Camera Autocalibration Methods Using Modern Approaches in Deep Learning

Ihor Leshchyshyn
ihorlesh@stanford.edu

1. Abstract

Single image calibration is the problem of predicting the camera parameters from one image. This problem is of importance when dealing with images collected in uncontrolled conditions by non-calibrated cameras, such as crowd-sourced applications.[2] It is also a fundamental step for numerous applications, such as 3D reconstruction, image undistortion, augmented reality and camera motion estimation. Existing calibration methods require multiple images of a calibration pattern (typically a checkerboard), assume the presence of lines, require manual interaction and/or need an image sequence.[1] Our networks automatically estimate the intrinsic parameters of the camera (focal length and distortion parameter) from a single input image. For training the CNN, we leverage a relatively small amount of images available on the Internet to automatically generate a dataset. Experiments successfully demonstrated the quality of our results, both quantitatively and qualitatively. In this project we evaluate several proposed automatic deep learning-based approaches that overcome all these limitations and work with a single image of general scenes. [2]

2. Introduction

Camera calibration refers to the estimation of their intrinsic parameters [Hartley and Zisserman 2004]. However, existing calibration methods have important limitations. For example, they require multiple observations of a calibration object (e.g., checker- board [Gasparini et al. 2009; Mei and Rives 2007; Scaramuzza et al. 2006], dot pattern [Shah and Aggarwal 1994] or sphere [Ying and Zha 2008]), and/or require the observation of specific structures in the scene (e.g., lines or vanishing points in structured scenes [Antunes et al. 2017; Barreto and Araújo 2005; Bräuer-Burchardt and Voss 2001; Melo et al. 2013; Swaminathan and Nayar 2000; Ying and Hu 2004]), and/or require estimating the camera motion from multiple images [Fitzgibbon 2001; Kang 2000; Micusík and Pajdla 2003; Xiong and Turkowski 1997; Zhang 1996].[1]
The goal of this project is to consider deep learning approaches to find camera parameters from a single image. Proposed approaches from papers [1,2] are built upon the recent developments in deep Convolutional Neural Networks (CNN): networks automatically estimate the intrinsic parameters of the camera (focal length and distortion parameter) from a single input image.
Our project focuses on two main aspects. First, a major challenge to train the network is the need for numerous training examples. In our context, we need plenty of images with different intrinsic parameters (focal length and distortion).[1] For this, we leverage collection of images available on the Internet, which allows us to automatically generate numerous images with different focal lengths and amounts of distortion. Then, we can train our neural networks on this generated data.
The second aspect is the comparison of different CNN architectures. We see that camera calibration from a single image follows the same framework as most machine learning problems. Given large amount of data, train the network to get the best results. While training big networks depend on hardware resources, the harder task is to find large amount of data.

3. Related work

Camera calibration aims to estimate the intrinsic parameters of the camera [Hartley and Zisserman 2004]. Several methods for camera calibration have been proposed, and can be divided into four main categories. The most popular and widely used category is based on a known calibration target (typically a checkerboard) placed in the scene and observed under different viewpoints in several images [Gasparini et al. 2009; Mei and Rives 2007; Scaramuzza et al. 2006; Shah and Aggarwal 1994]. Other targets have also been studied, such as dot pattern [Shah and Aggarwal 1994] or sphere [Ying and Zha 2008]. The methods belonging to this category are usually the most accurate, for example because the features in the images (e.g., checkerboard corners) can be precisely detected in the images and the calibration target model is known beforehand. An important limitation is that they require a specific calibration target and several images.[1] Therefore, they are cumbersome, time-consuming, and also not applicable to single images.
The second category is based on the presence of geometric structures in the scene, typically lines [Barreto and Araújo

2005; Workman et al. 2016; Zhang et al. 2015] and vanishing points [Antunes et al. 2017; Hughes et al. 2010]. For instance, the approach developed by Barreto and Araújo [2005] needs an image containing at least three lines that are manually given by the user, and Antunes et al. [2017] rely on orthogonal vanishing points. Therefore, these methods are limited to structured man-made scenes containing lines, and thus cannot deal with general environments, like landscapes or natural scenes, and when a large portion of the image is covered by an object, such as a face close-up.

The third category is camera self-calibration which jointly estimates the intrinsic parameters of the camera and its motion from a sequence of images. For example, Fitzgibbon [2001] solves a radial fundamental matrix and tri-focal tensor which allows to estimate together the distortion parameter with the epipolar geometry between two or three successive images. This seminal work led to several extensions with different configurations, number of images and minimal solutions [Jiang et al. 2014; Micusík and Pajdla 2003]. The two main limitations of self-calibration are the requirement of several images and the need to perform camera motion estimation, which is still a major challenge in itself (e.g., point correspondences, repetitive texture, lighting changes and motion ambiguity).

The last category is based on deep learning. One of the first attempts is the approach of Mendonça et al. [2002], which uses neural network to compute the camera parameters given 3D point locations from a calibration target and their respective 2D observations. Recent deep learning methods aiming to estimate camera parameters from a single image without calibration target have been attempted. However, all these techniques are designed to partly solve the calibration problem. For example, DeepFocal [Workman et al. 2015] predicts only the focal length (no distortion estimation). It is trained using few images (around 7,000). Hold-Geoffroy et al. [2018] estimate the focal length and camera orientation (no distortion parameter). They train a CNN on images generated from panoramas using standard pinhole model, and thus, are limited to perspective cameras. In contrast, our approach estimates both focal length and distortion. Therefore, it can handle a much wider range of cameras, such as fisheye and catadioptric cameras; and it also enables additional applications, such as SfM and image undistortion. Overall, estimating both the focal length and distortion requires dedicated methods and investigations, such as the selection of the distortion model, automatic training dataset generation, and network architectures.[1]

4. Technical Approach

This section presents the main aspects of our project: selection of the camera distortion model, generation of a large-scale dataset with ground truth intrinsic parameters, and comparison of network architectures.[1]

4.1. Projection and distortion model

In this project, we consider a wide FOV camera. It requires require specific projection models to map a 3D world point to the image. Various models have been developed. We consider two models. The first model is the unified spherical model [Barreto 2006; Mei and Rives 2007]. We picked it for several reasons. First, it is fully reversible; second, it can handle a very large range of distortions (from none and small to very large); and third, both the projection and back-projection processes admit closed-form solution which can be computed very efficiently. Moreover, the spherical model is compatible with a wider range of cameras than other models, for example perspective, wide-angle, fisheye and catadioptric cameras. Lastly, it is particularly convenient for our application since it involves a single distortion parameter ξ ranged between 0 and 1 (and slightly more than 1 for certain types of catadioptric camera [Ying and Hu 2004]). Therefore the value of ξ is convenient to bound, interpret and quantize, which are very desirable properties for training CNNs compared to existing polynomial models [Rong et al. 2016]. Notice that a single distortion parameter is generally considered enough to model the distortion [Fitzgibbon 2001]. [1]

The entire projection process can be expressed as: a a 3D world point $P_w = (X, Y, Z)$ is projected onto the sphere at $P_s = (Xs, Ys, Zs)$. This spherical point P_s is then projected onto the image plane at the location p = (x,y). This projection starts from a point O_c located at $(0, 0, \xi)$ above the sphere center O. The distance ξ between these two points models the geometric distortion of the camera. Therefore,

$p = (x, y) = (\frac{Xf}{\xi\sqrt{X^2+Y^2+Z^2}+Z} + u_0, \frac{Yf}{\xi\sqrt{X^2+Y^2+Z^2}+Z} + v_0)$ where (u_0, v_0) the pixel coordinates of the principal point in the image, f the focal length (with square pixels), and ξ the distortion parameter [Mei and Rives 2007]. One might notice that when $\xi = 0$ for perspective cameras, the model reduces to the standard pinhole perspective projection [Hartley and Zisserman 2004].[1]

Initially, we planned to include another models for radial distortion(i.e a two-parameter polynomial distortion). But according to [1], this model is only effective to approximate reasonably small distortions. Despite its popularity, this model has several important limitations. For example, the types of cameras which can be modeled with this representation are limited. For instance, it is not suitable for wide FOV cameras due to their large inherent distortions [Sturm et al. 2011].[1]

4.2. Generation of training dataset

To train the deep learning CNN, we need training examples. However, there is no existing large-scale dataset of wide FOV images with ground truth intrinsic parameters

that could be used to train a deep learning network. Concretely, to train our network, we need images with different focal lengths and a large variety of distortions, along with the corresponding ground truth values. In practice, it is cumbersome and virtually impossible to manually capture such data. Furthermore, the calibration of all these cameras would also be hardly feasible for a large-scale dataset. For these reasons, we generate a dataset synthetically.[1] A straightforward but naive approach would be to generate wide field-of-view images from a set of standard (perspective) calibrated images. However, adding distortion (i.e., increasing the field of view) in standard images inevitably leads to non-visible parts becoming visible, i.e., "occlusion" margins near the image borders typically shown as black area. This leads to non-realistic data. To overcome this limitation, we propose to leverage the large collections of panoramas available on the Internet because their complete 360-degree FOV can emulate any amounts of FOV (see Figure 1). Another advantage of using panoramas is that we can point the virtual camera to different orientations (azimuth and elevation) in order to observe different parts of the scene and mimic tilted cameras, which can provide additional images from a single panorama.[1]

In this project, we cannot use the original dataset from the paper[1]. Unfortunately, sun360 dataset is not available anymore. So, we use panoramas from a different dataset collected on the Internet. This dataset contains only [519] panoramas. Sun360 dataset contained [67,000] high-resolution 9104 × 4552px panoramas acquired in various scenes, such as indoor/outdoor, urban/natural and bright/dark.[1] But our dataset contains only indoor images. It makes training harder for two reasons: the dataset is smaller and it does not contain outdoor scenes.

Given an input panorama, the generation of a new image with a specific focal length and distortion value is composed of two main steps (see Figure 1). First, the panorama is linearly mapped onto the unit sphere. For this, let us consider a pixel (x,y) in the panorama, and write W and H respectively for the width and the height of the panorama. Then x is converted to the azimuth angle θ such that $x \in (1, W)$ is linearly mapped to $\theta \in (0, 2\pi)$,and similarly y is converted to the elevation angle ϕ such that $y \in (1, H)$ is linearly mapped to $\phi \in (\pi/2, \pi/2)$. By doing so for each pixel, we can project the panorama onto the sphere. The second step is to create a new synthetic image via a virtual camera, i.e., by reprojection with the desired values for focal length f and distortion parameter ξ. Some representative results are shown in Figure 1.

By following this approach, we generated [15542] images with ground truth intrinsic parameters and covering a large range of image configurations, such as different focal lengths and distortions. This is the biggest limitation of our project. One proposed solution is to generate more synthetic images from the given initial set. Alternatively, we can look for images of panoramas in the Internet. But this manual search is expensive.

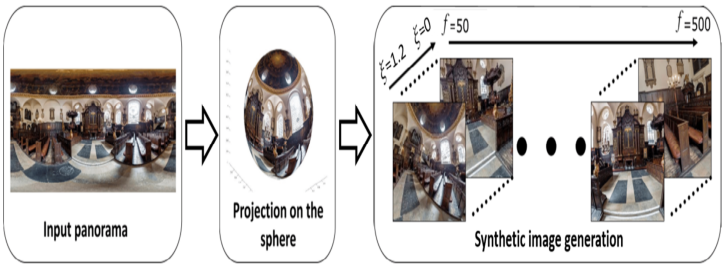

Figure 1. Given an input panorama, we automatically generate images with different focal lengths f and distortion values ξ,via the unified spherical model [Barreto 2006; Mei and Rives 2007]

4.3. Network architectures

Given an input image, we follow a deep learning-based approach to predict the distortion parameter and the focal length. We built upon a state-of-the-art architectures. We experimented with three different network architectures that we will describe below. For each network architecture, we solved the regression problem. For the regression problem, we used the sigmoid activation in the output layer(s) and the logcosh loss. Also, we used Huber loss from [2] because they suggest that it works better than their new proposed loss. But, in our results we did not see significant improvements. So, we stick to logcosh loss from [1]. We now describe the three different network architectures that we experimented with and evaluated.

The first architecture is Inception-V3 structure [Szegedy et al. 2016]. Our second network architecture is DenseNet-121. The third network architecture is ResNet50. All neural networks have two output dense layers: one for distortion estimation and one for focal length estimation.

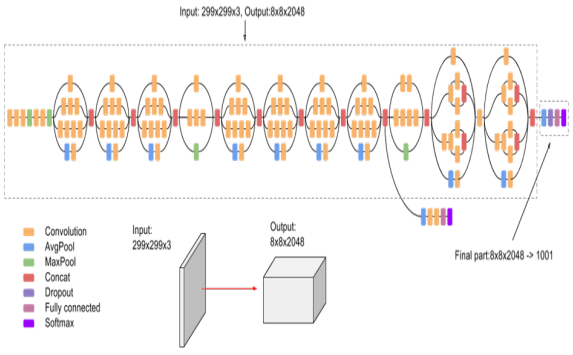

Figure 2. Inception-V3

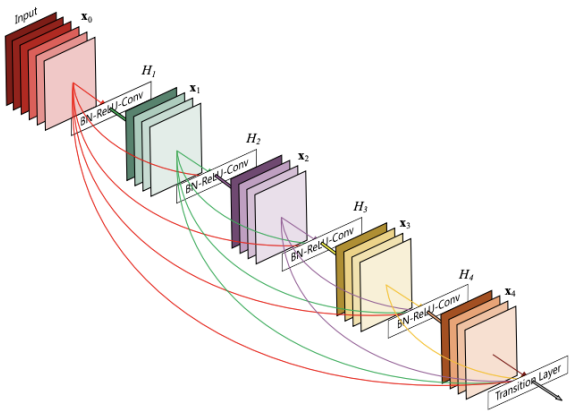

Figure 3. Densenet

5. Experiment

To train the networks, we generated a dataset composed of images with a resolution of 299 × 299px from all the panoramic images that we found. For regression training, we randomly sampled the values of focal length and distortion parameter on the same ranges described above (continuous dataset). In this way, we created [15542] images. We split the dataset into three subsets: 80 for training, 10 for testing, and 10 for validation. Each original panorama is used exclusively for training, or testing, or validation, i.e., none of the original panoramas belongs to more than one dataset.Three networks are pretrained on the ImageNet dataset, and we further train them on our generated dataset with early stop strategy to prevent overfitting [Raskutti et al. 2014]. We set the learning rate to 10^{-5} and used a batch size of 64. In this section, we evaluate and compare the performance of our network configurations. The performance is measured on the same continuous dataset. Figure 4 shows RMSE distribution for both distortion and focal length with respect to ground truth.

First, it shows that, surprisingly, our neural networks show

CNN	Focal length(RMSE)	Distortion(RMSE)
DeepCalib	304.72116	0.14645442
Inception-V3	304.7919	0.31428826
DenseNet-121	306.02767	0.35659145
ResNet50	309.2142	0.4194805

Figure 4. Results

almost the same performance as DeepCalib[1]. It is surprising because DeepCalib used SUN360 dataset which contains 67,000 images and they generated millions of synthetic images. On the other hand, we have only a small dataset of 519 images and generated 15542 synthetic images. Moreover, our performance may be almost the same as DeepCalib[1] because we trained only for 10 epochs. On the other hand, they trained 10,000 epochs. Unfortunately, training is expensive, so we stopped early.

Second, it shows that the three network architectures have a similar performance for focal extimation, and DeepCalib provides the highest performance, by a small margin. It is no coincidence because DeepCalib trained on larger and diverse set. DeepCalib used outdoor, urban/natural and bright/dark scenes. DeepCalib was exposed to a richer dataset. As a result, it is better on our dataset as well. We know that performance of neural networks depends heavily on data. On the other hand, our dataset contains only indoor samples generated from panoramas. Unfortunately, these samples expose approximately the same interiors. Nevertheless, the performance is almost the same. We also believe that training for a longer time, adding tuning parameters will improve the performance significantly.

The training and feed-forward execution of all proposed architectures are the same because number of parameters is approximately the same.

We also explored another aspect which is choosing loss function. DeepCalib used logcosh loss function. We also chose logcosh. One of the reasons was that we wanted to know how amount of data impacts training. We also tried the proposed Huber loss from [2]. But we did not get a significant improvements for 2 models, so we focused on logcosh. This loss fucntion works mostly like the mean squared error, but will not be so strongly affected by the occasional wildly incorrect prediction. In [2] authors also proposed their new loss function. We did not try this loss function because in the end authors said that it showed no improvements over Huber loss. Moreover, their proposed loss function is difficult to implement and has no closed solution. Therefore, it implies additional complexity and tuning. Since the performance is not increased, it is natural choice to stick to a simpler loss function. Of course, maybe better tuning is required for their proposed loss function but it is beyond the scope of this project.

We also decided to try to create more synthetic images from our initial dataset of 519 images by sampling randomly focal length and radial distortion. We trained on Inception-V3 architecture this bigger dataset but it did not show any improvements. We expected that more data would improve training. But it did not happen. Although we got more data, it did not imply a better training. We think it happened because the generated data still comes from the original 519 images and this dataset contains almost identical scenes. It is obvious that we need more original panoramas with diverse scenes.

We undistorted several images from all 4 model and we should say that the discrepancy between the values estimated by the network and the ground truth is difficult to assess on a qualitative level from a human perspective.
To challenge the robustness of the algorithm, DeepCalib proposed to undistort a set of wide FOV images "in the wild", i.e., downloaded from the Internet (without ground truth available). The images have been captured from unknown types of cameras and lenses having different characteristics, such as camera model, lens type, focal length, distortion, optics quality, resolution and light sensitivity. Given an image, we apply our network to estimate the focal length and distortion parameter. Then for the undistortion itself, the input distorted image is back-projected on the unit sphere using Eq. (2) and the estimated intrinsic parameters. This spherical image is then projected on the image plane with the desired intrinsic parameters. For instance, to generate a perspective image exempt of any distortions (pinhole model), we fixed ξ to 0 and the focal length to 150px. Note that other focal length values could be used for different zooming and cropping effects when ξ is set to 0. A set of representative results is available in Figure 5. It shows that the algorithm is able to correctly predict the distortion parameter and focal length under various scenarios and environments.[1]
We used the image(Figure 5) from the original paper because our parameters are almost the same. For human eyes, it is hard to detect a difference. The other reason is that we cannot find these images since the authors did not provide a link.
One of the limitation is that the unified spherical model and other existing lens distortion models have an ambiguity between the focal length and distortion parameter [Cornelis et al. 2002; Hartley and Kang 2007; Li and Hartley 2005], i.e., different combinations of focal length and distortion values may lead to the same (or similar) projection of a world point in the image. Therefore a promising extension of our approach is to consider different radial distortion models, as we mentioned it previously.

6. Conclusion

We have presented the results of evaluation of deep modern learning-based approach for automatic intrinsic calibration of wide FOV cameras. The approaches jointly predict focal length and radial distortion. We test the performance of the proposed architectures and modifications against DeepCalib. The only required input is a single image of a general scene, and the approach can automatically estimate the focal length and distortion parameter. For this, we used a method to automatically generate a dataset of wide FOV images with ground truth intrinsic parameters in order to train the CNN. We investigated three different network architectures, different loss functions, experimented with getting more synthetic samples from the same panoramas. We observed that DeepCalib is the network of choice in terms of accuracy. We have demonstrated the accuracy of our neural networks in experiments on synthetic data. Moreover, experiments also demonstrated that our approach can correctly handle wide FOV images "in the wild". We also plan to compare our results to several state-of-the-art calibration methods. A great advantage of this approach is that it can be successfully used for several practical cases when existing state-of-the-art methods cannot be applied, for example single image, images in the wild, unstructured scenes, absence of lines, no calibration target and/or manual process.[1] In future work, we will explore distortion calibration with other radial distortion models as we mentioned it before.

7. References

The project relies on two papers, here we included only relevant parts of the papers to our experimentation, but the project relies on all previous works from "References" of both papers. Refer to them in the papers[1, 2]:
[1] Oleksandr Bogdan, Francois Rameau, Viktor Eckstein, Jean-Charles Bazin. DeepCalib: A Deep Learning Approach for Automatic Intrinsic Calibration of Wide Field-of-View Cameras
[2] Manuel Lopez-Antequera, Roger Mari, Pau Gargallo, Yubin Kuang, Javier Gonzalez-Jimenez, Gloria Haro. Deep Single Image Camera Calibration with Radial Distortion
Simone Gasparini, Peter Sturm, and João P. Barreto. 2009. Plane-Based Calibration of Central Catadioptric Cameras. In ICCV.
Christopher Mei and Patrick Rives. 2007. Single View Point Omnidirectional Camera
Calibration from Planar Grids. In ICRA.
Davide Scaramuzza, Agostino Martinelli, and Roland Siegwart. 2006. A Toolbox for Easily Calibrating Omnidirectional Cameras. In IROS.
Shishir Shah and J. K. Aggarwal. 1994. A Simple Calibration Procedure for Fish-Eye (High-Distortion) Lens Camera. In ICRA.

8. Supplementary Material

Github link:
https://github.com/IhorLeshchyshyn/CameraAutocalibrationDeepLearning
Video presentation:
https://drive.google.com/file
/d/1YecapfUQbLflEgb7zgOFtpKmeHPhz53L/view?usp=sharing

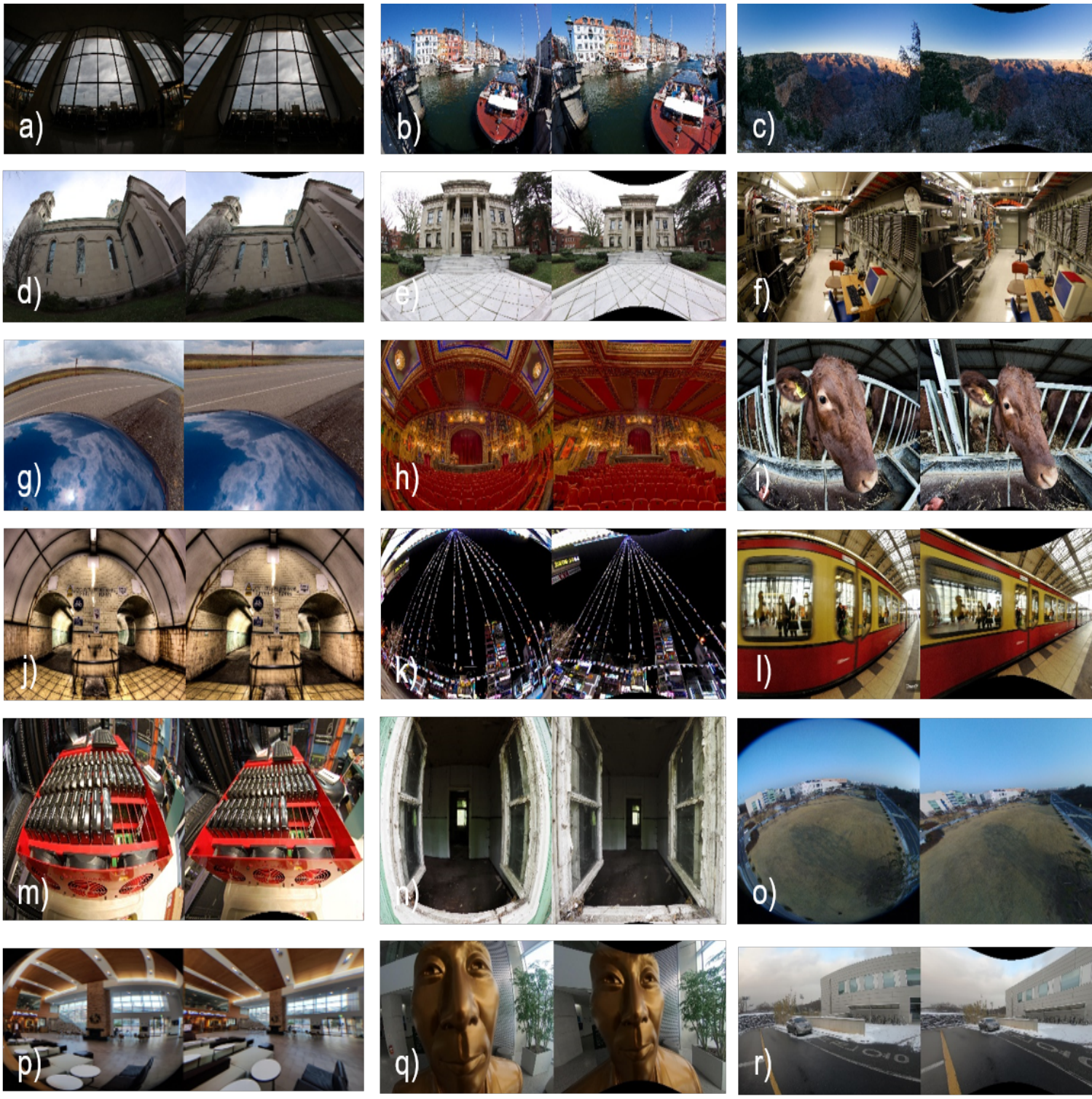

Figure 5. Examples of automatic undistortion results on images in the wild.