Chapter 7
DIC Test Planning with Blender – A Focus on Lighting

Marc A. Eitner, Jorge Alejandro Ricaurte, and Jayant Sirohi

Abstract The growing maturity of digital image correlation (DIC) capabilities lead to more applications outside academic and research environments, and thus need to be integrated into common workflows used by industry testing professionals. A major part of typical vibration or modal testing is the pretest planning analysis, where decisions about excitation and sensing hardware and their locations are made. For camera-based testing, this pretest planning is commonly neglected, though it could prove of significant advantage to reduce test uncertainty and limit the time spent at the test site. The open-source software Blender allows one to perform such pretest analysis and is used in this paper to showcase its application for a DIC test. An overview of the different lighting methods is given, along with the available camera setups. A variety of methods for filtering is shown and a method to simulate fluorescent light emission is provided, using Blender's emission shader.

Data from a previously performed test campaign on a model-scale helicopter rotor in hover (90 m/s tip speed) is used for this demonstration. The test involved stereoscopic, high-speed DIC and used a complex lighting method based on laser-induced fluorescence. The test setup is recreated in the software, which includes the rotor test stand geometry, the camera positions and settings, and the lighting. The discussion focuses on DIC lighting techniques and their implementations in Blender. A good match between the real-world images and the rendered images can be created, provided sufficient information about the light power and camera sensitivity is available, thereby enabling Blender's use for test planning of camera-based experiments such as DIC or photogrammetry.

Keywords Blender · Digital Image Correlation · Rotor · Structural Dynamics · Test Planning · Laser

Introduction

Camera-based test methods, such as photogrammetry, digital image correlation (DIC), or even pressure-sensitive paint (PSP) measurements, require a substantial amount of experimental effort. This starts with handling the transport of expensive equipment such as cameras, lenses, filters, lights, etc., and then requires a considerable effort in creating a test setup that achieves all the desired outcomes, such as proper focus, illumination, field of view, etc. Depending on the type of test, surface preparation of the test specimen is also commonly needed. During the test, camera vibrations and image noise influence test data and calibration, and camera positions need to be adjusted to account for glare and other unwanted lighting effects. All of these components need to be handled by the test engineers, commonly working under tight schedules in expensive facilities such as wind tunnels. This is especially true in big national laboratories or at customer sites. For this reason, this paper investigates the use of a simplified digital twin of the test setup, to enable some aspects of test planning a priori of the test. The test setup, including the test specimen, the light source(s), and the camera(s), are simulated in a virtual environment. Engineers can then use this digital setup to experiment and find optimal values for, for example, camera positions, lamp power, or camera settings, thus saving time during the actual test campaign.

The creation of such a virtual environment requires software that can handle all of these different aspects. Since the early days of rising computing power and 3D graphics, engineers have used software to render photorealistic images.

As early as 1982, Potmesil and Chakravarty added lens and aperture effects to simulate depth of field and improve the

Marc A. Eitner
ATA Engineering Inc.San Diego, CA
e-mail: marc.eitner@ata-e.com

Jorge Alejandro Ricaurte · Jayant Sirohi
The University of Texas at AustinDept. of Aerospace Engineering
e-mail: jorge.ricaurte@austin.utexas.edu; sirohi@utexas.edu

Janko Slavic et al. (eds.), *Computer Vision & Laser Vibrometry, Vol. 6*, Conference Proceedings
of the Society for Experimental Mechanics Series, https://doi.org/10.13052/97887-438-0151-1_7

realism of their rendered images [1]. Forty years later, a variety of 3D computer graphics software is available with simple user interfaces to allow even beginners to render photorealistic images. One of these tools (among many others) is the open-source software Blender [2], which is the focus of this paper.

The open-source software Blender [2] is a tool intended for artists to use advanced 3D computer graphics technology to create photorealistic images and movies. It is packed with features that allow character rigging and animation, design, texturing, modeling, and even physical simulations, ranging from rigid body dynamics to fluid simulations. While the original intent was artistic, Blender's versatility and ease of access has led to applications far outside the scope of its original design.

Blender's first application to DIC was by Rohe in 2019 [3], which introduced the structural dynamics community to the advantages of the software, and showcased its strength as a virtual test simulator and test planning software. Blender's Python-based programming interface allows automation of mesh deformations and enables deformation reconstruction even at sub-pixel levels. In this paper, the focus will be placed on the available lighting methods in Blender, that can be used to recreate the illumination conditions commonly used in camera-based experiments such as DIC or photogrammetry.

Blender Overview

Blender's primary use is to render high-quality images of computer-generated environments. Models can be imported in a variety of formats (obj.,stl.,abc.,. . .) that describe surface meshes. Additionally, Blender contains a modeling tool, where surface meshes can be created. Textures can be applied to the surface meshes through a variety of methods. Those textures can be imported from online sources, such as Poliigon [4], or can be created within Blender using a combination of image textures and shaders. Objects can be animated on a frame-by-frame basis to allow rendering of moving scenes. Blender is based on the Python programming environment and conveniently allows automation of task through scripting interface. There is also a post-processing step for rendered images, the compositing feature, where filters and other effects can be applied to images. Blender uses a global illumination scheme, where a path tracing algorithm accounts for surface illumination from direct and indirect light. Such computations are expensive to render, but Blender supports graphics process unit (GPU) rendering, making the computations orders of magnitude faster than if central processing unit (CPU) rendering is used.

Blender'S Camera Model

Cameras are created in Blender as objects with user-defined attributes. Blender supports orthographic, panoramic, and perspective cameras, with the latter being of primary importance. The perspective cameras are based on the pinhole model, where the aperture is a single point and the relation between the image coordinates (u, v) and the real-world coordinates (x_w, y_w, z_w) of an object in the field of view are described by the linear relation:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = [K]\,[R|t] \begin{pmatrix} x_w \\ y_w \\ z_w \\ 1 \end{pmatrix} \tag{1}$$

The camera matrix $[K]$ describes the intrinsic parameters of the camera, such as focal length, pixel resolution, and sensor size. The matrix $[R|t]$ describes the extrinsic parameters and contains the rotation angles and translations of the camera with respect to the world coordinate system origin, i.e., the camera's position and orientation in space. In Blender, the user directly defines all the camera intrinsic and extrinsic parameters. In a real-world experiment, the camera parameters are obtained through a calibration routine, where typically images of a plate with a distinct pattern (such as a checkerboard pattern) are taken, while the calibration target is translated and rotated. Such a calibration routine can also be performed in Blender, by the user importing a calibration target and imaging it with cameras in multiple positions and orientations. Those calibration images can then be exported and analyzed with commercial or open-source software [5] to obtain the matrices $[K]$ and $[R|t]$. A primary use for this specific process is the development and testing of calibration algorithms.

The settings for the camera are shown in Fig. 1. Depth-of-field effects can be activated by the user defining an f-stop value, along with a focal distance, which is either an actual distance or an object in the scene that is to be in focus. The focal length of an ideal convex lens is set by the user, as well as the sensor size (in mm). The resolution of the camera is that of the rendered images, and is set separately in the rendering menu. The location and rotation of the camera are based on a camera coordinate system located at the lens aperture.

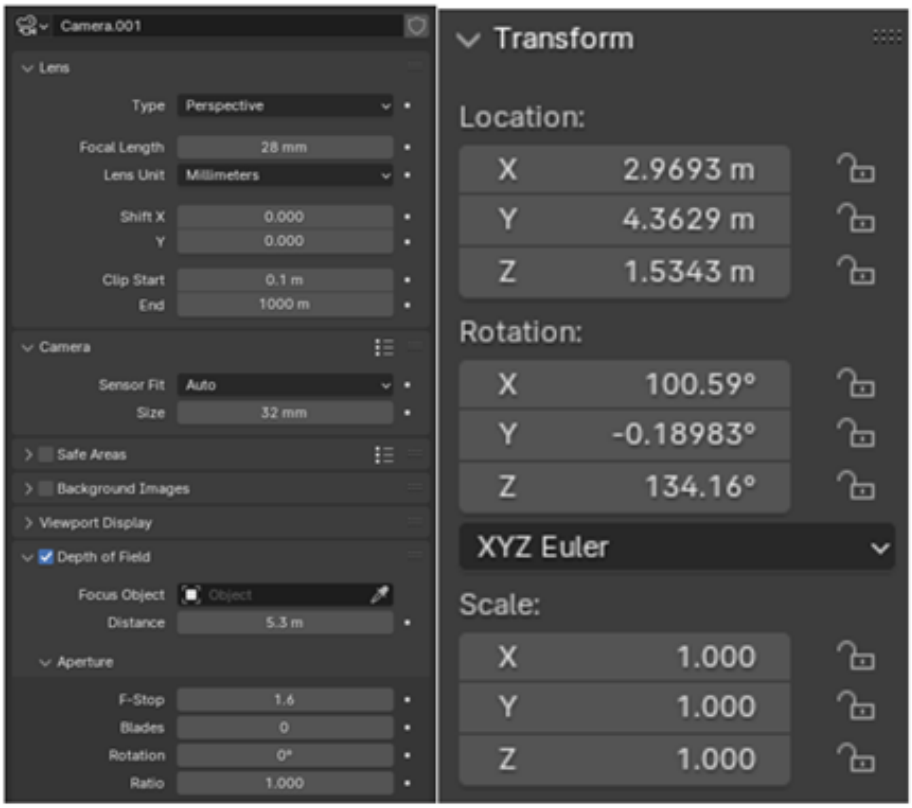

Fig. 1 Camera settings.

Note that the camera settings do not allow for effects from non-ideal lenses, such as barrel distortions or chromatic aberrations. Chromatic aberrations, if needed, can be added as a postprocessing step, using Blender's compositing feature, Lens Distortion Node. Radial distortions that are positive (pincushion distortion) or negative (barrel distortion), can be set using Movie Distortion Node, which allows the user to define the three radial distortion coefficients commonly used in most calibration routines. Alternative distortion models implemented in Blender are the Brown-Conrady model [6], which contains both radial and tangential distortions.

Lighting In Blender

From a test engineer's point of view, arguably the most difficult aspect of taking high-quality images is lighting. Setting the focal distance is typically straightforward; the test specimen needs to be in focus. Shutter speed, exposure length, ISO settings, and filters are in turn all dependent on the available light and are typically chosen in a time-consuming and trial-and-error manner. It is thus critical to have some idea of how much light is needed or available to choose appropriate settings and equipment.

Light sources can be added to a Blender scene as one of four objects:

1. Point: Omnidirectional point light source
2. Sun: Constant direction parallel rays light source
3. Spot: Directional cone light source
4. Area: Directional area light source

These objects can be placed in a scene or attached to a physical object. Common light sources used in experiments are LED panels. An example of such a panel is shown in Fig. 2. The square area light is attached to a simplified representation of the LED panel. A physical object attached to the light source helps to find appropriate space in the laboratory, considering the actual dimensions of the light.

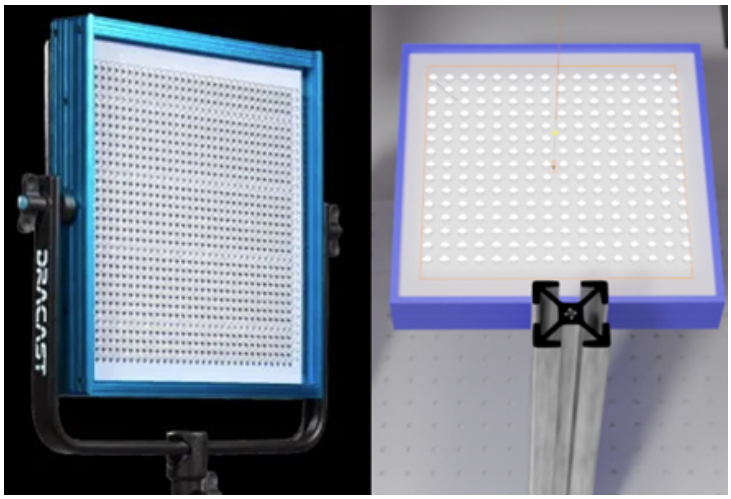

Fig. 2 Representation of area light in Blender. (Left) image of Dracast LED1000 Plus Series, 60 W power; (Right) simplified Blender representation.

Table 1 Suggested power values for typical lights [7]

Real-World Light	Power	Suggested Light Type
Candle	0.05 W	Point
800 lm LED bulb	2.1 W	Point
1000 lm light bulb	2.9 W	Point
1500 lm PAR38 floodlight	4 W	Area, disk
2500 lm fluorescent tube	4.5 W	Area, rectangle
5000 lm car headlight	22 W	Spot, size 125°

An important aspect to get right is the power setting of the light source, as this is not the electrical power needed by the light, but rather radiant power or radiant flux. The problem is that off-the-shelf lamps typically do not advertise the radiant power. Instead, consumer lamps often use the radiant flux in lumen, which accounts for the sensitivity of the human eye, as their measure of light power. The relation between radiant power and luminous flux depends on the type of lamp; a few examples for common light sources are provided on the Blender website and shown here in Table 1, which lists the appropriate radiant power settings and light type in Blender. The most common light sources for DIC experiments performed in laboratories are white LEDs. As a first approximation, the 800 lumen LED bulb listed in Table 1, can be scaled approximately to account for stronger lights, such as LED panels. The efficiency of LEDs varies from manufacturer to manufacturer and also depends on the quantity and type of single LEDs used in the lamp.

Exposure Settings

For a real camera image, the brightness of the image (the pixel value) is determined by four factors: (**??**) the light reflected and emitted from the imaged object, (2) the aperture, (3) the shutter speed, and (4) the sensor gain (ISO value). As discussed in the previous section, the light environment can be well simulated in Blender, by the user modeling the light sources with appropriate power settings as well as adjusting the surface texture of the imaged object. Blender's aperture and shutter speed settings have influence on the depth of field and motion blur, but they do not impact the image brightness (as opposed to reality). There is also no real ISO setting to adjust the image sensor gain. Instead, the image brightness can be controlled with a single Exposure setting (Render options → Color Management → Exposure). This exposure value A controls the relation between the initial rendered image brightness I_{render} and the output brightness I_{out} as follows:

$$I_{out} = I_{render} \times 2^{A} \tag{2}$$

The use of this setting is demonstrated with a simple setup shown in Fig. 3. The Dracast LED light was used to illuminate a flat panel showing a calibration pattern. A Phantom Miro M310 camera was used with a 28 mm focal lens and imaged the object with a variety of shutter speeds and aperture settings. Note that the ratio of light that reaches the sensor is approximately divided by 2 going from left to right in the figure. The ISO setting of the camera remained constant for all these

images. Since Blender does not have an ISO setting, the brightness of the darkest image (f/8, 1/10 s) needs to be matched manually. This was achieved with an exposure value of $A = 1.15$. By consecutively increasing the exposure value by one, which doubles the image brightness, the rendered image brightness matches the real-world images. Note that some of the brighter images contain saturated pixels, which complicates the visualization in Fig. 3; however, the concept of brightness adjustment and matching real-world camera settings still holds.

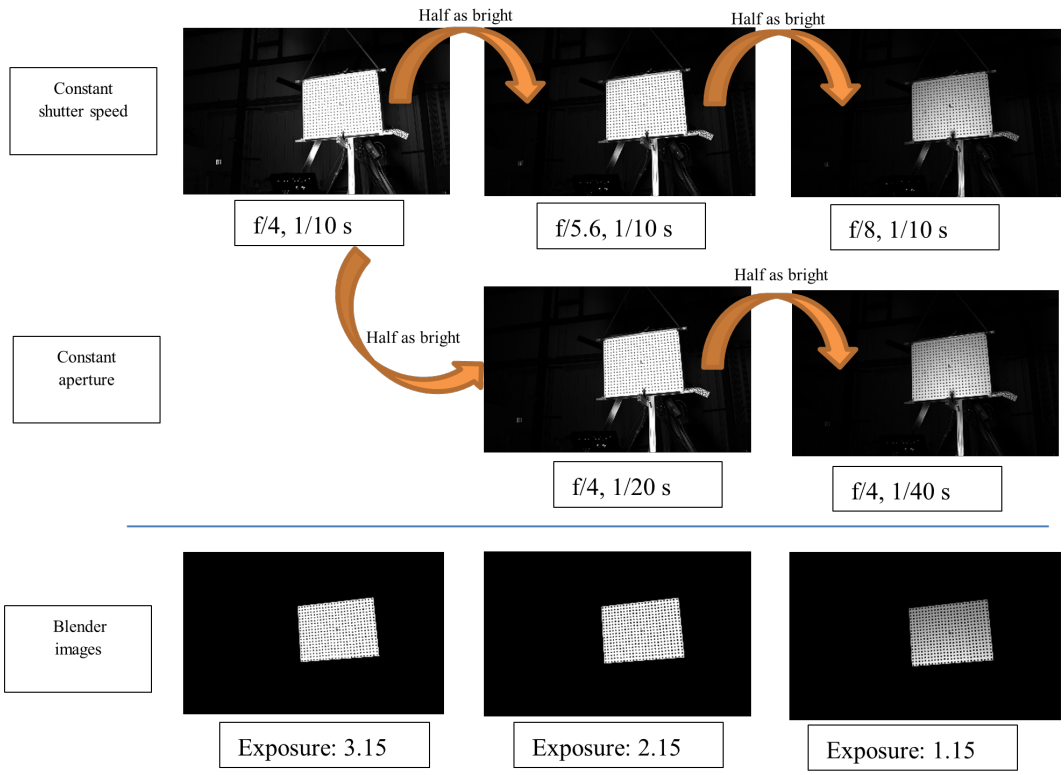

Fig. 3 Demonstration of Blender's Exposure value to match real-world images.

Filters in Blender

In many DIC/photogrammetry experiments, there is an insufficient amount of light available and it is undesirable to use lens filters. There are, however, multiple scenarios in which filter lenses might be needed for an experiment. One of these cases occurs when the surface of the imaged object is painted with fluorescent paint, which emits light in a specific narrow-band wavelength when excited. An example of such a test is shown in Fig. 4, where discrete markers are painted on the exit of a small-scale rocket model. The model is illuminated with a blacklight, which optically excites the paint, making it emit orange light. The contrast of the resulting images (marker to background) can then be greatly improved by using a filter lens on the camera. This can be done in two ways: (1) A low-pass filter lens can be used to remove the high-wavelength light (purple light emitted from the lamp), or (2) a band-pass filter lens can be used to only allow the fluorescent (orange) light emitted from the markers to reach the camera sensor. Either way, this method can lead to images with high contrast, which is desired to improve the DIC/photogrammetry process.

In Blender, many different methods of filtering specific light from an image exist. A simple method that will be demonstrated in the next section is to separate the initial render into its red, green, and blue channels. This is effective when dealing with light sources that are dominant in one of these channels, i.e., narrowband sources.

A second class of optical experiments that suffer from unwanted illumination occurs in hypersonic and high-temperature testing, when the model temperatures or that of the surrounding gas become so high, that their radiation becomes the dominant source of light. In such a case, a neutral-density filter needs to be employed, which is a broadband filter that cuts out a certain percentage of the total incoming light. These neutral density filters can most conveniently be simulated by adjusting the previously discussed Exposure value, which results in an overall reduction of brightness. For example, the addition of a

neutral density filter with optical density $d = 3$ results in a 1000x smaller image brightness, based on the relation:

$$\frac{I_{transmitted}}{I_{incident}} = 10^{-d} \tag{3}$$

To achieve the same reduction in brightness, Blender's current exposure value A_0 needs to be decreased by a value of approximately 10, since

$$\frac{I_{out}}{I_{render}} = 2^{A_0-10} = 2^{A_0} \times 2^{-10} \approx 2^{A_0} \times 0.00 \tag{4}$$

For dealing with fluorescence or high-temperature light emission Blender's primary shader allows to assign an "Emission" value, which turns the material into a light source. Similar to the light sources discussed in the previous section, the radiant power of the material can be set in W/m^2.

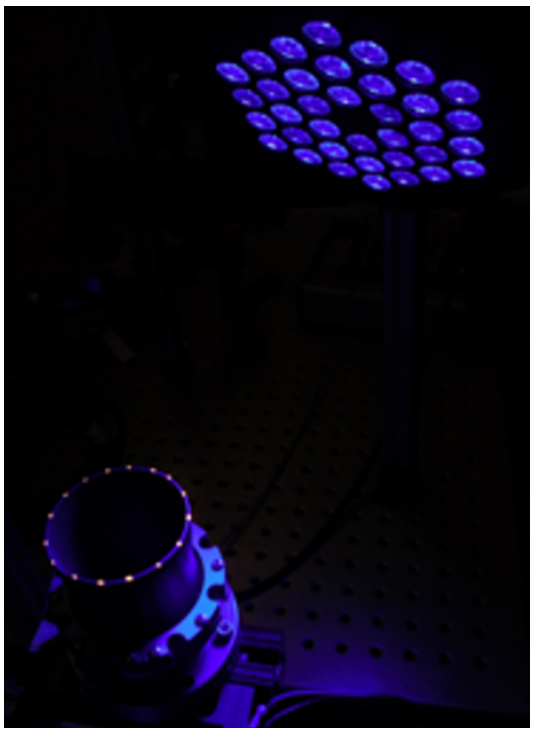

Fig. 4 Use of fluorescent paint in point tracking of rocket nozzle [8].

Application to a Dic Experiment of a Rotor Blade

The techniques discussed up to now are demonstrated on a real-world experiment, in which the vibration of a rotating helicopter blade model is measured using stereoscopic DIC. This specific experimental campaign was performed prior to the Blender simulations, which means the simulations are adjusted to match the experiment. The objective of this section is to showcase in what ways the Blender simulated experiment can represent an actual real-world experiment, and where its limitations are.

The experimental setup and its virtual reconstruction in Blender are shown in Fig. 5. The setup shows two high-speed, grayscale cameras (Phantom Miro M310), with an f-stop of 2.8, a 28 mm Nikon focal lens, and an exposure time of 0.5 ms. The cameras image a rotating rotor blade with tip speed 90 m/s.

An ND:YLF solid state laser was used in pulse mode to illuminate the rotor during rotation. The emitted light has a wavelength of 527 nm (green), and the firing rate was synced to the rotor to obtain images at precise azimuthal locations across a rotation. This is then filtered out using a 540 nm cut-off high-pass filter lens. The surface of the blade is painted with an orange spray paint with fluorescent properties, which emits orange light when excited [9]. Note that this is the same paint

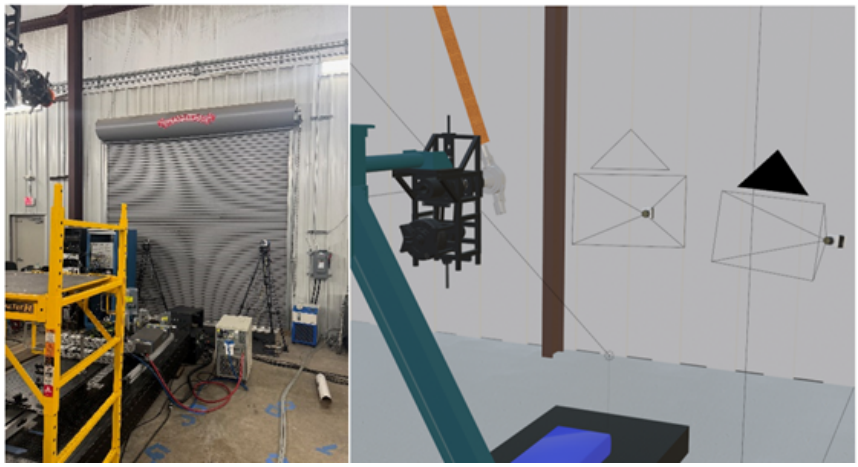

Fig. 5 View of laboratory (left) and recreation in Blender (right).

used on the rocket nozzle shown in Fig. 4, except this time it is excited by the high-power laser. Speckles of black paint are then added to the blade, which creates the random pattern needed for DIC.

In Blender, the CAD file of the rotor test stand is imported and a simplified laboratory environment is created. The camera parameters and positions are matched to the real world. The speckled orange pattern is mapped to the blade surface as an image texture, whereby the black dots are designated as the background (alpha) layer. An emission shader is then applied to the speckle pattern, resulting in an orange glow. This is to approximate the fluorescent emission. A cone light source that emits green light is added to the scene to represent the laser. The cone has a beam angle of 40° and a very strong "Blend" value, which means the intensity toward the edge of the cone is much weaker than along its centerline. This is done to achieve the effect that the blade is hardly illuminated near its tip. The rendered color image along with the real-world grayscale image are shown in Fig. 6. The rendered color image needs to be filtered appropriately to match the real-world image. Unfortunately, this is not as simple as removing all the green light from the scene (as is done in reality with the filter lens). The resulting image would show the entire blade emit light evenly. However, the fluorescent effect only acts on the parts of the blade that are illuminated (i.e., optically excited) by the laser light. This effect is achieved by first separating the red and green channels of the rendered image, and then multiplying them with each other. This effectively creates a spatial weighing of the red channel, which is mostly responsible for the orange emission. The entire process is summarized in Fig. 7.

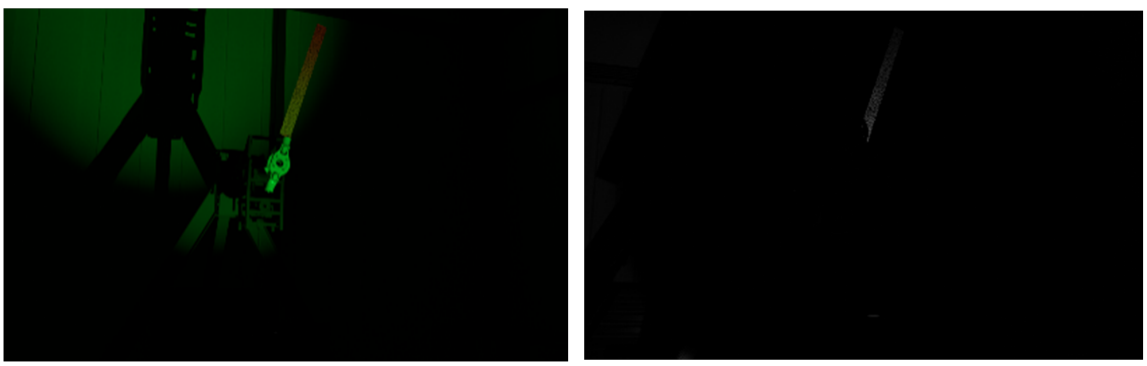

Fig. 6 Rendered color image (left) and real-world grayscale image (right).

At this point several issues need to be highlighted. First of all it can be seen that the rendered image compares very well with the real-world image, both in perspective and intensity. This was however only achieved by manually adjusting the radiant power of the emission shader (which creates the orange fluorescent light), as well as the radiant power of the green laser source. While the laser power settings are known, there is no information on the paint, that would allow to quantify the fluorescent emission. If such experiments are to be repeated, the fluorescence should be quantified to allow this process

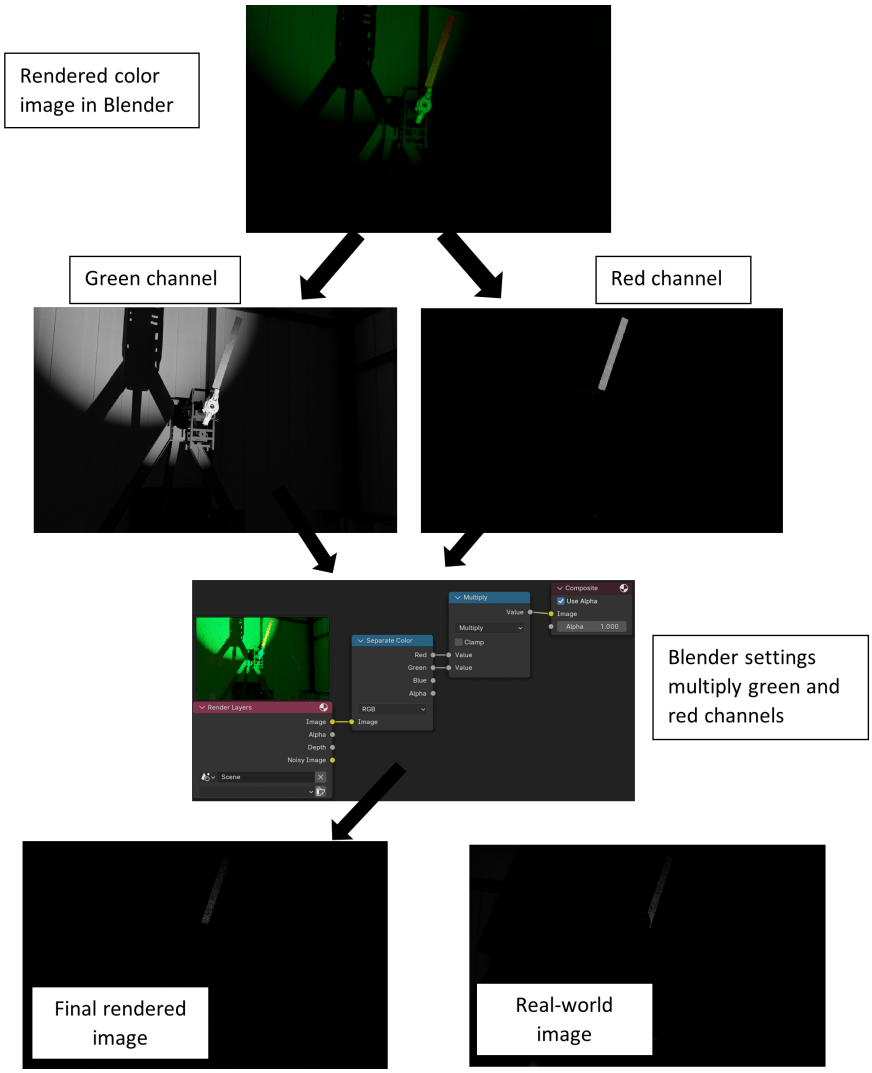

Fig. 7 Postprocessing of rendered color image to match real-world grayscale image.

to become more rigorous. An additional uncertainty comes from the laser cone intensity distribution, which was manually adjusted to match the real-world intensity fall off towards the blade tip. Again, this beam profile could be further quantified using a simple photo-diode in combination with the laser and diffusive lens.

Conclusion

Several techniques were presented that can be used to simulate DIC/photogrammetry experiments using the open-source software Blender. The focus was placed on lighting, and it was shown that Blender supports a variety of tools to model natural (sunlight) and artificial light sources (lamps). Blender's emission shader allows to even approximate effects such as fluorescence, which was demonstrated successfully on a real-world rotorcraft experiment. More advanced methods involving the emission of surfaces and even plasmas in high-temperature experiments, such as wind tunnels or plasma torches, could technically also be modeled using this same approach. By taking reference images of a given camera and light source, users can find appropriate exposure settings within Blender, which can then be used for reference in future simulated experiments. With a limited amount of effort in modeling the experimental setup in Blender, valuable time can be saved during experiments, which is often of primary importance when for test engineers performing tests outside of university laboratories, such as at national labs or customer sites. This can be achieved by optimizing the position of cameras in the scene, as well as camera settings, such as aperture, or picking a suitable lens. Furthermore, locations and power settings of light sources

can be approximated, provided sufficient information is known. This step might require some initial experimentation with the available equipment, to find for example, the relation between the camera ISO and the exposure setting in Blender. The combination of all of these steps have the potential to save valuable time during experiments performed under tight schedules.

References

1. M. Potmesil and I. Chakravarty, "Synthetic image generation with a lens and aperture camera model," ACM Transactions on Graphics (TOG), vol. 1, no. 2, pp. 85–108, 1982. W309W10080
2. The Blender Foundation, "Blender," 2024. [Online]. Available: https://www.blender.org/.W309W10080
3. D. Rohe, "Rohe, Daniel Peter. An Optical Test Simulator Based on the Open-Source Blender Software," SAND2019-12756C. Sandia National Lab, Albuquerque, NM, 2019.W309W10080
4. A. Price, "poliigon," [Online]. Available: https://www.poliigon.com/.W309W10080
5. "OpenCV," Open Source Vision Foundation, [Online]. Available: https://opencv.org/.W309W10080
6. D. C. Brown, "Decentering distorion of lenses," Photogrammetric Engineering, vol. 32, no. 3, pp. 444–462, 1966. W309W10080
7. "Blender Manual - Power of Lights," The Blender Foundation, [Online]. Available: https://docs.blender.org/manual/en/latest/render/lights/light_object.html#power-of-lights. [Accessed 2024].W309W10080
8. M. Eitner and J. Sirohi, "Modal parameter estimation of a reduced-scale rocket nozzle using blind source separation," Measurement Science and Technology, vol. 30, p. 095401, 2019. W309W10080
9. D. Uehara and J. Sirohi, "Full-field optical deformation measurement and operational modal analysis of a flexible rotor blade," Mechanical Systems and Signal Processing, vol. 133, 2019. W309W10080