# EVE: A Framework for Experiments in Virtual Environments

Jascha Grübel[1(✉)], Raphael Weibel[1], Mike Hao Jiang[1], Christoph Hölscher[1], Daniel A. Hackman[2], and Victor R. Schinazi[1]

[1] Chair of Cognitive Science, ETH Zürich, Zürich, Switzerland
jascha.gruebel@gess.ethz.ch
[2] USC Suzanne Dworak-Peck School of Social Work, University of Southern California, Los Angeles, CA, USA

**Abstract.** EVE is a framework for the setup, implementation, and evaluation of experiments in virtual reality. The framework aims to reduce repetitive and error-prone steps that occur during experiment-setup while providing data management and evaluation capabilities. EVE aims to assist researchers who do not have specialized training in computer science. The framework is based on the popular platforms of Unity and MiddleVR. Database support, visualization tools, and scripting for R make EVE a comprehensive solution for research using VR. In this article, we illustrate the functions and flexibility of EVE in the context of an ongoing VR experiment called *Neighbourhood Walk*.

## 1 Introduction

EVE (Experiments in Virtual Environments) is a user-friendly, novel framework that facilitates the setup, execution, and evaluation of experiments in virtual reality (VR). In contrast to previous frameworks, EVE is not necessarily a Middleware solution (i.e., allowing interaction with hardware from within the software through an abstract interface). Instead, it provides a comprehensive work environment that enables researchers to design different stages of an experiment. EVE is specifically designed for researchers without specialized training in computer science. A friendly graphical user interface (GUI) interface, user manual, and video tutorials are provided to help users with different levels of expertise. One strength of the framework lies in its ability to define, collect, and analyse different types of behavioural (e.g., eye-tracking, avatar control) and physiological (e.g., electrodermal, electrocardiography) data. The framework was originally designed to support spatial cognition and navigation studies but can be easily adapted to any research that uses VR. EVE operates under the platform of the free versions of the game engine and editor Unity [58] and the Middleware framework MiddleVR [35]. The framework capitalizes on Unity's use of the C#-programming language and adds database support, statistical evaluation with R-scripts [40], and Unity-based objects/assets to support different aspects of an experiment.


T. Barkowsky et al. (Eds.): KogWis/Spatial Cognition 2016, LNAI 10523, pp. 159–176, 2017.
https://doi.org/10.1007/978-3-319-68189-4_10

There are several reasons why running experiments in VR can benefit research in experimental psychology and other disciplines. Gaggioli [13] outlines at least four motivations. First, VR can provide for better ecological validity allowing for naturalistic behaviour within a simulated environment. Second, VR systems are flexible and provide programmers the ability to customize the environment and stimuli to the needs of researchers. Third, new VR technology can provide accurate sensorial feedback that can be difficult to capture and manipulate using traditional methods (i.e., videos). Fourth, VR systems can facilitate the accuracy and reliability of data collection, considerably speeding up the entire experimental process. If researchers are capable of accounting for vection (i.e., the perception of self-motion in the absence of physical motion) and presence, VR systems may be capable of achieving a high level of experimental control and ecological validity [26,45].

Despite these advantages, some researchers have argued that VR may not be capable of sufficiently emulating real-world behaviour. For example, Taube and colleagues [56] emphasize that actual locomotion is necessary to realistically capture different aspects of spatial orientation during navigation. While this may be the case for some experiments that require subjects to remain immobile and in a supine position (i.e., fMRI), there is now growing evidence that a correspondence may indeed exist in the navigation behaviour and aesthetic experiences of participants in the real world and VR with limited locomotion [20,23,64]. For example, Weisberg and colleagues [64] tested participants in a virtual replica of a university campus and found a similar pattern of spatial responses (i.e., pointing judgments and model building) as observed in the same real-world environment [48]. Indeed, researchers from various fields have successfully employed VR in their experiments [21,28,61,64].

Previous studies that have used VR often required the programming of custom-made tools specifically designed for the experiment in question [42–44,46,52,63]. Unfortunately, many of these tools were not sufficiently modular (e.g., custom scripts, specific hardware) to be adapted for other experiments and the research community in general. As a direct consequence, many researchers have not been able to produce similar experiments when only minor changes are needed. This can lead to delays and higher costs when preparing an experiment (e.g., the psychologist relies on the computer scientist for addressing every minor adaptation).

Another challenge with previous research occurs when participant data is collected outside the virtual environment (VE) via video or experimenter notes. Manual coding of data can lead to errors in accuracy and also limit the type and richness of data that can be collected, precluding certain research questions and analyses [25,34]. For example, physiological data (e.g., electrodermal activity), often sampled at high frequencies, generates numerous data points that must be properly synchronized to the stimulus presented in the VE. Similarly, navigation studies often rely on the exact position and movement of the avatar for statistical analysis. Here, automatic coding of the data is relatively precise and potentially more reliable than manual coding. Even with automatic coding, some frameworks

only allow researchers to store data in text files [2]. This can lead to additional delays and challenges with data organization and analyses.

Some studies have used Middleware frameworks [4,53] to collect data from different types of peripherals. Middleware allows for data retrieval without relying on specific hardware protocols. This is a critical step for collecting information about the participants' interaction with the VE. Middleware frameworks are capable of extracting inputs from sensors (e.g., eye tracker) into a simpler data stream for analyses. Despite the importance of Middleware for managing hardware, it is only one component in the design of a comprehensive experimental framework.

In this paper, we propose a general-purpose framework that allows for the simplification of the various steps involved in the creation, execution, and analysis of experiments in VR. EVE is more than just a Middleware solution. In contrast to previous frameworks (e.g., SNaP [2], CalVR [49]), EVE provides comprehensive support from the early design phases of an experiment through the generation, storage, organization, and analysis of the collected data. EVE is capable of acquiring data from a variety of physical (e.g., eye trackers, physiological devices) and virtual sensors (e.g., position trackers placed directly into the virtual environment) and storing this information into a neatly organized and accessible database for evaluation. We illustrate the functionality of EVE in the context of a complex, ongoing experiment called *Neighbourhood Walk*.

## 2 Related Work

Existing frameworks for VR tend to be limited to demonstration or visualisation purposes rather than the collection and manipulation of collected data (e.g., behavioural, physiological). Commercial VR frameworks such as EON Studio [10] or Vizard [67] offer development suites for VEs that focus on visualization and game play. Similar to Middleware frameworks such as CAVELib [31], VR Juggler[1] [3], FreeVR [51], and MiddleVR [35], these packages do not currently offer specific support to setup and run scientific experiments in VR.

The scientific community has also attempted to create VR frameworks that offer similar capacities as commercial and Middleware frameworks. Vrui [22] and CalVR [49] are C++-based frameworks offering platforms to develop a variety of visualisation applications. Vrui is often used for visualising volumetric data [4,24] while CalVR is an application typically used to visualise archaeological sites without having to physically visit them [53,60]. However, similar to their commercial counterparts, these packages do not offer sufficient options for the setup and execution of scientific experiments.

There have been some attempts at creating frameworks that are specifically oriented towards the setup and execution of experiments. For example, the SNaP framework [2] provides a limited set of features for collecting metrics focused on the position and orientation of the avatar with respect to the world. However,

[1] The project appears to be abandoned.

the environments used in this framework have to be designed with external 3D tools and could not be easily be adjusted/altered within the framework. When editing a specific asset (e.g., a building) in the virtual model, SNaP users are required to load an external application and reload the entire model after each adjustment. This two-step procedure prolongs the development process and makes efficient creation and deployment of experiments difficult. In addition, Virtools [9], the underlying software supporting SNaP, limits the ability to run experiments on arbitrary hardware.

CAVEStudy [14], the precursor to EVE, introduced a general structural layout for experiments and tried to standardize the experiment description in order to guide and facilitate experiment design and execution. CAVEStudy included a large set of physical and virtual sensors, database support, and predefined experimental elements used to develop tasks. However, the development of this framework was halted due to limitations with Vizard-based VR environments. In addition to editing challenges (described above for SNap), Vizard relies on dynamic scripting in Python. This type of dynamical scripting is useful for quick tests but becomes more error-prone when creating a large codebase. These issues are more easily addressed in a statically typed language such as C# used by Unity.

Virtual SILCton [8] is an example of a framework that was specifically developed to investigate individual differences in navigation [64]. A typical experiment in Virtual SILCton consists of learning a novel VE (i.e., a University campus) and completing a series of questionnaires (e.g., mental rotation, sense of direction) and spatial tasks (e.g., distance and direction judgements, model building). Virtual SILCton provides researchers with some flexibility when setting up their experiment. Researchers can select and order (from a preprogrammed set) a series of questionnaires and spatial tasks and define how participants learn the VE (i.e., free exploration or guided navigation). In addition, Virtual SILCton offers a suite of evaluation tools to export the data collected during the experiment. While Virtual SILCton provides a major step forward in the design of VR experiments, it is not sufficiently modular (e.g., all experiments take place in the same virtual world, researchers are limited to the preprogrammed questionnaires) to be adapted or enhanced for other experiments and research paradigms.

Some researchers have relied on modified games [25] at the cost of losing control over some features of the experiment (e.g., the ability to export the exact location of the avatar). Most games also do not provide any information regarding the geometry of the environment, making the analysis of navigation data difficult if not impossible. Although *modding* (i.e., the act of modifying a software to perform a function that was not originally intended by the developer) can overcome some of these difficulties, gameplay is still limited by the main game mechanics (e.g., a driving game cannot be used for pedestrian exploration). In addition, *modding* still falls within a grey zone of software development as it often involves hacking proprietary source code to enable obscure features. However, when properly implemented, some researchers have been capable of using *modded* games for experimental research. This is the case for the studies conducted by

Maguire and colleagues [27], [54] who adapted the video game *The Getaway* in order to conduct neuroscientific research with London taxi drivers.

Lloyd and colleagues [25] used the Sony Playstation 2 game *Driv3r* (Reflections Interactive Ltd.) to simulate a virtual tour of the city of Nice, France. Participants were asked to guide an experimenter who was driving the virtual car by verbally indicating turns along the route. One consequence of such an approach was that performance could only be encoded in the form of turning decisions. Path data (often critical for navigation studies) could not be exported from the gaming console. Other experimenters adapted *Half-Life* in order to investigate spatial integration (i.e., cognitive mapping) in humans [55]. Using the *Valve Hammer Editor*, these researchers were able to design and test participants within the *Half-Life* virtual environment. However, the researchers still relied on video recording to capture participant spatial behaviour during gameplay. Together, these results suggest that using commercial games for research often limits the type of interaction users can have with the VE and the accuracy and type of data that can be exported from the system.

Some researchers [32,33] also try to adapt previous experimental setups to fit the current experimental design [59]. For example, Usoh and colleagues [59] created a two-room environment in order to test immersion in VR. Participants moved from a training room (e.g., a simply furnished room) to a "pit" room. In the pit room, they were instructed to step on a ledge and look through a hole in the floor with a view to the room below. In the original experiment, participants reported their experience with questionnaires, and later variations used physiological sensors as a measure of presence [32]. While adapting previous experiments can have advantages in terms of development costs, these are only expandable to the extent that they fit the research paradigm.

When collecting data in VR, some researchers ask participants to answer questionnaires or complete different tests (in pen and paper format) while the participants are engaged in the VR task [34]. This technique can have important implications for immersion given that participants are forced to switch from the VR to interact with the experimenter. In addition to increasing post processing time, this method is also susceptible to human error during digitalization. Consequently, a framework is needed that allows for self-report questionnaires to be deployed in real-time in a VR context and to be sufficiently accurate so as to minimize any breaks in the sense of immersion.

Altogether, many previous VR studies and frameworks were developed in a task-specific manner and could rarely provide all the necessary elements and be generalized to support new experiments. In contrast, EVE was developed to support multiple experiments by providing features that are common to research in VR and could be adapted to different experimental paradigms from a variety of disciplines. EVE allows for easy handling of common tasks in experiments, promoting greater efficiency and quality of VR experiments. In turn, this enables researchers to focus less on the technical implementation of a study and more on central issues. Among these are experimental design and the creation of immersive VEs that are capable of simulating real-world aspects critical to the research question at hand.

## 3 Implementation

We will discuss the implementation of EVE in two parts. First, we focus on external software requirements. Second, we analyse how EVE supports scientists during three stages (i.e., setup, runtime, and evaluation) of a typical VR experiment. These stages are outlined in Fig. 1 and discussed in the following subsections.

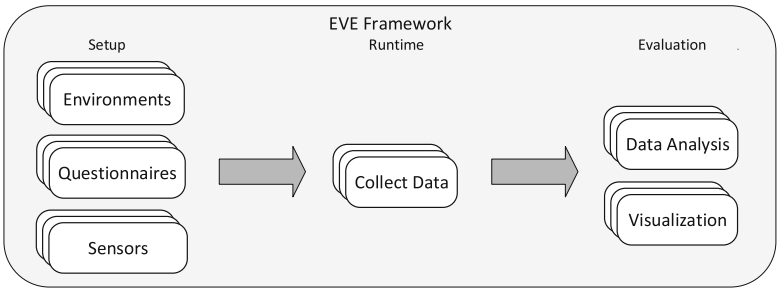


**Fig. 1.** Overview of the three stages of an experiment that are supported by the EVE framework. In the setup stage, experimenters can upload and manipulate their VE, load XML-based questionnaires, and place their virtual sensors. EVE generates executable files that can be used during runtime to start the experiment and data collection. In the evaluation stage, experimenters can perform different types of analyses and visualizations and export the collected data in different formats.

### 3.1 Required Software

The EVE framework is based on the Unity game engine and requires a working version of Unity 5 or above. Unity is a widely used game development engine that handles the graphics and physics computations necessary for video game applications. Users should follow the minimum system requirements[2] for developing and running games in Unity. Unity offers a series of step-based tutorials to help users learn the different aspects of the game engine.

EVE integrates with Unity via the C#-programming platform. Data acquisition, management, and evaluation are all provided by a database operating at the backend of the framework. As such, a MySQL database is necessary in order to run the basic version of the framework—although EVE can be easily adapted for other databases (e.g., PostgreSQL). For advanced evaluation purposes, a recent version of the statistical programming language R (e.g. version 3.3.1) along with several R-packages must be installed. The packages DBI [41], RMySQL [39], and dplyr [66] manage the access to the database. Our new package evertools [6] provides additional shorthand functions for accessing specific database entries (e.g., participants' paths) as well as some pre-arranged statistical evaluations. Additionally, evertools uses ggplot2 [65] to visualize the collected

[2] https://unity3d.com/unity/system-requirements.

data. EVE also supports Virtual Reality Peripheral Network (VRPN [57]) via *MiddleVR for Unity*. VRPN is a widely used VR standard to define hardware representation in software and access sensor data. MiddleVR must be installed in order to access these non-critical features. MiddleVR allows for data to be collected from any VRPN-enabled sensors via XML configuration files.

### 3.2 Experiment Setup

This first stage consists of setting up the static VE, creating the interactive parts of the environment, and linking up the sensors necessary to run the experiment. Static environments are the 3D models (e.g., city blocks, building interiors, mazes) that participants interact with (e.g., navigate) during the experiment. Unity provides an editor that allows for the importing and basic manipulation (e.g., moving, rotating, and scaling) of standard 3D model formats (e.g., fbx, dae). While only a limited number of ready-to-use environments are included with EVE (e.g., training scene), the framework provides a variety of tools to setup the interactive parts of a VE. EVE provides a large set of useful experiment-oriented assets (also known as *prefabs* in Unity). In EVE, prefabs can act as virtual sensors that are placed directly into the VE and allow for a variety of data collection options. Generally, virtual sensors allow researchers to define all points of interest in the VE and ensure proper data collection as the participants interact with them.

Interactions can also be attached to virtual sensors further extending their capacity. This includes, but is not limited to, (a) a logging object that records all the movements of the participant's avatar in the VE; (b) hidden markers that record when participants enter or exit a specific zone; (c) invisible walls that keep participants on the designated route and can display guiding messages; (d) visible way-points, goal markers, and collectible items that can guide a subject through the environment; (e) start and end points that can be selected at random and (f) trigger level changes (i.e., completing a part of an experiment and moving on to the next part).

EVE also provides a variety of diegetic and non-diegetic user interface (UI) elements [11] for displaying different types of information to participants. Diegetic UIs allow participants to remain immersed in the VE by embedding the UI in the virtual world. In contrast, non-diegetic UI elements are typically overlaid on top of the participant's view during gameplay. For example, a navigation task can include UI elements such as a mini-map of the environment, a list of goals, a timer, and text pop-ups in the form of floating text (see Fig. 2).

EVE supports the use of questionnaires during all stages of the experiment and can be used in conjunction with the different physical sensors. The basic version of the framework already includes some of the commonly used questionnaires in navigation research (e.g., SBSOD [18]). A novel feature of EVE is the provision of pop-up questionnaires during runtime. These questionnaires can be answered while participants are engaged in the task without diverting their attention. For example, participants can be probed about their current level of arousal via a digital pop-up version of the Self-Assessment Manikin (SAM) [5]


(a) *The mini-map and goal list* (b) *A highlighted destination*

**Fig. 2.** EVE's UI elements. (a) Non-diegetic UI: A mini map with a goal list (yellow dots) and the participant's current location (blue dot); (b) Diegetic UI: A text pop-up indicating a destination in a wayfinding task. (Color figure online)

that appears immediately after exposure to a particular stimulus (e.g., a building). Additionally, researchers can use their own questionnaires by providing XML-files tailor fitted to their needs.

The EVE framework gives researchers control over the execution order of the experiment. After the VEs and the questionnaires are prepared, researchers can complement their setup with Unity scenes that are already built into our framework (see Table 1). Here, researchers have to decide the order of events and place each scene into a linear execution order in the Unity editor. The last step consists of selecting and activating the virtual and physical sensors that will be used for data recording. EVE uses the XML-based sensor description in VRPN to describe sensors and enable the framework to properly read and store the associated values.

In addition to VRPN-based sensors, the framework also provides an abstract interface for eye-tracking (currently only deployed with SensoMotoric Instruments [50]), physiological data collection (currently deployed with PowerLab by ADInstruments [1]) and support for HL7 messages [17] used by various medical sensors. Support for additional behavioural and physiological sensors are expected in future versions of the framework.

**Table 1.** List of available scenes in the basic version of EVE.

| Scene | Description |
|---|---|
| Maze training | A maze environment to train the participants with the navigation controls (e.g., joystick) |
| Fixation screen | A white screen with a black cross in the middle used to centre gaze and to separate between sections in the experiment |
| Tutorial | A small room with a video wall explaining basic controls (e.g. how to use a controller) |
| Video screen | A simple playback screen to show videos during an experiment |

### 3.3 Runtime

After setting up the experiment in the Unity editor, an executable build can be created to run the experiment independent of the editor. The build contains the experiment as an executable file that can be started as a normal application on the respective platform. Unity can generate builds for all platforms allowing for experiments to be conducted on Mac, Windows, Linux, and mobile operating systems such as iOS and Android. Depending on the platform, it may be necessary to adapt the database capabilities of EVE to enable proper data collection. Currently, only the Windows database adapter is implemented. In addition, when VRPN or more complex VR systems like a CAVE [7] are used, MiddleVR must be configured and used to run the executable file. Head-mounted devices such as VIVE [19] and Oculus [37] can also be configured for use within either Unity or MiddleVR.

During runtime, EVE's logging manager records all of the data created during the experiment into a MySQL database (Fig. 3). This data is collected until the end of the experiment and can be retrieved with the R-package *evertools*. The evaluation features are part of the runtime build and will be discussed in the next section.

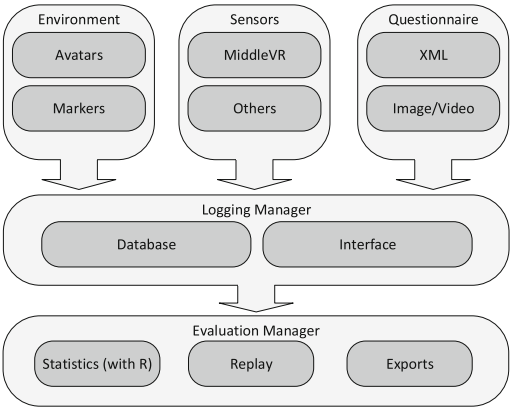


**Fig. 3.** Overview of the dataflow within the EVE framework. Physical and virtual sensors and questionnaires provide data input. The logging manager stores them in the database and offers an interface to retrieve the data. During evaluation, the data is preprocessed for easier use.

As expected, the framework cannot entirely replace an experimenter during data collection. While EVE will greatly speed up experimental runtime, experimenters still have a central role in overseeing study execution and answering any questions that participants may have.

### 3.4 Evaluation

The evaluation menu provides access to the underlying database and simplifies the management and visualization of the collected data. EVE's database uses a data layout in the 5$^{th}$ normal form [12]. While this allows for better handling of arbitrary sensor and questionnaire data (from the perspective of a framework), this form makes working with the database a more difficult task for experimenters without knowledge of database management. To overcome this hurdle, the evaluation menu offers a GUI functionality to visually interact with the data collected in the experiment. Instead of going through a series of database queries, experimenters can also use our R-package *evertools* to directly interact with the data (i.e., select, display, analyse, and export).

This functionality can be grouped into three main modules. The first module allows researchers to export data collected during the experiment. Currently, EVE can only export in CSV format. The next versions of the framework will include different export formats (e.g., MatLab, SPSS). The second module covers the visualization of data. A replay mode allows for a detailed inspection of the participants' spatial behaviour (e.g., walked path) from either a first-person or bird's-eye perspective. Researchers may inspect the data for validity or run additional post-processing algorithms including image processing and segmentation (e.g., spatial frequency, contrast). Some of these operations typically have high computational costs that can hinder the fidelity of the experiment if executed during runtime. For this reason, the evaluation build performs these computations in replay mode. Additional sensor data such as gaze patterns can also be overlaid on the replay screen.

The third module integrates the statistical programming language R into the EVE framework. The R-package *dplyr* is used to access the database. An auxiliary R-package *evertools* provides the researcher with direct access to sensor data belonging to a specific participant (bypassing the underlying database structure). The R-package *evertools* currently provides two built-in evaluations. First, researchers can extract the path length and the deviation of a single participant from the cohort. Second, researchers can extract different descriptive statistics regarding the duration of the various experimental phases. These tools can be visualised with *ggplot2* and used to discriminate between participants (e.g., outliers). Researchers may use *evertools* to access the data in their own R-scripts to further customize their analysis.

## 4 Example Study: Neighbourhood Walk

In this section we present the different features of EVE in the context of the experiment *Neighbourhood Walk* currently being developed as a collaboration among the authors. The experiment investigates whether virtual environments are capable of eliciting different types of behavioural and psychophysiological responses. Previous research has shown that different aspects of neighbourhood environments are related to varying levels of stress reactivity [15,16]. In order to

investigate these questions, we designed a virtual city with distinct neighbourhoods that are differentiated by building types, layout, green space, disorder, and noise among other features that are routinely observed in real world environments [38,47]. In the experiment, participants were asked to follow a route around the virtual city and collect a series of gems that are placed along the route. Participants also responded to a series of questionnaires about their background and the quality of the environment. Physiological data (i.e., galvanic skin response, heart rate variability and blood pressure) was collected throughout the experiment.

An efficient way to construct an experiment with EVE is to develop a protocol that defines the different stages of the experiment and to use it as a guide when setting up the work in Unity. Figure 4 is a sample protocol from the *Neighbourhood Walk* study. A protocol should contain all the steps of an experiment including those steps that are not directly performed by EVE. This allows researchers to develop a clear timeline and decide their level of interaction. For the *Neighbourhood Walk* study, the protocol contained the steps to be executed by the researcher (SR) and those performed by EVE (SE). The protocol also contained information regarding the placement of alignment windows (W). Alignment windows are Unity scenes provided by EVE that allow for the proper synchronization between the data collected by the physical sensors and the different stages of the virtual experiment (i.e., training, baseline, and main task).

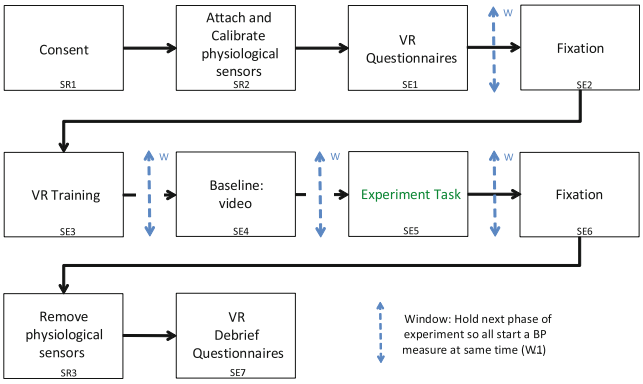


**Fig. 4.** The protocol describing each execution step of the *Neighbourhood Walk* study. The protocol contains steps to be executed by the experimenter (SR) and by EVE (SE). The protocol also indicates when windows (W) are used to synchronize between the various stages in the experiment.

In a typical experiment session for *Neighbourhood Walk*, the participant arrived at the lab, completed the consent process (SR1) and was connected to the different physiological sensors (SR2). At this stage, the experimenter started the Unity build, and the participant was presented with a digital questionnaire

(e.g., background information; SE1; Fig. 5(a)). A fixation screen (SE2) was then used to acquire physiological data before the start of the experiment. Once the fixation screen disappeared, the Unity scene for joystick training was loaded (SE3). Here, participants watched an instruction film on how to manipulate the joystick and later trained their skills on a simple maze environment. This training insured that all participants were sufficiently skilled at navigating the VE before the start of the experimental task. The next stage consisted of acquiring a physiological baseline (SE4). Participants were asked to sit still while watching a nature video. Once the baseline video was complete, the main task was loaded (SE5). For the main tasks, participants were asked to use the joystick to follow a route and collect a variety of gems placed along a virtual city neighbourhood (Fig. 5(b)). After the main task, participants move to a post-task fixation screen (SE6). Note that a Window (W) was used at different stages of the experiment in order to ensure that the physical sensors were synchronized to the stimulus presented on the screen. At the end of the last fixation screen, an information screen was loaded indicating that it was now safe for the experimenter to remove the physiological sensors (SR3). The last scene (SE7) consisted of a series of debriefing questionnaires about the VE.


(a) *An example question used during the* Neighborhood Walk *experiment.*


(b) *First-person perspective of a city block. The gem represents a typical virtual sensor with an interaction.*

**Fig. 5.** Questionnaire and interactions used in *Neighborhood Walk* study

In the case of the *Neighbourhood Walk*, we created a large virtual city (Fig. 6) in cooperation with external partners from VIS Games [62] and Zatun [68]. VIS Games provided us with the basic city environment (e.g., buildings, streets) while Zatun, in consultation with the authors, added the different assets (e.g., trees, flowers, garbage) that gave each of the city blocks their particular character (e.g., industrial, luxury homes). We also added a variety of virtual sensors and interactions to the different scenes. These virtual sensors allowed us to control different aspects of the experiment (e.g., scene changes, calibration windows) and to collect data about participant behaviour in the environment (e.g., gems collected). Some virtual sensors were also used as checkpoints when participants

**Fig. 6.** A top-down view of the city used in the *Neighbourhood Walk* study


**Fig. 7.** The evaluation menu including the visualization of a participant's path on a map of the city. The menu also provides basic information about the path and the option to export the data collected by the different sensors.

were navigating the virtual neighbourhood. These sensors encoded the time and location when the participant entered a specific area or street and allowed for a more fine-grained analysis of their spatial behaviour and psychophysiological state.

Data collected in the experiment can be accessed via the evaluation build (Fig. 7). In the case of the *Neighbourhood Walk* study, the experimenter can view and export a variety of data captured by the physical and virtual sensors. The evaluation menu also provides access to a top-down view of the city environment for quick visual inspection of the path walked by the participants. In addition, a first person replay function allows for different post processing analyses (e.g., image analysis) that could not otherwise be performed during runtime. Finally, we use the integrated R functionality of EVE to obtain statistical analysis of the path length and time spent in each section of the city.

## 5 Summary and Conclusion

In this paper, we describe a new framework to run experiments in VEs. The EVE framework provides a novel way to assist researchers from various disciplines with the setup, execution, and evaluation of experiments in VEs. EVE uses various functionalities offered by the Unity 3D editor to setup the different stages of an experiment. This allows researchers to rely on a powerful 3D-editing environment and game engine to setup and run their experiments. EVE also expands some basic functions in Unity by providing additional research oriented solutions including the collection, storage, visualisation, analysis, and export of experiment data. The framework is specifically designed to process sensor data from peripheral or interactive objects that are placed directly in the VE. EVE is also supported by a database working behind the scenes to organize and store the data collected during the experiment. A key feature of the framework is the evaluation menu that allows researchers to inspect, analyse (via custom made R scripts) and export participant data. The first release of EVE will allow users to export descriptive statistics. Future releases will include data mining functionality and inferential statistics. As such, EVE provides a series of modules that help researchers design and execute an experiment from start to finish.

Our example study, *Neighbourhood Walk*, illustrated some of the features and advantages of designing an experiment with EVE. In the study, the framework was used to support the collection and organization of data from a variety of questionnaires and both physical (e.g., blood pressure) and virtual sensors (e.g., collection of gems, location markers). In addition, EVE provided the necessary functionality to synchronize events occurring in the VE to these sensors. EVE also simplified any type of database management by providing an evaluation menu for the manipulation and exporting of the collected data.

A variety of developments are already planned for future releases of EVE. One such expansion consists of a simulation package for agents (based on avatars) that may be used to improve presence by creating more realistic VEs. These agents will also expand the functionality of EVE to include crowd simulation. An expansion is also under construction that will allow for testing multiple human participants in a single VE via networked computers (both locally and via the Internet). Additional expansions will also enable researchers to import and analyse data from real-world experiments. For example, using a 3D model of a real-world environment, researchers can reconstruct walked paths with the help of computer vision techniques applied to videos recorded from the participants' perspective. This would allow for the systematic analysis and reconstruction of behavioural data acquired in the real-world. Finally, future developments will be aimed at improving the experiment design process by using Finite State Machines (FSM). FSMs are a class of automata that represent sequential processes [29,30,36]. They can describe the experiment protocol in terms of states and transitions and can represent an experiment in graph form that can be visually understood by the experimenter but executed by the machine. FSMs provide a precise description of an experimental design and can reduce ambiguity inherent in text-based descriptions. One clear advantage of such an

approach is the ability to implement non-linear protocols during an experiment (e.g., different experiment environments being called in an arbitrary order).

EVE will be provided free of charge to the scientific community. It will be published as an open source project via the code-sharing platform github (https://cog-ethz.github.io/EVE/). The Chair of Cognitive Science at ETH Zürich will maintain and continue to develop the EVE framework. We also encourage researchers to help us improve and extend the framework to meet as many application demands as possible.

**Acknowledgements.** The authors would like to thank Ioannis Giannopoulos and Tyler Thrash for the valuable comments and suggestions during various drafts of this manuscript. We would also like to thank Katja Wolf and Fabian Schewetofski for the development and design of the evaluation screen, GUI interface, and many other features that have now become an integral part of the EVE framework. We also thank VIS Games for the free provision of the 3D models used in our research. Partial support for Daniel Hackman was provided by the *Robert Wood Johnson Foundation Health and Society Scholars Program* at the *University of Wisconsin-Madison* in the *Department of Population Health Sciences*.

## References

1. ADInstruments: Labchart (2016). http://www.adinstruments.com/products/labchart
2. Annett, M., Bischof, W.F.: VR for everybody: the SNaP framework. In: SEARIS Workshop in IEEE Virtual Reality, pp. 131–132 (2016)
3. Bierbaum, A., Just, C., Hartling, P., Meinert, K., Baker, A., Cruz-Neira, C.: VR juggler: a virtual platform for virtual reality application development. In: 2001 Proceedings of Virtual Reality, pp. 89–96. IEEE (2016)
4. Billen, M.I., Kreylos, O., Hamann, B., Jadamec, M.A., Kellogg, L.H., Staadt, O., Sumner, D.Y.: A geoscience perspective on immersive 3D gridded data visualization. Comput. Geosci. **34**(9), 1056–1072 (2016)
5. Bradley, M.M., Lang, P.J.: Measuring emotion: the self-assessment manikin and the semantic differential. J. Behav. Ther. Exp. Psychiatry **25**(I), 49–59 (2016)
6. Chair of Cognitive Science, ETH: EVE: A framework for experiments in virtual environments (2016). https://cog-ethz.github.io/EVE/
7. Cruz-Neira, C., Sandin, D.J., DeFanti, T.A.: Surround-screen projection-based virtual reality. In: Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques, pp. 135–142 (2016)
8. Dara-Abrams, D., Schinazi, V.R.: Virtual SILCton (2016). http://spactial.ci.northwestern.edu/
9. Dassault Systemes: Virtools (2016). http://www.3dvia.com/products/3dvia-virtools/
10. Eon Reality: Eon studio (2016). http://www.eonreality.com/eon-studio/
11. Fagerholt, E., Lorentzon, M.: Beyond the HUD. User interfaces for increased player immersion in FPS games. Ph.D. thesis, Chalmers University of Technology (2016)
12. Fagin, R.: Normal forms and relational database operators. In: Proceedings of the 1979 ACM SIGMOD International Conference on Management of Data, pp. 153–160. ACM (1979)

13. Gaggioli, A.: Using Virtual Reality in Experimental Psychology, vol. 2. IOS Press, Amsterdam (2016)
14. Grübel, J.: Assessing human interface device interaction in virtual environments. Bachelor thesis. ETH Zürich (2016). http://dx.doi.org/10.3929/ethz-a-010544699
15. Hackman, D.A., Betancourt, L.M., Brodsky, N.L., Hurt, H., Farah, M.J.: Neighborhood disadvantage and adolescent stress reactivity. Front. Hum. Neurosci. **6**, 277 (2012)
16. Hartig, T., Mitchell, R., De Vries, S., Frumkin, H.: Nature and health. Ann. Rev. Public Health **35**, 207–228 (2014)
17. Health Level Seven: HL7 Message Standard (2016). http://www.hl7.org/implement/standards/product_brief.cfm?product_id=146
18. Hegarty, M., Richardson, A.E., Montello, D.R., Lovelace, K., Subbiah, I.: Development of a self-report measure of environmental spatial ability. Intelligence **30**(5), 425–447 (2016)
19. HTC: HTC VIVE (2016). https://www.vive.com/
20. Kort, Y.A.W., Ijsselsteijn, W.A., Kooijman, J., Schuurmans, Y.: Virtual laboratories: comparability of real and virtual environments for environmental psychology. Presence Teleoperators Virtual Environ. **12**(4), 360–373 (2016)
21. Kraemer, D.J.M., Schinazi, V.R., Cawkwell, P.B., Tekriwal, A., Epstein, R.A., Thompson-Schill, S.L.: Verbalizing, visualizing, and navigating: the effect of strategies on encoding a large-scale virtual environment. J. Exp. Psychol. Learn. Mem. Cogn. **43**, 611–621 (2016). http://dx.doi.org/10.1037/xlm0000314
22. Kreylos, O.: Environment-independent VR development. In: Bebis, G., et al. (eds.) ISVC 2008. LNCS, vol. 5358, pp. 901–912. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-89639-5_86
23. 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. **54**, 363–375 (2016)
24. Laha, B., Sensharma, K., Schiffbauer, J.D., Bowman, D.A.: Effects of immersion on visual analysis of volume data. IEEE Trans. Vis. Comput. Graph. **18**(4), 597–606 (2016)
25. Lloyd, J., Persaud, N.V., Powell, T.E.: Equivalence of real-world and virtual-reality route learning: a pilot study. Cyberpsychol. Behav. **12**(4), 423–427 (2016)
26. Loomis, J.M., Blascovich, J.J.: Immersive virtual environment technology as a basic research tool in psychology. Behav. Res. Methods Instrum. Comput. **31**(4), 557–564 (2016)
27. Maguire, E.A., Nannery, R., Spiers, H.J.: Navigation around London by a taxi driver with bilateral hippocampal lesions. Brain **129**(11), 2894–2907 (2006)
28. Marchette, S.A., Vass, L.K., Ryan, J., Epstein, R.A.: Anchoring the neural compass: coding of local spatial reference frames in human medial parietal lobe. Nature Neurosci. **17**(11), 1598–1606 (2016)
29. McCulloch, W.S., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. Bull. Math. Biophys. **5**(4), 115–133 (2016)
30. Mealy, G.H.: A method for synthesizing sequential circuits. Bell Syst. Tech. J. **34**(5), 1045–1079 (2016)
31. Mechdyne: CAVELib (2016). http://www.mechdyne.com/software.aspx
32. Meehan, M., Brooks, F.P.: Physiological measures of presence in stressful virtual environments. ACM Trans. Graph. (ToG) **21**, 645–652 (2016)
33. Meehan, M., Razzaque, S., Insko, B., Whitton Jr., M., Brooks, F.P.: Review of four studies on the use of physiological reaction as a measure of presence in stressful virtual environments. Appl. Psychophysiol. Biofeedback **30**(3), 239–258 (2016)

34. Meijer, F., Geudeke, B.L.: Navigating through virtual environments: visual realism improves spatial cognition. CyberPsychol. Behav. **12**(5), 517–521 (2016)
35. MiddleVR: MiddleVR for unity (2016). http://www.middlevr.com/
36. Moore, E.F.: Gedanken-experiments on sequential machines. Automata Studies **34**, 129–153 (2016)
37. Oculus VR LLC: Oculus rift (2016). https://www.oculus.com/
38. Odgers, C.L., Caspi, A., Bates, C.J., Sampson, R.J., Moffitt, T.E.: Systematic social observation of children's neighborhoods using Google Street View: a reliable and cost-effective method. J. Child Psychol. Psychiatry **53**(10), 1009–1017 (2012)
39. Ooms, J., James, D., DebRoy, S., Wickham, H., Horner, J.: RMySQL: database interface and 'MySQL' driver for R (2016). https://cran.r-project.org/package=RMySQL
40. R Core Team: R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria (2016). https://www.r-project.org/
41. R Special Interest Group on Databases (R-SIG-DB), Wickham, H., Müller, K.: DBI: R database interface (2016). https://cran.r-project.org/package=DBI
42. Razzaque, S., Kohn, Z., Whitton, M.C.: Redirected walking. In: Proceedings of EUROGRAPHICS. vol. 9, pp. 105–106 (2016)
43. Razzaque, S., Swapp, D., Slater, M., Whitton, M.C., Steed, A.: Redirected walking in place. In: ACM International Conference Proceeding Series, vol. 23, pp. 123–130 (2016)
44. Riecke, B.E., Bodenheimer, B., McNamara, T.P., Williams, B., Peng, P., Feuereissen, D.: Do we need to walk for effective virtual reality navigation? Physical rotations alone may suffice. In: Hölscher, C., Shipley, T.F., Olivetti Belardinelli, M., Bateman, J.A., Newcombe, N.S. (eds.) Spatial Cognition 2010. LNCS (LNAI), vol. 6222, pp. 234–247. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14749-4_21
45. Riecke, B.E., Schulte-Pelkum, J.: An integrative approach to presence and self-motion perception research. In: Lombard, M., Biocca, F., Freeman, J., Ijsselsteijn, W., Schaevitz, R.J. (eds.) Immersed in Media, pp. 187–235. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-10190-3_9
46. Ruddle, R.A., Lessels, S.: For efficient navigational rich visual scene search, humans require full physical movement, but not a rich visual scene. Psychol. Sci. **17**(6), 460–465 (2016)
47. Sampson, R.J., Raudenbush, S.W.: Systematic social observation of public spaces: a new look at disorder in urban Neighborhoods 1. Am. J. Sociol. **105**(3), 603–651 (1999)
48. Schinazi, V.R., Nardi, D., Newcombe, N.S., Shipley, T.F., Epstein, R.A.: Hippocampal size predicts rapid learning of a cognitive map in humans. Hippocampus **23**(6), 515–528 (2016)
49. Schulze, J.P., Prudhomme, A., Weber, P., DeFanti, T.A.: CalVR: an advanced open source virtual reality software framework. In: IS&T/SPIE Electronic Imaging, vol. 8649, pp. 864902–864908 (2016). http://dx.doi.org/10.1117/12.2005241
50. SensoMotoric Instruments: SMI Eye-Tracking (2016). http://www.smivision.com/en/gaze-and-eye-tracking-systems/home.html
51. Sherman, W.R.: FreeVR (2016). http://www.freevr.org/
52. Slater, M., Khanna, P., Mortensen, J., Yu, I.: Visual realism enhances realistic response in an immersive virtual environment. IEEE Comput. Graph. Appl. **29**(3), 76–84 (2016)

53. Smith, N.G., Cutchin, S., Kooima, R., Ainsworth, R.A., Sandin, D.J., Schulze, J., Prudhomme, A., Kuester, F., Levy, T.E., DeFanti, T.A.: Cultural heritage omni-stereo panoramas for immersive cultural analytics - from the Nile to the Hijaz. In: 2013 8th International Symposium on Image and Signal Processing and Analysis (ISPA), pp. 552–557 (2013)
54. Spiers, H.J., Maguire, E.A.: Thoughts, behaviour, and brain dynamics during navigation in the real world. Neuroimage **31**(4), 1826–1840 (2006)
55. Sturz, B.R., Bodily, K.D., Katz, J.S.: Evidence against integration of spatial maps in humans. Anim. Cogn. **9**(207), 207–217 (2006)
56. Taube, J.S., Valerio, S., Yoder, R.M.: Is navigation in virtual reality with fMRI really navigation? J. Cogn. Neurosci. **25**(7), 1008–1019 (2016)
57. Taylor II, R.M., Hudson, T.C., Seeger, A., Weber, H., Juliano, J., Helser, A.T.: VRPN: a device-independent, network-transparent VR peripheral system. In: Proceedings of the ACM Symposium on Virtual Reality Software and Technology, vol. 900, pp. 55–61 (2016)
58. Unity Technologies: Unity3D (2016). http://unity3d.com/
59. Usoh, M., Arthur, K., Whitton, M.C., Bastos, R., Steed, A., Slater, M., Brooks, F.P.: Walking > walking-in-place > flying, in virtual environments. In: Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, pp. 359–364 (2016)
60. Vanoni, D., Ge, L., Kuester, F.: Intuitive visualization of reflectance transformation imaging for interactive analysis of cultural artifacts. In: International Conference on Augmented and Virtual Reality, vol. 8853, pp. 397–404 (2016)
61. Vass, L.K., Copara, M.S., Seyal, M., Shahlaie, K., Farias, S.T., Shen, P.Y., Ekstrom, A.D.: Oscillations go the distance: low-frequency human hippocampal oscillations code spatial distance in the absence of sensory cues during teleportation. Neuron **89**(6), 1180–1186 (2016)
62. VIS Games: Country landscape (2016). http://www.vis-games.de/
63. Wallet, G., Sauzeon, H., Pala, P.A., Larrue, F., Zheng, X.: Virtual/real transfer of spatial knowledge: benefit from visual fidelity provided in a virtual. Cyberpsychol. Behav. Soc. Networking **14**(7), 417–423 (2016)
64. Weisberg, S.M., Schinazi, V.R., Newcombe, N.S., Shipley, T.F., Epstein, R.A.: Variations in cognitive maps: understanding individual differences in navigation. J. Exp. Psychol. Learn. Mem. Cogn. **40**(3), 669–682 (2016)
65. Wickham, H.: ggplot2: Elegant Graphics for Data Analysis. Springer, New York (2016). http://ggplot2.org
66. Wickham, H., Francois, R.: dplyr: A grammar of data manipulation (2016). https://cran.r-project.org/package=dplyr
67. WorldViz LLC: Vizard (2016). http://www.worldviz.com/
68. Zatun: City development (2016). http://zatun.com/