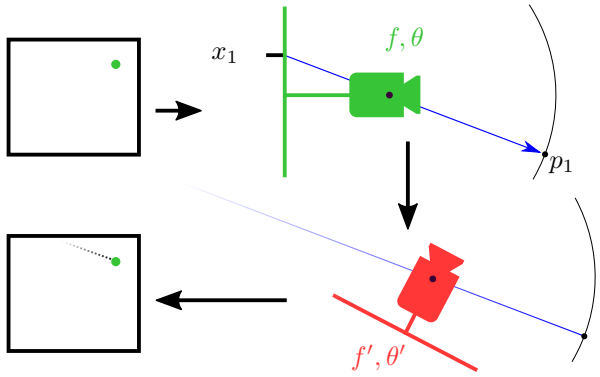

Figure 2. With the reprojection loss, points may be projected infinitely far away depending on the combination of the ground truth $\boldsymbol{\Omega}$ and estimated $\boldsymbol{\Omega}'$ parameters, producing large gradients that unstabilize learning.

B. Learned vs. Geometric-based Undistortion

To our knowledge, our method is the first to include distortion correction from a single image for projective cameras in the wild. Previous learning-based techniques either focused on fisheye distortion [1] or relied on datasets of images containing a sufficient amount of line segments with a specific length [2]. In this context, we focus on comparing the performance of our method with respect to plumb-line methods, that represent a classic and geometric-based solution to single-image undistortion [3, 4, 5].

Plumb-line methods estimate lens distortion based on the curvature of straight lines in the image [4]. Common steps of this family of algorithms are: 1. sub-pixel edge detection, 2. extraction of segment candidates, 3. optimization loop to estimate the distortion coefficients. Although potentially very precise, these methods struggle in scenes where straight lines are hard to detect or to discern from other sources of strong gradients, as in Figure 3. Moreover, the segment detection procedures usually require a careful tuning of several parameters. Instead of following an image processing approach, learning-based methods can detect subtler lines, as well as other indicators of radial distortion that might not be straight lines. For numerical evidence, we compare our method with a state of the art plumb-line algorithm by Santana-Cedrés et al. [6].

Since [6] uses a parameterization for radial distortion different to ours, we compare the methods by the photometric mean squared error with respect to images undistorted using the ground truth coefficients [7]. We perform this comparison on our test set. The plumb-line algorithm also expects input images to have a higher resolution, so instead of scaling to 224×224 pixels as required by our network, we feed it with images of 712 pixels in the shortest side.

We obtain a lower MSE in 89% of the images in the test set, but notice differences depending on the category of the source panorama. As shown in Figure 4, for outdoor images with few or no line segments (nature landscapes or open spaces with trees/monuments, e.g. *beach, forest, plaza* categories), our method performs best in more than 90% of the images. The difference narrows down for indoor and urban imagery, with our method outperforming [6] in 70-90% of the cases, depending on the category. This is expected as there are more line segments in images from these classes that the plumb-line algorithm can rely on (e.g. *office, restaurant, street* categories).

In terms of speed, the runtime for plumb-line methods depends on the number of segments that are detected, while methods based on CNNs have a constant execution time. We benchmark both methods' runtime on an Intel E5-2690v3 CPU and report an average runtime of 10.04s per image for the plumb-line method, while our method takes 0.33s per image. Our runtime is reduced to 1ms per image with a NVIDIA K80 GPU.

C. Qualitative results

We show qualitative results for our method on a set of images in Figure 5. These were not randomly selected but intended to showcase a variety of interesting examples, as we have already reported about the quantitative evaluation on the full test set in the main paper. Images on the top row have been downloaded from Mapillary and were taken using real cameras, while those on the bottom row are from our test set, generated by cropping and distorting panoramas from SUN360 [8].

Each image is fed to the network to obtain the predictions of the proxy parameters $(\hat{k}_1, \rho, \psi, F_v)$ from which we recover the original parameters $(k_1, k_2, \theta, \psi, f)$. These are used to undistort each image and to overlay a horizon line.

References

[1] X. Yin, X. Wang, J. Yu, M. Zhang, P. Fua, and D. Tao, "FishEyeRecNet: A Multi-Context Collaborative Deep Network for Fisheye Image Rectification," pp. 1–16, 2018. [Online]. Available: http://arxiv.org/abs/1804.04784 ii

[2] J. Rong, S. Huang, Z. Shang, and X. Ying, "Radial Lens Distortion Correction Using Convolutional Neural Networks Trained with Synthesized Images," in *Computer Vision – ACCV 2016*, S.-H. Lai, V. Lepetit, K. Nishino, and Y. Sato, Eds. Cham: Springer International Publishing, 2017, pp. 35–49. ii

[3] D. C. Brown, "Close-range camera calibration," *Photogrammetric Engineering*, vol. 37, no. 8, pp. 855–866, 1971. ii

[4] F. Devernay and O. Faugeras, "Straight lines have to be straight," *Machine vision and applications*, vol. 13, no. 1, pp. 14–24, 2001. ii

[5] D. Gonzalez-Aguilera, J. Gomez-Lahoz, and P. Rodríguez-Gonzálvez, "An automatic approach for radial lens distortion correction from a single image," *IEEE Sensors journal*, vol. 11, no. 4, pp. 956–965, 2011. ii

[6] D. Santana-Cedrés, L. Gomez, M. Alemán-Flores, A. Salgado, J. Esclarín, L. Mazorra, and L. Alvarez, "An iterative optimization algorithm for lens distortion correction using two-parameter models," *Image Processing On Line*, vol. 6, pp. 326–364, 2016. ii, iii

[7] R. Szeliski, "Prediction error as a quality metric for motion and stereo," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, Sep. 1999, pp. 781–788 vol.2. ii

[8] J. Xiao, K. A. Ehinger, A. Oliva, and A. Torralba, "Recognizing scene viewpoint using panoramic place representation," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 2695–2702. ii

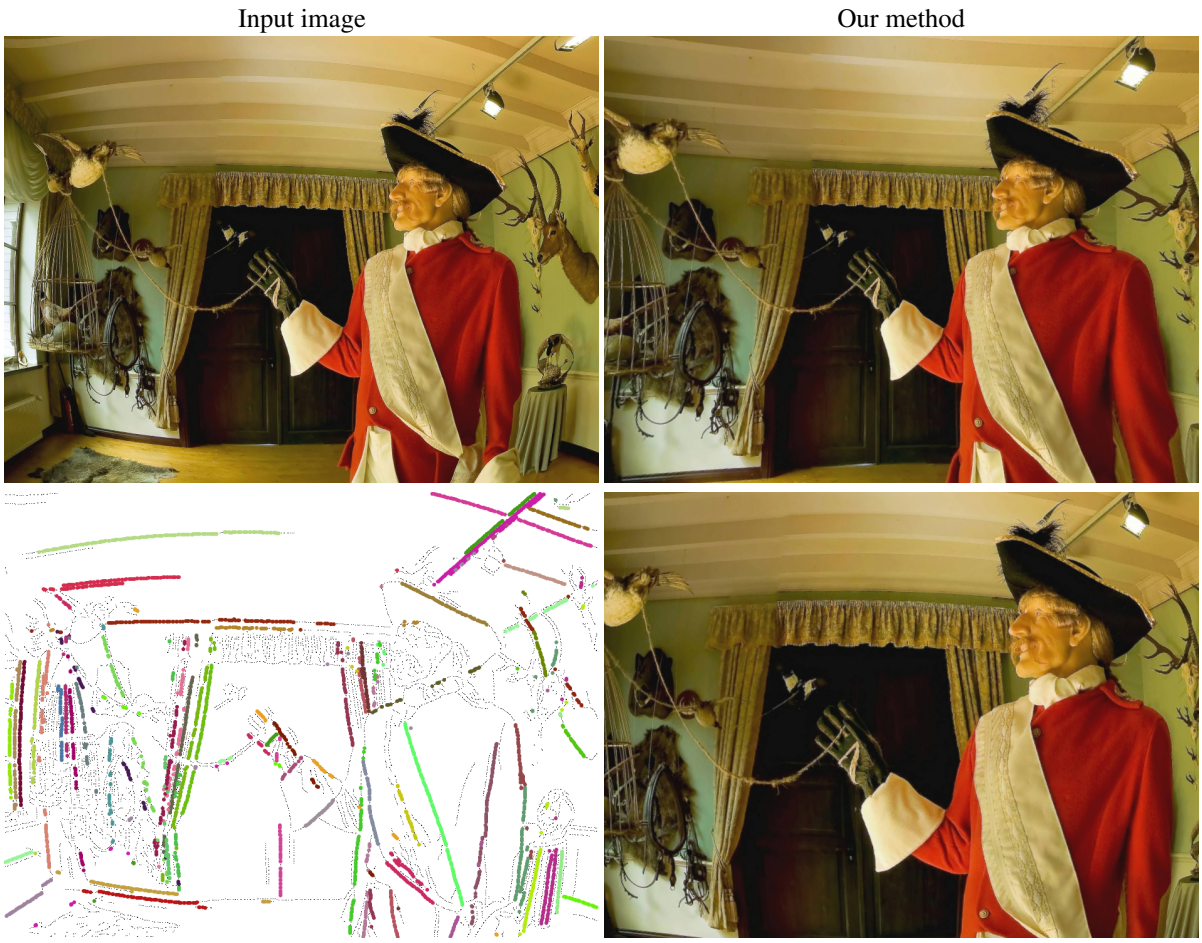

Figure 3. Plumb-line methods optimize the distortion parameters based on the curvature of segments that belong to straight lines in the 3D world. Robustly identifying such segments is not trivial: Here, the most relevant lines for the task are the ceiling beams, which remain undetected due to their low contrast. In parallel, the curtain folds and the pirate's sash, less significant, are detected as their edges are sharp.

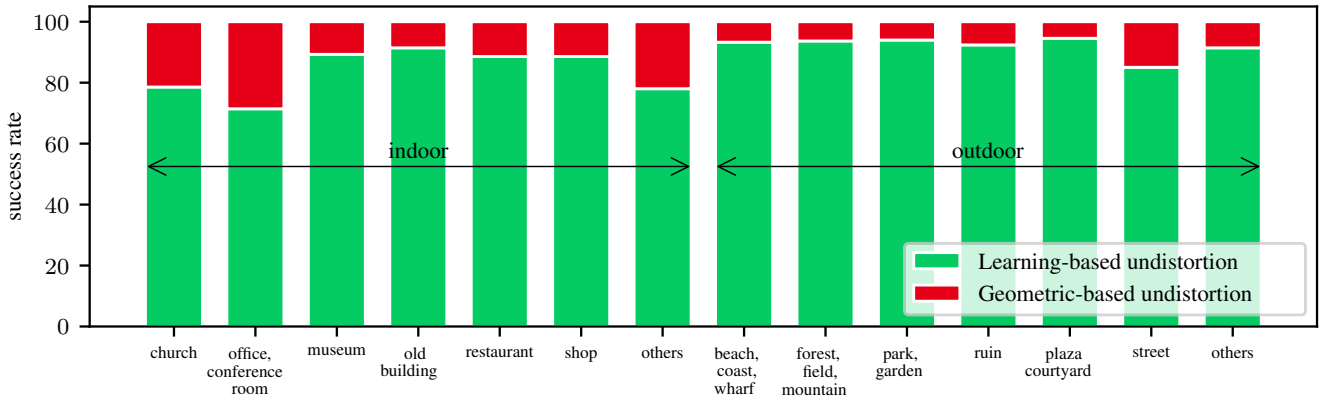

Figure 4. A comparison between learned and geometric-based single image undistortion in the wild, as evaluated in our test images generated from SUN360. Each bar represents the percentage of samples per method that achieved the lowest MSE with respect to the ground truth undistorted images. Certain categories depicting similar scenarios were merged to balance the number of samples per bar.

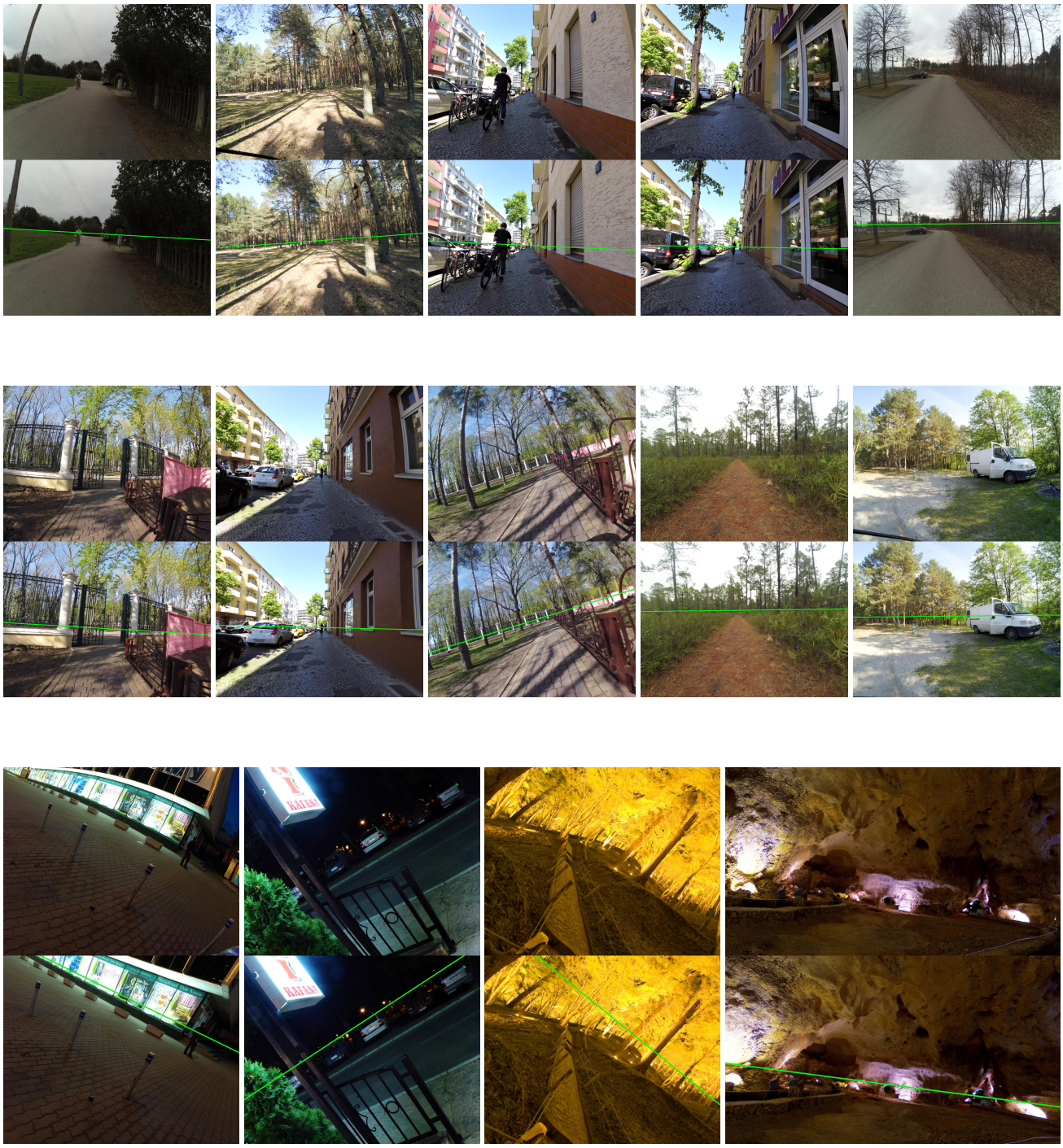

Figure 5. Qualitative results of our undistortion and tilt/roll estimation. Images in the first and second rows were collected from Mapillary and are from real cameras (not strictly following our distortion parametrization), while images in the third row are from our test set. For each image, the original version that is used as input to the network is displayed on the top, and the result of undistorting it using the predicted distortion parameters is shown on the bottom. We also overlay a horizon line in green using the rest of the predicted parameters.