Validating the Contextual Information of Outdoor Images for Photo Misuse Detection

Xiaopeng Li\textsuperscript{*}, Xianshan Qu\textsuperscript{*}, Wenyuan Xu\textsuperscript{†}, Song Wang\textsuperscript{*}, Yan Tong\textsuperscript{*} and Lannan Luo\textsuperscript{*}
\textsuperscript{*}University of South Carolina, Columbia \quad \textsuperscript{†}Zhejiang University

Abstract—The contextual information (i.e., the time and location) in which a photo is taken can be easily tampered with or falsely claimed by forgers to achieve malicious purposes, e.g., creating fear among the general public. A rich body of work has focused on detecting photo tampering and manipulation by verifying the integrity of image content. Instead, we aim to detect photo misuse by verifying the capture time and location of photos. This paper is motivated by the law of nature that sun position varies with the time and location, which can be used to determine whether the claimed contextual information corresponds with the sun position that the image content actually indicates. Prior approaches to inferring sun position from images mainly rely on vanishing points associated with at least two shadows, while we propose novel algorithms which utilize only one shadow in the image to infer the sun position. Meanwhile, we compute the sun position by applying astronomical algorithms which take as input the claimed capture time and location. Only when the two estimated sun positions are consistent can the claimed contextual information be genuine. We have developed a prototype called IMAGEGUARD. The experimental results show that our method can successfully estimate sun position and detect the time-location inconsistency with high accuracy. By setting the thresholds to be $9.4^\circ$ and $5^\circ$ for the sun position distance and the altitude angle distance, respectively, our system can correctly identify $91.5\%$ of falsified photos with fake contextual information.

Index Terms—Photo misuse · Capture time and location · Sun position · Shadows · Projective geometry.

I. INTRODUCTION

With the rapid development of digital technologies and Internet, photos have become increasingly pervasive in our daily life for conveying information. For example, people use photos to express emotions, share ideas, and illustrate news stories on social media \cite{1,2,3}. However, not all of the photos are of good quality and used in proper ways. The distribution of fake photos on social media for malicious purposes is a growing concern \cite{4}, which may undermine our trust in photography, interfere with law enforcement, and compromise national security, media, commerce, and more \cite{5,6}. Take the Hurricane Sandy happened at the northeastern U. S. in 2012 as an example. Numerous fake disaster photos and rumors were spread through social networks and caused panic and fear among the general public \cite{7}. As a result, the U.S. Federal Emergency Management Agency has set up a “rumor control” section to defend against misleading information caused by fake photos on social networks \cite{8}.

Modifying the content of an image is one of the most common methods to create fake photos; thus, most existing techniques focus on identifying falsified content within a photo to detect whether or not the photo is fake (or tampered) \cite{9,10,11}. For example, some approaches are proposed to detect copy-move manipulation in digital images \cite{9,10,11}, and some others are proposed to detect photo tampering by leveraging shadows and lighting of images \cite{12,13}.

However, sometimes an adversary does not modify the content of a photo, but instead directly misuses the photo and claims the photo is taken at a different time or location, to achieve malicious purposes, such as spreading misleading information, attacking and deceiving others, or generating chaos and panic. In such a case, existing countermeasures would fail to detect such fake photos as their content is intact. Consider the example of Figure 1 which shows a clip of an explosion video\cite{8}. This video was spread on social networks and claimed to be associated to the explosion occurred in Xuzhou, China on June 15th, 2017. However, it was actually captured in Baoding, China in April 2017. This example indicates a critical need

\hspace{1cm}Figure 1: The photo was captured in Baoding China in April 2017, but it was misused for a news event happened in Xuzhou China on June 15, 2017.

\hspace{1cm}http://www.twoeggz.com/news/1949256.html
for computer-aided, automated photo verification techniques that are capable of validating the capture locations (i.e., GPS coordinates) and times of photos to prevent photo misuse.

Validating the capture time and the location of a photo is a promising yet challenging task. Although most photos taken by modern cameras have timestamps and GPS information embedded in its metadata, simply relying on the metadata to determine when and where the photo has been taken is unreliable as the metadata can be easily altered by powerful digital media editing tools [15], [16]. Another possible way is to recognize the landmarks in photos and use the landmarks to validate the place where the photo has been taken. However, most image scenes, such as public lawns, parking lots, beaches and road sides, tend to be similar in many different places and, hence, this is also infeasible. Furthermore, objects such as clothing and colors of trees can imply the time when the photo has been captured, but they can only reveal a relatively long time span (e.g., a T-shirt is suitable from April through October in many places), and thus cannot be used to infer a precise or relatively short capture time of the photo.

Our observation is that shadows cast by the sun vary over time and location; that is, the shadow’s orientation and the length ratio between the vertical object and its shadow are directly affected by the position of the sun (i.e., the orientation and height), which changes with time. Moreover, even at the same time, the shadow’s orientation and the length ratio can still be various in two different cities, as the sun position also depends on the GPS coordinates. Therefore, the shadows of vertical objects are determined by the sun position, which is affected by the time and location.

Inspired by the observation, we propose to use shadows that appear often in outdoor images to verify the capture times and locations of images. Specifically, we first use the shadows of vertical objects in an image to estimate the sun position. We then compute another estimation of the sun position using the capture time and location of the image, which are obtained from the image metadata or claimed by the photo’s user. Note that we assume an adversary, if wants to misuse a photo, tends to modify the time and location in the metadata to align the metadata with those she claims. Finally, if the difference between the two estimations of the sun position is above the set threshold, we determine that the photo is misused.

While it is easy to measure the length ratio of a vertical object and its shadow in real world (then the length ratio can be used to compute the sun position), it is difficult to obtain such information from a 2D image. The reason is that the projection from a 3D scene to a 2D image is a non-linear transformation; as a result, the length ratio measured on the image could be significantly different from the true value in real world. In addition, the length ratio measured on the image depends not only on the scene in real world but also on the camera’s intrinsic parameters and viewing angle. Determining the shadow’s orientation from a 2D image is also challenging due to a lack of the third dimension information. Although single view reconstruction has been extensively studied in the field of computer vision, there is no generalized way to recover the relative positions of objects from a single image. Lalonde et al. analyze the distributions of natural illumination from images, and estimate the sun position based on a proposed probability model [17], [18]; however, the results show that the error is larger than 22.5° for half of the testing photos when predicting the sun’s position. Junejo et al. propose an approach that uses two vertical objects’ shadows in the image to estimate the sun position [19]; however, sometimes only one vertical object may exist in images.

By exploring the camera model and the geometry relations among the camera, the objects and their shadows, we devise new algorithms to estimate the sun position based on only one vertical object and its shadow. Specifically, we first investigate how a camera object pointing in 3D world is projected onto a 2D image, and then analyze the geometry constraints between the interested object, its shadow and the ground. Based on the relations, we recover the 3D positions of the object and its shadow, which can be used to infer the sun position.

We have developed a prototype, called IMAGEGUARD to verify the Time-Location of outdoor images. We collected around 200 photos across three different countries and 16 cities. We conducted three sets of experiments and analyzed the validation performance under different variables and thresholds. IMAGEGUARD achieves a best performance on detecting the falsifications of the time of a day. Overall, using the distance thresholds of sun position and sun altitude angle, we can correctly recognize 92.5% of all the true samples with an error of 8.5% in identifying false samples. Our experimental results show that IMAGEGUARD can successfully estimate sun position and verify the time-location of outdoor images with high accuracy.

Extension. This paper extends our previous work [20], in the following perspectives.

1. New algorithms (Section IV) are proposed which need only one shadow and the camera can be tilted downwards or upwards, while the previous algorithms need either two shadows or one shadow with the camera held vertically.
2. To examine the new algorithms, the previous dataset is enlarged by adding new images that were captured with cameras tilted at different angles (Section VI-A2). Moreover, a synthetic dataset (Section VI-A1) is added to examine the noise resilience of the new algorithms and analyze how the algorithms perform in different situations (Section VI-B).
3. A detailed discussion on the attacks against our framework is added (Section VII-A).

Scope of Our Paper. We focus on the validation of the time-location consistency of outdoor images to detect image misuse. That is, IMAGEGUARD will provide a Yes/No answer to the question: is the image misused? or, is the time or location the image being taken consistent to that the image being used? As IMAGEGUARD uses shadows in an image to verify the time and location, the outdoor image should contain at least one shadow. Moreover, we assume the adversary does not tamper with the content of the image (or at least the shadow part of the image); if the image has been modified, many existing techniques can be adopted to detect the fake image [9]–[11], [13], [14], which is out of the scope of our work.

Contributions. We make the following contributions.
The position of the sun in the sky is defined by an azimuth angle and an altitude angle. An azimuth angle describes the direction of the sun, and an altitude angle defines the height of the sun [21]. As shown in Figure 2, the sun azimuth angle $A$ is measured clockwise in the horizontal plane, from the north to the direction of the sun. It varies from 0° (north) through 90° (east), 180° (south), 270° (west), and up to 360° (north again). The altitude angle $h$ is measured from the horizontal to the sun and it thus ranges from $-90^\circ$ (at the nadir) through $0^\circ$ (on the horizon), up to $90^\circ$ (at the zenith). For instance, when the sun crosses the meridian, its azimuth is $180^\circ$ and altitude is at its largest value in a day.

**B. Motions of the Sun**

Observed from any location on the earth, the sun moves continuously across the sky throughout days and years. The position of the sun can be completely different observed at different time and places.

**Daily Path.** Because of the earth’s daily rotation, the sun appears to move along with the celestial sphere every day. It makes a 360° journey around the celestial sphere every 24 hours. Figure 3a shows three of the sun’s daily paths viewed on the earth. To an observer on the earth, the sun rises somewhere along the eastern horizon, and goes up to the highest point (zenith) around the noon, then goes down until it sets along the western horizon. Accordingly, the cast shadows of any objects move oppositely from somewhere along the west to somewhere along the east. Their lengths vary with the sun’s altitude angle. The higher the sun is above the horizon, the shorter the shadow is. Thus, the shadow that a camera takes at different times of the day will be different.

**Yearly Path.** The sun’s daily path across the sky also changes throughout the year. The reason is that the earth does not rotate on a stationary axis and the tilt in the axis varies each day with respect to the earth’s orbit plane. Observing on the earth, the sun looks higher in the summer than in the winter at the same time in the day. As shown in Figure 3a the sun follows different circles at different days in one year: most northerly on June 21st and most southerly on December 21st. The sun’s motion along the north-south axis over a year is known as the declination of the sun, denoted by $\delta$. Thus, the sun position inferred from photos taken at the same location and time but different days in a year will be different due to the sun’s declination.

**Different Latitudes.** As the sun travels across the sky, the observed altitude angle varies according to the latitude of the observer. The further north or south we go from the equator, the lower the sun’s altitude becomes. Figure 3b and Figure 3c show the sun’s altitude angle versus the azimuth angle observed at 25° north latitude and 40° north latitude respectively. The sun’s altitude angle observed at 25° north latitude is higher than the altitude angle observed at 40° north latitude at the same time. Thus, the sun position inferred from photos taken at the same time but different latitudes will be different.
(a) The path of the sun across the sky as observed on various dates in the northern hemisphere.
(b) at 25° north latitude in the U.S.
(c) at 40° north latitude in the U.S.

Figure 3: The same path of the sun observed at two latitudes.

Figure 4: The work flow of the proposed IMAGEGUARD framework.

B. Overview of IMAGEGUARD

Our goal is to validate whether the claimed capture time and location of an image are true.

Basic Idea. Although an attacker can modify the metadata and claim that a photo was taken at time $X$ and location $Y$, she won’t be able to change the “time” and “location” information that is embedded in the content of an image. Motivated by the observation that shadows are determined by the position of the sun varying with time and location, we use shadows that appear often in outdoor images to obtain the unchangeable information associated with time-location. Specifically, on one hand, we utilize the contents in images—vertical objects and resulting shadows—to extract the sun position. On the other hand, we utilize the claimed capture time and location to obtain a second estimation of the sun position. If these two estimations are close enough, we consider the capture time and location to be true with a high probability.

Workflow. Figure 4 shows the workflow of our approach. For convenience of description, we use the term **TL-inferred sun position** to refer to the sun position calculated from claimed capture time and location, and use the term **shadow-inferred sun position** to refer to the sun position estimated from shadows in the image. In this paper, the camera data mainly represents the camera’s focal length and orientation (i.e., the camera’s pitch and yaw). Vertical objects refer to the ones that are perpendicular to the ground plane.

Our framework works as follows. To compute the **TL-inferred sun position**, we collect the claimed capture time and GPS information from the image metadata and then apply astronomical algorithms. To calculate the **shadow-inferred sun position**, we extract the objects and the resulting shadows from the image using Matlab toolbox. We then run our algorithms based on camera projection model to recover the real positions of the objects and their shadows in real world, the sun position is computed according to the spatial geometrical relations between the objects and their shadows. In particular, the **camera data** is needed for estimating the sun’s azimuth angle, while it is not necessary for estimating the sun’s altitude angle.

Once the two estimations of the sun position are obtained, we determine the differences between these two estimations from three aspects and compare them with the selected thresholds. If the differences are smaller than the corresponding thresholds, we consider these two estimations are close enough indicating the same sun position, and hence the claimed capture time and location are true. Otherwise, they are considered to be falsified. Note that **capture time** in this paper denotes the date of year and the time of day unless otherwise indicated.

Assumption. Without loss of generality, we assume that the camera has zero skew and square pixels (i.e., unit aspect ratio), and it has zero roll angle, which corresponds to the typical situation where the camera looks at the front without tilting to left or right. Note that the camera is free to be tilted upwards or downwards. We further assume that the ground where the interested shadows rest on is approximated to level. Of course, at least one vertical object and its shadow have to be visible in the image. The objects can be human beings, road signs, lampposts, tree trunks and so on.
IV. SHADOW-INFERRED SUN POSITION

This section first provides an introduction to camera projection model, based on which we design the algorithms for estimating sun altitude and azimuth angles from one shadow in the image.

Throughout this section, we use the following notation conventions. A 3D point is denoted by a capital and bold letter (\( \mathbf{X} \)) and its projection on the 2D image is denoted by a lowercase and bold letter (\( \mathbf{x} \)). We also define \( \mathbf{X} = [X_i, Y_i, Z_i, 1]^T \) and \( \mathbf{x} = [x_i, y_i, 1]^T \) in homogeneous coordinates. A 3D vector is referenced by \( \mathbf{X}, \mathbf{X}' \). We refer to a vertical object in the world and its projection in the image as italicized letters \( \mathbf{o}_i \) and \( \mathbf{t}_i \) respectively. In addition, capital letters in roman font (\( M \)) denote matrices.

A. Camera Projection Model

Figure 5 illustrates the projection model of a pinhole camera, where the camera \( C \) has coordinate frame \( (X_c, Y_c, Z_c) \), and its center coincides with the origin of the world coordinate frame \( (X_w, Y_w, Z_w) \). We define \( Y_w \) to be the vertical direction and the \( X_wZ_w \) plane to be parallel to the ground plane. We represent camera orientation \( \mathbf{CO} = \{\theta, \phi\} \) using two parameters, where \( \theta \) denotes the camera’s inclination angle (i.e., pitch) with respect to vertical and \( \phi \) denotes the camera’s azimuth angle (i.e., yaw) with respect to North. The angle \( \phi \) indicates the direction the camera is facing (i.e., the image direction). Both angles are included in the image’s metadata.

To express the mapping from the 3D world to a 2D image, we model the projection of a pinhole camera as \( \mathbf{x} = \mathbf{P}\mathbf{X} \) from the point of view of projective geometry. \( \mathbf{P} \) is called the homogeneous camera matrix and written as

\[
\mathbf{P} = \mathbf{K}\mathbf{R}\mathbf{I}[0], \tag{1}
\]

where \( \mathbf{K} \) is the camera intrinsic matrix, \( \mathbf{R} \) relates the camera rotation with respect to the world coordinate frame, \( \mathbf{I} \) is a 3x3 unit matrix, and \( 0 \) is a zero vector and indicates that the camera is centered at the world coordinate frame.

The matrix \( \mathbf{K} \) in Eq. 1 can be given by

\[
\mathbf{K} = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{2}
\]

which assumes that the camera has zero skew, the intersection of the optical axis and the image plane is at the center of the image, and the pixels are square. Such assumptions are true for current camera technologies [23], [24]. In Eq. 2 \( (u_0, v_0) \) denotes the coordinates of the image center, and \( f \) indicates the camera’s focal length which is either included in the image metadata or inferred using algorithms in [19], [25], [26].

To determine the rotation matrix \( \mathbf{R} \), let’s say the camera rotates by \( \theta \) (i.e., the inclination angle) along the axis \( X_w \) w.r.t the world frame. Then we can write \( \mathbf{R} \) as follows:

\[
\mathbf{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{bmatrix}, \tag{3}
\]

where the angle \( \theta \) can be computed by camera inertial sensors. Without loss of generality, we assume that a photographer normally holds the camera with an inclination in the range of \((-45^\circ, 45^\circ)\). This, of course, is consistent with our common habits of photographing. Finally, we have the mapping matrix \( \mathbf{P} \) as follows:

\[
\mathbf{P} = \begin{bmatrix} f & u_0 \sin \theta & u_0 \cos \theta \\ 0 & f \cos \theta + v_0 \sin \theta & v_0 \cos \theta - f \sin \theta \\ 0 & \sin \theta & \cos \theta \end{bmatrix}. \tag{4}
\]

B. Estimating Sun Altitude Angle

To estimate the sun altitude angle, we consider the scenario where only one vertical object and its shadow are visible in the image. Figure 6 depicts how the object \( O_1 \) and its shadow \( S_1 \) are projected onto a 2D image by the camera \( C \) with an inclination angle \( \theta \). The intuition of this algorithm is to recover the positions (i.e., coordinates) of \( \mathbf{X}_1, \mathbf{X}_2 \) and \( \mathbf{X}_3 \) based on the camera model and the resulting projections.

To recover the coordinates of \( \mathbf{X}_1 \) and \( \mathbf{X}_2 \), we explored two constraints. First, considering the fact that \( \mathbf{X}_1 \) and \( \mathbf{X}_2 \) lie on the ground plane, they satisfy \( \mathbf{X}_i^T \mathbf{G} = 0 \) for \( i = 1, 2 \) and \( \mathbf{G} = \)

\[
\begin{align*}
\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{bmatrix}
\end{bmatrix}
\]
Second, these two points in space are mapped to the image according to the camera’s mapping matrix, so \( x_i = P \mathbf{X}_i \). By solving the above equations, we can obtain the coordinates of points \( \mathbf{X}_1 \) and \( \mathbf{X}_2 \). Then we have the vector \( \mathbf{X}_1 \mathbf{X}_2 \).

To determine the coordinates of \( \mathbf{X}_3 \), we also utilize two constraints. First, since the object \( O_1 \) is perpendicular to the ground plane (\( X_oZ_o \)), the point \( \mathbf{X}_3 \) has the same \( X \) and \( Z \) components with \( \mathbf{X}_2 \). Second, the projection of \( \mathbf{X}_3 \) from space onto the image also satisfies \( x_3 = P \mathbf{X}_3 \). Using these two constraints, we can obtain the coordinates of \( \mathbf{X}_3 \). Once the coordinates of \( \mathbf{X}_1 \) and \( \mathbf{X}_3 \) are obtained, we can calculate the vector \( \mathbf{X}_1 \mathbf{X}_3 \).

Finally, the angle between \( \mathbf{X}_1 \mathbf{X}_2 \) and \( \mathbf{X}_1 \mathbf{X}_3 \) is the altitude angle and can be computed as follows:

\[
\theta = \cos^{-1}\frac{(\mathbf{X}_1 \mathbf{X}_3)^T \mathbf{X}_1 \mathbf{X}_2}{\sqrt{(\mathbf{X}_1 \mathbf{X}_3)^T \mathbf{X}_1 \mathbf{X}_3} \sqrt{(\mathbf{X}_1 \mathbf{X}_2)^T \mathbf{X}_1 \mathbf{X}_2}}
\]

(5)

where the intermediate variables are as follows:

\[
\begin{align*}
\begin{cases}
m_a &= (f(x'_1 - x'_2) \sin \theta + (x'_1 y'_2 - x'_2 y'_1) \cos \theta)^2 + f^2(y'_2 - y'_1)^2 \\
m_b &= \cos^2 \theta (y'_1 + f \tan \theta)^2 (y'_2 + f \tan \theta)^2 \\
m_c &= f^2 (y'_2 - y'_1)^2 \\
m_d &= (f \sin \theta + y'_2 \cos \theta)^2 (f \cos \theta - y'_3 \sin \theta)^2
\end{cases}
\]

\( x'_i = x_i - u_0 \) and \( y'_i = y_i - v_0 \) for \( i = 1, 2, 3 \). The altitude angle \( \theta \) should be in the range of \((0^\circ, 90^\circ)\) when the sun is visible.

C. Estimating Sun Azimuth Angle

To estimate the sun’s azimuth angle \( A \) from one shadow in the image, we design the following algorithm. The scenario is illustrated in Figure 6. In particular, the point \( \mathbf{X}_3 \) is not necessary to be visible for estimating the azimuth angle. The true north \( N \) is set to be the reference direction in our algorithm. The unit vector \( \hat{T} = (0, 0, 1) \) has the same direction as the image direction.

The sun azimuth angle \( A \) equals the angle measured clockwise around point \( \mathbf{X}_1 \) from due north to the shadow. We calculate \( A \) as follows:

\[
A = \angle(N, \hat{T}) + \angle(\hat{T}, \mathbf{X}_1 \mathbf{X}_2),
\]

(6)

where \( \angle(\hat{T}, \mathbf{X}_1 \mathbf{X}_2) \) denotes the angle measured clockwise from \( \hat{T} \) to \( \mathbf{X}_1 \mathbf{X}_2 \), and \( \angle(N, \hat{T}) \) is the angle measured clockwise from \( N \) to \( \hat{T} \), which equals the orientation of the image direction. \( \angle(\hat{T}, \mathbf{X}_1 \mathbf{X}_2) \) is the only unknown variable in Eq. 6.

We represent the angle between \( \hat{T} \) and \( \mathbf{X}_1 \mathbf{X}_2 \) by \( \alpha \). If \( \angle(\hat{T}, \mathbf{X}_1 \mathbf{X}_2) \leq 180^\circ \), it equals \( \alpha \). Otherwise, it equals \((360^\circ - \alpha)\). Using the result of \( \mathbf{X}_1 \mathbf{X}_2 \) in last subsection, we calculated the angle \( \alpha \) as:

\[
\alpha = \cos^{-1}\frac{\mathbf{X}_1 \mathbf{X}_2^T}{\sqrt{\mathbf{X}_1 \mathbf{X}_2^T \mathbf{X}_1 \mathbf{X}_2}},
\]

(7)

where

\[
\begin{align*}
m'_a &= f(y'_2 - y'_1) \\
m'_b &= f(x'_1 - x'_2) \sin \theta + (x'_1 y'_2 - x'_2 y'_1) \cos \theta
\end{align*}
\]

V. Metadata-inferred Sun Position and Validation

In this section, we describe the process to validate the consistency of a photo’s capture time and location. The key idea is the following: we calculate the sun position using the capture time and location in the metadata of images. If the capture time and location are true, the sun position will match the one we estimated from shadows.

A. Metadata-inferred Sun Position

As mentioned in Section [11], the position of the sun depends on the time of day, the date and the location of the observer. Its movement across the sky obeys the rules that have been studied in astronomy. In this section, we discuss the astronomical algorithms that are used to calculate TL-inferred sun position, given the time and location.

We refer the time of day as the local time based on the standard time offsets of Coordinated Universal Time (UTC). However, the local standard time doesn’t provide an intuitive connection with the sun position. In astronomy, the solar time is often used to discuss the sun position. It works because the sun finishes a 360° rotation around the celestial sphere every 24 hours. The completed journey is divided into 24 hour, and one solar hour means that the sun travels a 15° arc [27]. The instant when the sun is due south in the sky or the shadow points to exactly north is called solar noon, which is 12:00 for solar time. Every 15° arc the sun travels, one hour is added to 12:00 under the 24-hour clock system, and the angle distance that the sun passes on the celestial sphere is defined as the hour angle \( H \) [27]. It is measured from the sun’s solar noon position, and ranges from 0° to +180° westwards and from 0° to −180° eastwards. The conversion between the local standard time \( t_l \) to the solar time \( t_s \) is as follows [27], [28]:

\[
t_s = t_l + ET + \frac{4 \min}{\deg}(\lambda_{std} - \lambda_l),
\]

(8)

where \( \lambda_l \) denotes the local longitude, and \( \lambda_{std} \) is the local longitude of standard time meridian, and \( ET \) stands for the equation of time, which describes the difference of the true solar time and the mean solar time [28]. The sun’s hour angle is calculated as follows:

\[
H = 15^\circ(t_s - 12). 
\]

(9)
Using the observer’s local horizon as a reference plane, the azimuth and altitude angles of the sun can be calculated as follows [21]:

\[
\tan(A) = \frac{\sin H}{\sin \varphi \cos H - \cos \varphi \tan \delta},
\]

\[
\sin(h) = \sin \delta \sin \varphi + \cos \varphi \cos \delta \cos H,
\]

where \(\varphi\) is the latitude of the observer’s location, and \(\delta\) is the sun’s declination angle and it can be calculated as below [21], [29]:

\[
\delta = -23.44^\circ \cos \left(\frac{360^\circ (N + 10)}{365^\circ}\right),
\]

where \(N\) is the number of days since January 1st. Note that the azimuth angle \(A\) calculated in Eq. 10 uses south as a reference. We can derive the azimuth angle according to its definition in Section II.

B. Consistency Validation

Once obtaining the shadow-inferred sun position and TL-inferred sun position, we check the difference between these two estimations by comparing their altitude angles and azimuth angles respectively. However, since there exists random and systemic errors in the shadow-inferred sun position, the estimation may not equal the “true” sun position. Thus, we have to select a threshold that is large enough to tolerate the errors yet small enough to detect the inconsistency between the shadow-inferred sun position and TL-inferred sun position. Intuitively, the closer these two sun positions are to each other, the more likely the capture time and location are true.

We define the altitude angles of shadow-inferred sun position and TL-inferred sun position to be \(h_s\) and \(h_m\) respectively, and the corresponding azimuth angles to be \(A_s\) and \(A_m\). Then the distance of the two altitude angles is \(d_h = |h_s - h_m|\), and the distance of the two azimuth angles is computed as \(d_A = |A_s - A_m|\). The likelihood of the consistency is inversely proportional to \(d_h\) and \(d_A\). However, the effects on \(d_h\) and \(d_A\) caused by fake capture time and/or location are different. For example, modifying the capture time from 12:00 p.m. to 13:00 p.m. may lead to 10° in \(d_A\) but only 2° in \(d_h\). So two different thresholds for \(d_h\) and \(d_A\) have to be selected. The capture time and location are considered to be true only when both \(d_h\) and \(d_A\) are within the thresholds. Besides, the sun position can be described by a pair of azimuth angle and altitude angle: \((A, h)\). We can also use the sun position distance that is computed as \(d_p = \sqrt{d_A^2 + d_h^2}\) to distinguish the two estimations of the sun position. Our goal is to choose appropriate variables and thresholds that can increase the probability of correct validation for inconsistent images and decrease the probability of false validation for consistent images. Section VI details the selection of thresholds in the validation experiment.

VI. EVALUATION

This section presents the results of our experiments. To evaluate the performance of the shadow-inferred sun position algorithms, we built a computer synthetic dataset and conducted extensive tests with it. To validate the effectiveness of the framework IMAGEGUARD, we gathered 200 photos in China, the U.S. and Japan in the span of about one year, and examined whether we can detect the modifications of capture time, date and location.

A. Datasets

We build two datasets for evaluating our algorithms and framework.

1) Dataset I: synthetic photos: We choose the city of New York, the 21st day of March 2017, and the time period from 9:00 a.m. to 4:00 p.m. to simulate the sun positions. We generate a photo of a synthetic object and its shadow every 5 minutes using the following simulated camera model.

Camera model. As shown in Figure 6, we build a pinhole camera model using the open source package: OpenCV [2]. The simulated camera has the following parameters: the focal length \(f = 3351.6\) pixels (equivalent to 3.99mm based on a 4.8mm wide CCD), the principle point \((u_0, v_0) = (2016, 1512)\) pixels. As expected with most modern CCD cameras, it has zero skewness and unit aspect ratio. We also set the camera height above the ground plane to be 1.6 meters (1m = 840000 pixels) which is close to the real height at which people hold a camera to take photos.

Scene setting. We place one vertical object on the ground plane along the axis of \(Z_w\) (refer to Figure 6). The object has a height of 1 meter and a distance of \(m\) meters from the camera. Mathematically, the object’s footprint \(X_2\) is at \((0, 1, m)\) meters for \(m = 5, 10\) in the world coordinate system. Under each scenario, we generate 84 photos of the synthetic object and its shadow.

2) Dataset II: real photos: This dataset consists of 200 photos taken by 10 different smartphones, including iPhone 5s, 6, 6 plus, 6s, 6s plus and 7. They were captured at 16 cities around China, the U.S. and Japan since September 2016. Among the 200 photos, 79 were taken in China, 7 were captured in Japan, and the rest were collected in the U.S. Each photo encloses the metadata that includes the real capture time and location. In addition, 76 out of the 200 photos were taken by cameras tilted forwards or backwards with different angles. Our dataset mainly contains three types of vertical objects: standing people, poles (e.g. road signs, lampposts) and tree trunks. These objects are frequently seen in the outdoor photos and are almost vertical to the ground.

We refer to the true metadata of the 200 photos as the positive samples. We generate the attack data by falsifying the metadata of the 200 photos and refer to the attack metadata as the negative samples. Note that multiple types of metadata may result in the same effect. For instance, modifying longitude one degree more to the west has the same effect on the sun position as changing the local time forward by four minutes. Thus, falsifying either longitude or the local time is equivalent. To simplify the analysis yet without loss of generality, we focus on three types of attacks that modify the following metadata:

- Falsified time of day, and true date and location.
- Falsified date, and true time and location.

2http://opencv-python-tutorals.readthedocs.io/en/latest/index.html
(a) Absolute errors in sun altitude angles.

(b) Absolute errors in sun azimuth angles.

(c) Absolute errors in sun altitude angles.

(d) Absolute errors in sun azimuth angles.

Figure 7: Performances on synthetic data containing Gaussian noise: Figure 7a and 7b depict the absolute errors in sun positions varying with the camera inclination angles, while Figure 7c and 7d show the absolute errors in sun positions varying with the distances of the object and camera.

- Falsified latitude of location, and true time and date.
  The “fake” times of day are randomly generated in the range from 8:00 a.m. to 17:00 p.m. when the sun is likely to be seen. The “fake” dates are randomly generated from the range within one year. The “fake” latitudes of location are randomly generated in the range of 25° and 50° of the Northern Hemisphere where most parts of the U.S., Japan and China locate. We have 200 negative samples for each type of attack metadata.

B. Evaluation of the Shadow-Inferred Sun Position Algorithms

Using Dataset I, we mainly examine the accuracy of the shadow-inferred sun position algorithms and their resilience to noise. We also analyze the impact of varying the camera inclination angle and the distance between the camera and object on estimating sun positions.

When applying our algorithms to the scene setting described in Section VI-A1, we add Gaussian noise to the interested object and shadow points (i.e., \(X_1, X_2, X_3\)). We refer to variance/covariance as the noise level. For each noise level, we performed 200 independent trails on each photo (84 photos in total). Then, we compute the absolute error with respect to the ground truth for each trail. Finally, we calculate the average absolute error of the 84 photos for each noise level.

To plot Figure 7a and 7b, we set the object to be 5 meters away from the camera and vary the camera inclination angle (\(\theta = 0^\circ, 10^\circ, 20^\circ\)); To plot Figure 7c and 7d, we set the camera inclination angle to be 0° and vary the distance (5, 10 meters) between the camera and the object.

Noise resilience. Figure 7 indicates that the absolute error increases almost linearly as the noise level grows. We obtain the maximum absolute error of 1.91° in sun altitude angles (Figure 7c) and 2.44° in sun azimuth angles (Figure 7d).
when the noise level reaches 4 pixels and the object is 10 meters away from the camera. The absolute errors in sun altitude angles (Figure 7c) are slightly smaller than the errors in sun azimuth angles (Figure 7b, 7d) under the same setting of parameters. In short, Figure 7 demonstrates that both algorithms can achieve good accuracy even in the high noise case (4 pixels), and the algorithm for estimating sun altitude angle provides a slightly stronger resilience to noise than the algorithm for estimating sun azimuth angle.

**Impact of different camera inclination angles.** Figure 7a and 7b show that varying the camera inclination angle has little effect on the noise resilience of both algorithms. As the inclination angle increases, the absolute error in either the sun altitude angle or the azimuth angle only increases a little bit at the same noise level. Our algorithms are robust to the camera inclinations.

**Impact of different distances from the camera to objects.** Figure 7c and 7d denote that increasing the distance between the camera and object has a negative impact on the noise resilience of both algorithms. As the distance increases, the absolute error grows. The error can be large if the noise level is very high and the distance is long. Thus, our algorithms can achieve better performance when the distance of the camera and the interested object is short (i.e., <10 m). Compared to the camera inclination angle (Figure 7a, 7b), our algorithms are more sensitive to the distance (Figure 7c, 7d).

In summary, our algorithms in Section IV are able to infer sun position with high accuracy and robust to noise.

### C. Evaluation of IMAGEGUARD

To evaluate the performance of IMAGEGUARD and to understand threshold selection, we conducted a set of experiments.

1) **Metric:** We use ROC curves to evaluate the performance of IMAGEGUARD by varying thresholds for our system. An ROC curve represents Receiver Operating Characteristic curve and is created by plotting true positive rate (TPR) against false positive rate (FPR), as the threshold varies. The true positive rate and false positive rate are defined as below.

\[
TPR = \frac{\# \text{ of true positives}}{\# \text{ of (true positives + false negatives)}}
\]
\[
FPR = \frac{\# \text{ of false positives}}{\# \text{ of (true negatives + false positives)}}
\]

where a true positive denotes the result that a positive sample is correctly identified as such, and a false positive is the one that a negative sample is identified as a positive sample by mistake.

The point (0, 1) on the ROC curve denotes 0 FPR and 100% TPR, which indicates an ideal system that can correctly identify all genuine photos and reject all falsified photos. In our experiment, we select the optimal threshold as the one that yields the minimum distance from the corresponding point on the ROC curve to the ideal point (0, 1). Another indicator that we use to evaluate the average performance of the validation is the area under the ROC curve (AUC). The closer it is to 1, the better the average performance is.

2) **Performance and Thresholds:** Based on the framework IMAGEGUARD, we performed consistency validation using the three types of falsified metadata in Dataset II. To understand how the altitude angle and azimuth angle influence the performance of the validation, we examine three distances separately: the distance of the altitude angles \(d_h\), the distance of the azimuth angles \(d_A\), and the distance of the sun positions \(d_p\). Here, the sun position is defined to be \((A, h)\), in which \(A\) refers to the azimuth angle and \(h\) refers to the altitude angle. To decide the best distance variable which can yield the maximum AUC and the optimal threshold of the variable, we analyze the ROC curves that are plotted by varying the threshold of each type of distance.

The results are presented in the set of ROC curves shown in Figure 8 and Figure 9. Each ROC curve with distinct color is plotted by varying the threshold of one type of the three distances. “TH-\(d_h\)” and “TH-\(d_A\)” denote varying the threshold of the altitude angle distance \(d_h\) and the azimuth angle distance \(d_A\) respectively. “TH-\(d_p\)” denotes varying the threshold of the sun position distance \(d_p\). For each type of attack metadata, we repeat the random generating of the 200 negative samples 5 times. And each false positive rate on the ROC curve is averaged over these repeated attack metadata.

Figure 8a indicates that the detection based on \(d_p\) slightly outperforms the one based on \(d_h\). However, the \(d_h\) based detection achieves better performance in detecting all the other types of attacks as shown in Figure 8b and Figure 8c especially in detecting falsified latitude. The result implies that \(d_h\) is more important in distinguishing different positions of the sun compared to \(d_A\) in general. Such a conclusion confirms with the result reported in Section VI-B, i.e., the average estimation error of the altitude angles is smaller than that of the azimuth angles. If only \(d_h\) is used for consistency validation, Figure 9 guides us to choose the optimal threshold of \(d_h\) to be 3.2° and it achieves combined (TPR, FPR) values of (89%, 23%) for all attacks, which means that 89% of positive samples can be correctly validated but 23% of negative samples will be mistakenly identified.

In addition, Figure 8 shows that the \(d_p\) based detection achieves the best performance in detecting falsified time of day, and has almost the same performance as the \(d_h\) based detection in detecting the other types of attacks. Once we only use \(d_p\) for consistency validation, Figure 9 leads us to choose the optimal threshold of \(d_p\) to be 9.4°, which achieves combined (TPR, FPR) values of (94%, 18.4%) for all attacks.

To improve the performance further, we examine both the \(d_p\) and \(d_h\) to validate the consistency of time and location. That is, a sample has to satisfy both the thresholds of \(d_h\) and \(d_p\) to be accepted by IMAGEGUARD. Plotting the ROC curves and finding the global optimal thresholds by varying two thresholds can be tricky. Thus, we chose the local optimal threshold for one variable and varied the other threshold to plot the ROC curve. This approach may not generate the global optimal thresholds for the two variables, but it strikes a balance between the optimum and the computational cost. We chose the threshold of \(d_p\) to be 9.4° and varied the threshold of \(d_h\). The resulting curve illustrates an improved performance than the one of using a single threshold as shown in Figure
Note that we cannot plot an integral ROC curve when the threshold of $d_p$ is fixed since the highest true positive rate will be decided by the fixed threshold, which is 94%. The curve “TH-$d_p$,$d_h$” in Fig 9 indicates that choosing the optimal threshold of $d_h$ to be 5° can correctly identify 92.5% positive samples, but cannot identify 8.5% of negative samples.

VII. DISCUSSIONS

In this section, we discuss the attacks against our framework and analyze two factors that may influence the estimation of sun position.

A. Attacks against IMAGEGUARD

Based on the evaluation results, we analyze the robustness of the framework IMAGEGUARD when detecting the falsifications of the time of day, the date and the location. IMAGEGUARD can confidently identify the attacks that cause violations of one of the thresholds of the altitude angle distance and the sun position distance. We provide the following examples to help have an intuitive understanding of how much the time and location change may result in an exceeding of the chosen thresholds.

Table I: Time modifications that can be detected.

| Date       | $\Delta T$ (minutes) |
|------------|----------------------|
| Dec. 21st  | $\geq 40$            |
| Mar. 21st  | $\geq 29$            |
| June 21st  | $\geq 27$            |

Table II: Date modifications that can be detected.

| Time       | $\Delta D$ (days) |
|------------|-------------------|
| Dec. 21st  | $\geq 33$         |
| Mar. 21st  | $\geq 13$         |
| June 21st  | $\geq 40$         |

First, we assume the location is New York city and it is not modified. We choose three representative dates (e.g., the 21st of December, March and June) when the sun looks lowest, intermediate and highest, respectively, to check how much modification of the time can be detected by IMAGEGUARD. We choose the time of 9:00 a.m. and 12:00 p.m. as the baselines. Table I shows the results. The sun appears to move fastest on June 21st and a modification more than 16 minutes from 12:00 p.m. will exceed the sun position threshold, while the sun moves slowest on December 21st and a modification greater than 40 minutes can be recognized.

Second, we assume both the time of the day and the location are true. We vary the date of the year for validations. We use the 21st of December, March and June as our baselines. As shown in Table II, IMAGEGUARD can correctly detect a change greater than 13 days from March 21st at 12:00 p.m. and at worst a 48-day change from June 21st at 9:00 a.m. The reason that the difference exists is because the sun’s daily path does not change at a constant rate throughout the year.

Third, we fix the time and the date but modify the location. We find that IMAGEGUARD is able to detect location modifications that are larger than 400 miles away from New York. There is another scenario: an attacker knows how to modify both the time and location of a photo such that the altitude angle and the sun position are within the thresholds and the modification can fool IMAGEGUARD. Luckily, the motivation of falsifying the metadata of a photo is to use it for a chosen event and the attacker may not be guaranteed to find such a combination.

Referring to the fake photo given in Section I, we show
how \textsc{ImageGuard} detects that it was not taken at the claimed dates and locations. For the image in Figure 1, although we do not have the required metadata information (e.g., the camera orientation) to estimate the azimuth angle as well as the exact sun position, we can estimate the altitude angle range. Given the claimed time (i.e., 16:50 on June 15, 2017) and the location (i.e., Xuzhou, China), we can calculate the sun altitude angle to be 29.1° based on astronomical algorithms. On the other hand, based on the image, we estimate the focal length to be 420 ± 20 pixels using algorithms in [19]. By assuming the camera inclination angle to be between −10° and 10°, we obtain the altitude angle to be between 52° and 58°. The distance between the two estimates is far beyond the threshold 5° in our experiments. Thus, we conclude that the date and location of this image were spoofed.

B. Influence of Ground Slope

One source of uncertainty concerns the possibility of ground slope which is associated with the accuracy in locating the “real” shadow’s position. If the ground where the shadow located is not truely flat and has a slope angle of ΔG with respect to the horizontal plane, ΔG will propagate as the altitude angle is estimated. It also causes a slight error in the estimated azimuth angle. But neither one of the errors will be larger than ΔG. On the other hand, if our goal is to verify the capture time of a photo (i.e., the capture location has been identified through other methods), we can measure the slope angle by visiting the place and use it to calibrate the estimated sun position. Based on the preciser estimation, our framework will be more confident in validating the capture time of the image.

C. Influence of Non-vertical Object

Possible tilt angle of the “vertical” object is another source of uncertainty. Although many well-structured man-made objects (e.g., road signs, lampposts, walls) usually include strong vertical edges with respect to the ground, some certain errors may exist. Upright standing human beings in images may also not be perfectly vertical to the ground. The tilt angle will mainly cause error in the estimated sun altitude angle and have little effect on the shadow orientation. The resulting error will not be larger than the tilt angle itself. The maximum tilt angle is 90° when the object tilt on the ground, and it causes an error equal to the true sun altitude angle which is less than 90°. On the other hand, if we can locate where the image was taken, the estimated altitude angle can be calibrated by measuring the tilt angle of the object. Then we are able to improve the confidence of our framework in validating the capture time.

VIII. RELATED WORK

In the field of computer vision, there have been many studies aimed at estimating the geolocation or the time of cameras and images. Studies have demonstrated that shadow trajectories inferred from multiple outdoor images can be used to determine the geolocation of stationary cameras [19], [24]. Sandnes [35] uses the relative lengths of objects and their shadows in images to estimate the sun altitude angle and further estimates the geolocation of the camera. Jacobs et al. [36] analyze the correlations of camera images with satellite images and other images with known locations to determine where these camera images were taken.

Researchers have also used sun position to estimate the capture time of images. Based on Lalonde et al.’s model [17], [18] that calculates the sun position by combining the cues in the image such as the sky and the shadows on the ground, Tsai et al. [37] and Kakar et al. [16] utilize the position of the sun to estimate the capture time for geo-tagged images. Other studies extract fluctuations of the Electric Network Frequency (ENF) signal from fluorescent lighting and verify the feasibility of using the ENF signal as a natural timestamp for videos in an indoor environment [38], [39]. Deep learning algorithms have recently been used to predict the contextual information of outdoor images. Volokitin et al. [40] apply convolutional neural networks for the predictions of temperature and time of the year.

Chen et al.’s [41] work is probably most related to ours. The authors train machine learning models to predict the sun altitude angle, temperature, humidity and weather condition associated with outdoor images, and then combine all the inferred information to detect tampering in the image metadata. Their results show that the absolute error in predicting sun altitude angle is larger than 10° for 45% of the testing photos. Our method can compute both sun altitude angle and azimuth angle without requiring a large amount of photos for training models. Moreover, our algorithms can achieve lower error (2.98°) in estimating sun altitude angle.

IX. CONCLUSION

We presented a new framework \textsc{ImageGuard} which uses two estimations of sun position—shadow-inferred sun position and TL-inferred sun position—to check whether the capture time and location of an outdoor image are true. Our framework exploits the relationship between the sun position in the sky and the time and location of an observer. We designed algorithms to obtain shadow-inferred sun position using vertical objects and their shadows in the image. Our experiments show that the algorithms can estimate the sun position from shadows in the image with satisfactory accuracy. The evaluation results demonstrate that \textsc{ImageGuard} can achieve combined (TPR, FPR) values of (92.5%, 8.5%) for the consistency validation. We believe that our results illustrate the potential of using sun position to validate the consistency of the capture time and location. Our work raises an open question that whether other image contents can be leveraged for validating the consistency of image’s contextual information.

APPENDIX A

PROOF OF EQUATION 5

To calculate the coordinates of $X_1$ and $X_2$, we have the following equations:

$$X_i^T G = 0,$$  \hfill (13a)

$$x_i = P X_i,$$  \hfill (13b)
where \( i = 1, 2 \) and \( \mathbf{G} = [0, 1, 0, h_c]^T \). By solving the above equations, we can obtain the coordinates of points \( \mathbf{X}_1 \) and \( \mathbf{X}_2 \). Then we have the vector \( \mathbf{X}_1 \mathbf{X}_2^T \). We define \( \mathbf{x}_i \) to be \( [x_i, y_i, 1] \) under the image coordinate frame, and \( \mathbf{X}_i \) to be \( [X_i, Y_i, Z_i, 1] \) under the world coordinate frame.

\[
\mathbf{X}_i = \begin{bmatrix}
    h_c \cos \theta \frac{x'_i}{|y'_i + f \tan \theta|} \\
    -h_c \\
    h_c \frac{y'_i}{|y'_i + f \tan \theta|} \\
    1
\end{bmatrix}
\]

(14)

Note that we divide each resulting coordinate by \( h_c \) in Eq. [15].

\[
\mathbf{X}_3 = \begin{bmatrix}
    \frac{x_2}{\cos \theta - y'_2 \sin \theta} \\
    \frac{Z_2}{\cos \theta - y'_2 \sin \theta} \\
    1
\end{bmatrix}
\]

(16)

Using the coordinates of \( \mathbf{X}_1 \) and \( \mathbf{X}_3 \), we can calculate the vector

\[
\begin{bmatrix}
    f(x'_1 - x'_3) \tan \theta + (x'_1 y'_3 - x'_3 y'_1) \\
    f(y'_1 \tan \theta) \\
    f(y'_3 - y'_1) \\
    f(x'_1 y'_3 - x'_3 y'_1)
\end{bmatrix}
\]

(17)

REFERENCES

[1] M. Imran, S. M. Elbassuoni, C. Castillo, F. Diaz, and P. Meier, “Extracting information nuggets from disaster-related messages in social media,” Proc. of ISCRAM, Baden-Baden, Germany, 2013.

[2] A. D. Miller and W. K. Edwards, “Give and take: A study of consumer photo-sharing culture and practice,” in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ser. CHI ’07. New York, NY, USA: ACM, 2007, pp. 347–356.

[3] A. Malik, A. Dhir, and M. Nieminen, “Uses and gratifications of digital photo sharing on facebook,” Telematics and Informatics, vol. 33, no. 1, pp. 129 – 138, 2016.

[4] C. Boidioud, S. Papadopoulos, Y. Kompatsiaris, S. Schifferes, and N. Newman, “Challenges of computational verification in social multimedia,” in Proceedings of the 23rd International Conference on World Wide Web, ser. WWW ’14 Companion. New York, NY, USA: ACM, 2014, pp. 743–748.

[5] M. Kasra, C. Shen, and J. O’Brien, “Seeing is believing: How people fail to identify fake images on the web,” SSRN, Oct 2016. [Online]. Available: https://ssrn.com/abstract=2951561

[6] C. Pasquini, C. Brunetta, A. F. Vinc, W. Comtet, and G. Boato, “Towards the verification of image integrity in online news,” in 2015 IEEE International Conference on Multimedia Expo Workshops (ICMEW), June 2015, pp. 1–6.

[7] A. Gupta, H. Lamba, P. Kumargun, and A. Joshi, “Faking sandy: characterizing and identifying fake images on twitter during hurricane sandy,” in ACM WWW ’13. ACM, 2013, pp. 729–736.

[8] Fema now has a rumor control section for misinformation. [Online]. Available: https://twitter.com/fema/status/264800761119113216

[9] J. Fridrich, D. Soukal, and J. Lukas, “Detection of copy move forgery in digital images,” in Digital Forensic Research Workshop, Aug. 2003.

[10] S. Bayram, H. T. Sencar, and N. Memon, “An efficient and robust method for detecting copy-move forgery,” in IEEE ICASSP’09. IEEE, 2009, pp. 1053–1056.

[11] X. Pan and S. Lyu, “Detecting image region duplication using sift features.” in IEEE ICASSP’10. IEEE, 2010, pp. 1706–1709.

[12] A. C. Popescu and H. Farid, “Exposing digital forgeries by detecting traces of resampling,” IEEE Transactions on Signal Processing, vol. 53, no. 2, pp. 758–767, Feb 2005.

[13] T. Carvalho, H. Farid, and E. Kee, “Exposing photo manipulation from user-guided 3d lighting analysis,” in SPIE/IS&T Electronic Imaging, International Society for Optics and Photonics, 2015, pp. 940-902–940-902.

[14] E. Kee, J. F. O’Brien, and H. Farid, “Exposing photo manipulation from shading and shadows,” ACM Trans. Graph., vol. 33, no. 5, pp. 165:1–165:21, Sep. 2014.

[15] E. Kee, M. K. Johnson, and H. Farid, “Digital image authentication from jpeg headers,” IEEE Transactions on Information Forensics and Security, vol. 6, no. 3, pp. 1066–1075, Sept 2011.

[16] P. Kakar and N. Sudha, “Verifying temporal data in geotagged images via sun azimuth estimation,” IEEE Trans. Information Forensics and Security, vol. 7, no. 3, pp. 1029–1039, 2012.

[17] J. F. Lalonde, A. A. Efros, and S. G. Narasimhan, “Estimating natural illumination from a single outdoor image,” in 2009 IEEE 12th International Conference on Computer Vision, Sept 2009, pp. 183–190.

[18] J. F. Lalonde, S. G. Narasimhan, and A. A. Efros, “What do the sun and the sky tell us about the camera?” International Journal of Computer Vision, vol. 88, no. 1, pp. 24–51, May 2010.

[19] I. Junejo and H. Foroosh, “Estimating geo-temporal location of stationary cameras using shadow trajectories,” in ECCV’08. Marseille, France: Springer, October 2008, pp. 318–331.

[20] X. Li, W. Xu, S. Wang, and X. Qu, “Are you lying: Validating the time-location of outdoor images,” in Applied Cryptography and Network Security. Cham: Springer International Publishing, 2017, pp. 103–123.

[21] D. Savio, Sundials: Design, Construction, and Use. Chichester, UK: Praxis Publishing, 2009.

[22] P. Harvey. ExifTool. [Online]. Available: http://www.sno.phy.queensu.ca/~phil/exiftool/

[23] R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision, 2nd ed. Cambridge University Press, 2004.

[24] L. Wu and X. Cao, “Geo-location estimation from two shadow trajectories,” in IEEE CVPR’10, 2010, pp. 585–590.

[25] Z. Zhang, “A flexible new technique for camera calibration,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, pp. 1330–1344, December 2000, mSR-TR-98-71, Updated March 25, 1999. [Online]. Available: https://www.microsoft.com/en-us/research/publication/a-flexible-new-technique-for-camera-calibration/

[26] ——, “Camera calibration with one-dimensional objects,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 7, pp. 892–899, July 2004.

[27] H. Karttunen, P. Kroger, H. Oja, M. Poutanen, and K. J. Donner, Fundamental Astronom, 5th ed. Springer-Verlag Berlin Heidelberg, 2007.

[28] K. E. Holbert and D. Srinivasan, “Solar energy calculations,” in Handbook Of Renewable Energy Technology, 2011, pp. 189–204.

[29] M. Iqbal, “An introduction to solar radiation.” Elsevier, 2012.

[30] T. Fawcett, “An introduction to roc analysis;” Pattern recognition letters, vol. 27, no. 8, pp. 861–874, 2006.

[31] J. Tian, C. Qu, W. Xu, and S. Wang, “Kinwrite: Handwriting-based authentication using inkjet;” in NDSS’13, 2013.

[32] A. VAILAYA, A. JAIN, and H. J. ZHANG, “On image classification: City images vs. landscapes,” Pattern Recognition, vol. 31, no. 12, pp. 1921 – 1935, 1998.

[33] Z. Zhang, C. Ding, X. Qiu, and F. Li, “The characteristics of the multipath scattering and the application for geometry extraction in high-resolution sar images,” IEEE Transactions on Geoscience and Remote Sensing, vol. 53, no. 8, pp. 4687–4699, Aug 2015.

[34] H. Lee, E. Shechtman, J. Wang, and S. Lee, “Automatic upright adjustment of photographs with robust camera calibration,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 36, no. 5, pp. 833–844, May 2014.

[35] F. E. Sandnes, “Determining the geographical location of image scenes based on object shadow lengths,” Journal of Signal Processing Systems, vol. 65, no. 1, pp. 35–47, Oct 2011.

[36] N. Jacobsq, S. Satkin, N. Roman, R. Speyer, and R. Pless, “Geolocating stories,” in 2007 IEEE 11th International Conference on Computer Vision, Oct 2007, pp. 1–6.

[37] M. H. Tsai, W.-C. Jou, W.-H. Cheng, M.-C. Hu, I.-C. Shen, T. Lim, K.-L. Hua, A. Ghoneim, M. A. Hossain, and S. C. Hidayati, “Photo sundial: estimating the time of capture in consumer photos,” Neurocomputing, vol. 177, pp. 529–542, 2016.
[38] R. Garg, A. L. Varna, and M. Wu, “Seeing enf: natural time stamp for digital video via optical sensing and signal processing,” in *ACM MM’11*. ACM, 2011, pp. 23–32.

[39] W.-H. Chuang, R. Garg, and M. Wu, “How secure are power network signature based time stamps?” in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, ser. CCS ’12. New York, NY, USA: ACM, 2012, pp. 428–438.

[40] A. Volokitin, R. Timofte, and L. Van Gool, “Deep features or not: Temperature and time prediction in outdoor scenes,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2016.

[41] B. C. Chen, P. Ghosh, V. I. Morariu, and L. S. Davis, “Detection of metadata tampering through discrepancy between image content and metadata using multi-task deep learning,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, July 2017, pp. 1872–1880.