December 2019

Exposure Prediction in Video Capturing

Jinglun Gao

Szepo Robert Hung

Ruben Velarde

Gang Sun

Exposure Prediction in Video Capturing

Abstract:

This publication describes methods, techniques, and apparatuses that enable a handheld electronic device, such as a smartphone, to capture high-quality static images (photographs) and accurately determine an exposure configuration when capturing videos (multiple frames per second (FPS)). To do so, the smartphone utilizes an auto-exposure (AE) algorithm. The AE algorithm combines data from motion sensors (*e.g.*, gyroscopes and accelerometers) and optical flow data from image sensors (*e.g.*, complementary metal-oxide-semiconductor (CMOS) image sensor). The AE algorithm increases video quality without using additional computational power, more battery life, nor more volatile and non-volatile memory.

Keywords:

Auto-exposure algorithm, AE algorithm, frames per second, FPS, motion sensor, gyroscope, accelerometer, optical flow, luma statistics, exposure configuration, image brightness, exposure prediction, video capturing, video frame, exposure prediction.

Background:

Original device manufacturers (ODMs) of various electronic devices (*e.g.*, smartphones, digital cameras, tablets) may embed one or more cameras in or on the electronic device that enable a user to capture static images (a single frame) or videos (multiple frames per second (FPS)). The quality of the captured frame(s), among other factors, depends on the brightness of the captured frame(s), where an exposure control determines the brightness of the captured frame. The exposure control can have different exposure configurations so that the captured frame(s) can have different brightness effects, as is illustrated in Figure 1.

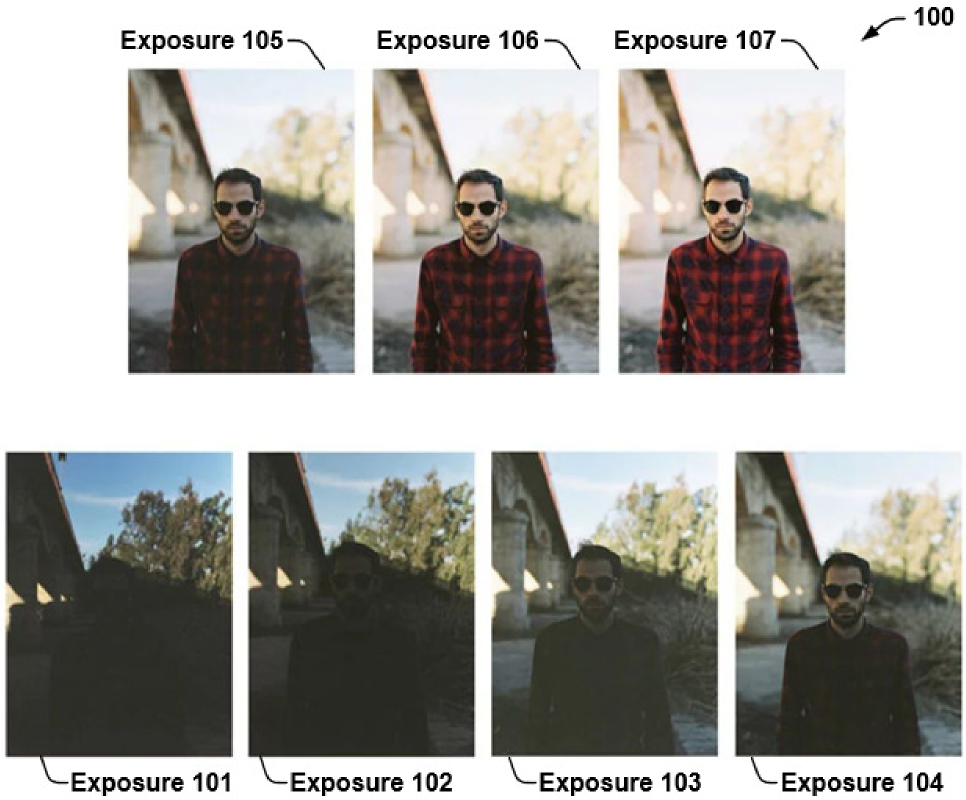

Figure 1

Figure 1 illustrates an example environment 100 of the user capturing the same static scene with seven different exposure configurations, which are illustrated as exposure 101, 102, 103, 104, 105, 106, and 107. The user incrementally increases (from 101 to 107) the exposure to capture the seven images (seven single frames) illustrated in Figure 1, where the image labeled *exposure 101* has the lowest exposure, and the image labeled *exposure 107* has the highest exposure. Figure 1 helps illustrate that to capture images in darker environments, the user needs to increase the exposure. Darker environments may be a result of low-level light conditions, the color of the background of the scene (*e.g.*, white, yellow, green, purple, black), and other light conditions that affect the amount of light, light absorption, and/or light reflection in the scene.

To aid the user determine the exposure configurations, the electronic device often utilizes an auto-exposure (AE) algorithm. Among other things, the AE algorithm utilizes exposure metering (metering), which measures (meters) the reflected light and determines the optimal exposure. The AE algorithm is particularly useful when capturing videos. Conventional video capturing techniques meter an input of a current frame *N* and update the exposure configuration for the next frame *N+1*, as is illustrated in Figure 2.

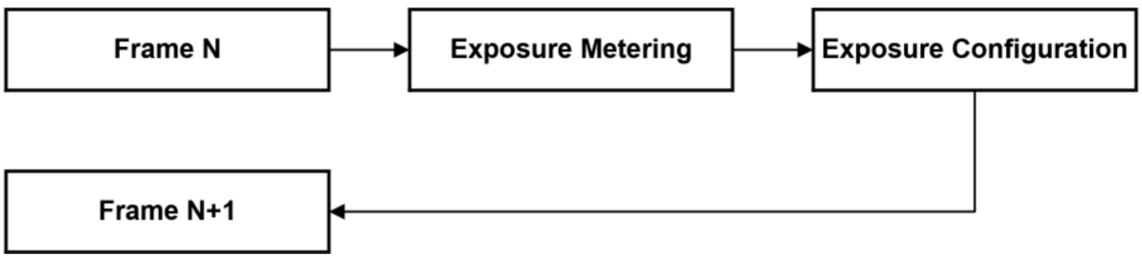

Figure 2

Figure 2 illustrates a common pipeline that conventional AE algorithms often utilize. Assuming the AE algorithm captures the video at 30 FPS, the time difference between the current frame *N* and the next frame *N+1* is 33.33 milliseconds (ms). Thus, the conventional AE algorithm can update the exposure configuration every 33.33 ms. In many cases, such a solution may be adequate to determine the brightness of the next frame *N+1* when the electronic device is stationary. Nevertheless, such a solution often fails to adequately determine the brightness of the next frame *N+1*.

For example, assume the user is riding with a friend on the Glacier Express train through the Swiss Alps. As the train passes through narrow valleys, around tight curves, through tunnels, and crosses bridges, the brightness of the scenes differ dramatically throughout the trip. The user decides to take several videos throughout the train ride using a smartphone that utilizes a conventional AE algorithm to capture some of the most interesting parts of the train ride. In one scene, the user wants to capture the dramatic effect of the train coming out of a tunnel. The AE algorithm of the smartphone updates the exposure configuration every 33.33 ms and is capable of decreasing the exposure fast enough to adjust for the incremental increase in ambient light as the train nears the end of the tunnel. As the user and their friend witness this beautiful scene, the friend exclaims, “That is so beautiful!” The user turns the smartphone toward their friend and then back towards the end of the tunnel as the train exits the tunnel. Moments later, the user plays back the video to “relive” that moment, and they notice that the video was overexposed when the user turned their phone toward their friend and underexposed when the user turns their phone toward the end of the tunnel.

Thus, in aspects, the conventional techniques and methods may not properly support exposure configuration when the electronic device is not stationary, as is the case when the user captures a video using a handheld electronic device, because updating the exposure configuration every 33.33 ms (equivalent to 30 FPS) may not be fast enough. Capturing a video using more than 30 FPS (*e.g.*, 100 FPS) is computationally expensive (*e.g.*, image processing), power-hungry (*e.g.*, battery power), requires more volatile (*e.g.*, dynamic random-access memory (DRAM)) and non-volatile (*e.g.*, Flash) memory, and is difficult to share due to the required large files.

Therefore, it is desirable to have a technological solution that can successfully capture a video by predicting the exposure configuration when the user utilizes a handheld electronic device.

Description:

This publication describes methods, techniques, and apparatuses that enable a handheld electronic device, such as a smartphone, to capture high-quality static scenes (photographs) and accurately determine the exposure configuration when capturing videos.

Image-Based Motion Prediction

Conventional AE algorithms, using optical flow, can detect motion in a two-dimensional (2D) image domain, which is often suitable for capturing a static scene (a photograph) because it offers an intense local motion. To capture a video, the conventional AE algorithms may rely on techniques that stabilize an image by using only a portion of an image sensor output window, as is illustrated in Figure 3.

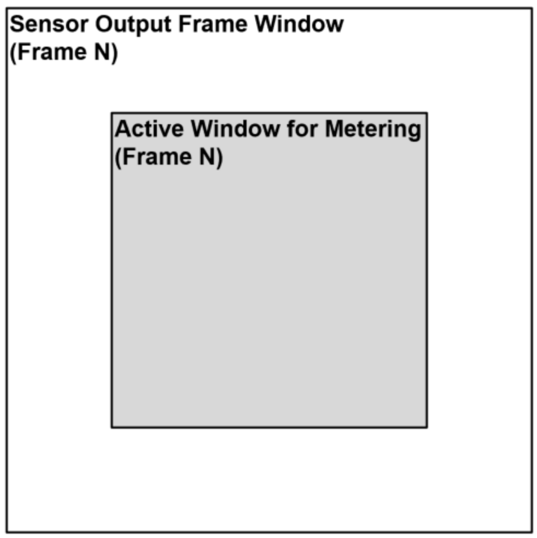

Figure 3

As is illustrated in Figure 3, a conventional AE algorithm crops (*e.g.*, 80% in each direction) the sensor output frame window of the current frame *N* captured by an image sensor. Examples of image sensors include complementary metal-oxide-semiconductor (CMOS) image sensor and charge-coupled device (CCD) image sensors. Then, the conventional AE algorithm performs exposure metering on the center window of the frame *N*, herein referred to as "active window for metering," and uses that metering to configure an exposure configuration for the next frame *N+1*. Without considering a movement of the smartphone, exposure metering of a centered symmetrical cropping of the current frame *N*, as illustrated in Figure 3, is an acceptable way to predict the exposure configuration for the next frame *N+1*. Recall the example of the user on the train, this conventional AE algorithm performs poorly when the user moves the smartphone while capturing the video.

Combination of Gyro-Based and Image-Based Motion Prediction

ODMs often embed motion sensors (*e.g.*, gyroscopes, accelerometers) in or on the smartphone that can detect and calculate the motion of the smartphone in six degrees of freedom (6DoF) in a three-dimensional (3D) space, as is illustrated in Figure 4.

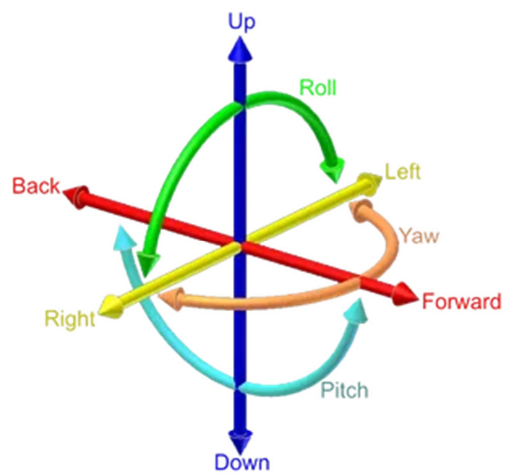

Figure 4

As is illustrated in Figure 4, the motion sensors of the smartphone can measure the surge (forward or backward), heave (up or down), and sway (left or right), combined with rotation about three axes, which are often referred to as yaw (normal axis), pitch (transverse axis), and roll (longitudinal axis). The smartphone can further map the 6DoF to a 3D image domain by using several filters (*e.g.*, a Kalman filter) to predict the motion in the next frame $N+1$. Thus, the smartphone can predict whether the next frame $N+1$ is up or down, left or right, and forward or backward, compared to the current frame N. The AE algorithm can take the predicted motion into account to determine the exposure configuration for the next frame $N+1$, as is illustrated in Figure 5.

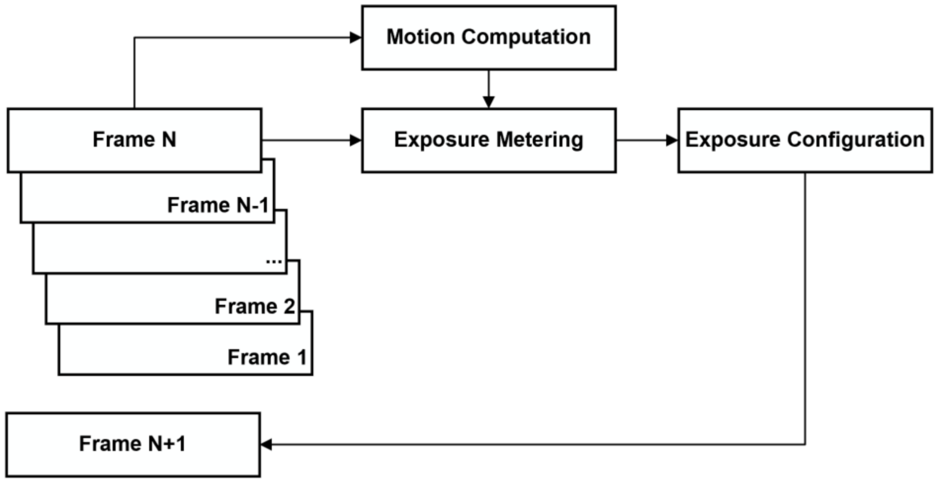

Figure 5

Figure 5 illustrates a pipeline that allows the AE algorithm to consider the motion of the smartphone when capturing a video. Using data from the smartphone's motion sensors, the AE algorithm can apply image stabilization by adjusting how the AE algorithm crops the current frame *N* by favoring the direction of the next frame *N+1*, as is illustrated in Figure 6.

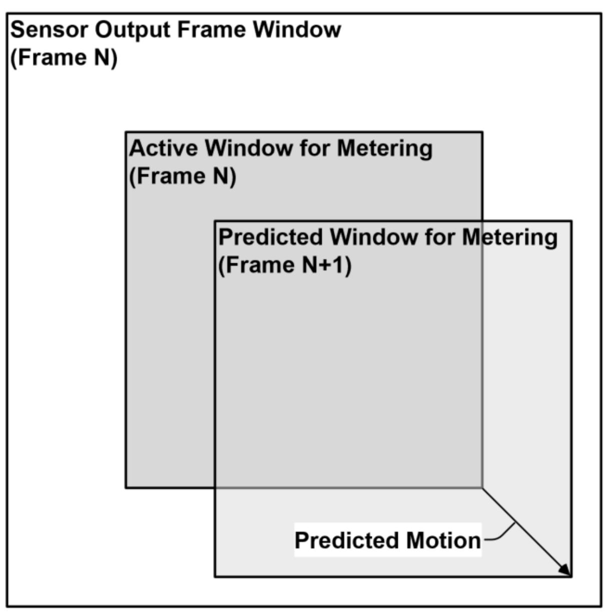

Figure 6

Similar to Figure 3, the AE algorithm in Figure 6 crops the output window of the current frame *N* captured by the image sensor. The AE algorithm of Figure 6, however, uses a combination of gyro-based (and/or accelerometer-based) and image-based motion prediction and performs exposure metering on an area of the sensor output frame window of the current frame *N* that favors the predicted motion of the next frame *N+1*, herein referred to as "a predicted window for metering." The AE algorithm of Figure 6 uses that metering to configure an exposure configuration for the next frame *N+1*. Recalling again the example of the user on the train—if the user utilizes a smartphone that supports video capturing using the described AE algorithm, in Figures 4, 5, and 6, which uses a combination of gyro-based (and/or accelerometer-based) and image-based motion prediction, the user can move the smartphone toward their friend and then back towards the end of the tunnel and capture a video that minimizes overexposure and/or underexposure.

Latency Comparison

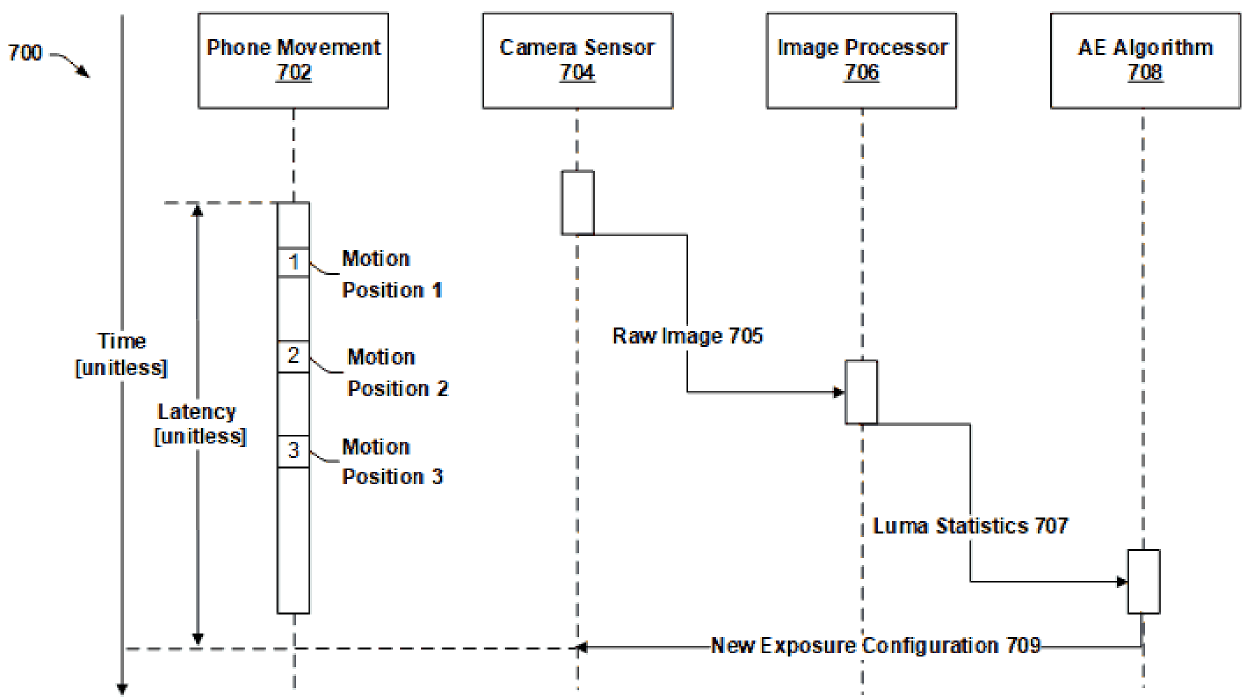

Figure 7

Figure 7 illustrates an example environment 700 of a smartphone with an AE algorithm 708 that utilizes only image-based motion prediction. A camera sensor 704 accumulates the photons of the current frame *N* in 33.33 ms (for 30 FPS) and generates a current frame *N* that contains a raw image 705. The AE algorithm sends the raw image 705 to an image processor 706. The image processor 706 processes the raw image 705 and, among other calculations, generates luma statistics 707. The luma statistics 707 determine the brightness of the current frame *N*. The AE algorithm 708 receives the luma statistics 707 and determines a new exposure configuration 709 for the next frame *N+1*. If the current frame *N* is brighter than needs to be, the AE algorithm 708 reduces the exposure time in the new exposure configuration 709. On the other hand, if the current frame *N* is darker than needs to be, the AE algorithm 708 increases the exposure time in the new exposure configuration 709.

In the example of Figure 7, however, between the current frame *N* and the next frame *N+1*, the user moves the smartphone pointing at several positions, herein labeled as motion positions 1, 2, and 3. When the smartphone uses the AE algorithm 708, the exposure time for the next frame *N+1* is independent of the motion of the smartphone. Specifically, in the example of Figure 7, the new exposure configuration 709 reflects motion position 1 instead of motion position 3. The smartphone's camera is pointing at motion position 3, but the AE algorithm 708 uses data from motion position 1 to make a new exposure configuration 709. Therefore, there is a considerable lag between the current frame *N* and the next frame *N+1*. Note that in Figure 7 and in the subsequent Figure 8, time and latency are illustrated as unitless—the comparison is qualitative instead of quantitative.

Alternatively, consider an AE algorithm 808 that utilizes a combination of gyro-based (and/or accelerometer-based) and image-based motion prediction, as is illustrated in Figure 8.

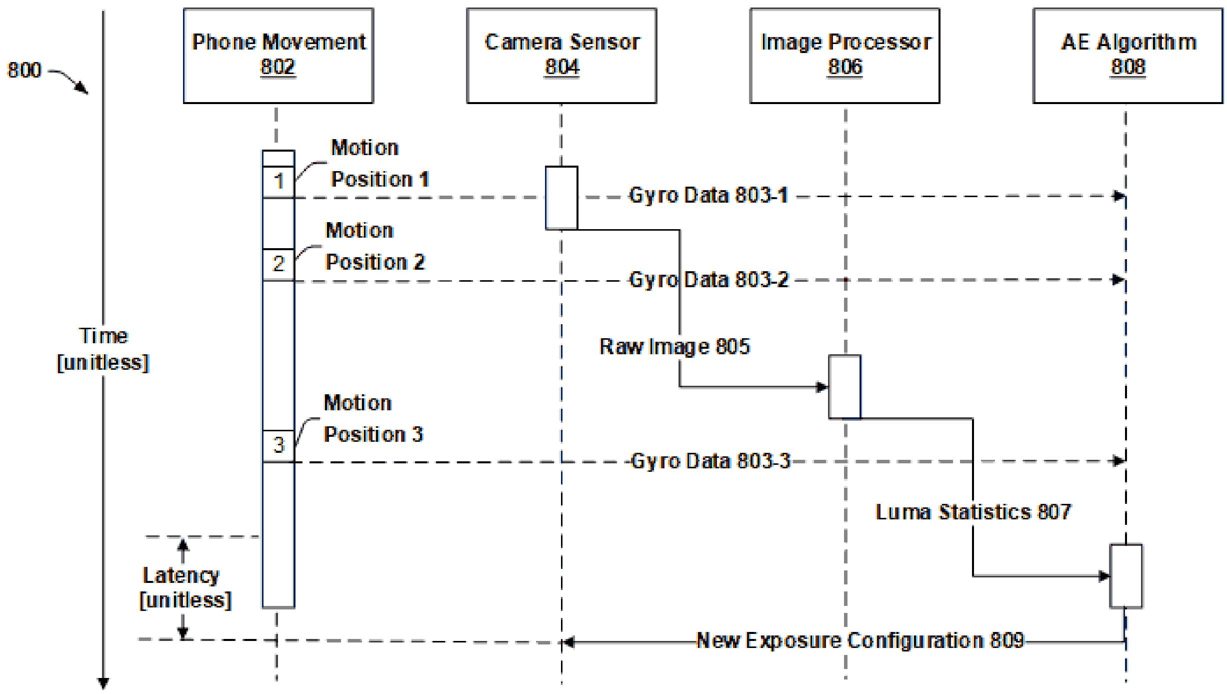

Figure 8

Figure 8 illustrates an example environment 800 of a smartphone with an AE algorithm 808 that utilizes a combination of gyro-based (and/or accelerometer-based) and image-based motion prediction. Similar to Figure 7, in Figure 8, a camera sensor 804 accumulates the photons of the current frame *N* in 33.33 ms (for 30 FPS) and generates a current frame *N* that contains a raw image 805. The AE algorithm 808 sends the raw image 805 to an image processor 806. The image processor 806 processes the raw image 805 and generates luma statistics 807. The luma statistics 807 determine the brightness of the current frame *N*. The AE algorithm 808 receives the luma statistics 807. In addition, the AE algorithm 808 receives gyro data 803-1 of the motion position 1, gyro data 803-2 of the motion position 2, and gyro data 803-3 for the motion position 3. Thus, the AE algorithm 808 uses the raw image data 805, the luma statistics 807, *and* the gyro data 803-3 to determine the new exposure configuration 809 for the next frame *N+1*. As a result, latency is considerably lower because it reflects data from position motion 3, which are more in-line with the next frame *N+1*.

In conclusion, an auto-exposure algorithm that utilizes a combination of gyro-based (and/or accelerometer-based) and image-based motion prediction increases video quality without using additional computational power, more battery life, nor more volatile and non-volatile memory.

References:

[1] Patent Publication: US20130027581A1. Adaptive auto exposure adjustment. Priority Date: July 29, 2011.

[2] Auysakul, Jutamanee, He Xu, and Vishwanath Pooneeth. "A Hybrid Motion Estimation for Video Stabilization Based on an IMU Sensor." *Sensors (Basel)* 18, no. 8 (August 17, 2018). https://doi.org/10.3390/s18082708.