International Journal of Geo-Information

MDPI

Article

Construction and Optimization of Three-Dimensional Disaster Scenes within Mobile Virtual Reality

Ya Hu [1], Jun Zhu [1,*], Weilian Li [1], Yunhao Zhang [1], Qing Zhu [1], Hua Qi [1], Huixin Zhang [2], Zhenyu Cao [3], Weijun Yang [4] and Pengcheng Zhang [4]

1 Faculty of Geosciences and Environmental Engineering, Southwest Jiaotong University, Chengdu 610031, China; huya@swjtu.edu.cn (Y.H.); vgewilliam@163.com (W.L.); zhangyh0506@163.com (Y.Z.); zhuq66@263.net (Q.Z.); qi-3dgis@126.com (H.Q.)

2 College of Physics and Engineering, Chengdu Normal University, Chengdu 611130, China; wzfei_001@163.com

3 Sichuan Geomatics Center, Chengdu 610041, China; scgisczy@163.com

4 Guangzhou Urban Planning & Design Survey Research Institute, Guangzhou 510060, China; fazeyang@gmail.com (W.Y.); guangzhou2000@126.com (P.Z.)

* Correspondence: zhujun@swjtu.edu.cn; Tel.: +86-130-844-26186

Received: 14 April 2018; Accepted: 12 June 2018; Published: 14 June 2018

Abstract: Because mobile virtual reality (VR) is both mobile and immersive, three-dimensional (3D) visualizations of disaster scenes based in mobile VR enable users to perceive and recognize disaster environments faster and better than is possible with other methods. To achieve immersion and prevent users from feeling dizzy, such visualizations require a high scene-rendering frame rate. However, the existing related visualization work cannot provide a sufficient solution for this purpose. This study focuses on the construction and optimization of a 3D disaster scene in order to satisfy the high frame-rate requirements for the rendering of 3D disaster scenes in mobile VR. First, the design of a plugin-free browser/server (B/S) architecture for 3D disaster scene construction and visualization based in mobile VR is presented. Second, certain key technologies for scene optimization are discussed, including diverse modes of scene data representation, representation optimization of mobile scenes, and adaptive scheduling of mobile scenes. By means of these technologies, smartphones with various performance levels can achieve higher scene-rendering frame rates and improved visual quality. Finally, using a flood disaster as an example, a plugin-free prototype system was developed, and experiments were conducted. The experimental results demonstrate that a 3D disaster scene constructed via the methods addressed in this study has a sufficiently high scene-rendering frame rate to satisfy the requirements for rendering a 3D disaster scene in mobile VR.

Keywords: 3D disaster scenes; mobile VR; diverse scene data representation; scene optimization; adaptive scheduling

1. Introduction

Disasters often cause a large number of casualties and considerable property losses [1,2]. A three-dimensional (3D) visualization of a disaster scene can be used to represent the current, past, and future conditions of the disaster site. Such a visualization provides an intuitive decision analysis platform for disaster-related personnel, which is of great significance for mitigating the potential adverse effects caused by disasters. Much work has been conducted in this field, such as personal computer (PC)-based 3D visualizations of flood disaster scenes [3–11], earthquake disaster scenes [12–16], and geological disaster scenes [17–19]; immersive experiences of disasters, based on PC virtual reality (VR) systems [20–25]; and 3D disaster simulation and visualization analyses, based on diverse computing systems (such as desktops, laptops, tablets, and smartphones) [26].

Due to the urgency and scientificity of disaster emergency responses [27–29], all types of emergency personnel in different locations need to perceive and recognize disaster environments faster and better, which will require more mobile and immersive visualizations of 3D disaster scenes. However, with regard to these requirements, methods addressed in the existing research have the following deficiencies: (1) these methods lack the capacity for mobility, since 3D disaster visualizations based on PC and PC VR systems are generated indoors and cannot satisfy the requirements for outdoor disaster visualization; (2) the methods offer only weak immersion—for instance, although disaster visualization based on diverse computing systems can solve the mobility problem, the display devices of these mobile terminals (such as smartphones) are physically small. Therefore, the devices' immersion is inadequate, which hinders the sufficient perception and recognition of the disaster environment. In other words, in the existing methods of 3D disaster scene visualization, mobility, and immersion cannot coexist.

With the development of mobile head-mounted displays (HMDs), smartphones, and the mobile internet, mobile VR has gradually entered people's lives [30,31]. Mobile VR involves placing a smartphone inside a mobile HMD, which allows a user to become immersed in a virtual environment by wearing the mobile HMD. The emergence of mobile VR allows anyone to experience immersive VR content anywhere [32]. Thus, mobile VR combines mobility with immersion. The combination of mobile VR with 3D disaster visualization—that is, the 3D visualization of disaster scenes in mobile VR—can sufficiently solve the problems associated with existing methods of 3D disaster scene visualization, allowing users to perceive and recognize disaster environments faster and better.

A 3D visualization of a scene in mobile VR requires a high rendering frame rate, usually more than 30 fps (ideally, approximately 60 fps), to ensure that the scene is immersive [33] and to prevent users from feeling dizzy [32,34]. In the 3D visualization of scenes with a small amount of data, the scene-rendering frame rate can easily satisfy the requirements for mobile VR, as described in [34]. However, the models of 3D disaster scenes usually include large data terrain models, disaster simulation models, etc. Therefore, the question of how to draw them efficiently in order to satisfy the high frame-rate requirements for the rendering of 3D disaster scenes poses a technical challenge.

A 3D visualization of a disaster scene in mobile VR is essentially a high-frame-rate rendering of a disaster scene on a smartphone. Constrained by the low performance of previous mobile phones, few studies have been carried out on the 3D visualization of scenes similar to disaster scenes on mobile phones. Noguera et al., based on client/server (C/S) hybrid rendering technology, implemented a method for the 3D visualization of terrain models on mobile phones, such as the Nokia N95 and iPhone 3 GS [35–37]. Magliocchetti et al. introduced a method for the 3D visualization of terrain models on smart phones, such as the HTC Touch Magic and HTC Nexus One. This method follows certain data access standards in the geospatial domain, such as the Web Map Service (WMS), and has a relatively high scene-rendering frame rate [38]. Trujillo and Suárez implemented a method for the 3D visualization of terrain models, based on a virtual globe, on mobile devices such as the iPad 2, the Samsung Galaxy Tab, and the iPhone 4 [39,40]. Santana et al. also implemented the 3D stereoscopic rendering of terrain models and building block models on the iPhone 6 smartphone on the basis of a virtual globe [41].

The work reviewed above mainly achieved the 3D visualization of terrain models based on mobile phones. Because of the low mobile phone performance and slow network speeds at the time, some of the reported work needed to rely on specific server- and client-side programs to perform collaborative visualization, such as hybrid rendering based on C/S [35–37], or the real-time conversion of large-sized tiles on servers into small-sized tiles for transmission over the network [38]. These methods are not optimal for the performance and display environments of current smartphones, and they are also very tedious to implement. Some other researchers [39–41] claim to have implemented the visualization of terrain models based on the open-source technology Glob3 Mobile [42], but the related website did not provide any source code or development documents. As a result, the 3D stereoscopic rendering function reported by Santana et al. [41], the most important function for scene visualization in mobile

VR [43], cannot be independently reproduced on the basis of Glob3 Mobile; in addition, other functions necessary for the 3D visualization of disaster scenes in mobile VR also cannot be implemented on the basis of Glob3 Mobile.

Another type of work that is similar to the 3D rendering of disaster scenes in mobile VR is the PC-based 3D visualization of disaster scenes [6,8,17,19]. However, unlike mobile-VR-based visualization, which requires a high scene-rendering frame rate, the PC-based 3D visualization of disaster scenes requires only a frame rate of at least 25 fps. Current smartphones are far inferior to PC systems in terms of performance and screen size. Consequently, a 3D disaster scene visualization method that is suitable for high-performance PC systems (for example, a method for the representation of scene data on such systems) will not be suitable for constructing higher-frame-rate 3D disaster scenes on lower-performance smartphones.

In general, the existing related visualization work is either for high-performance PCs or for previous low-performance mobile phones, or alternately the 3D graphics development package used by these work is not available, and these work therefore cannot provide a sufficient solution for the high frame rate visualization of 3D disaster scenes in mobile VR.

At present, there are many methods for improving the scene-rendering frame rate, such as optimizing scene data organization, optimizing scene rendering, improving hardware performance, and reducing the amount of scene data [34,35]. The scene data organization refers to the organization of scene models into spatial hierarchies, such as quadtrees, and its development is also mature [34]. The scene rendering optimization, such as draw call batching, is a concern for 3D rendering engines, such as Unity3D. Better hardware can significantly improve scene rendering performance, but the use of high-performance mobile phones is often limited in disaster emergency situations. However, as mentioned above, the models of 3D disaster scenes usually include large-data terrain models, disaster simulation models, etc., so it is necessary to reduce the amount of scene data in order to improve the scene-rendering frame rate, so as to satisfy the high frame-rate requirements for the stereoscopic rendering of 3D disaster scenes based in mobile VR. However, the existing methods for reducing the amount of scene data are either for high-performance PCs [6,8,17,19] or for earlier low-performance mobile phones [35–40], and are not suitable for the performance and display environments of current smartphones. Therefore, it is necessary to explore new scene data reduction methods, adapted for the performance and display environments of current smartphones, in order to achieve high scene-rendering frame rates. To reduce the amount of scene data, based on existing smartphones in common use and a popular open-source development framework, this paper discusses the construction and optimization of 3D disaster scenes. This paper is organized as follows. Section 2 describes a method for 3D disaster scene construction and optimization in mobile VR. Using a flood disaster as an example, Section 3 describes a plugin-free prototype system, and analyzes experiments performed with this prototype. Section 4 presents the conclusions of this study, and discusses its contributions and our plans for future work.

2. Methodology

2.1. General Design

The construction of a 3D disaster scene involves a large amount of information and data, far beyond the storage and processing capabilities of mobile phones. Therefore, the data involved in the construction of a 3D disaster scene must be stored and processed using indoor high-performance computers, and then the visualization of and interaction with a 3D disaster scene on the mobile phones can be implemented through the network. In other words, the mobile VR-based 3D disaster scene visualization system adopts either the C/S or the browser/server (B/S) architecture, instead of the architecture where the program and data are stored on the mobile phones. To use the C/S architecture, users need to download a dedicated APP (application) program. To save users the trouble of downloading and installing APP programs, it is desirable for users to be able to use a browser for

the 3D visualization of and interaction with disaster scenes. To this end, in this study, a plugin-free browser/server (B/S) framework was designed to satisfy the requirements for the construction and visualization of a 3D disaster scene in mobile VR. This framework is illustrated in Figure 1.

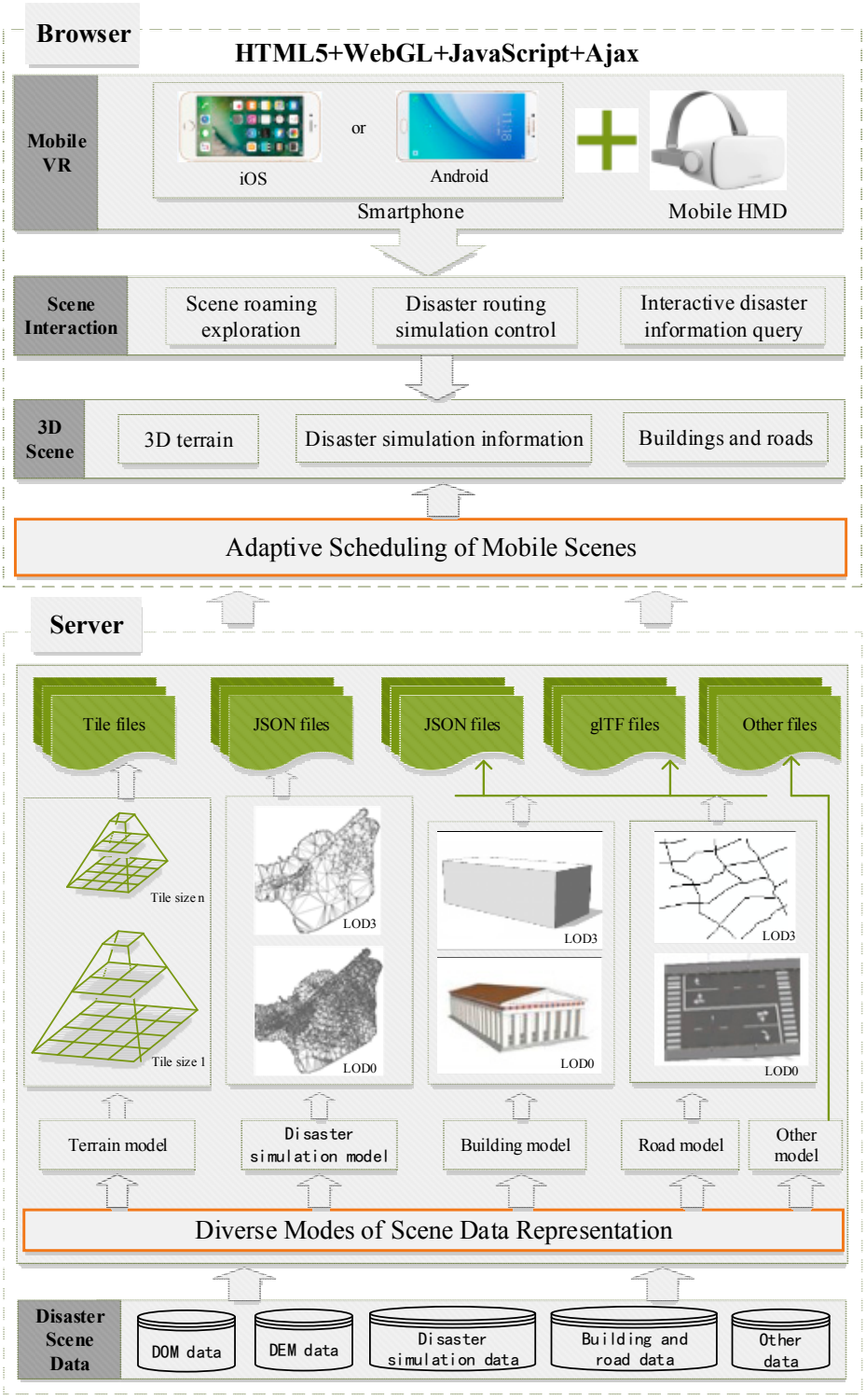

Figure 1. Plugin-free browser/server (B/S) framework for the construction and visualization of three-dimensional (3D) disaster scenes based on mobile virtual reality (VR).

On the server side, data related to the construction of a 3D disaster scene are represented diversely and published through the network. Models of 3D disaster scenes typically include terrain models, disaster simulation models, house models, and road models [6,11,12,14,16]. For terrain models, the original image data and the digital elevation model (DEM) data are sliced to form multiple tile-size pyramid models, thereby making it possible for a browser to access a specific tile-size image pyramid model or DEM pyramid model, depending on the performance and operating environment of the user's smartphone. For disaster simulation models, the simulated results of disaster models are optimized to generate level-of-detail (LOD) models, in order to facilitate network transmission and visualization. For house models, it is necessary to generate diverse representations of the houses based on the available modeling data or models. For example, if only data that reflect the house outline and height information are available, then a block model representation of a house may be generated, and the contour and height information will need to be sent to the browser side as a JSON (JavaScript object notation) file for the 3D visualization of the house; if 3D house models are available, these models can be sent to the browser side as glTF (GL Transmission Format) files for the 3D visualization of the houses. For road models, the processing is similar to that of house models, and thus is not detailed here.

On the browser side, depending on the necessary frame rate or the desired visualization effect, the appropriate models are loaded to visualize the 3D disaster scene. Then, based on this 3D scene, through gaze-based interaction and by means of the smartphone and the mobile HMD, the immersive and effective exploration of the 3D disaster scene can be implemented.

The browser side is implemented using HTML5, WebGL, JavaScript, and Ajax. HTML5 is the newest version of the HTML standard. It not only has new syntactical features, but also delivers rich multimedia content without the need for additional support or plug-ins [44]. It is supported by many popular browsers, such as Chrome, Firefox, and Internet Explorer. The Canvas element in HTML5 provides the ability to render WebGL content. WebGL is a JavaScript API (Application Programming Interface) for plugin-free 3D rendering within any compatible web browser. Several open-source Web Virtual Globe applications that are based on WebGL technology support open web mapping service standards, such as WMS, TMS (Tile Map Service), and WMTS (Web Map Tile Service), and offer common functions such as 3D terrain visualization and interactive roaming in 3D scenes [45]. These applications include Cesium [46], WebGLEarth [40], and OpenWebGlobe [47]. JavaScript is an object-oriented language and can interact with HTML source code, thus enabling sites with dynamic content. Ajax is used for efficient data communication between the server and clients. These modern technologies enable our browser-based web application for 3D disaster scene visualization, which works across platforms, requires no plugins, and is easy to maintain.

2.2. Diverse Modes of Scene Data Representation

Models of 3D disaster scenes typically include terrain models, disaster simulation models, house models, and road models [6,11,12,14,16]. Tasks similar to the 3D rendering of disaster scenes in mobile VR are the PC-based 3D visualization of disaster scenes [6,8,17,19] and the 3D visualization of terrain on mobile phones [35–40]. In PC-based 3D disaster scene visualization, because of the high performance and relatively uniform screen sizes and screen resolutions of PCs, a single scene data representation is typically adopted. For example, a terrain model may be represented in the form of a pyramid model with a single tile size, while for a disaster simulation model, a single visual representation mode may be adopted based on the original simulated results. In 3D terrain visualization on mobile phones, because of the lower performance, smaller screen sizes, and lower screen resolutions of the existing mobile phones at the time of the cited studies, the 3D terrain scenes are usually represented with smaller-sized terrain tiles. For example, the tile sizes of a DEM pyramid model may be 8 × 8 pixels [35,37,38] or 12 ×12 pixels [39,40], and the tile size of an image pyramid may be 64 × 64 pixels [35,37]. That is to say, the existing scene representations used in similar previous works are either single representations for use on high-performance PCs or small-tile-size

representations for use on low-performance mobile phones. However, the performance, screen sizes, and screen resolutions of current smartphones are neither the same as those of PCs, nor the same as those of previous low-performance phones; consequently, the existing scene data representations are not suitable for the performance and display environments of current smartphones. Therefore, it is necessary to explore diverse scene data representation methods adapted for the performance and diverse display environments of current smartphones, in order to achieve the effective 3D stereoscopic rendering of disaster scenes in mobile VR.

A terrain model is an important part of a 3D disaster scene. To allow users to efficiently roam modeled terrain, a tile-based LOD representation is typically used for a terrain scene [39]. The chosen tile size is affected primarily by the hardware performance, screen size, and screen resolution [40]. A larger tile size results in a greater number of triangles and a larger amount of textural data to be processed for scene rendering, which increases the requirements for hardware performance and results in a more detailed visual scene. As described in the previous paragraph, the existing terrain tile pyramid models are either single-tile-size pyramid models for use on PCs (the tile size of a DEM pyramid model is typically 64 × 64 pixels, and the tile size of an image pyramid is typically 256 × 256 pixels) or small-tile-size pyramid models for low-performance phones. Since the performance and screen sizes of current smartphones are between those of PCs and previous low-performance phones, the existing terrain tile pyramid models are not suitable for the performance and display environments of current smartphones. Therefore, this paper proposes a terrain pyramid model representation scheme based on various tile sizes (e.g., image pyramid models with tile sizes of 32 × 32 pixels, 64 × 64 pixels, and 128 × 128 pixels) to achieve a variety of terrain model representations, thereby making it possible for different users to achieve high scene-rendering frame rates and an improved visualization effect.

As mentioned above, the existing representations of disaster simulation models are applicable on PCs and consist of single visual representations based on the original simulated results. However, the performance and screen sizes of current smartphones are not comparable to those of PCs. Therefore, for use on smartphones, disaster simulation models should not be represented by single models, but should instead be represented by LOD models, in order to allow smartphones to achieve higher scene-rendering frame rates and an enhanced scene-visualization effect. Similarly, house models and road models also should not be represented by single models, but should instead be represented by LOD models. The diverse scene data representations used in the proposed framework are shown in Figure 1.

2.3. Optimization of Mobile Scene Representation

Based on the diverse modes of scene data representation proposed above, the optimization of mobile scene representation can be realized. Because of the relatively poor performance of smartphones, it is necessary to reduce the amount of data used to model 3D disaster scenes to maintain a high scene-rendering frame rate. Due to the small screen sizes of smartphones, an appropriate reduction of the scene data can also achieve a satisfactory visualization effect. A common method of scene data reduction is to load a simplified LOD model into a 3D scene. In the case of discrete LOD models, it is relatively simple to reduce the scene data by directly loading a simplified LOD model. The case of a continuous terrain model is slightly more complex; here, scene data reduction can be realized by reducing the maximum tile level that can be loaded into the 3D scene and loading only tiles of smaller sizes. These two approaches to scene optimization are detailed as follows.

1. Reducing the maximum tile level that can be loaded for a specific tile-size image or DEM pyramid model: if the maximum tile level that can currently be loaded is L_{max} (e.g., 17), then the maximum tile level can be reset to $L = L_{max} - i$ ($i = 1, 2, \ldots, i \leq L_{max} - L_{min}$), where L_{min} denotes the lowest level of the tile pyramid model.

The relationship between tile level i and the number of tiles per row or per column is shown in Equation (1). To aid in description, tile partitioning is performed using the Mercator-based schema with only one tile at level 0 (see [48]).

$$R_i = 2^i, C_i = 2^i \quad (1)$$

In the above formula, R_i denotes the number of tiles in a row at tile level i, and C_i denotes the number of tiles in a column at tile level i; thus, the total number of tiles at tile level i is 2^{2i}. When the viewpoint is close to the scene in a certain area, we must load 256 tiles at level 4. If we reduce the maximum tile level from 4 to 2, we then need to load only 16 tiles at level 2 for the same area. This example shows that this method is greatly effective in reducing the number of tiles in a scene, thus considerably reducing the number of triangles or textures to be rendered.

2. Loading tiles of smaller sizes: if the tile size of the currently loaded image or DEM pyramid model is N and the maximum tile level that can be loaded is M, we can consider loading tiles with a tile size of S, where $S = N/2^i$ ($i = 1, 2, 3, \ldots$), while simultaneously limiting the maximum tile level that can be loaded to M.

When loading smaller-sized tiles, the number of tiles loaded in the scene does not change as long as the maximum tile level that can be loaded and the user's viewpoint and orientation remain unchanged.

If the tiles are image tiles, then the number of texture pixels used for texture mapping can be greatly reduced by reducing the tile size. The total number of texture pixels in the scene is related to the number of tiles in the scene and the tile size, as shown in Formula (2):

$$N_{pixel} = K_{tile} \times TileWidth^2 \quad (2)$$

In the above formula, N_{pixel} denotes the total number of texture pixels in the scene, K_{tile} denotes the number of tiles in the scene, and $TileWidth$ denotes the tile size. When loading smaller-sized tiles, as long as the number of tiles in the scene remains unchanged, the total number of texture pixels in the scene decreases in proportion to the square of the tile size.

If the tiles are DEM tiles, then the number of triangles to be rendered in the scene can be greatly reduced by reducing the tile size. The number of triangles to be drawn per tile depends on the tile size, as shown in Equation (3) (the method of [40] is used to eliminate gaps between tiles):

$$N_{delta} = 2 \times TileWidth^2 + 8 \times TileWidth \quad (3)$$

In this formula, N_{delta} denotes the number of triangles to be drawn per tile, and $TileWidth$ denotes the tile size (i.e., each side of the tile contains $TileWidth + 1$ elevation points). For example, when the DEM tile size is changed from $TileWidth$ to $TileWidth/2$, the total number of triangles is reduced to $1/4 \times N_{delta} + 2 \times TileWidth$, which is approximately $1/4$ of the original number. Therefore, reducing the DEM tile size can greatly reduce the number of triangles to be drawn per tile, thus considerably reducing the number of triangles to be rendered in the scene.

2.4. Adaptive Scheduling of Mobile Scenes

When a scene is rendered on a smartphone, adaptive scene scheduling can be achieved by monitoring the scene-rendering frame rate. When the scene-rendering frame rate is high, a more detailed LOD model can be loaded to improve the scene-visualization effect. When the scene-rendering frame rate is low, a more simplified LOD model can be automatically loaded to improve the scene-rendering efficiency. In this way, a balance is achieved between the scene-visualization efficiency and the visualization effect. Strategies for adaptive scene scheduling are listed in Table 1. In Table 1, the selection of the representation form of the road models is to illustrate that when a user selects a certain level models of LOD models to represent the roads, the user can further schedule the scene by adjusting the number of models to be loaded.

Table 1. Strategies for adaptive scene scheduling.

Scene Model Type	Representation Form	Strategies for Adaptive Scene Scheduling
terrain models	image/digital elevation model (DEM) pyramid models with multiple tile sizes	1. raising/lowering the maximum tile level that can be loaded in the scene 2. loading tiles of larger/smaller sizes
disaster simulation models	a group of TIN (Triangulated Irregular Network) models with a level-of-detail (LOD) structure	loading a more detailed/simplified LOD model
house models	LOD models	loading a more detailed/simplified LOD model
road models	road centerlines	loading all/main road centerlines

3. System Implementation and Experimental Analysis

Floods are the most common disasters. Therefore, this paper uses a flood disaster as an example to illustrate the construction and optimization of 3D disaster scenes in mobile VR.

3.1. Implementation of the Prototype System

Node.js v6.11.2 was used to build the network server in the prototype system. This server stores terrain data of multiple tile sizes, LOD models of flood simulation data, and house data. The browser side was implemented using HTML5, JavaScript, and Cesium. Cesium, a leading open-source library for 3D virtual globes, was adopted because it provides adequate support for stereoscopic rendering and for cross-platform and cross-device dynamic visualization without plugins [49]. A dam-failure flood area [8] was chosen for the experiments. The prototype system can run on browsers that support HTML5 and WebGL, such as Chrome and Firefox. Figure 2 shows the visual effect achieved when running the prototype system on a smartphone using Chrome.

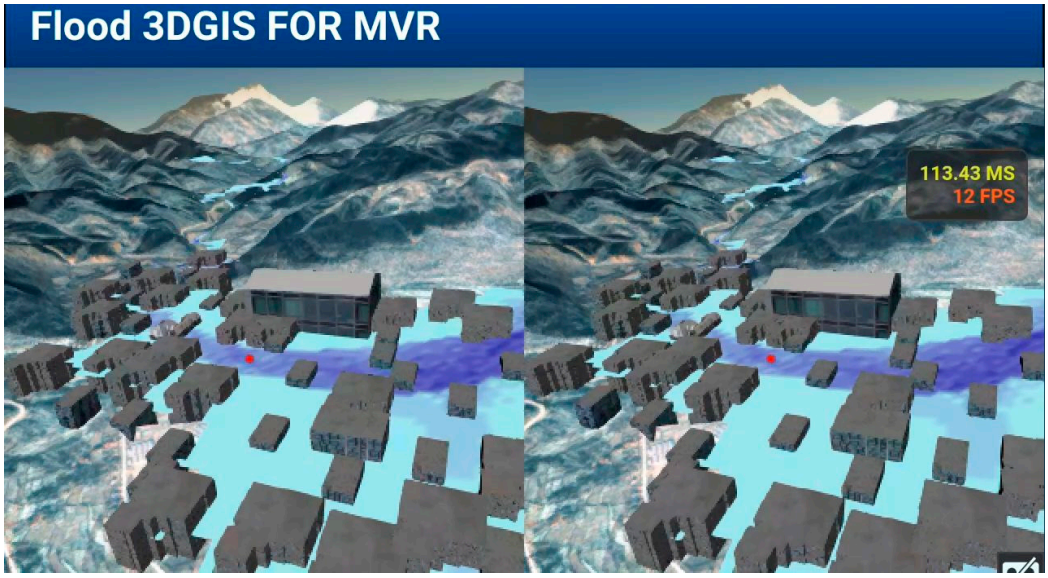

Figure 2. Screenshot of the prototype system on a smartphone.

When run on a common smartphone, the prototype system can satisfy the high rendering-frame-rate requirements for rendering a 3D flood scene in mobile VR. The prototype system also provides multiple gaze-based scene interaction methods, including scene roaming exploration, flood routing simulation control, and interactive flood information querying. Through these methods

of interaction, users can effectively explore and control 3D flood scenes based in mobile VR, thus making the system user-friendly.

3.2. Experimental Data and Environment

3.2.1. Experimental Data

The resolution of the original image data from the experimental area is 1 m, and the resolution of the original DEM data from the experimental area is 10 m. For these raw data, a geographic-based schema (i.e., there are two tiles at level 0, as described in [48]) was used for tile partitioning, in order to produce hierarchical and block data. The specifications of the tile data used in the experiments are shown in Table 2. Since the Cesium open-source framework currently supports only one DEM tile size (i.e., 64), a DEM pyramid model with only a tile size of 64 was provided in this study.

Table 2. Specifications of the tile data used in the experiment.

Tile Type	Tile Size (Pixel)	Minimum Tile Level	Maximum Tile Level
image tiles	64	10	19
	128	10	18
	256	10	17
	512	10	16
	1024	10	15
DEM tiles	64	10	15

Four different LOD flood simulation datasets, denoted by LOD0, LOD1, LOD2, and LOD3, were used in the experiments. LOD0 contains the most detailed data, whereas LOD3 contains the most simplified data. For details, see [26]. The house data included the most simplified house block models as well as the 3D house models expressed in the glTF format, and the size of each glTF file was about 3 M.

3.2.2. Experimental Environment

The mobile HMD selected for the experiments was the Baofeng Mojing CC (Beijing, China), and the experiments were performed in a wireless Wi-Fi network environment with a bandwidth of 10 Mbps. The prototype system was run using the Chrome browser on Android and iOS smartphones. The specifications of the two mobile phones used for testing are shown in Table 3.

Table 3. Specifications of the two smartphones used for testing.

Smartphone	CPU	GPU	Memory Size	Operating System
MI 5s	Snapdragon MSM8996SG-AB	Adreno 530	3 G	Android 8.5.4
iPhone 7	Apple A10 Fusion	PowerVR GT7600	2 G	iOS 10

3.3. Scene Construction and Optimization Experiments

3.3.1. Experimental Method

To verify the scene construction and optimization methods, the prototype system was run on the two smartphones specified above and used to roam through the modeled scene along a certain path to test the influence of the proposed methods on the scene-rendering frame rate. The scene roaming direction was the same as the direction of the flood routing. The elevation of the roaming flight path was 800 m, and the height of the path from the ground was between 50 m and 200 m, to ensure that the tile level of the scene tiles loaded during flight would reach the maximum tile level allowed to be loaded. The flight time was 120 s.

3.3.2. Experimental Results

In the experiments, the influence of the loading of different flood routing models and house models on the scene-rendering frame rate was tested. In addition, the effect of reducing the maximum tile level allowed to be loaded, and the influence of loading different-sized tiles on the scene-rendering frame rate were tested. The results of the experiments are shown in Figures 3–5. The *x* axis (Time (s)) in Figures 3–5 indicates the time of flight along the roaming path.

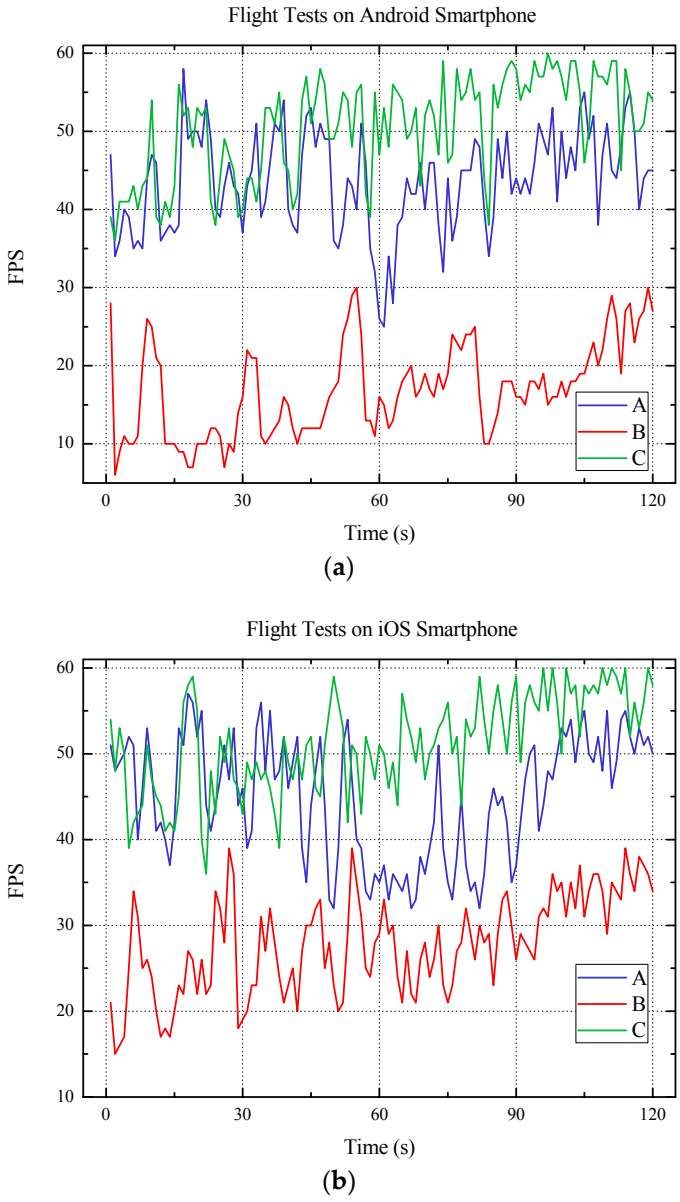

Figure 3. Tests of the influence of the loading different house models and flood routing models on the rendering frame rate: (**a**) flight tests on the Android smartphone; (**b**) flight tests on the iOS smartphone.

In Figure 3, curve B represents a case in which the 3D house models, the most detailed flood routing model (LOD0), and the terrain model were loaded in the scene. Curve A represents a case in which the most simplified house block models, the most detailed flood routing model (LOD0) and the terrain model were loaded in the 3D flood scene. Curve C represents a case in which the most simplified house block models, the most simplified flood routing model (LOD3), and the terrain model were loaded in the scene.

On both the Android and Apple smartphones, the frame rate represented by curve A is higher than that of curve B, which indicates that loading more simplified house models can improve the scene-rendering frame rate. The frame rate represented by curve C is higher than that of curve A, particularly in the second half of the roaming test, which indicates that loading a more simplified flood routing model can improve the scene-rendering frame rate. The reason for the more prominent increase in the frame rate in the second half of the test may be that the second half of the roaming path passed through a larger flood-inundated area.

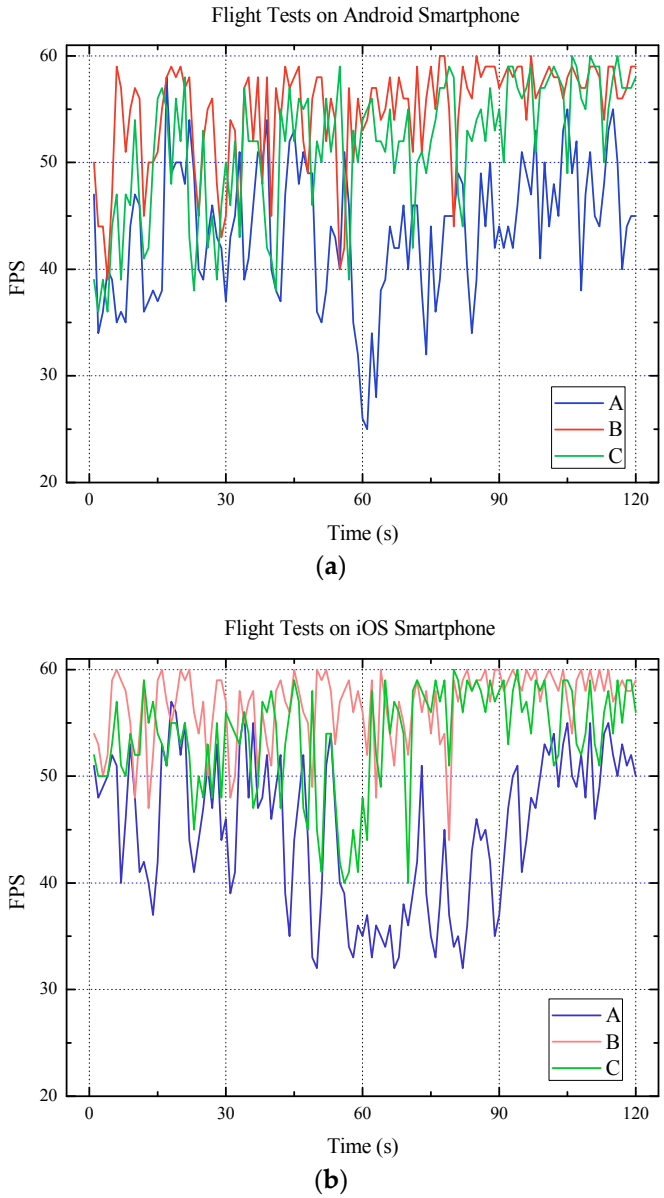

Figure 4. Tests of the influence of reducing the maximum tile level that can be loaded on the rendering frame rate: (**a**) flight tests on the Android smartphone; (**b**) flight tests on the iOS smartphone.

In Figure 4, curve A represents a case in which the maximum loadable image tile level (with a tile size of 256 × 256 pixels) in the scene was 17, and the maximum loadable DEM tile level was 15. Curve B represents a case in which the maximum loadable image tile level (with a tile size of 256 × 256 pixels) was 15, and the maximum loadable DEM tile level was 15. Finally, curve C represents the case in which the maximum loadable image tile level (with a tile size of 256 × 256 pixels) was 17, and the maximum loadable DEM tile level was 13.

For both the Android and Apple smartphones, the frame rates represented by curves B and C are both higher than that of curve A, indicating that reducing the maximum tile level that can be loaded is useful for enhancing the scene-rendering frame rate. The frame rate represented by curve B is slightly higher than that of curve C, which shows that the effect of reducing the maximum loadable image tile level is greater than the effect of reducing the maximum loadable DEM tile level.

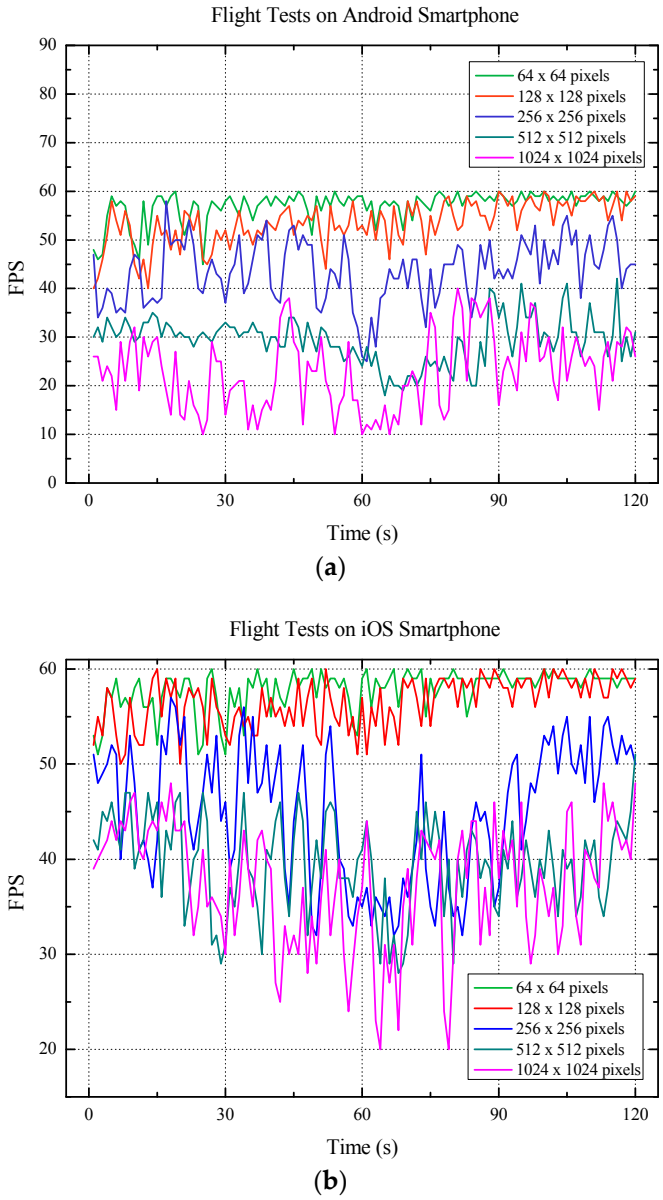

Figure 5. Tests of the influence of the loading image tiles of various tile sizes on the rendering frame rate: (**a**) flight tests on the Android smartphone; (**b**) flight tests on the iOS smartphone.

In Figure 5, the five curves represent the cases in which the sizes of the image tiles loaded in the scene were 64 × 64 pixels, 128 × 128 pixels, 256 × 256 pixels, 512 × 512 pixels, and 1024 × 1024 pixels. On both the Android and Apple smartphones, the scene-rendering frame rate corresponding to the conventional image tile size of 256 was generally above 30 fps, meaning that it satisfied the basic frame-rate requirement for rendering a 3D flood scene in mobile VR; however, the frame rate fluctuated sharply. Loading tiles at a tile size of 128 or 64 dramatically increased the scene-rendering frame rate and resulted in lower frame rate fluctuations, demonstrating that the frame rate can be significantly improved by loading smaller tiles.

Conversely, when larger tiles with a size of 512 or 1024 were loaded, the frame rate was low. This was particularly true for the Android devices, for which most of the frame rates were below 30 fps.

The experiments described above show that loading simplified house models and flood routing simulation models, reducing the maximum tile level that can be loaded in the scene, and loading smaller-sized tiles can all effectively improve the scene-rendering frame rate to satisfy the high frame-rate requirements for the rendering of a 3D flood scene in mobile VR. The scene-rendering frame rate is affected by many factors, such as mobile phone performance, operating systems, browsers, network conditions, 3D rendering programs, scene data organization methods, roaming paths, and scene data volume. However, for the scene-rendering frame rate curves in the experiment result graphs, such as the five scene-rendering frame rate curves in Figure 5a, when rendering them, the corresponding mobile phone, operating system, browser, network condition, 3D rendering program, scene data organization method, and roaming path are the same, and the only difference is the amount of scene data. It can be seen that it is precisely because the amount of data in the scene is different that it leads to different frame rate curves; in addition, it is precisely because of the adoption of the diverse representations of scene data, the optimization of mobile scene representation, and the scheduling of mobile scenes proposed in this paper that the scene has different scene data amounts, which provide ways for the high-frame-rate visualization of 3D disaster scenes in mobile VR. These findings prove that the method proposed here is feasible for the construction and optimization of a 3D disaster scene based in mobile VR.

4. Conclusions and Future Work

We described the construction and optimization of 3D disaster scenes based in mobile VR. A plugin-free B/S framework was designed that makes full use of the capabilities of indoor high-performance computers to process and publish the information necessary for the construction of 3D disaster scenes, and uses a smartphone and a mobile HMD to enable the immersive exploration of and interaction with a scene. Certain key technologies for scene optimization were discussed, including diverse modes of scene data representation, representation optimization of mobile scenes, and adaptive scheduling of mobile scenes. Using a flood disaster as an example, a prototype system was developed and used to conduct various experimental analyses. The experimental results show that the proposed scene construction and optimization method can effectively satisfy high frame-rate requirements for the rendering 3D disaster scenes in mobile VR. The contributions of this study are summarized as follows.

First, diverse modes of scene data representation were proposed. For terrain models, the organization of the scene data into terrain pyramid models of multiple tile sizes was proposed. For disaster simulation models, a LOD-based representation was proposed. These diverse representation modes for the scene data provide a basis for the efficient visualization of 3D disaster scenes based in mobile VR.

Second, strategies for optimizing the representation of mobile scenes and for adaptive scene scheduling were proposed. Loading simplified disaster simulation models, reducing the maximum tile level that can be loaded in the scene, and loading smaller-sized tiles are all approaches that can effectively improve the scene-rendering frame rate to satisfy the high frame-rate requirements for the rendering of 3D disaster scenes in mobile VR.

Finally, using a flood disaster as an example, a prototype system was developed based on the Cesium open-source framework. The prototype system can run on commonly used smartphones. Using a mobile HMD, the immersive exploration of and interaction with a 3D flood scene was implemented. The work described in this study can serve as a reference for the construction and optimization of other geographic scenes in mobile VR.

Despite the achievements described above, this paper has some shortcomings. For example, since the Cesium open-source framework currently supports only one DEM tile size (i.e., 64), a DEM pyramid model with only a tile size of 64 was provided in this study, so this paper only tested the

influence of the loading image tiles of various sizes on the scene-rendering frame rate. The influence of the loading DEM tiles of various sizes on the scene-rendering frame rate should be studied in the future.

Author Contributions: Y.H. and J.Z. conceived of the initial idea for this study, designed the experiments, and wrote the paper; Y.H., W.L., Y.Z., and H.Z. performed the experiments and analyzed the results; Q.Z. and H.Q. contributed the experimental data and supplied the infrastructure for the experiments; and Z.C., W.Y., and P.Z. provided important suggestions and reviews.

Funding: This paper was supported by the National Key Research and Development Program of China (Grant No. 2016YFC0803105), the Fundamental Research Funds for the Central Universities (2682018CX35), the National High Technology Research and Development Program of China (Grant No. 2015AA123901), and the Smart Guangzhou Spatio-temporal Information Cloud Platform Construction (Grant No. GZIT2016-A5-147).

Acknowledgments: The authors would like to thank the anonymous reviewers for their constructive comments and suggestions.

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

References

1. Yin, Y.P.; Wang, W.P.; Zhang, N.; Yan, J.K.; Wei, Y.J. The June 2017 Maoxian landslide: Geological disaster in an earthquake area after the Wenchuan Ms 8.0 earthquake. *Sci. China Technol. Sci.* **2017**, *60*, 1762–1766. [CrossRef]
2. Shultz, J.M.; Galea, S. Mitigating the Mental and Physical Health Consequences of Hurricane Harvey. *JAMA* **2017**, *318*, 1437–1438. [CrossRef] [PubMed]
3. Chan, Y.; Mori, M. Web-based Flood Monitoring System Using Google Earth and 3D GIS. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium, Vancouver, BC, Canada, 24–29 July 2011; pp. 1902–1905.
4. Lai, J.S.; Chang, W.Y.; Chan, Y.C.; Kang, S.C.; Tan, Y.C. Development of a 3D virtual environment for improving public participation: Case study—The Yuansantze Flood Diversion Works Project. *Adv. Eng. Inform.* **2011**, *25*, 208–223. [CrossRef]
5. Li, Y.; Gong, J.H.; Zhu, J.; Song, Y.Q.; Hu, Y.; Ye, L. Spatiotemporal simulation and risk analysis of dam-break flooding based on cellular automata. *Int. J. Geogr. Inf. Sci.* **2013**, *27*, 2043–2059. [CrossRef]
6. Evans, S.Y.; Todd, M.; Baines, I.; Hunt, T.; Morrison, G. Communicating flood risk through three-dimensional visualization. *Proc. Inst. Civ. Eng. Civ. Eng.* **2014**, *167*, 48–55.
7. Li, Y.; Gong, J.H.; Liu, H.; Zhu, J.; Song, Y.Q.; Liang, J.M. Real-time flood simulations using CA model driven by dynamic observation data. *Int. J. Geogr. Inf. Sci.* **2015**, *29*, 523–535. [CrossRef]
8. Zhu, J.; Yin, L.Z.; Wang, J.H.; Zhang, H.; Hu, Y.; Liu, Z.J. Dam-break flood routing simulation and scale effect analysis based on virtual geographic environment. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2015**, *8*, 105–113. [CrossRef]
9. Zhu, J.; Zhang, H.; Yang, X.F.; Yin, L.Z.; Li, Y.; Hu, Y.; Zhang, X. A collaborative virtual geographic environment for emergency dam-break simulation and risk analysis. *J. Spat. Sci.* **2016**, *61*, 133–155. [CrossRef]
10. Curebal, I.; Efe, R.; Ozdemir, H.; Soykan, A.; Sonmez, S. GIS-based approach for flood analysis: Case study of Kecidere flash flood event (Turkey). *Geocarto Int.* **2016**, *31*, 355–366. [CrossRef]
11. Winkler, D.; Zischg, J.; Rauch, W. Virtual reality in urban water management: Communicating urban flooding with particle-based CFD simulations. *Water Sci. Technol.* **2018**, *77*, 518–524. [CrossRef] [PubMed]
12. Lu, X.Z.; Han, B.; Hori, M.; Xiong, C.; Xu, Z. A coarse-grained parallel approach for seismic damage simulations of urban areas based on refined models and GPU/CPU cooperative computing. *Adv. Eng. Softw.* **2014**, *70*, 90–103. [CrossRef]
13. Li, B.R.; Wu, J.P.; Pan, M.; Huang, J. Application of 3D WebGIS and real-time technique in earthquake information publishing and visualization. *Earthq. Sci.* **2015**, *28*, 223–231. [CrossRef]
14. Xiong, C.; Lu, X.Z.; Hori, M.; Guan, H.; Xu, Z. Building seismic response and visualization using 3D urban polygonal modeling. *Autom. Constr.* **2015**, *55*, 25–34. [CrossRef]
15. Xu, Z.; Lu, X.Z.; Law, K.H. A computational framework for regional seismic simulation of buildings with multiple fidelity models. *Adv. Eng. Softw.* **2016**, *99*, 100–110. [CrossRef]

16. Redweik, P.; Teves-Costa, P.; Vilas-Boas, I.; Santos, T. 3D City Models as a Visual Support Tool for the Analysis of Buildings Seismic Vulnerability: The Case of Lisbon. *Int. J. Disaster Risk Sci.* **2017**, *8*, 308–325. [CrossRef]
17. Wu, J.; Chen, G.Q.; Zheng, L.; Zhang, Y.B. GIS-based Numerical Modelling of Debris Flow Motion across Three-dimensional Terrain. *J. Mt. Sci.* **2013**, *10*, 522–531. [CrossRef]
18. Liu, J.Q.; Tang, H.M.; Zhang, J.Q.; Shi, T.T. Glass landslide: The 3D visualization makes study of landslide transparent and virtualized. *Environ. Earth Sci.* **2014**, *72*, 3847–3856. [CrossRef]
19. Yu, M.; Huang, Y.; Xu, Q.; Guo, P.; Dai, Z.L. Application of virtual earth in 3D terrain modeling to visual analysis of large-scale geological disasters in mountainous areas. *Environ. Earth Sci.* **2016**, *75*, 563. [CrossRef]
20. Kuester, F.; Hutchinson, T.C. A virtualized laboratory for earthquake engineering education. *Comput. Appl. Eng. Educ.* **2006**, *15*, 15–29. [CrossRef]
21. Kellogg, L.H.; Bawden, G.W.; Bemardin, T.; Billen, M.; Cowgill, E.; Hamann, B.; Jadamec, M.; Kreylos, O.; Staadt, O.; Sumner, D. Interactive visualization to advance earthquake simulation. *Pure Appl. Geophys.* **2008**, *165*, 621–633. [CrossRef]
22. Zaalberg, R.; Midden, C.J.H. Living behind dikes: Mimicking flooding experiences. *Risk Anal.* **2013**, *33*, 866–876. [CrossRef] [PubMed]
23. Gong, X.L.; Liu, Y.J.; Jiao, Y.; Wang, B.J.; Zhou, J.C.; Yu, H.Y. A novel earthquake education system based on virtual reality. *IEICE Trans. Inf. Syst.* **2015**, *E98D*, 2242–2249. [CrossRef]
24. Philips, A.; Walz, A.; Bergner, A.; Graeff, T.; Heistermann, M.; Kienzler, S.; Korup, O.; Lipp, T.; Schwanghart, W.; Zeilinger, G. Immersive 3D geovisualization in higher education. *J. Geogr. High. Educ.* **2015**, *39*, 437–449. [CrossRef]
25. Li, C.Y.; Liang, W.; Quigley, C.; Zhao, Y.B.; Yu, L.F. Earthquake safety training through virtual drills. *IEEE Trans. Vis. Comput. Graph.* **2017**, *23*, 1388–1397. [CrossRef] [PubMed]
26. Liu, M.W.; Zhu, J.; Zhu, Q.; Qi, H.; Yin, L.Z.; Zhang, X.; Feng, B.; He, H.G.; Yang, W.J.; Chen, L.Y. Optimization of simulation and visualization analysis of dam-failure flood disaster for diverse computing systems. *Int. J. Geogr. Inf. Sci.* **2017**, *31*, 1891–1906. [CrossRef]
27. Voigt, S.; Giulio-Tonolo, F.; Lyons, J.; Kucera, J.; Jones, B.; Schneiderhan, T.; Platzeck, G.; Kaku, K.; Hazanika, M.K.; Czaran, L. Global trends in satellite-based emergency mapping. *Science* **2016**, *353*, 247–252. [CrossRef] [PubMed]
28. Zou, M.; Yuan, Y. China's comprehensive disaster reduction. *Int. J. Disaster Risk Sci.* **2010**, *1*, 24–32.
29. Wu, Q.; Zhou, W.F.; Guan, E.T. Emergency responses to water disasters in coalmines, China. *Environ. Geol.* **2009**, *58*, 95–100. [CrossRef]
30. Jeong, K.; Kim, J. Event-Centered Maze Generation Method for Mobile Virtual Reality Applications. *Symmetry* **2016**, *8*, 120. [CrossRef]
31. Powell, W.; Powell, V.; Brown, P.; Cook, M.; Uddin, J. Getting Around in Google Cardboard—Exploring Navigation Preferences with Low-Cost mobile VR. In Proceedings of the 2nd IEEE Workshop on Everyday Virtual Reality, Greenville, SC, USA, 20 March 2016; pp. 5–8.
32. Han, S.; Kim, J. A Study on Immersion of Hand Interaction for Mobile Platform Virtual Reality Contents. *Symmetry* **2017**, *9*, 22. [CrossRef]
33. Millen, M.I.; Kreylos, O.; Hanann, B.; Jadamec, M.A.; Kellogg, L.H.; Staadt, O.; Sumner, D.Y. A geoscience perspective on immersive 3D gridded data visualization. *Comput. Geosci.* **2008**, *34*, 1056–1072.
34. Boulos, M.N.K.; Lu, Z.H.; Guerrero, P.; Jennett, C.; Steed, A. From urban planning and emergency training to Pokémon Go: Applications of virtual reality GIS (VRGIS) and augmented reality GIS (ARGIS) in personal, public and environmental health. *Int. J. Health Geogr.* **2017**, *16*, 7. [CrossRef] [PubMed]
35. Noguera, J.M.; Segura, R.J.; Ogayar, C.J.; Joan-Arinyo, R. Navigating large terrains using commodity mobile devices. *Comput. Geosci.* **2011**, *37*, 1218–1233. [CrossRef]
36. Noguera, J.M.; Barranco, M.J.; Segura, R.J.; Martinez, L. A mobile 3D-GIS hybrid recommender system for tourism. *Inf. Sci.* **2012**, *215*, 37–52. [CrossRef]
37. Noguera, J.M.; Segura, R.J.; Ogayar, C.J.; Joan-Arinyo, R. A scalable architecture for 3D map navigation on mobile devices. *Pers. Ubiquitous Comput.* **2013**, *17*, 1487–1502. [CrossRef]
38. Magliocchetti, D.; Conti, G.; De-Amicis, R. I-MOVE: Towards the use of a mobile 3D GeoBrowser framework for urban mobility decision making. *Int. J. Interact. Des. Manuf.* **2012**, *6*, 205–214. [CrossRef]

39. Trujillo, A.; Suárez, J.P.; Calle-La, M.D.; Gómez, D.; Pedriza, A.; Santana, J.M. Glob3 Mobile: An Open Source Framework for Designing Virtual Globes on iOS and Android Mobile Devices. In *Progress and New Trends in 3D Geoinformation Sciences*; Pouliot, J., Daniel, S., Hubert, F., Zamyadi, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 211–229.
40. Suárez, J.P.; Trujillo, A.; Santana, J.M.; Calle-La, M.D.; Gómez, D. An efficient terrain Level of Detail implementation for mobile devices and performance study. *Comput. Environ. Urban Syst.* **2015**, *52*, 21–33. [CrossRef]
41. Santana, J.M.; Wendel, J.; Trujillo, A.; Suárez, J.P.; Simons, A.; Koch, A. Multimodal Location Based Services—Semantic 3D City Data as Virtual and Augmented Reality. In *Progress in Location-Base d Services 2016*; Gartner, G., Huang, H.S., Eds.; Springer: Berlin/Heidelberg, Germany, 2017; pp. 329–353.
42. Glob3 Mobile. Available online: http://glob3.sourceforge.net/mobile.html (accessed on 29 March 2018).
43. Kim, M.; Lee, J.; Jeon, C.; Kim, J. A study on interaction of gaze pointer-based user interface in mobile virtual reality environment. *Symmetry* **2017**, *9*, 189.
44. Ringe, S.; Kedia, R.; Poddar, A.; Patel, S. HTML5 Based Virtual Whiteboard for Real Time Interaction. In Proceedings of the 4th International Conference on Advances in Computing, Communication and Control, Mumbai, India, 1–2 April 2015; pp. 170–177.
45. Yin, L.Z.; Zhu, J.; Zhang, X.; Li, Y.; Wang, J.H.; Zhang, H.; Yang, X.F. Visual analysis and simulation of dam-break flood spatiotemporal process in a network environment. *Environ. Earth Sci.* **2015**, *74*, 7133–7146. [CrossRef]
46. Li, W.W.; Wang, S.Z. PolarGlobe: A web-wide virtual globe system for visualizing multidimensional, time-varying, big climate data. *Int. J. Geogr. Inf. Sci.* **2017**, *31*, 1562–1582. [CrossRef]
47. Christen, M.; Nebiket, S.; Loesch, B. Web-based large-scale 3D-geovisualisation using WebGL: The OpenWebGlobe project. *Int. J. Inf. Model.* **2012**, *1*, 16–25. [CrossRef]
48. Sample, J.T.; Loup, E. *Tile-Based Geospatial Information Systems: Principles and Practices*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 5–15.
49. Cesium. Available online: https://cesiumjs.org/ (accessed on 31 March 2018).

© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).