Performance Evaluation of Ground AR Anchor with WebXR Device API

Daehyeon Lee 1, Woosung Shim 1, Munyong Lee 1, Seunghyun Lee 2, Kye-Dong Jung 2 and Soonchul Kwon 3,*

1 Department of Smart System, Kwangwoon University, Seoul 01897, Korea; dleogus13813@kw.ac.kr (D.L.); sws5779@gmail.com (W.S.); ansdyd0803@kw.ac.kr (M.L.)
2 DIngenium College, Kwangwoon University, Seoul 01897, Korea; shlee@kw.ac.kr (S.L.); gdehung@kw.ac.kr (K.-D.J.)
3 Graduate School of Smart Convergence, Kwangwoon University, Seoul 01897, Korea
* Correspondence: ksc0226@kw.ac.kr; Tel.: +82-2-940-8637

Abstract: Recently, the development of 3D graphics technology has led to various technologies being combined with reality, where a new reality is defined or studied; they are typically named by combining the name of the technology with “reality”. Representative “reality” includes Augmented Reality, Virtual Reality, Mixed Reality, and eXtended Reality (XR). In particular, research on XR in the web environment is actively being conducted. The Web eXtended Reality Device Application Programming Interface (WebXR Device API), released in 2018, allows instant deployment of XR services to any XR platform requiring only an active web browser. However, the currently released tentative version has poor stability. Therefore, in this study, the performance evaluation of WebXR Device API is performed using three experiments. A camera trajectory experiment is analyzed using ground truth, we checked the standard deviation between the ground truth and WebXR for the X, Y, and Z axes. The difference image experiment is conducted for the front, left, and right directions, which resulted in a visible difference image for each image of ground truth and WebXR, small mean absolute error, and high match rate. In the experiment for measuring the 3D rendering speed, a frame rate similar to that of real-time is obtained.

Keywords: hit-test; performance evaluation; WebXR Device API; XR

1. Introduction

Web eXtended Reality (WebXR) is a standard used to support the rendering of a 3D scene with appropriate hardware to present a virtual environment in a web environment or to add graphic images to the real environment. In other words, it is used to provide extended reality in a web environment. In the past, extensive studies for supporting virtual environments in a web environment have been performed, such as WebVR application programming interface (API) and WebAR API [1,2]. However, because these APIs were developed to service a particular virtual environment, supporting other virtual environments became difficult and modifications were time-consuming. According to Mozilla, some browsers may still support WebVR API, but they have already been removed from the relevant web standards, are in the process of being removed, or are only being maintained for compatibility purposes [3]. Therefore, this paper used a new WebXR Device API including these APIs, and evaluated how well it performs through experiments.

The World Wide Web Consortium (W3C) group released the Web eXtended Reality Device Application Programming Interface (WebXR Device API) in 2018. This interface allows instant deployment of extended reality services to any extended reality platform requiring only an active web browser. In addition, both augmented reality and virtual reality can be supported with minimal modifications. Moreover, large downloads are not required and users can immediately experience virtual environments on the website. For this reason, research on this interface is actively underway. However, the current version is...
unstable, which can cause a deviation in the anchor position when the measuring device is moved, which can reduce the stability of the anchor. When this issue occurs, the 3D model appears to be in a location other than the location specified by the user. As a result, users can be less immersed in the AR environment. Therefore, the need for research on interface performance evaluation has emerged [4].

In this study, we aimed to evaluate the performance of the WebXR Device API and conducted a performance evaluation study. It is important to determine the user could sufficiently use the API or additional development of the API is necessary and the developer is worth investing time in the development of the API.

The experiment was conducted in two different ground environments, and experiments were conducted after loading a 3D model through the WebXR Device API in a mobile web environment. The first experiment measures the camera trajectory in a web environment when the device moves after loading a 3D model. In the second experiment, after the 3D model is loaded, a real model with the same size and shape as the 3D model is created and placed in the same position for accuracy measurement through the difference image, mean absolute error, and match rate. The third experiment is a 3D model rendering speed measurement experiment.

The structure of this paper is as follows. In Section 2, the background theory is explained. In Section 3, the experimental environment and experimental methods for performance evaluation are described. In Section 4, the experimental results and discussions of a performance evaluation using the proposed method are presented. Finally, Section 5 presents conclusions, limitations, and future research plans.

2. Background Theory

Section 2 describes the definitions of the various virtual environments, the types of each environment, and the differences. In addition, the contents of the virtual environment used in this paper are explained. Section 2.1 describes eXtended reality. At this time, the difference between Virtual Reality, Mixed Reality, and Augmented Reality will be explained. Section 2.2 describes Augmented Reality. Section 2.3 describes the anchor generation method for ground detection in this paper. Section 2.4 describes the WebXR Device API.

2.1. eXtended Reality

eXtended Reality (XR) refers to human–machine interaction using all real and virtual environments created by computer technology and wearable devices. XR is a concept that encompasses the entire spectrum of the reality–virtual continuum introduced by Paul Milgram from “complete reality” to “complete virtual”. Figure 1 shows the reality–virtual continuum. Virtual Reality (VR) is a completely digital environment and Augmented Reality (AR) is close to a completely real environment, with Mixed Mixed Reality (MR) in between. In contrast, XR is defined as a concept that includes all virtual environments [5,6].

![Reality–virtuality continuum](image)

**Figure 1.** Reality–virtuality continuum.

2.2. Augmented Reality

AR is one of the types of virtual environments. It is a computer graphic technique that combines virtual objects or information with the real environment such that the object appears to exist in the real environment. AR can additionally provide information that is
difficult to obtain or is intended to be transmitted with a special purpose by synthesizing virtual objects in a real environment. Unlike VR, which presupposes a completely virtual environment, AR provides augmentation of information based on reality and aims to further increase the effect of reality by combining virtual objects with the real environment. In addition, AR serves to provide additional information necessary for the actual environment. By projecting virtual information on an image that a user is viewing, the distinction between the real environment and the virtual environment is blurred. According to Ronald Azuma, AR is defined as a combination of real and virtual images, enabling real-time interaction in a three-dimensional space [7–10].

2.3. Using Anchor for Ground Detection

To provide AR to the desired location, holding an accurate anchor on space is crucial. An anchor is a reference point appearing at a position to float a virtual object in a plane that can be easily indicated in a spatial coordinate system. When an object needs to be directional in AR implementation, it is a concept that includes x, y, and z coordinates in space and the rotation value of each axis. The methods for holding an anchor are largely separated into four types. The first is Image Anchor, which recognizes the image as a marker. The next is Ground Anchor, which recognizes the floor surface and implements it. The next one is point Cloud Anchor, which is implemented based on the characteristics of the point cloud. The last one is Object Anchor, which is generated by object recognition. Figure 2 shows a flow chart of how to recognize ground in a virtual environment. When the camera is looking at the real environment, if the camera is moved, point cloud information about the real environment is extracted using a Simultaneous Localization And Mapping (SLAM) algorithm. After that, the point cloud extracted by the RANdom SAmple Consensus (RANSAC) algorithm is sampled. A mesh is generated for the sampled point cloud. Thereafter, a normal vector orthogonal to the mesh is calculated, and a normal vector is a dot product for an adjacent mesh to check whether it has a constant normal vector. Grounding is checked by specifying a threshold for the dot product normal vectors. Finally, the values identified as the ground are projected onto the screen using 2D to 3D projection [11–17].

![Figure 2. Ground detecting flowchart.](image)

2.4. WebXR Device API

WebXR Device API was developed to serve XR in a web environment. It was released by the W3C group in 2018 and is still under development; the latest versions of the WebXR Anchor Module and WebXR Augmented Reality Module used in this study were released in June 2020. Previously, separate APIs were developed for WebVR [18,19] and WebAR [20]; moreover, WebVR was designed to support only VR. However, the WebXR Device API supports both VR and AR on the web. Support for the AR function was added by the WebXR Augmented Reality module; later, the development of the WebVR API was stopped.

WebXR has been extensively compared with Khronos Group’s Open eXtended Reality (OpenXR) [21] Software Development Kit (SDK). OpenXR is an open, royalty-free standard for accessing VR and AR platforms and devices. It was developed by a working group managed by the Khronos Group consortium. The interim version of this standard, OpenXR 1.0, was released in July 2018, allowing developers to provide feedback. OpenXR handles the same basic functions as the WebXR Device API for native applications. Therefore, WebXR and OpenXR seem to have a similar relationship with the Web Graphics Library (WebGL) [22,23] and Open Graphics Library (OpenGL) [24]. Both WebGL and OpenGL are open source libraries published and managed by the Khronos group. In addition, WebGL was developed in a form that translated the API of OpenGL 3.0 into Javascript. In contrast,
OpenXR is published and managed by the Khronos group, while WebXR is published and managed by the W3C Group. These two APIs are other APIs being developed by different standards organizations, and many of the same concepts are expressed in different ways on shared subjects. However, when building an XR service, implementing the functions of WebXR using OpenXR becomes possible.

3. Materials and Methods

Section 3 describes the materials and methods used in this paper. Section 3.1 describes the hit-test method to create the ground anchor. Section 3.2 describes the setup for the experiment. Section 3.3 describes the experimental environment. This includes the device used in the experiment, web browser, 3D model, etc., and Section 3.4 describes the experimental methods.

3.1. Hit-Test

Hit-test (hit detection, picking, or picking correlation) is the process of determining whether a cursor controlled by a user intersects a specific graphic object (point, line, or surface) drawn on the screen in computer graphics programming. There are various types of cursors such as a mouse cursor, an interface of a touch screen, or a touchpoint. Hit-testing can be performed on the movement or activation of a specific pointing device. The order of the hit-test used in this paper is as shown in Figure 3a. After the user accesses the web page, it checks whether it supports WebXR. If it is not supported, the page is terminated, and in the opposite case, the web page is activated. At this time, the AR button to access the immersive web environment is activated. Thereafter, when the AR button event occurs, the immersive web starts. After accessing the immersive web, it recognizes the ground and creates a ground anchor. At this time, as shown in Figure 3b, find the ground using the normal vector. This process is repeated until an anchor is created. When the Hit event occurs on the generated ground anchor, it checks whether the 3D model is loaded in the current immersive web page. If a 3D model exists, the 3D model is made invisible through the Hit event. In the opposite case, the 3D model is visualized. This process is performed every time a Hit event occurs. When a user requests termination from an immersive web page, it closes the web page.

![Figure 3. Hit-test algorithm (a) flowchart; (b) normal vector generate; (c) ground anchor specify method.](image)

3.2. Experimental Setup

Section 3.2 describes how to set up for the experiment. Section 3.2.1 describes how to setup the software. This chapter includes differences between OpenXR and WebXR, and dif-

---

**Figure 3.** Hit-test algorithm (a) flowchart; (b) normal vector generate; (c) ground anchor specify method.
ferences between WebGL and OpenGL. Section 3.2.2 describes how to setup the hardware. This chapter includes how to create a 3D model and how to create a ground environment.

3.2.1. Software Setup

In this paper, the experiment was conducted by modifying the hit-test supported by the WebXR Device API, and the location of the ground anchor initially designated was saved to synchronize the location with the actual model, and to load the 3D model of the hit-test. How to designate a ground anchor is explained in Figure 3c. The ground anchor designation method is also called ray casting, and the “rayOrigin” is designated using a virtual ray in the state of initial access to the immersive web. After that, when the device is moved, the “rayDirection” is designated using a new virtual ray, and the intersection point where the two lines meet is designated as the ground anchor. WebXR has been extensively compared with Khronos Group’s Open eXtended Reality (OpenXR) Software Development Kit (SDK). OpenXR is an open, royalty-free standard for accessing VR and AR platforms and devices. It was developed by a working group managed by the Khronos Group consortium. The interim version of this standard, OpenXR 1.0, was released in July 2018, allowing developers to provide feedback. OpenXR handles the same basic functions as the WebXR Device API for native applications. Therefore, WebXR and OpenXR seem to have a similar relationship with the Web Graphics Library (WebGL) and Open Graphics Library (OpenGL). Both WebGL and OpenGL are open source libraries published and managed by the Khronos group. In addition, WebGL was developed in a form that translated the API of OpenGL 3.0 into Javascript. In contrast, OpenXR is published and managed by the Khronos group, while WebXR is published and managed by the W3C Group. These two APIs are other APIs being developed by different standards organizations, and many of the same concepts are expressed in different ways on shared subjects. However, when building an XR service, implementing the functions of WebXR using OpenXR becomes possible.

3.2.2. Hardware Setup

Figure 4a,b shows the ground environment created for the difference image extraction experiment. It consists of a square with a side of 2 m. An arbitrary location ① is selected where the object is placed; it is 1.5 m away from location ②. For real users to use the WebXR environment, a certain range of space is required; therefore, a square environment was constructed. In addition, the performance of the ground anchor may differ depending on the texture of the floor surface, reflection of light, and shadows during the experiment. So we used ground with a large amount of light reflection as in (a), and a drawing paper as in (b) the texture was unified, and the floor environment with weak reflection was constructed. In this paper, performance evaluation was conducted by conducting the same experiment in different (a) and (b) environments.

In order to measure the accuracy of the 3D model loaded at the position of the ground anchor in the WebXR environment, a 3D model with the same shape as the real model was created. Figure 4c shows the 3D model created for the experiment; Figure 4d shows the actual model. Each model is a cube with a side of 15 cm. The checkered pattern on each side was composed of square tiles with a side of 3.75 cm. The errors of the developed 3D model and real model were compared in the WebXR environment. In the case of a flat-patterned model, a difference on the surface of an object does not appear when the difference image is extracted; only differences on the outline are detected. In contrast, for a checkered pattern, the difference on the surface, as well as the outline, can be checked; that is, more detailed difference values can be checked. Therefore, in this study, performance evaluation was conducted using a checkered pattern model to extract the difference image of the object more accurately.

Figure 4e shows the environment of the device used in the experiment. The device used in the experiment is placed at positions ①, ②, and ③, as shown in Figure 4a,b. The experiment was conducted and measurements were obtained from a height of 62 cm from
the ground. Figure 4f shows the tilt of the device arranged as shown in Figure 4e. The inclination was measured with a level meter using a gyroscope sensor provided as standard with the device used in the experiment. The X-axis was parallel to the floor and the Y-axis was inclined 70° from the floor. The experiment was carried out by loading the object at the position ①. The hit test module of the WebXR API is a system that augments the 3D model after specifying the ground anchor; therefore, it was carried out with an inclination toward ① as shown in Figure 4a,b.

![Figure 4](image)

**Figure 4.** Experimental setup for performance evaluation. (a) Strong reflection ground; (b) weak reflection ground; (c) 3D model; (d) real model; (e) device setup; (f) surface level.

### 3.3. Experimental Environment

Table 1 presents a description of the experimental environment. When using the Google Chrome browser, measurement and debugging were performed using the Google Development Tool “inspect” that supports a debugging environment. To test the accuracy of the ground anchor and the loaded model, the hit module of the WebXR Device API was modified and used.
Table 1. Technical basics for experiment.

| Classification          | Information                                                  |
|-------------------------|--------------------------------------------------------------|
| Measurement Tool        | Google Development Tool “inspect”                           |
| Browser                 | Google “Chrome”, version 84.0.4147.89                       |
| 3D File Format          | .glb                                                         |
| Model                   | Samsung galaxy Note 20 Ultra (SM-N986)                      |
| Processor               | Qualcomm snapdragon 865+                                    |
| RAM                     | 12 GB                                                       |
| ROM                     | 256 GB                                                      |
| Display                 | 3088 × 1440 (496 ppi) Dynamic AMOLED 2X                    |
| Sensor                  | Acceleration, Gyro, Light Proximity                         |
| Network                 | 5 G Network                                                 |
| OS                      | Android 10                                                  |

3.4. Experimental Method

Section 3.4 describes the experimental method. The experiment consists of three parts and is explained through each sub-chapter. These describes how to proceed with the experimental methods used in this paper: WebXR camera trajectory, checked pattern for difference image, and frames per second.

3.4.1. WebXR Camera Trajectory

The camera trajectory refers to the change in X, Y, and Z-axis movement during an arbitrary measurement time using the sensor of the device. In this paper, a head mounted display (HMD) that recognizes a head tracker is regarded as ground truth, and after loading a 3D model in WebXR, the device and ground truth were fixed and moved at the same time. The amount of change according to move was measured. The difference in the camera trajectory was extracted as the standard deviation based on the result value based on the ground truth. Equation (1) is explained for obtaining the standard deviation.

\[ v = \frac{1}{n} \sum_{i=1}^{n} (X_i - \bar{X})^2, \sigma = \sqrt{v} \]  

3.4.2. WebXR with Checked Pattern

The method of obtaining the difference image from the captured image is explained in Equation (2). The image in which the actual model is used to obtain the difference image is denoted as \( R \), the image captured by the 3D model is denoted as \( V \), and the image of only the background is denoted as \( B \).

\[ \text{Difference image} = |(B - R) - (B - V)| \]  

Equation (3) is used for obtaining the MAE of the difference image. For each image, to obtain an average of the error between pixels, the absolute error between the measured value and the absolute value is obtained; all the obtained absolute errors are added. The average is calculated by dividing this by the number of absolute errors. \( x \) is the X-axis pixel coordinate of the image, \( y \) is the Y-axis pixel coordinate, \( W \) is the width, and \( H \) is the height.
In addition, the pixel values of the x and y coordinates of the image where the real object is captured are \(R(x, y)\) and pixel values of the x and y coordinates of the image where the 3D model is captured are \(V(x, y)\); only the background is captured. The pixel value of the image’s x and y coordinates is \(B(x, y)\). At this time, \(R(x, y)\) is called “real value”, \(V(x, y)\) is called “measured value”.

\[
\text{Mean of Difference image} = \frac{1}{W \times H} \sum_{x=1}^{W} \sum_{y=1}^{H} |(B(x, y) - R(x, y)) - (B(x, y) - V(x, y))| \quad (3)
\]

Equation (4) is used to determine the Intersection over Union (IoU) of the union image and the intersection image. If the number of intersection pixels and the number of union pixels are the same, they are considered to match. The union image is extracted by removing the background image from the real model image. The difference image of the real model image from the real model image is added to the difference image of the 3D model image from the real model image. The resultant value is calculated from the union image to obtain the intersection image; the match rate is then calculated as follows. The image in which the real model and the 3D model overlap is referred to as \((R, V)\), background image as \((B)\), 3D model image as \((V)\), and actual model image as \((R)\).

\[
\text{Match rate} = \frac{\{(R, V) - (B)\} - \{(R, V) - (R)\} + \{(R, V) - (V)\}}{\{(R, V) - (B)\}} \quad (4)
\]

3.4.3. Frames Per Second (FPS)

Equation (5) was used to find the number of FPS. The duration of start time to end time was measured and the difference was calculated. The absolute value was the result value; it was divided by the number of frames during measurement. \(T_S\) is the start time and \(T_E\) is the end time (unit of \(T\) is ms). \(F\) is the number of frames.

\[
\text{Frame per second} = \frac{F}{|T_S - T_E|} \times 1000 \quad (5)
\]

4. Results and Discussion

4.1. WebXR Camera Trajectory

In order to experiment with the camera trajectory along each axis, a comparative experiment between ground truth and WebXR was performed for each of the X, Y, and Z axes. An HMD that included a head tracker was selected for the ground truth experiment, which showed high accuracy in terms of camera trajectory because the head tracker performed the function of tracking the display device. It has been used in several studies due to its high accuracy within the sensor range [25–27].

After loading the 3D model in the same position as the head tracker that the ground truth could recognize, the device was moved 50 cm in each direction of the X-axis direction (left, right), 50 cm in each direction of the Y-axis direction (up, down), and 50 cm in each direction of the Z-axis direction (front, back). When moving about each axis, the difference between the ground truth and the camera trajectory of WebXR was analyzed.

Figure 5a is the result of an experiment to evaluate the accuracy of the camera trajectory about the X-axis. The experiment was conducted for about 520 frames. The highest point of ground truth was 0.3189, and at this time, the same frame value for the ground with strong reflection was 0.4624, and the same frame value for the ground with weak reflection was 0.3819. The lowest point of ground truth is \(-0.2251\), and at this time, the same frame value of the ground with strong reflection was \(-0.4284\), and the same frame value of the ground with weak reflection was \(-0.2336\).

Figure 5b is the result of an experiment to evaluate the accuracy of the camera trajectory about the Y-axis. The experiment was conducted for about 1220 frames. The highest point of ground truth is 2.2366, and at this time, the same frame value of the ground with strong reflection was 2.1113, and the same frame value of the ground with weak reflection was...
2.1609. The lowest point of ground truth was 1.8774, and at this time, the same frame value of the ground with strong reflection was 1.8199 and the same frame value of the ground with weak reflection was 1.8485.

Figure 5c is the result of an experiment to evaluate the accuracy of the camera trajectory about the Z-axis. The experiment was conducted for about 1230 frames. The high point of the ground truth was 1.4055, and at this time, the same frame value of the ground with strong reflection was 1.5649, and the same frame value of the ground with weak reflection was 1.5525. The lowest point of ground truth was 1.1606, and at this time, the same frame value of the ground with strong reflection was 1.1614, and the same frame value of the ground with weak reflection was 1.1551. As a result of the experiment, when comparing the ground truth and WebXR for each axis, it was confirmed that there was a difference in the camera trajectory based on an anchor that was arbitrarily designated.

![Graphs](image.png)

**Figure 5.** Accuracy evaluation of the camera trajectory for WebXR against the ground truth. (a) X-axis; (b) Y-axis; (c) Z-axis.

Table 2 presents the standard deviation according to the difference between the ground truth (GT: ground truth) and the camera trajectory of WebXR. The results of the experiment showed the standard deviation of the ground with strong reflection (SR: strong reflection) on the X-axis is 0.73 cm, and the standard deviation of the ground with weak reflection (WR: weak reflection) is 0.21 cm. On the Y-axis, the standard deviation of the ground with strong reflection is 0.32 m, and the standard deviation of the ground with weak reflection is 0.38 cm. Additionally, in the Z-axis, the standard deviation of the ground with strong reflection was 0.55 cm, and the standard deviation of the ground with weak reflection was 0.51 cm.
Table 2. Standard deviation of the comparison of ground truth and WebXR.

|       | GT-WR | GT-SR |
|-------|-------|-------|
| X-axis (cm) | 0.21  | 0.73  |
| Y-axis (cm)  | 0.32  | 0.38  |
| Z-axis (cm)  | 0.51  | 0.55  |

4.2. WebXR with Checked Pattern

Figure 6 shows an image captured through the device after loading the actual model and the 3D model in the WebXR environment. The image was captured when moving to the left (\(\alpha\)) and right (\(\beta\)) from the image at (\(\gamma\)). When an object was loaded based on the anchor (\(\delta\)) shown in Figure 4a,b, the difference between the position of the real model and the 3D model was checked. In order to calculate the difference between the anchor and model loading accuracy of the WebXR Device API, the image captured only in the real model was considered as ground truth.

The captured image had a resolution of 1440 × 3088. To calculate the difference image, an evaluation was performed using MATLAB software (Ver.R2020a). Because most calculated images had the same pixel value when the difference image was extracted, the background was identified as black (RGB #000000); the parts where the difference between the ground truth and the WebXR image was a non-zero value in the image.

![Figure 6. eXtended Reality image in each direction.](image)

The result derived using Equation (1) is expressed through the difference image in Figure 7; it shows the difference image measured at positions \(\alpha\), \(\beta\), and \(\gamma\). The results of the experiment show the difference between the ground truth and WebXR images, which shows the accuracy of the WebXR anchor. Figure 7 shows the image with the background removed to obtain the mean absolute error (MAE) of the image captured at positions \(\alpha\), \(\beta\), and \(\gamma\). MAE is the amount of the error in the measured value; it is the difference between the “measured value” and the “real value”. In this study, the actual value was designated as the ground truth and the measured value was designated as WebXR. In addition, MAE represents a qualitative measure of the accuracy of the statistical estimation; the lower the number, the higher will be the accuracy.
In this study, the MAE is a value between 0 and 255. After obtaining images of the real model and 3D model, the background is removed to calculate the mean absolute error of the model. Figure 8 shows images where the background is removed in each direction to obtain the MAE. A binary image is created from the image in which the background is removed. Then, the average absolute error is calculated.

Table 3 shows the mean of the difference image. Experimental results are rounded to the fifth decimal place. In weak reflection ground, through experiments conducted for positions ①, ②, and ③, mean differences of 0.1926, 0.7154, and 0.9611, respectively, were obtained. In addition, when normalized as a percentage, results of 0.0756%, 0.2805%, and 0.3770% were confirmed. In strong reflection ground, through experiments conducted for positions ①, ②, and ③, mean differences of 0.2833, 1.3664, and 1.1958, respectively, were obtained. In addition, when normalized as a percentage, results of 0.1110%, 0.5358%, and 0.4689% were confirmed.
Table 3. Match rate of difference images in WebXR with checkered pattern.

| Direction | Mean of Difference image | Normalization |
|-----------|--------------------------|---------------|
| GT-WR     | 0.1926                   | 0.0756%       |
|           | 0.7154                   | 0.2805%       |
|           | 0.9611                   | 0.3770%       |
| GT-SR     | 0.2833                   | 0.1110%       |
|           | 1.3664                   | 0.5358%       |
|           | 1.1958                   | 0.4689%       |

Figure 9 shows the intersection image of the real model images and the mask image of the union image in each direction to obtain the match rate between regions. In this paper, only a part of the mask image obtained through the experiment of the match rate between regions is expressed as shown in Figure 9. The number of pixels in the non-black (RGB #000000) part of each image is extracted. The match rate between regions is calculated using the extracted number of pixels. The match rate uses the ratio of the number of pixels in the intersection image and the number of pixels in the union image.

Table 4 shows the match rates between the intersection image and the union image. The total number of pixels in each image was 4,446,720 from the experiment. In weak reflection ground, the number of pixels in the intersection image are 57,513, 69,283, and 78,231 for 1°, 2°, and 3°, respectively. The number of pixels in the union image are 62,113, 77,936, and 88,145, and match rates of 92.6%, 88.9%, and 88.8% were obtained. In strong reflection ground, the number of pixels in the intersection image are 56,826, 62,567, and 73,255 for 1°, 2°, and 3°, respectively. The number of pixels in the combined image are 61,346, 90,720, and 90,963, and match rates of 92.6%, 69.0%, and 80.5% were obtained.

Table 4. Match rate in WebXR.

| Direction | Specified Rate Resolution | Intersection Pixels | Union Pixels | Match Rate (%) |
|-----------|---------------------------|---------------------|--------------|---------------|
| WR        | Front (1°)                | 4,446,720           | 57,513       | 62,113        | 92.6          |
|           | Left (2°)                 | 4,446,720           | 69,283       | 77,936        | 88.9          |
|           | Right (3°)                | 4,446,720           | 78,231       | 88,145        | 88.8          |
| SR        | Front (1°)                | 4,446,720           | 56,826       | 61,346        | 92.6          |
|           | Left (2°)                 | 4,446,720           | 62,567       | 90,720        | 69.0          |
|           | Right (3°)                | 4,446,720           | 73,255       | 90,963        | 80.5          |
4.3. 3D Model Rendering FPS Evaluation in WebXR

FPS represents the rate at which frames change in seconds. In general, 15 frames are supported for animation, 24 frames for movies, 30 frames for television, and 60 frames for broadcasting such as sports. The human eye can process a variable frame rate and recognizes more than 8 continuous FPS as natural motion; it can be said that real-time performance is guaranteed when the number of FPS is more than 30. If real-time performance is guaranteed, users can use it naturally while providing services. Therefore, FPS measurement was performed to check whether XR guarantees real-time performance in a web environment. The FPS experiment was conducted after loading a 3D model in a web environment.

Figure 10 shows the result of measuring FPS after the 3D model loading in the WebXR environment while moving in each direction from ① to ② and ③ as shown in Figure 4a,b. The experiment was conducted for 174 s. The X-axis is the time (s), and the Y-axis is the frame count rendered per second. The results confirmed that an average of 29.7 FPS were processed.

![Figure 10. 3D model rendering FPS of WebXR.](image)

4.4. Discussion

The experimental results conducted in this paper were conducted with the ground environment, the 3D model to be loaded, and the device environment for measurement fixed. In order to compare the performance according to the difference in the ground environment, the experiment was divided into the ground with strong reflection and the ground with weak reflection. In addition, performance differences according to environments were analyzed by conducting the same experiment in different ground environments.

In the case of the camera trajectory experiment, a 3D model was loaded in the WebXR environment while the HMD was used as ground truth and the mobile device was fixed, and the fixed device was simultaneously moved by the X, Y, and Z axes. Experiments were conducted for each axis to improve the accuracy of the amount of change when moving in 3D space based on the anchor. In the case of an HMD used as ground truth, the head tracking function can be used to more accurately check the result value of the movement amount when the device is moved. Through this, an experiment was conducted by designating an anchor and loading a 3D model from a mobile device at the same location as the head tracker. Even if the experiment is conducted as in this paper, if the accuracy of the anchor is inferior, when the device is moved after loading the 3D model, a difference in the amount of movement may occur in the mobile device compared to the ground truth. In order to clearly confirm the result of this difference, the standard deviation of the moving amount was calculated in this paper.
In the case of the difference image experiment, the device was moved to the left and right while the horizontal and viewpoint were fixed. The movement to each side was designated as an angular position based on a straight line connecting the position of the front and the anchor vertically. Ground images from each direction, images with only 3D models loaded, images with only real models fixed, and images loaded with both real and 3D models were captured. If the anchor’s accuracy is poor, we can check if the result value is not black (#000000) when calculating the difference image of the captured image. In addition, in this paper, the reliability evaluation method used in the image processing field was used to improve the reliability of the accuracy of the performance evaluation. The average absolute error and IoU were used to express the result mathematically and to obtain the error by pixel and by region, respectively.

Among the methods used, the mean absolute error was used to check the error mean for the pixel values of the difference image. The normalization value of the mean absolute error can be confirmed as the same result as using the mean absolute percentage error (MAPE) method. Additionally, IoU was used to check the match rate for the area of the difference image. In the image processing field, there are various methods for evaluating the reliability of the error rate and the match rate, such as MAE, mean squared error (MSE), MAPE, and IoU. Among them, the mean absolute error can be checked most intuitively because there is no change in the result value of the difference image because the inter-pixel calculation is performed in the calculation process. In addition, IoU is used as a measure of the accuracy between the corresponding regions by dividing the area of the intersection region of the two regions to be compared by the value of the sum region. Therefore, it was used to confirm the agreement rate between regions.

In the case of the FPS experiment, after loading the 3D model in the WebXR environment, the experiment was conducted to measure the rendering speed. In addition, to improve the consistency of the experiment, the speed at which the 3D model is rendered was measured during the difference image experiment.

5. Conclusions

The purpose of this paper was to evaluate the performance of the WebXR Device API, which was released to provide XR services in a web environment. The interface was recently released and the currently released tentative version has poor stability; therefore, a performance evaluation study was conducted for smooth service. For the performance evaluation, the experiment conducted in this paper produced a 3D model and a floor environment. Additionally, we approached extended reality in a web environment through a web server built for the experiment. After that, the 3D model was loaded and proceeded. The first experiment is an experiment on the camera trajectory when the camera moves in the X, Y, and Z axes. The second is the difference image, mean absolute error, and IoU by placing a real model having the same size and shape as the loaded 3D model in the same position. The third experiment consisted of three experiments measuring 3D model rendering speed. In the first and second experiments, we measured and analyzed the comparison with WebXR by specifying the ground truth.

Since the proposed performance evaluation method focuses on expressing a virtual environment in a web environment, we performed an experiment by fixing the measurement equipment and ground environment and limiting the 3D model. However, in the case of real users, various devices, ground environments, and 3D models can be used. In future studies, the performance evaluation methods in various types of experimental environments will be studied. Finally, the experimental results of this study are expected to be the basis for the technical performance evaluation method of the WebXR interface in the future.
Author Contributions: Conceptualization, D.L.; methodology, S.K.; software, M.L., D.L., and W.S.; investigation, D.L. and M.L.; writing—original draft preparation, D.L.; writing—review and editing, S.L. and K.-D.J.; supervision, S.K.; project administration, S.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Acknowledgments: This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korean government (MSIT) (No. 2020-0-00192, Development of Open Anchor-based AR Cloud for Universal Mixed Reality Service).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Rodrigues, A.B.; Dias, D.R.C.; Martins, V.F.; Bressan, P.A.; de Paiva Guimarães, M. WebAR: A web-augmented reality-based authoring tool with experience API support for educational applications. In Proceedings of the International Conference on Universal Access in Human-Computer Interaction, Vancouver, BC, Canada, 9–14 July 2017; pp. 118–128.
2. Neelakantam, S.; Pant, T. Introduction to VR and WebVR. In Learning Web-Based Virtual Reality; Apress: Berkeley, CA, USA, 2017; pp. 1–4.
3. WebVR API. Available online: https://developer.mozilla.org (accessed on 23 August 2021)
4. Jones, B.; Waliczek, N. WebXR device API. W3C Working Draft. This Specification Describes Support for Accessing Virtual Reality (VR) and Augmented Reality (AR) Devices, Including Sensors and Head-Mounted Displays. 2019. Available online: https://www.w3.org/TR/webxr/ (accessed on 1 July 2021).
5. Macintyre, B.; Smith, T.F. Thoughts on the Future of WebXR and the Immersive Web. In Proceedings of the 2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct), Munich, Germany, 16–20 October 2018; IEEE: Munich, Germany, 2018.
6. Shi, S.; Nahrstedt, K.; Campbell, R. A real-time remote rendering system for interactive mobile graphics. ACM Trans. Multimed. Comput. Commun. Appl. 2012, 8, 1–20. [CrossRef]
7. Azuma, R.T. A Survey of Augmented Reality. Presence: Teleoperators and Virtual Environments. Presence Teleoperators Virtual Environ. 1997, 6, 355–385. [CrossRef]
8. Azuma, R.; Baillot, Y.; Behringer, R.; Feiner, S.; Julier, S.; MacIntyre, B. Recent advances in augmented reality. IEEE Comput. Graph. Appl. 2001, 21, 34–47. [CrossRef]
9. Azuma, R. Tracking requirements for augmented reality. Commun. ACM 1993, 36, 50–51. [CrossRef]
10. Milgram, P.; Takemura, H.; Utsumi, A.; Kishino, F. Augmented reality: A class of displays on the reality-virtuality continuum. In Telemanipulator and Telepresence Technologies; Das, H., Ed.; 1995; pp. 282–292. Available online: https://www.spiedigitallibrary.org/conference-proceedings-of-spie/2351/1/Augmented-reality--a-class-of-displays-on-the-reality/10.1117/12.197321.short?SSO=1 (accessed on 1 July 2021).
11. Renius, O. A Technical Evaluation of the WebXR Device API for Developing Augmented Reality Web Applications. Linkoping University, Department of Computer and Information Science, Human-Centered Systems. ISRN: LIU-IDIA/LITH-EX-A–19/032-SE. 2019. Available online: http://www.diva-portal.se/smash/get/diva2:1328199/FULLTEXT01.pdf (accessed on 1 July 2021).
12. Eisert, P.; Fechteler, P. Low delay streaming of computer graphics. In Proceedings of the 2008 15th IEEE International Conference on Image Processing, San Diego, CA, USA, 12–15 October 2008; pp. 2704–2707.
13. Duguet, F.; Drettakis, G. Flexible point-based rendering on mobile devices. IEEE Comput. Graph. Appl. 2004, 24, 57–63. [CrossRef] [PubMed]
14. Baratto, R.A.; Kim, L.N.; Nieh, J. THINC. In Proceedings of the Twentieth ACM Symposium on Operating Systems Principles—SOSP’05, Brighton, UK, 23–26 October 2005; pp. 277–290
15. Lamberti, F.; Sanna, A. A Streaming-Based Solution for Remote Visualization of 3D Graphics on Mobile Devices. IEEE Trans. Vis. Comput. Graph. 2007, 13, 247–260. [CrossRef] [PubMed]
16. Noimark, Y.; Cohen-Or, D. Streaming scenes to MPEG-4 video-enabled devices. IEEE Comput. Graph. Appl. 2003, 23, 58–64. [CrossRef]
17. Lv, Z.; Yin, T.; Han, Y.; Chen, Y.; Chen, G. WebVR—Web Virtual Reality Engine Based on P2P network. J. Netw. 2011, 6, 990-998. [CrossRef]
18. Gunkel, S.; Prins, M.; Stokking, H.; Niamut, O. WebVR meets WebRTC: Towards 360-degree social VR experiences. In Proceedings of the 2017 IEEE Virtual Reality (VR), Los Angeles, CA, USA, 18–22 March 2017.
19. Barsoum, E.; Kuester, F. WebVR: An interactive web browser for virtual environments. In Stereoscopic Displays and Virtual Reality Systems XII; Woods, A.J., Bolas, M.T., Merritt, J.O., McDowall, I.E., Eds.; SPIE: Bellingham, WA, USA, 2005.
20. OpenXR. Royalty-Free, Open Standard that Provides High-Performance Access to Augmented Reality (AR) and Virtual Reality (VR)—Collectively Known as XR—Platforms and Devices. Available online: https://www.khronos.org/registry/OpenXR/ (accessed on 29 July 2019).

21. Parisi, T. WebGL: Up and Running; O’Reilly Media, Inc.: Sebastopol, CA, USA, 2012.

22. Rego, N.; Koes, D. 3Dmol.js: Molecular visualization with WebGL. Bioinformatics 2014, 31, 1322–1324. [CrossRef] [PubMed]

23. Angel, E.; Shreiner, D. Interactive Computer Graphics with WebGL; Addison-Wesley Professional: Boston, MA, USA, 2014.

24. Woo, M.; Neider, J.; Davis, T.; Shreiner, D. OpenGL Programming Guide: The Official Guide to Learning OpenGL; Version 1.2. Pearson Education; Addison-Wesley Longman Publishing Co., Inc.: Boston, MA, USA, 1999.

25. Lee, J.; Hafeez, J.; Kim, K.; Lee, S.; Kwon, S. A Novel Real-Time Match-Moving Method with HoloLens. Appl. Sci. 2019, 9, 2889. [CrossRef]

26. Hwang, L.; Ha, S.; Gentet, P.; Lee, J.; Kwon, S.; Lee, S. Verification of an Accommodative Response for Depth Measurement of Floating Hologram Using a Holographic Optical Element. Appl. Sci. 2020, 10, 6788. [CrossRef]

27. Ha, S.; Hwang, L.; Lee, J.; Gentet, P.; Kwon, S.; Lee, S. A Study on Augmented 3D Display Optimization Based on Holographic Optical Element for High Depth and Magnified Image. Front. Phys. 2021, 9, 218. [CrossRef]