VR for Everybody: The SNaP Framework

Michelle Annett, Walter F. Bischof
Department of Computing Science, University of Alberta
mkannett@cs.ualberta.ca, wfb@ualberta.ca

Abstract

Over the past decade, Virtual Reality (VR) has garnered widespread use in psychology, cognitive neuroscience, and especially spatial navigation research. Current VR systems are difficult to design and configure, particularly when arrays of hardware peripherals are used. Issues with hardware integration, architectural design, and usability have led to the creation of inflexible and immobile VR systems. We present a potential solution to these issues, the SNaP Framework. The framework integrates VR input and output peripherals and an easy-to-use configuration medium (XML) with a popular software development suite (Virtools) to create a VR system that supports users with different computer skills. We report on the architecture of the SNaP Framework and provide details from a study that measured the usability of the framework. We conclude with a discussion of experiences encountered while designing and implementing the system.

1. Introduction

Over the past decade, there have been many advances in the field of virtual reality (VR). Although progress in 3D modeling and hardware devices have made VR more exciting, immersive, and interactive, it is still very difficult to design and deploy a VR environment. While there are a number of software solutions and systems available to researchers [1, 2, 3], most require knowledge of programming languages, APIs, and libraries. These solutions often employ difficult configuration methods and complex behavioural algorithms. It is also common for these solutions to permit only restricted user control and to be inflexible and inextensible. For users with limited exposure to VR or programming, it can be very difficult to integrate VR into their work.

VR has been used in a variety of domains for experimental purposes, including, for example, therapy and rehabilitative medicine. As more domains begin to use VR for experimental purposes, it is crucial that VR systems support common experimental procedures or paradigms. Systems should allow users to specify VR environment configurations for experimental training and test trials (as well as trial blocking and repetition), specify paradigm constraints and goals, systematically record behavioural measurements, and allow for the integration of other behaviour measurement systems (e.g., EEG, fMRI, MEG, and eye-trackers). Current VR systems do not support such requirements, making it difficult to use VR in experimental situations.

Given the problems of using VR in research, we designed a system that assists researchers in specifying and deploying virtual environments for experimental purposes. The Spatial Navigation Paradigm, or SNaP, framework allows novice and expert VR users to create highly immersive environments for spatial navigation research. Using a commercial software development suite (Virtools), a number of Python-based modules, and an XML-based configuration medium, the SNaP Framework makes it trivial for both novices and experts to specify and deploy virtual environments.

In this paper, we first describe the architecture and features of the SNaP Framework. We then compare the usability and effectiveness of the SNaP Framework to existing systems and provide results from a pilot usability study. We conclude with a discussion of the experiences encountered while developing and implementing the SNaP Framework.

2. Proposed System

The goals of the SNaP Framework are twofold. First, we want to provide support for a variety of experimental protocols. Second, we want to create a framework that novices can use to easily specify and deploy environments and experts can use to easily modify and extend existing experimental setups. The architecture of the framework is described next, followed by a description of how users can specify and deploy spatial navigation experiments.

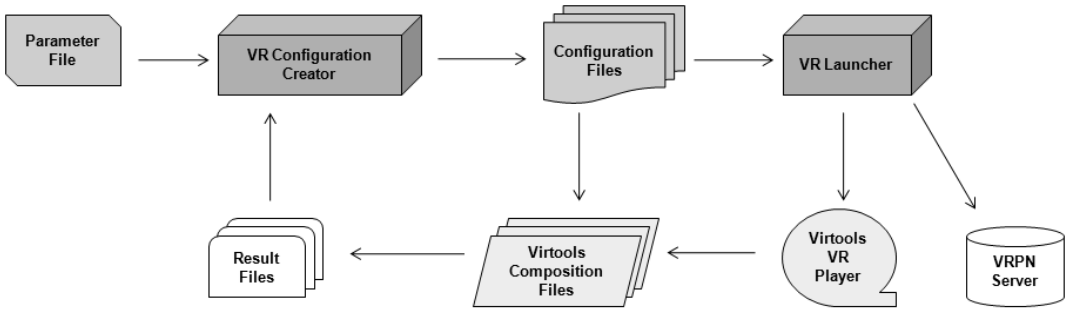

Figure 1. SNaP Framework Architecture.

```
<ExperimentSpecification>

<setup>
   <experiment code="ComplexMaze"/>
   <inputdevice type="keyboard"/>
   <outputdevice type="monitor"/>
   <alternativedevice type="EEG"/>
   <baseline repetitions="5"/>
   <measures type="camera,path,overall"/>
   <phases number = "2"/>
</setup>

<phase>
   <trainblock number = "1">
      <trial keywords="feedback,goal15,time3"/>
   </trainblock>
   <testblock number = "1">
      <trial keywords="nofeedback,goal15,time3"/>
   </testblock>
</phase>

<phase>
   <trainblock number = "1">
      <trial keywords="feedback,goal19"/>
   </trainblock>
   <testblock number = "1">
      <trial keywords="nofeedback,goal19"/>
   </testblock>
</phase>

</ExperimentSpecification>
```

Figure 2. Example of an XML Parameter file for the Complex Maze Paradigm [4].

2.1 Architecture

There are a number of modules, software components, and files that work together to allow a user to specify and deploy a virtual environment using the SNaP Framework (Figure 1). As each experiment that is performed is different, the SNaP Framework requires that an XML-based *parameter file* be written for each one. Using the format specified in an XML XSD schema, a complete experiment (i.e., number of trials, trial types, trial conditions, and desired peripherals) can be described using a limited number of XML nodes and attributes (Figure 2).

Once created, this parameter file is used as input to a Python module, the *VR Configuration Creator*. The VR Configuration Creator transforms each trial in the parameter file into a separate XML-based *configuration file*. This file specifies the environmental and protocol configurations for a given experimental trial (e.g., input and output peripherals, feedback, the trial goal, 3D model locations and orientations, and the behavioural measurements to record).

Every experimental paradigm has an associated virtual world. Each virtual world is described in a *Virtools Composition file*. This file contains a number of custom scripts (to control user interaction, behavioural recording), environment models (e.g., floors, walls, goal objects, a virtual character, cameras, and an experimental holding room), and universal modules (to handle paradigm requirements). The *XML Parser* and *World Generator* modules transform the specifications detailed in a configuration file into a trial-specific virtual environment.

Once the XML Parser has parsed a configuration file and the World Generator has generated a virtual world, the environment is then deployed to an output context. Another Python module, the *VR Launcher*, determines which deployment context is desired. The VR Launcher works with the *Virtools VR Player* to render an environment. The SNaP Framework can display virtual worlds using a number of peripherals: CAVEs, HMDs, and single or multiple monitors.

The VR Launcher also determines which behavioural measurement systems and input peripherals are desired and if a *VRPN server* should be started. The open source VRPN server is used to transform input devices into generic device types. In the SNaP Framework, one common script encapsulates all possible peripheral device types and reads data from the VRPN server when necessary. The values obtained from these generic device types control character behaviour. Participant interaction in the SNaP Framework can be achieved using a mouse, keyboard, joystick, space mouse, wand, head tracker, or Wiimote.

Each composition file also contains two additional

modules: the *Goal Monitoring Module* and the *Metric Module*. While the participant is navigating through an environment, the Goal Monitoring Module monitors the participant's progress to determine if the trial goal has been fulfilled. Simultaneously, the Metric Module records a variety of participant behavioural measurements (e.g., participant path information, camera view frustums, and overall experimental results) and writes them to text and bitmap files.

Once the goal of the trial has been completed, the participant's behavioural results are written, the VR Configuration Creator writes another configuration file, and the next experimental trial is deployed. With this architecture, it is easy for both novices and experts to specify, configure, and deploy VR-based experiments.

2.2 SNaP Framework Usage

To assist users with the creation of virtual environments, the SNaP Framework provides a template Virtools composition file. Within this file are all of the modules mentioned in Section 2.1, as well as a number of common 3D models (e.g., walls, a floor, and a virtual character). We used the SNaP Framework's template environment to implement five spatial navigation paradigms: the Complex Maze [4], the Cheng Task [5], the Bucket World [6], the Scatter Hoarding Task [7], and the Virtual Morris Water Maze [8]. Each of these implementations is included in the current version of the SNaP Framework.

To create an experiment, the user must write a parameter file following the paradigm's XML XSD Schema. Users experienced in programming can make a number of modifications to an existing paradigm implementation using the Virtools Dev environment, or they can implement new spatial navigation paradigms using the template environment.

Once a parameter file is written, the user can double click on the experiment's batch script (e.g., 'Run Complex Maze.bat') and the experiment is deployed via the VR Launcher, using the deployment context and peripherals specified in the parameter file. The user is not required to perform any additional actions to switch between peripheral devices or experiments. Using this method, users of all skill sets can easily deploy an environment.

3. Evaluation

All VR systems have different architectures, support different peripherals, render virtual environments differently, implement different paradigms and protocols, use different behavioral measurement techniques, and require different steps methods to perform tasks. It is thus difficult to compare the effectiveness and usability of the SNaP Framework to other systems. Two types of evaluations were performed to evaluate the effectiveness and usability of the SNAP framework. First, as described in Section 3.1, we devised a set of simple tasks that were required for all VR experimental implementations (not just spatial navigation) and determined how such tasks would be performed using the SNaP Framework and other systems. Second, we performed a pilot study to determine novice and expert user's opinions on the ease of use and effectiveness of the SNaP Framework. The study is described in Section 3.2

3.1 SNaP Framework versus Other Systems

One way to measure the usability between different VR systems is to compare the steps required to perform a variety of abstract tasks that are required for all experimental VR paradigms (e.g., design an environment, setup an experimental trial, and deploy an experiment). Four such tasks are provided in Table 1, along with a comparison between the SNaP Framework and three other VR systems, NeuroVR [9], VR Worlds 2 [10], and custom systems created by research groups [11, 12, 13].

An analysis of the steps required to perform these tasks makes it clear that the SNaP Framework has simplified a number of them, mainly the setup and deployment of an experiment, as well as switching between peripherals and between experiments. Given that a scripting language is not used to specify an experiment and that minimal input is required from the user, the SNaP Framework is currently one of the simplest systems that can be used to create and deploy a VR experiment.

The SNaP Framework excels in switching between VR devices because it integrates the flexibility built into VRPN with the abstraction provided by the Virtools VR Player. The user is not required to configure and setup peripheral devices or alternative behavioural recording devices. Hence, the SNaP Framework has made it easier for researchers to obtain behavioural measurements and to integrate new peripherals into experiments.

The experiment protocol support provided in the template environment allows expert users to easily implement experimental trials and enforce paradigm constraints. It is also possible to analyze and generalize results between participants and across paradigms because the SNaP Framework allows for the deployment of multiple paradigms that use the same

Table 1. Steps required to perform four tasks required for virtual reality experiments.

Create a virtual environment: • SNaP Framework: The user creates a number of 3D models using a modeling program, such as Maya, imports them into the Virtools Dev, and modifies the template environment as needed. • NeuroVR: The user selects a pre-made environment and makes modifications to it using a drag-and-drop or option-based interface. Additional 3D models can be added, but only if they have been created with the Blender modeling program. • VR Worlds 2: The user is presented with a blank canvas and adds 3D models or customized characters to it from the VR Worlds 2 library. • Custom Systems: The user creates an environment using a specific 3D modeling program, or modeling specification, and interfaces with all of the components and hardware in their system themselves. *Easiest Method:* SNaP Framework, NeuroVR, or VR Worlds 2.	**Setup or specify an experiment:** • SNaP Framework: The user writes an XML-based parameter file containing the types of trials and peripherals to be included in an experiment. • NeuroVR: A new environment is created for each experimental trial. Using the NeuroVR interface, each trial is opened in succession. • VR Worlds 2: The user creates a new environment for each trial. Each trial is opened manually, in succession, from within the VR Worlds 2 interface. Alternatively, a user can create a large environment with multiple experiment environments. In this large environment, the participant is transported to a new location on each trial. • Custom Systems: The user creates a parameter file containing a list of experimental trials and modifications for each trial. Alternatively, a complete trial sequence can be programmed into the system. *Easiest Method:* SNaP Framework.
Deploy an experiment: • SNaP Framework: The user double clicks on a paradigm specific batch script and the experiment is run, trial-by-trial. • NeuroVR: From within the NeuroVR player, the user selects the scene to deploy. Once this experimental trial has been completed, the GUI interface is used to load the next scene. Continuous trial playback is not supported. • VR Worlds 2: The user opens each new environment, compiles, and then plays it. Once the trial has been completed, the user opens the next environment. • Custom Systems: The user clicks on a batch script or types a command into a terminal or console window. The various experimental trials are presented automatically. *Easiest Method:* SNaP Framework or Custom Systems.	**Switch between peripheral devices:** • SNaP Framework: The user declares which devices are to be used in the experiment parameter file using the necessary peripheral keywords. • NeuroVR: The user clicks on the name of a peripheral in a list or selects 'keyboard', 'mouse' or 'gamepad' using radio buttons. The addition of other peripherals is not supported. • VR Worlds 2: The user clicks on the 'Hardware' tab and selects either a monitor or HMD. The addition of other peripherals is not supported. • Custom Systems: The user configures the desired peripheral and creates a new program or module to handle the device. Most often, modulation between peripherals is not permitted. *Easiest Method:* SNaP Framework.

behavior and program logic, configuration media, user interaction techniques, and metric gathering algorithms.

Regardless of the user's skill set, the SNaP Framework greatly reduces the time required to implement an experiment and simplifies the process of using multiple peripherals in an experiment.

3.2 Usability Study

The SNaP framework was also evaluated in a usability study. The purpose of the study was to gather feedback from users about the usability and effectiveness of the SNaP Framework.

Participants

Eight participants (4 males and 4 females, with an age range of 19 – 48 years) participated in the study. Three participants were computer novices who were unfamiliar with VR technologies (Group A). Three other participants were computer experts but were unfamiliar with VR technologies or virtual environments (Group B). The remaining two participants were computer and VR experts (Group C). At least one participant in each group was unfamiliar with experimental testing procedures (i.e., trials, blocks, phases). Most participants were unfamiliar with the spatial navigation paradigms they were asked to use.

Tasks

All participants were asked to specify and deploy two experiments using different spatial navigation paradigms. Three different paradigms were used: the Cheng Task, the Bucket World, and the Complex Maze. These tasks required participants to write an XML-based parameter file for each experiment and to use the provided VR Launchers to deploy each experiment. Participants gained experience specifying different input and output peripherals and levels of metrics and setting up different trial and experimental conditions.

The two expert users were asked to perform two additional tasks. In these tasks, several modifications were made to the Complex Maze and Scatter Hoarding tasks. These tasks required the participants to manipulate the modules in the Virtools composition files to include different types of experimental feedback (both auditory and visual). Participants gained experience understanding the SNaP Framework modules and using the Virtools Dev interface to modify the framework as needed.

Measurements

After the completion of all tasks, each participant was asked to complete a modified version of the IBM Post-Study System Usability Questionnaire [14]. The questionnaire had 20 questions that assessed overall user satisfaction and system usefulness, simplicity, and effectiveness. Participants were also videotaped and a number of objective measurements were recorded, including the time to task completion, the number of XML typos made, the number of compilations or executions required (for the experts only), and the number and nature of questions asked by participants.

Results

Results show that it took on average ten minutes for each participant to specify and deploy an experiment. Half of the participants made no errors writing the XML parameter files or deploying the experiments, while the others made only one or two errors and were able to correct them very quickly. For the expert tasks, both users were able to complete each task in less than twenty minutes and asked less than four questions, most of which were related to technical nuances of Virtools.

The questionnaire results indicated that users felt that the framework made it easy to create and deploy experiments. All users also felt that the SNaP Framework made the integration and modulation between peripherals simple; most were quite astonished that this task could be transformed into something so trivial. Lastly, all participants were satisfied with the ease-of-use and ease-of-learning of the framework and felt very comfortable using it to create experiments. A participant in Group A indicated, "I was not frustrated [using the framework] even though my computer knowledge is limited". The two expert users also felt that it was easy to understand the modules contained within the Virtools composition files and felt it was quite simple to modify an existing paradigm using the framework. Overall, the study shows that the SNaP Framework simplifies the process of specifying and deploying VR experiments and allows users to easily switch between various input and output peripherals.

4. Experiences

In analyzing the functionalities needed for the SNaP Framework, it was apparent that it would be faster and more economical to enhance an existing VR system rather than to create a custom system from scratch. Among the available systems, Virtools was chosen because it allowed for the most developer freedom and flexibility. Using the Virtools Dev interface, environments and behaviour can be created using a visual programming language, a scripting language, or C++ plug-ins. Virtools also comes with support for a variety of 3D modeling programs and is compatible with a wide variety of peripherals.

Although Virtools has been used for research in the past, the SNaP Framework is one of the first projects to build into, and on top of, Virtools. Using Virtools in this way created a number of problems, the foremost being the number of 'hacks' required to make the suite and its environments work properly. One such 'hack' involved the amount of information that Virtools' allows users to send from master to slave machines. As the amount of information we needed to send from one machine to another exceeded the allowable amount, we had to create separate streams of master and slave program logic to be executed whenever a boolean value was distributed. With this solution, events appear to be distributed across all machines, when in reality each machine is running its own instantiation of paradigm logic.

Another Virtools problem, only visible to developers, lies in the architecture of the Virtools suite itself. Virtools uses the VR Player, VRPN server, Virtools Dev, and a large number of configuration files to create and deploy experiments. It was difficult to have Virtools always ready to handle all of the peripherals we wanted to use. It was also very difficult to create interactions between Virtools and additional modules, such as the VR Configuration Creator or VR Launcher.

In general, XML proved to be a beneficial choice for the configuration media because it is human readable and acts as a form of documentation for the framework. Many participants in the usability study indicated that they easily and quickly understood the structure and formatting of XML.

As there were a number of common 3D models required in all paradigms we implemented, it was trivial to design each model and then simply apply different textures or add different properties to them. This reduced the time needed to create a virtual environment.

One difficulty encountered while designing and implementing the framework concerned the amount of flexibility and level of user control that should be provided to users. It is reasonable to expect novice users to write a configuration script or parameter file, but unreasonable to require them to write C++ code to handle behavioural measurements or user navigation. By creating and structuring the configuration files with

a limited, but extensible, number of modifications, the framework can be used by both, novice and expert users. Expert users are provided with a template environment that contains all elements necessary to gather metrics, to process user interactions, and to generate an environment. This increases the usability of the framework while requiring the smallest amount of programming knowledge possible. Experts can use the template environment to implement new paradigms in a short amount of time and with minimal effort.

5. Conclusions

The SNaP Framework uses the Virtools development platform, custom Virtools scripts, XML-based specification files, and custom Python modules to allow novices and experts to specify, configure, and deploy spatial navigation experiments. The framework allows users to switch between paradigms, input and output devices, and behavioural measurement systems with ease. It also supports the implementation of experimental paradigms and protocols and accurately records participant behaviour.

The usability study and the comparison of the SNaP Framework with other systems illustrated that the SNaP Framework simplifies the process of creating VR-based experiments for spatial navigation research. The ideas and experiences generated by the framework should help avoid some of the hardware issues that have plagued virtual reality systems and those wanting to use VR for experimental purposes.

Acknowledgements

WFB was supported by a grant from the Natural Sciences and Engineering Research Council of Canada.

References

[1] A. Bierbaum, C. Just, P. Hartling, K. Meinert, A. Baker, and C. Cruz-Neira. VR Juggler: a virtual platform for virtual reality application development. In *Proceedings of Virtual Reality 2001*, 2001, pp. 89 - 96.

[2] S. Baumann, C. Neff, S. Fetzick, G. Stangl, L. Basler, R. Vereneck, and W. Schneider. A Virtual Reality System for Neurobehavioural and Functional MRI Studies. *CyberPsychology & Behavior*, 6(3), 2003, pp. 259 - 266.

[3] H. R. Nagel and E. Granum. Vr++ and its application for interactive and dynamic visualization of data in virtual reality. In *Proceedings of the Eleventh Danish Conference on Pattern Recognition and Image Analysis,* 2002.

[4] W. F. Bischof and P. Boulanger. Spatial Navigation in Virtual Reality Environments: An EEG Analysis. *CyberPsychology & Behaviour*, 6(5), 2003, pp. 487 - 495.

[5] D. M. Kelly and W. F. Bischof. Reorienting in images of a three-dimensional environment. *Journal of Experimental Psychology: Human Perception and Performance*, 31(6), 2005, pp. 1391 - 1403.

[6] B. R. Sturz, M. F. Brown, and D. M. Kelly. Facilitation of spatial pattern learning with visual cues in real and virtual environments: implications for associative accounts of spatial learning. *Psychonomic Bulletin & Review*, (in press).

[7] W. F. Bischof and D. M. Kelly. Navigation and Localization in open environments. In *Proceedings of the 16th Annual Meeting of the Canadian Society for Brain, Behaviour, and Cognitive Science*, 2006, pp. 49.

[8] S. D. Moffat and S. M. Resnick. Effects of Age on Virtual Environment Place Navigation and Allocentric Cognitive Mapping. *Behavioural Neuroscience*, 116(5), 2002, pp. 851 - 859.

[9] G. Riva, A. Gaggioli, D. Villani, A. Preziosa, F. Morganti, R. Corsi, G. Faletti, and L. Vezzadini. A Free, Open-Source Virtual Reality Platform for the Rehabilitation of Cognitive and Psychological Disorders. *Virtual Rehabilitation,* 2007, pp. 159 - 163.

[10] Psychology Software Tools, Inc. *VR Worlds 2*. http://www.vrworlds2.com/index.cfm. Accessed: January 2009.

[11] D. S. Pine, J. Grun, E. A. Maguire, N. Burgess, E. Zarahn, V. Koda, A. Fyer, P. R. Szeszko, and R. M. Bilder. Neurodevelopmental aspects of spatial navigation: a virtual reality fMRI study. *NeuroImage*, 15(2), 2002, pp. 396 - 406.

[12] M. Wolter, C. Armbrüster, and J. T. Valvoda. High Ecological Validity and Accurate Stimulus Control in VR-based Psychological Experiments. In *Proceedings of Eurographics Symposium on Virtual Environments*, 2007, pp. 25 - 32.

[13] R. Mraz, J. Hong, G. Quintin, W. R. Staines, W. E. McIlroy, K. K. Zakzanis, and S. J. Graham. A Platform for Combining Virtual Reality Experiments with Functional Magnetic Resonance Imaging. *CyberPsychology & Behaviour*, 6(4), 2003, pp. 359 - 368.

[14] J. R. Lewis. IBM Computer Usability Satisfaction Questionnaires: Psychometric Evaluation and Instruction for Use. *International Journal of Human-Computer Interaction*, 7(1), 1995, pp. 57 - 78.