Article

A Heterogeneous Distributed Virtual Geographic Environment for Crowd Evacuation Experiments

Shen Shen[1,2], Jianhua Gong[1,3,*], Jianming Liang[4,*], Wenhang Li[1], Dong Zhang[1], Lin Huang[1,2] and Guoyong Zhang[1,2]

[1] State Key Laboratory of Remote Sensing Science, Institute of Remote Sensing and Digital Earth, Chinese Academy of Sciences, Beijing 100012, China; dslwz2002@163.com (S.S.); mylihang@163.com (W.L.); 852152455@qq.com (D.Z.); huanglin@radi.ac.cn (L.H.); zhangguoyong010@163.com (G.Z.);
[2] University of Chinese Academy of Sciences, Beijing 100049, China
[3] Zhejiang-CAS Application Center for Geoinformatics, Jiaxing 314199, China
[4] School of Life Sciences, Arizona State University, Tempe 85287, USA
* Correspondence: gongjh@radi.ac.cn (J.G.); ljm355@163.com (J.L.); Tel.: +86-10-6484-9299 (J.G. & J.L.)

Abstract: Due to their strong immersion and real-time interactivity, helmet-mounted VR devices are becoming increasingly popular. Based on these devices, an immersive virtual geographic environment (VGE) provides a promising method for research into crowd behavior in an emergency. However, the current cheaper helmet-mounted VR devices are not popular enough and will continue to coexist with PC-based systems for a long time. Therefore, a heterogeneous distributed virtual geographic environment (HDVGE) could be a feasible solution to solve the heterogeneous problems caused by various types of clients, and support implementation of virtual crowd evacuation experiments, with large numbers of concurrent participants. In this study, we developed an HDVGE framework and put forward a set of design principles to define the similarities between the real world and the VGE. We discussed the HDVGE architecture and proposed an abstract interaction layer, a protocol-based interaction algorithm and an adjusted dead reckoning algorithm to solve the heterogeneous distributed problems. We then implemented an HDVGE prototype system focusing on subway fire evacuation experiments. Two types of clients are considered in the system, PC and all-in-one VR. Finally, we evaluated the performances of the prototype system and the key algorithms. The results showed that in a low-latency LAN environment, the prototype system can smoothly support 90 concurrent users consisting of PC and all-in-one VR clients. HDVGE could serve as a new means of obtaining observational data about individual and group behavior in support of human geography research.

Keywords: virtual geographic environment; virtual geographic experiment; virtual reality; VRGIS; heterogeneous distributed clients

1. Introduction

1.1. Background

Geo-visualization started with two-dimensional (2D) mapping and later developed to include three-dimensional (3D) interactive rendering. Starting from the mid-1990s, with the advent of the virtual reality modeling language [1], the applications of virtual reality geographic information system (VRGIS) in various fields have been rapidly developing [2]. VRGIS offers an immersive GIS environment and could be considered an advanced form of geo-visualization [2]. Since VRGIS needs to process massive amounts of GIS data, and render VR graphics at the same time, VRGIS applications at this time can only run on high-end workstations, even requiring supercomputers. A virtual geographic environment (VGE) is a VRGIS-based platform for multidimensional visualization, dynamic process simulation and geo-collaboration [3, 4]. The second wave of VR has brought better graphics hardware and cheaper helmet-mounted display (HMD), greatly improving the availability and affordability of VR technology. This will also promote an upgrade of VRGIS, and facilitate the development of new theories and methods of geo-visualization, geo-analysis and geo-collaboration.

VR-based virtual experiments have been widely used in spatiotemporal behavioral research. In the field of cognitive behavior research, virtual behavioral experiments have been used to study spatial cognition, path selection, obstacle avoidance and other factors [5–9]. In the game industry, in order to improve game design and enhance the gaming experience, data mining and visualization have been used to analyze users' spatiotemporal behaviors in massively multiplayer online role-playing games (MMORPGs) [10–12]. In education and training, virtual reality simulators such as flight [13], driving [14], fire escape [15] are used to train the user's skills and enhance learning effectiveness. In the field of urban design and planning, the impact of the urban environment upon pedestrian decision-making behavior [16], pre-occupancy assessment [17], and guidance layout [18] also need the support of observational data about users' spatiotemporal behaviors, which could be easily acquired in the virtual environment.

In recent years, there have mainly been three crowd behavior research methods. First, computer vision technology has been used to extract pedestrian motion trajectories from surveillance video and other multimedia data. With trajectories, researchers can identify and analyze pedestrian behavioral patterns and the characteristics of spatial-temporal motion [15, 16][21, 22]. The second has been to use a classic social force model [23, 24] and field model [25,26] to model, simulate and analyze pedestrian behavior in specific situations [27,28]. Third, controlled real-world crowd experiments [29,30] have been designed with particular research goals in mind, in order to obtain observations that faithfully represent pedestrian movement patterns in the real world.

Real-world evacuation experiments are difficult to implement due to two reasons. First, it is difficult to resolve the safety issues in evacuation experiments, involving the participation of real people. Second, it is difficult to capture and quantitatively describe the spatiotemporal behavior observed in emergency scenarios. A virtual geographical experiment (VGEx) is becoming a promising research method. Immersed in a helmet VGE, the user not only has a strong sense of presence and immersion, but can flexibly control an avatar to evacuate in an emergency scene. This new form of experiment not only avoids the aforementioned safety problems, but can also faithfully capture the behavioral characteristics of real people. This would greatly facilitate behavior analysis and rule discovery. However, most of the current emergency drill systems have been designed only for single-users [15][31, 32]. There are few multi-user collaborative experimental platforms to support the study of crowd behavior.

1.2. *Related works*

Head-mounted VR devices can provide participants with a strong sense of immersion and real-time interactivity, which can enhance the effectiveness of existing research. Moussaïd et al. [33] constructed a multi-person collaborative virtual environment. They carried out crowd movement experiments in both real and virtual scenes. They also proved the feasibility of using a shared 3D virtual environment to carry out crowd experiments by involving real people. A cave automatic virtual environment (CAVE) is an immersive virtual reality environment where projectors are directed to between three and six of the walls of a room-sized cube [34]. Weiya Chen [35] used a large CAVE-like system to design an immersive, multi-user and multi-sensor virtual environment. The system used infrared devices to track user movements, and used shutter glasses to provide immersion and access to head activity data. However, the virtual environment in [33] is not sufficiently immersive, and large CAVE-like system devices in [35] are expensive.

Currently, there are many multiplayer online communities and games, such as Second Life and World of Warcraft. However, they are not suitable for user behavior research. On the one hand, we cannot obtain user behavior data from them. On the other hand, they are not designed for a specific experimental purpose.

In reality, the widespread adoption of helmet-mounted VR devices will require a long time, and until then, we will continue to depend upon PC based systems. Therefore, to carry out large scale virtual experiments for crowd evacuation, we need to solve the compatibility problems arising from heterogeneous clients. The heterogeneous issues mainly come from the following three aspects: 1) The computational capacity varies greatly across clients, for example, PCs usually have much higher performance all-in-one VR units; therefore, how can different configurations of heterogeneous computing platforms be made compatible with each other? 2) Participants engage in experiments with heterogeneous interactions. How can the consistency of user experience across different clients be ensured? 3) The LAN is not an ideal network environment for organizing large numbers of users to participate in evacuation experiments. If necessary, organizers may need to use the Internet. Lastly, how can a heterogeneous network environment be supported?

In this paper, we propose the following methods to solve the above heterogeneous problems. We use several optimization methods to ensure that our system runs smoothly even on low-configuration computing platforms. We propose an abstract interaction layer to adapt to the interactions of heterogeneous devices, and use protocol-based interaction algorithms to ensure the consistency of user experience across different clients. We use a distributed architecture design to solve the problem of the non-synchronization of user states caused by network delays, so that the system can be deployed in heterogeneous networks. This design allows us to conduct virtual experiments both using the LAN and the Internet, which could facilitate participation from different geographic locations. The specific algorithms will be described in section 3.2.

In section 2, we introduce the conceptual framework and design principles of heterogeneous distributed virtual geographic environment (HDVGE). Then, we design the overall architecture and key technologies of the experiment platform in section 3. In section 4, we implement a prototype system for crowd evacuation in a subway fire scene. Through performance analysis, we demonstrate the capability of this prototype system to support heterogeneous distributed virtual experiments for crowd evacuation. The proposed system can serve as a data collection tool for further behavior analysis. In section 5, we discuss the key issues in the experiments. In the end, we summarize the conclusions and highlight future work.

2. HDVGE conceptual framework

2.1. Conceptual framework

The human-environment relationship refers to the relationship between the existence and development of human society, or the human activities and the geographical environment [36]. The geographical environment here is considered as the entire geographic environment encompassing natural and human elements. They are intertwined in accordance with certain rules and are closely integrated. The literature [37] suggests that VGEx can be categorized into virtual natural geographic experiments and virtual human geographic experiments. Virtual natural and human elements constitute the “environment”, while individuals, groups, and society in the VGE constitute the “human” in a virtual experiment. Due to network and other technological limitations, collaborative VGEx, even based on distributed technology, cannot normally support the number of concurrent users of a real human society. However, virtual crowd experiments at the individual or group level could be affordable. According to the human-environment relationship theory [36], we categorize the VGEx into three types of elements: human, entity and environment. These elements and their relationships are shown in Figure 1.

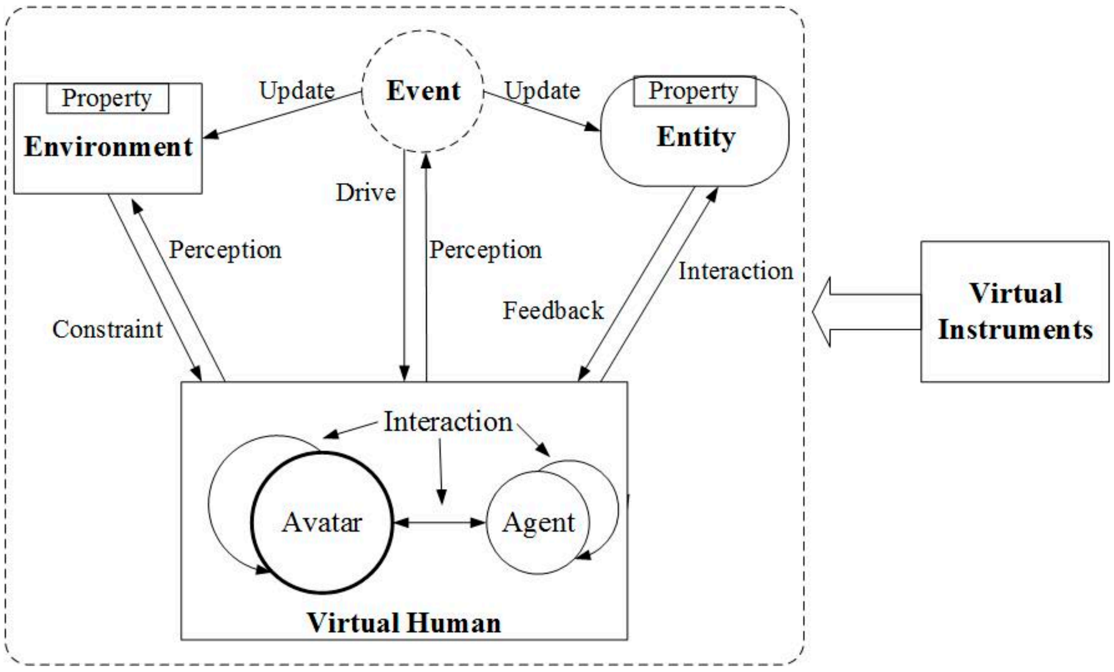

Figure 1 HDVGE conceptual framework

- Virtual Human：HDVGE regards the human as the core, and emphasizes the subjectivity of human beings. The virtual human in the HDVGE include avatars controlled by real users and agents driven by computer programs. They can interact with other elements as well as themselves, such as interaction between avatars or avatars and agents. Multiple virtual human form a group through social relations, roles and tasks. Massive groups in collective activities form a crowd.
- Environment：The environment is the background of the VGEx and is the static part of the VGE. We mainly use 3D models to simulate the real geographical environment, including terrain, vegetation, architecture, etc., which constitute the physical part of the VGE. Like the real environment, it can be perceived and recognized. At the same time, different environments will constrain virtual human’s behaviors.
- Entity：Entity refers to dynamic variable entities in VGE. They have the ability to interact with humans. They can not only be perceived by humans but can also provide feedback to the humans. For example, small obstacles in evacuation process, such as desks and chairs, can affect the route choice of virtual humans, meanwhile, their state variables can also be changed by the humans. In HDVGE, how the consistency of entity state variables in heterogeneous clients can be maintained is an essential issue.

In addition, HDVGE includes an important non-physical element, the event. An event can not only express natural and human geographic processes but can also express their interactions, such as the transfer of crowds in the course of mountain floods, and the evacuation of crowds during subway fires. Events are represented by the updating of attributes of entities and environments. They can be perceived by humans, and they can also drive people to respond. From the experiment organizer’s point of view, virtual instruments are their tools for observing and documenting the spatiotemporal states of various elements in virtual experiments.

2.2. HDVGE design principles

In the literature [38], the similarities in human behavior between the virtual and real worlds have been studied, and a mapping principle been proposed. As the research purposes and objects differ from one virtual world to another, a research frame based on four aspects has been put forward: group size, traditional controls and independent variables, contextual and social architecture factors, and directionality. Similarly, the virtual-real geographic similarity principal has been proposed in the literature [37], which establishes similar principles from four aspects: geographic space-time, geographic attribute, geographic attributes group and geographic spatial cognition. The higher the similarity between the virtual and real, the closer is user behavior in the virtual experiment to the real-world. From the above principles, we propose the following HDVGE design principles.

- Similarity in geographic space-time: The time and space in VGE should be similar to that in the real geographic environment (RGE). That is, the spatial scale and time scale in VGE should be identical to the real ones. This similarity provides principles for modeling the 3D virtual environment and process simulation. It requires strict reference to the size and proportion of real space when modeling VGE. Additionally, the time scale of the VGE cannot be changed.
- Similarity in spatial attributes: The spatial attributes and distributions of entities and processes in the VGE should be similar to that in the RGE. VGEx includes the simulation of processes in natural geography and human geography. This principle stipulates that the modeling and presentation of objects and geographic processes should be similar to reality.
- Similarity in group composition: Through observation of pedestrians in public places [39], at least 70% of pedestrians in a given population are not traveling alone, but walk in groups. In a VGEx for crowd evacuation, group composition and member attributes must be similar to reality. Because of the limitation of the 3D modeling, VGEx cannot provide every user with an elaborate avatar. We use avatar models which are easy to discern to represent the members of the group, while those who are outside the group use an avatar with a different appearance. This similarity provides the basis for group modeling, observation, record and analysis.
- Similarity in perception: The subject of VGEx perceives the environment, entities, other subjects' spatiotemporal positions, attributes and group relations from a first-person perspective. The results of this perception process should be similar to the perception results of RGE. For example, during a fire, the subjects' perceptions of the evacuating crowd and their own companions in the VGEx, should be similar to those in an actual fire environment. This similarity could stimulate the subject to behave similarly to reality. Therefore, this similarity provides the framework and principles for virtual scene design, process simulation, and interactions between multiple subjects.

In addition, HDVGE designs are also constrained by factors such as VR device performance, usability, and user experience [40]. The design of an HDVGE should consider these aspects in an integrated manner. The ultimate design would reflect a compromise between the various factors mentioned above.

3. HDVGE architecture and key technologies

3.1. HDVGE Architecture

According to the framework and design principles described in section 2, considering user experience, system performance and research objectives, we propose a detailed architecture for an HDVGE, as shown in Figure 2.

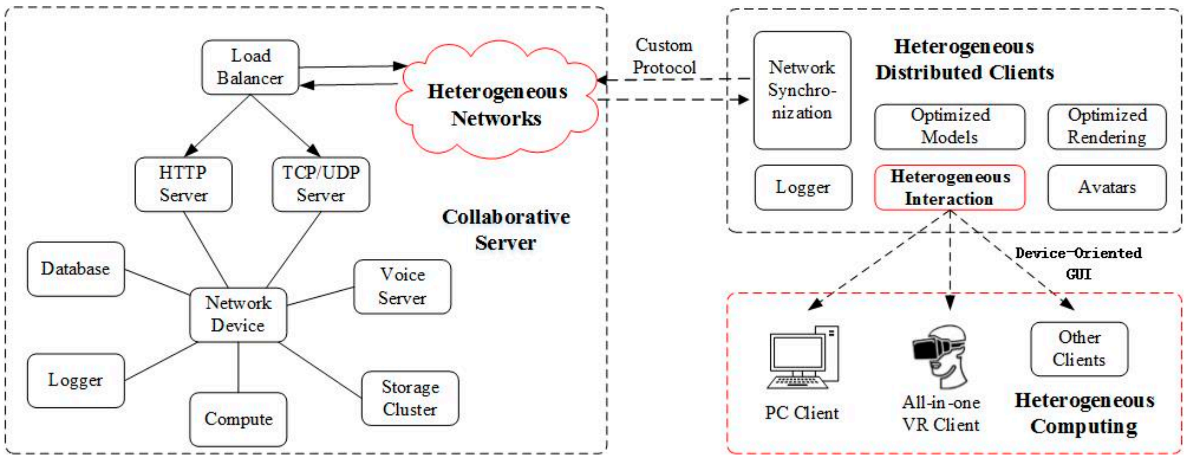

Figure 2 Architecture of HDVGE

- Collaborative Server

A collaborative server is primarily responsible for providing network services to the HDVGE. The VGEx for crowd evacuation often requires multiple participants distributed in different geographic locations. With this in mind, we adopted an authoritative-server-based architecture instead of a peer-to-peer client. The server supports deployment in heterogeneous network environments, to meet the experimental needs in either a LAN or WAN. The authoritative server maintains the states of all the objects in the virtual scene and is responsible for computing and updating them. Each operation on a client requires sending a synchronization request to the server. The server periodically performs object status verification, computation, updating, and then sends the latest statuses back to the target client. Due to the performance of heterogeneous clients varying enormously, this architecture shifts computation stress from the client side to the server side, enabling clients to focus on the high-fidelity rendering of HDVGE. In addition, the architecture is easy to scale given the possibility of large numbers of concurrent users.

The collaborative server can provide a variety of services. The TCP service is mainly used for frequently updated user data, such as the location, orientation and action of the avatar. The HTTP service is mainly used for transactional network requests, such as user authentication and background management. Voice communication between users in a group is achieved through voice services. The database is mainly used to store structured data during the experiment, while the storage cluster is mainly used to store unstructured data, and data that needs to be serialized. The logging module periodically records the status and behavior of all users in the experiment. Due to the high real-time requirements and interactivity, when there are large number of massive concurrent users, the load balancing module is responsible for the scheduling of computing resources and storage, and distributing the tasks according to the actual situation.

- Heterogeneous Clients

Although all-in-one VR offers a better sense of immersion, its availability and performance remain limited. As large numbers of users are required in virtual crowd evacuation experiments, the system has been designed to be able to interact with heterogeneous clients so that PC users can be included. All-in-one VR uses high-precision sensors and gamepads as input devices, HMD as the output device, while a PC uses the traditional mouse and keyboard as input devices, and the monitor as the output device. To accommodate interactions between heterogeneous clients, we designed a device-oriented graphic user interface (DGUI). On a PC, it appears as a screen-space user interface, while on an all-in-one VR, it is a view-centered user interface that follows the user's head movement. The DGUI is mainly used to display user-related information. The interaction algorithm for heterogeneous clients will be described in detail in section 3.2.

As an HDVGE is a common workspace shared by multiple users, 3D modeling of the environment must be sufficiently photorealistic. In light of the limited computing and rendering capabilities of the heterogeneous clients, 3D models and scene rendering must also be optimized to meet the requirements of user experience and system availability. A heterogeneous distributed virtual environment is a trade-off between high fidelity and availability. As an important reference point for users to perceive the virtual environment, avatars' skin, bones and animations also need to meet the above requirements. We distinguish between different groups of avatars using easily identifiable colors of clothing. The avatars' skeletal animations meet the non-verbal communication needs between users in the HDVGE.

We add a logging module to the heterogeneous client, which is responsible for recording the local user's locations, orientations, actions and other status information. This module is different from the server-side logging module, which records the status data for all users. Although the log data have some degree of redundancy, it increases the reliability of data logging.

The network synchronization module is used mainly to communicate with the server. Data transmission is based on the TCP protocol. Heterogeneous distributed clients perform collaborative tasks through the network. Its real-time and interactivity are important factors that affect the user experience. The architecture of the HDVGE uses the ideas of authoritative server and dumb client. All clients will send their own status changes to the server, and then the collaborative server forwards them to each client as requested. This architecture can reduce the client's computing pressure and avoid cheating. However, the disadvantage is that since all the data sent and received must first go through the collaborative server, the overall performance is greatly affected by the speed of the network. Therefore, the client prediction algorithm is very important. The algorithm will be described in detail in section 3.2.

3.2. Key technologies

3.2.1. Abstract interaction layer for heterogeneous clients

Interactive devices and methods vary greatly between HDVGE clients. Interactive devices on PC clients are keyboard, mouse and monitor. They use the mouse to control the viewpoint rotation, use the keyboard to control the viewpoint movement, trigger skeletal animation and other actions. Interactive devices of all-in-one VR clients mainly use the HMD, gamepad, and stereo screen. The HMD is equipped with a high-precision 9-axis sensor, which is a combination of three sensors: a 3-axis accelerometer, a 3-axis gyro and a 3-axis electronic compass. Among them, the HMD mainly uses the 3-axis gyroscope to measure, obtain the attitude parameters of the helmet, and then reconstruct the user's 3D motion. That is, the user controls the rotation of the viewpoint using the HMD, controls the viewpoint movement using the gamepad, and triggers the skeleton animation using buttons. Although the devices and methods vary considerably, the ultimate goals are the same. Therefore, in order to be compatible with different interactions between heterogeneous clients, we designed an abstract interaction layer (AIL), so that the different interactive methods can achieve the same results. Figure 3 shows a typical interaction process. The left and right sides of the figure represent the interaction process of the all-in-one VR and the PC client, respectively.

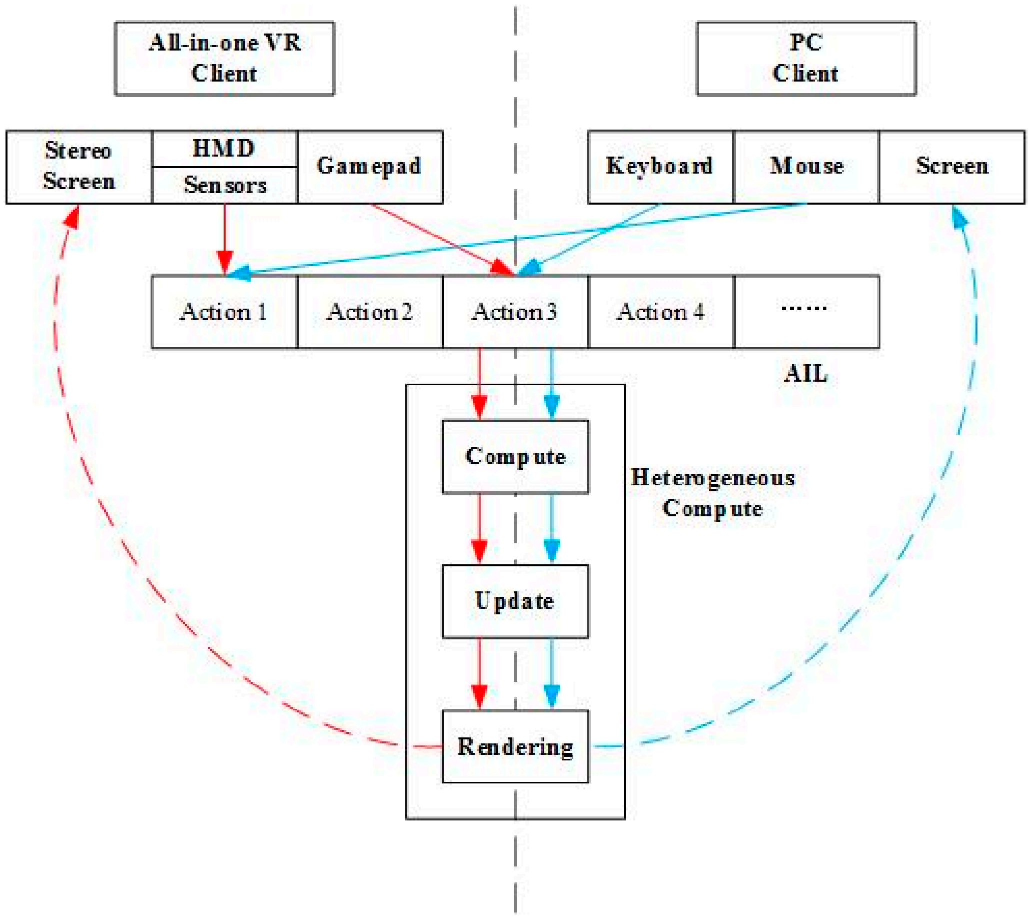

Figure 3 Flow chart of abstract interaction layer for heterogeneous clients

The AIL is a collection of predefined actions. It is responsible for converting interactions from heterogeneous clients into standard actions. First, we define the standard actions that can be recognized by system. Second, we establish a mapping relationship between the various types of operations from heterogeneous clients and AIL standard actions. This mapping relationship bridges the differences between heterogeneous clients, and guarantees that different operations from heterogeneous clients can produce the same effect. Third, according to the specific interaction action, each client computes and updates the rendering scene in its own heterogeneous computing platform. Finally, the rendering results are sent to the client's display device.

3.2.2. Protocol-based interactions between heterogeneous clients

The standard actions defined in AIL also provide a common language for interactions between the heterogeneous clients. For heterogeneous clients to communicate and interact with each other, we propose protocol-based interactions to implement the collaboration between the heterogeneous clients. The agreement is a data structure that the system appoints in advance for data exchange between the clients. It can be understood and applied by each client in its own form, thereby masking the differences between clients.

A typical data transmission process based on a custom protocol is shown in Figure 4. First, a user of an all-in-one VR changes his status locally. Then, based on the type of heterogeneous client, this interaction is mapped to standard actions by the AIL. Next, the system uses the custom protocol to encode the standard action. To improve network transmission efficiency, the encoded result is converted to binary form before being transmitted to the server. After receiving the status update from client 1, the server calculates and processes the status information and then forwards to other clients in the current scene.

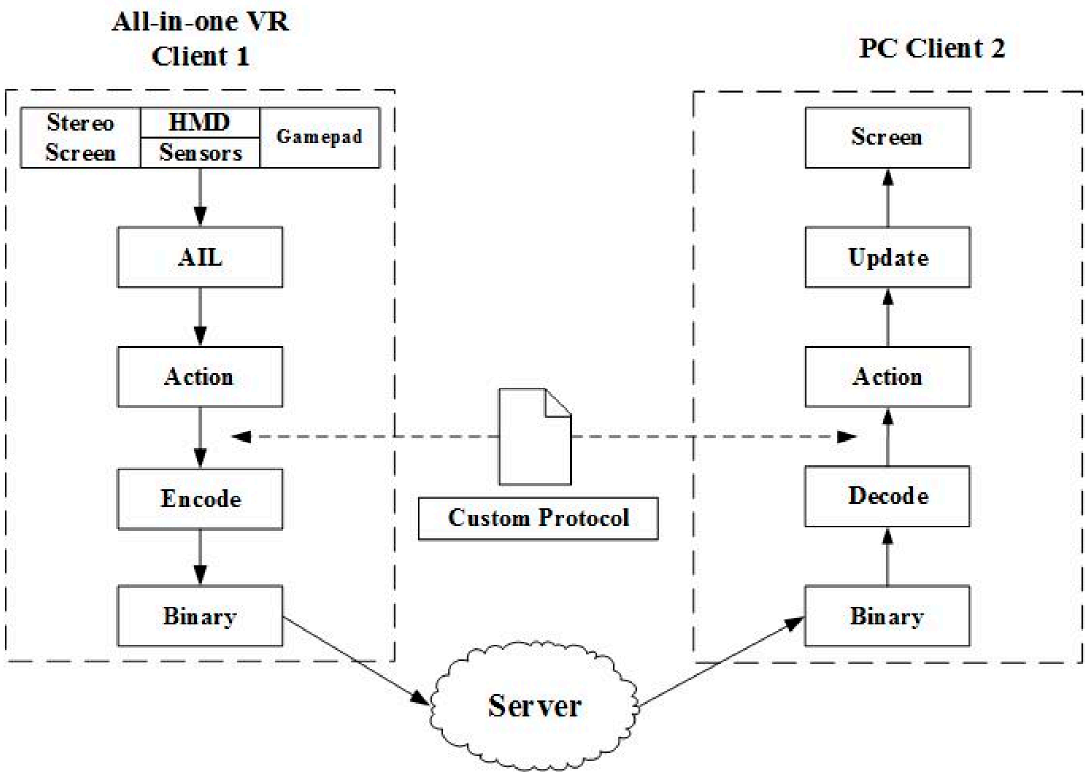

Figure 4 Flow chart of protocol-based interactions for heterogeneous clients

Let's take PC client 2 as an example. After receiving the binary data, client 2 first converts it to text data and decodes it. Then, it continues to restore the interaction of client 1 according to the custom protocol, and updates the local client 1 status. Finally, based on the latest status of client 1, client 2 performs calculation, rendering, and output display, and responds according to its needs and feedback. All client-side interactions are similar to this.

3.2.3. Adjusted dead reckoning algorithm for client prediction

The network environment in which the distributed client is located to a large extent determines the system's real-time experience. Network latency is a key issue affecting the overall performance of the system. In an HDVGE, not only does the status of the virtual avatars in the client needs to be synchronized but many other scene elements also require consistent maintenance through the server, such as interactions between avatars and entities, user entry and exit events, and the instantiation and deletion of networked entities. Virtual scene maintenance can ensure the consistency of scene, entities, avatars and other elements in each client to avoid user perception differences caused by distributed clients.

A typical client state synchronization process of an HDVGE is shown in Figure 5. We assume that network latency for each client is constant throughout the process. The initial position of client 1 is (10, 10), and it moves one unit along the x-axis. Client 1 sends a new status to the server while moving the local avatar. The data reaches the server after t1 time. Then, the server receives the new status sent by client 1 and starts to broadcast to other clients. The broadcasted data reaches client 1 after time t2 and reaches client 2 after time t3. At this point, client 2 can see the new status of client 1. In the process, the network delay of client 1 is (t1 + t2), while the new status of client 1 reaches client 2 after (t1 + t3). That is, the status of client 1 as seen by client 2 is actually the former's status before time (t1 + t3).

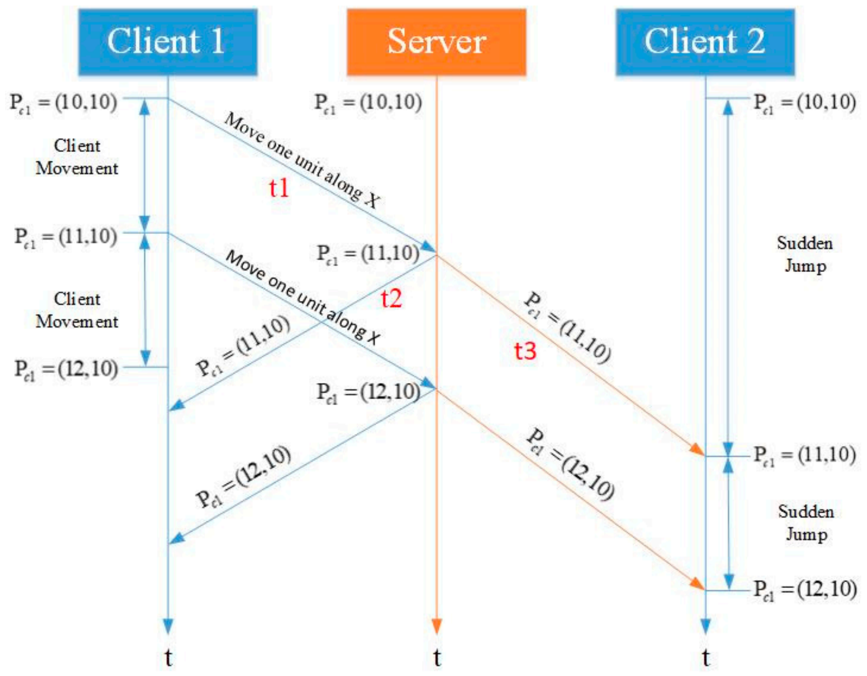

Figure 5 States synchronization process between clients

Out-of-sync status between distributed clients can cause serious problems. Assume that in the HDVGE for crowd evacuation experiment, users need to control the avatars to escape quickly from the scene of the fire. Client 1 and client 2 are in one group and need to escape together. Client 2 sees the status of client 1 before time (t1 + t3), which is slightly behind client 2. Thus, client 2 stops and waits for client 1. However, in fact, client 1 has already come to the front. At this point client 1 sees client 2 falling behind. Therefore, client 1, in turn, will need to stop and wait for client 2. It can be seen that the out-of-sync status between distributed clients will eventually make client 1 and client 2 stop moving. We need to predict the state of the next moment based on the client's current state, so that the avatars of different clients appear to be synchronized. We also need to minimize deviations between the true and the predicted values.

There are many algorithms that predict a moving object's future states based on the latest state. The most widely used are dead reckoning (DR) and the Kalman filter (KF). DR is an algorithm to predict the motion parameters of a moving object. It predicts the state of an object based on the latest position, velocity, and acceleration and is widely used in the fields of aviation and navigation [41]. Curtiss Murphy [42] uses projective velocity blending, which mixes the newly acquired velocity with the current velocity, and predicts the position combined with the time variable. The KF is an algorithm that estimates the state of a system from measured data. It is commonly used in guidance, navigation, and control systems. In computer vision applications, the KF is used for object tracking to predict an object's future location, to account for noise in an object's detected location, and to help associate multiple objects with their corresponding tracks [43]. Comparing the two, the DR algorithm is more widely used and has been applied in networked games [44,45]. Therefore, we adjusted the DR algorithm to provide client-side predictions, and compare its performance with KF.

The algorithm in [42] uses a fixed update rate to implement the DR algorithm. In HDVGE, the client uploads the data to the server only when its status data changes. Then, the server pushes the data to other subscribed clients. Thus, the client status data are not sent and received at regular intervals. Due to the asynchronous updating scheme, we cannot use the original algorithm to calculate the velocity blending factor needed for predicting the future position, and therefore we developed a modified form of the algorithm. The formulas of the modified algorithm are shown in (1) - (4).

$$\mathbf{V}_b = \mathbf{V}_0 + (\mathbf{V}_0^{'} - \mathbf{V}_0)T_w \tag{1}$$

Peer-reviewed version available at ISPRS Int. J. Geo-Inf. 2018, 7, 54; doi:10.3390/ijgi7020054

$$\mathbf{P}_t = \mathbf{P}_0 + \mathbf{V}_b T_t + \frac{1}{2}\mathbf{A}_0^{'} T_t^2 \tag{2}$$

$$\mathbf{P}_t^{'} = \mathbf{P}_0^{'} + \mathbf{V}_0^{'} T_t + \frac{1}{2}\mathbf{A}_0^{'} T_t^2 \tag{3}$$

$$\mathbf{Q}_t = \mathbf{P}_t + (\mathbf{P}_t^{'} - \mathbf{P}_t) T_w \tag{4}$$

The velocity blending factor T_w is a normalized value determined according to the client data update time. Formula (1) calculates the blended velocity $\mathbf{V}_b$ using the velocity blending factor, where $\mathbf{V}_0$ represents the current velocity, $\mathbf{V}_0^{'}$ represents the last known velocity. Formula (2) projects the future position $\mathbf{P}_t$ after T_t from the current position $\mathbf{P}_0$, the blended velocity $\mathbf{V}_b$ and the latest known acceleration $\mathbf{A}_0^{'}$, where T_t represents the time elapsed since the last data update. Formula (3) projects the future position $\mathbf{P}_t^{'}$ after T_t based on the last known position $\mathbf{P}_0^{'}$, last known velocity $\mathbf{V}_0^{'}$, and last known acceleration $\mathbf{A}_0^{'}$. Formula (4) blends the results of (2) (3) to obtain the final projected position Q_t.

As seen from the formulas, the predicted position is a linear combination of the current position, and the known position. During the operation of the system, each time the data are updated, new data will be used to correct the current data, to reduce errors and improve accuracy. We need to adjust the velocity blending factor based on the actual system data update time. The larger the T_w value, the greater the weight of the current positions; the smaller the T_w, the greater the weight of the last known position.

4. Prototype system

4.1. Heterogeneous distributed virtual evacuation prototype system

Based on the above architecture design and key algorithms, we implemented a heterogeneous distributed virtual evacuation prototype system. The system is based on a subway fire scene and can support multi-user collaborative virtual evacuation drills. The server side of the system uses the Smartfox Server as the TCP server for data synchronization, and Flask as the HTTP server. The client side uses the Unity3D game engine as the development platform. We use the all-in-one PicoVR and a mid-range PC as a heterogeneous interaction and computing client. The refresh frequency of the PicoVR HMD is 90 Hz. The monocular resolution is 1200*1080 and the field of view is 102 degrees. The all-in-one is equipped with a gamepad. With the high-precision 9-axis sensor in the HMD, the system enables 3 degrees of freedom interactions. The price of this all-in-one device is about $450, and the main hardware is a Qualcomm Snapdragon 820 CPU, Adreno 530 GPU, 4GB of RAM, and Qualcomm QCA6174A wireless card. As a result, the all-in-one no longer requires high-performance PC support, which reduces experiment costs.

The virtual scene mainly consists of a manually modeled 3D subway station, in which the platform is approximately 8 m in width and 90 m in length. There are four exits in total, which are labeled with A, B, C, and D, and located at the two ends of the platform. There are round pillars in the middle of the platform. A fire breaks out on one side of the subway, thus two exits on one end are blocked. The user-controlled avatar sets off from the center of the platform, climbs the stairs, crosses the gates, and finally reaches the correct exit. Considering the weak performance of the all-in-one, the prototype system does not use real-time lighting. Instead, the system uses some area light sources and bakes the lighting into lightmaps. To improve the fidelity of the fire scene, the system uses particle systems to simulate heavy black smoke. At the same time, the system establishes weak lighting to create a low-visibility scene. Accompanied by a sharp fire alarm, the system creates a sense of urgency to the user both visually and audibly. The pillars in the middle of the platform have a clear exit-point marking, which is self-illuminated to ensure clarity. The pillars in the middle of the platform have clear exit-oriented signs that use self-luminous materials to ensure a clear view.

The system uses low-precision 3D models as avatars. The number of triangles in each model are between 600 and 700. To make it easy to distinguish avatar's group information, we designed the coat texture of avatars of the same group to have the same color. We design three skeletal animations for each avatar, where "run" is used to represent the user's escape animation, "idle" is used to indicate that the user is not moving, and "greet" is used for non-verbal communication between the group members. We have implemented two different modes of interactions, all-in-one VR and PC, both of which use the first-person perspective. We use the device-oriented GUI to display the local user group, flag, correct exit, evacuation countdown, system prompts and other news. Both types of clients record the position, orientation, movement and other status data of the avatar in 0.3 second intervals. Figure 6 is a prototype system diagram.

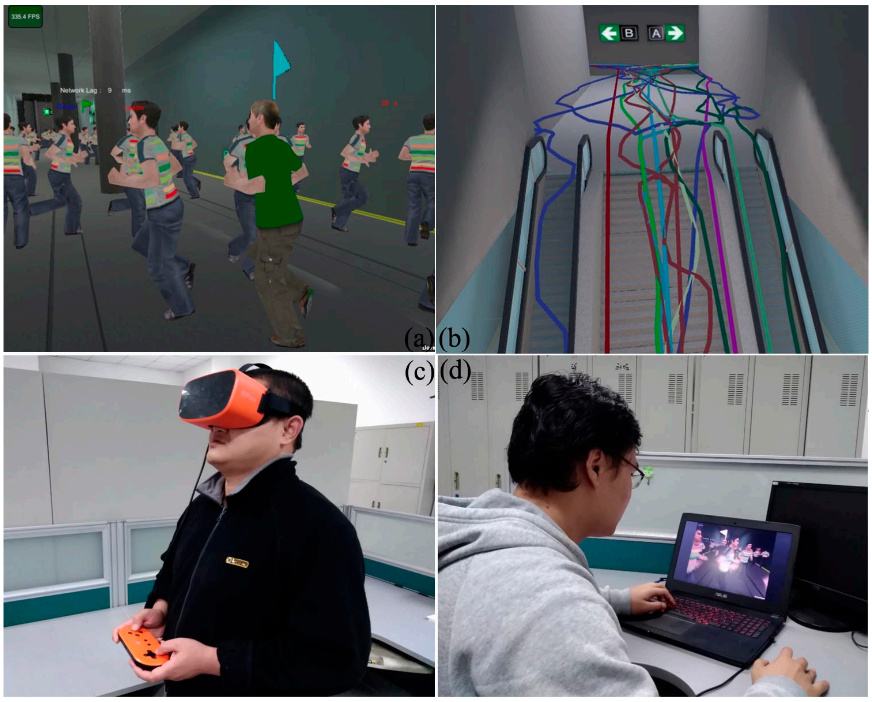

Figure 6 Prototype system (a) overview of subway scene; (b) trajectory visualization; (c) all-in-one VR client; (d) PC client

Since the LAN environment is pure, and it is easy to simulate complex conditions such as network latency, the prototype system server is deployed in the LAN. At the same time, we implement HTTP services to complete user authentication, configuration parameters and so on. We define in advance the data structures of the request and response between server and client, which are used to transfer the data such as status and message between the local and remote users. The server records the status, actions and events of all the users in the virtual scene, according to the data sent by the user.

4.2. Performance evaluation of key algorithms

Since the AIL and protocol-based interaction algorithms for heterogeneous clients cannot be measured using numeric values, we have supplemented the implementation of the prototype system. The availability of the system can prove the effectiveness of our additions. Therefore, only the adjusted dead reckoning (ADR) algorithm is evaluated here.

- Adjusted dead reckoning algorithm

We used a prototype system to conduct a virtual evacuation experiment for a crowd in a subway fire scene. Taking the trajectories of user activities collected in the experiment as an example, we implement the KF, with a constant acceleration model and ADR for position prediction. To study the running time and prediction accuracy of the algorithm under different update frequencies, we use a uniform distribution of fixed intervals to simulate the update frequency with some randomness. We use the total time consumed by the prediction algorithm to evaluate the algorithm's time complexity, and use root mean squared error (RMSE) to measure the deviation from the observed value to the true value.

The test results are shown in Figure 8. In this test, we implement the ADR algorithm that takes a velocity blending factor of 0.3. This means that the predicted value receives higher priority than the latest updated position data. As seen from Figure 7 (a), from the accuracy of the algorithm, the prediction error of the ADR algorithm is smaller than the KF algorithm by an average of 0.961 m, which is lower by 31.76%. From the running time, the single run time of ADR is almost negligible, while the KF algorithm requires an average of 0.23 ms. This is because KF needs to update the state transition model and covariance model in each time step, and performs a series of matrix multiplications. Therefore, we recommend that the ADR algorithm should be considered in 3D rendering programs that require high real-time performance. Figure 7 (b) shows the prediction results of the ADR algorithm and the KF algorithm. The XZ plane is the user's activity plane, and the positive Y axis represents the height value. The error in the prediction trajectory of the KF algorithm increased after the avatar moved vertically. The trajectory predicted by the ADR algorithm is subject to unsatisfactory accuracy when the avatar's acceleration changes, but the error is quite small in other places. This is more in line with the actual situation.

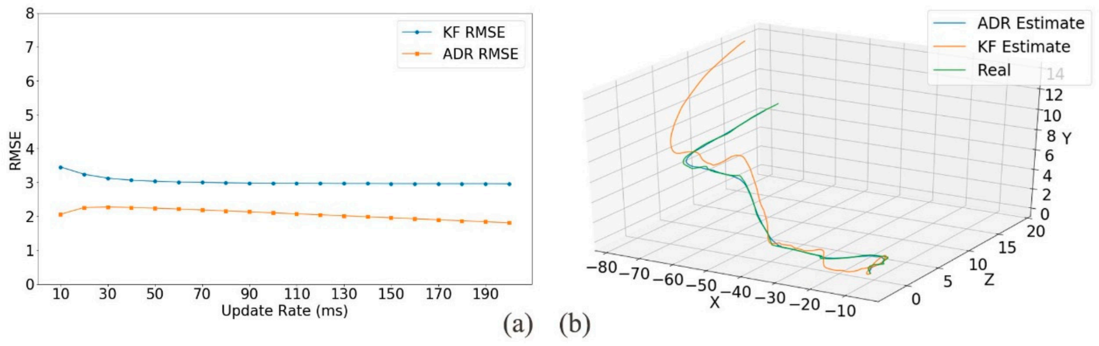

Figure 7 Comparison between ADR and KF

4.3. System overall performance test

To verify the usability of the prototype system, we conducted an overall system performance test. Here, an important question we aim to address is: with the heterogeneous clients of general configuration, how many concurrent users can the system support in distributed virtual experiments? We assume that the hardware and networks of all clients are the same. The performance metrics of the prototype system are mainly influenced by the number of concurrent users and the lag of packets. Therefore, we take them as two factors used in a factorial experiment. The number of concurrent users includes 5 levels, namely, 10, 30, 50, 70 and 90. To simulate different network environments, we use software to add delays to the packets sent and received. The packet lag includes 4 levels, namely, 0, 10, 20, and 30 milliseconds; that is, each replicate of the experiment contains all 20 treatments, and each treatment contains 5 replicates.

The prototype system performance indicators include resource consumption, rendering pressure, and network latency. Resource consumption is measured by CPU usage, memory usage and network throughput. The indicator of rendering pressure is the frame per second (FPS) of the client. Network latency is measured by the overall latency recorded by the client. We use a PC and an all-in-one VR as test clients, and develop a simulation program for simulating a given number of distributed clients. The main hardware in the PC: Intel (R) Core (TM) i5 750, NVIDIA GeForce GTX

650 and 8GB RAM. The configuration of the all-in-one VR is described in section 3.1. The main hardware in the server: Intel (R) Core (TM) i7 6700, NVIDIA GeForce GTX 1060, 8GB RAM.

At the beginning of the test, the simulation program created a specified number of concurrent avatars and allowed them to move randomly in the scene. The user controlled the avatar, ran through the crowd, and finally reached the other end on the subway platform from where the fire started. The server and the clients of two types recorded the resource consumption during the running of the program. The clients additionally recorded the frame rate and the overall network delay data. The result of each test is the average of each parameter in the process. We took the average of each indicator in each test as the test result.

4.4. Data analysis

As the hardware and computing capacity vary greatly between the server, PC and all-in-one VR, the evaluation indicators are also different, and we will discuss them separately.

- Server side

The resource consumption of the prototype system with different numbers of concurrent users on the server side is shown in Figure 8 (a) (b) (c). The system resource consumption increases with the number of concurrent users. With 90 concurrent users, this process takes up to 10% of the CPU. Memory usage increases significantly with the number of concurrent users, with the maximum being 320 MB. Network sent traffic (up to 3500KB per second) is several times higher than network received traffic (up to 500KB per second). This is because after the server received a user update, it sends the update to all other users. When the packet lag is 0, the network traffic both received and sent reaches the highest level. One of the possible reasons is that the latency has caused data packets to get stuck in the network, without reaching the server processing flow on time. Some packets are discarded due to timeout and are no longer being processed. This results in a reduction in the total network traffic. In general, the system's CPU usage is not high. This process does not occupy much of the server's resources, and more concurrent users can be supported.

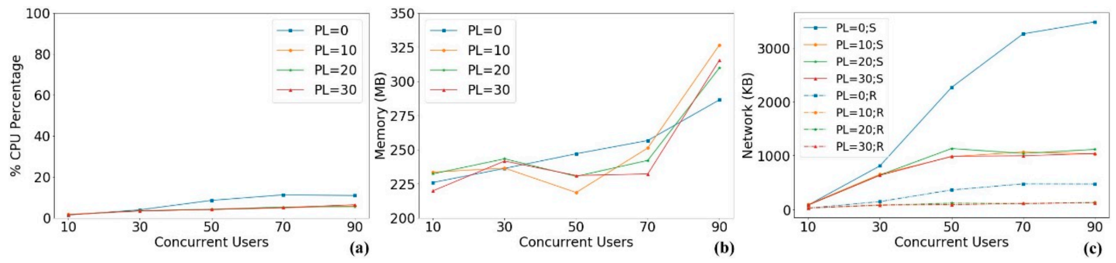

Figure 8 Server resource consumption with different numbers of concurrent users and packet lags (PL stands for packet lag. S refers to send traffic while R refers to received traffic. Abbreviations in other figures have the same meanings.)

- PC side

Figure 9 shows the resource consumption of the PC client under different numbers of concurrent user and packet lags. (a) shows that as concurrent users increase or as packet lags increase, CPU usage does not increase significantly. (b) shows that memory usage is mainly affected by the number of concurrent users. Since the client only needs to send the status data of the local user, the network sent traffic is stable. On the other hand, the client needs to receive the status data of all other remote users, so there is an increasing process in (c) as the number of concurrent user increases. However, an increase in packet lags causes network congestion and some data is discarded. With the increase of concurrent users, the decline of FPS in (d) is obvious, but it is still at a very high level. It can be seen from (e) that in the absence of packet lags, the increase in the number of concurrent users has no effect on the network delay on the PC side. However, once the packet lag is introduced, the impact of both factors on the overall network delay is approximately logarithmic. Overall, the prototype system is stable on this medium-configured computer.

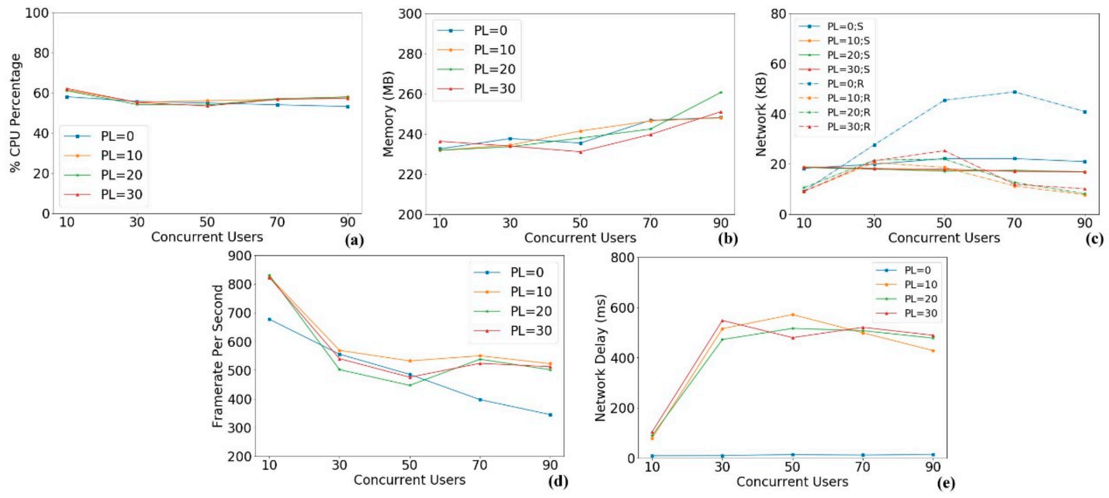

Figure 9 PC resource consumption in different number of concurrent users and different packet lags

- VR side

Because the all-in-one PicoVR system is Android, its performance is measured in a slightly different way. In the following indicators, CPU Time refers to the average of the total time consumed in the most recent 30 frames. The larger the value, the greater the overall pressure on the device. Memory refers to the used heap size. FPS has been limited by their SDK, up to 60 FPS.

In Figure 10, we can see from (a) that the number of concurrent users and packet lags have no significant impacts on CPU time-consuming. Used heap size in (b) increases with concurrent users, and has no obvious relationship with packet lag. The network traffic in (c) is similar to the PC client. The FPS in (d) shows a decreasing trend with the number of concurrent users but has no obvious relationship with packet lag. This shows that the packet lag does not affect the FPS. That is, packet lag can lead to poor interactivity, but it does not affect the real-time user experience. (e) shows that the impact of these two factors on the overall network delay in the VR client is also logarithmic.

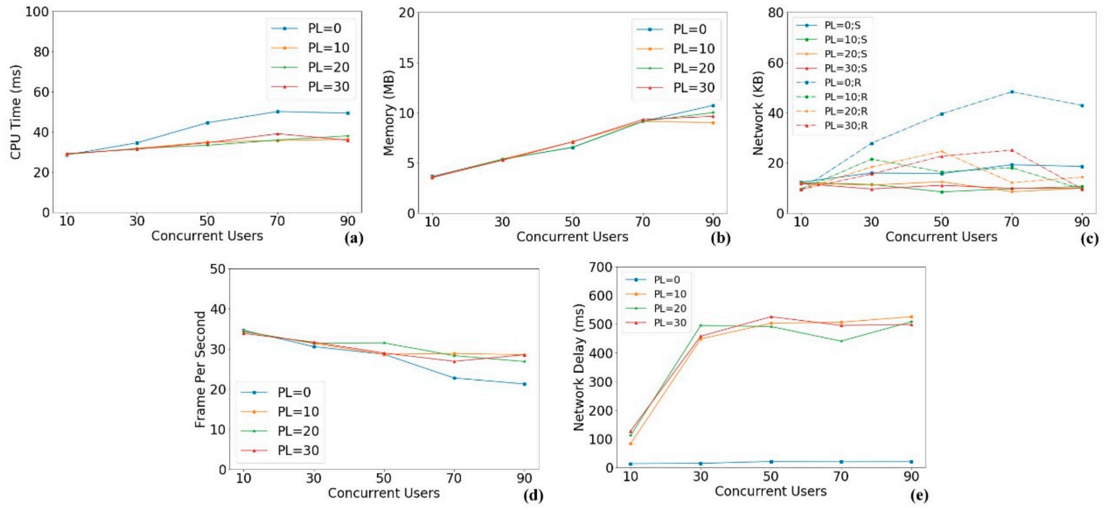

Figure 10 All-in-one VR' resource consumption for different numbers of concurrent users and packet lags

In summary, the overall performance of the server, PC client and all-in-one VR is stable and less demanding on system resources. The number of concurrent users does not have significant effect on the overall performance. The main bottleneck of system expansion depends on the performance of the all-in-one VR. Packet lag has a great impact on the overall network delay, and will make system performance decline rapidly. The conditions such as network congestion and packet loss, will have negative effect on system performance and user experience. In an actual experiment, high network latency should always be avoided.

5. Discussion

Organizing large numbers of people over a network to conduct virtual experiments is a challenging task. The LAN environment is pure and has low network delays, generally less than 20 ms. Our HDVGE prototype system supports the participation of 90 or more concurrent users in collaborative virtual evacuation experiments. The main limitation is the performance of the all-in-one VR. Under the conditions of 90 concurrent users and no packet lags, the PC client can maintain a rendering performance of approximately 300 FPS, while the all-in-one VR can only run at approximately 20 FPS, which could barely meet the requirements for a real-time user experience and interactivity. With the continuous development of hardware, all-in-one VR will become more powerful and better meet the experimental requirements. At present, the heterogeneous distributed architecture is probably the most effective option to conduct virtual experiments with high numbers of concurrent users.

The Internet is a complicated network environment subject to high latency due to the large number of users distributed around the world. Network delay has more influence on real-time and interactivity than numbers of concurrent users. It should be noted that overall network delay and packet lag is not a simple linear relationship. This shows that the Internet experimental environment may introduce more complex factors, which would make it more difficult to meet the real-time and interactivity requirements of HDVGE. When conducting virtual experiments on an HDVGE, the network environment should be selected according to the actual experimental needs.

When conducting virtual crowd evacuations in emergency scenarios, a very important question to consider, is that of how tension can be created for the participants? As mentioned in the literature [25], there are several main ways. First, create more realistic emergency elements, such as dim lights and thick smoke; second, set a time limit using a countdown to urge participants to escape; third, develop experimental policies, the shorter the time needed for a successful evacuation is, the better the payoff will be. In practice, these methods all play a role in the experiment, but the immersion and presence brought by the HMD VR device can provide a better user experience. For non-immersive devices, the 3D virtual environment on the computer screen is independent of the participant’s cognitive space. While in a helmet-based VGE, the virtual environment space and cognitive space are closely coupled, so that the user’s cognition of the virtual world is consistent with that of the real world.

6. Conclusions and future works

In recent years, low-cost HMD VR devices are becoming popular, but they are not sufficiently widely available. Hence, they will coexist with PC based systems for a long time. To solve the heterogeneous problems caused by various types of clients and to support implementation of virtual crowd evacuation experiments with large numbers of concurrent participants, the HDVGE represents a feasible solution. In this paper, we present HDVGE as a practical solution and demonstrate the technical feasibility of HDVGE.

First, we have proposed an HDVGE framework for crowd evacuation, and analyzed the design principles of the HDVGE platform based on this framework. We then designed the architecture and the key technologies of the experiment platform. Finally, using a subway fire as an example, we implemented an HDVGE prototype system for crowd evacuation. Through testing and analyzing the key algorithms and overall performance, we demonstrated the effectiveness of the proposed system.

The results show that in a low-latency LAN environment, the system could support 90 concurrent users for collaborative virtual experiments as heterogeneous distributed clients. System performance bottlenecks were dependent on the all-in-one VR. Packet lag had a great impact on the overall network delay, and would result in a rapid decline in the system performance, leading to further issues such as network congestion, packet loss, etc. In actual experiments, high-latency network environments should be avoided.

The HDVGE could serve as a new means of obtaining observational data on individual and group behaviors. Future work will include: 1) Crowd behavior varies greatly in different scenes; so analyzing the behavioral data of individuals and groups in different scenarios may lead to different conclusions. 2) Compared with the data of a real scene experiment, we will analyze the similarities in the behaviors between the virtual and real scenes.

Acknowledgements: This research is supported by the National Natural Science Foundation of China (41371387), National Key Research and Development Program of China (2016YFB0502502), Sensing Location Semantics Based on Multisource Big Geodata (2017YFB0503602), Pre-research Project of Equipment Development Department (315050501) and Internal Program of SLRSS China（Y7Y00200KZ）.

Author Contributions: Shen Shen, Jianhua Gong, Jianming Liang and Wenhang Li conceived and designed the methods; Dong Zhang, Lin Huang and Guoyong Zhang performed the experiments; Shen Shen analyzed the data and wrote the paper; all the authors reviewed and edited the manuscript.

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

References

1. Raggett, D. Extending WWW to support platform independent virtual reality. In *Proc. Internet Society/European Networking*; 1995; p. 242.
2. Haklay, M. E. Virtual reality and GIS: Applications, trends and directions. *Virtual Real.* **2002**, 47–57.
3. Lin, H.; Chen, M.; Lu, G.; Zhu, Q.; Gong, J.; You, X.; Wen, Y.; Xu, B.; Hu, M. Virtual Geographic Environments (VGEs): A New Generation of Geographic Analysis Tool. *Earth-Science Rev.* **2013**, *126*, 74–84, doi:10.1016/j.earscirev.2013.08.001.
4. Xu, B.; Lin, H.; Chiu, L.; Hu, Y.; Zhu, J.; Hu, M.; Cui, W. Collaborative virtual geographic environments: A case study of air pollution simulation. *Inf. Sci. (Ny).* **2011**, *181*, 2231–2246, doi:10.1016/j.ins.2011.01.017.
5. Mallot, H.; Gillner, S.; Veen, H. Van; Bülthoff, H. Behavioral experiments in spatial cognition using virtual reality. *Spat. Cogn.* **1998**, 447--467, doi:10.1007/3-540-69342-4_21.
6. Bülthoff, H. H.; Campos, J. L.; Meilinger, T. Virtual Reality as a Valuable Research Tool for Investigating Different Aspects of Spatial Cognition (Abstract). *Spat. Cogn.* **2008**, 1–3, doi:10.1007/978-3-540-87601-4_1.
7. Notelaers, S.; De Weyer, T.; Goorts, P.; Maesen, S.; Vanacken, L.; Coninx, K.; Bekaert, P. HeatMeUp: A 3DUI serious game to explore collaborative wayfinding. *IEEE Symp. 3D User Interfaces 2012, 3DUI 2012 - Proc.* **2012**, 177–178, doi:10.1109/3DUI.2012.6184219.
8. Fajen, B. R.; Warren, W. H. Behavioral dynamics of steering, obstable avoidance, and route selection. *J. Exp. Psychol. Hum. Percept. Perform.* **2003**, *29*, 343–362, doi:10.1037/0096-1523.29.2.343.
9. Kretz, T.; Hengst, S.; Roca, V.; Perez Arias, A.; Friedberger, S.; Hanebeck, U. D. Calibrating dynamic pedestrian route choice with an Extended Range Telepresence System. *Proc. IEEE Int. Conf. Comput. Vis.* **2011**, 166–172, doi:10.1109/ICCVW.2011.6130239.
10. Anders Drachen; Thurau, C.; Togelius, J.; Yannakakis, G. N.; Bauckhage, C. Game Data Mining. In *Game Analytics*; 2013; pp. 205–253 ISBN 978-1-4471-4768-8.
11. Guardini, P.; Maninetti, P. Better Game Experience Through Game Metrics: A Rally Videogame Case Study. In *Game Analytics*; 2013; pp. 325–361 ISBN 978-1-4471-4768-8.
12. Medler, B. Visual Game Analytics. In *Game Analytics*; 2013; pp. 403–433 ISBN 978-1-4471-4768-8.
13. Gower Jr., D. W.; Fowlkes, J. E. Simulator Sickness in the UH-60 (Black Hawk) Flight Simulator. **1989**, *60*.
14. Brooks, J. O.; Goodenough, R. R.; Crisler, M. C.; Klein, N. D.; Alley, R. L.; Koon, B. L.; Logan, W. C.; Ogle, J. H.; Tyrrell, R. A.; Wills, R. F. Simulator sickness during driving simulation studies. *Accid. Anal. Prev.* **2010**, *42*, 788–796, doi:10.1016/j.aap.2009.04.013.

15. Cha, M.; Han, S.; Lee, J.; Choi, B. A virtual reality based fire training simulator integrated with fire dynamics data. *Fire Saf. J.* **2012**, *50*, 12–24, doi:10.1016/j.firesaf.2012.01.004.
16. Natapov, A.; Fisher-Gewirtzman, D. Visibility of urban activities and pedestrian routes: An experiment in a virtual environment. *Comput. Environ. Urban Syst.* **2016**, *58*, 60–70, doi:10.1016/j.compenvurbsys.2016.03.007.
17. Kuliga, S. F.; Thrash, T.; Dalton, R. C.; Hölscher, C. Virtual reality as an empirical research tool - Exploring user experience in a real building and a corresponding virtual model. *Comput. Environ. Urban Syst.* **2015**, *54*, 363–375, doi:10.1016/j.compenvurbsys.2015.09.006.
18. Schrom-Feiertag, H.; Schinko, C.; Settgast, V.; Seer, S. Evaluation of guidance systems in public infrastructures using eye tracking in an immersive virtual environment. *Proc. 2nd Int. Work. Eye Track. Spat. Res.* **2014**, *1241*, 62–66, doi:10.1080/13875868.2016.1228654.
19. Zhou, B.; Tang, X.; Zhang, H.; Wang, X. Measuring crowd collectiveness. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 1586–1599, doi:10.1109/TPAMI.2014.2300484.
20. Yi, S.; Li, H.; Wang, X. Understanding pedestrian behaviors from stationary crowd groups. *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* **2015**, *07–12–June*, 3488–3496, doi:10.1109/CVPR.2015.7298971.
21. Solera, F.; Calderara, S.; Cucchiara, R. Socially Constrained Structural Learning for Groups Detection in Crowd. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *38*, 995–1008, doi:10.1109/TPAMI.2015.2470658.
22. Shao, J.; Kang, K.; Loy, C. C.; Wang, X. Deeply learned attributes for crowded scene understanding. *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* **2015**, *07–12–June*, 4657–4666, doi:10.1109/CVPR.2015.7299097.
23. Helbing, D.; Molnár, P. Social force model for pedestrian dynamics 1995, *51*, 5.
24. Li, W.; Gong, J.; Yu, P.; Shen, S.; Li, R.; Duan, Q. Simulation and analysis of congestion risk during escalator transfers using a modified social force model. *Phys. A Stat. Mech. its Appl.* **2014**, doi:10.1016/j.physa.2014.10.044.
25. Treuille, A.; Cooper, S.; Popović, Z. Continuum crowds. *ACM Trans. Graph.* **2006**, *25*, 1160, doi:10.1145/1141911.1142008.
26. Hughes, R. L. A continuum theory for the flow of pedestrians. *Transp. Res. Part B Methodol.* **2002**, *36*, 507–535, doi:10.1016/S0191-2615(01)00015-7.
27. Li, W.; Li, Y.; Yu, P.; Gong, J.; Shen, S.; Huang, L.; Liang, J. Modeling, simulation and analysis of the evacuation process on stairs in a multi-floor classroom building of a primary school. *Phys. A Stat. Mech. its Appl.* **2017**, *469*, 157–172, doi:10.1016/j.physa.2016.11.047.
28. Torrens, P. M. High-resolution space-time processes for agents at the built-human interface of urban earthquakes. *Int. J. Geogr. Inf. Sci.* **2014**, *28*, 964–986, doi:10.1080/13658816.2013.835816.
29. Hoogendoorn, S. P.; Daamen, W. Pedestrian Behavior at Bottlenecks. *Transp. Sci.* **2005**, *39*, 147–159, doi:10.1287/trsc.1040.0102.
30. Helbing, D.; Isobe, M.; Nagatani, T.; Takimoto, K. Lattice gas simulation of experimentally studied evacuation dynamics. *Phys. Rev. E* **2003**, *67*, 67101, doi:10.1103/PhysRevE.67.067101.
31. Wang, C.; Li, L.; Yuan, J.; Zhai, L.; Liu, G. Development of emergency drills system for petrochemical plants based on WebVR. *Procedia Environ. Sci.* **2011**, *10*, 313–318, doi:10.1016/j.proenv.2011.09.051.
32. Rüppel, U.; Schatz, K. Designing a BIM-based serious game for fire safety evacuation simulations. *Adv. Eng. Informatics* **2011**, *25*, 600–611, doi:10.1016/j.aei.2011.08.001.

33. Moussaïd, M.; Kapadia, M.; Thrash, T.; Sumner, R. W.; Gross, M.; Helbing, D.; Hölscher, C. Crowd behaviour during high-stress evacuations in an immersive virtual environment. *J. R. Soc. Interface* **2016**, *13*, 20160414, doi:10.1098/rsif.2016.0414.
34. Cruz-Neira, C.; Sandin, D. J.; DeFanti, T. A.; Kenyon, R. V.; Hart, J. C. The CAVE: audio visual experience automatic virtual environment. *Commun. ACM* **1992**, *35*, 64–72, doi:10.1145/129888.129892.
35. Chen, W. Collaboration in Multi-user Immersive Virtual Environments. *Phd. Diss.* **2016**.
36. Qingshan, Y.; Lin, M. Human-Activity-Geographical-Environment Relationship,Its System and Its Regional System. *Econ. Geogr.* **2001**, *21*.
37. Jianhua, G. On Thought and Methodology of Virtual Geographic Experiment. **2013**.
38. Williams, D. The mapping principle, and a research framework for virual worlds. *Commun. Theory* **2010**, *20*, 451–470, doi:10.1111/j.1468-2885.2010.01371.x.
39. Moussaïd, M.; Perozo, N.; Garnier, S.; Helbing, D.; Theraulaz, G. The walking behaviour of pedestrian social groups and its impact on crowd dynamics. *PLoS One* **2010**, *5*, 1–7, doi:10.1371/journal.pone.0010047.
40. McMahan, R.; Kopper, R.; Bowman, D. Principles for Designing Effective 3D Interaction Techniques. In *Handbook of Virtual Environments*; Human Factors and Ergonomics; CRC Press, 2014; pp. 285–311 ISBN 978-1-4665-1184-2.
41. Wikipedia Dead reckoning --- Wikipedia{,} The Free Encyclopedia 2017.
42. Murphy, C. Believable Dead Reckoning for Networked Games. In *Game Engine Gems 2*; A K Peters/CRC Press, 2011; pp. 307–328 ISBN 978-1-56881-437-7.
43. Wikipedia Kalman filter --- Wikipedia{,} The Free Encyclopedia 2017.
44. Pantel, L.; Wolf, L. C. On the suitability of dead reckoning schemes for games. *Proc. 1st Work. Netw. Syst. Support games - NETGAMES '02* **2002**, 79–84, doi:10.1145/566500.566512.
45. Shi, W.; Corriveau, J. P.; Agar, J. Dead reckoning using play patterns in a simple 2d multiplayer online game. *Int. J. Comput. Games Technol.* **2014**, *2014*, doi:10.1155/2014/138596.