An Artist-Friendly Workflow for Panoramic HDRI

Sébastien Lagarde, Sébastien Lachambre, Cyril Jover
Unity Technologies

Contents

1 Introduction

Image-based lighting (IBL) with high dynamic range images (HDRI) is a well-known concept in the VFX and videogame industries, since Paul Debevec's SIGGRAPH'97 paper [DM97] that popularized the technique. Panoramic HDRI as a lighting source allows one to easily reproduce lighting from the real world and to better integrate a CG object inside an environment—a technique that's widely used today.

Figure 1: Example of HDRI lighting.

With the current interest in physically based rendering (PBR), there is a growing need for accurate HDRI creation to use them as a light source. An accurate HDRI:

- captures the full range of lighting in the scene, without clipping
- has minimal lens artifacts
- is linear
- doesn't include any artistic effects

In addition to the above requirements, we want to be able to recover the real-world intensity of an HDRI's texels. Combined with physical light units, this allows us to use panoramic HDRIs and virtual lights together, with the correct lighting ratio. This ratio is usually eye-balled by artists. This document is oriented toward artists and describes an artist-friendly workflow for producing and using accurate HDRIs with the best practices we found when shooting our own HDRIs for Unity. For this workflow, we have avoided using complex devices or building a custom setup, so as many artists as possible will be able to reproduce our steps.

Before digging into the HDRI creation process, it is important to first understand the underlying mechanism that allows us to convert the scene lighting into a digital image. This is described in the next section.

2 The path of light from an emitted light source to a digital image

Typical consumer cameras used by artists to capture HDRI are designed to produce an image that is close to what human eyes are seeing. The different elements that compose such a camera are chosen to enforce this design. The various imaging pipeline steps can be described as:

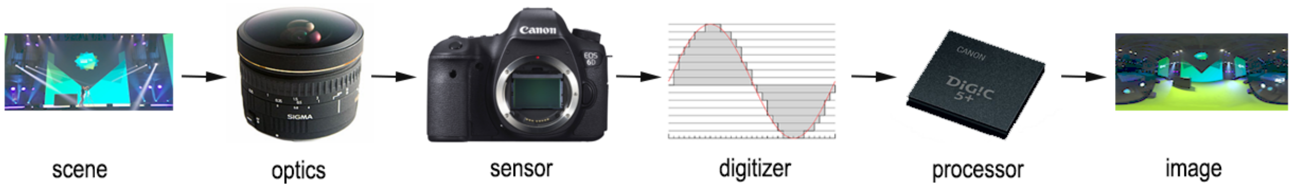

Figure 2: High level overview of the path of light in a camera.

- **scene:** represents the real world that is measured in radiometric units.
- **optics system:** orients and focuses light onto the sensor surface.
- **sensor:** records the accumulated visible light during the exposure time of the shot.
- **digitizer:** converts the analog lighting signal to a digital one.
- **processor:** converts the digital signal to a digital image.

Image capture is a destructive process: the light is filtered and transformed to match human vision and so information is lost. Understanding the path of light through a camera helps us to understand the data found in a digital image and is the purpose of this section. To keep things concise, this document only covers cameras with a Complementary Metal Oxide Semiconductor (CMOS) sensor and an `RGBG` Bayer filter as they are the most popular technologies found in our context of HDRI capture by artists. In particular, our explanation may be illustrated with a Canon 6D camera as it is the model used at Unity. Readers interested in getting deeper into the details of other kinds of camera systems should refer to the book *The Manual of Photography* [AT11].

In a real world scene, photons are emitted by various light sources, such as the sun. These photons go through the lens optics, which focus them onto the sensor. They are filtered by an Optical Low-pass Filter (OLPF) that blurs the signal to reduce aliasing, followed by an infrared (IR) filter that removes near-infrared-wavelength photons. The photons are then focused with a microlens onto a pixel of the CMOS sensor array. Each pixel has a red, green or blue filter in front of it called a Bayer filter. Pixels accumulate an electric charge that is proportional to the number of photons that it collects.

After the sensor has been exposed to light for a shot, the charge from each pixel is read out and converted to a voltage. The pixel charge is reset to zero, i.e. to black. The conversion process adds electrical circuit noise to the light voltage. Further noise is added when the voltage is amplified in order to work with the camera in low light at increased ISO. The voltage is then converted into a binary number by a 10-bit to 14-bit analog-to-digital converter. Metadata such as camera white balance or the sensor's color filter are added and stored in a RAW file in the camera's memory.

The RAW data is stored in *native* RGB, or RGB in camera color space, which is specific to the camera. This is then transformed to XYZ color space, then sRGB. Various image processing techniques can be performed at this stage, such as white balance, sharpness or gamma correction.

The following paragraphs explain, in detail, the path of the light for the various steps described above. Readers that are either already familiar with camera internals or aren't interested in going deeply into details can proceed directly to Section 3.

2.1 Color

Color derives from the spectrum of light interacting in the eye with the spectral sensitivities of the light receptors. This section defines a few terms involved in this process.

2.1.1 Radiance and Luminance

Light units are related to light measurements, which are split into two categories:

- **radiometric**: deals with "pure" physical quantities and is used in the context of optical radiation measurement and spectral rendering[1].
- **photometric**: concerned only with radiation falling within the visible spectrum.

Quantities derived in radiometry and photometry are closely related: photometry is essentially radiometry weighted by the sensitivity of the human eye. These two forms have been widely covered in the literature [Rei+08]. The most commonly used radiometric and photometric quantities are listed in Table 1. The energy subscript e is used for radiometric quantities and the visual subscript v is used for photometric quantities.

Quantity	Radiometric term	Units	Photometric term	Units
Energy	Radiant energy Q_e	J (Joule)	Luminous energy Q_v	$lm.s$
Power	Radiant flux Φ_e or Radiant power	$\frac{J}{s}$ or Watt(W)	Luminous flux Φ_v or Luminous power	Lumen (lm)
Power per solid angle	Radiant intensity I_e	$\frac{W}{sr}$	Luminous intensity I_v	$\frac{lm}{sr}$ or Candela (cd)
Power per area	Radiant exitance M_e or Irradiance E_e	$\frac{W}{m^2}$	Luminous exitance M_v or Illuminance E_v	$\frac{lm}{m^2}$ or Lux (lx)
Power per area per solid angle	Radiance L_e	$\frac{W}{m^2.sr}$	Luminance L_v	$\frac{lm}{m^2.sr} = \frac{cd}{m^2}$ or Nit (nt)

Table 1: Radiometric and photometric quantities.

The sensitivity of the human eye is represented by the CIE photometric curve $V(\lambda)$. It follows a bell-shaped curve (Figure 3) that represents how efficiently our eyes pick up certain light wavelengths. The sensitivity of human eyes peaks at 555nm, which appears to us as green. At this wavelength the sensitivity function value is 1 unit, meaning 100% efficiency.

2.1.2 The electromagnetic spectrum

The visible spectrum is the portion of the electromagnetic spectrum that is visible to the human eye. Electromagnetic radiation in this range of wavelengths is called *visible light* or simply *light*. A typical human eye will respond to wavelengths from about 390nm to 700nm [Wikk]. This is the portion that

[1] Optical radiation is radiation that obeys the principles of optics whose wavelength region approximately includes ultraviolet, visible and infrared radiation.

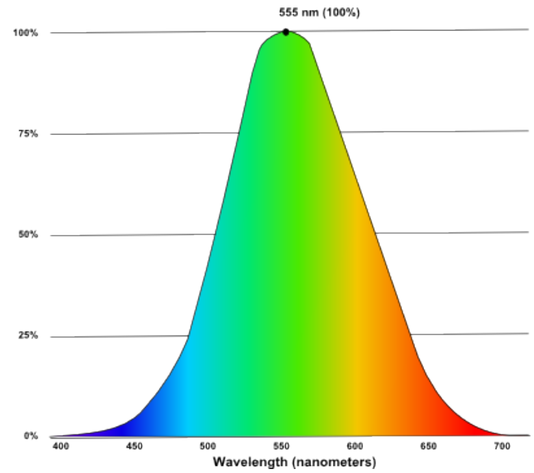

Figure 3: The sensitivity curve of the human eye.

a typical camera tries to capture. Infrared is invisible radiant energy, electromagnetic radiation with longer wavelengths than those of visible light, extending from the nominal red edge of the visible spectrum at 700 nanometers. See Figure 4.

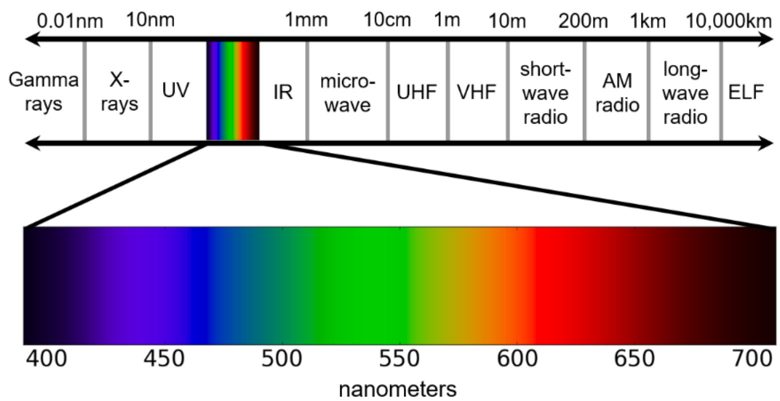

Figure 4: Light spectrum. Image courtesy of Naty Hoffman, used with permission.

2.1.3 CIE Color Matching system

The human visual system responds to three color signals: blue, green and red light. In 1931 an experiment was performed, and the three primary colors of red (700nm), green (546.1nm) and blue (425.8nm) were chosen such that they will each stimulate only one of the three cones of a human observer. The right amount of each of these primaries can, in theory, create any visible color to a human observer. One needs to measure what amount of light of each of the three primaries is needed to reproduce each color (1–5nm wide) in the visible spectrum. These responses are called the CIE standard observer color matching functions $\bar{r}(\lambda)$, $\bar{g}(\lambda)$, $\bar{b}(\lambda)$. Based on these curves, the CIE XYZ color system was defined to make resulting color matching functions $\bar{x}(\lambda)$, $\bar{y}(\lambda)$, $\bar{z}(\lambda)$ (see Figure 5[2]) positive for all wavelengths and the $\bar{y}(\lambda)$ curve equal to the CIE photometric curve $V(\lambda)$. These curves give a way to convert a spectrum to three values—for example XYZ or RGB. For more details on this topic, please refer to [Wikc].

[2] Separate sets of three color matching functions are specified for the 2° Standard Observer and 10° Supplementary Standard Observer. In this document we only refer to the first set.

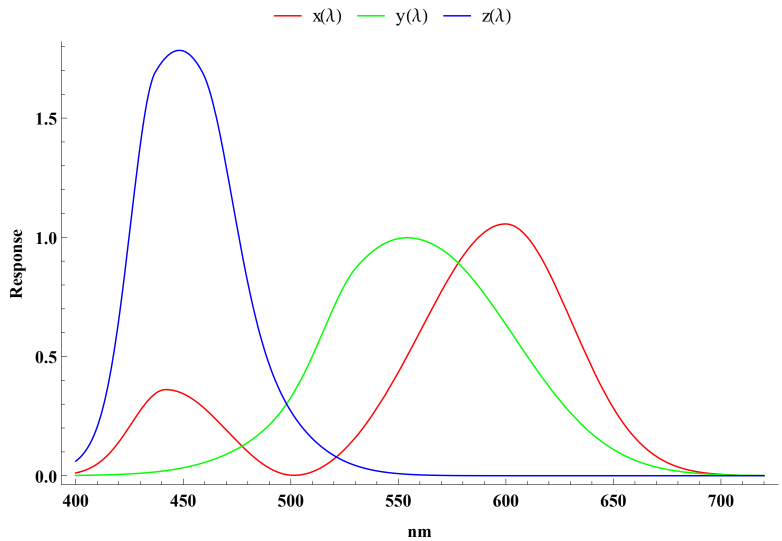

Figure 5: The $\bar{x}(\lambda)$, $\bar{y}(\lambda)$, $\bar{z}(\lambda)$ CIE color matching functions.

2.2 Scene

The world is described by color science using radiometric units, and a real world scene is composed of light that emits photons[3] and matter. The photon can exhibit wave-like behavior, resulting in phenomena such as refraction or diffraction. It also behaves as a particle when interacting with matter at a subatomic level, exchanging energy in discrete amounts. The amount of energy exchanged during such an interaction is

$$E = \frac{hc}{\lambda}, \tag{1}$$

where E is photon energy, h is Planck's constant, c is the speed of light, and λ is the photon's wavelength. This means that shorter wavelengths of light will have higher photon energy. The photons will interact with the camera optics system before being captured by the sensor.

2.3 Optics

An actual optical system is composed of multiple lenses, an aperture, a barrel and motors. The lens is necessary in order to collect light and acquire an image; its transmissive optical device affects the focus of a light beam through refraction. Figure 6 shows an example of lens configuration for a Nikon 24–70mm.

[3] A photon is a single quantum of energy that can only exist at the speed of light. It is the fundamental element carrying all electromagnetic radiation.

Figure 6: Nikon 24–70mm f/2.8E ED VR optical lenses. Source [Phoa].

An *ideal* lens forms geometrically accurate images, but actual lenses don't since the refractive index of glass varies with light wavelength. The multiple lenses cause optical side effects due to the nature of light: vignetting, aberrations and glare. As each of these affects light intensity, it's important to understand their impact.

Vignetting

Vignetting is a reduction of an image's brightness at the periphery compared to the center. There are several causes of vignetting, and the actual physical phenomenon is very complicated. Two main causes are natural vignetting and optical vignetting.

Natural vignetting is the *cosine fourth* law of illumination falloff:

> The brightness of the image away from the optical axis falls off at a rate proportional to the cosine to the fourth power of the angle the light makes to the perpendicular at the focal plane.

Optical vignetting is caused by the physical dimensions of a multiple-element lens. Rear elements are shaded by elements in front of them, which reduces the effective lens opening for off-axis incident light. Wide-open apertures tend to accentuate the effect of optical vignetting.

Figure 7: Example of vignetting, note the darkening in the corner.

Aberrations

There are two classes of aberrations: monochromatic and chromatic aberrations [Wikg]. Aberrations can be split into two categories: those that affect all parts of the image field including the central zone (axial influence) and those that affect only rays passing obliquely through the lens and do not affect the central zone (spherical influence) [AT11]. The effects of these oblique errors increase with the distance of an image point from the lens axis.

Monochromatic aberrations are all spherical-influence aberrations[4]:

- Spherical aberrations are a loss of definition in the image due to spherical lens geometry.
- Coma causes off-axis point sources to appear distorted, with a “tail”.
- Field curvature produces a curved image due to a curved lens.
- Astigmatism is similar to coma but happens for small objects at the edges of the field striking an uncorrected lens asymmetrically.
- Distortion deforms and bends light beams and makes lines appear curved in images. The most common forms are barrel and pincushion distortion.

[4]The five monochromatic aberrations listed in this document are also called Seidel aberrations.

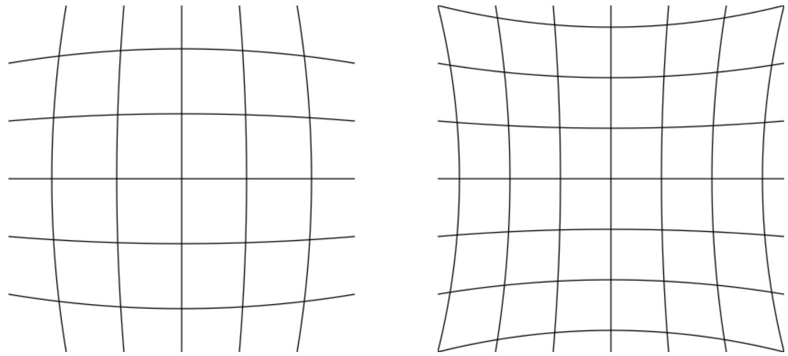

Figure 8: Left: Barrel distortion. Right: Pincushion distortion.

Chromatic aberrations:

Chromatic aberration is caused by variation of refractive indices for each light wavelength of the lens.

- Lateral chromatic aberration is when the lens can't focus all wavelengths of the light to the same convergence point (red/green fringes, blue/yellow fringes). Lateral chromatic aberration has a spherical influence.

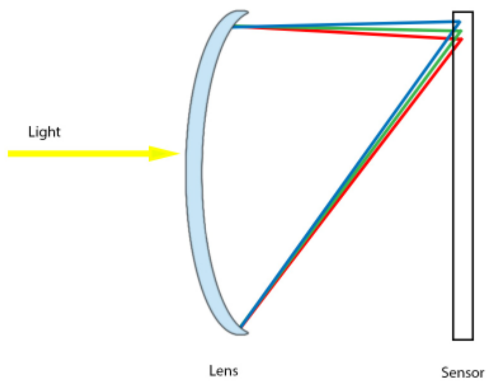

Figure 9: Lateral chromatic aberration.

- Axial chromatic aberration appears when different wavelengths of light are focused at different distances from the lens. Axial chromatic aberration has an axial influence.

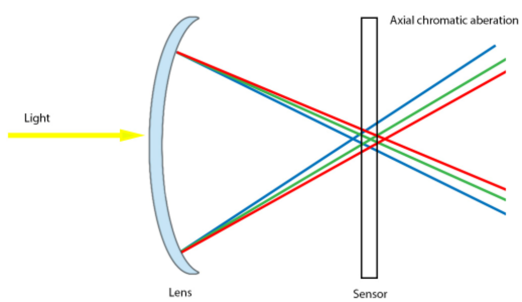

Figure 10: Axial chromatic aberration.

Figure 11: Left: Axial and lateral chromatic aberration. Middle: Axial chromatic aberration. Right: No chromatic aberration.

Glare

Glare is an artifact caused by multiple reflections and refractions in a camera lens. Some of this reflected light is spread uniformly over the surface of the image sensor, and is referred to as ghosting, lens flare or veiling glare. Its effects are greater in the shadowed areas of the image, leading to a reduction in the image illuminance range (contrast).

Figure 12: Left: Lens flare. Right: Veiling glare (simulation).

2.4 Sensor

Once photons are out of the camera optics they pass through various filters: an optical low-pass filter (OLPF), followed by an infrared (IR) filter, then a Bayer filter, before finally being recorded on a CMOS image sensor. See Figure 13. This section provides details for each of these elements.

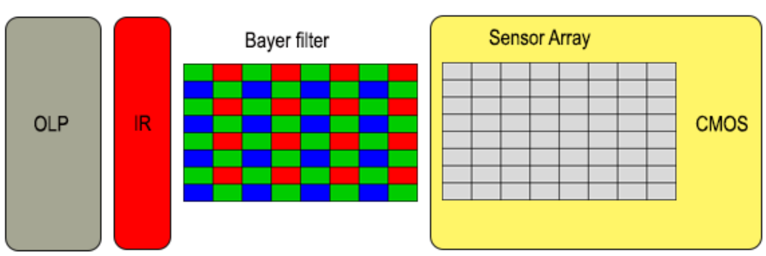

Figure 13: Overview of camera recording system.

A technical description of a CMOS sensor can be found in [Ren15]. A CMOS sensor is composed of an array of photodiodes, often referred to as pixels. Photodiodes are the key elements of a digital image sensor. They are formed from various layers of silicon, and photons interact with the silicon to release a variable number of electrons—a function of the photons' wavelength. This is called the photoelectric effect [Wikh] and is illustrated in Figure 14. Electrons are collected then they are converted into a voltage (see next section).

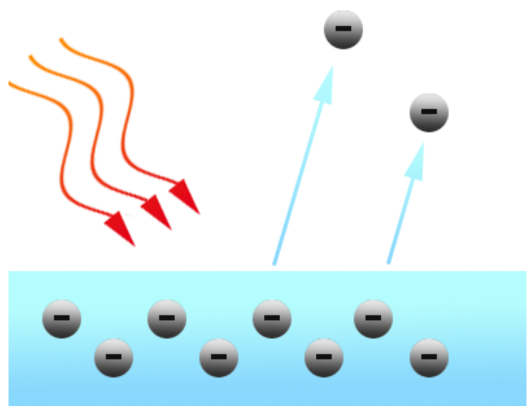

Figure 14: The photon-silicon interaction releases electrons due to the photoelectric effect.

The ratio of incident photons converted to electrons is called quantum efficiency (QE). For example, ten incident photons producing four electrons gives a quantum efficiency of 40%[5].

The geometric structure of a pixel is something like a tunnel and the photodiode is at the bottom of the tunnel in the silicon substrate. Each pixel is equipped of a tiny lens called a microlens that greatly improves the QE as they collect light that would otherwise fall outside the photosensitive area of the pixel.

Images captured from a CMOS sensor, being based on the photoelectric effect, will be monochrome (black and white). In order to distinguish between colors, a color filter array (CFA) is used. A CFA is a mosaic of tiny color filters placed over the pixels of an image sensor in order to capture color information. The most popular mosaic pattern is a sequence of red, green, and blue filters, `RGBG`[6], named the Bayer filter after Kodak engineer Bryce E. Bayer. See Figure 15 (left). With such a mosaic pattern, a photodiode array of 640×480 pixels contains a total of 307,200 pixels covered by 76,800 Bayer quartets.

[5]The penetration depth of a photon in the silicon depends on its wavelength. A deeper penetration depth for a given active area produces less photoelectrons and decreases quantum efficiency.

[6]Note that although this pattern is fairly standard, sensors from different camera manufacturers may have a different "phase". That is, the starting color on the top left pixel may be different. The four options, typically referred to as `RGGB`, `BGGR`, `GBRG`, and `GRBG`, indicate the raster-wise order of the first four colors in the filter.

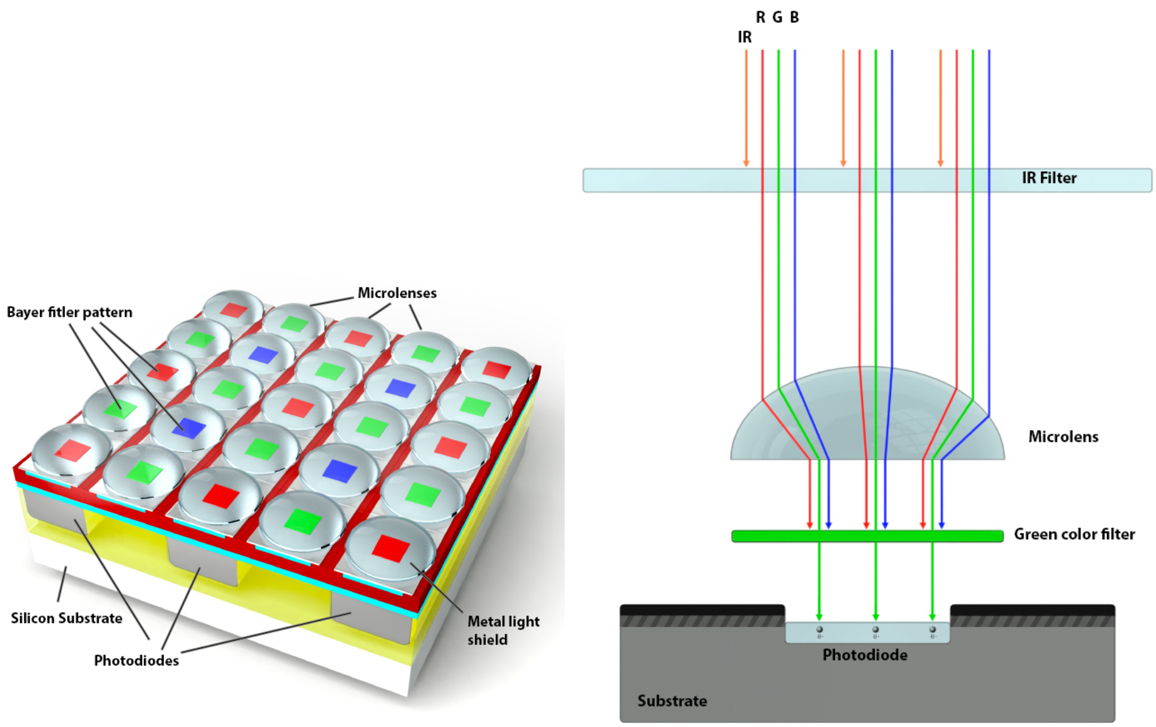

Figure 15: Left: A microlens array on a photodiode array with a Bayer filter. Right: The cross section of a photodiode with an IR filter.

The image sensor measurements are subject to degradation caused by photon and electron leakage, so cross-channel color contamination is unavoidable (red/green fringes). This phenomenon is called cross-talk.

CMOS sensors are sensitive to wavelengths from approximately 350nm to 1050nm, mainly due to the nature of the silicon. This range includes near-infrared wavelengths ($>$ 800nm). To prevent unnatural-looking images, cameras employ an IR filter. An IR filter blocks the transmission of infrared while letting visible light through. See Figure 15 (right). Figure 16 shows the visual impact of an IR filter.

Figure 16: Left: Photography with the internal IR filter in place. Right: With the IR filter removed (simulation).

The spectral sensitivity of a digital camera is determined by the intrinsic sensor QE, the CFA and the IR filter. For the purpose of the following discussion, a Gaussian function will be used to represent the various spectral sensitivity curves as described in [Kri15]. A typical sensor's spectral sensitivity is shown in Figure 17 along with the impact of the IR filter.

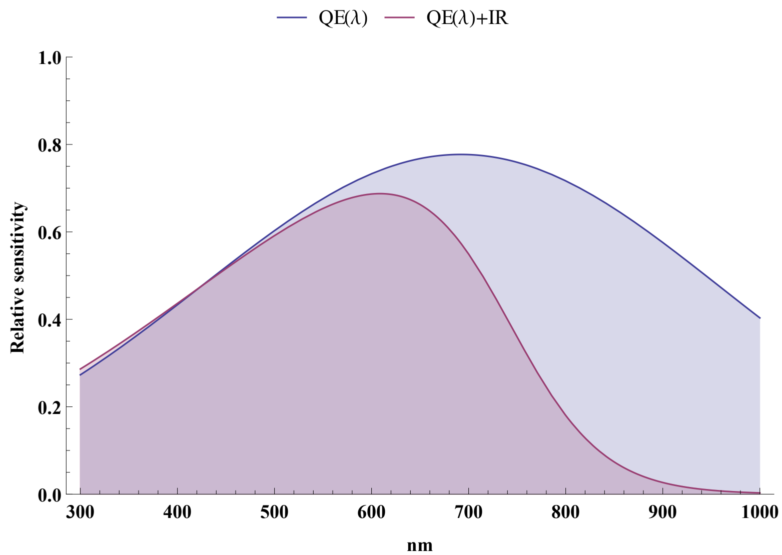

Figure 17: Impact of an IR filter on the QE of a sensor. The grey area is the sensor QE curve and red area is the QE reduced by the IR filter.

Transmission spectral profiles for an imaginary Bayer CFA are shown in Figure 18 with the effective spectral sensitivities after accounting for QE with the IR filter (Figure 17).

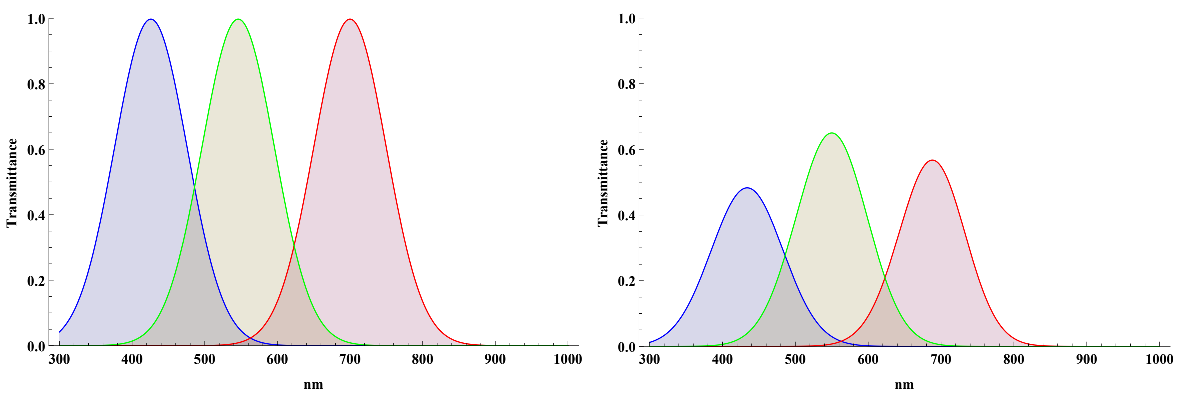

Figure 18: Left: Transmittance of CFA. Right: Effective (relative) spectral sensitivity after accounting for QE with an IR filter (Figure 17).

If a flat spectral input (i.e. a neutral input) is convolved with such effective spectral sensitivities, then the relative exposure is RGB: 0.8875, 1.0, 0.804. The exposure is proportional to the area under the spectral sensitivity curves. Thus, the output will have a green tint to the color instead of being neutral or white. The color imbalance (relative to neutral) can be corrected by changing the CFA, or later in the image chain using a color balancing method on the image—see Figure 19. Note that this color balance is coupled to the sensor, IR filter and CFA and is distinct from the white balance (for simulating the chromatic adaptation of human vision) described in Section 2.7.

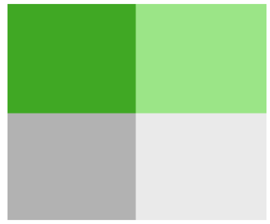

Figure 19: A neutral input produces green patches (first row) with the filters and spectral sensitivities of Figure 18. It should be adjusted to look more neutral (second row) by either changing the CFA or using a color balancing method.

The goal of these effective spectral sensitivity curves is to convert the light spectrum to what should be perceived by the human eye, i.e. the CIE standard observer. Thus they should be close to the CIE color matching curves. One theoretical measure of how well a digital camera's spectral sensitivity records color relative to the CIE color matching curves is the Luther Criteria [Lut27]. It states that if the spectral sensitivities of a digital camera are a linear combination of the CIE color matching curves, then the camera will record the way the CIE standard observer would "see" the world[7].

This ability of a camera to reproduce accurate colors can be represented by the sensitivity metamerism index (SMI) defined in ISO standard 17321. SMI is represented by a number lower than 100. A value equal to 100 is perfect color accuracy, meaning that the camera met the Luther criteria (which never happens in practice). More details, as well as SMI for a few cameras, can be found on DXO mark website [DXOb]. A Canon 6D has an SMI of 69, which is a good score. Figure 20 shows the effective spectral sensitivities of two different cameras.

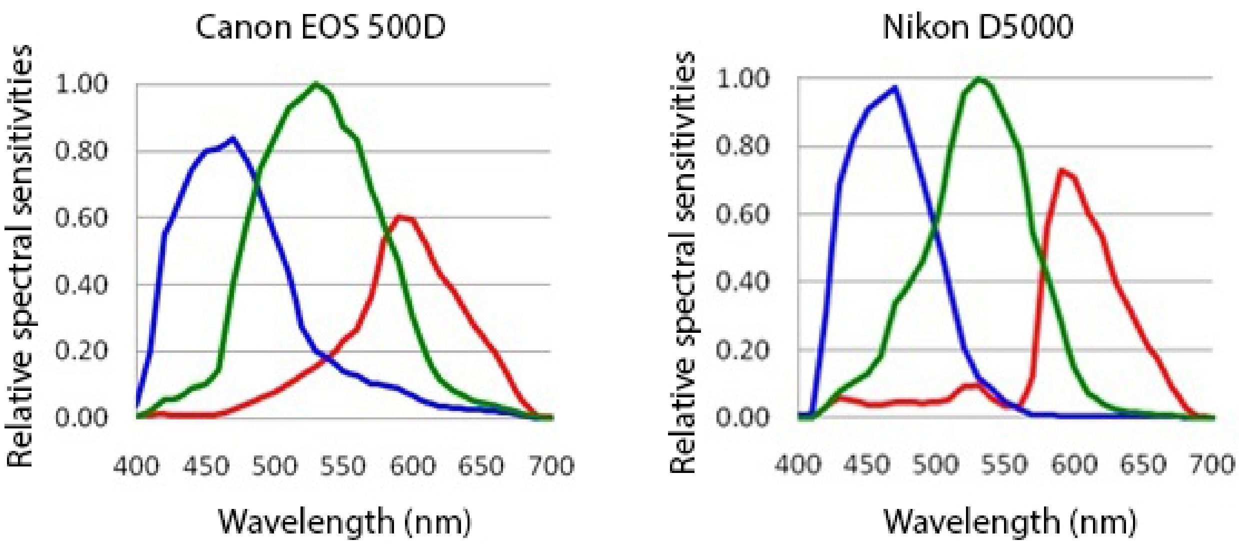

Figure 20: Relative spectral sensitivity for two consumer digital cameras. Source [DXOa].

Finally, the optical low-pass filter (OLPF)—sometimes called an anti-aliasing filter—is used to attenuate frequencies beyond the Nyquist limit of the sensor, to prevent aliasing. High frequency waves cause moiré and false color[8]. Moreover the OLPF can reduce or eliminate the color banding artifacts caused by the CFA [Kri98]. It is positioned before the IR filter. Several images of the same point will

[7]A linear combination of the original primaries include negative terms, meaning that there is no physical way one can reproduce all colors with either original primaries or the linear combination of them for the $\bar{x}(\lambda)$, $\bar{y}(\lambda)$, $\bar{z}(\lambda)$ color matching system. Hence it is not possible to have a perfect CIE colorimetric match to all spectra even when using spectral sensitivities that meet the Luther criteria.

[8]Moiré occurs in scenes containing repetitive details, such as patterns in textiles and clothing or in the strong vertical

be produced, blurring in different directions resulting in a less sharp image. See the impact of removing the OLPF in Figure 21. The OLPF can be located before, after or both before and after the IR filter.

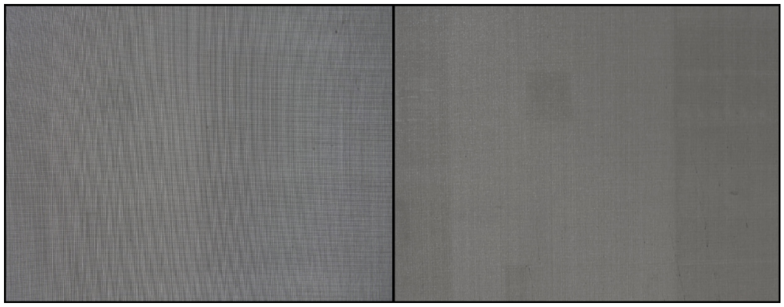

Figure 21: Left: The OLPF is removed (simulation). Right: Photography with the OLPF in place.

2.5 Digitizer

CMOS designs are built around active pixel sensor (APS) technology [Wika]. It means that each pixel is essentially a photodiode and three transistors, performing the functions of resetting or activating the pixel, amplification and charge conversion, and selection or multiplexing. See Figure 22.

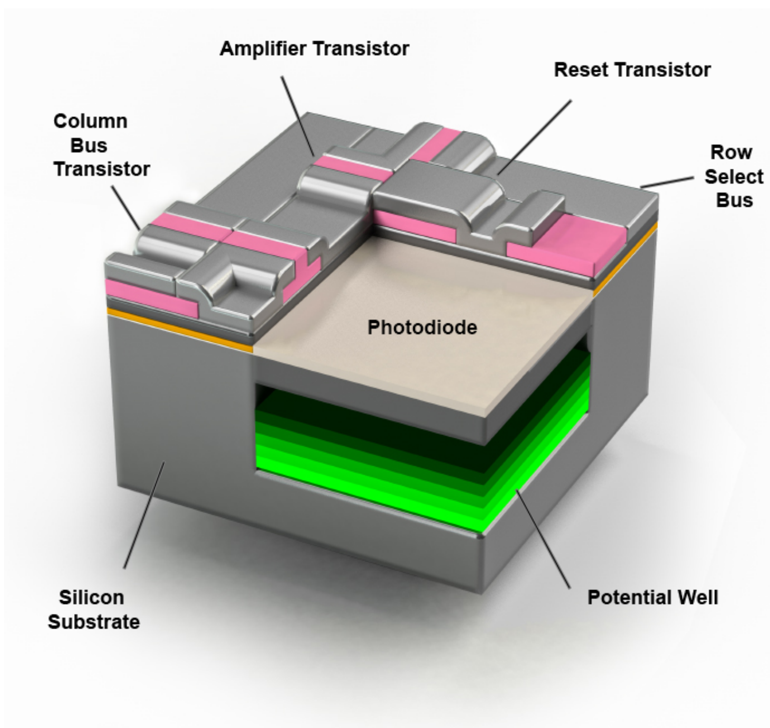

Figure 22: Anatomy of photodiode.

Electrons produced by the incoming photons are collected in a potential well [Wiki] until the integration period (exposure time) is finished. The charge accumulated (voltage) at each photo-site is tiny, of the order of femtovolts, and requires amplification inside the pixel before it is used[9].

lines of architecture. These patterns do not appear in nature, which is why moiré and false color very rarely occur with landscape and nature photography.

[9]The amplification can be different for each pixel's color channel to compensate for differences in relative sensitivity.

When a full well saturates, electrons are split from the saturated pixel to surrounding areas, causing corruption of those signals, known as blooming.

The voltage is then transferred in sequential rows and columns to an output amplifier. The pixel charge is also reset to zero (i.e. black). The voltage is amplified again based on the ISO control of the camera and then transferred to an on-chip analog-to-digital converter (ADC). See Figure 23. The response of the CMOS sensor is considered to be linear relative to the input signal.

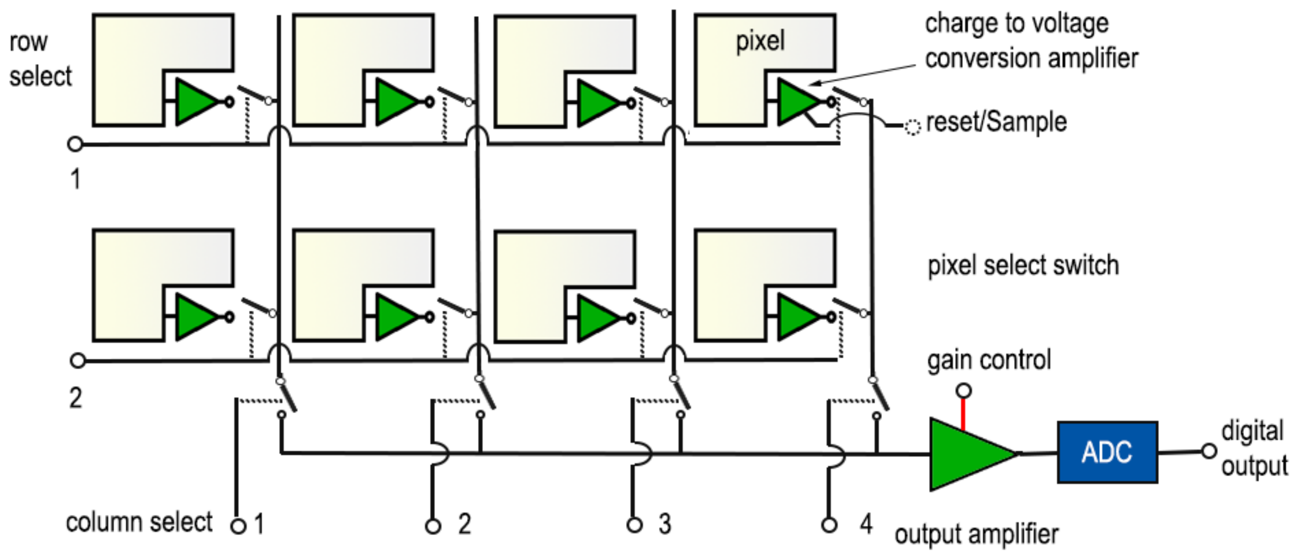

Figure 23: Overview of the camera recording system.

The ADC converts amplified voltage data into a binary number. When the ADC digitizes the dynamic range, it breaks it into individual steps. The total number of steps is specified by the bit depth of the converter. Most DSLR cameras work with 12 bits (4096 steps) or 14 bits (16384 steps) and a camera phone is 10 bits (1024 steps). A Canon 6D is 14 bit.

For every pixel in the sensor, the brightness data is represented by a number from 0 to $2^{bitdepth} - 1$ and is stored in a file. This, together with metadata, is the RAW format.

Noise is a significant artifact of the digitizer process. Noise factors are grouped into two types: temporal and fixed-pattern noise [JW12]. The main sources of noise are listed below.

Temporal noise (changes at each acquisition):

- **photon noise**: a Poisson distribution of incident photons.
- **dark current noise** (or thermally generated noise): an accumulation of charges (electrons) can happen even in an absence of light.
- **readout noise**: reading, resetting, amplification, quantizing and more (hardware is imperfect).

Fixed-pattern noise (present in every acquisition):

- **photoresponse non-uniformity** (PRNU): the surface area of the individual pixels may vary across the sensor array, generating different signal gain.
- **dark signal non-uniformity** (DSNU): differences in the pixel reset circuitry. The circuitry may be more effective in some pixels than others.

2.6 Processor

In this document, the processor refers to the software processing performed on RAW data to get a readable image with linear data. We exclude camera processing that performs the white balance, image enhancement and JPEG compression. Figure 24 shows an overview of the main steps performed by the processor.

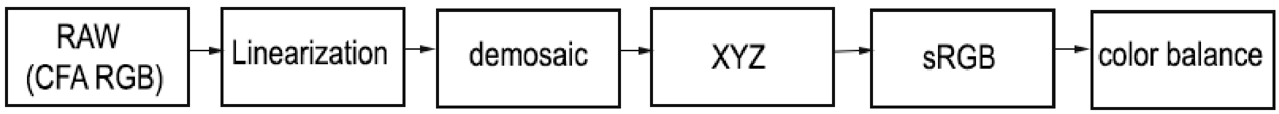

Figure 24: Overview of the processor steps.

Each manufacturer (e.g. Canon or Nikon) has its own software that converts its proprietary RAW format (.CR2 for Canon[10] and .NEF for Nikon). Generic solutions also exist, such as `dcraw` [Cof], which handles various RAW formats. This section provides more details about the different steps performed by these pieces of software. For the interested reader, [Sum14] provides MATLAB code for these steps.

Linearization + Normalization

The RAW data is not always a linear image. Some cameras (e.g Nikon) apply a non-linear transformation for storage purposes. A lookup table with transform data can be found in the EXIF [Wike] metadata by using the EXIF tools from Phil Harvey [Har] and are used to linearize the RAW data.

Once linearized, a raw image has a possibly non-zero minimum value to represent 'black' and a saturation point of the physical CMOS sensor to represent 'white'. The data is remapped using this min/max range and normalized to [0, 1].

Note: the output may also be larger than the expected pixel dimensions of the camera, including a border of unexposed pixels to the left and above the meaningfully exposed ones.

Demosaicing

After a raw image has been obtained from a CMOS sensor by a Bayer filter, it must be converted into a standard red, green and blue (RGB) format through an interpolation methodology called demosaicing. There are a variety of demosaicing algorithms, such as Nearest Neighbor Interpolation, Bilinear Interpolation, and Bi-cubic Interpolation. A survey of various algorithms can be found in [LMY10].

Conversion to XYZ

The color values obtained through the color demosaicing process are called native RGB or camera space RGB. As described in Section 2.4, due to the spectral sensitivity curves of the color filter and the QE of the CMOS sensor, the native RGB data may not provide a faithful color rendition as it doesn't respect the Luther criteria. An additional color correction step is required in order to get the RGB values into a colorimetric color space such as XYZ (generated with the CIE color matching curve $\bar{x}(\lambda)$, $\bar{y}(\lambda)$, $\bar{z}(\lambda)$). This can be approximated by a 3×3 color matrix[11].

Generating the correct values for the 3×3 color matrix requires a great deal of image science

[10] Canon's RAW format (.CR2) has been partially reverse engineered [lcl15].

[11] We can transform native RGB into XYZ by imposing the Luther criteria: that the sensor response curves are a linear combination of the CIE color matching function. However, this is still an approximation.

knowledge and is called spectral response curve characterization[12]. [PNB13] provides a method to quickly characterize cameras. These matrices have been calculated by Adobe for various cameras and can be found in DNG files converted from RAW files as described in the DNG specification [Adoc]. This is how `dcraw` has collected its own set of matrix transforms.

Conversion to sRGB

After the conversion to XYZ, the color is converted to sRGB using the associated 3×3 color matrix [Lin].

Color balance

As described in Section 2.4, imbalances of color due to the effective spectral sensitivity of the sensor may happen, which can be compensated for by a color balance step. Do not confuse this color balance step with the white balance due to the illuminant (see Section 2.7). The manufacturer provides, in EXIF metadata, combined white balance weights for both the color balance of the sensor and the illuminant. However in this document we do not want to use the white balance from the camera since `dcraw` already normalizes its conversion matrix so that a white input outputs a white value, which solves this imbalance[13].

2.7 Chromatic and photometric calibration

Chromatic adaptation is the human visual system's ability to adjust to changes in illumination in order to preserve the appearance of object colors. It is responsible for the stable appearance of object colors despite the wide variation of light [Wikb]. This is not the case for a camera. In order for a camera to mimic human vision, it must perform this chromatic adaptation. This is called white balance. When using a RAW format, there is no white balance applied to the data, so it is up to the processor to perform it.

Photometric (or luminance) calibration is a way to recover real-world (absolute) intensity from the relative intensity found in RAW data. The first question to ask is: "in what units is the RAW data?"

Real world scenes are in radiometric units: the radiance goes through the lens and the sensor accumulates irradiance on its pixel area. Then electrons are generated and a digital value is produced. This process is linear and this digital value can be qualified as relative radiance RGB, or relative luminance RGB. The *relative* qualifier means we are missing a factor to change it into an absolute unit. As the difference between radiance and luminance is also a factor (lumens and watts are different by a factor of around 683 in XYZ space) there is no way to qualify this relative unit as radiometric or photometric unless we convert it to an absolute unit. In this document, we use the term relative luminance and we perform a calibration step to transform the relative luminance to absolute luminance.

[12] Note that there are spectral response curves and camera response curves in the literature. These are two different characterizations. The first is about the transform from native RGB to XYZ, the second is about the relation between the incoming light and the image pixel values (which should be linear for a RAW file).

[13] With a Canon 6D a linear output of `dcraw` that only performs demosaicing and applies weights of 1, 1, 1, 1 to the Bayer filter (which bypasses the correction of imbalanced color) produces a green tinted image.

3 Equipment and softwares

3.1 Equipment

HDRI acquisition requires many pieces of photography equipment. Here, we provide the list of equipment we use to capture HDRIs at Unity. In general, we prefer to rely on affordable devices with a good quality-to-price ratio. Every time we move to a new location for HDRI acquisition, we use this list as a checklist and have found it to be a good practice to adopt.

Figure 25: Equipment list.

1. Camera: Canon EOS 6D
2. Lens: Fisheye Sigma 8 mm f/4 EX-DG Circular (include filter support)
3. Tripod: Manfrotto 475 B.Digital pro tripod black
4. Tripod bag
5. Nodal Ninja Head 4 (highly recommended)
6. Remote: CamRanger + USB cable
7. Mobile: Android or IOS

8. Laptop: Mac Book Pro (optional: dedicated to storage and processing data during long trip)
9. Secondary camera battery
10. Memory Card 256GB (Class 10)
11. Colorchecker Classic (X-rite)
12. Colorchecker Passport Photo (X-rite)
13. Lux meter: Sauter SO 200K (additional battery can be necessary)
14. Luminance meter: Konica Minolta LS - 110 (additional battery can be necessary)
15. Spirit level
16. HOYA ND filter: 16, 32, 64, 500, 1000. Unity use power of 2 ND filters to simplify the workflow. HOYA are recommended due to their filters' reduced vignetting and color shift.
17. Bag
18. Screwdriver
19. Lenspen (lens cleaner tools)
20. Notepad and pen to record information about the photographs: location, time, weather, lux, luminance, range, EV step, lens setting
21. Marker
22. Ninja Nodal Nadir Adapter

During long shooting sessions, be aware of the amount of data that can be captured. Additional batteries and hard drives are important to avoid interrupting the shooting session. Also, it is good practice to take a picture of the notepad before shooting to identify which pictures belong to which HDRI. When planning a trip, be aware of the size of the equipment. The Manfrotto tripod and head are heavy, so moving to a lot of locations without a car is difficult. Nevertheless, such heavy devices are very stable in case of wind (stability allows easier HDRI reconstruction) and important for low shutter speed. At Unity we decided to deal with this restriction.

3.1.1 Fisheye lens

In the equipment list there is a fisheye lens. This type of lens has a 180° field of view (fov) and produces an hemispherical distorted image:

Figure 26: Example of fisheye lens shot.

In fisheye lenses the geometry of image formation and image photometry limits the field of view of a distortion-free lens to about 120°. From 120° to 180° there is barrel distortion. Figure 27 shows a Sigma 8mm Circular Fisheye lens similar to what is used at Unity to capture HDRIs.

Figure 27: Sigma 8mm f/3.5 EX DG Circular Fisheye optical lenses. Source [Phob].

3.1.2 Neutral density filters

The equipment list includes neutral density (ND) filter [Wikf]. An ND filter reduces the amount of light entering the lens by reducing the intensity of all wavelengths of light equally, giving no changes in hue or color rendition. Applying an appropriate ND filter is equivalent to stepping down one or more additional EVs. See Figure 28.

NDnumber notation	EV reduction	Transmittance
	0	100%
ND2	1	50%
ND4	2	25%
ND8	3	12.5%
ND16	4	6.25%
ND32	5	3.125%
ND64	6	1.563%
ND100	6.64	1%
ND128	7	0.781%
ND256	8	0.391%
ND400	8.64	0.25%
ND512	9	0.195%
ND1024 (also called ND1000)	10	0.098%
ND2048	11	0.049%
ND4096	12	0.024%
ND8192	13	0.012%

Table 2: Various ND filter properties. Source [Wikf].

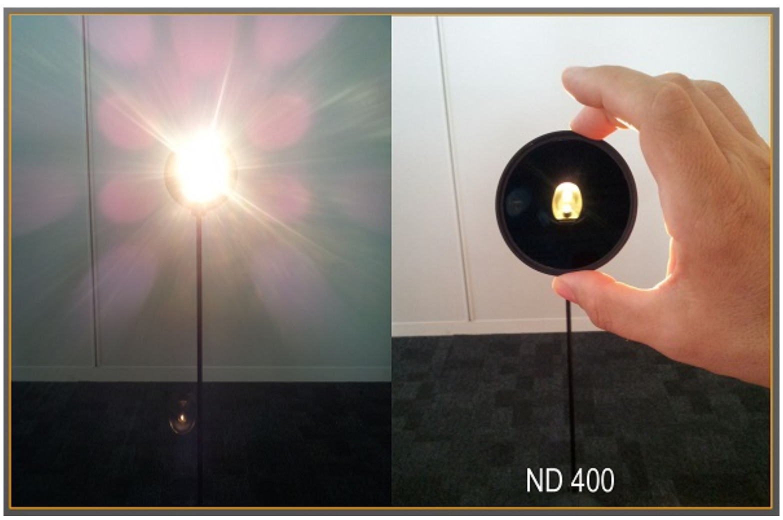

Figure 28: Example of ND400 Filter.

The strength of an ND filter depends on its NDnumber notation, which is mostly a power of two. Decreasing light intensity by 1 EV requires doubling the NDnumber[14]. Table 2 gathers information from various ND filters.
An ND filter cannot be mounted directly on a fisheye lens due to its curvature. Fisheye lenses require a *filter support* to do so (that will crop the field of view). It is alawys provided with the fisheye lens.

ND filters can be mounted on top of each other. In this case, their EV reduction is additive: An ND32 on top of an ND1000 produces an EV reduction of 15. We recommend having various power of 2 ND filters to be able to deal with a variety of high intensities.

[14]To retrieve the number of EVs reduced from using ND filter of a given NDnumber, use $EV = \frac{\log(\text{NDnumber})}{\log(2)} = $ log2(NDnumber).

3.2 Software

There is a lot of software available to work with HDRIs. Some perform an all-in-one operation, others are dedicated to single steps in the HDRI process. More software implies a more complicated workflow (and can imply more cost), but can increase accuracy[15]. At Unity, we have tried to get a good balance between accuracy, amount of software involved, cost, and ease of use by artists. We built our workflow around two popular software packages used by game artists and two unknown ones for specific tasks:

- `dcraw` [Cof]: Convert RAW to TIFF.
- ViewNX 2 [Nik]: Axial chromatic aberration correction.
- Adobe Photoshop CC [Adob]: Lateral chromatic aberration. correction, HDRI calibration.
- PTGui [PTG]: Create HDR panorama - PTGui Pro is required.

Adobe Photoshop is already widely used by artists and should not involve any extra cost, `dcraw` and ViewNX 2 are free, so only a moderate fee is needed to pay for PTGui Pro.

[15] During our study, we have found that several software packages aim at producing pretty results rather than accurate ones, we believe this is because this software targets the photography industry.

4 Panoramic HDRI acquisition

4.1 What to capture?

Captured HDRIs are mainly used to author an asset in the context of look development[16]. It is not necessary that the HDRI looks beautiful, but it should be as close as possible to the game environment and create interesting lighting conditions. Typical examples are the Star Wars games that have clearly identified environments (sequoia forest, sand desert, snow desert, etc.) where the best option is to capture the HDRI at the same location used for the movie. See Figure 29.

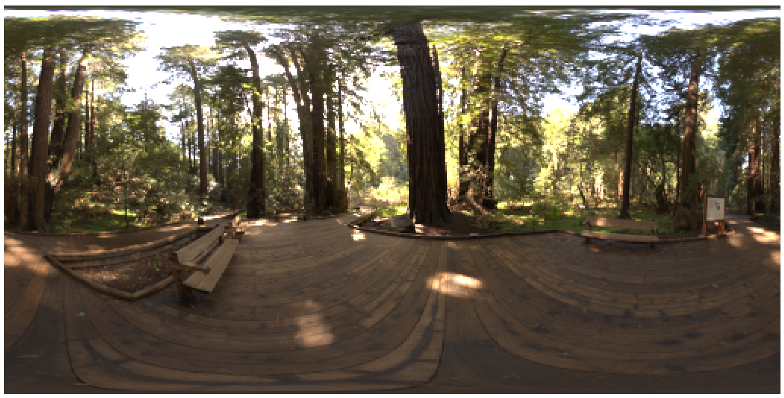

Figure 29: Example of an HDRI that can be used for asset authoring in the context of a game level located on Endor in the movie *Star Wars: Return of the Jedi*.

Artists tend to author their assets with an HDRI that includes a bright light source like the sun because it creates good highlights. They later use other HDRIs with different lighting conditions (overcast sky, night, dramatic, indoor, etc.) to check the correct behavior of the asset.

4.2 Equipment setup

The following section describes how we setup the equipment to capture the HDRI at Unity. The tripod and nodal head are mandatory as various shots need to be perfectly aligned for a sharper final image at potentially low shutter speeds.

4.2.1 Tripod and nodal head

First, install the Nadir Adapter on the nodal head. Then mount the camera on the tripod with the nodal head. The camera should be at around eye level. With this setting, the 360° panorama looks as if you were in the scene.

4.2.2 Camera alignment with nodal head

The nodal head gives 360° freedom of movement. To achieve perfect stitching without wasting time on the computer for the alignment task, you should take the time to level and align the camera to find the no-parallax pivot point.

- Level the tripod.
- Use the nodal head to put the lens center at the pivot point. Use a level to center the camera.

[16]Captured HDRI for lighting in games are not widely adopted due to the lack of control over weather conditions and their static nature, particularly in the context of games with dynamic time of day.

- The colored ring on the lens is the center.
- Attach the CamRanger on the back side of the head to follow the camera rotation.

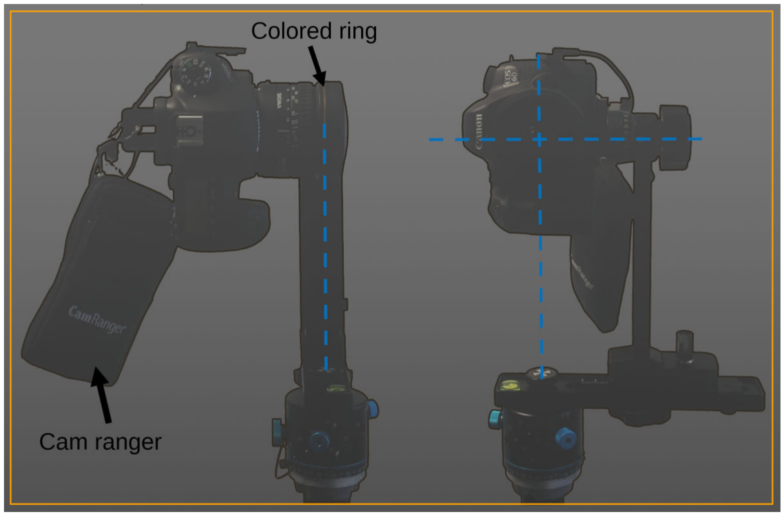

Figure 30: Camera alignment.

4.2.3 Remote camera controller

A remote camera control such as CamRanger is required to reduce camera shaking during acquisitions and therefore prevent blurry images. Use the CamRanger to program all the shots to perform in advance, based on the range and step information provided in the following sections.

4.2.4 Lens cleaning

Before any shot, the lens must be cleaned as dust on the lens produces a lot of artifacts.

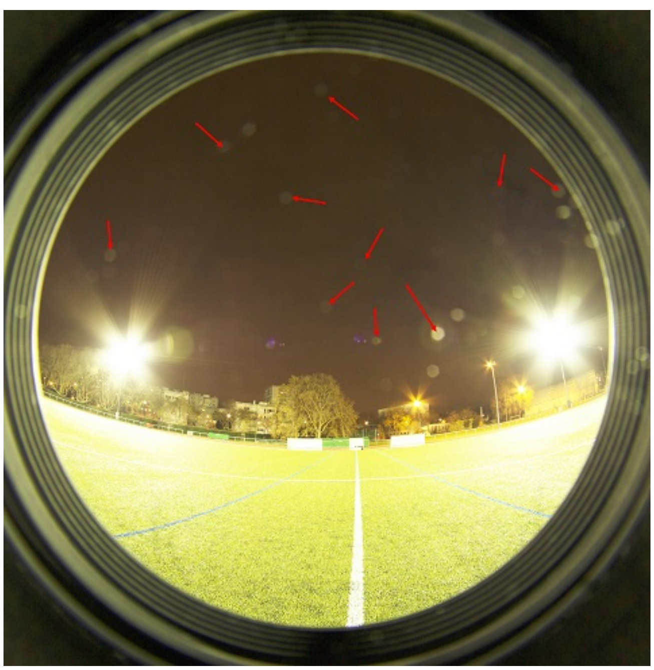

Figure 31: Example of dirty lens.

4.2.5 Camera orientation and number of views

To build a 360° panorama with a fisheye lens, we use the following settings:

- 3 views (horizontal 0°, 120°, 240°) at +30° vertical
- 3 views (horizontal 60°, 180°, 300°) at −30° vertical

We found that this setting gives fast and good quality results in practice. Note that 30° is easy to set up with the Ninja head. With an angle of +30°, the sky/top side is completely captured. With an angle of -30°, the tripod is barely visible.

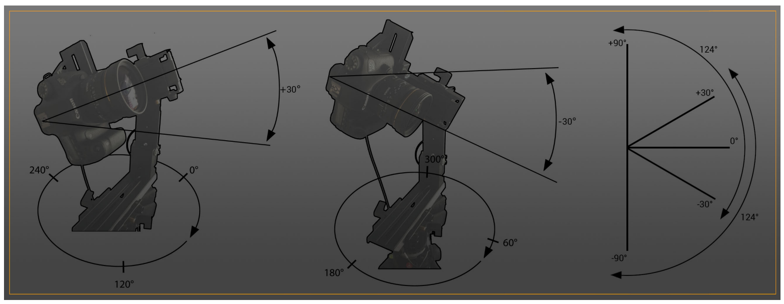

Figure 32: Illustration of the six views use for panorama.

Choosing 30° instead of 45° gives more detail at the horizon as artifacts increase proportionally to the distance to lens center. The horizon often contains a lot of detail that helps to match overlapping shots and it often includes moving objects that need to be detected correctly to be removed. Moreover, with 30° and a field of view of 124° it covers the range −32° to 92°, so it correctly includes the whole sky.

Note: the fisheye lens has a field of view of 180° but distortion, vignetting and chromatic aberration appears at the edge of the lens (> 120°). It is better to use more views with some of them overlapping, in order to discard the faulty region during postprocessing. The stitching also needs overlapped areas to match images together. Lastly, the ND filter used to capture very bright light sources (see section 4.4.1) causes the effective field of view to be 124° (40° for the support, plus 16° for the ND Filter) and not 180°[17].

To have a snap during the rotation for faster shooting, we set the head to a 60° rotation step.

Figure 33: Nodel head angle settings.

[17] We use PTGui and the crop tool to get the effective angle. Note that the nodal head support will be visible in the view anyway, meaning that a full 180° field of view is never possible.

An extra bottom view is captured to remove the tripod entirely:

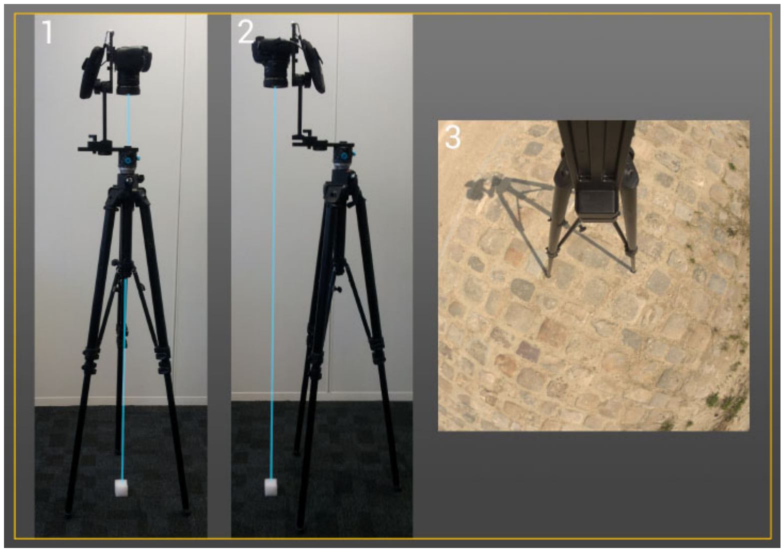

Figure 34: Bottom view.

- Put a marker under the tripod at the center point. (1)
- Place the camera at the bottom orientation. (1)
- Turn the Nadir Adapter to 180°. (2)
- Move the tripod until the focus is on the marker and ensure that the tripod feet do not cover the same places. (2)
- Lock the head to avoid movement during the bottom shot.
- Remove the marker.
- Do the bottom view. (3)

To sum up, we use seven views to cover the 360° panorama.

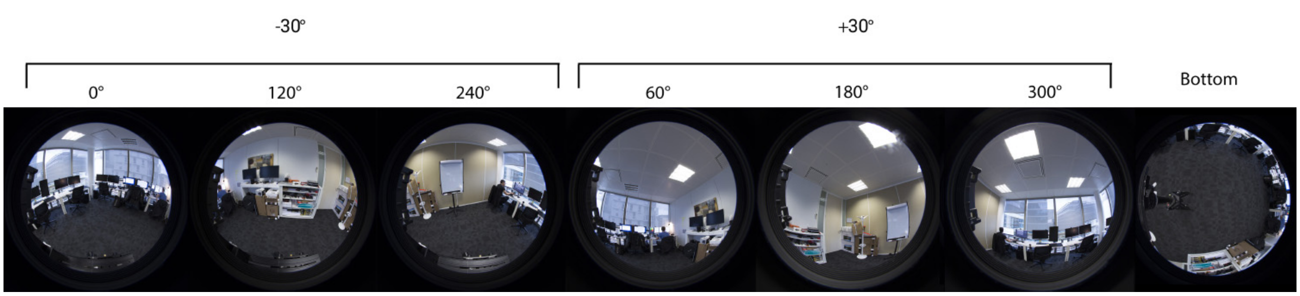

Figure 35: The seven views for panorama plus tripod removal.

4.3 Camera settings

An HDRI is a combination of shots taken with different exposures (brackets). HDRI acquisition requires taking several shots of varying exposure of each view. The exposure is based on three parameters:

shutter speed (t), aperture (N) and ISO (S). A given combination of these parameters can be summarized as an *exposure value* (EV). An EV is, by convention, defined for ISO 100 and denoted EV_{100}. Thus, we have the following relationship:

$$EV_{100} = \log_2(\frac{N^2}{t}) + \log_2(\frac{S}{100}). \tag{2}$$

Unity's HDRIs are acquired with fixed aperture and ISO values, and varying shutter speeds to control the exposure. The following section covers the values used for these three parameters.

Remember to:

- Setup camera to use RAW format.
- Place camera in manual mode.
- Disable automatic white balancing and set the white balance to 6500K. RAW files are not white balanced and we do not apply any white balancing to them in our current workflow, but we use the setting to make sure that camera previews all use the same white balancing.
- Disable all color and contrast enhancing features.

4.3.1 Aperture and Focus

Capturing an accurate HDR panorama requires us to have the best sharpness (without optical artifacts) for both near and far elements of the scene.

Small apertures (bigger f-stop values) cause diffraction [col], produce softer results and significant flare. Wide apertures (smaller f-stop values) can cause significant vignetting, shallow depth of field, and is more sensitive to optical aberrations.

Choosing the best aperture is a balance between all of these artifacts.

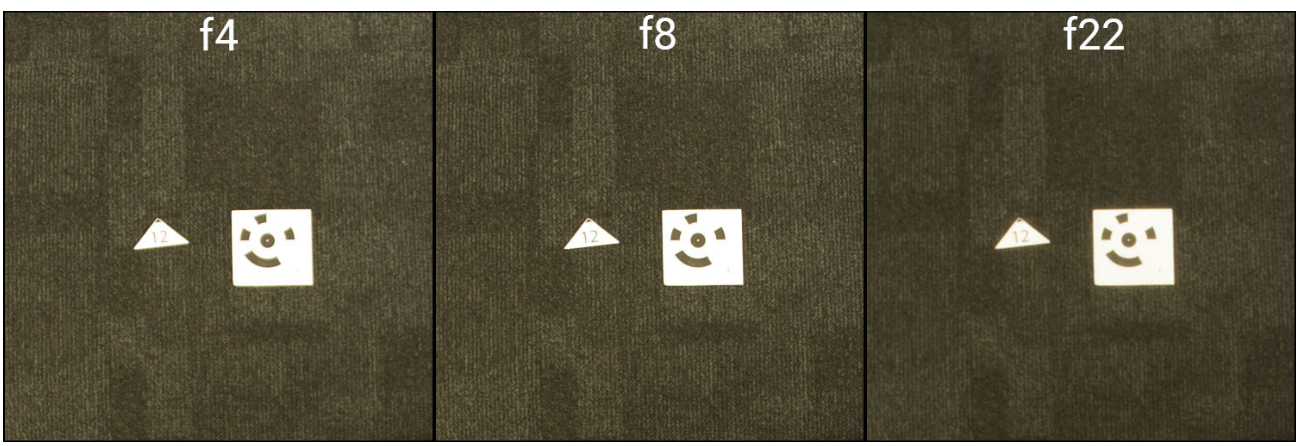

Figure 36: A close up of three shots with different aperture. Left: Softer look from small aperture. Middle: Sharp look from medium aperture. Right: Blurry look and veiling glare from wide aperture.

Figure 37 shows different shots with a pair of aperture/focus settings for both near and far scenes.

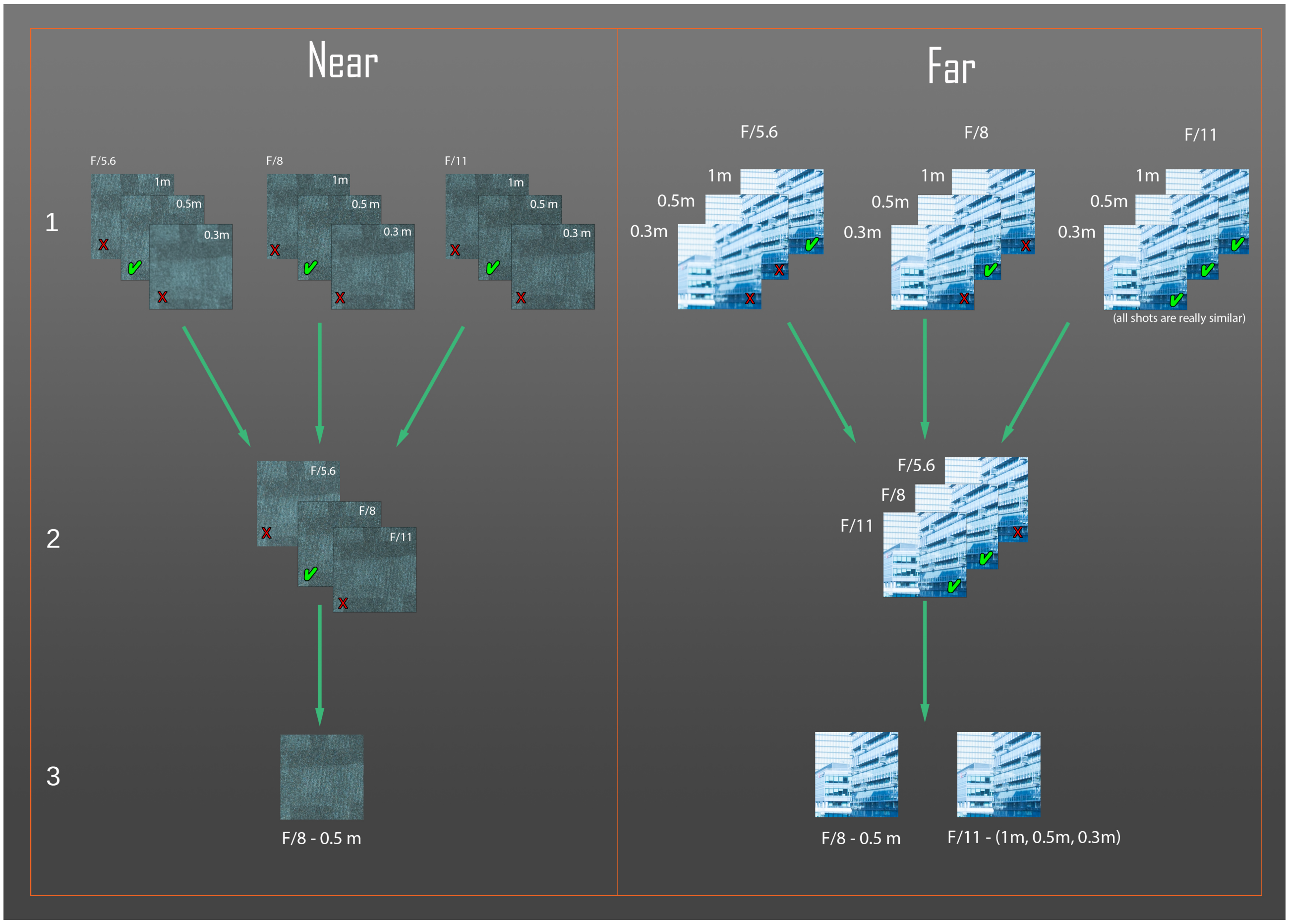

Figure 37: Aperture settings experiment. Tests show that f/8 is a good setting.

1. Focus distance test: the test is done at three efficient focus distances. For the near setting, the test is performed on the ground, which is normally the closest element to the camera.
2. Aperture test: all apertures are compared with their valid focus.
3. Results: f/8 at 0.5m seems to be the best choice for a panorama. f/11 can also give good results without precise manipulations.

4.3.2 ISO

Changing the camera's ISO setting modifies the sensitivity of the light sensor. Lower numbers mean the sensor is less sensitive. This setting is useful for shooting in different lighting conditions, particularly in dark environments. A higher ISO can be used when needed to shoot faster in case of moving environments, like clouds in the wind. If the ISO is doubled, the shutter speed must be divided by 2 to get a similar exposure. As explained in Section 2.4, the ISO controls the gain amplification in the CMOS, which means that higher ISO values lead to noisier images.

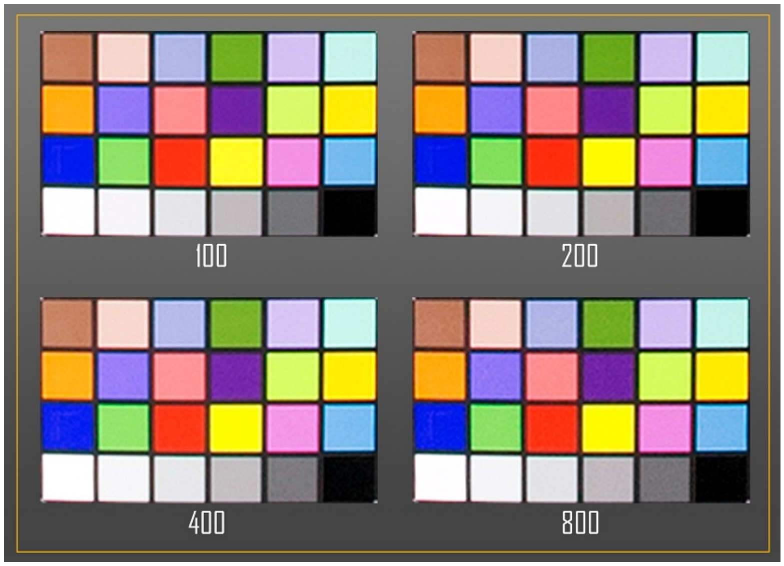

Figure 38: Noise generated by ISO settings.

Whenever possible, always set ISO to 100 to minimize noise. If at the maximum shutter speed (30s on Canon 6D) the darkest area is not covered, it is necessary to increase the ISO to capture the full range. Also, in some specific cases where shooting speed is crucial—a sunrise, for instance—we find it useful to increase ISO despite the resulting noise.

4.3.3 Shutter speed

The shutter speed controls the amount of time that the sensor is exposed to light. Each time the shutter speed is doubled, the quantity of light on the sensor is doubled.

This parameter is the only varying parameter for panoramic shooting and is thus used to control the exposure of the various shots. Several brackets are necessary to capture the dynamic range (i.e. contrast) of the scene in order to correctly reconstruct the HDRI. Finding the full range depends on the scene's context (indoor, outdoor, day, night, ...) and weather conditions (sunny, cloudy, ...). We recommended determining the range before each scene acquisition to avoid wasting time on unnecessary shots or useless results (e.g. clipped lighting). To get the full range of the scene, we use the preview mode:

- Fix ISO and aperture with the values provided in the previous section—we usually use ISO 100 and aperture f/8 with a focal distance of 0.5m.
- Orient the camera so its direction is parallel to the ground, turn on preview mode with the histogram activated and rotate the camera on the tripod to cover the whole panorama. During the process, identify the darkest and brightest areas of the scene.
- Use the main dial to adjust the shutter speed in the darkest and brightest areas until the histogram is not clamped, see Figure 39. The brightest areas are the most difficult to capture completely. The histogram can look correct but the highlights are still bloomed. To address this, we increase the shutter speed until bloom completely disappears around the bright region.

Figure 39: Camera preview histogram.

1. Preview button
2. Dark areas not clamped
3. Bright areas not clamped
4. Highlight checking

- The previous information provides the set of shutter speed values to use and the EV range of the scene. See Section 4.3.5.

Note: the start value in the CamRanger application must be set to *shadows point*[18] (lower exposure).

This range determination process is valid for overcast skies or interiors. On sunny days or in the presence of very bright light sources, the camera cannot capture high ranges without an ND filter. We describe the specific setup and how to determine ranges for capturing very bright light sources in Section 4.4.

Image clipping detection with histograms

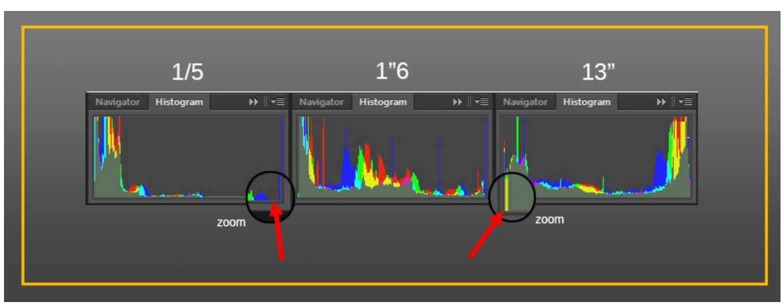

Figure 40: Histogram. Left: Bright area, Middle: Gray area, Right: Dark area.

The first histogram (1/5) does not clamp the brightest values. The last histogram (13”) does not clamp the darkest values (1 pixel empty from the left). It is important to note that with a fisheye lens, the dark part of the histogram will exhibit a weird pattern due to the circular fisheye format. This is demonstrated with the white scene test below.

White scene test

[18] CamRanger also has an option for *middle point* or *highlight point* (higher exposure).

Photo of a fully white scene:

Figure 41: White scene histogram with fisheye lens dark areas.

In a fully white scene, we expect the histogram to increase at the rightmost part. Here, the histogram is different because the fisheye format produces dark areas.

4.3.4 Constant EV step

An EV is expressed in a base-2 logarithmic scale, one positive step corresponds to a factor of two in luminance (twice the brightness), one negative step corresponds to a factor of a half (half the brightness). This scale is almost perceptually linear, meaning that the difference between +0 and +1 EV looks almost the same as in between +1 and +2 EV. Constant EV step increments/decrements provide the best result when reconstructing an HDRI. The more steps are used, the longer the shooting takes.

Speed can be a key factor in the case of a scene where the lighting changes rapidly, like during a sunset or when some desired elements are moving. In this case, a large increment of 3 EVs can be necessary. However, using large increments minimizes the information available for the reconstruction process and can affect quality. The smaller the EV step, the lower the noise in the HDRI.

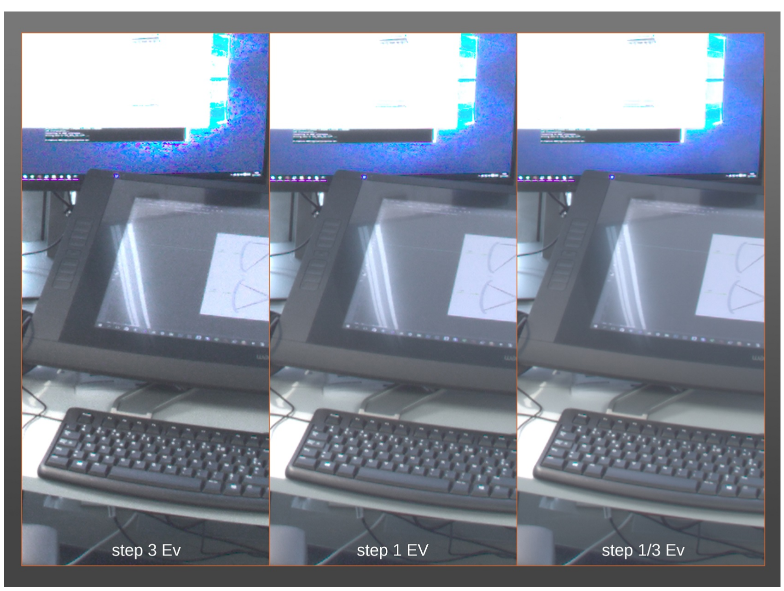

Figure 42: HDRI reconstruction for different EV steps.

More EV steps also reduces ghosting artifacts. With large numbers of shots, moving people can be completely removed from the HDRI.

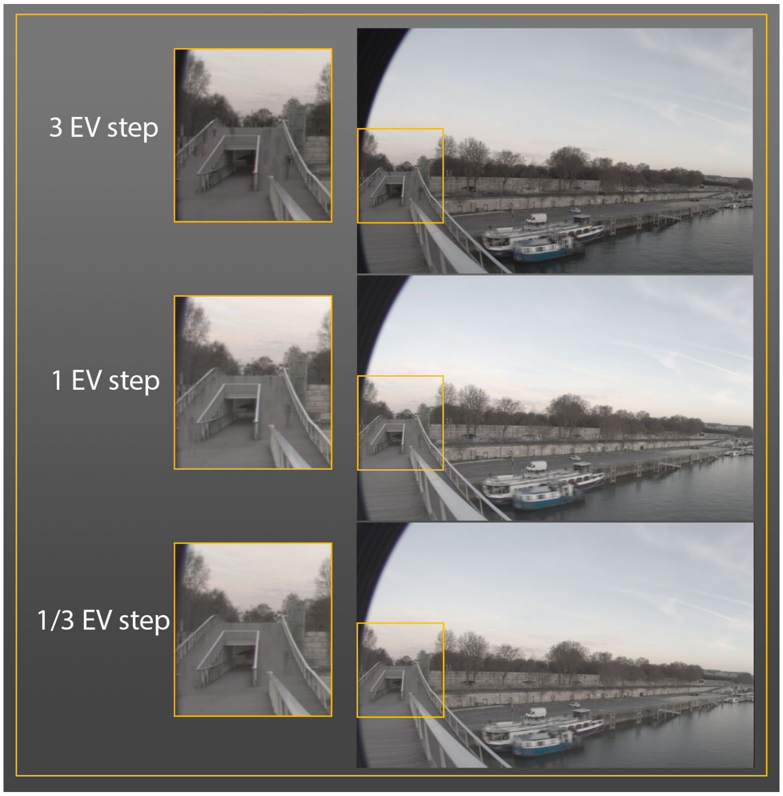

Figure 43: Impact of EV step on ghosting. Smaller EV step minimizes ghosting.

Overlapping also helps to minimize ghosting effects because it can give reference to the same space from different angles. In the scene below, people who are walking in front of the camera during the shooting are removed completely after stitching.

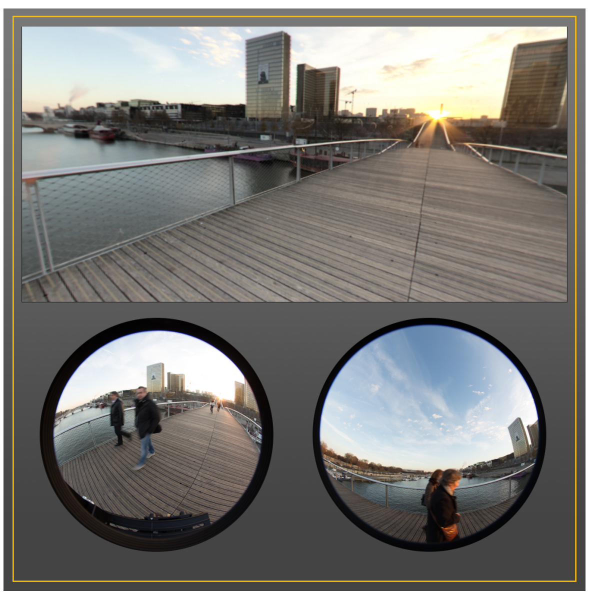

Figure 44: Example of pedestrian removal.

Of course, safe scenes with a minimum of movement (no vehicles, no people, no wind) will provide the best results.

To sum up, the number of shots to take (based on the EV step) depends on the quality expected and the time allowed to produce the HDRI. The Unity team decided to use 1 EV step. In the case of a wide light range (sun capture or other intense lighting), 1/3 EV generates a lot of data that is difficult to manage (for example, consider the storage and processing time).

4.3.5 Number of brackets and set of shutter speed values

The EV range is defined by the difference between the shutter speed to capture the brightest and darkest areas. Every time the shutter speed is doubled, an EV is added. The number of brackets to do by view is

$$\mathrm{ceil}\left(\frac{EV range}{EV step} + 1\right). \tag{3}$$

At Unity, we use Table 3 to easily find the number of brackets to do and with which shutter speed values[19].

1/4000	1/2000	1/1000	1/500	1/250	1/125	1/60	1/30	1/15	1/8	1/4	1/2	1	2	4	8	15	30

Table 3: Reference table providing shutter speed settings for various EV ranges.

Check the table for the longest and shortest shutter speed values needed based on the bright and dark areas seen before. The number of columns in between the two values is the number of shots to do. Each value between the two values is the shutter speed to use for the bracket.
Here is an example for a scene with brightest value not clamped at 1/8" and darkest value not clamped at 8". Fixed camera settings is ISO 100 and aperture f8. And Step of 3 EVs.

- Between 1/8" and 8", there is a difference of 6 EVs (7 different settings in the table): 1/8" - 1/4" - 1/2" - 1" - 2" - 4"- 8".
- The number of shots to take is (6 / 3) + 1 = 3. With 1/8", 1", and 8".
- In CamRanger, define the shadow point to 1/8", step to 3 EV and number of shots to 3.

This will produce the following brackets:

[19] Table 3 presents the agreed standards for shutter speeds [Wikj]. These values are close to powers of two. Using near power of two values is simpler (as opposed to using, for example, steps of 2/3 shutter speeds) and simplifies our usage of power of two ND filters.

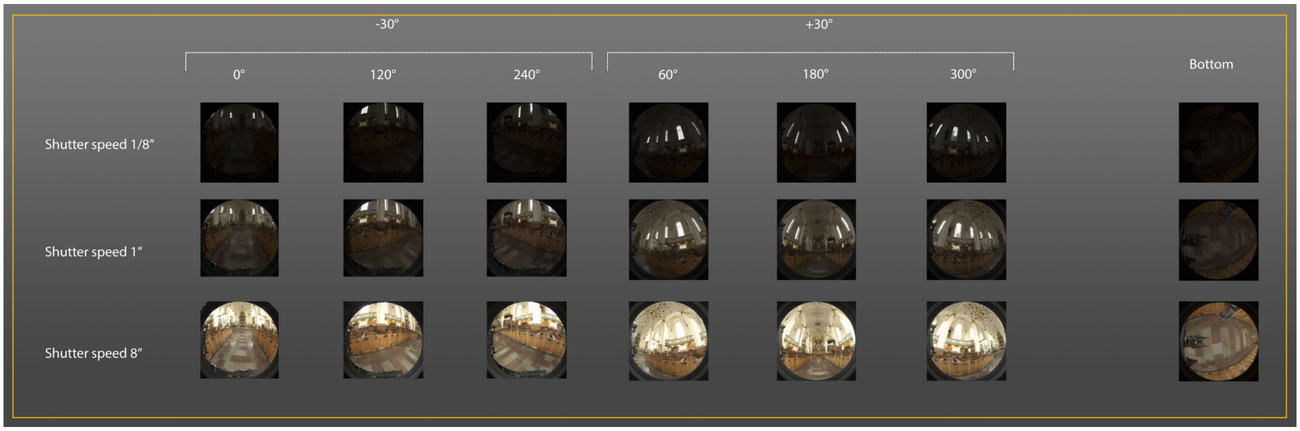

Figure 45: Example of all views required for a panoramic HDRI.

4.4 Sun and other bright light sources

4.4.1 Dealing with high intensity

As explained in Section 4.3.3, creating an HDRI requires taking shots at various exposures, and shutter speed is the only varying parameter in Unity's HDRI capture process to control this exposure. Note however that shutter speed control is bounded. On a Canon 6D for example, shutter speed can vary from 30s to 1/4000". For scenes with very bright light sources such as spot lights in a football stadium or the sun, which can emit up to 30 EV, such settings are insufficient and the intensity of the lights will be clipped as well as highlights on objects.

Figure 46: Panoramic HDRI with multiple very intense light sources with various lens artifacts and dirt due to high intensity.

In order to capture such high intensity values without clipping, it is required to use ND filters (See Section 3.1.2). Complex automated setups can be used to capture very bright light sources. [Stu+04] uses a scripted camera with ND1000 and variations of aperture and shutter speeds to capture the sun. At Unity, we wanted to promote artist friendly solutions and chose to incorporate a manual step in our HDRI capture process. This section describes the process to get full range of a very bright light source with ND filters.

Estimate which filter is needed

- Face the camera towards the very bright light source. The light should be in the middle of the view and it will be the first view position (0°) of the Unity capture process described in Section 4.2.5.

Use the zoom button to fit the brightest area in preview mode. See left of Figure 47.

- Set the shutter speed to the minimum allowed (1/4000" for Canon 6D). Install one or more ND filters in front of the lens (from lower to higher EV reduction) until the light source disappear. Then decrease the shutter speed until the range limit of the scene is found, like in the regular process. The histogram is not always helpful for this step as the bright source area can be really small, so simply control the preview until the bright light source disappears. See Right of Figure 47.
- When shooting a clear sky with sun scene ND1000+ND500 is the default ND filters choice.

Figure 47: Left: Camera facing the light source. Right: Different filters are installed until the bright light source disappears.

It is important to use as few ND filters as possible as they reduce the field of view available to the camera:

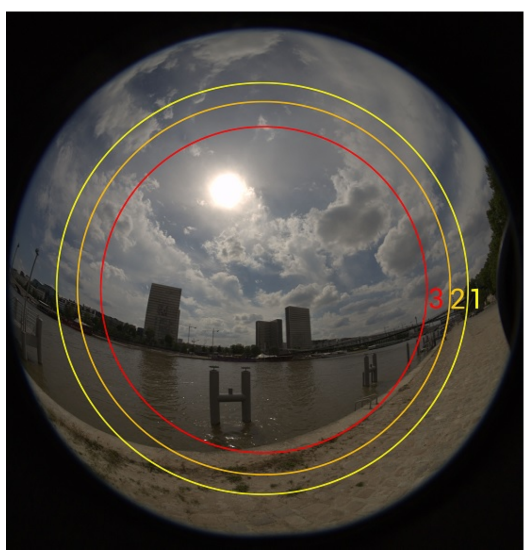

Figure 48: 1) Filter support crops at 139°, 2) One ND filter crops at 124°, 3) Two ND filters crop at 110°.

EV	Virtual	No filter	ND16	ND32	ND64	ND500	ND1000	ND500 + ND1000
0	1/4096	1/4000	1/250	1/125	1/60	1/8	1/4	
-1	1/8192		1/500	1/250	1/125	1/15	1/8	
-2	1/16384		1/1000	1/500	1/250	1/30	1/15	
-3	1/32768		1/2000	1/1000	1/500	1/60	1/30	
-4	1/65536		1/4000	1/2000	1/1000	1/125	1/60	
-5	1/131072			1/4000	1/2000	1/250	1/125	
-6	1/262144				1/4000	1/500	1/250	
-7	1/524288					1/1000	1/500	
-8	1/1048576					1/2000	1/1000	
-9	1/2097152					1/4000	1/2000	
-10	1/4194304						1/4000	1/8
-11	1/8388608							1/15
-12	1/16777216							1/30
-13	1/33554432							1/60
-14	1/67108864							1/125
-15	1/134217728							1/250
-16	1/268435456							1/500
-17	1/536870912							1/1000
-18	1/1073741824							1/2000
-19	1/2147483648							1/4000

Table 4: Reference table providing shutter speed values to use when using ND filter.

Determine EV range, number of shots and start shutter speed

ND filters allow us to emulate a decrease of EV without changing the camera settings. It can be helpful to think about it as if we were able to decrease the shutter speed below the camera's limit, generating a *virtual* shutter speed value[20]. This *virtual* shutter speed is shown on the *Virtual* column of Table 4.

The shooting is split in two parts. The first part is performed with the ND filters for all *virtual* shutter speed below the camera's shutter speed limit. The second part is done with the usual process.

The number of brackets with ND filter to capture can be retrieve in Table 4 by following a similar process than with Table 3. The table list of the ND filter use at Unity. From previous step (Estimate which filter is needed) we have the shutter speed that clip the bright light for a given ND filter. The number of row between row EV-1 and this bright clip' shutter speed is the number of brackets to perform with which shutter speed. EV 0 is redundant with the 1/4000" shutter speed without ND filters and is not required. If ever equivalent exposures are captured due to the CamRanger software, they should be deleted. Brackets without ND filters are preferred because they are less cropped. Non power of 2 ND filters (ND100, ND400) should be avoided to simplify the capture process.
The EV range of the scene is the number of brackets with ND filters added to the number of brackets without minus one.

Example: An ND32 filter with shutter speed 1/4000 is needed to capture the bright light source. The dark area is not clip with 1/8 shutter speed without ND filter. We use 1 EV constant step. Looking

[20] Instead of shutter speed, it could have been *virtual* ISO or aperture. Only the final exposure (EV) matters. It is important to understand that even if a parameter is virtually emulated, it does not affect the result as a regular parameter. Changing ISO 100 to a virtual ISO 10 does not reduce the resulting noise for example.

into Table 3 and Table 4:

- 5 brackets with an ND32 filter will be captured with shutter speeds: 1/4000", 1/2000", 1/1000", 1/500", 1/250".
- 10 brackets without ND filter will be capture with shutter speeds: 1/4000", 1/2000", 1/1000", 1/500", 1/250", 1/125", 1/60", 1/30", 1/15", 1/8".
- Scene EV range is 14 (15 bracket - 1).

The sun without clouds can require up to EV -19, thus the limit of the reference table. To capture the sun an ND1000000 filter can be used, but such a filter is not easy to find. Moreover, a long exposure time can be a problem in case of such intense lighting because of movement (for example, clouds). It is best to do more series with few ND filters in order to use faster shutter speed settings. This is the case highlighted with ND1000 + ND500 (19 EV reduction) in the table. The first series of brackets are captured with both filters from 1/4000" to 1/15", then a second series with ND1000 only is captured from 1/4000" to 1/8", and lastly the usual process is performed.

In this case, it is very important to place the sun at the center of the view at 0°. As shown in Figure 48, two ND filters crop the image up to 110°. This is a problem because three views do not add up to 360°. However, in the range capture with two filters (-19 to -11 EV) only the sun is visible, the rest of the image (not the sun) is black so there is no impact to cropping at 124° instead of 110° when performing the stitching. An alternative could be to capture four views instead of three in case of highlights in the missing area.

Shooting session

For each of the seven views of our HDRI capture, there are two (or three) brackets series to capture with: one with and one without the ND filter. It is simpler to do the same numbers of brackets by series to ease the stitching step. Where brackets with the ND filter are totally black, they can be replaced by black images and shooting time can be saved. Often the bottom view does not need brackets with ND filters. It is however not recommended for other views, because some low lighting could be lost.

For a view, begin by the series of bracket with the ND filters, then remove the support where the ND filters are screwed. The following figure shows an example of such a bracket series.

Figure 49: Example of brackets captured for a view with sun.

4.4.2 Dealing with lens flare and veiling glare

Lens flare can be problematic for very bright light sources as it covers a large part of the scene and can change the luminance of surrounding pixels (see Section 2.3 for more details about this artifact). They are mainly visible with a contrasted background. In the case of the sun, the condition of the atmosphere and the surrounding cloud tends to minimize the problem, but with a clear sky the problem remains:

Figure 50: Examples of lens flare. Right: Spotlight in a stadium. Middle: Sun with cloud. Right: Sun in clear sky.

One way to minimize the problem is to use a wide aperture:

Figure 51: Lens flare artifacts for increasing aperture size.

At Unity, we chose to use a fixed aperture of f8 so it was not an option.

Alternatively, we tried to hide the very bright light source during the capture to remove lens flare. Following is an indoors experiment where a piece of cardboard matching the light's shape is attached to a wire cable at a distance that corresponds to the light's solid angle:

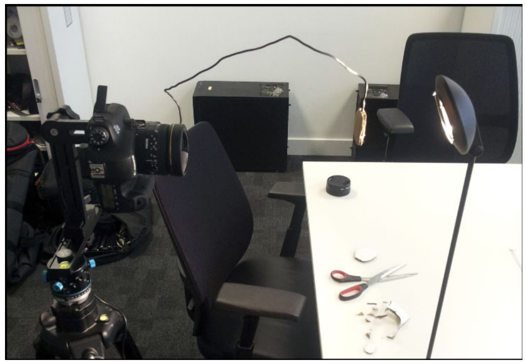

Figure 52: Wire and cardboard used to hide a very bright light source.

After both regular and cardboard HDRIs are reconstructed, they are composed in Adobe Photoshop with the cardboard mask:

Figure 53: Example of hiding a very bright light source and compositing.

This method works but makes the workflow a lot more complex and is not suitable for multiple light sources such as the stage in the following HDRI:

Figure 54: An HDRI with lots of lens flare artifacts after reconstruction.

This method can be applied to the sun in a clear sky. The cardboard is more difficult to setup correctly in this case due to the wind and distance to the sun.

Figure 55: Example of sun flare hiding.

Due to the extra complexity, we chose not to correct lens flares except in rare cases.

Veiling glare with contrasted scenes and very bright light sources can affect the luminance of a lot of pixels in the image (see Section 2.3 for more details about this artifact). But in practice it is uncommon for an HDRI. [Tal+07] provided a complex method to remove veiling glare but it is only practical in case of studio lighting. At Unity, we chose to ignore this artifact as it is unnoticeable in captured HDRIs.

4.5 Acquired calibration data

In order to calibrate an HDRI for both chrominance (white balance) and luminance as explained in Section 2.7, it is necessary to acquire some extra information.

4.5.1 White balance data

A consumer camera is designed to mimic the process of the human eye, i.e., it will provide a white balanced RGB image to look like the captured real world scene. In the field of regular photography, it is common to calibrate a series of shots under the same lighting conditions with a first shot including a ColorChecker. This makes sense because in this case the photographer wants to capture what is observed. But white balancing in a simple photograph may be difficult in the presence of high frequency lighting with multiple light sources and extreme contrasts. In the context of a panoramic HDRIs, this is almost always the case.

With a 360° capture all lighting conditions are spanned. Thus it is not possible to place a ColorChecker at a particular location with the same lighting conditions. Put the ColorChecker in shadow? Put it in sun light? In between? If the ColorChecker is setup for each view, then we get a color shift during the HDRI reconstruction. The correct way to handle white balancing for HDRIs will be to do an automatic white balance at runtime as part of the game's postprocessing.

Automatic white balancing is not the subject of this document and we chose to still perform some kind of white balance for panoramic HDRI here even if it makes little sense. In order to perform white balancing, additional data should be acquired.

To acquire white balanced data, during the acquisition process after the bottom view, recapture the bottom view (for all bracketed exposures) but this time with an Xrite chart: either the white balance chart, ColorChecker or ColorChecker Passport. We chose a single white patch calibration rather than the multiple patch calibration of ColorChecker as presented in [McA12] to reduce workflow complexity.

The Xrite white balance chart is recommended compared to the ColorChecker for white balancing because it is more neutral, and tends to behave better in various lighting conditions [Mye09] (the neutral 8 patch of the ColorChecker approximates the white balance chart). The following figure shows the spectrum of both targets:

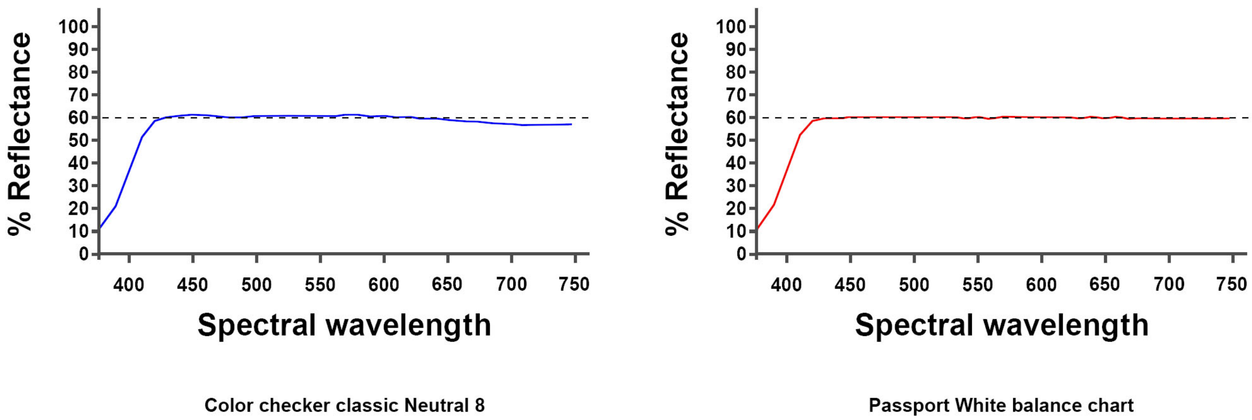

Figure 56: Spectrum of the white patch of the ColorChecker and the white target. Source [Mye09].

As explained in the introduction of this section, there is no correct location to put the white balance chart. Setting the chart on the bottom view is an option to average all the lighting sources. Try to have it perpendicular to the lens and avoid tilting it. To avoid occlusion from the tripod, the chart is set at 1 meter from the camera.

Figure 57: Example of ColorCheckers. 1: ColorChecker. 2: Passport white balance. 3: Calibration data placement.

4.5.2 Luminance calibration data

In order to reconstruct absolute luminance HDRIs (i.e., with real world luminance), it is necessary to take some light measurements at the location where the HDRI has been acquired. There are three kind of devices to measure such light:

- Lux meters which measure illuminance.
- Spot meters which measure EV.
- Luminance meters which measure luminance.

With the measurement in the captured scene and the one retrieved from the reconstructed HDRI, it is possible to make them match and recover an absolute HDRI [She10] [Cha15]. A lux meter is a cheap and accessible device, which we have decided to use for our artist friendly workflow. However, we still use a luminance meter, which is a much more expensive device, as an optional validation for the lux meter method.

Measuring lighting for enclosed indoor environments only requires a single measurement, the lighting is usually fixed. However, for outdoor environments, lighting changes are frequent (clouds are moving, sun light varies etc.), so outdoor light measurements are challenging. To avoid adding too much complexity to the workflow, we decided to only perform two measurements in this case and average them, one at the beginning of the shooting session and one at the end. But keep in mind that in case of changing conditions, the HDRI itself is built with different lighting conditions for each view, so it is not possible to be fully accurate.

Lux meter

The lux meter[21] measures the illuminance of the scene (lighting coming from the upper hemisphere oriented in the sensor direction). Try to measure illuminance at the same position as the camera: the sensor should be as close as possible to the lens capture point. See left of Figure 58.

Caution: The sensor environment should be free from any occlusion - whether from the tripod, human body (crouched when taking measurements) and not in shadow. This should be representative of the captured environment.

Luminance meter

The luminance meter measures the luminance of a very small area in a scene. During HDRI acquisition, we measure areas of easily identifiable and uniform surface. Sheets of paper of various reflectance for example but it is convenient to use the white balance patch (in the case of the ColorChecker use the grayscale patch, since luminance measurements are more accurate on a grayscale area [Ina05], patches colored with high chroma are problematic [Var14]). For Unity HDRIs, all measurements are taken on the ColorChecker Passport white balance target at the center point of the patch. The luminance meter pivot point when measuring the values must be as close as possible to the camera pivot point. Take care to measure at the center of the patch. The luminance can vary and the area averaged may be bigger than expected, so larger uniform areas in the visor is better. See right of Figure 58.

[21] A true lux meter must be used for the measurement, not the app for smart phones. The app for smart phones provides wrong values [Gol16].

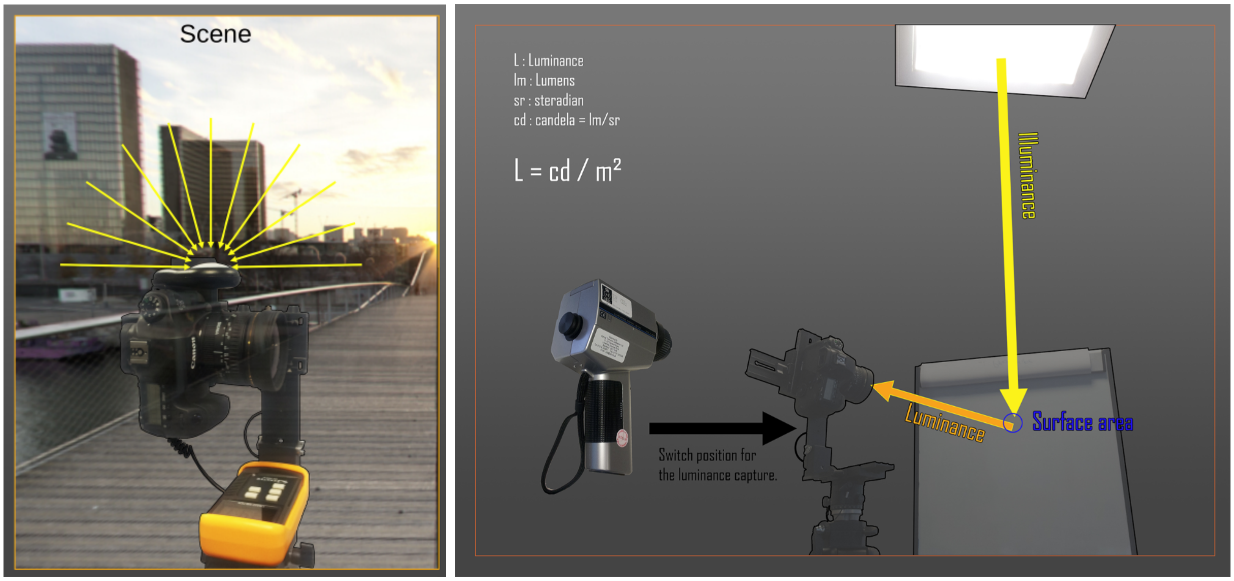

Figure 58: Two different light measurement at capture time. Left: Lux meter used to capture illuminance of a scene. Right: Luminance meter used to measure luminance of a surface area.

5 Linear reconstruction of HDRIs

The reconstruction of an HDRI includes the following steps:

- Extracting bracketed exposures from the camera in RAW format.
- Converting it to a workable TIFF format.
- Applying chromatic aberration and vignetting correction.
- Stitching bracketed exposures into an HDR panorama.
- White balance and luminance calibration, plus range validation.

5.1 RAW to linear RGB 16 bit TIFF

The theory for converting a RAW image to a digital image has been covered in depth in Section 2.6. A RAW file[22] contains camera sensor data, i.e., native RGB values in a Bayer mosaic pattern. To be manipulated, the RAW file must be converted to a *linear RGB* 16 bit (or 32 bit) file. In this document we call *linear RGB* the color space defined with sRGB primaries and the D65 white point but without any gamma correction.

`dcraw` is used to convert RAW files extracted from the camera into a linear 16 bit TIFF. PTGui can load a RAW file directly (PTGui uses `dcraw` internally, like most HDRI tools) but our team decided to perform some extra manipulation (like chromatic aberration and vignetting corrections) before sending the shots to PTGui. `dcraw` has many options: we use the following command lines in our workflow:

- `dcraw` options: -v -4 -T [file].
- -v: print verbose messages. To follow the conversion process.
- -4: Convert to linear 16 bit format to keep a 16 bit linear workflow. No gamma applied, no automatic brightness correction.
- -T: Write in TIFF instead of PPM. To have a compatible format for Photoshop and PTGUI.

These options generate a 16 bit TIFF in linear RGB. Having a good linear reconstruction is important to have correct HDR values, thus we prefer `dcraw` compared to other software such as Adobe Camera Raw [Adoa] due to its improved control and because it conserves linearity. Figure 59 and Figure 60 shows a test performed to evaluate the linearity of the reconstruction process between Adobe Camera Raw and `dcraw`. Various brackets of increased exposures of a ColorChecker have been captured. Each curve on the right represent the relative luminance values of increased exposure of an individual patch extracted from the RAW conversion.

[22] In the context of a Canon 6D the RAW format is CR2. CR2 is a proprietary format but some information can be extracted as it has been reverse engineered [lcl15].

Adobe Camera Raw linear test:

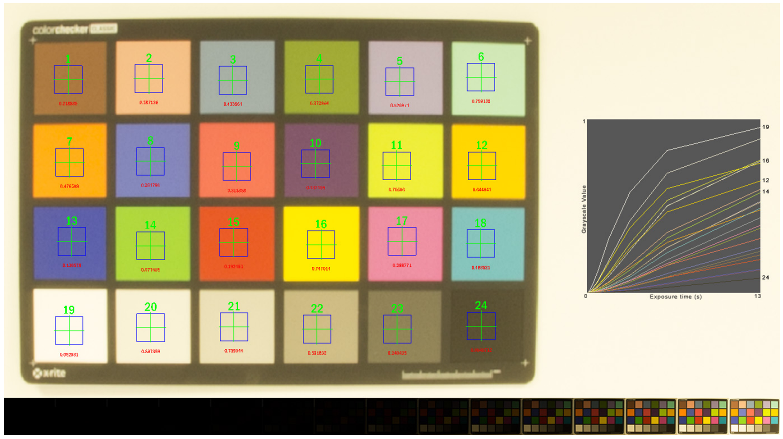

Figure 59: Adobe Camera Raw linear test.

`dcraw` linear test:

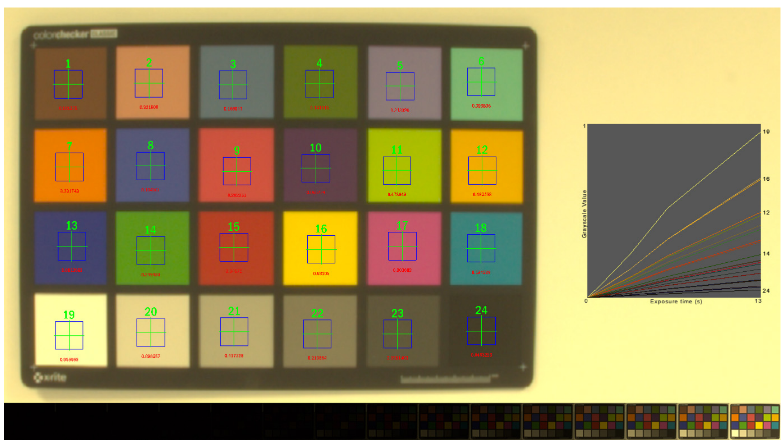

Figure 60: `dcraw` linear test.

Adobe Camera Raw seems to apply gamma correction during the conversion to TIFF (or other formats), and there is no option provided to prevent it. An experiment done by Mike Boers [Boe13] shows similar results. Moreover, Adobe Camera Raw adds some contrast enhancement by default, and some black-box color tweaks for aesthetic reasons.

5.2 Correcting image artifacts

Lenses produce artifacts as described in Section 2.3. A list of the major lens artifacts is presented in Table 5. For each artifact, the table describes if we have a solution for the artifact (described in this document), if it is applied in our process, and which software is used. This section will discuss the cases we correct.

Lens flare correction is described in Section 4.4.2. However, with the setup being too complex, particularly for multiple light sources, we chose to apply it only in the case of the sun in a clear sky.

Artifact name	Correction available	Correction applied	Software	Remark
Laterial CA	Yes	Yes	Adobe Photoshop	
Axial CA	Yes	Yes	ViewNX2	
Coma	No	No		Edge only
Field curvature	No	No		Edge only
Astigmatism	No	No		Edge only
Vignetting	Yes	No	Adobe Photoshop	No correction needed
Distortion	Yes	Yes	PTGui	
Veiling glare	No	No		
Lens flare	Yes	No		Setup too complex

Table 5: List of various lens artifacts. The table lists if a solution to correct an artifact is available, if this solution is applied in our process and which software is used for the correction.

5.2.1 Lateral chromatic aberration (CA)

Lateral CA must be corrected before axial CA for better results. The lateral CA can be minimized with a simple RGB scaling. Use the Photoshop lens correction tools (Shift+CTRL+R) from Adobe Photoshop to correct one of the middle range brackets.

For a Canon 6D with a Sigma fisheye lens 8mm, using *Fix Red/Cyan Fringe*: +30, *Fix Green/Magenta Fringe*:-30, *Fix Blue/Yellow Fringe*: 0 works fine, but it is required to find the correct values in case of a different camera or lens:

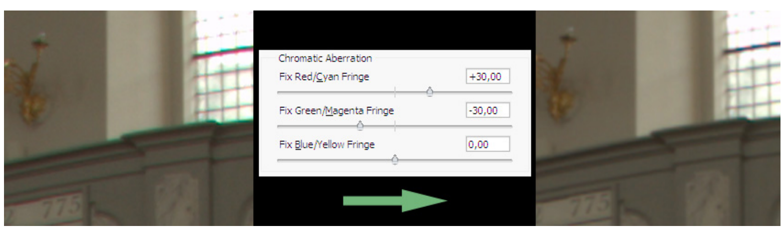

Figure 61: Correction of lateral chromatic aberration.

This correction needs to be applied on all brackets for each view. Create a Adobe Photoshop script and apply it to the folder where all the brackets are located.

5.2.2 Axial chromatic aberration (CA)

We use ViewNX 2 (Nikon software) to minimize axial chromatic aberration. Note that this software accepts 16 bit TIFF but not 32 bit TIFF.

Figure 62: Correction of axial chromatic aberration.

Remark: This process takes a long time when applied to all images.

5.2.3 Vignetting

It is important to know the impact of the vignetting of the lens/ND filters for a given aperture as it modifies the intensity of the recovered pixels. For this, we need to characterize it.

Lens characterization

To characterize the lens vignetting, we perform a shot at f/8 in front of a white screen (display a full screen white texture in Adobe Photoshop) with a white paper between the camera and the screen. A direct shot of the screen shows the LED pattern so we use a white paper to hide it. Note that the paper also shows a pattern. To remove it, we move the paper continuously and shoot with a shutter speed of 2”, which produces blur. The test should be done in a dark room or in a box to be protected from environment lighting[23]:

[23] Other methods can be used. For instance, [Ina06] uses a series of shots of a target, incrementally rotated, while [DXO13] uses a more complex setup than the one we describe here. Our method is not perfect, but it is simple and good enough for our needs.

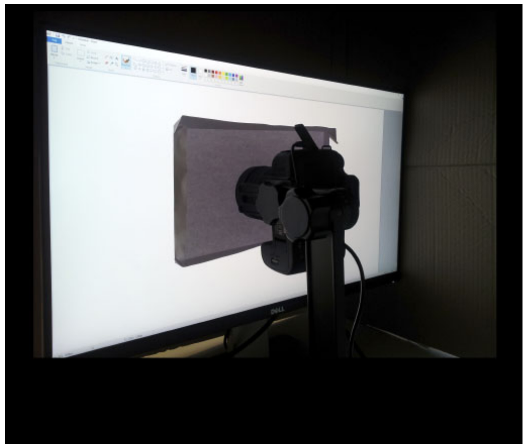

Figure 63: Vignetting characterization with simple setup. White paper on a white screen.

Characterization process for a given lens:

- Capture a shot of a moving white paper in front of a white screen (in RAW).
- Convert RAW to TIFF 16bit (See section 5.1).
- Open image in Adobe Photoshop and switch to 32 bit mode.
- Apply a Gaussian blur of 4 to average the noise effect.
- Normalize the image: apply a white balance with an RGB of 1.0, 1.0, 1.0 with the pixel at the center of the screen (assumed to be perfect for the lens) as described in Section 5.8.1.
- Resulting image is the vignetting.

Here is the result of this process:

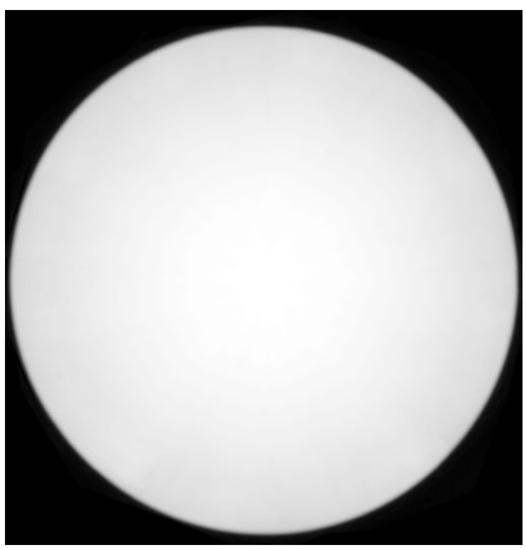

Figure 64: Characterized lens vignetting artifact.

To perform vignetting correction on an image:

- Open image in Adobe Photoshop and switch to 32 bit mode.

- Add a layer on top of the current *background* layer and set blend mode to divide.
- Paste vignetting image in top layer.
- Resulting image is the image without lens vignetting.

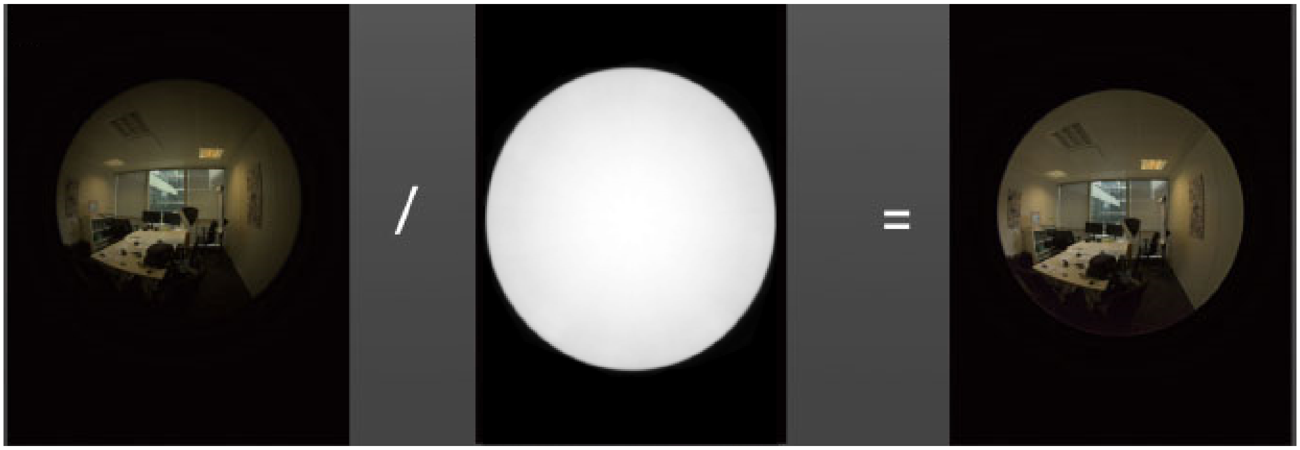

Figure 65: Apply vignetting correction. Left: Vignetted image. Middle: Lens vignetting. Right: Image corrected.

Plotting the values of the normalized image shows the attenuation strength depends on the angle:

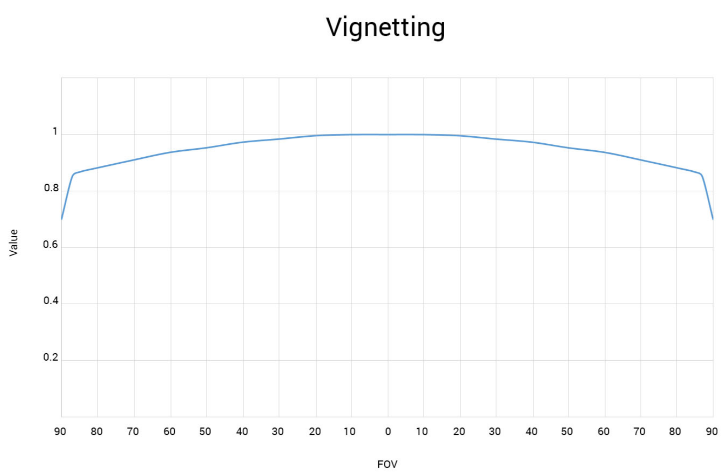

Figure 66: Vignetting from Sigma 8mm fisheye lens for f8. The vignetting is symmetric, so 60°/70° on one side means 120°/140° on both sides.

Three views of 120° are required to cover the 360° panorama, see Section 4.2.5. The area between 120° and 140° is where the images overlap each other and are used by PTGui to stitch the image. For a FOV between 0° and 120° the attenuation is barely visible and does not impact the final result. The following example shows our PTGui reconstruction with and without lens vignetting correction:

Figure 67: Impact of vignetting on HDRI reconstruction. 1: Overlapped area reference (PTGui). 2: HDR with vignetting (Nuke). 3: HDR corrected (Nuke)

For a Sigma 8mm fisheye lens at f/8, the values of both corrected and uncorrected are very close and at Unity, we chose to not correct the lens vignetting to simplify the worfklow.

ND filter characterization

Due to their design, ND filters are not fully neutral: they exhibit some vignetting and transmission loss depending on the brand, which may impact the capture. To characterize an ND filter we use the same setup as for the lens - we take a shot with and without the ND filter and calculate the difference.

All shots with and without an ND filter need to have the same exposition. By default without an ND filter the camera settings are: shutter speed 2", ISO 100, and aperture f/8. This means that the EV reduction of the ND filter needs to be compensated by first increasing the shutter speed, then the ISO (do not change the aperture) to match the default exposure. For example: for an ND500 with 9 EV reduction, use shutter speed 30" and ISO 3200.

Characterization process for each ND filter used:

- Capture a shot of a moving white paper in front of a white screen without an ND filter (but with the filter support) (in RAW).
- Capture a shot of a moving white paper in front of a white screen with the ND filter (in RAW) with the correct exposure.
- Convert RAW to TIFF 16 bit (See section 5.1).
- Open the image with the ND filter in Adobe Photoshop and switch to 32 bit mode.
- Add a layer on top of current *background* layer and set blend mode to divide.
- Paste image without the ND filter in top layer.
- Flatten and apply a Gaussian blur of 4 to average the noise effect.
- Resulting image is the vignetting with chromatic shift and EV compensation (see below).

Note: to create the texture filter correctly, the filter support should stay in the same place. To guarantee that, we recommended putting a marker on the filter support to align it with a reference point on the lens.

Figure 68: ND filter characterization. Left: Image with the ND filter. Middle: Image without a filter. Right: Vignetting texture (pink pixels are artifacts due to the noise in the black area from the fisheye lens. They have no importance).

Here is the characterization of the Hoya ND filters we use at Unity:

Figure 69: Characterization of various Hoya ND filters. The image is displayed with a range remapping of [0.9 .. 1.0] to [0 .. 1]. This means that only the attenuation from 0.9 to 1.0 is shown. The image shows that vignetting is barely visible and can be ignored. A very slight shift in blue is visible on the ND32 and ND500 (the yellow pattern).

In addition to characterizing vignetting, our setup allows us to detect when an ND filter doesn't match its specification. For example we have detected that our ND1000 is not a true 10 EV reduction as we get vignetting values above 1. This highlights that the ND1000 does not perform a reduction of 10 EV but a bit less. This means using a ND filter 1000 with an EV reduction of 10 will add a small amount of light, which needs to be compensated for.

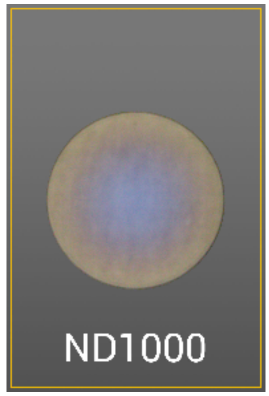

Figure 70: Display of characterization of the Hoya ND1000. The image is displayed with a shift of -1. This means that all pixels have values above 1.

As highlighted by this experiment and others on the Internet [Car], Hoya ND filters are generally good: they exhibit almost no color balance (the yellow on the image of the ND filter characterization) and insignificant vignetting. For our workflow based on such equipment, we only perform a correction when using an ND1000 to compensate for EV reduction mismatch.

The process above characterizes one ND filter but does not represent the full correction for vignetting. To get the full vignetting characterization to use we need to take into account lens vignetting and all ND filters used:

- Open lens vignetting image in Adobe Photoshop and switch to 32 bit mode.
- Add a layer on top of current *background* layer and set blend mode to multiply.
- Paste ND filter vignetting image into the top layer.
- Repeat two previous steps for each ND filter used.
- Flatten.
- Resulting image is the full vignetting that can be use to correct the vignetted image.

5.3 Convert 16 bits to 32 bits TIFF

PTGui processes 16 bit TIFFs and 32 bit TIFFs differently. It automatically applies a non-linear transform on 16 bit TIFF inputs (it applies gamma correction to convert from sRGB to linear space) and does nothing for 32 bit TIFF inputs (i.e., considers it as linear space). `dcraw` outputs 16 bit TIFF so it needs to be converted to 32 bit TIFF before being sent to PTGui to avoid any modification of the inputs.

This is done with Adobe Photoshop after the various CA corrections and before vignetting correction (but can be done after). The image is loaded then saved as 32 bits TIFF (create an Adobe Photoshop script to execute this on a folder). Note that for the axial chromatic aberration ViewNX 2 does not support TIFF 32 bits.

5.3.1 Distortion

The distortion due to the circular fisheye format is corrected by PTGui during the stitching process.

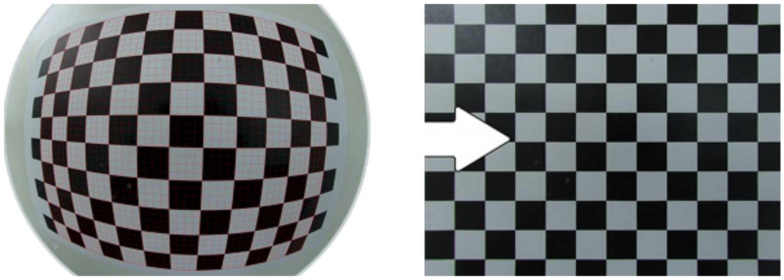

Figure 71: Example of fisheye distortion correction.

5.4 Create the HDR panorama

PTGui Pro is used to stitch bracketed exposures into an HDR panorama (the Pro version is required to reconstruct the HDRI). The goal is to stitch well exposed images together, apply the stitching template to other overexposed or underexposed images, then generate an HDRI[24].

In case of very bright light sources taken with the process described in Section 4.4.1 it is required to have the same number of brackets for each view to load images in PTGui. If the bottom view is not taken with the ND filter, some black images should be created to generate the same number of brackets as the other views.

Stitch bracketed exposures into a HDR panorama

Step in PTGui:

- Import the images in PTGui (linear TIFF 32 bits). It is preferable to load the bracket according to its exposure. This helps PTGui for the reconstruction.
- In case of very bright light sources it is necessary to manually edit the exposure of the shots with ND filters. PTGui uses the EXIF metadata created at shot time to recover the exposure but the camera has no information about what kind of ND filters are in use.
  - Active the *Advanced Mode* in *Project Assistant* tab.
  - Choose the *Image Parameters* tab and edit the exposure values of shots captured with ND filters. The goal is to replace these values with their effective one taking into account ND filters EV reduction.
  - Use Table 4 to retrieve the virtual shutter speed corresponding to the current shutter speed and use that as the new exposure. Note: PTGui refers to *exposure* instead of *shutter speed*, but it is equivalent as it is the only varying parameter.

[24]It seems that this is how PTGui works internally: first stitch, then generate HDR. PTGui is able to support the opposite, generating HDR in an external program (such as Adobe Photoshop) then stitch but, according to PTGui FAQ, the control point generator is less sensitive when HDR source images are used.

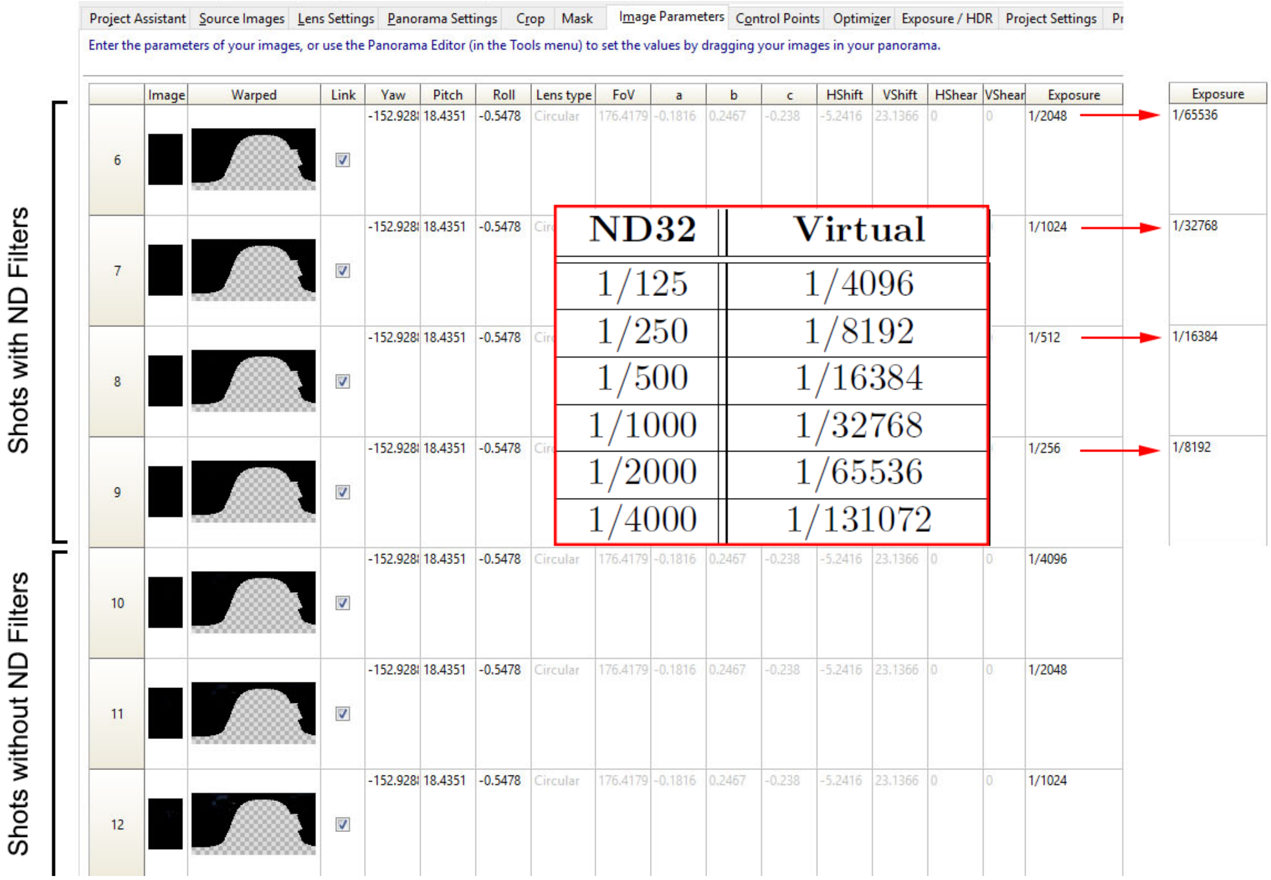

Figure 72: Example of exposure correction with an ND32 filter in PTGui. All the shots with ND filters are edited. Table 4 is use to retrieve the new exposure values. A value of 1/2048 need to be replace by its equivalent virtual shutter speed of 1/65536, 1/1024 by 1/32768 etc.

- Crop the area of the fish eye.

Figure 73: PTGui image cropping.

- In the *Bracketed Exposures* option window, chose *Enable HDR mode and link the bracketed images* to align the image with *True HDR* method[25]. It is best to link first before applying masking. When exposures are linked, the mask that is applied on a view is copied to all exposures.

If the lens parameters are not recognized properly, they can be entered manually.

- Set *Lens Type* to *Circular Fisheye*
- Enter the FOV depending if filter support or ND filters are used:
  No filter = 180°
  Filter support = 139°
  One ND filter = 124°
  Two ND filters = 124° (It should be 110° but we need to cover 360° for the panorama, see Section 4.4.1 for more details).

• Mask the tripod and the nodal head (with a fisheye and a nodal head the tripod is present in the shot, this will be removed during the stitching step).

• Mask areas unwanted in the bottom view and keep only the center to override the tripod in other shots.

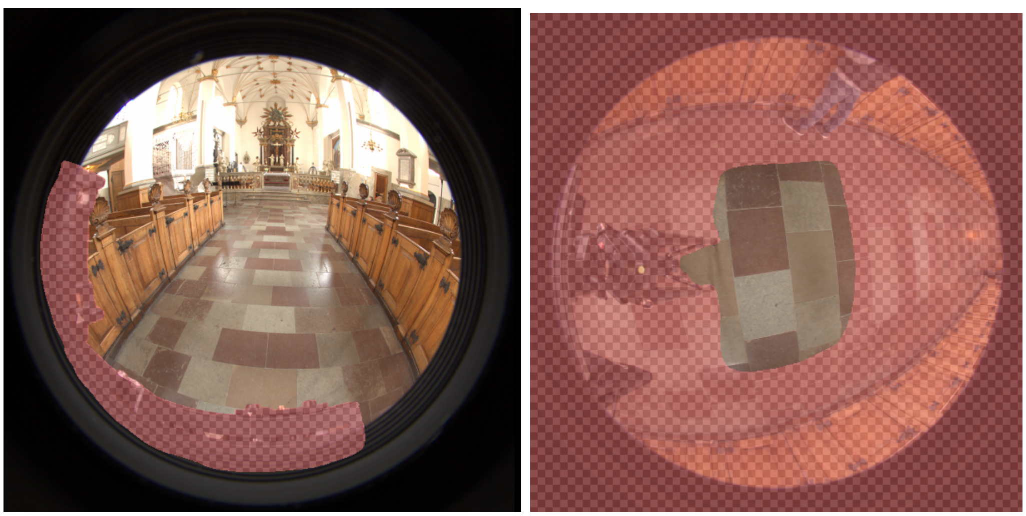

Figure 74: Left: PTGui tripod masking. Right: PTGui tripod masking on bottom view.

• The horizon can be adjusted with one view taken as reference. Modify the pitch and roll to set the real values (Pitch +/- 30, Roll 0) and use the optimizer with the modified image set as the anchor image.

• The result is an HDR panorama (all views have been stitched) but **is not calibrated**. At Unity, we export HDRIs as 8192x4096 latitude/longitude EXRs.

[25]When exposure is edited, big exposure values are converted by PTGui to float and sometime it doesn't accept linking them. Saving and reloading the project can solve this issue.

Figure 75: Uncalibrated panoramic HDRI.

- With very bright light sources, verify in Adobe Photoshop that no artifacts are visible at any exposure.

Figure 76: Example of varying exposures of an HDRI with a sun.

5.5 Correct equipment misalignment

In case the equipment was not perfectly aligned (it often happens when the user doesn't have enough experience) or the bottom view was not perfectly centred, the reconstructed HDR panorama exhibits deformation at horizon. It can also exhibit seams where the bottom view is stitched with other views. To correct this:

- In PTGui, re-adjust the horizon with the *Panorama Editor* to fix the deformation[26] It is important to have the horizon at the nodal pivot as the absolute HDRI process relies on it (see Section 5.8.2).

[26] If a simple move is not enough, it is recommended to use advanced options: use the advanced button in the *Project Assistant* tab to switch to this mode. The advanced mode adds new tabs. In the *Optimizer* tab, the reference view can be modified and in the image parameters the yaw, pitch and roll can be adjusted.

Figure 77: PTGui horizon correction.

- If necessary, add some control points for the bottom view and the others towards ground views (0°, 120°, 240°) and realign the images in *True HDR*.
- If necessary, apply an offset with Adobe Photoshop on all bracketed exposures of the bottom view to compensate for the move of the lens capture point. This will minimize the seams:
  - In PTGui, save the project.
  - The offset is determined only on one bracket. Choose a good exposure to work with and keep this same exposure for all views. Delete others. (Do not save! The project will be reloaded later.)
  - With Adobe Photoshop, apply an offset to the bottom view (1: red arrows) and check the modification with the PTGui preview (2).

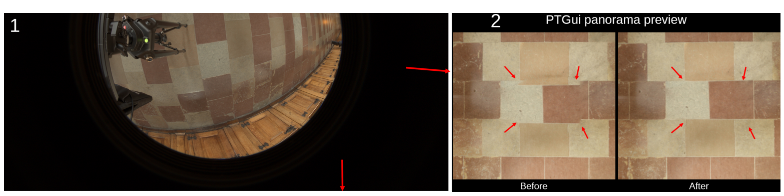

Figure 78: Bottom view alignment.

  - Load all others bracketed exposures of the bottom view in Adobe Photoshop. Replicate the same offset on all images.
  - Reload the project[27].

[27]The PTGui panorama preview has a hot reloading feature to quickly see any change performed in Adobe Photoshop.

5.6 Create the HDR panorama with the ColorChecker target

To white balance, it is necessary to reconstruct a second HDR panorama with the same process as Section 5.4. But this time replacing the bottom view images with the bottom view images including the ColorChecker.

- Start from the previous reconstructed HDR panorama.
- If an offset has been applied previously, the same offset for the bottom view images must be applied to the bottom view images with ColorChecker.
- Replace bottom view images one by one (we found that the multi-replace option of PTGui was not reliable).
- If the exposures have been edited (for very bright light sources), this needs to be done again as replacing the image discards the settings.
- Adjust the mask on the ColorChecker. The ColorChecker must be covered by a green area so PTGui will not blend it with the other views during the stitching (this means the final image will contain an unmodified ColorChecker).

Figure 79: PTGui mask adjustment for a ColorChecker.

- Create a new HDR panorama including the ColorChecker.

Figure 80: Uncalibrated panoramic HDRI with a ColorChecker.

5.7 Correcting common problems

In some conditions, it is not possible to avoid difficulties such as tripod shadow or incorrect view alignments. The Adobe Photoshop tools can be used to correct them directly in the reconstructed HDR panorama. In the case below, the HDRI was corrected via stamp and select/warp tools.

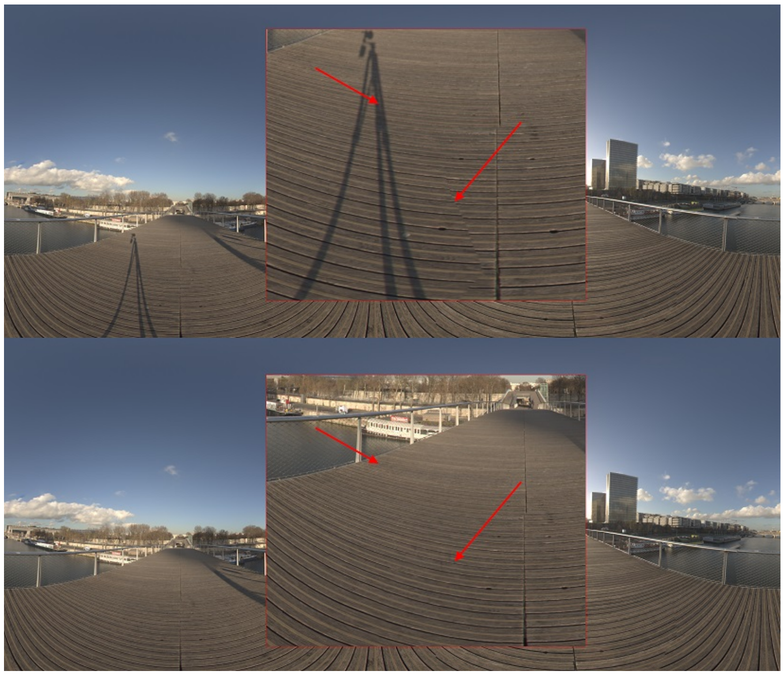

Figure 81: HDRI tripod shadow removal.

The sun capture is really sensitive if clouds are moving a lot. Due to the very high intensity of the sun, cloud movements create artifacts during the HDR process.
In the HDRI below, some clouds that are moving in front of the sun during the capture create an artifact (border in red).

Figure 82: An artifact caused by moving clouds.

If the highlights are not stitched correctly, this can produce similar artifacts (black areas in the light). This can be resolved by masking the highlight in green in the most appropriate view.

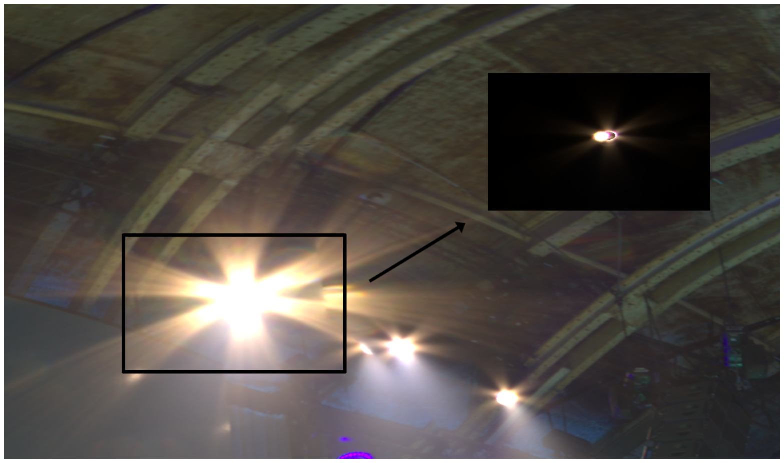

Figure 83: Stitching artifacts from incorrect highlights.

When capturing the sun, the time spent removing the ND filters between each round can cause a sun misalignment because the sun is moving. In practice we observe a shift of around two pixels for one minute. This shift causse error during PTGui's reconstruction of the sun.

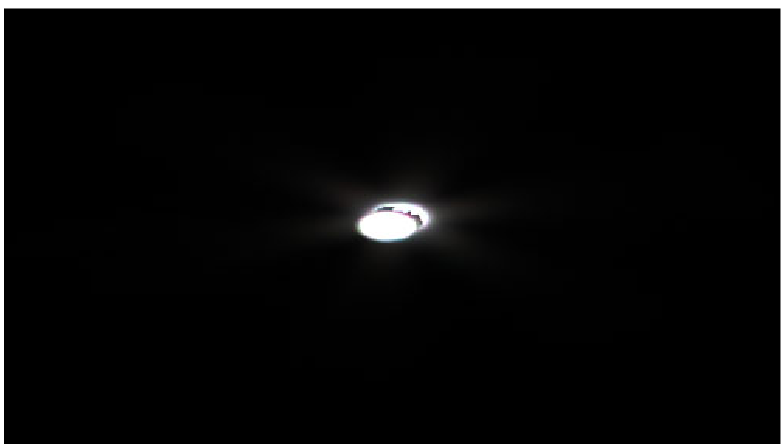

Figure 84: Incorrect sun reconstruction due to misalignment.

It is important to correct this artifact as the sun is very intense. The sun usually requires ND500 + ND1000. Removing the ND500 on top of ND1000 after the first round takes us around one minute; removing the ND 1000 after the second round is insignificant. To correct the shift between first and second rounds we align the first round of shots to the second one with Adobe Photoshop with a *move* command.

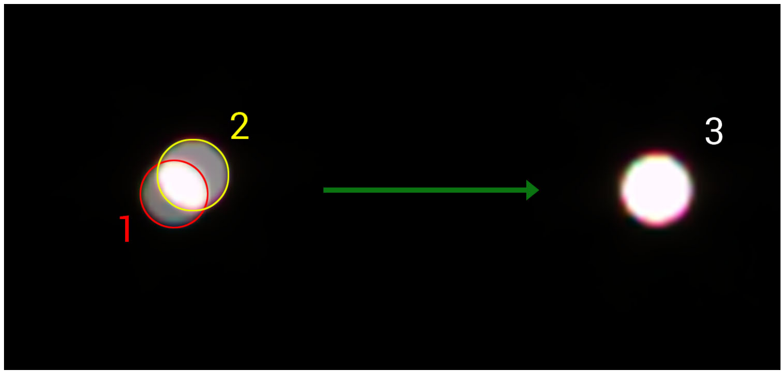

Figure 85: 1: Shot with ND500 + ND1000. 2: Shot with ND1000. 3: Corrected shot with Adobe Photoshop where shot with ND500 + ND1000 is aligned with the shot with ND1000.

5.8 Calibration and range validation

After acquisition and processing, an HDRI needs to be calibrated in both chrominance (white balance) and luminance as explained in Section 2.6.

5.8.1 White balance process

Adobe Photoshop is used to do the white balancing. For this, an RGB factor will be applied on the white unbalanced HDRI. We use the 'scaled monitor R, G, B' method described in [Wikd]. The factor is calculated with the help of the white balance chart (60% white patch) present in the HDRI.

$$\begin{aligned} R_{wb} &= R_{gain}R, \quad \text{where } R_{gain} = \frac{0.6}{R} \\ G_{wb} &= G_{gain}G, \quad \text{where } G_{gain} = \frac{0.6}{G} \\ B_{wb} &= B_{gain}B, \quad \text{where } B_{gain} = \frac{0.6}{B} \end{aligned} \tag{4}$$

The Adobe Photoshop *solid color* adjustment layers are used to compute these equations. The goal of this section is to go from a white unbalanced HDRI (left image) to a white balanced HDRI (right image):

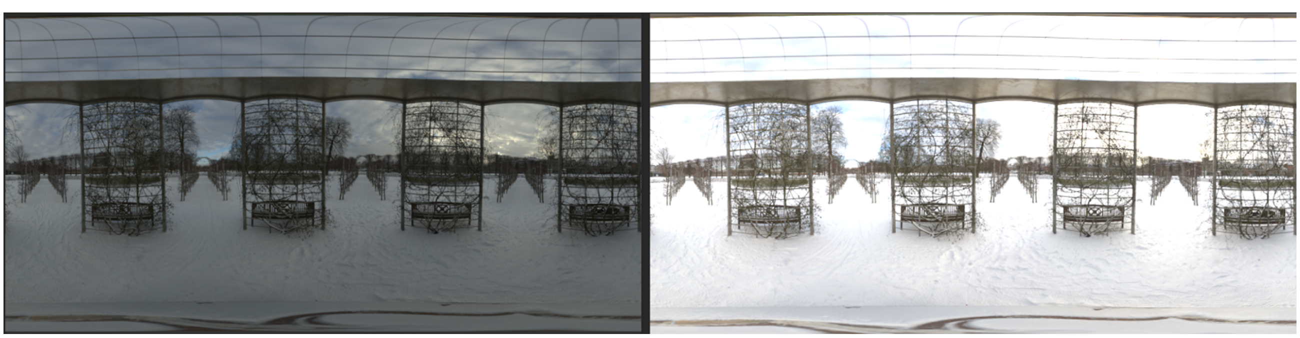

Figure 86: Example of white balance.

- Open the HDRI white unbalanced in Adobe Photoshop.
- Create a layer on top of it and name it *WB*, then copy-paste the HDRI with the white balance target in it..

- If the white balanced target is not visible (this may happen in case of a dark HDRI), then adapt the exposure of the *WB* layer to see it correctly. Then apply the same exposure adjustment to the background layer.
- Select the eyedropper tool, set the sample size to *5 by 5 Average* and pick the color value in the center of the white color target.

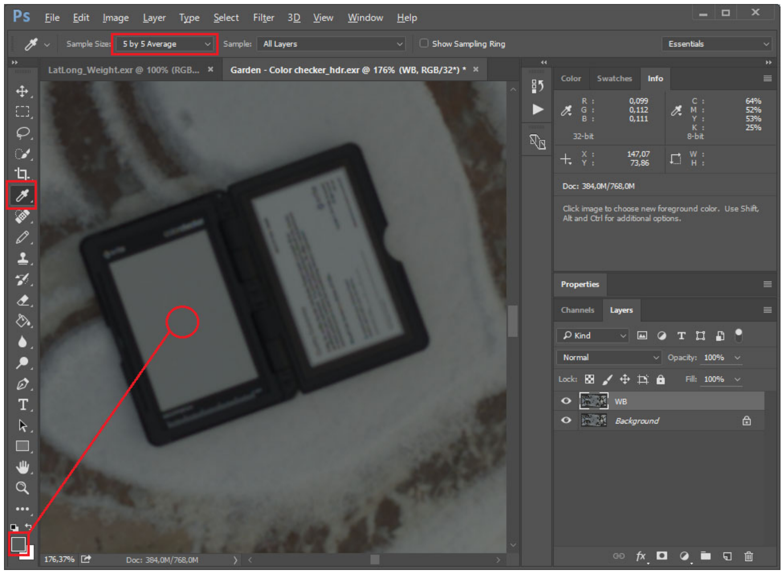

Figure 87: Adobe Photoshop setup.

- Create a new solid color adjustment layer **in 32-bit value**, set the blend mode to divide, and pick the foreground color as solid color (use Ctrl+Backspace to fill a solid layer with foreground color; this may be used when creating a Photoshop action).
- Create a second solid color adjustment layer **in 32-bit value**, set the blend mode to multiply and the value to RGB 0.6, 0.6, 0.6.

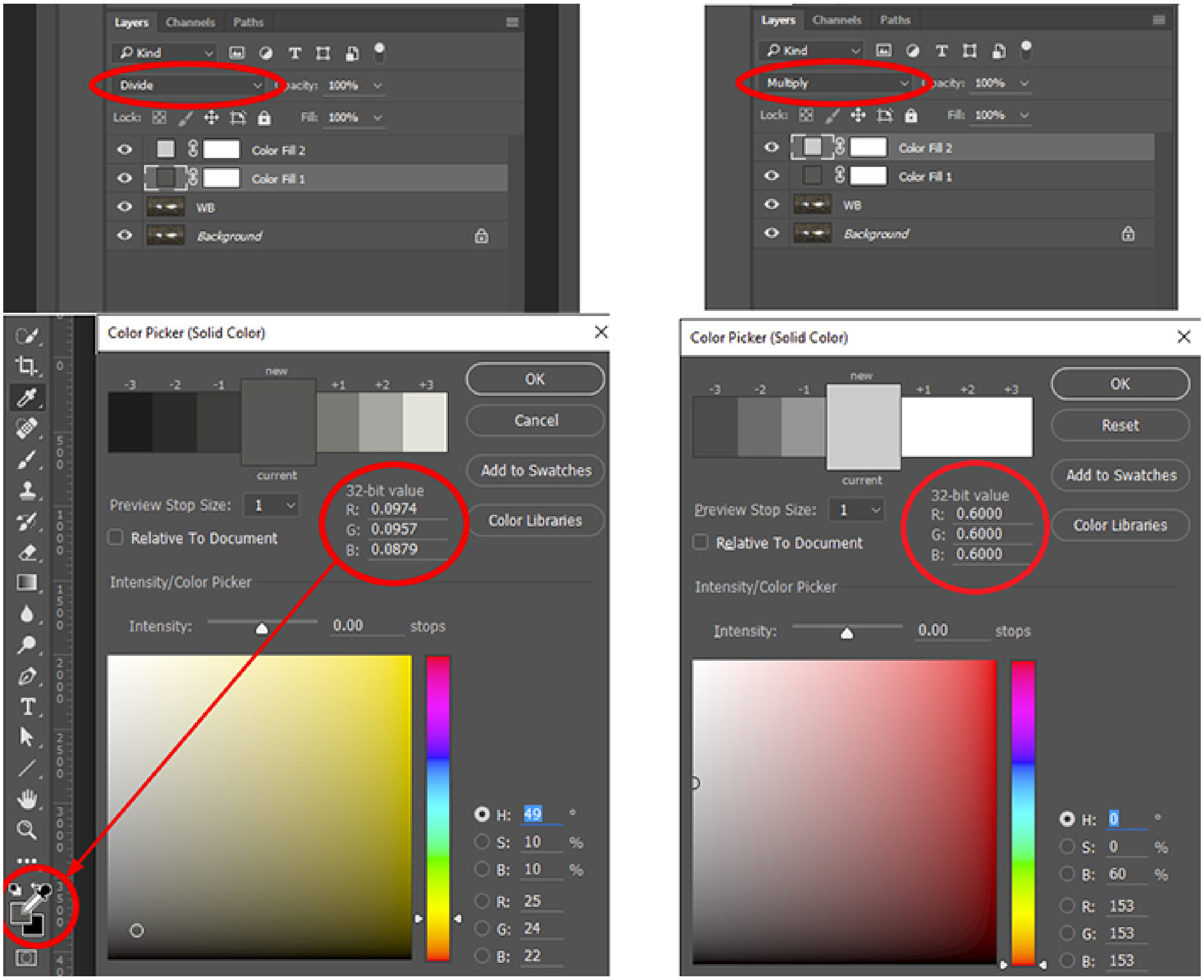

Figure 88: Adobe Photoshop setup.

- Verify that the white balance target is now RGB 0.6, 0.6, 0.6 then delete the *WB* layer.
- Flatten the layers using: *Layer / Flatten*
- Optional: the generated HDRI is just relative luminance. An exposure correction may be applied at this step without having any impact on the luminance calibration[28]. An exposure correction can be used to produce more pleasant thumbnails in various software packages, such as Unity. The thumbnail usually looks like the Adobe Photoshop view. Below is an example of a white balanced HDRI (left image) and a more pleasant correction (right image):

[28] Note that after the white balance process, the HDRI should be similarly equalized in terms of light intensity between each other (so switching from one HDRI to another does not imply large intensity changes). Applying an exposure correction may break this balance.

Figure 89: Example of thumbnail exposure.

- Save the image as the final HDR panorama.

5.8.2 Absolute HDRIs

To get an HDRI with physical units, an additional process is required. The goal is to find the factor that converts the relative luminance of the pixels to an absolute luminance in $cd.m^{-2}$. For this, we use the up direction lux meter measurement $illum_{measured}$ taken at capture time, in Section 4.5.2:

$$RGB_{calibrated} = \frac{measurement_{real\ world}}{measurement_{image}} = \frac{illum_{measured}}{illum_{image}} RGB, \tag{5}$$
[29]

where $illum_{image}$ is the HDRI upper hemisphere integral of the relative luminance of pixels (same direction that was used for $illum_{measured}$):

$$\int_{\Omega_{\text{upper}}} L \ \cos(\theta)\ \omega. \tag{6}$$

For a latitude/longitude image the solid angle (or Jacobian) of a pixel is

$$\omega = \sin(\theta)\, d\phi\, d\theta = \sin(\theta) \frac{2\pi}{w} . \frac{\pi}{h}, \tag{7}$$

where w and h are the width and height of the image, respectively.
The pseudocode to calculate $illum_{image}$ is provided in Listing 1.

```

// Convert RGB to luminance
// With radiometrically linear rgb, sRGB primaries and a D65 white point
Luminance(linearRgb) { return dot(linearRgb, vec3(0.2126729, 0.7151522, 0.0721750)); }

sum = 0;

for (y = 0; y < height;  y++)
{
    for (x = 0; x < width;  x++)
    {
        theta = (y / (height - 1)) * PI;

        if (cos(theta) > 0) // Only consider the upper hemisphere
            sum += Luminance(Pixels[x, y]) * cos(theta) * sin(theta);
    }
}

sum *=  2 * PI * PI / (width * height);
```

Listing 1: Integrate upper hemisphere of a latitude/longitude image.

[29]The calibration could be performed on the luminance Y of the xyY color space but this complicates the process for the artist.

Code is not artist friendly. To keep our workflow simple, we simulate this code with a texture and Adobe Photoshop commands. First, we generate a texture that stores the weights to be applied to the HDRI pixels. Based on Listing 1, the weight calculation is $\text{Pixels}[x,y]\cos(\theta)\sin(\theta)\,2\pi^2$. Listing 2 gives the pseudocode to perform this operation.

```
// Calculation is independent of x, so a 1-pixel width texture is fine
Pixels[1, height]

for (y = 0; y < height; y++)
{
    theta = (y / (height - 1)) * PI;
    Pixels[0, y] = (cos(theta) > 0) ? cos(theta) * sin(theta) * 2 * PI * PI : 0
}

WriteToExr(Pixels) // dump the 1 x height texture into EXR file
```

Listing 2: Calculate latitude/longitude weight texture.

This weight texture needs to be linearly scaled in width and height to match the HDRI size before any processing[30]. It is mandatory that the HDRI is reconstructed with the same up direction as the one used for the weight texture. Second, within Adobe Photoshop, we multiply the HDRI with the luminance weights: `vec3(0.21, 0.72, 0.07)`[31] and the scaled weight texture. Third, we implicitly divide by the number of pixels via a linear resize of the image to 1×1. The resulting value in this single pixel is $illum_{image}$[32]. With $illum_{measured}$ and $illum_{image}$ known, we can apply Equation 5 to get the absolute HDRI.

The complete process to obtain an absolute HDRI with Adobe Photoshop is shown in following diagram:

Figure 90: Luminance calibration workflow overview.

- Open the HDRI and the weight texture (if the HDRI was already open, close it and open it again).
- Resize the weight texture to the same size as the HDRI.
- Add a Channel Mixer adjustment layer on top of the HDRI to generate a grayscale image[33]. Set it up as a monochrome output with value RGB 21, 72, 7.
- Copy the weight into a layer on top of the Channel Mixer, and set its blend mode to multiply.

[30] Generating the weight at the same resolution as the HDRI is not necessary, a bilinear scale have enough precision for our purpose. This means it can be adapted to any HDRI size

[31] This corresponds to the luminance weights rounded to two decimals, for radiometrically linear RGB, with sRGB primaries and a D65 white point.

[32] In order to validate that the weight texture is correct, try to linearly resize it to 1×1. The resulting value should be π (3.1416) because $\int_{\Omega_{\text{upper}}} \cos(\theta) = \pi$.

[33] Adobe Photoshop has a Grayscale image mode. But in Adobe Photoshop CC 2015 we use the weights vec3(0.25, 0.698, 0.052) instead of the D65 derived one.

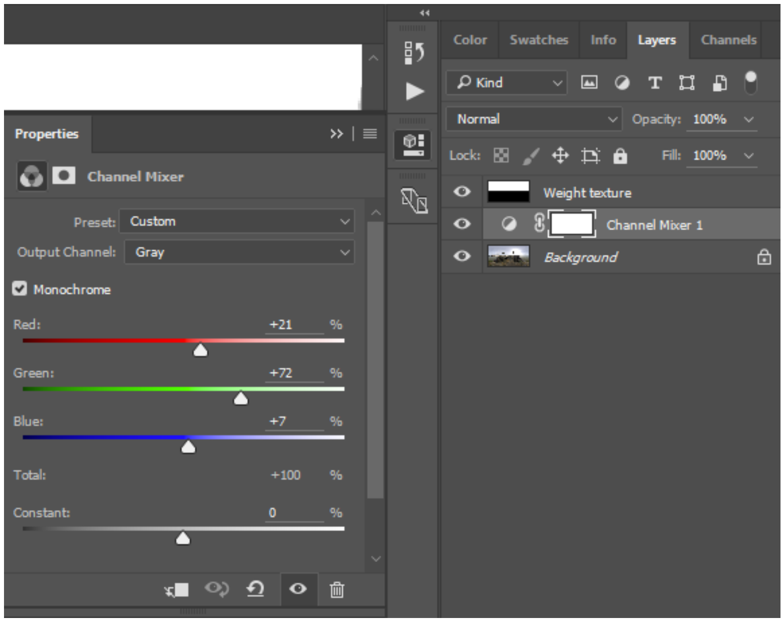

Figure 91: Adobe Photoshop setup.

- Flatten the image.
- Perform an image resize to 1x1 using *Bilinear* as resampling algorithm. Followed by a resize to the original size of the HDRI (this will replicate the averaged value on all pixels). The pixel value is the $illum_{image}$ value.
- Duplicate the *Background* layer (the current) and name it *Average*. Set blend mode to divide.
- Paste the original HDRI texture in between *Background* and *Average* layer (set the blend mode to normal).
- Create a new solid color adjustment layer **in 32-bit value** on top of the 'average' layer with multiply blend mode. Set the color value equal to $illum_{measured}$.
- Flatten the image, it is the absolute HDRI. The HDRI is usually completely white without exposition adjustments.

  Remark: Because Adobe Photoshop does not allow editing a value greater than 20, it may be necessary to use a composition of few solid colors to get the correct value. For example, in the screenshot below the measured value is 180. In order to get this value, create a solid color RGB 18.0, 18.0, 18.0 and another one with color RGB: 10.0, 10.0, 10.0. The resulting multiply value will be 180.

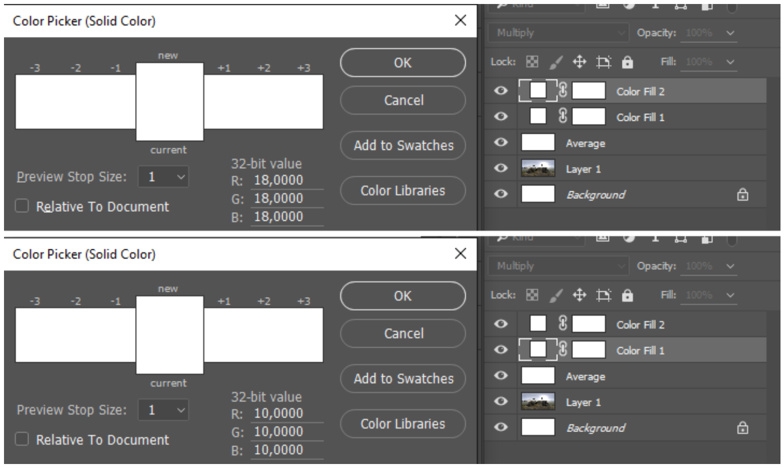

Figure 92: Adobe Photoshop setup.

The various Adobe Photoshop operations presented here can be gathered in *Actions* (Adobe Photoshop commands) for ease of use. The supplemental material associated with this document includes the weight texture and the Actions we use at Unity.

5.8.3 Maximum range

In case of very bright light sources, the relative HDRI may have values greater than 65504 (max value in a EXR float 16bit file). There is a high likelihood that these values will be clipped either due to Adobe Photoshop itself (the current version of Adobe Photoshop doesn't support 32 bit values when saving in EXR, so it clips them to 16 bit values) or in the game engine. We recommend re-exposing the HDRI in Adobe Photoshop in order to remap the maximum value of the HDRI to 65504 to avoid this clipping.

The limitation of the maximum value being 65504 for the relative HDRI is even more applicable for absolute HDRI and always happens with the sun in clear sky. In practice the luminance calibration factor $\frac{illum_{measured}}{illum_{image}}$ is applied in the game engine with a simple multiplier on the HDRI value fetched in the shader. To get this factor, merge the "Average and Solid Color" layer and read the resulting value. This also ensures we keep good thumbnails.

5.8.4 Validation

To validate the accuracy of using the illuminance measurements approach to calibrate the HDRI we have performed comparisons with luminance measurements from a luminance meter. A luminance meter is much more expensive and a less common device than a lux meter thus the reason why we do not use it in our workflow. During the illuminance measurement of the white color target, we also measure the luminance emitted from it to the point of view of the camera. We reconstruct the absolute HDRI that includes the white color target based on the lux meter measurement. Then we compare the grayscale pixel value of the center of the white color target with our luminance measurement.

Table 6 provides matching statistics for various Unity HDRI shots in various conditions. The first HDRI of the table named **Station** was shot in an interior without any natural lighting and very stable artificial lights. The luminance measurements after reconstruction match. It was enough to validate our lux meter method approach. All other measurements are less accurate for various reasons explained below. The takeaway from this table is that the inaccuracy of the result does not come from the illuminance or luminance meter method but from our real world measurements.

The accuracy of real scene measurements, either with the lux meter or the luminance meter, can vary a lot depending on context. First, it is difficult to manually setup the measurement device at the exact location of the camera pivot point and direction. Second, when outside, light can vary a lot due to moving clouds. In interiors, lights can flicker or have varying intensities, or sun light coming from windows can be influenced by moving clouds.

Name	**Interior Exterior**	**Stability**	**Lux**	**Nits**	**Recovered Nits**	**Match**
Station	int	stable	127	18.22	17.75	97%
Christian Church	int	unstable	64	9.75	10.30	94%
Trinitatis Church	int	unstable	73	8.94	10.32	84%
Unite	int	unstable	580	160	190	81%
Treasure Island	ext	unstable	51000	6700	6940	96%
Seine Bridge 02	ext	unstable	36	7.9	9.4	81%
Stadium	ext	stable	113	26.5	20.5	77%
SeaSide	ext	unstable	11800	1541	1910	76%
Marina Beach	ext	unstable	58000	6300	8270	69%
Jardin Plantes	ext	unstable	382	34	45.2	67%
Treasure Island 2	ext	unstable	31000	3600	4850	65%
KirbyCove	ext	unstable	1800	580	310	53%
Stade Small	ext	unstable	4000	468	708	49%
Garden	ext	unstable	7900	1000	1570	43%

Table 6: Accuracy of our method for various HDRIs. Accuracy is highly dependent on lighting conditions during the measurement.

Interiors such as **Christian/Trinitatis Church** are affected by sunlight coming from windows, but they still have accurate reconstruction. Interiors with a lot of strong LED projectors in the context of the **Unite** HDRI were causing lighting instabilities. Regarding the result outside, all the HDRIs presented here were done in the presence of moving clouds, showing the large variance of the results. Several of them were shot at late or early hours when the sun was moving fast (such as near sunset). Moreover, due to our inexperience, most HDRIs were shot in around thirty minutes causing a lot of lighting condition variations between each view. The **Treasure Island** was shot with a sun high in the sky and during a period where the clouds had not hidden it which explains its more accurate result. It should be noted that the order of magnitude is still respected for all HDRIs. Considering that the sky lighting conditions are changing, the illuminance measurement method can currently be more accurate than the luminance measurement method. We also stress that we don't consider the luminance meter to be the most accurate measure, but we compare two different measures and see if they match showing that the process is correct. We try to average the illuminance value between the first and last shot but in case of fast changing lighting conditions the value can vary a lot. Moreover, the HDRI itself is composed of views with different lighting conditions in this case.

Alternatively, a quick validation/correction of the absolute HDRI can be done with known light values. In a correctly reconstructed image, a clear sky should have a luminance of around 8000 $cd.m^{-2}$ and an overcast around 2000 $cd.m^{-2}$. For night scenes, values of 3000 to 5000 $cd.m^{-2}$ are not uncommmon under street light illumination. The luminance of the moon is around 2500 $cd.m^{-2}$, other objects are rather quite dark with values $< 1cd.m^{-2}$ when lit only by environment lighting [McN]. The sun is around $1.6 * 10^9 cd.m^{-2}$.

6 An example of HDRI usage

6.1 Look development

HDRIs produced at Unity are used for look development. Look development is a step in the game production process where the artists try to achieve a desired look for their assets. At their disposal there is a viewer used to visualize the assets under various lighting conditions. Lighting comes from panoramic HDRIs. Using real world panoramic HDRIs as light sources guarantees that the lighting is perfect, allowing the artists to focus on the materials of an asset. Once validated, if the asset looks bad in a game level, it is because of the level lighting, not the material.

Figure 93: Example of the viewer with the HDRI browser used at Unity.

6.2 Dealing with the sun or very bright sources at runtime

Sun or equivalent bright light sources are often a problem for real-time graphics:

- High intensities cause precision issues and the sun is often clamped by game engines to 65504 (max 16 bit floating point value)
- High intensities generate a lot of noise when using some techniques like pre-filtered importance sampling for HDRI lighting, usually used in game engines [LR14].
- Games want to control the orientation of the sun, particularly for the shadow direction.
- Sometimes the sun is clipped in the HDRI itself and needs to be replaced by a more accurate version.

Figure 94: Example of a white balance HDRI with sun.

To work around these issues, it is common in the VFX and game industries to remove the sun from HDRIs and replace it by an analytic representation. This section presents various methods to do so.

6.2.1 Sun or very bright sources removal in HDRIs

The intensity of the sun is not constrained to its own disc. The whole sky color comes from the single and multiple scattering of the light with particles in the air. The consequence is that the intensity of the sky is higher around the sun. Hiding the sun with a piece of cardboard, as described in Section 4.4.2, doesn't prevent this gradient. Here is an example of a luminance gradient surrounding the sun in a clear sky:

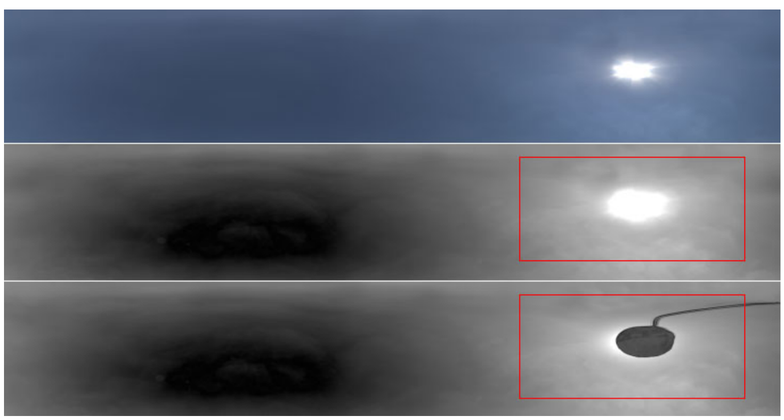

Figure 95: The first image is a slice of an HDRI, the second and third are luminance representations remapped from darkest sky value (black) to highest sky value outside of sun disc (white), one with the sun, one with the sun hidden by a piece of cardboard.

One thing to note is that we don't want to remove the sun in a physical way, as it would result in a black sky, but in an artistic way. Removing the Sun can mean removing the sun disc, removing the single scattering or something in-between. For our Unity viewer tool, our goal is to use an HDRI to approximate the lighting in the shadow of the sun (See Section 6.2.2). For this, we remove the sun and its surrounding area from the HDRI by replacing it with a representative sky. Two artist friendly solutions can be used for this purpose:

1. Duplicate a representative part of the sky to replace the sun area.
    - Load the white balanced HDRI in Adobe Photoshop.

- Create a luminance mask to work only on the area of high intensity. This luminance mask is only necessary if you have no sky information overlapping the sun area (like the tree on the figure):
  - Add a new folder to contain everything needed to produce the mask and select it.
  - Add a Channel Mixer adjustment layer to generate a grayscale image. Setup it as monochrome output with value RGB 21, 72, 7.
  - Add a solid color layer with the highest sky value picked in the area representative of the sky. Set this layer to Lighten.
  - Add another solid color with same value. Set this layer to Difference.

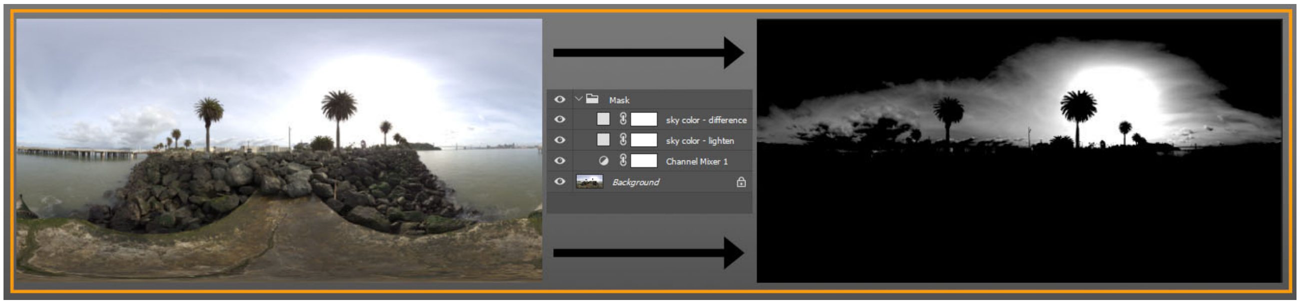

Figure 96: Adobe Photoshop setup.

- Duplicate background.
- With the Adobe Photoshop stamp tool, find a clear zone in the sky that is big enough to cover the sun disc and is representative of the average intensity of the sky.

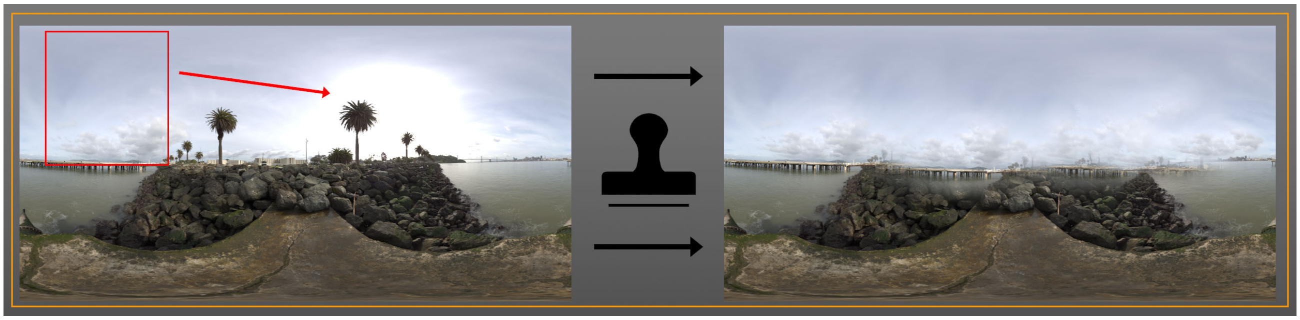

Figure 97: Adobe Photoshop setup.

- Apply the mask above on the stamped layer to have the final result. If necessary, do a retake with the stamp tool to have a nice transition.

Figure 98: White balanced HDRI with the sun removed.

2. Clamp the high values of the sun and the surrounding area.
    - Load the white balanced HDRI in Adobe Photoshop.
    - Add a solid color layer on top. Pick a representative sky value at the opposite side to the sun as RGB.
    - Set the solid color layer to Darken mode (this applies a min between the two images).
    - If needed, build a mask to only affect the sun area (opposite sky values can have lower values than the ground part).
    - Save the new HDRI. This approach is simple but can exhibit non natural reflection.

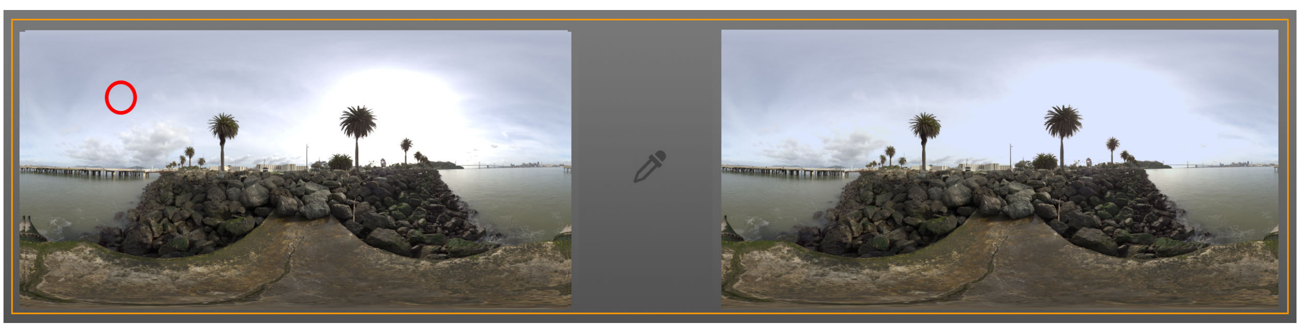

Figure 99: White balance HDRI with sun clamped.

6.2.2 Hard environment shadow for look development

When there is a very bright light source in an HDRI, artists like to visualize their asset with a hard shadow, similar to what they would get in a game level with a directional light. But in the context of real time rendering, for performance reasons, when an HDRI is used to light an object there is no geometry occlusion and thus, no shadows[34].

[34] In practice GPU raytracing could be used for real time visualization but at Unity we aim to have a WYSIWYG (what you see is what you get) approach. Raytracing gives results that are too good compared to what we can achieve in practice in games.

Figure 100: Left: Asset lit with an HDRI only. Right: Path traced offline version of the same asset. Note the presence of shadows.

This section describes a method to emulate hard shadows for HDRIs with very bright light sources. For other HDRIs producing soft shadows or no shadows like overcast sky, we chose to do nothing.

Usually, viewers add an analytic directional light that casts shadow to emulate hard shadows. This method adds extra lighting information and breaks the perfect lighting provided by the HDRI (as it was captured from the real world). At Unity we preferred not to introduce any extra lighting. We instead prepare a *shadow HDRI* that is then used when a pixel is determined to be in the shadow of the brightest light source. To do so, we generate a screen space shadow map based on the brightest light direction and then use it as a mask to select which HDRI (the regular or the shadow one) is used for this pixel.

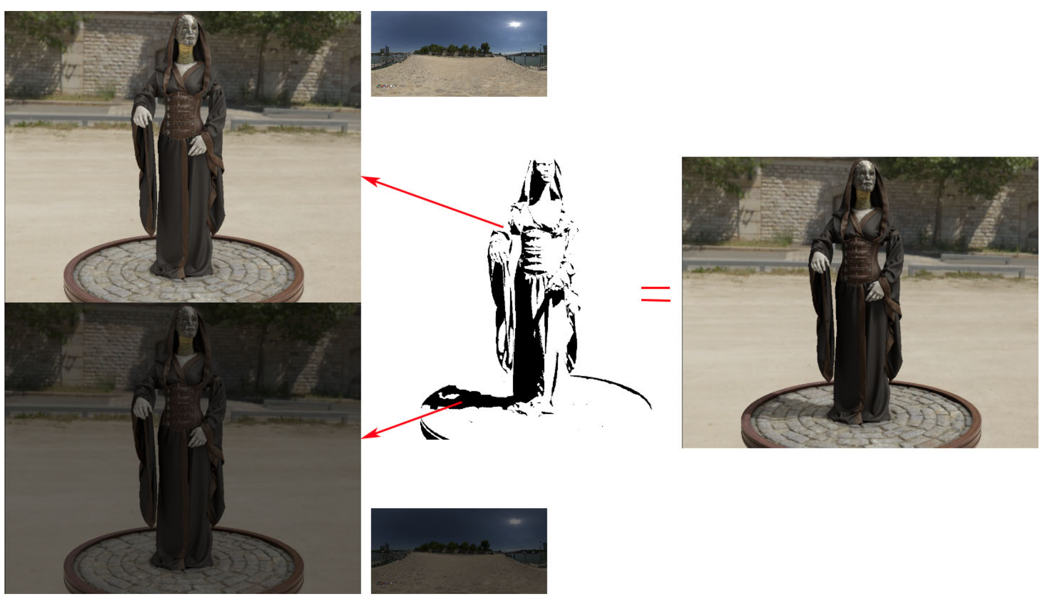

Figure 101: Top left: Regular HDRI. Bottom left: Shadow HDRI. Middle: Screen space shadow mask. Right: Final result. Note the absence of extra lighting.

In the Unity viewer tool we automatically generate a *shadow HDRI* by decreasing the exposure of the regular HDRI. It is a good default solution (that doesn't require artist intervention) but as the very bright light source is still present in the *shadow HDRI* it generates undesired bright highlights. The alternative is that an artist provides their own version of the *shadow HDRI* by removing the bright

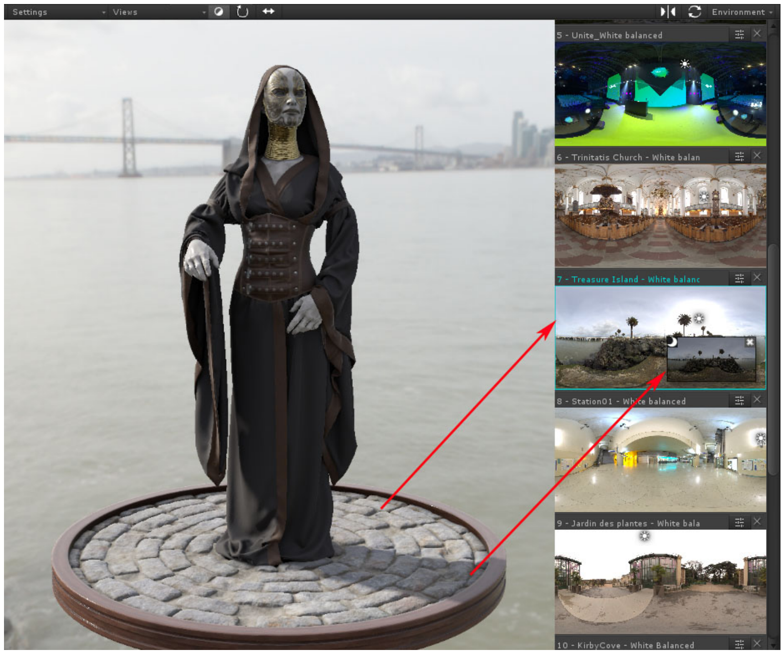

Figure 102: Shadow HDRI used to emulate shadows from a very bright light source.

light source as explained in Section 6.2.1. Figure 102 shows an example of such an artist provided shadow HDRI in our viewer tool.

To determine the accuracy of our method we have performed an offline path traced rendering with Mitsuba [Jak10][35] and compared the resulting shadow with the one from both our shadow HDRI methods (automatic and manual).

Figure 103: Rows represent two different HDRIs. Columns represent different rendering. First: Path traced, Second: Automatic with modulation of 0.3, Third: Sun removed with method 1, Fourth: Sun removed with method 1 and modulation of 0.8.

[35] Our team has developed a Mitsuba plugin for Unity that supports a subset of Unity Standard shaders. This plugin will be open sourced after the publication of this document.

Let's classify shadows between self shadows (shadows produced by the object on itself) and shadows (shadows projected by the object on the environment). We found that the automatic solution, despite having wrong highlights from the bright source, provides a good match for self shadows with a modulation around 0.3. The shadow tint is incorrect due to the bright light source being present in the shadow HDRI. With a shadow HDRI authored by hand with method 1 from the previous section we get a good shadow tint. But the self shadow is not dark enough as our shadow HDRI assumes there is no self occlusion. We found that an acceptable compromise is to use an authored HDRI with a modulation of 0.8.

6.3 Analytic light sources

When the sun or a very bright light source is removed from an HDRI it can be replaced with a directional light. It is possible to calculate the corresponding light intensity to match the lighting result of the original HDRI:

- Calculate the $illum_{HDRI}$ for the upper direction of the original HDRI.
- Calculate the $illum_{ShadowHDRI}$ for the upper direction of the shadow HDRI.
- Calculate the light angle θ (in radians) from the spherical (latitude/longitude) coordinates of the brightest point. This can be done in Adobe Photoshop by using the Y coordinate (top-down convention) normalized by the height of the HDRI and multiplied by π:

$$\theta = \frac{Y}{Height}\,\pi. \tag{8}$$

- Given

$$illum_{HDRI} = illum_{DirLight}\cos(\theta) \; + \; illum_{ShadowHDRI}, \tag{9}$$

  calculate light intensity

$$illum_{DirLight} = \frac{illum_{HDRI} \; - \; illum_{ShadowHDRI}}{\cos(\theta)}. \tag{10}$$

- Calculate light direction from spherical (latitude/longtitude) coordinates of the brightest point converted to a Cartesian direction.
- Calculate light color from the sun disc in sRGB. Set the sample size to something representative of the Sun disc like "11 by 11 Average" and pick the color value in the center of it. Then divide this RGB by the max of R, G or B.

$$RGB_{DirLight} \; = \frac{RGB_{AverageSunDisc}}{\max(R,G,B)_{AverageSunDisc}}. \tag{11}$$

  This value is RGB in linear space and it needs to be converted to sRGB:

$$sRGB_{DirLight} \; = \text{RGB\_to\_sRGB}(RGB_{DirLight}). \tag{12}$$

The calculation of $illum_{DirLight}$ can be performed on relative or absolute luminance HDRIs, it still provides correct relative or absolute values. $illum_{HDRI}$ can be calculated in Adobe Photoshop with similar steps to those in Section 5.8.2

- Open the HDRI and the weight texture.

- Resize the weight texture to the size of the HDRI.
- Add a Channel Mixer adjustment layer on top of the HDRI to generate a grayscale image. Set it up as monochrome output with values RGB 21, 72, 7.
- Copy the weight in a layer on top of the Channel Mixer, and set its blend mode to multiply.
- Flatten the image.
- Perform an image resize to 1x1 using *Bilinear* as the resampling algorithm. The pixel value is the $illum_{HDRI}$ value.

In game engines, use the light properties computed above (direction, light color, light intensity in lux) with a *shadow HDRI* to mimic the lighting of the original HDRI.

As described in [LR14], the usual directional lighting calculation performed in game engines corresponds to the maths used for a light with an intensity in lux. Also be aware that $illum_{DirLight}$ doesn't include the division of the diffuse component by π. It is frequent that game engines do an implicit division by π in light units [Lag]. As a result, the usual value provided to the shader[36] for lighting is:

$$RGBLight = \text{sRGB\_to\_RGB}(sRGB_{DirLight})\,\frac{illum_{DirLight}}{\pi}. \tag{13}$$

Here is a comparison of the results obtained with this calculation:

Figure 104: Left: Path tracer, Middle: Analytic directional lighting with *shadow HDRI*, Right: Regular HDRI + *Shadow HDRI* lighting as described in Section 6.2.2.

[36] If color and intensity are used to fill a light structure in Unity, with current version 5.4 and below, it is required to perform a correction on $illum_{DirLight}$ due to an incorrect linear lighting calculation performed by the engine. $illuminanceUnity_{DirLight} = illum_{DirLight}^{0.4545}$

Results show that the analytic lighting correctly matches the image based lighting in term of intensity. This validates the approach.

Note: we use Unity for the real-time rendering in this document. The path traced version in this document has a different light transport to Unity and the BRDFs don't match exactly between the two. It also produces some noise due to the low number of samples used. In particular, Unity (with version 5.4 and below) has a poor approximation of the HDRI integral explaining the blue tint on the ground discrepancy that can be seen in the regular HDRI + shadow HDRI screenshot. This tint is less visible when the light is replaced by the analytic light which has correct math. To compare the images, focus on the intensity.

6.4 Absolute HDRIs

Absolute HDRIs are in case the game engine uses physically based light units for other lights [LR14]. This ensures that values are in the right range. Unity doesn't support this yet but it is planned to do so.

7 Conclusion

This document includes a large amount of information and we hope that readers will find interesting both our gathering of knowledge and our practical method to capture and reconstruct accurate HDRIs. In the appendix we provide a template document that includes our shutter speed tables used to calculate the EV range and number of shots to take when capturing an HDRI. To conclude, our method consists of several steps:

Capture

With equipment from Section 3:

- Setup the tripod, nodal head, CamRanger, and align pivot point. Don't forget to have a clean lens.
- Setup camera to RAW, manual mode, disable all color/contrast enhancing and automatic white balance.
- Fix ISO to 100 and aperture to f/8 with focal length of 0.5m.
- Calculate the EV range of the scene.
- If necessary determine the ND filters to use.
- Use the histogram to check when the darkest and brightest brackets don't clip.
- Calculate the number of brackets required (EV range / EV step) + 1. Recommended 1 EV step.
- Program CamRanger with start value, number of shots and EV step.
- Capture seven views with required number of bracket: horizontal (0°, 120°, 240°) with vertical of 30°, horizontal (60°, 180°, 300°) with vertical of -30° and bottom view for tripod removal.
- In the case of a very bright light source there is a first round of shots taken with the ND filter and second round without.
- Perform an extra bracketed bottom view with white balance target.
- Use the lux meter to measure the illuminance at the camera position in the up direction.

Reconstruction

- Extract RAW picture and convert to 16 bit linear TIFF: `dcraw.exe -v -4 -T [ file ]`.
- Correct chromatic aberration with Adobe Photoshop and ViewNX 2.
- Correct vignetting with the characterization of lens/ND filters if necessary.
- Convert to 32 bit TIFF with Adobe Photoshop.
- Create the HDR panorama with and without the ColorChecker with PTGui.
- Correct horizon deformation if needed.
- Perform white balance and luminance calibration in Adobe Photoshop.
- Remap value if maximum value is above 65504 to be compliant with 16 bit precision.

Future

In the future, we would like to improve our measurement method to generate absolute HDRIs. In particular we have not compared our lux and luminance meter measurements with a spot meter measurement as in [Cha15]. We would like to investigate accuracy of these various devices and what is the best way to get an accurate measurement of the real world with changing lighting conditions.

We also would like to investigate more accurate chrominance calibration methods, such as performing the calibration on the whole ColorChecker instead of the white patch.

Lastly we would like to investigate run time white balancing instead of performing the white balance offline in the HDRI, as it is wrong by definition.

8 Acknowledgements

We would like to thank Unity for providing us with the equipment used in this document. We also want to thank all our reviewers: Julien Ignace, Jonathan Dupuy, Fabien Christin and especially Stephen Hill and Stephen McAuley for having organised this course and for all their support.

Bibliography

[Adoa] Adobe. *Adobe Camera Raw*. URL: https://www.adobe.com/support/downloads/product.jsp?product=106&platform=Windows.

[Adob] Adobe. *Adobe Photoshop CC*. URL: http://www.photoshop.com/products/photoshop.

[Adoc] Adobe. *Digital Negative (DNG) Specification*. URL: https://www.adobe.com/content/dam/Adobe/en/products/photoshop/pdfs/dng_spec_1.4.0.0.pdf.

[AT11] E. Allen and S. Triantaphillidou. *The Manual of Photography*. Elsevier/Focal Press, 2011. ISBN: 9780240520377. URL: https://books.google.fr/books?id=IfWivY3mIgAC.

[Boe13] M. Boers. *Linear RAW Conversions*. 2013. URL: http://mikeboers.com/blog/2013/11/07/linear-raw-conversions.

[Car] B. Carnathan. *10 Stop Neutral Density Filter Review*. URL: http://www.the-digital-picture.com/Reviews/10-Stop-Neutral-Density-Filter.aspx.

[Cha15] D. Chan. "Real-World Measurements for Call of Duty: Advanced Warfare". In: *Physically Based Shading in Theory and Practice, ACM SIGGRAPH 2015 Courses*. SIGGRAPH '15. Los angles, California: ACM, 2015.

[Cof] Coffin. *DCRaw*. URL: http://www.centrostudiprogressofotografico.it/en/dcraw/.

[col] C. in colour. *Lens diffraction and photography*. URL: http://www.cambridgeincolour.com/tutorials/diffraction-photography.htm.

[DM97] P. E. Debevec and J. Malik. "Recovering High Dynamic Range Radiance Maps from Photographs". In: *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '97. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1997, pp. 369–378. ISBN: 0-89791-896-7. DOI: 10.1145/258734.258884. URL: http://www.pauldebevec.com/Research/HDR/.

[DXOa] DXO. *An in-depth case study of the use of DxOMark data*. URL: http://www.dxomark.com/Reviews/Canon-500D-T1i-vs.-Nikon-D5000/Color-blindness-sensor-quality.

[DXOb] DXO. *DxOMark measurements for lenses and camera sensors*. URL: http://www.dxomark.com/About/In-depth-measurements/Measurements/Color-sensitivity.

[DXO13] DXO. *DXO vignetting*. 2013. URL: http://www.dxomark.com/About/In-depth-measurements/Measurements/Vignetting.

[Gol16] J. Goldschmidt. *Luxmeter App versus measuring device: Are smartphones suitable for measuring illuminance?* 2016. URL: http://www.dial.de/DIAL/en/dialacademy/blog/light/luxmeter-app-versus-measuring-device.html.

[Har] P. Harvey. *Exif tools*. URL: http://exiftool.sourceforge.net/.

[Ina05] M. Inanici. *Evaluation of high dynamic range photography as a luminance data acquisition system*. 2005. URL: http://faculty.washington.edu/inanici/Publications/LRTPublished.pdf.

[Ina06] M. Inanici. "Evaluation of high dynamic range photography as a luminance data acquisition system". In: *Lighting Research and Technology* 38 (2006), pp. 123–136. URL: https://facades.lbl.gov/sites/all/files/lbnl-57545.pdf.

[Jak10] W. Jakob. *Mitsuba renderer*. 2010. URL: http://www.mitsuba-renderer.org.

[JW12] P. B. C. J. E. Farrell and B. A. Wandell. "Digital camera simulation". In: *Applied Optics* 51.4 (Feb. 2012), A80–A90. URL: http://scien.stanford.edu/jfsite/Papers/ImageCapture/2012AppliedOptics.pdf.

[Kri15] M. A. Kriss. "Color Reproduction for digital cameras". In: *Handbook of Digital Imaging*. 2015.

[Kri98] M. A. Kriss. "Tradeoff Between Aliasing Artifacts and Sharpness in Assessing Image Quality". In: *Proceedings IS&T PICS Conference*. IS&T PICS Conference. New York, NY, USA, 1998.

[Lag] S. Lagarde. *Pi or not to Pi*. URL: https://seblagarde.wordpress.com/2012/01/08/pi-or-not-to-pi-in-game-lighting-equation/.

[lcl15] lclevy. *Understanding What is stored in a Canon RAW .CR2 file, How and Why*. 2015. URL: http://lclevy.free.fr/cr2/.

[Lin] B. Lindbloom. *RGB/XYZ Matrices*. URL: http://www.brucelindbloom.com/index.html?Eqn_RGB_XYZ_Matrix.html.

[LMY10] O. Losson, L. Macaire, and Y. Yang. "Comparison of color demosaicing methods". In: *Advances in Imaging and Electron Physics* 162 (July 2010), pp. 173–265. DOI: 10.1016/S1076-5670(10)62005-8. URL: https://hal.archives-ouvertes.fr/hal-00683233.

[LR14] S. Lagarde and C. de Rousiers. "Moving Frostbite to PBR". In: *Physically Based Shading in Theory and Practice, ACM SIGGRAPH 2014 Courses*. SIGGRAPH '14. Los angles, California: ACM, 2014.

[Lut27] R. Luther. "Aus dem Gebiet der Farbreizmetrik". In: *Zeitschrift für technische Physik 8*. 1927, pp. 540–558. URL: Translation:%20http://www.iscc.org/pdf/LutherColorStimMetrics.pdf.

[McA12] S. McAuley. "Calibrating Lighting and Materials in Far Cry 3". In: *Practical Physically Based Shading in Film and Game Production, ACM SIGGRAPH 2012 Courses*. SIGGRAPH '12. Los Angeles, California: ACM, 2012, 10:1–10:7. ISBN: 978-1-4503-1678-1. DOI: 10.1145/2343483.2343493. URL: http://selfshadow.com/publications/s2012-shading-course/.

[McN] McNeel. *Working with HDRIs*. URL: http://wiki.mcneel.com/accurender/nxt/documentation/beyond/hdri.

[Mye09] R. D. Myers. *Color checker Passport Technical review*. 2009. URL: http://www.rmimaging.com/information/ColorChecker_Passport_Technical_Report.pdf.

[Nik] Nikon. *ViewNX 2*. URL: http://www.nikonusa.com/en/nikon-products/product/imaging-software/viewnx-2.html.

[Phoa] PhotographyLife. *Nikon 24-70mm f/2.8E ED VR Announcement*. URL: https://photographylife.com/nikon-24-70mm-f2-8e-ed-vr-announcement.

[Phob] PhotographyLife. *Sigma 8mm f/3.5 EX DG Circular Fisheye*. URL: https://photographylife.com/lenses/sigma-8mm-f3-5-ex-dg-circular-fisheye.

[PNB13] D. K. Prasad, R. Nguyen, and M. S. Brown. “Quick Approximation of Camera’s Spectral Response from Casual Lighting”. In: *2013 IEEE International Conference on Computer Vision Workshops* (2013), pp. 844–851. DOI: http://doi.ieeecomputersociety.org/10.1109/ICCVW.2013.116.

[PTG] PTGui. *PTGui Pro*. URL: https://www.ptgui.com/download.html.

[Rei+08] E. Reinhard, E. A. Khan, A. O. Akyuz, and G. M. Johnson. *Color Imaging: Fundamentals and Applications*. AK Peters, June 2008. URL: http://www.cs.bris.ac.uk/Publications/pub_master.jsp?id=2001311.

[Ren15] M. W. D. Renato Turchetta Kenneth R. Spring. *Introduction to CMOS Image Sensors*. 2015. URL: http://www.olympusmicro.com/primer/digitalimaging/cmosimagesensors.html.

[She10] C. R. Shelby Doyle. *hdr photography using Photoshere*. 2010. URL: http://web.mit.edu/SustainableDesignLab/projects/TeachingResources/HDR_II_Photosphere.pdf.

[Stu+04] J. Stumpfel, A. Jones, A. Wenger, C. Tchou, T. Hawkins, and P. Debevec. “Direct HDR Capture of the Sun and Sky”. In: *Proceedings of the 3rd International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*. Stellenbosch, South Africa, Nov. 2004. URL: http://ict.usc.edu/pubs/Direct%20HDR%20Capture%20of%20the%20Sun%20and%20Sky.pdf.

[Sum14] R. Sumner. *Processing RAW Images in MATLAB*. 2014. URL: https://users.soe.ucsc.edu/~rcsumner/rawguide/.

[Tal+07] E.-V. Talvala, A. Adams, M. Horowitz, and M. Levoy. “Veiling Glare in High Dynamic Range Imaging”. In: *ACM SIGGRAPH 2007 Papers*. SIGGRAPH ’07. San Diego, California: ACM, 2007. DOI: 10.1145/1275808.1276424. URL: http://doi.acm.org/10.1145/1275808.1276424.

[Var14] M. Varghese Wanat. *Colorimetric calibration of high dynamic range images with a ColorChecker chart*. 2014. URL: https://www.cl.cam.ac.uk/~rkm38/pdfs/varghese14color_calib.pdf.

[Wika] Wikipedia. *Active pixel sensor*. URL: https://en.wikipedia.org/wiki/Active_pixel_sensor.

[Wikb] Wikipedia. *Chromatic Adapation*. URL: https://en.wikipedia.org/wiki/Chromatic_adaptation.

[Wikc] Wikipedia. *CIE 1931 color space*. URL: https://en.wikipedia.org/wiki/CIE_1931_color_space.

[Wikd] Wikipedia. *ColorBalance*. URL: https://en.wikipedia.org/wiki/Color_balance.

[Wike] Wikipedia. *EXIF*. URL: https://en.wikipedia.org/wiki/Exchangeable_image_file_format.

[Wikf] Wikipedia. *NeutralDensityFilter*. URL: https://en.wikipedia.org/wiki/Neutral-density_filter.

[Wikg] Wikipedia. *Optical aberration*. URL: https://en.wikipedia.org/wiki/Optical_aberration.

[Wikh] Wikipedia. *Photoelectric Effect*. URL: https://en.wikipedia.org/wiki/Photoelectric_effect.

[Wiki] Wikipedia. *Potential well*. URL: https://en.wikipedia.org/wiki/Potential_well.

[Wikj] Wikipedia. *Shutter speed*. URL: https://en.wikipedia.org/wiki/Shutter_speed.

[Wikk] Wikipedia. *Visible Spectrum*. URL: https://en.wikipedia.org/wiki/Visible_spectrum.

Name:
Location:

Date:

Time:

Weather:

Standard capture:

1/4000	1/2000	1/1000	1/500	1/250	1/125	1/60	1/30	1/15	1/8	1/4	1/2	1	2	4	8	15	30

Very bright sources capture:

EV	Virtual	No filter	ND16	ND32	ND64	ND500	ND1000	ND500 + ND1000
0	1/4096	1/4000	1/250	1/125	1/60	1/8	1/4	
-1	1/8192		1/500	1/250	1/125	1/15	1/8	
-2	1/16384		1/1000	1/500	1/250	1/30	1/15	
-3	1/32768		1/2000	1/1000	1/500	1/60	1/30	
-4	1/65536		1/4000	1/2000	1/1000	1/125	1/60	
-5	1/131072			1/4000	1/2000	1/250	1/125	
-6	1/262144				1/4000	1/500	1/250	
-7	1/524288					1/1000	1/500	
-8	1/1048576					1/2000	1/1000	
-9	1/2097152					1/4000	1/2000	
-10	1/4194304						1/4000	1/8
-11	1/8388608							1/15
-12	1/16777216							1/30
-13	1/33554432							1/60
-14	1/67108864							1/125
-15	1/134217728							1/250
-16	1/268435456							1/500
-17	1/536870912							1/1000
-18	1/1073741824							1/2000
-19	1/2147483648							1/4000

Settings:
EV step = 1 F/8 ISO 100
Lux in the scene (on the top of the camera):

Luminance on 60% white patch: