IMEKO TC6 International Conference on Metrology and Digital Transformation
September 19 – September 21, 2022, Berlin, Germany

# USING BLENDER AS A TOOL TO SIMULATE 3D CAMERA BASED MEASUREMENTS

*C.Pottier* [a], *F.Eghtedari* [b], *J.Petzing* [a], *P.Kinnell* [a]

[a] Loughbrough University, Loughborough, LE11 3TU.
They are with Wolfson School of Mechanical, Electrical & Manufacturing Engineering.
E-mail : c.pottier@lboro.ac.uk, j.petzing@lboro.ac.uk, p.kinnell@lboro.ac.uk.
[b] Manufacturing Technology Centre (MTC). E-mail : fariborz.eghtedari@the-mtc.org

***Abstract*** – Blender is a free and open-source 3D animation software, that has the potential to be used as a metrology simulation tool, to build digital models that can be used in the design and optimisation of camera-based measurement systems. In this work the feasibility of using of Blender for measurement system simulation is explored. Using Blender, a simple camera-based measurement task was simulated, in which a camera is used to measure the observable diameter of a white sphere. The aim was to check if the virtual cameras created in Blender can perceive and measure the sphere in the same manner as real cameras when repeating the procedure in a closely matching real-world experimentation.

***Keywords***: Blender, Camera calibration, Digital model, MATLAB, Pinhole camera, Software accuracy

## 1. INTRODUCTION

Cameras are routinely used to monitor large 3D spaces and increasingly the data they provide is being used to measure the position or pose of objects in the space. This can then be used to establish a digital model of the space which can used for a variety of coordination, control or supervisory tasks, with many potential applications, e.g., manufacturing process control, human-robot interaction, and indoor mobile robot guidance. Camera-based measurement systems are cost effective and flexible, and therefore can be applied to a wide range of measurement challenges. However, when designing a camera system factors such as the number of cameras needed, suitable mounting locations for cameras, the camera line of sight or field of view (FOV) and associated occlusions, access to calibration artefacts or other fiducial reference markers, and the influence of environmental factors such as light or reflections, must be carefully considered, as all of these factors will affect performance. Often this requires time consuming physical testing, where cameras and placed, test images are acquired and analysed, and if necessary, the setup is modified so the test can be repeated.

A more preferable approach would be to design and test the camera network using a digital model. A digital model is defined as a computerised replica or representation of an object. Consequently, a digital model of multiple camera systems, and the working environment, could be generated that allows simulation of the camera systems and optimization of camera and object location within the environment.

The aim of the work presented is to investigate the suitability of using Blender software to simulate camera-based measurements in a 3D space. Blender was chosen as the modelling environment because it is a free and open-source 3D computer graphics software toolset used for creating animated films, visual effects, interactive 3D applications, virtual reality, etc, as such it combines the functionality to model the interactions of scenes with light sources, and generate simulated images based on modelled cameras. Blender is a pertinent choice due to its rendering engine based on ray-tracing and the python scripting option. Ray tracing allows the modelling of light sources, and their interactions with 3D objects in a visually realistic way, to create photorealist scenes. The Python scripting capability, based on API (application programming interface) commands, can be used to customize the application, and write specialized tools, bringing the freedom to manipulate and automate the scene created. This enables many variants of a simulated scene and the associate network of cameras and light sources, that form the measurement system, to be automatically created in a parametric way. In this way it is interesting for the automated exploration of design choices when creating a camera-based 3D measurement system. Recent digital applications relevant to metrology have been developed using Blender, such as BALINDER [1] and BlenderProc [2]. However, as generally Blender is not typically used for metrology simulation applications, there are no evaluations of Blender for this purpose. This work presents initial steps taken to explore the suitability of using Blender to simulate basic camera-based measurement tasks.

## 2. METHODOLOGY

To evaluate Bender, a simple test measurement was defined, and an equivalent measurement setup was then simulated in Blender. The test measurement was to measure the diameter of a sphere using a Raspberry Pi V2 camera, located at a range of distances to the sphere.

A 3D sphere was chosen because it allows us to test Blender's ability to simulate the interaction of light with 3D objects. A sphere is also convenient because a 3D sphere is always seen as a 2D circle, regardless of the orientation in which it is viewed. This feature compensates for the imperfections of the configuration, allowing this experiment to be reproduced without adding new sources of error to the physical misalignment of the camera. However, the sphere is an interesting object because its geometry introduces consequences for the measurement of the sphere's diameter using a 2D camera. In Figure 1, point C represents the optical centre of the camera, and points A and B are the intersection points between the ends of the FOV and the circle. These ends are tangent to the circle.

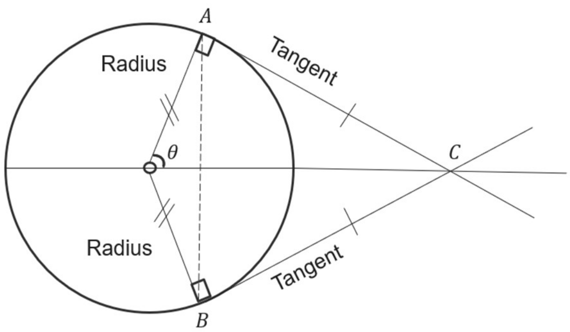


Figure 1: Relationship between a sphere and camera field of view

The segment [AB] is not equal to the real sphere's diameter, but represents the diameter perceived by the camera because these points are the interception points between the camera's FOV, and the sphere. The angle ϴ will increase with increasing distance between the sphere and the camera. As the distance of the camera from the sphere increases to infinity, then the angle ϴ tends towards 90˚, and the difference between then real sphere's diameter and the diameter observed by the camera tends towards zero. Therefore, the outcome criterion for the experiments was a comparison between deviation of the measured observable sphere diameter and maximum observable diameter based on the distance of the camera from the sphere. The theoretical observable sphere diameter was calculated using the geometry shown in Figure 1, and Pythagoras's theorem.

In Blender, the Raspberry Pi V2 camera was simulated using a pinhole camera model, with a focal length of 3.03 mm, an image resolution of 1,640 x 1,232 pixels, a sensor size of 3.68 mm, and with square pixels of 1.12 µm square.

### *2.2. Experiment*

The experiment consisted of measuring the diameters and centres of a 0.09 m diameter white virtual sphere with MATLAB, from pictures taken by the simulated virtual pinhole camera located at a distance varying from 0.2 m to 2 m, in steps of 0.2 m, from the sphere. The whole scene was generated in Blender through a Python script.

A MATLAB toolbox was developed to detect the sphere in the simulated picture, and return key elements, based on functions developed by MathWorks. The processing steps followed are:

- Detect the sphere in the Blender image.
- Transform the RGB image to a binary image.
- Measure the diameter of the sphere (used of the *regionprops* function).
- Calculate the diameter of the sphere.

The experimentation was then repeated in the real world, using the same MATLAB toolbox, a real camera, and a sphere with the same diameter, to compare the results obtained with the modelling environment.

### *2.3. Real system presentation*

The system used in reality is illustrated in Figure 2:

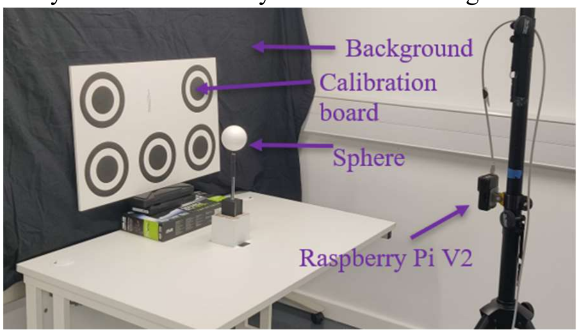


Figure 2: Real system

The real-world experiment is comprised of a white polyester sphere with a diameter of 0.09 m, measured using vernier callipers, a pinhole camera with the same characteristics as the one used in Blender were used, and located on the x-axis. A dark background was used to create high contrast between the white room and the white sphere. The experiment took place in a controlled environment with two light sources of 38.65 W (luminous flux measured with a hand-held light meter). A white table was placed under the sphere so that the light would reflect off it and illuminate the sphere from below.

To obtain an accurate value for the camera's focal length, the intrinsic parameters of the camera were established by a checkerboard calibration process. The distance between the camera and the centre of the sphere was also determined by a calibration process in which the extrinsic parameters were calculated by finding the centre of five large circles of known diameter on a calibration board (see Figure 2). The calibration board was placed behind the sphere, and a picture of both objects (separately) was taken.

### 2.4. Virtual system presentation

Figure 3 shown the scene designed in Blender.

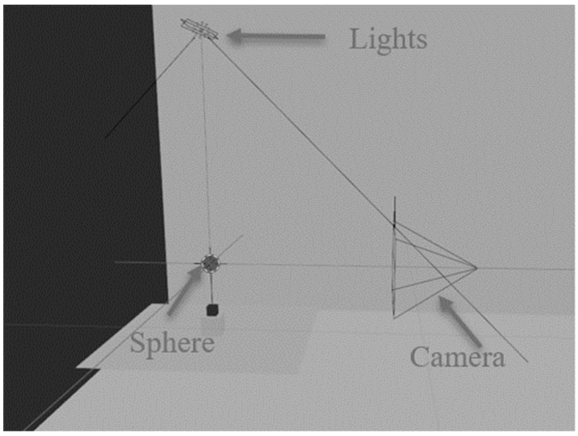


Figure 3: Blender system presentation

The Blender model is composed of the same elements that the real model is, with the same dimensions, at the same locations. The white sphere is made up from 32 polygons, noting that objects created in Blender are all based on a polygon mesh. Hence, the number of polygons represent the resolution of the sphere, consequently the more polygons the sphere is modelled with, the more "spherical" the sphere will be.

To determine the appropriate number of polygons to use to model the sphere, the same experiment was carried out with 100 and 1000 polygons. It was found that 32, 100 and 1000 polygons gave the same answer.

Two square area lights with a power of 38.65 W were placed at 1 m from the sphere, rotated by $\pm 45°$, to recreate the light diffuser in the laboratory. The square shape was chosen to match the reality, and the power measured with a light meter. The scene was placed in a cube so that the light would bounce off the wall.

In Blender, the BSDF (bidirectional scattering distribution function) main node was used to apply a texture to the different elements of the model. White elements were considered bright and light-reflective, and black elements were considered bright and light-absorbent.

### 2.5. Image processing

Pictures were taken by the real and simulated cameras at a distance between 0.2 m and 2 m, in steps of 0.2 m, and treated with the same algorithms. The only modification to the data processing was cropping the real images to further exclude unwanted environmental image artefacts. The algorithm used grey scale images, causing all image components to be black, white or grey. The black background board was put behind the sphere to recreate the Blender scene and optimise the contrast necessary for use of the MATLAB circle detection algorithm.

## 3. RESULTS

The real experiment was run three times with the process described previously. The result is shown in Figure 4.

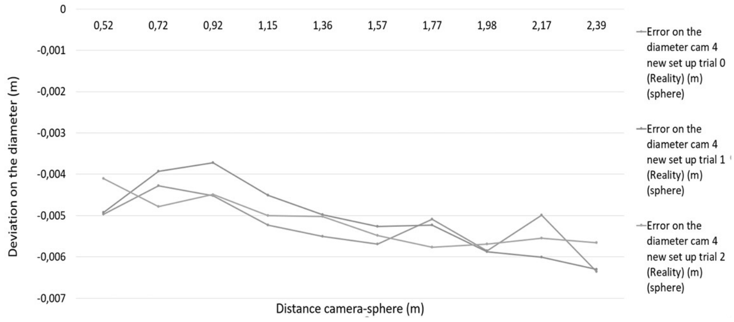


Figure 4: Deviation from theoretical observable sphere diameter in the real

In Figure 4, the x-axis represents the distance between the cameras and the sphere in metres, and the left y-axis is the difference between the theoretical and measured diameter observable in metres. The three trials give the similar results, with the same result shape, with a variation of $\sim 1\,mm$. The deviation on the sphere diameter seems to increase slightly with the increasing of the distance between the camera and the sphere.

Table 1: Deviation on the diameters for a distance between the camera and the sphere of 0.5 m and 1.36 m for the different trials.

| Trial | Camera-sphere distance: 0.5 m | Camera-sphere distance: 1.36 m |
|---|---|---|
| 0 | -4.97 mm | -5.50 mm |
| 1 | -4.25 mm | -4.98 mm |
| 2 | -4.11 mm | -5.0 mm |

The Blender answer is giving by Figure 5.

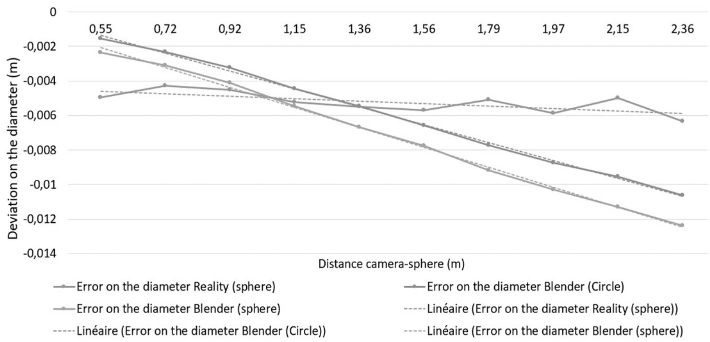


Figure 5: Deviation from theoretical observable sphere diameter in the real and virtual environment

To compare the physical results with the Blender simulated measurements, the results from the physical test are plotted together with the blender simulation results for the same measurement in Figure 5. Due to the ability of Blender to allow perfect geometric alignment of the camera and measurement objects, in addition to the white sphere, a white circle was also imaged at each of the distances considered. The Blender results for the circle and sphere measurements were both similar, indicating that the more complex light interactions between the 3D geometry of the sphere and the camera, were not a source of significant error in Blender. It can however be seen that in Blender the impact of distance of the camera from the target object on measurement accuracy is more sever, in the case of both the sphere and circle.

Noted that this experiment does not attempt to correct the geometric bias illustrated in Figure 1. This experiment aims to show the impact of different sources of errors on both measurements, as well as the differences between the digital and real systems.

Table 2: Deviation on the diameters for a distance between the camera and the sphere of 0.5 m and 1.36 m for the circle and the sphere

| Artefact | Camera-sphere distance: 0.5 m | Camera-sphere distance: 1.36 m |
|---|---|---|
| Circle | -1.53 mm | -5.46 mm |
| Sphere | -2.35 mm | -6.69 mm |

With Table 2, we conclude that the geometry brings some errors. The variation between the results with the sphere and the circle is around 1 $mm$.

Moreover, the results in Blender are better than in reality between 0.5 m and 0.9 m, before having the same result than the reality at 1.15 m for the sphere and 1.36 m for the circle, and finally worse than the reality between 1.56 m and 2.36 m. Table 3 summarizes this evaluation:

Table 3: Comparison between the real and virtual results for three different camera-sphere distances.

| | Camera-sphere distance: 0.5 m | Camera-sphere distance: 1.36 m | Camera-sphere distance: 1.79 m |
|---|---|---|---|
| Real | -4.97 mm | -5.50 mm | -5.09 mm |
| Blender circle | -1.53 mm | -5.46 mm | -7.74 mm |
| Blender sphere | -2.35 mm | -6.69 mm | -9.17 mm |

Through these results it is demonstrated that Blender is not perfect, but generally it respects the laws of optical physics, and the geometry illustrated in Figure 1 is reasonably simulated.

The difference between the physical and simulated measurements may come from the resolution of the image, because when the sphere is further away from the camera, it is composed of fewer pixels. In addition, the sphere is blurred, which leads to detection errors.

It is clear that the impact of distance from the camera on the measurement is more sever in Blender, this different might come from the simulation of the camera, light and texture set up.

In this experiment, the ability of Blender to mimic reality was examined. Blender is a 3D animation software, testing its capabilities can be quite difficult as it is a black box, it is impossible to have access to all the information that makes up the software.

In this article, the approach of comparing the two environments was taken to check whether Blender can be used as a virtual model for a digital twin. As the digital twin is composed of three elements, a physical object, its digital representation qualified as a twin because of its accuracy, and two communication loops between them (data and information/process transmission) [6], the mimicry capabilities from the observer's point of view are evaluated.

According to the results found in this study, further investigations about camera and environment modelling will be done to enhance the model accuracy.

## ACKNOWLEDGMENTS

This work was supported by the Manufacturing Technology Center (MTC) in Coventry (UK), and Loughborough Universit in Loughborough (UK)

## REFERENCES

[1] Reitmann, S.; Neumann, L.; Jung, B. BLAINDER—A Blender AI Add-On for Generation of Semantically Labeled Depth-Sensing Data. Sensors 2021, 21, 244. https://doi.org/10.3390/s21062144

[2] Denninger, Maximilian & Sundermeyer, Martin & Winkelbauer, Dominik & Zidan, Youssef & Olefir, Dmitry & Elbadrawy, Mohamad & Lodhi, Ahsan & Katam, Harinandan Teja. (2019). BlenderProc.

[3] B. Bartos, K. Stein, "Visualizing simulated temperatures of a complex object calculated with FTOM using open source software (BLENDER)," Proc. SPIE 10794, Target and Background Signatures IV, 1079407 (9 October 2018); https://doi.org/10.1117/12.2325380

[4] Brandão E, Morgado G, Fonseca WD. A ray tracing engine integrated with Blender and with uncertainty estimation: Description and initial results. Building Acoustics. 2021;28(2):99-118. doi:10.1177/1351010X20964758

[5] Raspberry PI <https://www.raspberrypi.org/documentation/accessories/camera.html>

[6] David Jones, Chris Snider, Aydin Nassehi, Jason Yon, Ben Hicks, Characterising the Digital Twin: A systematic literature review, CIRP Journal of Manufacturing Science and Technology, Volume 29, Part A, 2020, Pages 36-52, ISSN 1755-5817, https://doi.org/10.1016/j.cirpj.2020.02.002.