Michelle K. Annett*
Walter F. Bischof
Department of Computing Science
University of Alberta
Edmonton, Alberta, Canada
T6G 2E8

Investigating the Application of Virtual Reality Systems to Psychology and Cognitive Neuroscience Research

Abstract

With interest in virtual reality (VR) technologies, techniques, and devices growing at a quick pace, many researchers in areas such as psychology or cognitive neuroscience want to use VR. The software and VR systems available today do not support the skill sets or experimental requirements of this group of users. We describe a number of concerns and requirements that researchers express and focus on the extent to which today's VR systems support non-VR experts. The work then concludes with a number of suggestions and potential development avenues that should be undertaken to ensure that VR systems are usable by a large range of researchers, regardless of their programming skills or technical backgrounds.

1 Introduction

Virtual reality (VR) permits people to manipulate, interact with, immerse themselves in, and navigate through highly mimetic or fantasy-based 3D environments. With interest in VR growing, virtual environments have been used in a diverse range of research areas, including, for example, scientific visualization (LaViola, Prabhat, Forsberg, Laidlaw, & van Dam, 2008), physical rehabilitation (Rizzo & Kim, 2005), architectural design (Drettakis, Roussou, Reche, & Tsingos, 2007), engineering (Weidlich, Cser, Polzin, Cristiano, & Zickner, 2009), and job skill training (Watanuki & Kojima, 2006). AQ: 1

Within the areas of psychology and cognitive neuroscience, virtual reality has been used to study spatial navigation, social disorders, and phobias (Maguire et al., 1998; Pine et al., 2002; Mraz et al., 2003; Riva, 2005; Wolter, Armbrüster, Valvoda, & Kuhlen, 2007; Lehmann, Vidal, & Bülthoff, 2008). VR has been used in these settings because it allows precise stimulus and environmental control, enables accurate performance and behavioral measurements to be recorded, and allows researchers to present environments and situations that would be unethical or impossible to present in traditional settings.

With increased interest from noncomputing science and nonengineering researchers, it is important to assess how well current VR systems meet the needs of such a diverse group of users. If we want to make VR universally ac-

Presence, *Vol. 19, No. 2, April 2010,* 000–000

*Correspondence to mkannett@cs.ualberta.ca.

cessible, then we need to make the systems of today and tomorrow more usable by non-VR experts, regardless of their programming skills or prior VR exposure. Four qualitatively different groups of users can be identified in these application areas: scientists, content creators, programmers, and VR experts.

- VR experts are assumed to have extensive programming backgrounds and exposure to VR. They can design 3D models and environments, integrate new VR hardware, and retarget and deploy environments easily. They expect that systems provide scripting languages to implement behavior and environment constraints.
- Programmers are assumed to have good programming and scripting skills, but little exposure to VR technologies or 3D modeling. They prefer to use scripting or low-level programming languages to implement basic behaviors. This group of users is often asked to build and deploy experimental environments for others.
- Content creators often have limited programming experience and very little exposure to VR hardware or technologies. This group of users often has strong 3D modeling or animation skills and is routinely tasked with creating environment content. They often work with programmers to create virtual environments.
- Scientists use VR for clinical or experimental purposes. Scientists are unable to set up VR environments due to their limited programming skills. They expect support for different experimental paradigms and methodologies, behavioral recording, and the ability to choose between input, output, and behavioral recording peripherals.

The expectations of these four groups all relate to the usability of VR systems. Some expectations are general, applying to all VR environment applications, while others are task specific, relating only to experimental situations (such as in psychology).

- How are virtual environments designed?
- How are paradigms (i.e., experimental methodologies) or environmental constraints implemented?
- Do systems allow users to choose between different input peripherals (e.g., keyboard, joystick, space mouse, or head trackers) or deployment contexts (e.g., head-mounted displays, HMDs, CAVEs, single or multiple monitors)?
- How is user interaction incorporated into (and used in) VR environments?
- How are environmental constraints integrated into a system?
- How are behavioral measurements recorded?
- Can current systems interface with other systems and hardware (e.g., EEG, fMRI, MEG, eye trackers, game controllers, or haptic devices)?

As the number of users wanting to use VR increases, it is important to solve these issues so that a solid foundation for future systems can be created. The remainder of this paper addresses how the architecture of VR systems supports these different groups of users.

2 Current Solutions

For VR to be useful in many different domains, it is important to look at how current VR technologies and systems address usability requirements. We first review current virtual reality systems that have been used in psychology and cognitive neuroscience. Next, we analyze these systems to identify which usability problems still need attention.

2.1 VR Systems and Platforms

Many software suites, both commercial and open source, have been used in psychology and cognitive neuroscience to implement paradigms and run experiments. In this area of research, two methods have been used to implement paradigms: (1) create custom in-house systems composed of freely available or tailor-made components, and (2) use open source or commercial systems that integrate VR devices with environment creation software. Systems from each category are described next.

2.1.1 Canned Systems. ONe of the first systems created to help psychologists and cognitive neuroscientists was Presentation. Presentation is a platform that uses a drag-and-drop GUI and scripting languages to create and control simple 3D stimuli, navigable spaces, and experiments. It interfaces with a variety of input devices, supports eye-tracking hardware, and allows for the integration of behavioral recording devices (e.g., fMRI, EEG, or MEG). However, it does not support the inclusion of complex virtual scenes, nor does it allow for the stereo deployment of environments.

Two of the newer platforms available to psychologists and neuroscientists are VR Worlds 2 and NeuroVR (Riva et al., 2007). VR Worlds 2 focuses on creating realistic virtual environments for various application areas (e.g., drug rehabilitation, spatial navigation, phobia therapies, and anxiety disorders). It combines a drag-and-drop interface with 3D object libraries, custom event handling, data logging, peripheral device interfacing, and motion tracking. Unfortunately, VR Worlds 2 does not give the user much control over the environments that can be created. Users are limited to using preprogrammed environment options such as "city street—night" or "city street—day" and cannot choose which behavioral measurements to record. NeuroVR is a Blender-based platform that contains a drag-and-drop, icon-based editor interface for creating and modifying virtual environments. NeuroVR has a library of premade 3D models. Environments can be deployed to an HMD or a monitor, and it supports head trackers, joypads, keyboards, and mice. It lacks, however, many important features, such as stereo capability, realistic animations, behavioral monitoring support, extensible environments, a scripting medium, and extended peripheral support.

Virtools is a system very different from NeuroVR, Presentation, and VR Worlds 2. It is a general VR development system that creates highly interactive, immersive virtual environments and has shown much promise in psychology and cognitive neuroscience research (Riva et al., 2003). Using the simple drag-and-drop metaphor of visual programming, the Virtools Development Environment (Virtools Dev) allows users to design environments and implement user behaviors with ease. The Virtools Dev drag-and-drop interface contains a library of 3D objects and uses a visual programming language composed of behavioral building blocks. It also provides support for midlevel scripting using the Virtools Scripting Language (VSL) and low level programming via a C++ SDK. Virtools deploys virtual worlds to a number of immersive output devices using the Virtools Player.

Several other general, commercial VR development systems have also been used to implement paradigms in psychology and cognitive neuroscience, including, for example, Quest3D, WorldViz, WorldUp, VR4Max, and Superscape VRT. These systems support visual programming, have scripting capabilities, support the inclusion of multiple peripherals and various 3D model formats, and have a number of specialized modules that handle avatar behavior, camera control, and stereo deployment.

In recent years, there have also been a number of online virtual environments or development systems, such as Second Life, There, or HiPiHi, that could be used in psychology or cognitive neuroscience experiments. These systems strive to be completely user generated. There are numerous tools in place to ensure that users of all skill levels and ages can create quality content and interactions. These systems also have support for a number of input peripherals and allow users to have full 3D navigation and interaction. While they provide much support for novice users, they do not allow users to record any type of measurements or create custom interaction patterns, they require an internet connection, and they have limited flexibility and extensibility.

2.1.2 Custom VR Systems. Many custom systems have been developed to implement and deploy virtual reality based experiments. These systems use a combination of graphics packages, game engines, environment generators, programming languages, APIs, and libraries.

A number of APIs, languages, and libraries such as VR++, OpenSG, Open Scene Graph, VRML, X3D, and COVE have been used to handle peripheral integration, environment specification, and environment rendering. Scripting or programming languages are commonly used to implement user interaction or

environmental constraints. For example, Wolter et al.'s (2007) ReactorMan system uses OpenSG, OpenGL, OpenAL, and ViSTA to generate paradigms and C++ to handle user interaction. Kelly and Bischof (2005) used POV-Ray to generate images of 3D environments and Python to deploy their environment and gather behavior-based measurements. Mraz et al. (2003) developed an fMRI-compatible VR system that relied on OpenGL, WorldUp, C, and Visual Basic for environment creation and behaviors.

2.1.3 Solutions to Device Integration. Peripheral integration has been a long-standing problem with virtual reality systems. Most VR environments require only one input and output peripheral, but if complex devices, such as head trackers, haptic devices, CAVEs, or space mice are required, they can be very difficult to integrate into a VR system.

In a custom system, libraries or APIs are used to handle virtual environment deployment and input peripherals. For example, CAVELib is an API that can deploy virtual environments to immersive devices. It is based on C and C++ and allows developers to create 3D environments without worrying about stereo projections, camera viewports, threading, cluster distribution, or synchronization. An open source alternative to CAVELib is VRJuggler, an object orientated API that uses C++ to interface with other APIs such as OpenGL and OpenAL. It allows users to retarget environments without recompilation and is compatible with a number of output devices, including HMDs, CAVEs, and Power walls. Trackd and VRPN are middleware applications that use a server and daemons to interface between input peripheral devices and virtual environments, systems, or libraries. Both solutions abstract devices into categorical base-type classes (e.g., analog devices, buttons, and tracker devices), making it is easy to integrate and support a range of peripheral devices without having to rewrite source code or create custom device drivers. Trackd and VRPN make it easy for programmers or VR experts to integrate new input devices into a custom VR system.

For measurement devices and equipment, there are currently no solutions similar to VRPN or CAVELib.

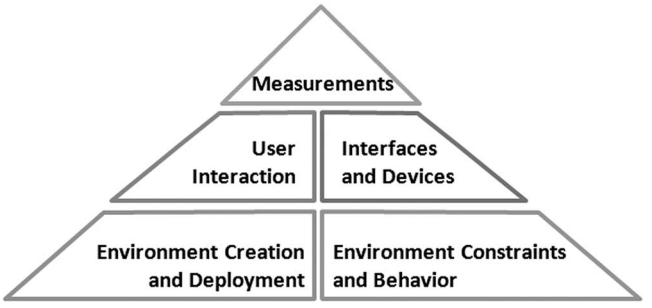

Figure 1. *Five components of VR systems that would benefit from usability assessment and improvements.*

For off-the-shelf measurement systems such as EEG, MEG, EMG, or eye trackers, VR experts usually write custom modules to interface with their VR system (e.g., Bischof & Boulanger, 2003) or use additional software such as MATLAB or LABVIEW (e.g., Presentation). In the case of fMRI, not only do researchers need to use customized software packages to synchronize measurement data, but they also need to create special input and output devices that require minimal head movements from users and can also be used within an fMRI magnet (Pine et al., 2002; Hoffman, Richards, Coda, Richards, & Sharar, 2003; Mraz et al., 2003; Lee, Lim, Wiederhold, & Graham, 2005; Beck, Wolter, Mungard, Kuhlen, & Sturm, 2007).

2.2 Assessment of Current Solutions

We evaluate the usability and user requirements of current systems by reviewing five aspects of VR systems that are important to VR-based experiments, as shown in Figure 1. The findings of this evaluation are summarized in Table 1. F1 T1

- **Environment Creation and Deployment.** How are 3D models created? What skills are required to design these models? How are behaviors added to models? Is a drag-and-drop GUI or a scripting or programming language used?
- **Environment Constraints and Behavior.** How are behaviors such as object avoidance added to an environment? Are experimental protocols sup-

Table 1. *Comparison of VR Systems Used in Psychology and Neuroscience*

VR system type	Environment creation and deployment	Environment constraints and behavior	Interfaces and devices	User interaction	Measurement	Intended user group
A. Presentation	The Scenario Scripting Language is used to create experiments and the Presentation Scripting Language is used to integrate media. Environments are deployed using the Presentation interface and GUI elements.	Each experimental trial (scenario) is created using the Scenario Scripting Language. Each scenario is added to the main experiment file using a GUI.	A limited list of input and output peripherals are supported and selected using a GUI. Plug-ins can be written to handle additional devices.	User interaction and navigation types are specified using the Scenario or Presentation Scripting Languages.	Predetermined metrics are recorded (e.g., general events, stimuli events) Additional metrics can be added using the Presentation Scripting Language.	Programmers, VR experts
B. Neuro VR	A premade environment is selected and then modified. Additional Blender 3D models can be imported. Environments are deploying using the NeuroVR player.	A new environment is created for each experimental trial. Behaviors are not customizable.	A limited list of input and output peripherals are supported and selected using a GUI. Additional devices can be added.	There is no control over user interaction or navigation types.	No metric support.	Scientists, content creators
C. VR Worlds 2	A user is presented with a blank canvas and adds 3D models or characters to it from the VR Worlds 2 library. External models or content is not permitted.	A new environment is created for each experimental trial. Behaviors are not customizable.	A limited list of input and output peripherals are supported and selected using a GUI. Additional devices can be added.	User interaction can be specified via a GUI. There is no control over navigation types.	Predetermined metrics are recorded (e.g., user position, orientation, time, events).	Scientists

tapraid4/z92-psen/z92-psen/z9200210/z922133d10g | coonanl | S=6 | 3/10/10 | 11:46 | Art: z92-2133 | Input-maw(maw)

Table 1. *Continued*

VR system type	Environment creation and deployment	Environment constraints and behavior	Interfaces and devices	User interaction	Measurement	Intended user group
D. Virtools/SNaP framework	3D models can be created using many different modeling programs, or a prebuilt environment can be modified. 3D content can be integrated using the Virtools Library. Experimental paradigms are implemented using parameter files and experiments are run by clicking on a batch file.	The user writes an XML parameter file to define the blocks, phases, and trials wanted in a given experiment. Additional behaviors can be customized using the Virtools Visual Programming Language, Scripting Language, or SDK.	User declares the devices they would like to use in their parameter file (using keywords). Additional devices can easily be added via new configuration files.	User behaviors and navigation types are specified in parameter files or can be implemented using the Virtools Visual Programming Language, Scripting Language, or SDK. Environments can support full 3D interaction and navigation.	Predetermined metrics are recorded (e.g., participant path, general metrics, and camera-viewpoint images). Additional metrics can be specified using a parameter file, the Virtools Visual Programming Language, Scripting Language, or SDK.	Scientists, programmers, content creators, VR experts
E. Second Life/ There/HiPhHi	Users of any skill level can create content using the content creators that are provided. All environments and situations are run within client applications on a PC.	There is no support for experimental paradigms (trials, phases, or blocks). Plug-ins could be added to the software in the future to allow for this support.	A limited list of input and output peripherals are supported and selected using a GUI. Plug-ins can be written to handle additional devices.	Environments support full 3D interaction and navigation. There is no control over user interaction or navigation types.	No metric support.	Programmers, content creators
F. Custom systems	3D models are created using modeling software, VRML or X3D. Environments are deployed by running batch files, opening executables, using the command line, or a GUI.	Experiments are run using configuration files or by hard coding an experiment into a virtual environment's code. There is usually little support for experimental trials, phases, or blocks.	A VR expert or programmer configures their desired peripheral and creates a new program module to handle the device.	Interaction and navigation is decided during development time. It can be difficult reuse or add new interaction due to the custom nature of these systems.	All metrics are implemented by the VR expert or programmer who creates the system. Metrics are usually written to a text file or video file.	Programmers, VR experts
Most flexible systems	A, B, C	A, D, F	A, D, E, F	A, C, D, F	D, F	

ported? Are users required to program their own experimental constraints or is a list of available constraints provided?

- **User Interaction.** Are common navigation metaphors and interaction methods used in all environments, or do different environments have different metaphors? Are environments walk-through only, or do they support full 3D interaction?
- **Interfaces and Devices.** Which devices are available to users? Does each device need to be configured separately? Can behavioral measurement devices (e.g., EEG, fMRI, or eye trackers) be integrated?
- **Measurements.** How can users gather behavioral measurements in experiments? Is built-in recording provided or does measurement recording need to be implemented?

2.2.1 Environment Creation and Deployment. Canned systems are very effective for creating and deploying virtual environments. Several systems provide GUIs and visual programming languages to assist users in designing new virtual environments (NeuroVR, VR Worlds 2, WorldUp, etc.). New 3D models can be added to an environment from a library of models, models can be retextured, repositioned, reoriented, and rescaled, and environments can be previewed from within a GUI window. Some systems, such as NeuroVR and Virtools, have a separate virtual environment player to handle deployment, while other systems use the same interface for creating, debugging, and deploying an environment (e.g., custom VR systems such as the Mraz et al., 2003, or Maguire et al., 1998, systems, or canned systems including Presentation, VR Worlds 2, and WorldViz).

2.2.2 Environment Constraints and Behaviors. Considering that many researchers have used VR in the past, it is surprising that there is no VR support for experimental goals, trials, blocks, phases, or training and test environments. Most canned systems only support the creation of a single environment, and thus a single experimental trial. Custom solutions mainly rely on configuration files that describe each trial, or they create large environments composed of a number of smaller virtual environments, one for each trial (e.g., NeuroVR, WorldViz, VR Worlds 2, and Presentation).

2.2.3 User Interaction. The types of user interactions required for psychology and neuroscience are dictated by the paradigms that are used; hence it is common for most virtual environments to be walk-through or navigation only. In experiments involving spatial navigation or phobias, it is usually inappropriate to allow users to perform actions that cannot be performed in real-world settings (e.g., flying, teleportation, or walking through walls). Because of these restrictions, most psychology and cognitive neuroscience systems have used a common set of navigation and interaction metaphors (i.e., enabling a character to walk, run, or jump, setting up collision detection, identifying a set of objects that can be picked up or selected). Most often, the user interactions introduced into these environments are specified using visual programming languages, GUIs, or scripting languages (e.g., NeuroVR, Virtools, WorldViz, VR Worlds 2, Presentation, and custom systems). Virtools, for example, provides three layers of user support: a low level SDK supports VR experts in designing complex navigation patterns or behaviors, a midlevel scripting language can be used by VR experts and programmers to configure navigation and interaction behavior, and a high level visual programming language allows users to activate behaviors such as object avoidance or navigation.

2.2.4 Interfaces and Devices. While it can be easy to configure a keyboard and monitor for a virtual environment, it can be challenging to integrate other VR peripherals or to have more than one device available in an experiment. For custom systems, VRPN and Trackd do an excellent job of handling device integration, configuration, and data values. Unfortunately, most scientists and content creators are not able to write the configuration files needed by these or similar systems. An option-based system that could automatically configure, write, and reference each configuration file would thus be very beneficial for these users.

Lastly, there is little support for devices such as EEG,

fMRI, EMG, MEG, or eye trackers. Almost all systems compatible with these devices use custom solutions that are neither flexible nor portable (e.g., Pine et al., 2002; Mraz et al., 2003; Maguire et al., 1999). It would be beneficial to have a VRPN-type solution (possibly with a GUI interface) to integrate, activate, synchronize, and record data from these devices.

AQ: 2

2.2.5 Measurements. Measurements usually come in two types, those from external, behavioral measurement devices (such as EEG, fMRI, MEG, or EEG) and those measured by the VR system (such as time between actions, time-on-task, or input device information). While external measurements are relevant to only psychological and neuroscience experiments, the other measurements could be beneficial to areas such as gaming and entertainment (to change in-game play based upon a user's actions) or CAD type systems (to provide information about which employees worked with a given environment or to be used as a subversion-type system). Currently, VR systems do not provide suitable methods for defining and recording behavioral measurements. In almost all systems, users have to program paradigm-specific recording modules or plug-ins (e.g., ReactorMan; Kelly & Bischof, 2005; Mraz et al., 2003; Presentation; Second Life), something that most scientists and content creators do not have the skills to do.

2.3 Usability Issues

2.3.1 System Flexibility versus User Capabilities. Determining the appropriate balance between users and their skill sets is crucial when creating a successful VR system. Given the characteristics and requirements outlined in Section 2.2 and Table 1, VR systems can be compared along the dimensions of the intended user group and system flexibility, as shown in Figure 2. This comparison reveals that the systems available today are unsuitable for most scientists, content creators, and programmers.

F2

On the one hand, canned systems provide drag-and-drop or option-based interfaces to create environments, but do not allow users full environmental control. While programmers, content creators, and scientists can use these systems to easily create and deploy environments, they do so at the cost of environment control and flexibility. On the other hand, custom systems allow VR experts full access to all elements of an environment, but are too complicated for programmers, content creators, or scientists to use, not only because they require substantial programming knowledge, but also because the setup and deployment of environment and peripheral devices is complicated.

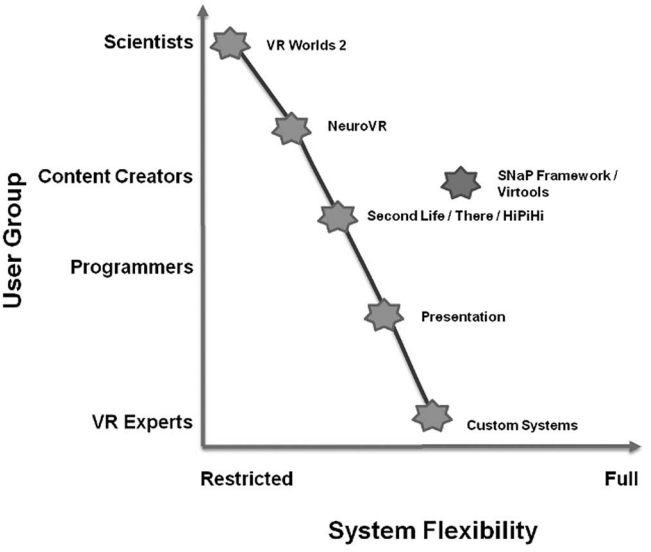

Figure 2. *A comparison between the user groups that each VR system is aimed toward and the flexibility that is inherent in each system, from very restricted to fully flexible.*

2.3.2 System Usefulness. For evaluating the efficacy and usability of software, it is common to perform a user study comparing the performance or self-reported usability beliefs of users to those reported in the literature. However, for most environments, comparisons between the behavior-based results are not valid because environments are deployed using a different system setup and peripherals, virtual environments appear differently, different navigation metaphors are used, and measurements are recorded using different methods or timing schemes. The generalization of results would be easier if paradigms were implemented using systems that employed similar navigation and interaction techniques and measurement methods.

Some comparisons are performed by asking researchers to assess the ease of developing a VR-based experiment. In most cases, a set of specific tasks is very difficult to create because all systems have different architectures and media. The best that can be done is to create a set of general, abstract tasks that are required for all virtual reality experiments. It remains to be determined which set of tasks would be fair for all participants and how task performance could be best assessed.

3 The SNaP Framework

Based on the usability issues discussed previously, we created a user-friendly, VR-based system, the SNaP framework, that is targeted toward spatial navigation research and can be used effectively by scientists, content creators, programmers, and VR experts (Annett & Bischof, 2009). The SNaP framework combines five spatial navigation paradigms, a number of VR peripherals, an easy-to-use configuration medium (XML), and a popular software development platform (Virtools) into a user-friendly VR research system. Each experiment can be quickly deployed to a variety of output contexts (e.g., monitors, HMDs, and CAVEs) and can handle a variety of input peripherals (e.g., joysticks, keyboards, mice, space mice, wands, and head trackers) using only two XML schema file parameters. A wide range of users can use the framework: it has a drag-and-drop GUI, a limited number of premade 3D models for scientists, content creators, and programmers, scripting possibilities for programmers, and a C++ SDK that can be used to implement complex logic for VR experts.

The SNaP framework helps to avoid some of the hardware issues that have plagued virtual reality environments for many years. It permits users to switch easily between paradigms and input and output devices. A single file format is used to specify an experiment, environmental components are generated in the same order, environments are visually similar, input device data is encapsulated consistently, behaviors are measured similarly, and behavioral requirements are constant across all paradigm implementations.

4 Discussion

Much work has been done to increase the usability of VR systems and virtual environments. Even with specific software solutions (e.g., Trackd and VRPN) and different software systems (e.g., Virtools, VR4Max, NeuroVR, and VRWorlds2), there are still many unsolved issues that need to be addressed.

The SNaP framework enables scientists, content creators, programmers, and VR experts to effortlessly implement and deploy virtual reality-based paradigms. Since it uses standardized, hierarchical configuration media and a universal specification schema, it is easy for a scientist to configure and specify a complete experiment. The visual programming, scripting language, and SDK provided by Virtools allow each group of users to make modifications and extensions (Annett & Bischof, 2009). VRPN, combined with Python modules, makes it easy to modulate between different input and output devices (Annett & Bischof, 2009).

One of the most important and difficult remaining usability questions concerns the optimal level of user control that future systems should possess. Even with Virtools, and all the abstracted elements and universal components of the SNaP framework, it becomes apparent that it is not possible to create a system that allows novice users to have full control over all aspects of an environment or experiment. In the short term, it may be reasonable to restrict the capabilities of users, but as users begin implementing complex paradigms or interaction metaphors; this is not a feasible solution. It appears that virtual reality technologies have a (usability) limitation boundary, as shown in Figure 3. Scientists can use a number of prebuilt VR systems, but for the most part, these systems are too restrictive. On the other hand, most custom systems are too difficult for scientists or content creators to use. Given the current state of the technology, it appears that this boundary cannot be surpassed. This is due to a number of problems, including system architectures, the integration of output peripherals and behavioral recording devices, the lack of support for experiments, and difficulties with complex user interactions.

The peripheral solutions introduced by VRPN and

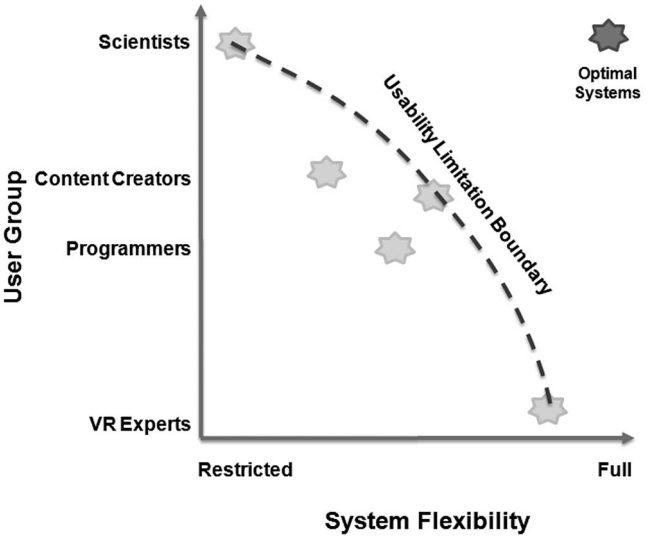

Figure 3. *A graph depicting the usability limitation boundary. This boundary indicates the points at which the usability of current VR systems is bounded. The star in the top right corner of the graph indicates where optimal or future VR systems should fall. If systems are able to reach this location, then they are assumed to be highly usable by a wide range of users, including scientists, content creators, programmers, and of course VR experts.*

Trackd have helped to solve the problems of integrating and using different input devices in VR systems. We believe that the same principles can be extended to the integration of behavioral and alternative devices such as haptic devices, game controllers (e.g., the Wiimote and Wii Balance Board), or eye trackers. While users still need to be trained to understand EEG, fMRI, and MEG devices, simplifying the synchronization and activation of such devices is a first step in the right direction. Finally, collections of add-on modules for interfacing with VRPN to record common behavioral actions, such as selecting or placing an object, button presses, joystick movements, or periods of inactivity, are beneficial and time-saving, although not every behavioral metric can or should be abstracted into an algorithm or building block.

As indicated earlier, it is difficult to compare the results obtained with different VR systems. Differences may be due to participants, tasks, virtual environment setups, or interaction metaphors. The SNaP framework creates a potential solution to this problem by implementing a wide range of VR experiments for psychology and cognitive neuroscience using the same visual appearance, interaction metaphors, and recording techniques. The SNaP framework is the only system to provide support for experimental protocols, paradigms, or behavioral measurements. Future VR systems will have to do the same to increase the usefulness and popularity of VR systems as experimental tools.

In all, the majority of VR systems today are usable by only a small group of users, namely those with programming skills, thus excluding large user groups who do not want to learn to program or do not want to spend much time creating virtual environments. The SNaP framework provides one solution, namely a system that is equally accessible to a wide range of users, from scientists to content creators to programmers to VR experts.

5 Conclusions

This work has assessed the usability of current VR systems while focusing on the requirements that scientists, content creators, programmers, and VR experts have. Most of the requirements relate to the use of VR interfaces and devices, the implementation of user interactions and behaviors, the skills required to deploy experiments, the gathering of behavioral results, and support for environmental or experimental constraints. We reviewed a number of ways in which virtual reality systems and environments are created, and we identified a number of issues that these systems handle quite well and not so well. We also provided a number of suggestions and potential avenues for future systems. We hope that, by following our suggestions, the VR systems of tomorrow will be more user-friendly than they are today. Not only will VR experts be able to design, implement, and deploy virtual environments, but so will a much wider range of users.

Acknowledgments

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

References

Annett, M., & Bischof, W. F. (2009a). The SNaP framework: A tool for assessing spatial navigation. *Proceedings of the 14th Annual CyberTherapy and CyberPsychology Conference 2009, Annual Review of Cybertherapy and Telemedicine, 7,* 61–65.

Annett, M., & Bischof, W. F. (2009b). VR for everybody: The SNaP framework. *Proceedings of 2nd Workshop on Software Engineering and Architectures for Realtime Interactive Systems (SEARIS)* (pp. 25–30). Aachen, Germany: Shaker.

Beck, L., Wolter, M., Mungard, N., Kuhlen, T., & Sturm, W. (2007). Combining virtual reality and functional magnetic resonance imaging (fMRI): Problems and solutions. *HCI and Usability for Medicine and Health Care, 4799,* 335–348.

Bischof, W. F., & Boulanger, P. (2003). Spatial navigation in virtual reality environments: An EEG analysis. *CyberPsychology & Behavior, 6*(5), 487–495.

Drettakis, G., Roussou, M., Reche, A., & Tsingos, N. (2007). Design and evaluation of a real-world virtual environment for architecture and urban planning. *Presence: Teleoperators and Virtual Environments, 16*(3), 318–332.

Hoffman, H. G., Richards, T., Coda, B., Richards, A., & Sharar, S. R. (2003). The illusion of presence in immersive virtual reality during an fMRI brain scan. *CyberPsychology & Behavior, 6*(2), 127–131.

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

LaViola, J. J., Jr., Prabhat, Forsberg, A. S., Laidlaw, D. H., & van Dam, A. (2008). Virtual reality-based interactive scientific visualization environments. *Trends in interactive visualization: State-of-the-art survey* (pp. 225–250). Berlin: Springer.

Lee, J., Lim, Y., Wiederhold, B. K., & Graham, S. J. (2005). A functional magnetic resonance imaging (fMRI) study of cue-induced smoking craving in virtual environments. *Applied Psychophysiology and Biofeedback, 30*(3), 195–204.

Lehmann, A., Vidal, M., & Bülthoff, H. H. (2008). A high-end virtual reality setup for the study of mental rotations. *Presence: Teleoperators and Virtual Environments, 17*(4), 365–375.

Maguire, E. A., Burgess, N., Donnett, J. G., Frackowiak, R. S., Frith, C. D., & O'Keefe, J. (1998). Knowing where and getting there: A human navigation network. *Science, 280*(5365), 921–924.

Mraz, R., Hong, J., Quintin, G., Staines, W. R., McIlroy, W. E., Zakzanis, K. K., et al. (2003). A platform for combining virtual reality experiments with functional magnetic resonance imaging. *CyberPsychology & Behavior, 6*(4), 359–368.

Pine, D. S., Grun, J., Maguire, E. A., Burgess, N., Zarahn, E., Koda, V., et al. (2002). Neurodevelopmental aspects of spatial navigation: A virtual reality fMRI study. *NeuroImage, 15*(2), 396–406.

Riva, G. (2005). Virtual reality in psychotherapy: Review. *CyberPsychology & Behavior, 8*(3), 220–230.

Riva, G., Alcãniz, M., Anolli, L., Bacchetta, M., Baños, R., Buselli, C., et al. (2003). The VEPSY UPDATED project: Clinical rationale and technical approach. *CyberPsychology & Behavior, 6*(4), 433–439.

Riva, G., Gaggioli, A., Villani, D., Preziosa, A., Morganti, F., Corsi, R., et al. (2007). A free, open-source virtual reality platform for the rehabilitation of cognitive and psychological disorders. *Virtual Rehabilitation,* 159–163.

Rizzo, A. A., & Kim, G. J. (2005). A SWOT analysis of the field of virtual reality rehabilitation and therapy. *Presence: Teleoperators and Virtual Environments, 14*(2), 119–146.

Watanuki, K., & Kojima, K. (2007). Virtual reality based knowledge acquisition and job training for advanced technical skills using immersive virtual environment. *Advanced Mechanical Design, Systems, and Manufacturing, 1*(1), 48–57.

Weidlich, D., Cser, L., Polzin, T., Cristiano, D., & Zickner, H. (2009). Virtual reality approaches for immersive design. *International Journal on Interactive Design and Manufacturing, 3*(2), 103–108.

Wolter, M., Armbrüster, C., Valvoda, J. T., & Kuhlen, T. (2007). High ecological validity and accurate stimulus control in VR-based psychological experiments. In *Proceedings of Eurographics Symposium on Virtual Environments/ Immersive Projection Technology Workshop,* 25–32.

AQ: 5 AQ: 6 AQ: 7