What Do the Sun and the Sky Tell Us About the Camera?

Jean-François Lalonde, Srinivasa G. Narasimhan, and Alexei A. Efros
School of Computer Science, Carnegie Mellon University
{jlalonde,srinivas,efros}@cs.cmu.edu
http://graphics.cs.cmu.edu/projects/sky

September 23, 2011

Abstract

As the main observed illuminant outdoors, the sky is a rich source of information about the scene. However, it is yet to be fully explored in computer vision because its appearance in an image depends on the sun position, weather conditions, photometric and geometric parameters of the camera, and the location of capture. In this paper, we analyze two sources of information available within the visible portion of the sky region: the sun *position*, and the sky *appearance*. By fitting a model of the predicted sun position to an image sequence, we show how to extract camera parameters such as the focal length, and the zenith and azimuth angles. Similarly, we show how we can extract the same parameters by fitting a physically-based sky model to the sky appearance. In short, the sun and the sky serve as geometric calibration targets, which can be used to annotate a large database of image sequences. We test our methods on a high-quality image sequence with known camera parameters, and obtain errors of less that 1% for the focal length, 1° for azimuth angle and 3° for zenith angle. We also use our methods to calibrate 22 real, low-quality webcam sequences scattered throughout the continental US, and show deviations below 4% for focal length, and 3° for the zenith and azimuth angles. Finally, we demonstrate that by combining the information available within the sun position and the sky appearance, we can also estimate the camera geolocation, as well as its geometric parameters. Our method achieves a mean localization error of 110km on real, low-quality Internet webcams. The estimated viewing and illumination geometry of the scene can be useful for a variety of vision and graphics tasks such as relighting, appearance analysis and scene recovery.

Keywords: sky · camera calibration · physics-based vision · time-lapse video · camera geolocation

Contents

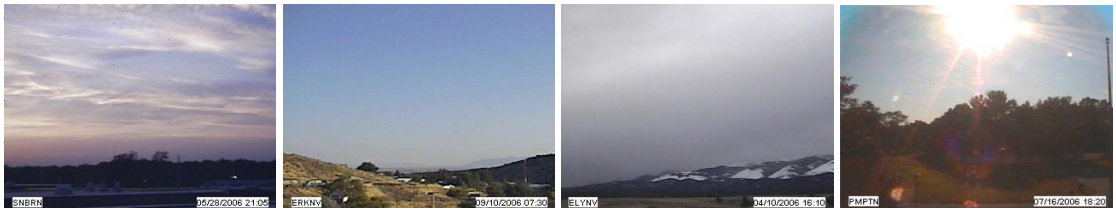


Figure 1: The sky appearance is a rich source of information about the scene illumination.

1 Introduction

When presented with an outdoor photograph (such as images on Fig. 1), an average person is able to infer a good deal of information just by looking at the sky. Is it morning or afternoon? Do I need to wear a sunhat? Is it likely to rain? A professional, such as a sailor or a pilot, might be able to tell even more: time of day, temperature, wind conditions, likelihood of a storm developing, etc. As the main observed illuminant in an outdoor image, the sky is a rich source of information about the scene. However it is yet to be fully explored in computer vision. The main obstacle is that the problem is woefully under-constrained. The appearance of the sky depends on a host of factors such as the position of the sun, weather conditions, photometric and geometric parameters of the camera, and location and direction of observation. Unfortunately, most of these factors remain unobserved in a single photograph; the sun is not often visible in the picture, the camera parameters and location are usually unknown, and worse yet, only a small fraction of the full hemisphere of sky is actually seen.

However, if we observe the same small portion of the sky *over time*, we would see changes in sky appearance due to the sun and weather that cannot be perceived within a single image. In short, this is exactly the type of problem that might benefit from analyzing a time-lapse image sequence, which is acquired by a static camera observing the same scene over a period of time.

The main contribution of this paper is to show what information about the camera is available in the visible portion of the sky in a time-lapse image sequence, and how to extract it. For this, we exploit two important cues – the sun position and the appearance of the sky. Our analysis demonstrates that it is possible to recover the viewing and illumination geometry from an image sequence, which is equivalent to estimating the focal length as well as the zenith (with respect to vertical), and azimuth (with respect to North) angles of the camera. In short, we show how the sky can be used as a calibration target for estimating camera orientation and focal length. Additionally, we also show how the sun and the sky can be used to estimate the camera latitude and longitude.

We present an overview of the 4 main results of this paper in Table 1. Algorithm 1, introduced in Sec. 3, computes the camera focal length f_c, zenith θ_c and azimuth ϕ_c angles given as input the sun position in images, the GPS coordinates (latitude and longitude) of the camera, as well as the date and time of capture of each image. This algorithm requires the sun to be manually identified in a few frames throughout the entire sequence, a process which takes only a few minutes per sequence. Note that the GPS coordinates and the date and time of capture are commonly available in online webcams.

Algorithm 2, presented in Sec. 4, uses the sky appearance as its only input. From several

Algorithm	Section	Inputs	Outputs
Alg. 1	Sec. 3	Sun position GPS coordinates Date and time	Camera parameters (f_c, θ_c, ϕ_c)
Alg. 2	Sec. 4	Clear sky images	Camera parameters (f_c, θ_c)
Alg. 3	Sec. 4	Clear sky images GPS coordinates Date and time	Camera parameters (f_c, θ_c, ϕ_c)
Alg. 4	Sec. 6	Clear sky images Sun position Date and time	Camera parameters (f_c, θ_c, ϕ_c) GPS coordinates

Table 1: Overview of the different algorithms introduced in this paper, which extract various information about the camera from the sun and the sky.

images of the clear sky, it can be used to estimate the camera focal length and zenith angle. In this case, GPS and time information are not required, therefore it can be applied to any set of images captured by a static camera. Algorithm 3, also in Sec. 4, shows how knowledge of the GPS coordinates and time and date of capture allows the recovery of the camera azimuth angle ϕ_c as well. This can be obtained completely automatically.

Finally, Algorithm 4 from Sec. 6 demonstrates that by combining the sun position with the sky appearance, the GPS coordinates can also be estimated, along with the camera focal length, zenith and azimuth angles. In short, the sun and the sky can be used to locate and calibrate the camera.

An immediate practical result of our work is the recovery of the camera orientation and zoom level, even if we do not have physical access to the camera. We validate Algorithms 1, 2 and 3 on a sequence where the ground truth camera parameters are known, and demonstrate that our methods make error of less that 1% in focal length, at most 3° in zenith angle, and at most 1° in azimuth angle. In addition, we also evaluate these algorithms on 22 real, low-quality webcam sequences from the AMOS (Archive of Many Outdoor Scenes) database [15], which contains image sequences taken by static webcams over more than a year. The selected sequences cover a wide range of latitudes (28° – 48°) and longitudes (74° – 124°), and are composed of a total of over a quarter of a million daytime images which were given as input to our algorithms. Unfortunately, ground truth is not available for these sequences, and we do not have physical access to the cameras. Instead, we analyze the consistency between parameters obtained from Algorithms 1 and 3, and show that the mean deviation is 4% for the focal length, and less than 1.5° and 3° for the zenith and azimuth angles respectively. We also validate Algorithm 4 on 8 of these sequences, and report a mean localization error of 110km.

For all these algorithms, we assume that a static camera is observing the same scene over time, with no roll angle (i.e. the horizon line is parallel to the image horizontal axis). We also assume that the sky region has been segmented, either manually or automatically [15, 13], and that the sun position has already been identified in images. For the algorithms that exploit the sky appearance (2, 3 and 4), we also assume that the camera has been radiometrically calibrated, for instance by adapting the method of Lin et al. [27] to operate on edges extracted over several frames.

Estimating the camera parameters effectively results in the recovery of the direction of main illumination (the sun) with respect to the camera. Once these parameters are estimated, we also show how we can estimate the atmospheric turbidity as well as the cloud layer from each image in a sequence. The turbidity encodes the degree of scattering (from clear to hazy to overcast), and can be of great use in outdoor illumination modeling. In addition to this application, this can be used by many existing vision and graphics algorithms. In vision, photometric stereo [33], image-based architectural modeling [42], and facial recognition [11] are well-known examples of applications which either require knowledge of the relative viewing and illumination geometry, or might greatly benefit from it. In graphics, appearance transfer of objects [23] and faces [1], as well as object relighting [6] and manipulation [18] are all examples where illumination conditions must be estimated. Now that illumination direction can be estimated in webcam sequences, it is our hope that this approach will spur novel research that exploits this exciting new data source. For example, the recent work of Lalonde et al. [26] propose appearance and illuminant transfer applications based on a large dataset of webcams calibrated using this approach.

2 Related work

Our approach is based on the idea that multiple images acquired from the same position are useful to observe the variations in illumination, while keeping everything else fixed. This insight has been very popular to solve many problems, including background subtraction [38], shadow detection and removal [41], video factorization and compression [36], radiometric calibration [19], and camera geo-location [17]. In this paper, we will consider using this information for a novel problem – understanding how we can recover camera parameters from the sun and the sky. The most relevant previous work can be grouped into three general categories: outdoor illumination modeling, sun and sky appearance analysis.

Outdoor illumination modeling Natural illumination is important to understand to accurately model the appearance of objects in outdoor images, so it has received a lot of attention in the computer vision community. One line of research aims to characterize the properties of outdoor illumination as a function of atmospheric conditions. For instance, Slater and Healey [34] determined that a 7-dimensional PCA representation captures 99% of the variance of the spectral distribution of natural illumination, based on a synthetically-generated dataset of spectral lighting profiles. Dror *et al.* [9] performed a similar study by using a set of HDR environment maps as input.

Another line of research analyzes the combined effect of scene reflectance and lighting by looking at the appearance of scenes under natural light. For example, Chiao *et al.* [3] determined that 3 linear bases are enough to represent the spectral composition of reflectance in forest images. Sato and Ikeuchi [33] introduced a model of outdoor illumination that represents both sunlight (directional), and skylight (hemispherical) separately, which is widely accepted now. These ideas have found applications in outdoor color representation and classification [2], surveillance [40], and robotics [28].

Narasimhan *et al.* [29] introduced a dataset of high quality registered and calibrated images of a fixed outdoor scene captured every hour for a year. By fixing the scene and viewing geometry, it is possible to analyze the effect of illumination, weather and aging conditions more directly. This idea has recently been exploited by several researchers. For example, Koppal and

Narasimhan [21] showed that scene points can be clustered according to their surface normals, irrespective of material and lighting, by observing their appearance over time, as illumination changes. On a global scale, Jacobs *et al.* [15] observed that the main variations in scene appearance are common across scenes by applying PCA on a large dataset of webcam sequences. Sunkavalli *et al.* [37] have also demonstrated impressive color constancy results, by fitting an illumination model to such an image sequence.

In contrast, our work investigates a novel, tangential research direction by modeling the sky appearance directly. The challenge is to infer information about the entire sky hemisphere when only a small portion is visible. We will show that we can achieve that goal by using a physically-based sky model, whose parameters can be recovered by fitting it to an image sequence where the sky appearance changes over time.

Sun position analysis The sun position has been exploited mostly in the robotics community. Cozman and Krotkov [5] extract sun altitudes from images and use them to estimate camera latitude and longitude. Trebi-Ollennu *et al.* [39] describe a system that estimates camera orientation in a celestial coordinate system, that is used to infer the attitude of a planetary rover. Both these techniques yield precise estimates, but require expensive additional hardware (digital inclinometer [5] and custom sun sensor [39]). In comparison, our method recovers the viewing geometry from the sun position annotated in images captured by any ordinary camera.

Sky appearance analysis The sky appearance has long been studied by physicists. One of the most popular physically-based sky model was introduced by Perez *et al.* [30], and was built from measured sky luminances. This model has been used in graphics for relighting architectural models [42], and for developing an efficient sky rendering algorithm [31]. Alternatively, Stumpfel *et al.* [35] proposed to capture the sky directly into an HDR environment map format, and used it for both rendering and relighting [8]. Surprisingly however, very little work has been done on extracting information from the visible sky. One notable exception is the work of Jacobs *et al.* [16] where they use the Perez sky model to infer the camera azimuth by using a correlation-based approach. In our work, we address a broader question: what does the sky tell us about the camera? We show how we can recover three camera extrinsic and intrinsic geometric properties simultaneously, from the sun position and the sky appearance.

3 Camera geometry from the sun position

We begin by introducing our sun-based algorithm, where the camera parameters are estimated from the sun *position* in a sequence of images. For this section, we assume that the sun position has already been identified in images.

3.1 Sun position model

Let us illustrate the geometry of the problem in Fig. 2. The sun, indicated by its zenith and azimuth angles (θ_s, ϕ_s), is observed by a camera and its center is projected at pixels (u_s, v_s) in the image. The camera, whose local reference frame is denoted by $(\mathbf{x}_c, \mathbf{y}_c, \mathbf{z}_c)$, is rotated by angles (θ_c, ϕ_c) and centered at the world reference frame $(\mathbf{x}_w, \mathbf{y}_w, \mathbf{z}_w)$. We assume that the camera has no roll angle, i.e. its horizon line is parallel to the image $\mathbf{u}$-axis.

The coordinates of the sun in the image (u_s, v_s) can be obtained by multiplying its 3-D coordinates $\mathbf{s}$ by the camera projection matrix $\mathbf{M}$, where

$$\mathbf{s} = \begin{bmatrix} x_s \\ y_s \\ z_s \\ 1 \end{bmatrix} = \begin{bmatrix} \sin\theta_s \cos\phi_s \\ \sin\theta_s \sin\phi_s \\ \cos\theta_s \\ 1 \end{bmatrix} , \tag{1}$$

in homogeneous coordinates and $\mathbf{M}$ has the form

$$\mathbf{M} = \mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix} . \tag{2}$$

Here $\mathbf{t} = \mathbf{0} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^\mathsf{T}$ since the center of projection is located at the origin. The rotation matrix R is given by:

$$\mathbf{R} = \begin{bmatrix} \cos(\theta_c - \frac{\pi}{2}) & 0 & -\sin(\theta_c - \frac{\pi}{2}) \\ 0 & 1 & 0 \\ \sin(\theta_c - \frac{\pi}{2}) & 0 & \cos(\theta_c - \frac{\pi}{2}) \end{bmatrix} \begin{bmatrix} \cos\phi_c & \sin\phi_c & 0 \\ -\sin\phi_c & \cos\phi_c & 0 \\ 0 & 0 & 1 \end{bmatrix} , \tag{3}$$

and assuming a simple camera model (no skew, square pixels), we can express the intrinsic parameters matrix $\mathbf{K}$ as:

$$\mathbf{K} = \begin{bmatrix} 0 & -f_c & 0 \\ 0 & 0 & f_c \\ 1 & 0 & 0 \end{bmatrix} . \tag{4}$$

With these definitions in hand, we now see how we can recover the parameters (f_c, θ_c, ϕ_c) of a camera that is observing the sun *over time*.

3.2 Recovering camera parameters from the sun position

This process is akin to general camera calibration (see [10] for more details), except that $\mathbf{M}$ is constrained to be of the form in (2). Suppose the sun is visible in N images taken from a sequence, and that we know the coordinates $\mathbf{p} = \begin{bmatrix} u_s & v_s \end{bmatrix}^\mathsf{T}$ of its center in the images. From

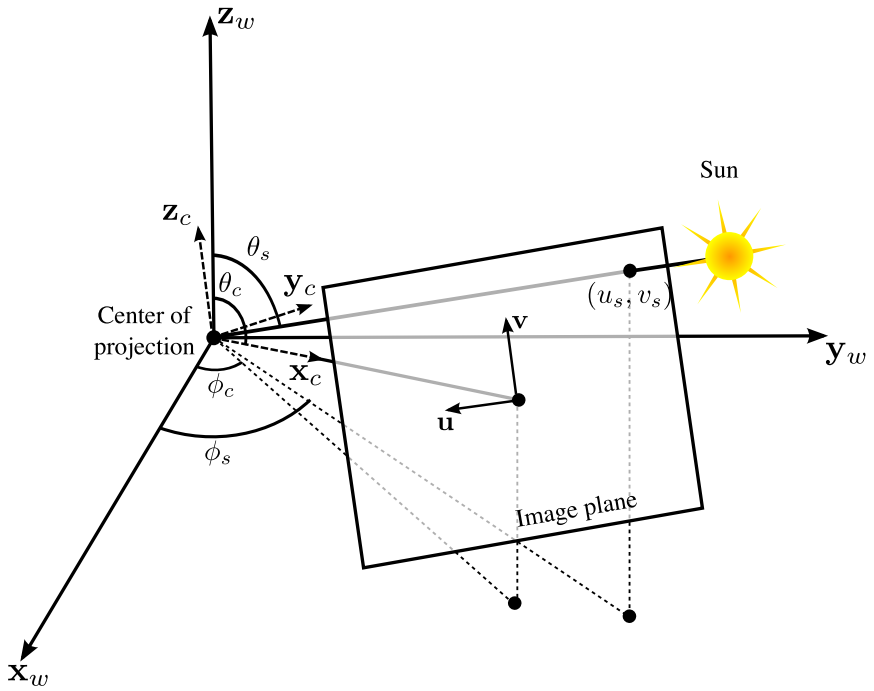


Figure 2: Geometry of the sun labeling problem. When the sun is visible in an image, its center gets projected at coordinates (u_s, v_s), which correspond to angles (θ_s, ϕ_s) with respect to the world reference frame $(\mathbf{x}_w, \mathbf{y}_w, \mathbf{z}_w)$. The camera has zenith and azimuth angles (θ_c, ϕ_c), and its focal length f_c, not shown here, is the distance between the origin (center of projection), and the image center. The camera local reference frame is denoted by $(\mathbf{x}_c, \mathbf{y}_c, \mathbf{z}_c)$ and is centered at the world reference frame.

the GPS location and the time of capture of each image, we can also obtain the corresponding sun angular positions (θ_s, ϕ_s) by using [32].

If $\mathbf{m}_i$ is the ith row of $\mathbf{M}$, then following the standard camera calibration procedure [10] we get that each image defines two equations:

$$\begin{aligned} \left(\mathbf{m}_1 - u_s^{(i)}\mathbf{m}_3\right) \cdot \mathbf{s}^{(i)} = 0 \ , \\ \left(\mathbf{m}_2 - v_s^{(i)}\mathbf{m}_3\right) \cdot \mathbf{s}^{(i)} = 0 \ . \end{aligned} \tag{5}$$

As detailed in Appendix A, this results in a system of $2N$ equations and 8 unknowns. When $N \geq 4$, homogeneous linear least-squares can be used to compute the value of the matrix $\mathbf{M}$ as the solution of an eigenvalue problem. The camera parameters can then be retrieved from the individual entries of $\mathbf{M}$ (see Appendix A for details).

We observe that we can improve the results quality by using these estimates as initial guess

in a non-linear minimization which optimizes the camera parameters directly:

$$\min_{f_c,\theta_c,\phi_c} \sum_{i=1}^{N} \left((\mathbf{m}_1 - u_s^{(i)}\mathbf{m}_3) \cdot \mathbf{s}^{(i)} \right)^2 + \left((\mathbf{m}_2 - v_s^{(i)}\mathbf{m}_3) \cdot \mathbf{s}^{(i)} \right)^2 \quad . \tag{6}$$

This non-linear least-squares minimization can be solved iteratively using standard optimization techniques such as Levenberg-Marquadt or `fminsearch` in MATLAB.

The entire sun-based calibration process is summarized in Algorithm 1: from a set of sun positions in images, with the corresponding GPS location and date and time of capture of each image, the algorithm recovers the camera parameters (f_c, θ_c, ϕ_c).

Algorithm 1: Camera geometry from the sun position

Input: Sun position in images: (u_s, v_s);
Input: GPS location of the camera;
Input: Date and time of capture of each image.

1 Compute the sun angles (θ_s, ϕ_s) from the GPS, date and time of capture using [32];
2 Build matrix $\mathbf{M}$;
3 Solve linear system (28);
4 Refine estimate by minimizing (6).

Output: Camera parameters: (f_c, θ_c, ϕ_c).

3.3 Validation using synthetic data

In order to thoroughly validate our approach, we test it under a very wide variety of conditions using synthetically-generated data.

3.3.1 Synthetic data generation

For a given set of camera parameters, sun coordinates $\mathbf{p}$ are generated by uniformly sampling the visible sky area (above the horizon line). Their corresponding 3-D sun vectors $\mathbf{s}$ are then found by applying the sequence of equations opposite to the one shown in the previous section. Gaussian noise is added to $\mathbf{p}$ to simulate error in sun center detection. All images have dimension 320×240 pixels.

We evaluate the influence of five variables on the quality of the results: the three camera parameters (f_c, θ_c, ϕ_c), the number of available images N, and the labeling noise variance σ^2. From this space of variables, we sample 213 different combinations by varying each one at a time, while maintaining the others constant at a default value, shown in Table 2. The range of each variable is also shown in that table. In order to account for the randomness in point selection, each experiment is performed 20 times.

3.3.2 Quantitative evaluation

Fig. 3 shows error curves for 6 different scenarios, illustrating the effect of varying each parameter presented in the previous section on estimating the camera parameters. Figs. 3a and 3b show

Variable name	Range	Default value	Comments
Focal length f_c	$[100, 2000]$ px	1000 px	18° field of view
Zenith angle θ_c	$[80°, 100°]$	90°	Looking straight
Azimuth angle ϕ_c	$[-180°, 180°]$	0°	Facing North
Number of images N	$[5, 50]$	20	–
Sun detection noise σ^2	$[0, 10]$ px	–	–

Table 2: Range and default values for each variable used in the experiments on synthetic data. Note that the range for θ_c is limited such that the horizon line remains visible at the given focal length.

that small focal lengths (or, alternatively, wide fields of view) result in larger estimation error, although it still remains below 3% error for f_c, and below 0.7° for θ_c, even in the high noise case ($\sigma^2 = 10$) Similar results are obtained for ϕ_c (not shown here). As expected, increasing N also decreases estimation error, as shown in Fig. 3d and 3f.

Plots obtained from varying θ_c and ϕ_c are shown in Fig. 3c and 3e, and result in mostly constant curves over the parameter range. θ_c is varied such that the horizon line remains visible in the image at the given focal length (from Table 2). In short, Fig. 3 demonstrates that this algorithm can simultaneously recover the focal length, zenith and azimuth angles of a very wide set of cameras, given a sequence of sun positions.

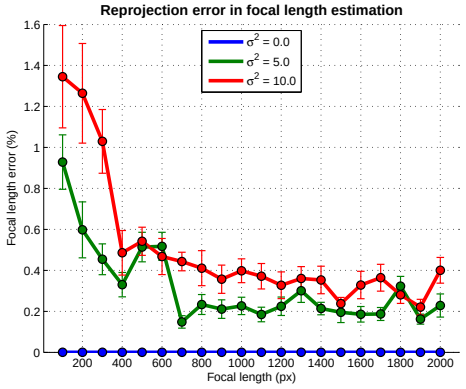
3.4 Validation using ground truth camera geometry

In addition to synthetic data, we also evaluate our algorithm on a sequence of images of the sky, captured by a calibrated camera with ground truth parameters: focal length, zenith and azimuth angles. We first present the acquisition and calibration setup, then show that both algorithms estimate camera parameters that are very close to ground truth.

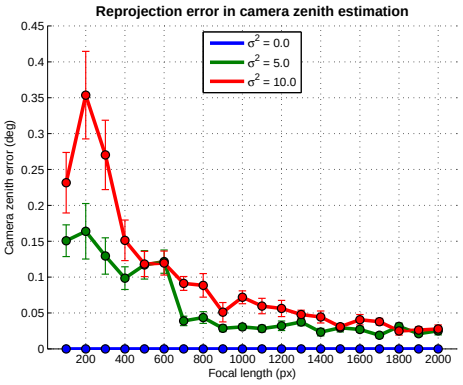
3.4.1 Acquisition setup

We captured a high-quality sequence of the sky by placing a Canon Digital Rebel XT equipped with a 18mm lens outdoors during an entire day (see Fig. 4a). Using a computer-controlled script, the camera continuously captured images at 5-minute intervals between 10:45 until 20:25, on April 17th, 2009. Fig. 4b shows example images from the resulting sequence. The camera was placed at the GPS coordinates of 40.367° of latitude, and −80.057° of longitude. The images were captured in RAW mode, which allows us to convert them to the JPEG format with a linear response function. We kept 8-bit dynamic range to simulate the behavior of a typical webcam.

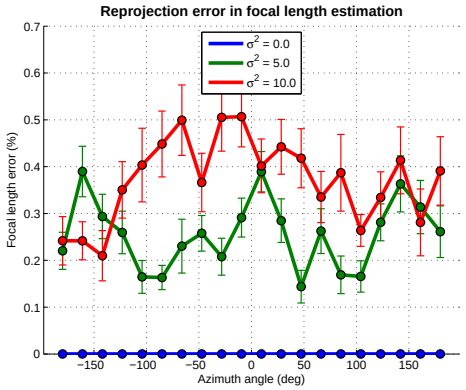
The intrinsic camera parameters were recovered using the MATLAB® camera calibration toolbox. The ground truth focal length was determined to be 2854 pixels. To compute the ground truth zenith angle, we placed a levelled calibration target in the field of view of the camera, and computed the intersection of parallel lines from the target. This intersection point indicates the horizon line in the image, from which the zenith angle can be computed using (18). The resulting zenith angle is 71.3°. The ground truth azimuth angle was computed using a satellite map of the capture location and was found to be 93.5° West.



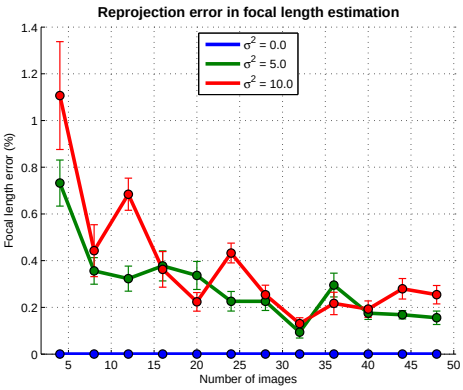
(a) $f_c \rightarrow f_c$



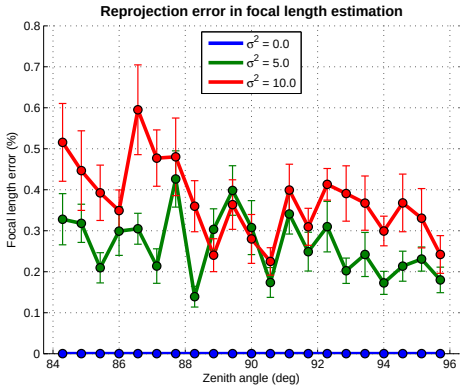
(b) $f_c \rightarrow \theta_c$



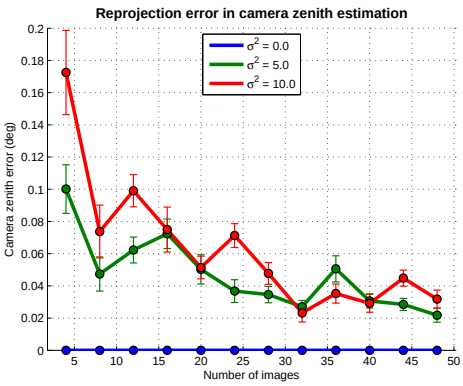
(c) $\theta_c \rightarrow f_c$



(d) $N \rightarrow f_c$



(e) $\phi_c \rightarrow f_c$



(f) $N \rightarrow \theta_c$

Figure 3: Synthetic data evaluation of camera parameters estimation from the sun position. A representative sample of the entire set of experiments performed is shown, the remaining plots can be found in [24].

(a) (b)

Figure 4: Ground truth data acquisition. (a) Acquisition setup. The calibration target, placed on a tripod in front of the camera and levelled with the ground, is used to recover the camera zenith angle. (b) Example images from 12:00 (top-left) to 20:00 (bottom-right), in one-hour increments. The images are shown with a linear camera response function.

Parameter	Ground truth	Sun estimate	Error
Focal length f_c	2854 px	2881 px	0.9%
Zenith angle θ_c	71.3°	70.2°	1.1°
Azimuth angle ϕ_c	93.5° W	94.3° W	0.8°

Table 3: Comparison with ground truth.

3.4.2 Calibration results

After manually labeling the sun in all the frames in which it is visible, the sun-based calibration algorithm was applied to recover the camera parameters. The results, shown in Table 3, demonstrate that the focal length can be recovered within 0.9% of its true value, and the zenith and azimuth angles are obtained within 1.1° and 0.8° of their true values respectively.

4 Camera geometry from the sky appearance

Unfortunately, the sun might not always be visible in image sequences, so the previous algorithm applicability might be limited in practice. Therefore, we now focus our attention on a different source of information more widely available: the sky *appearance*. We will consider clear skies only, and address the more complicated case of clouds at a later point in the paper.

4.1 Physically-based model of the sky appearance

First, we introduce the physically-based model of the sky that lies at the foundation of our approach. We will first present the model in its general form, then in a useful simplified form, and finally demonstrate how it can be written as a function of camera parameters.

4.1.1 Perez sky model

The Perez sky model [30] describes the luminance of any arbitrary sky element as a function of its elevation, and its relative orientation with respect to the sun. It is a generalization of the CIE standard clear sky formula [4], and it has been found to be more accurate for a wider range of atmospheric conditions [14]. Consider the illustration in Fig. 5. The *relative* luminance l_p of a sky element is a function of its zenith angle θ_p and the angle γ_p with the sun:

$$l_p = f(\theta_p, \gamma_p) = [1 + a \exp(b/\cos\theta_p)] \times \left[1 + c \exp(d\gamma_p) + e \cos^2\gamma_p\right] \quad , \tag{7}$$

where the 5 constants a, b, c, d, e specify the current atmospheric conditions, and all angles are expressed in radians. As suggested in [31], those constants can also be approximated by a linear function of a single parameter, the turbidity t. Intuitively, the turbidity encodes the amount of scattering in the atmosphere, so the lower the t, the clearer the sky. For clear skies, the constants take on the following values: $a = -1$, $b = -0.32$, $c = 10$, $d = -3$, $e = 0.45$, which corresponds approximately to $t = 2.17$.

The model expresses the *absolute* luminance L_p of a sky element as a function of another arbitrary reference sky element. For instance, if the zenith luminance L_z is known, then

$$L_p = L_z \frac{f(\theta_p, \gamma_p)}{f(0, \theta_s)} \quad , \tag{8}$$

where θ_s is the zenith angle of the sun. Fig. 6 illustrates the luminance predicted by the Perez sky model for different values of t and θ_s.

4.1.2 Azimuth-independent sky model

By running synthetic experiments, we were able to determine that the influence of the second factor in (7) becomes negligible when the sun is more than 100° away from a particular sky element (see Appendix C). In this case, the sky appearance can be modeled by using only the first term from (7):

$$l'_p = f'(\theta_p) = 1 + a \exp(b/\cos\theta_p) \quad . \tag{9}$$

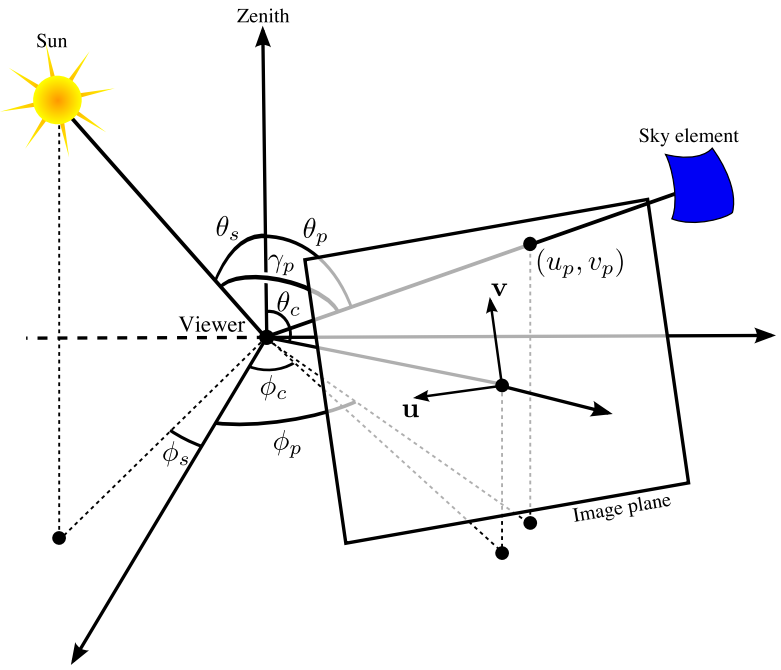


Figure 5: Geometry of the problem when a camera is viewing a sky element (blue patch in the upper-right). The sky element is imaged at pixel (u_p, v_p) in the image, and the camera is rotated by angles (θ_c, ϕ_c). The camera focal length f_c, not shown here, is the distance between the origin (center of projection), and the image center. The sun direction is given by (θ_s, ϕ_s), and the angle between the sun and the sky element is γ_p. Here (u_p, v_p) are known because the sky is segmented.

This equation effectively models the sky *gradient*, which varies from light to dark from horizon to zenith on a clear day. L'_p is obtained in a similar fashion as in (8):

$$L'_p = L_z \frac{f'(\theta_p)}{f'(0)} \ . \tag{10}$$

4.1.3 Expressing the sky model as a function of camera parameters

Now suppose a camera is looking at the sky, as in Fig. 5. We can express the general (7) and azimuth-independent (9) models as functions of camera parameters. Let us start with the simpler azimuth-independent model.

From (9), we see that we only need to find an expression for θ_p, since a and b are fixed. We obtain the following relation (see Appendix B for the full derivation):

$$\theta_p = \arccos\left(\frac{v_p \sin\theta_c + f_c \cos\theta_c}{\sqrt{f_c^2 + u_p^2 + v_p^2}}\right) \ , \tag{11}$$

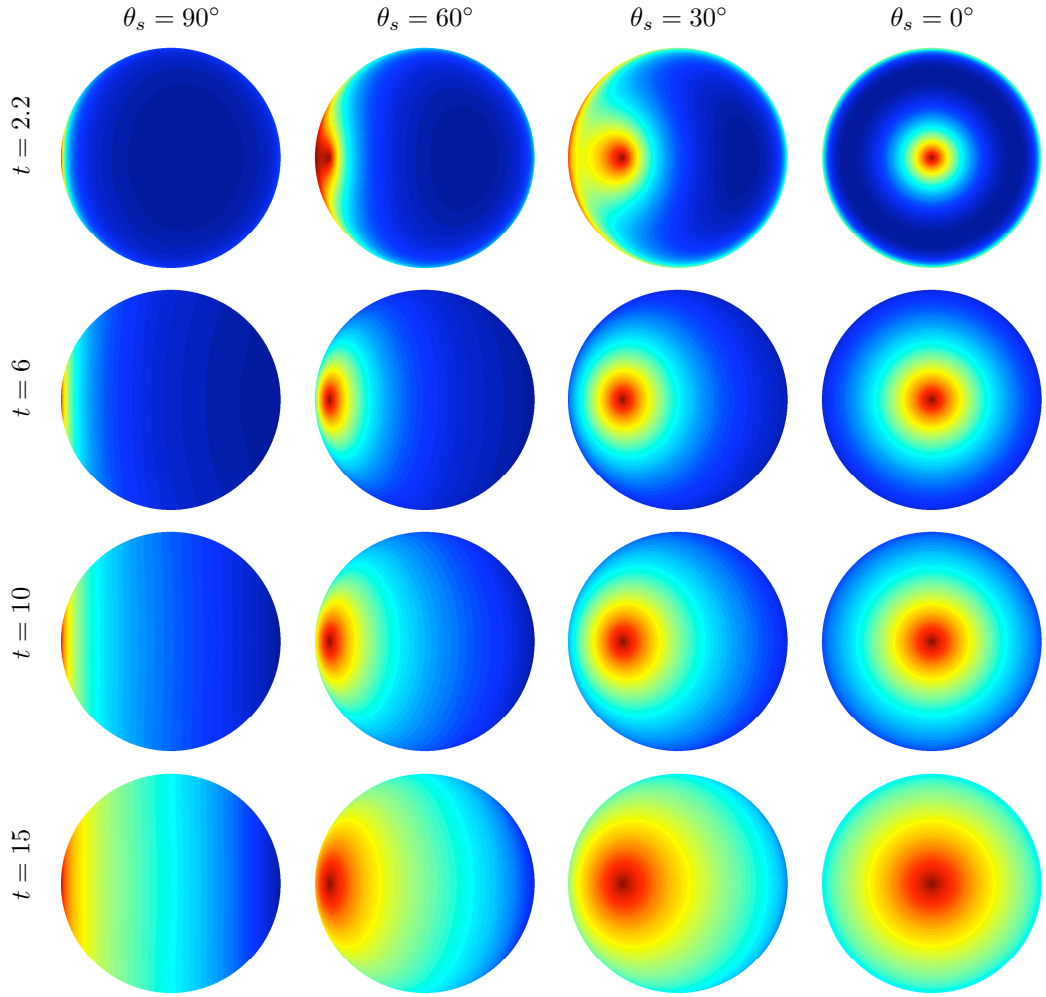


Figure 6: Perez sky model (7), plotted for different turbidities t (rows), for different sun positions θ_s (columns). Each plot represents the entire hemisphere, centered at the zenith (i.e. looking straight up). For example, the sun is at the horizon in the left-most column ($\theta_s = 90°$), and at the zenith in the right-most column ($\theta_s = 0°$).

where u_p and v_p are the image coordinates of a sky element, f_c is the camera focal length, and θ_c is its zenith angle (see Fig. 5). We substitute (11) into (9) to obtain the final equation. Throughout this paper, we will refer to this model using:

$$l'_p = g'(u_p, v_p, f_c, \theta_c) \ . \tag{12}$$

In the general sky model case (7), we also need to express γ_p as a function of camera parameters:

$$\gamma_p = \arccos\left(\cos\theta_s \cos\theta_p + \sin\theta_s \sin\theta_p \cos(\phi_p - \phi_s)\right) \ , \tag{13}$$

where θ_s and ϕ_s are the sun zenith and azimuth angles. We already found the expression for θ_p in (11), so the only remaining unknown is ϕ_p, the azimuth angle of the sky element. Following the derivation from Appendix B, we obtain:

$$\phi_p = \arctan\left(\frac{f_c \sin\phi_c \sin\theta_c - u_p \cos\phi_c - v_p \sin\phi_c \cos\theta_c}{f_c \cos\phi_c \sin\theta_c + u_p \sin\phi_c - v_p \cos\phi_c \cos\theta_c}\right) \quad . \tag{14}$$

We substitute (11), (13), and (14) into (7) to obtain the final equation. For succinctness, we omit writing it in its entirety, but present its general form instead:

$$l_p = g(u_p, v_p, \theta_c, \phi_c, f_c, \theta_s, \phi_s) \quad . \tag{15}$$

Before we present how we use the models presented above, recall that we are dealing with *ratios* of sky luminance, and that a reference element is needed. Earlier, we used the zenith luminance L_z as a reference in (8) and (10), which unfortunately is not always visible in images. Instead, we can treat this as an additional unknown in the equations. Since the denominators in (8) and (10) do not depend on camera parameters, we can combine them with L_z into a single unknown scale factor k.

4.2 Recovering camera parameters from the sky appearance

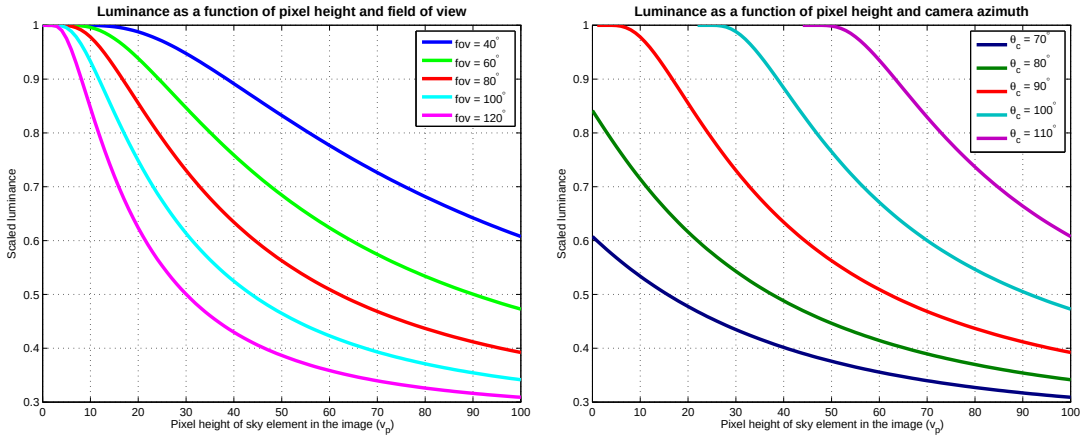
In the previous section, we presented a physically-based model of the clear sky that can be expressed as a function of camera parameters. Now, if we are given a set of images taken from a static camera, can we use the clear sky as a calibration target and recover the camera parameters from the sky appearance only?

4.2.1 Recovering the focal length and zenith angle

Let us first consider the simple azimuth-independent model (12). If we plot the predicted luminance profile for different focal lengths as in Fig. 7a (or, equivalently, for different fields of view), we can see that there is a strong dependence between the focal length f_c and the shape of the luminance gradient. Similarly, the camera azimuth θ_c dictates the vertical offset, as in Fig. 7b. From this intuition, we devise a method of recovering the focal length and zenith angle of a camera from a set of images where the sun is far away from its field of view (i.e. at least 100° away). Suppose we are given a set $\mathcal{I}$ of such images, in which the sky is visible at pixels in set $\mathcal{P}$, also given. We seek to find the camera parameters (θ_c, f_c) that minimize

$$\min_{\theta_c, f_c, k^{(i)}} \sum_{i \in \mathcal{I}} \sum_{p \in \mathcal{P}} \left(y_p^{(i)} - k^{(i)} g'(u_p, v_p, \theta_c, f_c)\right)^2 \quad , \tag{16}$$

where $y_p^{(i)}$ is the observed intensity of pixel p in image i, and $k^{(i)}$ are unknown scale factors (Sec. 4.1.3), one per image. f_c is initialized to a value corresponding to a 35° field of view, and θ_c is set such that the horizon line is aligned with the lowest visible sky pixel. All $k^{(i)}$'s are initialized to 1. This process is summarized in Algorithm 2.



(a) (b)

Figure 7: Luminance profiles predicted by the azimuth-independent model (12). For clear skies, intensity diminishes as pixel height above the horizon (x-axis) increases. (a) The camera zenith angle is kept constant at $\theta_c = 90°$, while the field of view is varied. (b) The field of view is kept constant at $80°$, while the camera zenith angle is varied. Both parameters have a strong influence on the shape and offset of the predicted sky gradient. Note that the curves in (b) are not translations of the same curve along the x-axis, because they are expressed in pixels in the image, not angles.

Algorithm 2: Camera zenith and focal length from the sky appearance

Input: Clear sky images.

1 Find set $\mathcal{I}$ of images where the sun is far away from the field of view;
2 Solve the non-linear minimization (16).

Output: Camera parameters: (f_c, θ_c).

4.2.2 Recovering the azimuth angle

From the azimuth-independent model (12) and images where the sun is far from the camera field of view, we were able to estimate the camera focal length f_c and its zenith angle θ_c. If we consider the general model (15) that depends on the sun position, we can also estimate the camera azimuth angle using the same framework as before.

Suppose we are given a set of images $\mathcal{J}$ where the sky is clear, but where the sun is now closer to the camera field of view. Similarly to (16), we seek to find the camera azimuth angle which minimizes

$$\min_{\phi_c, k^{(j)}} \sum_{j \in \mathcal{J}} \sum_{p \in \mathcal{P}} \left(y_p^{(j)} - k^{(j)} g(u_p, v_p, \theta_c, \phi_c, f_c, \theta_s, \phi_s) \right)^2 . \tag{17}$$

We already know the values of f_c and θ_c, so we do not need to optimize over them. Additionally,

if the GPS coordinates of the camera and the time of capture of each image are known, the sun zenith and azimuth (θ_s, ϕ_s) can be computed using [32]. Therefore, the only unknowns are $k^{(j)}$ (one per image), and ϕ_c. Since this equation is highly non-linear, we have found that initializing ϕ_c to several values over the $[-\pi, \pi]$ interval and keeping the result that minimizes (17) works the best. This process is summarized in Algorithm 3.

Algorithm 3: Camera geometry from the sky appearance

Input: Clear sky images;
Input: GPS location of the camera;
Input: Date and time of capture of each image.

1 Apply Algorithm 2 on clear sky images to recover f_c and θ_c;
2 Find set $\mathcal{J}$ of images where the sun is close to the field of view;
3 Compute the sun angles (θ_s, ϕ_s) from the GPS, date and time of capture using [32];
4 Solve the non-linear minimization (17).

Output: Camera parameters: (f_c, θ_c, ϕ_c).

4.3 Validation using synthetic data

In order to thoroughly evaluate our model, we have performed extensive tests on synthetic data generated under a wide range of operating conditions. We now present our data generation algorithm, followed by plots showing that we can effectively recover the parameters of a diverse set of cameras.

4.3.1 Synthetic data generation

We evaluate the influence of seven variables on the quality of the results: the three camera parameters (f_c, θ_c, ϕ_c), the number of available clear sky images N, the sky visibility (percentage of unoccluded sky above the horizon), the camera latitude, and the noise variance σ^2. Given a set of camera parameters and sun positions, we generate synthetic images by using our sky model (15). Sun positions are generated by sampling every hour over an entire year at a given latitude, and applying [32]. Note that longitude does not affect the results since daytime images can be chosen such that $\theta_s < \frac{\pi}{2}$. We simulate limited visibility by occluding the sky from the horizon line, one row at a time.

We build set $\mathcal{I}$ by randomly picking N sun positions that are at least 100° away from the camera field of view. If no such point is available given the geometry, we select those that are furthest away from the camera. We build set $\mathcal{J}$ by randomly selecting N sun positions. In both cases, we make sure that the sun is never directly visible by the camera. 1000 points are then randomly picked for each image, and used in the optimization. In order to evaluate the influence of each variable independently, we vary one at a time, while keeping the others at their default values shown in Table 4. Each experiment is performed 15 times to account for the randomness in point and sun position selection.

Variable name	Symbol	Default value	Comments
Focal length	f_c	750 px	24° field of view
Zenith angle	θ_c	90°	Looking straight
Azimuth angle	ϕ_c	0°	Facing North
Number of images	N	15	
Sky visibility	–	100%	No occlusion
Latitude	–	0°	Equator

Table 4: Default values for each variable used in the experiments on synthetic data.

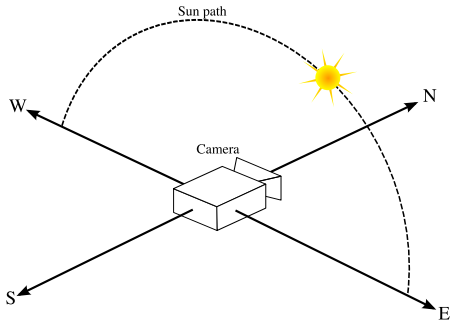


Figure 8: Graphical illustration of the influence of camera azimuth. If the camera is at the Equator (latitude=0°) and facing North (or South, equivalently), the sun path is further away from the camera than if it was facing East (or West). Its influence is less noticeable, therefore it is harder to recover the camera azimuth.

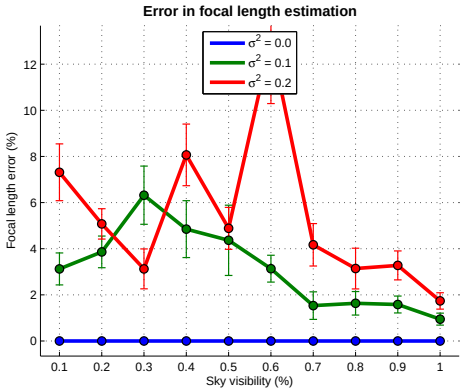
4.3.2 Quantitative evaluation

Fig. 9 shows the effect of varying the variables on the camera parameters estimation error. As expected, Fig. 9a shows that increasing the visible portion of the sky decreases the estimation error.

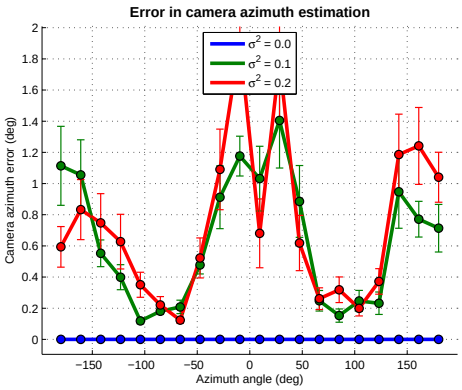
Fig. 9b represents the influence of the camera azimuth angle ϕ_c on the estimation error of the same parameter, and shows that error is typically higher when $\phi_c = 0$ (North) or $\phi_c = \pi$ (South) in noisy conditions. In this configuration, the sun is always relatively far from the camera, so its influence is not as visible, see Fig. 8 for a graphical illustration of the situation.

Figs 9c and 9d shows the effect of varying focal length f_c on estimating f_c and θ_c respectively. We note the higher the f_c (or, equivalently, the narrower the field of view), the larger error. In this situation, the visible portion of the sky hemisphere is smaller, and variations in intensity more subtle to capture.

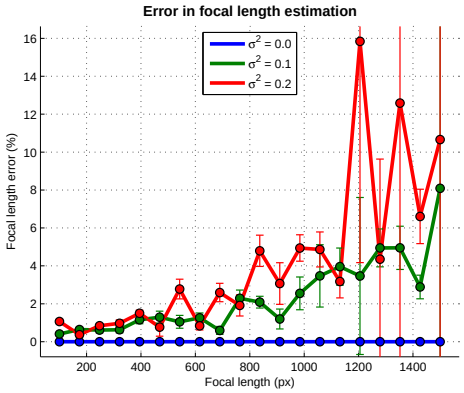
Finally, we note that the latitude does not seem to affect the estimation error, as varying it over the $[0°, 90°]$ range yields mostly constant curves as shown in Fig. 9f. The same behavior is also observed in Fig. 9e for θ_c, which is varied such that the horizon line remains visible in the image at the given focal length.



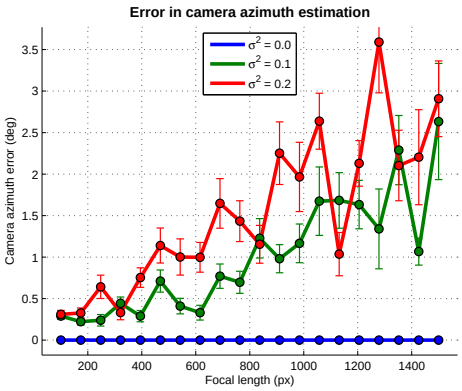
(a) $v \rightarrow f_c$



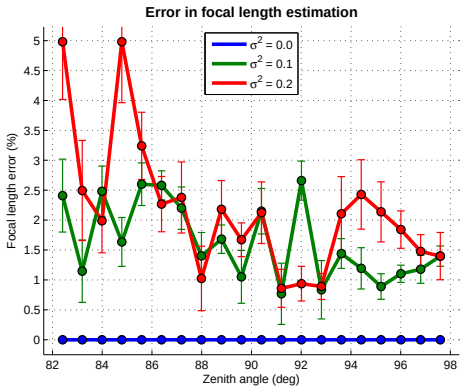
(b) $\phi_c \rightarrow \phi_c$



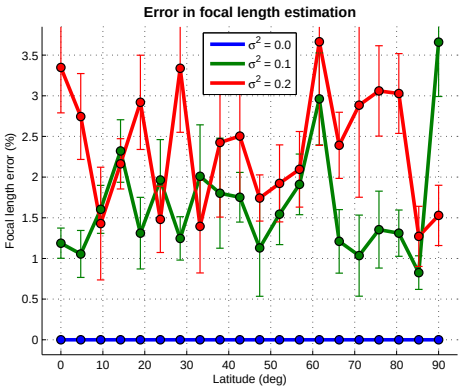
(c) $f_c \rightarrow f_c$



(d) $f_c \rightarrow \phi_c$



(e) $\theta_c \rightarrow f_c$



(f) latitude $\rightarrow f_c$

Figure 9: Synthetic data evaluation of camera parameters estimation from the sky appearance. Each of the 6 representative plots are shown at three different noise levels.

Parameter	Ground truth	Sky estimate	Error
Focal length f_c	2854 px	2845 px	0.3%
Zenith angle θ_c	71.3°	74.4°	3.1°
Azimuth angle ϕ_c	93.5° W	94.4° W	0.9°

Table 5: Comparison with ground truth for the sky-based algorithm.

4.4 Validation using ground truth camera geometry

In addition to synthetic data, we also evaluate our algorithm on the same sequence of images of the sky as in Sec. 3.4, captured by a calibrated camera with ground truth parameters.

After manually segmenting the sky, we applied our sky-based calibration algorithm which yielded the camera parameters estimates shown in Table 5. The algorithm is able to recover, *entirely automatically*: the focal length within 0.3% of the true value, the zenith angle within 3.1°, and the azimuth angle within 0.9°.

5 Calibrating the webcams of the world

We have already shown that our algorithms achieve good estimation results on synthetic data and on one set of high-quality images. In this section, we demonstrate how our algorithms perform on a wide range of operating conditions, by testing them on a large set of real, typical low-quality webcam data such as the ones found in the AMOS database [15]. We first evaluate each technique independently, and then compare their results together to establish their consistency, which confirms that both can reliably be used in a real application setting.

We test our algorithms on 22 webcam sequences from the AMOS database in which the sun is visible, which amounts to a total of approximately a quarter of a million individual daytime images. The selected webcams are located in the continental United States, their individual GPS locations are shown in Fig. 10. Although they are all in the same country, they cover a wide range of latitudes (28° – 48°) and longitudes (74° – 124°).

5.1 Using the sun position

We first present results obtained by using the sun position to recover the camera parameters. Although no ground truth is available, we can assess the estimation quality by displaying the predicted sun position on every image, as well as the estimated horizon line v_h, and inspect the results visually. v_h is obtained by:

$$v_h = -f_c \tan\left(\frac{\pi}{2} - \theta_c\right) \quad . \tag{18}$$

Fig. 11 shows examples of sun position and horizon line prediction on several frames for different image sequences, where the sun was not manually labeled. The predicted and actual sun positions overlap, which indicates that the camera parameters are recovered successfully.

A useful by-product of our approach is that the sun position can be predicted for all frames in the sequences, even if it is not visible. For example, in Fig. 11 Seq. 347, the sun position in the second and fourth frames can be predicted even if it is occluded by the scene.

5.2 Using the sky appearance

We now present results obtained by using the sky appearance to recover the camera parameters from real image sequences. Intensity information can be corrupted in several ways in low-quality webcam sequences: non-Gaussian noise, slight variations in atmospheric conditions, vignetting, saturation, under-exposure, etc. Most importantly however, the camera response function may be non-linear, yielding significant distortions in the sky appearance, thus leading our estimation process astray. We rely on [27] which estimates the inverse response function by using color edges gathered from a single image. For additional robustness, we detect edges across several frames.

Additionally, recall that the optimization procedures (16) and (17) require clear sky image sets $\mathcal{I}$ and $\mathcal{J}$, where the sun is far and close to the camera respectively. We first give more details about how this is done, and then present results on real image sequences.

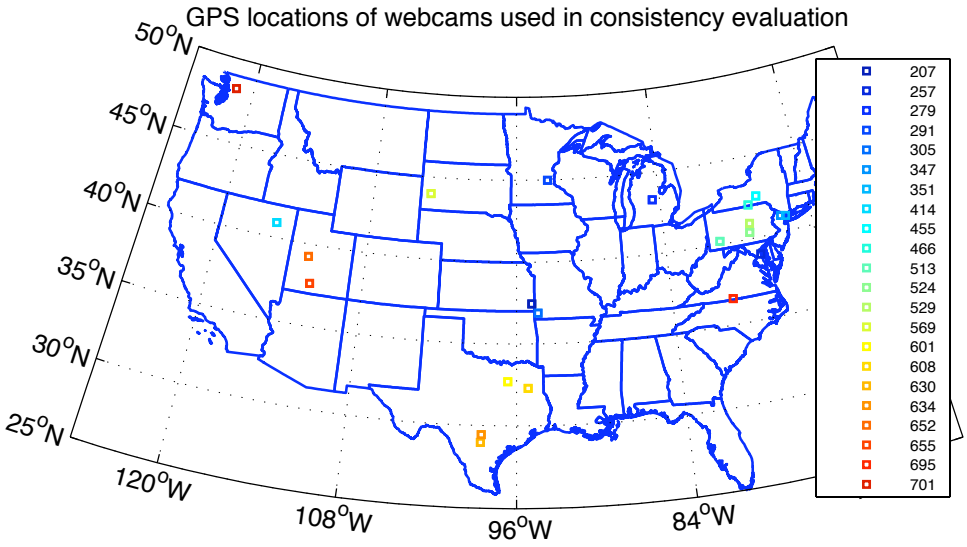


Figure 10: Map of GPS coordinates of each webcam used in the consistency evaluation. Since the sequences selected from the AMOS database come from the US, only this area of the world is shown. However, as we have shown in the synthetic evaluation, the algorithms are not limited to this area. Each webcam is represented by a different color.

5.2.1 Finding clear sky images automatically

The algorithm presented in Sec. 4.2 uses clear sky images as input. In order to avoid having to painstakingly select such images by hand from a long image sequence, we propose an algorithm which does not require any knowledge of the camera parameters to do so automatically.

To build set $\mathcal{I}$, we mentioned that the sun should be at least 100° away from the camera field of view. Unfortunately, this is impossible to know a priori, so we propose an algorithm that finds clear sky images that do not seem affected by the sun. To do so, we approximate the sky model (12) by a vertical quadratic of the form:

$$\alpha(v_p - v_{min})^2 + \beta = 0 \ , \tag{19}$$

where v_{min} is the lowest visible sky pixel, and α and β are the quadratic coefficients. We then fit this simple model to all the images in the sequence using linear least-squares. Images with low residual error and $\alpha < 0$ should exhibit a smooth vertical gradient that correspond to a clear sky, since it varies from light to dark from horizon to zenith. We found that retaining the top 10% of images with $\alpha < 0$ based on their residual error, and then keeping the top N by sorting them in decreasing order of $|\alpha|$ yields consistently good results across image sequences. Note that if the sun was close to the camera, it would most likely create a horizontal gradient in the image, thereby reducing the quality of a fit to a vertical quadratic. We show example images recovered by this algorithm in Fig. 12a.

Building set $\mathcal{J}$ requires finding images where the sun influence is visible at varying degrees. This is a harder problem than before for three main reasons: 1) the sun might never be very close

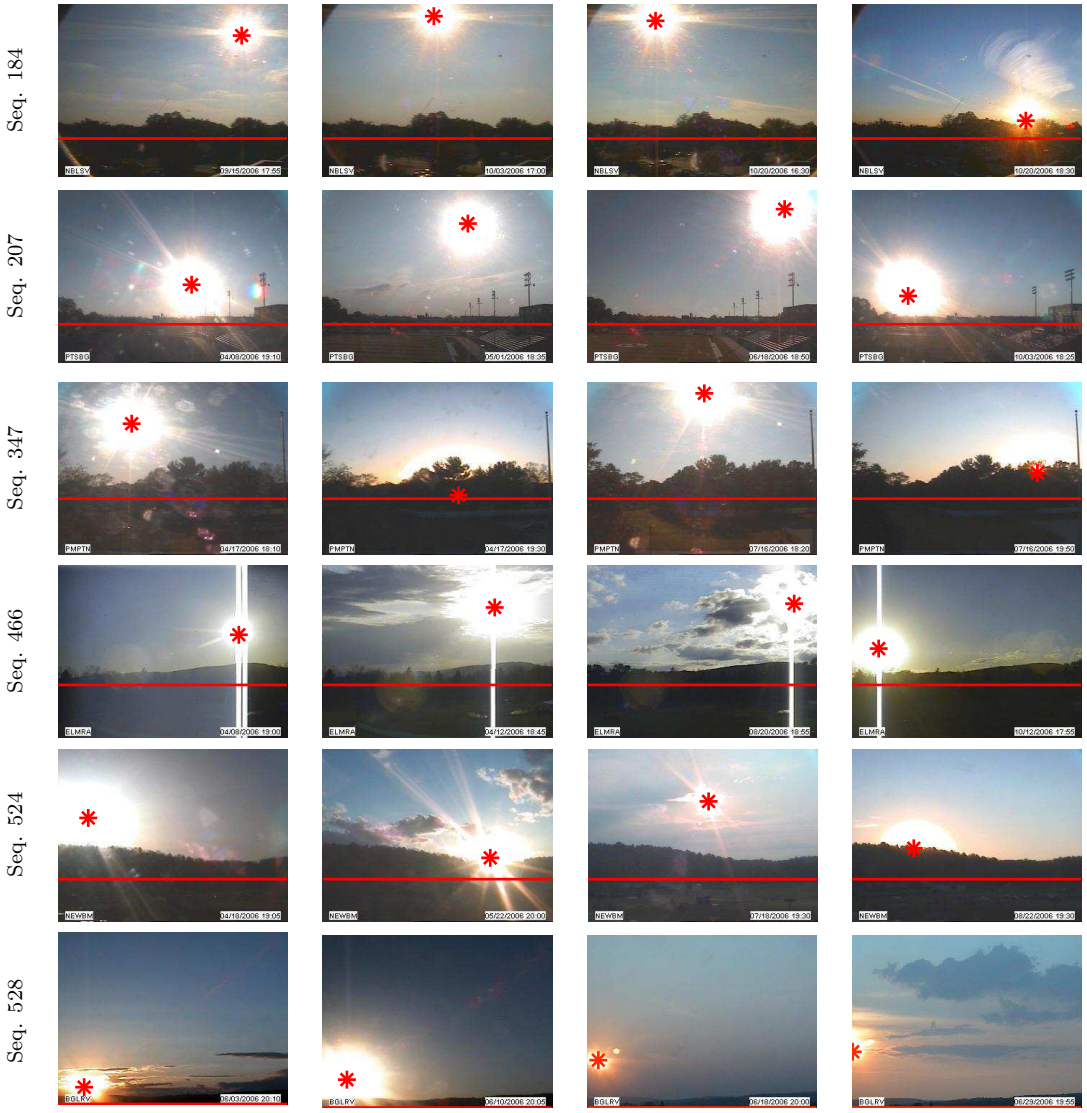


Figure 11: Visual evaluation of our camera calibration algorithm from the sun position. The camera parameters determined by our method are used to predict where the sun (red star) and the horizon (red line) will appear in images. Note that the sun position can be predicted even if it is occluded by the scene protruding above the horizon. The images shown here were not used in the optimization.

to the camera (see Fig. 8), so its influence might be subtle; 2) it is hard to model the appearance of the sun influence on individual images because it may come from different directions, unlike the sky gradient that is always vertical (19); and 3) $\mathcal{J}$ needs to contain images where the sun

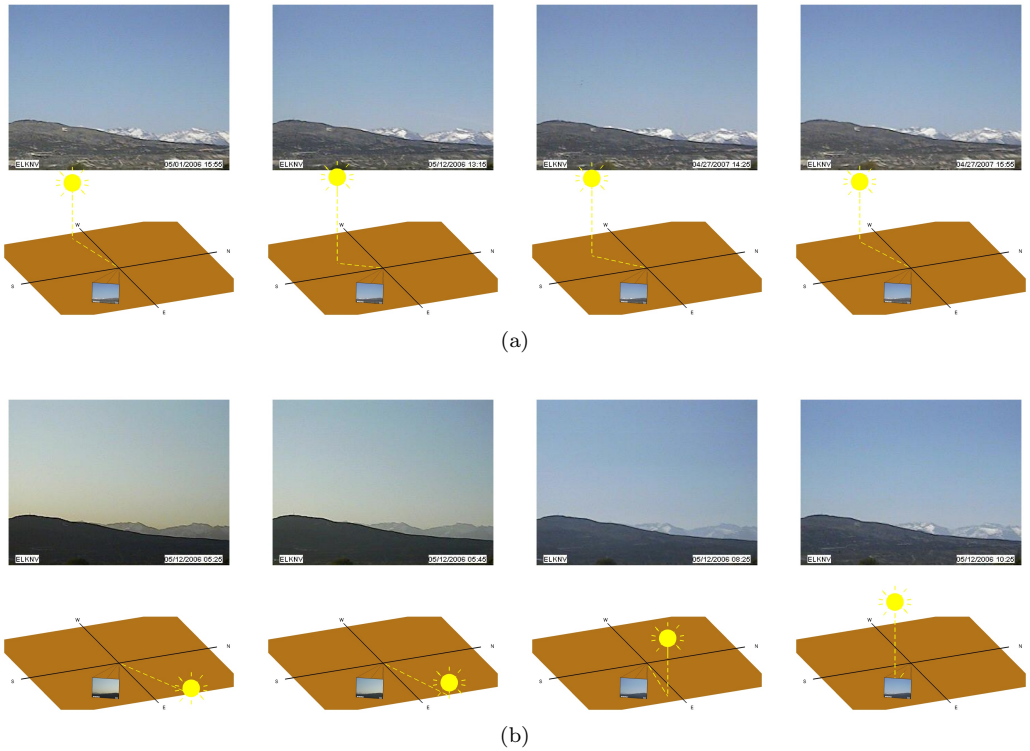


Figure 12: Clear sky images recovered automatically for sequence 414. Samples from the sets (a) $\mathcal{I}$, where the sun influence is not noticeable, and from (b) $\mathcal{J}$, where the sun induces changes in the sky appearance throughout the set. The actual sun position, relative to the camera, is shown in the bottom rows. Note that the relative sun position is never explicitly known to our image selection algorithm.

is at different positions to insure robustness in the estimation. Therefore, we must enforce that the recovered images be taken at different times of day. Instead of trying to find individual clear images, our strategy is to find clear *days*. We score each day in a sequence by counting how many images with $\alpha < 0$, and select the top 4 days, which should contain mostly clear skies. To filter out clouds that may appear, we then select the N smoothest images, based on their average u and v gradients (we use finite differences). Example images recovered by this algorithm are shown in Fig. 12b.

5.2.2 Visualizing a webcam dataset

Since the sun position might not be visible in the sequence, we cannot apply the same method for validation as we did when the sun is visible. Instead, we must rely on visible cues in the images, such as the horizon line, shadows, differently-lit building surfaces, etc. Fig. 13 shows qualitative results obtained by applying our method on 6 different sequences taken from AMOS database.

The recovered camera parameters are consistent with the sky appearance. For instance, the sun appears to be just to the right of the image in Fig. 13c-(e), which is reflected in the diagrams. In Fig. 13d, the buildings are brightly lit, which indicates that the sun must be behind the camera. Similarly, the tower in Fig. 13f is front-lit by the sun, but it has an orange hue, so the sun must be near the horizon line, and behind the camera as well. Shadows also hint to the sun position: right of the camera in Fig. 13a, and behind in Fig. 13b. We also note that the recovered horizon lines are aligned with the real one when visible, as in Fig. 13e-(f). The horizon line position in the image is predicted by (18).

5.3 Validation by consistency between the two approaches

When the sun is visible, we can run both algorithms on the same sequence and compare their estimates. Since both methods rely on two different sources of data (sun position and sky appearance), we quantify their performance by analyzing their consistency.

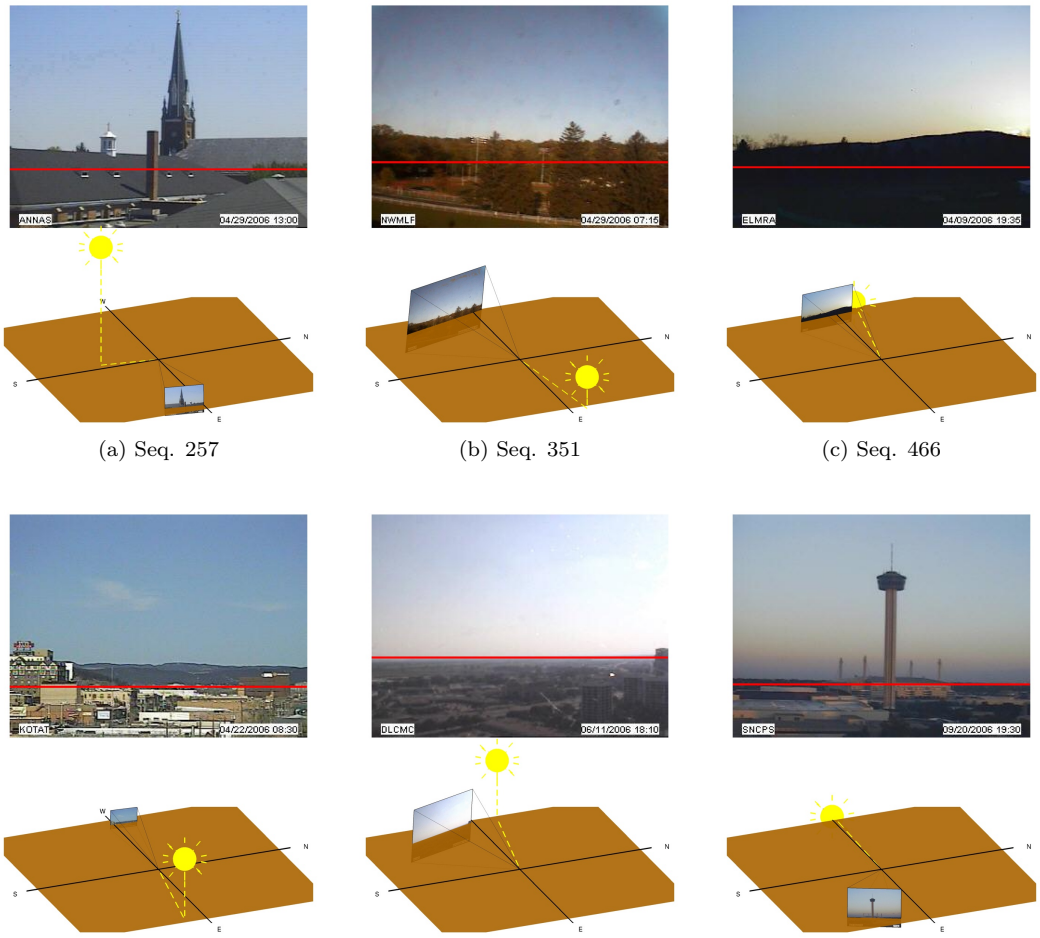
Numerical results of the estimates for the sun-based algorithm $(f_{sun}, \theta_{sun}, \phi_{sun})$, the sky-based algorithm $(f_{sky}, \theta_{sky}, \phi_{sky})$, and their agreement $(\Delta f, \Delta\theta, \Delta\phi)$ for all 22 sequences are shown in Table 6. Results for sequences indicated by ** are obtained *entirely automatically*. Automatic sky segmentation is performed by running the geometric context algorithm [13] on 40 randomly chosen images from the sequence, and averaging the sky probability map. The other sequences require some degree of manual intervention, either for specifying the sky segmentation, or for retrieving images for set **J**. In the latter case, the intervention consists of manually choosing 3 or 4 mostly clear days from the sequence (see Sec. 8 for more details).

Consistency in focal length is evaluated by using $\Delta f = \frac{|f_{sky} - f_{sun}|}{f_{sun}} \times 100$, and is found to be at most 9.3%, but typical values range from 1.8% to 6.2%. Consistency in zenith and azimuth angles is evaluated by computing the angular deviation. For the zenith angle, deviation is at most 6.5°, with typical values from 0.3° to 2.5°. For the azimuth angle, it is at most 8.1°, with typical values ranging from 1.2° to 4.3°.

Fig. 14 shows all the cameras drawn on the same plot, illustrating their difference in focal length, zenith and azimuth angles. The horizontal plane is shown in brown, and crosses each image at the horizon line. The parameters used to generate this drawing are recovered from the sky appearance algorithm. Since the sun is visible in all of them, the cameras either face East (sunrise) or West (sunset). This visualization provides an intuitive way to explore the cameras of a large dataset.

Sequence name and location	Focal length f_c (px)			Zenith angle θ_c (°)			Azimuth angle ϕ_c (°)		
	f_{sun}	f_{sky}	Δf (%)	θ_{sun}	θ_{sky}	$\Delta\theta$	ϕ_{sun}	ϕ_{sky}	$\Delta\phi$
207 (Cherokee, KS)	447.1	430.2	3.8	97.9	97.6	0.3	88.8	90.5	1.7
257** (Riva, MD)	941	951.4	1.1	96.9	93.4	3.5	257.9	260.5	2.6
279 (Lafayette, MI)	713.4	684.9	4	91.5	91.7	0.2	114.7	118.3	3.6
291 (Minneapolis, MN)	356.8	338.7	5.1	88.8	90.3	1.5	38.9	32.3	6.6
305 (Neosho, MO)	1039.5	973.3	6.3	98.1	98.7	0.6	108.8	109.3	0.5
347** (Butler, NJ)	387.5	372.3	3.9	96.3	97.5	1.4	80.2	81.8	1.6
351** (New Milford, NJ)	381.7	393.9	3.2	97.4	97.2	0.2	104.7	107.4	2.7
414** (Elburz, NV)	1067.4	1032.4	3.3	95.9	97.1	1.2	240.6	239.1	1.5
455 (Marathon, NY)	757.3	816.3	7.8	98.3	95.8	2.5	134.1	131.8	2.3
466 (Elmira, NY)	651.6	669.4	1.1	1.45	-0.06	0.1	93.4	95.5	2.1
513 (Hunker, PA)	1204	1134	5.8	92.3	92	0.3	73	71.9	1.1
524 (N. Bloomfield, PA)	670.3	664.7	0.8	95.4	95.3	0.1	71.2	68.6	2.6
529 (Mifflinburg, PA)	884.9	918.9	3.9	89.9	90.5	0.6	56	60.3	4.3
569 (Rapid City, SD)	1389.1	1362.6	1.9	93.4	92.8	0.6	84.2	85.1	0.9
601 (Mesquite, TX)	442.8	484.3	**9.3**	93.3	94.1	0.8	105.8	109.4	3.6
608** (Tyler, TX)	355.9	357.4	0.4	95.3	97.5	2.4	47.6	52	4.4
630 (Pleasanton, TX)	474.7	473.6	0.2	90.5	91.9	1.4	100.9	102.1	1.2
634 (San Antonio, TX)	665.8	706.6	6.1	98.2	95.2	3	242.4	244.4	2
652 (Delta, UT)	363.6	395.5	8.8	103.6	100.7	2.9	265.2	265.3	0.1
655 (Tropic, UT)	1430.8	1363.6	4.7	94.2	95.4	1.2	289.9	292.2	2.3
695** (Danville, VA)	700	714.8	2.1	92.4	92.4	0	59.3	51.3	**8**
701 (Darrington, WA)	632.2	652.2	3.2	105.2	98.7	**6.5**	124.8	130.4	5.6

Table 6: Quantitative comparison of camera calibration results obtained by the two methods presented in this paper: the sun position and the sky appearance. For each method, estimated values for f_c, θ_c and ϕ_c are shown and indicated by their corresponding subscripts. Consistency is evaluated by comparing the values together and is indicated by Δ. Worst consistency results are indicated in bold. All images have 320×240 pixels dimension. Results were automatically recovered for sequences indicated by **, the others required manual intervention at some stage in the process, either to define the sky region, or to select images for set $\mathcal{J}$.



(a) Seq. 257 (b) Seq. 351 (c) Seq. 466

(d) Seq. 569 (e) Seq. 601 (f) Seq. 634

Figure 13: Camera parameters recovered from the sky appearance. The horizon line is represented by the red line on the images. Below each image is a drawing illustrating the recovered camera-sun geometry. The sun is drawn at the location corresponding to the date and time of the image. The brown plane is horizontal, and intersects with the image plane at the horizon line.

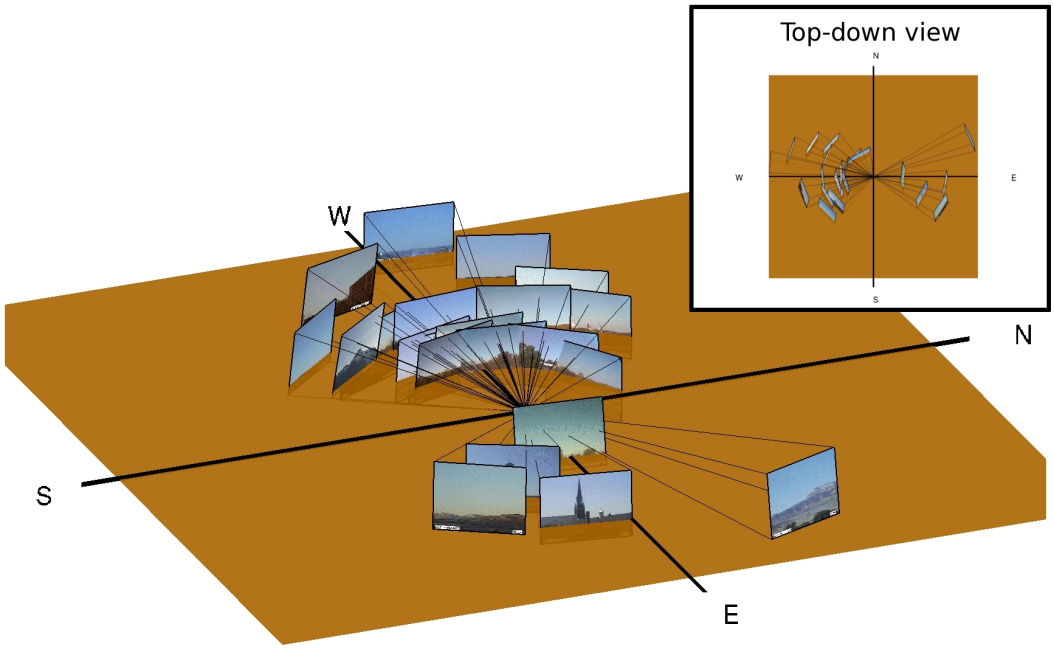


Figure 14: All the 22 cameras used in the consistency evaluation shown on the same display. The parameters used to generate the figure were obtained by using the sky-based algorithm. All cameras are either facing East or West, so the sun is visible at dawn or dusk respectively. The inset shows a top-down view for better visualization of the cameras azimuths.

6 Geolocating the camera using the sun and the sky

The techniques presented so far have been considering either the sun position or the sky appearance as two independent sources of information that could be used to recover the camera parameters. We now demonstrate that, by combining them, we can avoid the requirement of having to know the GPS location of the camera, and also estimate it.

Recovery of the GPS location, or geolocation, has been explored in a variety of scientific fields. In biology, scientists are tracking many marine animals using light sensors. The sunset and sunrise times are found by analyzing the light intensity profiles captured by these sensors, which are then used to geolocate the animals and track them [12]. In robotics, altimeters are used on outdoor mobile robots to accurately detect the sun position, and compute the GPS location from several observations [5]. In computer vision, Jacobs et al.[17] determine the position of webcams by correlating their intensity variations computed over several months with sunlight satellite images of the same period.

6.1 Algorithm

In our work, we show how we can estimate the latitude and longitude of the camera, as well as its geometric parameters, from an image sequence in which the sun and sky are visible. We introduce the following algorithm:

Algorithm 4: Camera localization from the sun and the sky

Input: Sun position in images: (u_s, v_s);
Input: Clear sky images;
Input: Date and time of capture of each image.

1 Apply Algorithm 2 on clear sky images to recover f_c and θ_c;
2 Compute the sun angles (θ_{si}, ϕ_{si}) from (f_c, θ_c) and the sun labels (u_s, v_s) using (22);
3 Solve the non-linear minimization (23) to estimate the latitude l and longitude L;
4 Solve the non-linear minimization (17) to recover the camera azimuth ϕ_c.

Output: Camera parameters: (f_c, θ_c, ϕ_c);
Output: Camera latitude and longitude: (l, L).

We now detail lines 2 and 3 of Algorithm 4: how to recover the latitude l and longitude L of the camera given the date and time of capture, the camera zenith angle θ_c and focal length f_c, as well as the sun position in images (u_s, v_s). Our approach relies on an equation expressing the sun zenith and azimuth angles θ_{sg} and ϕ_{sg}, as a function of time, date, latitude and longitude on Earth [31]:

$$\begin{aligned} \theta_{sg} &= \frac{\pi}{2} - \arcsin\left(\sin l \sin\delta - \cos l \cos\delta \cos\frac{\pi t}{12}\right) \\ \phi_{sg} &= \arctan\left(\frac{-\cos\delta \sin\frac{\pi t}{12}}{\cos l \sin\delta - \sin l \cos\delta \cos\frac{\pi t}{12}}\right) \quad , \end{aligned} \tag{20}$$

where δ is the solar declination in radians, and t is solar time in decimal hours. δ and t are given

by:

$$\begin{aligned} \delta &= 0.4093 \sin\left(\frac{2\pi(J-81)}{368}\right) \quad , \\ t &= t_s + 0.17 \sin\left(\frac{4\pi(J-80)}{373}\right) - 0.129 \sin\left(\frac{2\pi(J-8)}{355}\right) + \frac{12L}{\pi} \quad , \end{aligned} \tag{21}$$

where t_s is standard time in UTC coordinates in decimal hours, and J is the julian date (the day of the year as an integer in the range 1 to 365).

The relationship between the sun pixel coordinates (u_s, v_s) in the images and its zenith and azimuth angles θ_{si} and ϕ_{si} is the following:

$$\begin{aligned} \theta_{si} &= \arccos\left(\frac{v_s \sin\theta_c + f_c \cos\theta_c}{\sqrt{f_c^2 + u_s^2 + v_s^2}}\right) \\ \phi_{si} &= \arctan\left(\frac{f_c \sin\phi_c \sin\theta_c - u_s \cos\phi_c - v_s \sin\phi_c \cos\theta_c}{f_c \cos\phi_c \sin\theta_c + u_s \sin\phi_c - v_s \cos\phi_c \cos\theta_c}\right) \quad , \end{aligned} \tag{22}$$

see Appendix B for the derivation.

Note that at the ground truth GPS location, $\theta_{sg} = \theta_{si}$ and $\phi_{sg} = \phi_{si}$. We can therefore recover the GPS location by solving the following least-squares minimization problem:

$$\min_{l,L} \sum_{k=1}^{N} \angle(\overrightarrow{s}(\theta_{sg}^{(k)}, \phi_{sg}^{(k)}), \overrightarrow{s}(\theta_{si}^{(k)}, \phi_{si}^{(k)}))^2 \quad , \tag{23}$$

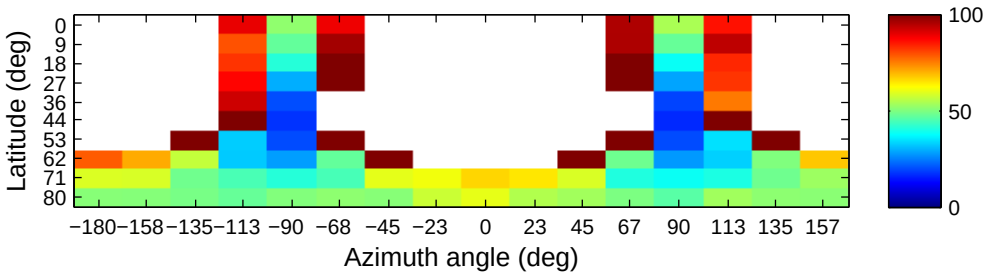
where N is the number of images where the sun has been labeled, $\angle(\cdot)$ denotes the angular difference, and $\overrightarrow{s}(\theta, \phi)$ is the vector obtained by expressing the angles (θ, ϕ) in Cartesian coordinates. A solution in l and L can be recovered using a non-linear least-squares optimizer. We have experimentally found that first minimizing the error on zenith angles $\angle(\theta_{sg}, \theta_{si})$ and using its solution to initialize (23) resulted in greater stability. This entire process is summarized in Algorithm 4.

6.2 Camera localization results

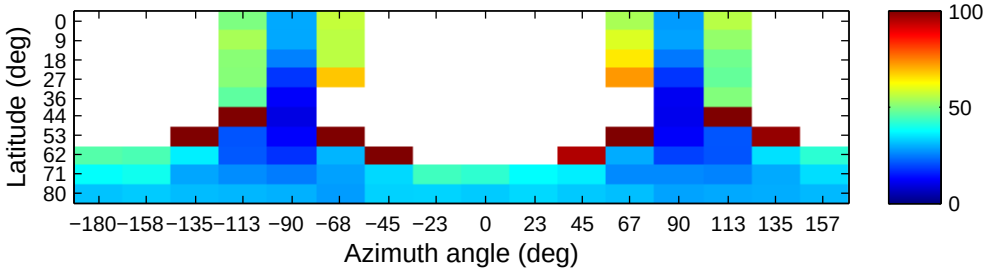
To evaluate the precision of our algorithm, we tested it over a large set of conditions using synthetic data, as well as on our ground truth sequence.

6.2.1 Synthetic data

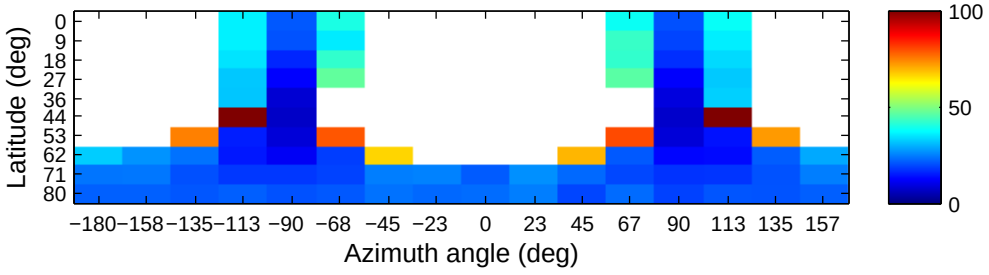
We evaluated the performance of our algorithm on synthetic data, obtained by varying the latitude l, longitude L, camera azimuth angle ϕ_c, and number of available images n. The resulting 4-dimensional parameter space was discretized the following way: 9° increments for both l and L, 23° increments for ϕ_c, and $n = 20, 50, 100$. For each point in the that parameter space, n sun positions are randomly generated according to (20) and (22) with gaussian noise of variance $\sigma^2 = 5px$. Each experiment is repeated 15 times to account for randomness, and the mean over all these tries are reported. Fig. 15 shows the errors (in km) obtained by our algorithm at every point in this parameter space. Since longitude does not seem to affect the precision of



(a) $n = 20$



(b) $n = 50$



(c) $n = 100$

Figure 15: Error in GPS coordinates estimation, in km, for different values of n: (a) 20, (b) 50, and (c). Since longitude does not seem to affect the precision of the results, the errors shown here are averaged over all longitudes. The white cells indicate configurations where the sun is never visible.

the results, the errors shown are averaged over all values of longitude. The white cells indicate configurations where the sun is never visible, so the GPS position cannot be recovered. When $n = 100$, the mean error is 25km.

Sequence name	Ground truth		Estimated		Error (km)
	Latitude (°)	Longitude (°)	Latitude (°)	Longitude (°)	
257	38.97	-76.61	38.47	-76.40	58.44
279	43.32	-84.60	42.06	-84.13	145.82
513	40.20	-79.61	39.42	-80.61	122.25
524	40.41	-77.13	40.09	-77.66	56.72
569	44.00	-103.24	45.21	-103.54	136.65
601	32.69	-96.62	31.72	-95.50	150.78
630	28.98	-98.50	28.58	-97.79	81.53
695	36.60	-79.38	37.77	-79.25	129.84

Table 7: Detail of localization results for 8 sequences taken from the AMOS dataset. On average, our method makes a localization error of 110km.

6.2.2 Ground truth results

We also applied our technique on 8 sequences from the AMOS database [15], where the ground truth GPS positions are known. We obtain a mean localization error of 110km (straight-line distance on the surface of the Earth), and the results for each individual sequence are shown in Table 7. On average, each sequence was localized by using 48 images as input.

7 Application: estimating clouds and sky turbidity

Now that we have recovered camera parameters, either from the sun position or sky appearance, we demonstrate how to use the same physically-based model to handle challenging weather conditions. Until now, we have only dealt with clear skies, but alas, this is not always true! In this section, we present a novel cloud segmentation algorithm which will allow us to deal with any type of weather.

Clouds exhibit a wide range of textures, colors, shapes, and even transparencies. Segmenting the clouds from the sky cannot be achieved with simple heuristics such as color-based thresholding as they are easily confounded by the variation in their appearances. On the other hand, our physically-based model predicts the sky appearance, so any pixel that differs from it is an outlier and is likely to correspond to a cloud. Using this intuition, we now consider two ways of fitting our model to skies that may contain clouds. We perform all processing in the xyY color space because it was determined that it offers the best agreement with the Perez sky model in [31].

7.1 Least-squares fitting

The first idea is to follow a similar approach as we did previously and fit the model (15) in a non-linear least-squares fashion by adjusting the coefficients a, b, c, d, e and the unknown scale factor k independently in each color channel. This approach was proposed in [42], and works quite well in the absence of clouds. When clouds are present, we observe that fitting 5 coefficients gives too much freedom to the model, so we constrain the optimization and reduce the number of variables by following [31] and expressing the five weather coefficients as a linear function of a single value, the turbidity t. Strictly speaking, this means minimizing over $\mathbf{x} = \begin{bmatrix} t & k^{(1)} & k^{(2)} & k^{(3)} \end{bmatrix}$:

$$\min_{\mathbf{x}} \sum_{l=1}^{3} \sum_{p \in \mathcal{P}} \left(y_p^{(l)} - k^{(l)} g(u_p, v_p, \theta_s, \phi_s, \tau^{(l)}(t)) \right)^2 \quad , \tag{24}$$

where l indexes the color channel. Here the camera parameters are fixed, so we omit them for clarity. The vector $\tau^{(l)}(t)$ represents the coefficients $(a, \ldots, e)$ obtained by multiplying the turbidity t with the linear transformation $M^{(l)}$: $\tau^{(l)}(t) = M^{(l)} \begin{bmatrix} t & 1 \end{bmatrix}^\mathsf{T}$. The entries of $M^{(l)}$ for the xyY space are given in the appendix in [31]. The $k^{(l)}$ are initialized to 1, and t to 2 (low turbidity). Unfortunately, solving this simplified minimization problem yields unsatisfying results. The L2-norm is not robust to outliers, so even a small amount of clouds will bias the results.

7.2 Regularized fitting

In order to increase robustness to outliers, we compute a data-driven prior model of clear skies $\mathbf{x_c}$, which we use to add 2 terms to (24): 1) we assign more weight to pixels we believe are part of the sky; and 2) we penalize parameters with a large L2 divergence from the prior. Equation (24) becomes

$$\min_{\mathbf{x}} \sum_{l=1}^{3} \sum_{p \in \mathcal{P}} w_p \left(y_p^{(l)} - k^{(l)} g(u_p, v_p, \theta_s, \phi_s, \tau^{(l)}(t)) \right)^2 + \beta \|\mathbf{x} - \mathbf{x_c}\|^2 \quad , \tag{25}$$

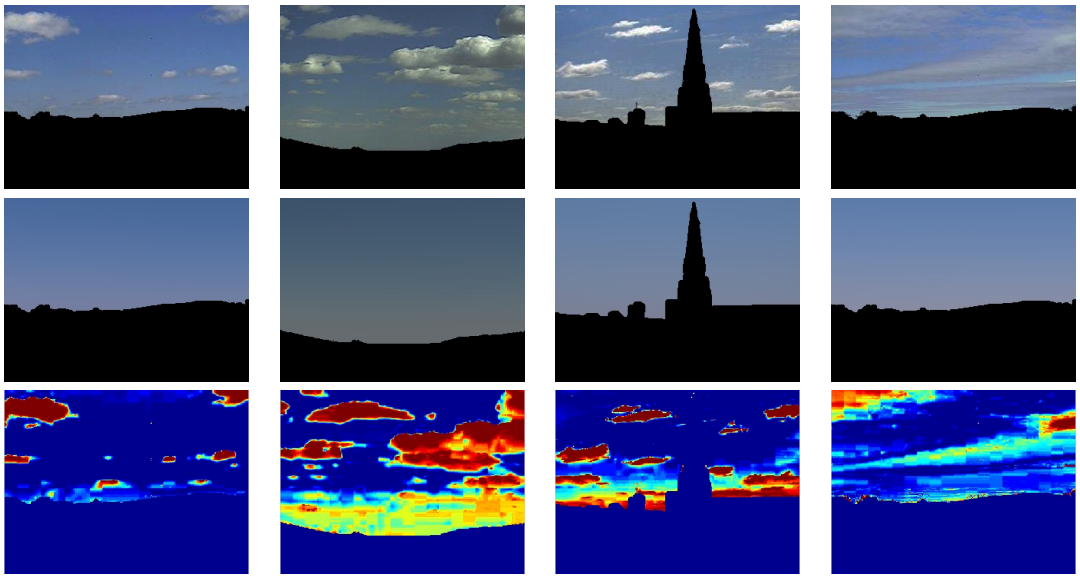


Figure 16: Sky-cloud separation example results. *First row*: input images (radiometrically corrected). *Second row*: sky layer. *Third row*: cloud segmentation. The clouds are color-coded by weight: 0 (blue) to 1 (red). Our fitting algorithm is able to faithfully extract the two layers in all these cases.

where, $w_p \in [0,1]$ is a weight given to each pixel, and $\beta = 0.05$ controls the importance of the prior term in the optimization. We initialize $\mathbf{x}$ to the prior $\mathbf{x_c}$.

Let us now look at how $\mathbf{x_c}$ is obtained. We make the following observation: clear skies should have low turbidities, and they should be smooth (i.e. no patchy clouds). Using this insight, if minimizing (24) on a given image yields low residual error and turbidity, then the sky must be clear. We compute a database of clear skies by keeping all images with turbidity less than a threshold (we use 2.5), and then keep the best 200 images, sorted by residual error. Given an image, we compute $\mathbf{x_c}$ by taking the mean over the K nearest neighbors in the clear sky database, using the angular deviation between sun positions as a distance measure (we use $K = 2$). This allows us to obtain a prior model of what the clear sky should look like at the current sun position. Note that we simply could have used the values for $(a, \ldots, e)$ from Sec. 4.1.1 and fit only the scale factors $k^{(l)}$, but this tends to over-constrain, so we fit t as well to remain as faithful to the data as possible. For example, the mean estimated turbidity is $t = 2.06$ for Sequence 257, very close to the clear sky model $t = 2.17$ used in Sec. 4.1.1.

To obtain the weights w_p in (25), the color distance λ between each pixel and the prior model is computed and mapped to the $[0,1]$ interval with an inverse exponential: $w_p = \exp\{-\lambda^2/\sigma^2\}$ (we use $\sigma^2 = 0.01$ throughout this paper). After the optimization is over, we re-estimate w_p based on the new parameters $\mathbf{x}$, and repeat the process until convergence, or until a maximum number of iterations is reached. The process typically converges in 3 iterations, and the final value for w_p is used as the cloud segmentation. Cloud coverage is then computed as $\frac{1}{|\mathcal{P}|}\sum_{p\in\mathcal{P}} w_p$.

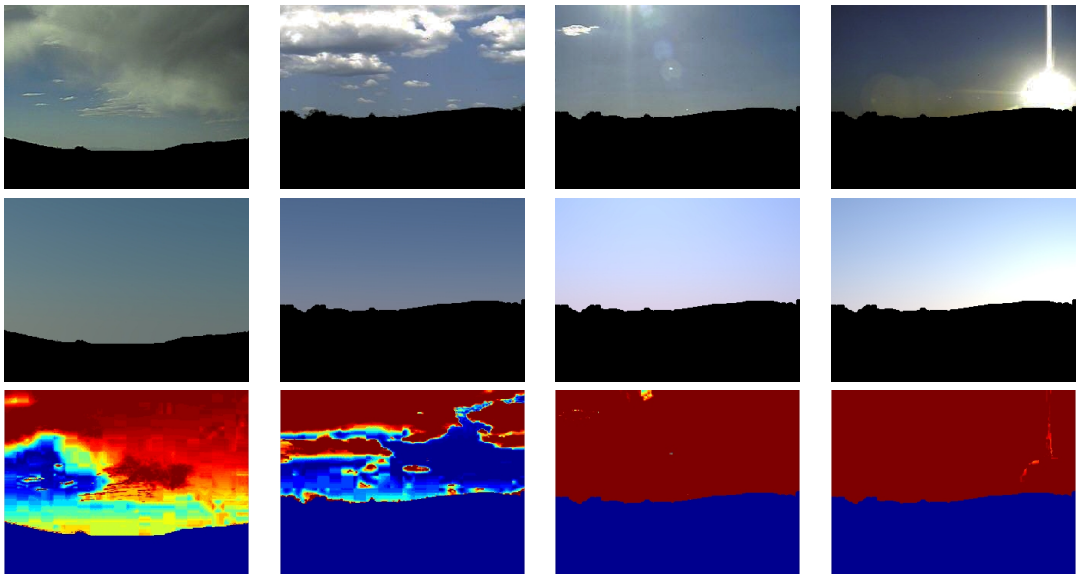


Figure 17: More challenging cases for the sky-cloud separation, and failure cases. *First row*: input images (radiometrically corrected). *Second row*: sky layer. *Third row*: cloud layer. The clouds are color-coded by weight: 0 (blue) to 1 (red). Even though the sky is more than 50% occluded in the input images, our algorithm is able to recover a good estimate of both layers. The last two columns illustrate a failure case: the sun (either when very close or in the camera field of view) significantly alters the appearance of the pixels such that they are labeled as clouds.

7.3 Results

Fig. 16 shows typical results of cloud layers extracted using our approach. Note that unweighted least-squares (24) fails on all these examples because the clouds occupy a large portion of the sky, and the optimization tries to fit them as much as possible, since the quadratic loss function is not robust to outliers. A robust loss function behaves poorly because it treats the sky pixels as outliers in the case of highly-covered skies, such as the examples shown in the first two columns of Fig. 17. Our approach injects domain knowledge into the optimization by using a data-driven sky prior, forcing it to fit the visible sky. Unfortunately, since we do not model sunlight, the estimation does not converge to a correct segmentation when the sun is very close to the camera, as illustrated in the last two columns of Fig. 17.

8 Discussion

Before concluding, we discuss three important problems related to the sky-based calibration algorithm that arise in practice, namely radiometric issues, varying weather conditions, and the need for date and time of capture. It is important to understand the various elements other than the camera parameters that may also affect the sky appearance, in order to factor out their influence and isolate the effects solely due to camera parameters.

8.1 Radiometric issues

Our sky-based algorithm relies on the sky pixel intensities and assumes that they faithfully represent the real sky radiance. Unfortunately, radiance undergoes a series of unknown transformations [20] before being observed as pixel intensities. In particular, we must consider gain, dynamic range, camera response function, vignetting, and sensor noise. In this section, we discuss how each one of these unknown transformations are dealt with in this paper.

Gain is an unknown scale factor which is applied to radiance, and can vary from one image to the next because of Automatic Gain Control (AGC). Our approach is insensitive to AGC because it estimates an unknown scale factor k at each image (see Sec. 4.1.3), in which the gain gets incorporated.

The exposure controls the amount of light that is captured by the camera, and may or may not vary across images depending on the camera. A particular exposure may result in under-exposed or saturated pixels when the corresponding scene is too dark or too bright, respectively. These incorrectly-exposed pixel values have been truncated to fit the dynamic range of the camera, therefore are not accurate representations of the scene radiance. This problem can be solved by ignoring pixels that have intensity less than 2/255 or higher than 254/255 in the optimizations.

The camera response function is a (typically non-linear) transformation that maps radiance values to pixel intensities. This is usually computed by acquiring several images of the same scene at different exposures [7]. Unfortunately, we cannot assume this is the case in an image sequence because the frequency of acquisition might be too low, and illumination conditions might be different from one frame to the next which breaks the constant radiance assumption of such methods. Instead, we mentioned that we rely on [27], which estimates the response function from color edges computed over several images. This method suffers from two important drawbacks: 1) selection of the weight λ, which controls the relative importance between the data and prior terms in the optimization, has to be done empirically; and 2) the images might not have enough different colors to cover the entire RGB cube, so the set of available edges might be restricted to a small region in the color space. Future work includes recovering the camera response function from techniques which are geared towards using multiple images from the same [19] or different [22] scenes as input.

Vignetting is a common issue that arises when dealing with low-quality, wide-angle lenses typical of webcams. It can significantly alter the intensity of pixels located near the corners of the image. Although elegant vignetting removal solutions have been proposed [20, 22, 43], we simply ignore pixels that are far from the image center (e.g. 120 pixels for a 320×240 pixel image), and have found this approximation to be sufficient with our test sequences.

The last issue is the one of sensor noise, which can be significant in low-quality webcam images. Because our algorithm operates on several randomly-chosen sky pixels gathered across many images, and the Gaussian noise assumption underlying our least-squares minimization ap-

proach, we have found our algorithm to be robust to the noise level present in our test sequences, which is also confirmed by the synthetic experiments performed in Sec. 4.3.2.

8.2 Weather conditions

Although we already presented in Sec. 7 how we can represent challenging weather conditions once the camera parameters have been recovered, recall that finding these very parameters relied on clear sky images in the first place. We presented in Sec. 5.2.1 two algorithms that automatically select clear skies to build sets $\mathcal{I}$ and $\mathcal{J}$ from a large set of images. Unfortunately, because camera parameters are unknown initially, we had to rely on image-based heuristics to guide these algorithms. We now discuss how these algorithms are affected by slight variations in weather conditions, which in turn has an effect on the camera parameters estimation. Luckily, both of them need not be perfect, and we observe that they are robust to clouds being present in roughly $10 - 15\%$ of their respective input image sets.

Empirically, we observe that building set $\mathcal{I}$ (clear skies where the sun does not affect the sky appearance) succeeds in approximately 90% of the time. The main failure case is when a thin layer of semi-transparent clouds cover the entire image, and smoothly modify the vertical sky gradient. Additionally, presence of large amounts of haze close to the horizon is another source of noise because it is not predicted by the clear sky model. For the set $\mathcal{J}$ (clear skies where the moving sun effect is visible), performance decreases and the automatic method is used in only 25% of the sequences in Table 6. The algorithm typically fails when the sun is very close to the camera field of view, and induces very large changes in the sky appearance. Unfortunately, these failure cases can only be detected by manually inspecting the resulting images, so future work includes determining a better way of finding clear sky images that is more robust to stronger variations in weather.

8.3 Are date and time of capture necessary?

We have shown in Algorithm 2 that, given only clear sky images, it is possible to estimate the camera focal length and zenith angle. But can we go further? Could we also recover the azimuth angle, and even GPS coordinates, given just the images as input?

One possibility would be to have a webcam which captures at precise regular intervals, closely spaced in time (e.g. every minute), over a very long period of time (e.g. one year). In short, this regular time spacing gives their time of capture up to translation and scale. From only the images of such a webcam, it should be possible to track the sun position and get a good estimate of sunset and sunrise (i.e. when the predicted θ_s given θ_c and f_c is equal to $90°$). Given many sunset or sunrise estimates, it might be possible to recover the time translation and scale factor by correlating their *relative* sunset/sunrise times with real times gathered from an astronomical almanac. This could be used to recover the actual date and time of capture of each image.

Unfortunately, real webcams are not so regular: their capture frequency may vary slightly, they might become unavailable for a period of time, etc. Dropping a single frame would adversely effect the algorithm, so the feasibility of such an approach imposes undue restrictions on data capture. The date and time of capture are stored with virtually all captured images and hence can be exploited, thus avoiding such restrictions on image acquisition.

9 Summary

In this paper, we analyze two sources of information available within the visible portion of the sky region: the sun *position*, and the sky *appearance*. From the sun coordinates in images, we show how we can extract the camera focal length and its zenith and azimuth angles. For the sky appearance, we express a well-known physically-based sky model in terms of these camera parameters and fit it to clear sky images using standard minimization techniques. We test our methods on a high-quality image sequence with known camera parameters, and obtain errors of less that 1% for the focal length, 1° for azimuth angle and 3° for zenith angle. We then show that both these techniques consistently recover the same parameters on synthetic and real image sequences. We evaluate their performance by calibrating 22 real, low-quality image sequences distributed over a wide range of latitudes and longitudes. Finally, we demonstrate that by combining the information available within the sun position and the sky appearance, we can also estimate the camera geolocation, as well as its geometric parameters. Our method achieves a mean localization error of 110km on real, low-quality Internet webcams. Once the camera parameters are estimated, we show how we can use the same model to segment out clouds from sky and build a novel bi-layered representation. We now plan to use the proposed sky illumination model to see how it can help us predict the illumination of the scene.

Acknowledgements

The authors would like to thank Nathan Jacobs for sharing his webcam dataset and for fruitful discussions. We would also like to thank Tom Stepleton, and especially Mohit Gupta for very helpful suggestions in reviewing this paper. This research is supported in parts by an ONR grant N00014-08-1-0330 and NSF grants IIS-0643628, CCF-0541307 and CCF-0541230. A. Efros is grateful to the WILLOW team at ENS Paris for their hospitality. Parts of the results presented in this paper have previously appeared in [25].

A Calibrating the camera from the sun position: derivation of the linear system of equations

In Sect. 3, we presented an overview of the method employed to find an initial estimate of the camera parameters from the sun position gathered over several frames. For completeness, we now present all the details of the derivation.

Recall the following goal: we wish to recover the projection matrix $\mathbf{M}$, which we constrain to be of the form $\mathbf{M} = \mathbf{K}\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}$, where $\mathbf{R}$ and $\mathbf{K}$ are defined in (3) and (4) respectively. Written explicitely, we have:

$$\mathbf{M} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix} = \begin{bmatrix} f_c \sin\phi_c & -f_c \cos\phi_c & 0 & 0 \\ -f_c \cos\phi_c \cos\theta_c & -f_c \sin\phi_c \cos\theta_c & f_c \sin\theta_c & 0 \\ \cos\phi_c \sin\theta_c & \sin\phi_c \sin\theta_c & \cos\theta_c & 0 \end{bmatrix}. \tag{26}$$

Each observation is a pair of sun 3-D coordinates $\mathbf{p}$, and its corresponding location in an image (u_s, v_s). If $\mathbf{m}_i$ is the ith row of $\mathbf{M}$, then each observation defines two equations (following [10]):

$$\begin{aligned}(\mathbf{m}_1 - u_s\mathbf{m}_3) \cdot \mathbf{s} = 0 \ , \\ (\mathbf{m}_2 - v_s\mathbf{m}_3) \cdot \mathbf{s} = 0 \ .\end{aligned} \tag{27}$$

If we have N such observations, we can write the linear system of equations from (27) directly in matrix notation with the form $\mathbf{Pm} = \mathbf{0}$:

$$\begin{bmatrix} x_w^{(1)} & y_w^{(1)} & 0 & 0 & 0 & -u_s^{(1)}x_w^{(1)} & -u_s^{(1)}y_w^{(1)} & -u_s^{(1)}z_w^{(1)} \\ 0 & 0 & x_w^{(1)} & y_w^{(1)} & z_w^{(1)} & -v_s^{(1)}x_w^{(1)} & -v_s^{(1)}y_w^{(1)} & -v_s^{(1)}z_w^{(1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_w^{(N)} & y_w^{(N)} & 0 & 0 & 0 & -u_s^{(N)}x_w^{(N)} & -u_s^{(N)}y_w^{(N)} & -u_s^{(N)}z_w^{(N)} \\ 0 & 0 & x_w^{(N)} & y_w^{(N)} & z_w^{(N)} & -v_s^{(N)}x_w^{(N)} & -v_s^{(N)}y_w^{(N)} & -v_s^{(N)}z_w^{(N)} \end{bmatrix} \begin{bmatrix} m_{11} \\ m_{12} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{31} \\ m_{32} \\ m_{33} \end{bmatrix} = \mathbf{0}. \tag{28}$$

This is a system of $2N$ equations and 8 unknowns. When $N \geq 4$, homogeneous linear least-squares can be used to compute the value of the vector $\mathbf{m}$ that minimizes $|\mathbf{Pm}|^2$ as the solution of an eigenvalue problem.

Because $\mathbf{M}$ is rank-deficient (rank = 2), it can only be recovered up to an unknown scale factor. However, observe that the third row in $\mathbf{M}$ must have unit length [10], so we normalize $\mathbf{m}$ by $\epsilon = \pm\sqrt{m_{31}^2 + m_{32}^2 + m_{33}^3}$ before applying (29). After normalization, the camera parameters (f_c, θ_c, ϕ_c) can be recovered by:

$$\begin{aligned}\theta_c &= \arctan\left(\frac{\sqrt{m_{31}^2 + m_{32}^2}}{m_{33}}\right) \\ f_c &= \sqrt{m_{11}^2 + m_{12}^2} \\ \phi_c &= \arctan\left(\frac{m_{11}}{-m_{12}}\right) \ .\end{aligned} \tag{29}$$

The sign of ϵ is chosen such that the points $\mathbf{s}$ lie in front of the camera (i.e. have positive x coordinates).

B Expressing the sky model as a function of camera parameters: full derivation

In Sect. 4.1.3, we presented a way to express the sky model as a function of camera parameters, which made the assumption that the camera zenith and azimuth angles were independent in order to come up with a simpler model. In this appendix, we derive the exact expressions for θ_p and ϕ_p, the zenith and azimuth angles corresponding to a pixel at coordinates (u_p, v_p) in the image, as illustrated in Fig. 5.

We first convert the (u_p, v_p) coordinates to a point $\mathbf{s}'$ in the 3-D camera reference frame $(\mathbf{x}_c, \mathbf{y}_c, \mathbf{z}_c)$, and then rotate it to align it with the global reference frame $(\mathbf{x}_w, \mathbf{y}_w, \mathbf{z}_w)$. The coordinates of the point in the camera reference frame are

$$\mathbf{s}' = \begin{bmatrix} x'_s \\ y'_s \\ z'_s \end{bmatrix} = \begin{bmatrix} f_c \\ -u_p \\ v_p \end{bmatrix} \quad . \tag{30}$$

The rotation that maps the point $\mathbf{s}'$ in the camera reference frame to a point $\mathbf{s}$ in the world reference frame is given by R^{-1}, where R has already been defined in (3). We apply the rotation to express the point in the world reference frame:

$$\mathbf{s} = \begin{bmatrix} x_s \\ y_s \\ z_s \end{bmatrix} = R^{-1}\mathbf{s}' \quad . \tag{31}$$

Finally, the angles are obtained by converting into spherical coordinates:

$$\theta_p = \arccos\left(\frac{z_s}{\sqrt{x_s^2 + y_s^2 + z_s^2}}\right), \quad \phi_p = \arctan\left(\frac{y_s}{x_s}\right) \quad . \tag{32}$$

We obtain the final, exact equations for θ_p and ϕ_p by substituting (30) into (31), and the resulting expression into (32):

$$\theta_p = \arccos\left(\frac{v_p \sin\theta_c + f_c \cos\theta_c}{\sqrt{f_c^2 + u_p^2 + v_p^2}}\right) \tag{33}$$

$$\phi_p = \arctan\left(\frac{f_c \sin\phi_c \sin\theta_c - u_p \cos\phi_c - v_p \sin\phi_c \cos\theta_c}{f_c \cos\phi_c \sin\theta_c + u_p \sin\phi_c - v_p \cos\phi_c \cos\theta_c}\right) \quad . \tag{34}$$

C Determination of minimum angular difference for the azimuth-independent sky model

In this appendix, we elaborate on the synthetic experiments that are performed in order to evaluate the conditions in which our azimuth-independent sky model (9) introduced in Sect. 4.1.2 is valid. As in Sect. 4, we consider only clear skies where turbidity $t = 2.17$ (see the first row of Fig. 6 for a visualization of the Perez sky model (7) at that particular turbidity).

We proceed to evaluate the influence of the azimuth-dependent component of the Perez sky model (second factor in (7)). Our goal is to determine sun-camera configurations where that influence is mostly constant over the image. Given the field of view of the camera, we generate synthetic sky images over all possible sun positions. More precisely, we generate images that cover the sun zenith angle $\theta_s \in [0, \frac{\pi}{2}]$, and the sun relative azimuth angle $\Delta\phi_s = \phi_s - \phi_c$ with respect to the camera $\Delta\phi_s \in [-\pi, \pi]$. For each synthetic image, we then compute:

$$r = \frac{\max(c^*)}{\min(c^*)} \quad , \tag{35}$$

where c^* is the mean column in the image, computed over the visible sky region only. In other words, we summarize the effect of the sun on an image by a single value r, which captures how the sky columns are affected. When $r = 1$, the sun has no effect on the sky. Unfortunately, this is never the case, as the sun will *always* have some effect on the sky when it is clear. Therefore, we approximate that the sun has little effect when $r \leq 1.1$. In Fig. 18, we plot r over the entire $(\theta_s, \Delta\phi_s)$ space. The white lines are the $r = 1.1$ isocontours, and the shaded regions indicate configurations where $r < 1.1$.

For the typical case of 35° field of view shown in Fig. 18b, we can safely affirm that when the sun is at least 100° away from the camera field of view, then $r \leq 1.1$, except in a region located immediately behind the camera where it rises up to $r = 1.2$. When the field of view diminishes to 20° as in Fig. 18a, then the number of sun-camera configurations where $r \leq 1.1$ is much larger, as the sun has to be closer to the camera to induce a noticeable influence on the sky appearance. The opposite effect is observed in the case of a larger field of view, as shown in Fig. 18c.

In conclusion, we used synthetic experiments to explore the validity of our azimuth-independent sky model (9), and we showed that for standard cameras, the sun has little influence on the sky when it is at least 100° away from the camera field of view.

References

[1] D. Bitouk, N. Kumar, S. Dhillon, P. N. Belhumeur, and S. K. Nayar. Face swapping: automatically replacing faces in photographs. *ACM Transactions on Graphics (SIGGRAPH 2008)*, 27(3), 2008. 6

[2] S. D. Buluswar and B. A. Draper. Color models for outdoor machine vision. *Computer Vision and Image Understanding*, 85(2):71–99, 2002. 6

[3] C.-C. Chiao and T. W. Cronin. Color signals in natural scenes: characteristics of reflectance spectra and effects of natural illumination. *J. Opt. Soc. Am. A*, 17(2), February 2000. 6

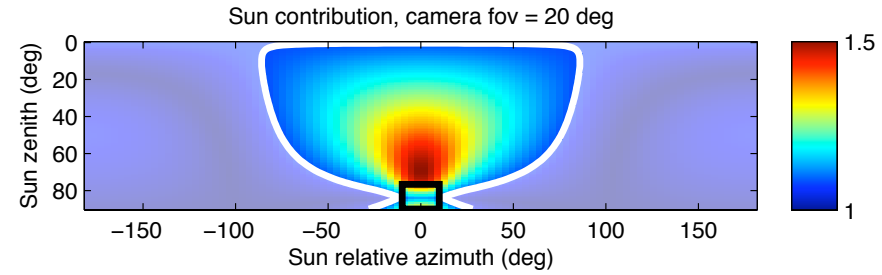
[4] C. T. Committee. Spatial distribution of daylight – luminance distributions of various reference skies. Technical Report CIE-110-1994, International Commission on Illumination, 1994. 14

[5] F. Cozman and E. Krotkov. Robot localization using a computer vision sextant. In *IEEE International Conference on Robotics and Automation*, 1995. 7, 31

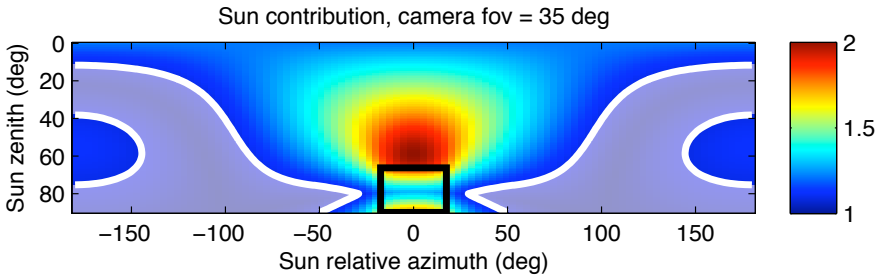
[6] P. Debevec. Rendering synthetic objects into real scenes: bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proceedings of ACM SIGGRAPH 1998*, 1998. 6

[7] P. Debevec and J. Malik. Recovering high dynamic range radiance maps from photographs. In *Proceedings of ACM SIGGRAPH 1997*, August 1997. 38

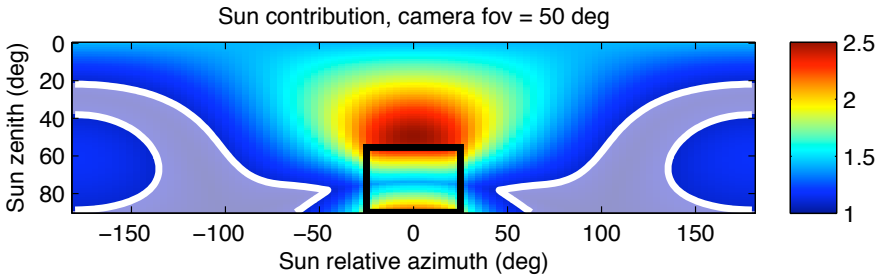
[8] P. Debevec, C. Tchou, A. Gardner, T. Hawkins, C. Poullis, J. Stumpfel, A. Jones, N. Yun, P. Einarsson, T. Lundgren, M. Fajardo, and P. Martinez. Estimating surface reflectance properties of a complex scene under captured natural illumination. Technical Report ICT-TR-06.2004, USC ICT, 2004. 7



(a)



(b)



(c)

Figure 18: Synthetic experiments to evaluate the influence of the sun on the sky gradient, generated for different camera fields of view: (a) 20°, (b) 35°, and (c) 50°. Each value is obtained by first synthesizing the sun-dependent component of the Perez sky model (second term in (7)) at the corresponding sun zenith θ_s and relative azimuth $(\phi_s - \phi_c)$ angles. The white lines are the isocontours corresponding to $r = 1.1$, and the shaded areas indicate that $r < 1.1$, where r is defined in (35). The black rectangles indicate the camera field of view.

[9] R. O. Dror, A. S. Willsky, and E. H. Adelson. Statistical characterization of real-world illumination. *Journal of Vision*, 4:821–837, 2004. 6

[10] D. A. Forsyth and J. Ponce. *Computer vision a modern approach*. Prentice Hall, 2003. 8, 9, 41

[11] R. Gross, S. Baker, I. Matthews, and T. Kanade. Face recognition across pose and illumination. In S. Z. Li and A. K. Jain, editors, *Handbook of Face Recognition*. Springer-Verlag, June 2004. 6

[12] R. Hill. Theory of geolocation by light levels. In B. J. LeBouef and R. M. Laws, editors, *Elephant Seals: Population Ecology, Behavior, and Physiology*, chapter 12, pages 227–236. University of California Press, 1994. 31

[13] D. Hoiem, A. A. Efros, and M. Hebert. Geometric context from a single image. In *IEEE International Conference on Computer Vision*, 2005. 5, 27

[14] P. Ineichen, B. Molineaux, and R. Perez. Sky luminance data validation: comparison of seven models with four data banks. *Solar Energy*, 52(4):337–346, 1994. 14

[15] N. Jacobs, N. Roman, and R. Pless. Consistent temporal variations in many outdoor scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007. 5, 7, 23, 34

[16] N. Jacobs, N. Roman, and R. Pless. Toward fully automatic geo-location and geo-orientation of static outdoor cameras. In *Workshop on applications of computer vision*, 2008. 7

[17] N. Jacobs, S. Satkin, N. Roman, R. Speyer, and R. Pless. Geolocating static cameras. In *IEEE International Conference on Computer Vision*, 2007. 6, 31

[18] E. A. Khan, E. Reinhard, R. Fleming, and H. Büelthoff. Image-based material editing. *ACM Transactions on Graphics (SIGGRAPH 2006)*, August 2006. 6

[19] S. J. Kim, J.-M. Frahm, and M. Polleyfeys. Radiometric calibration with illumination change for outdoor scene analysis. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008. 6, 38

[20] S. J. Kim and M. Polleyfeys. Robust radiometric calibration and vignetting correction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(4), April 2008. 38

[21] S. J. Koppal and S. G. Narasimhan. Clustering appearance for scene analysis. In *IEEE International Conference on Computer Vision*, 2006. 7

[22] S. Kuthirummal, A. Agarwala, D. B. Glodman, and S. K. Nayar. Priors for large photo collections and what they reveal about cameras. In *European Conference on Computer Vision*, 2008. 38

[23] J.-F. Lalonde, D. Hoiem, A. A. Efros, C. Rother, J. Winn, and A. Criminisi. Photo clip art. *ACM Transactions on Graphics (SIGGRAPH 2007)*, 26(3), August 2007. 6

[24] J.-F. Lalonde, S. G. Narasimhan, and A. A. Efros. Camera parameters estimation from hand-labelled sun positions in image sequences. Technical Report CMU-RI-TR-08-32, Robotics Institute, Carnegie Mellon University, July 2008. 12

[25] J.-F. Lalonde, S. G. Narasimhan, and A. A. Efros. What does the sky tell us about the camera? In *European Conference on Computer Vision*, 2008. 40

[26] J.-F. Lalonde, A. A. Efros, and S. G. Narasimhan. Webcam clip art: Appearance and illuminant transfer from time-lapse sequences. *ACM Transactions on Graphics (SIGGRAPH Asia 2009)*, 2009. 6

[27] S. Lin, J. Gu, S. Yamazaki, and H.-Y. Shum. Radiometric calibration from a single image. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2004. 5, 23, 38

[28] R. Manduchi. Learning outdoor color classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1713–1723, November 2006. 6

[29] S. G. Narasimhan, C. Wang, and S. K. Nayar. All images of an outdoor scene. In *European Conference on Computer Vision*, pages 148–162, May 2002. 6

[30] R. Perez, R. Seals, and J. Michalsky. All-weather model for sky luminance distribution – preliminary configuration and validation. *Solar Energy*, 50(3):235–245, March 1993. 7, 14

[31] A. J. Preetham, P. Shirley, and B. Smits. A practical analytic model for daylight. In *Proceedings of ACM SIGGRAPH 1999*, August 1999. 7, 14, 31, 35

[32] I. Reda and A. Andreas. Solar position algorithm for solar radiation applications. Technical Report NREL/TP-560-34302, National Renewable Energy Laboratory, November 2005. 9, 10, 19

[33] Y. Sato and K. Ikeuchi. Reflectance analysis under solar illumination. In *Proceedings of the IEEE Workshop on Physics-Based Modeling and Computer Vision*, pages 180–187, 1995. 6

[34] D. Slater and G. Healey. What is the spectral dimensionality of illumination functions in outdoor scenes? In *IEEE Conference on Computer Vision and Pattern Recognition*, 1998. 6

[35] J. Stumpfel, A. Jones, A. Wenger, C. Tchou, T. Hawkins, and P. Debevec. Direct HDR capture of the sun and sky. In *Proceedings of AFRIGRAPH*, 2004. 7

[36] K. Sunkavalli, W. Matusik, H. Pfister, and S. Rusinkiewicz. Factored time-lapse video. *ACM Transactions on Graphics (SIGGRAPH 2007)*, 26(3), August 2007. 6

[37] K. Sunkavalli, F. Romeiro, W. Matusik, T. Zickler, and H. Pfister. What do color changes reveal about an outdoor scene? In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008. 7

[38] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: principles and practice of background maintenance. In *IEEE International Conference on Computer Vision*, 1999. 6

[39] A. Trebi-Ollennu, T. Huntsberger, Y. Cheng, E. T. Baumgartner, B. Kennedy, and P. Schenker. Design and analysis of a sun sensor for planetary rover absolute heading detection. *IEEE Transactions on Robotics and Automation*, 17(6):939–947, December 2001. 7

[40] Y. Tsin, R. T. Collins, V. Ramesh, and T. Kanade. Bayesian color constancy for outdoor object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2001. 6

[41] Y. Weiss. Deriving intrinsic images from image sequences. In *IEEE International Conference on Computer Vision*, 2001. 6

[42] Y. Yu and J. Malik. Recovering photometric properties of architectural scenes from photographs. In *Proceedings of ACM SIGGRAPH 1998*, July 1998. 6, 7, 35

[43] Y. Zheng, S. Lin, and S. B. Kang. Single-image vignetting correction. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2006. 38