IEEE Access
The journal for rapid open access publishing

IEEE Xplore Digital Library
http://ieeexplore.ieee.org/Xplore

Brain-computer interface control in a virtual reality environment and applications for the internet of things

Christopher G. Coogan[1], and Bin He[1,2]
[1]Department of Biomedical Engineering, University of Minnesota, Minneapolis, MN 55106 USA
[2]Department of Biomedical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213 USA

***Abstract*— Brain-computer interfaces (BCIs) have enabled individuals to control devices such as spellers, robotic arms, drones, and wheelchairs, but often these BCI applications are restricted to research laboratories. With the advent of virtual reality (VR) systems and the internet of things (IoT) we can couple these technologies to offer real-time control of a user's virtual and physical environment. Likewise, BCI applications are often single-use with user's having no control outside of the restrictions placed upon the applications at the time of creation. Therefore, there is a need to create a tool that allows users the flexibility to create and modularize aspects of BCI applications for control of IoT devices and VR environments. Using a popular video game engine, Unity, and coupling it with BCI2000, we can create diverse applications that give the end-user additional autonomy during the task at hand. We demonstrate the validity of controlling a Unity-based VR environment and several commercial IoT devices via direct neural interfacing processed through BCI2000.**

***Index Terms*— Brain computer interface, virtual reality, internet of things, unity, sensorimotor rhythms**

I. Introduction

Brain-computer interfaces (BCIs) are devices that monitor and decode brain activity and create control signals to control virtual or physical objects [1]. With the advent of commercial-grade virtual reality (VR) devices, graphical processing units (GPUs), the Internet of Things (IoT), and advanced robotics, that external application can easily be a videogame, therapeutic virtual experience, or home devices such as coffee pots and televisions. Users can interact with both the physical and virtual world and to any device with access to the internet, via BCIs. BCI research and devices are commonly used in both healthy and clinical populations. While healthy individuals can utilize BCI for recreational applications, disabled individuals can use BCI for rehabilitation or assistive devices. Those individuals with a motor impairment would particularly find IoT applications useful inside their own home by being able to control their personal devices. Likewise, having BCI technology that benefits the healthy population could generate public interest in the field to advance it further. Significant advances in the underlying neurophysiological research has been attained in the past several years and we can now harness that to build complementary applications [1]-[5].

There are various methods of recording neurophysiological signals, each having their pros and cons. For healthy subjects the most often used neural signal acquisition device would be the electroencephalogram (EEG). The benefits of using EEG is that it is noninvasive, has excellent temporal resolution, and is more cost effective. On the other hand, electrocorticography (ECoG) operate very similarly to EEG except that the electrodes are placed directly onto the cortical surface. This increases the overall signal to noise (SNR) ratio and allows higher frequencies to be recorded, but also requires surgical implantation and is therefore only available to the clinical population. There are multiple categories of neurological control techniques that can be incorporated into brain computer interfaces such as sensorimotor rhythms (SMR), steady-state visual evoked potentials (SSVEP), and the P300 evoked response, all of which are easily discernable using an EEG. SMR signals are elicited through the act of motor imagination (MI). By performing an imagined act such as the opening/closing of a hand, dorsiflexion of the foot, or protrusion/retraction of the tongue we can record fluctuations in the sensorimotor cortex. In the case of MI of the hand, an event related desynchronization / synchronization can be seen on the contra/ipsilateral cortex in the frequency band of 8-13 Hz [1].

In traditional BCI research users are given a particular task utilizing one, or several, of the neurological control techniques. Several examples of BCI successfully utilizing these signals include controlling drones, robotic arms, virtual avatars, video games, and wheelchairs [7]-[13]. A critical component of the BCI feedback mechanism is the visual stimulus provided to the user to give them an indication of their progress during the task. This type of feedback presented to the user may alter their performance on a given BCI task [14], [15]. Many advances have been made from both a neuroscientific and an engineering perspective with regards to BCI, but little has been accomplished from a human-device interaction point of view. If BCI performance is task-dependent and the task presented to the user does not motivate or incentivize them to perform well, there is room for improvement [3]. A tool is needed that users can create not only devices and applications that provide therapeutic benefits and improvements in the quality of life, but applications that people are comfortably using long term.

Corresponding author: Bin He (e-mail: bhe1@andrew.cmu.edu).
This work was supported in part by the U.S. National Science Foundation under Grant DGE-1069104 and National Institutes of Health under Grant AT009263.

Digital Object Identifier: 10.1109/ACCESS.2018.2809453

By combining a game engine for content creation, various application programming interfaces (APIs) for the control of external IoT devices, and virtual/augmented reality libraries for immersive experiences, users can create the next generation of easy-to-use BCI applications for both research and personal use.

II. Materials and Methods

A. BCI2000

Signals from both EEG and ECoG devices can be recorded, analyzed, and utilized using a popular software package, BCI2000 [16]. BCI2000 allows for the acquisition of neural recordings from a wide-range of 3rd party systems, formatting and sending that raw data to pre-built or custom signal processing modules, creating a control signal based upon the signal used and the application to be sent to, and finally, the creation of an application to be presented to the end-user. Due to the modular nature of BCI2000 each module connects to each other via a network protocol. Upon initialization of each module an IP address and port number are assigned and information can be passed back and forth. Native BCI2000 use allows an experimenter to pass commands to the Operator module via command line arguments, shell scripting, batch files, or a graphical user interface (GUI). In the background, a telnet connection is created, which users utilize by writing commands directly to the operator module. These commands determine the network parameters, flags to run multiple BCI2000 instances, where to route particular signals, and specify variables to be created and triggered during particular tasks in the application layer.

To begin running an experiment in BCI2000, the user inputs parameters into the operator layer as a series of instructions. These inputs determine what type of signal acquisition device is to be used, what filtering pipeline, and what application the user will interact with, along with subject-specific parameters such as subject and task information. The source module is responsible for communicating with the acquisition hardware. The signal processing module receives the output from the source module and creates a control signal to send to the application layer based upon the filter and task.

Because of BCI2000's modularity of components, supplanting the underlying application layer only requires forwarding signals into an external application, in our case, Unity3D. While BCI2000 is a fundamental tool for BCI research, its application layer is primarily designed for experimental use in laboratory settings. As for visual feedback it's primarily suited for display on a computer monitor with simple graphics. By having 3rd party graphical software communicate and supplement the native BCI2000 modules, we gain the ability to create much more elaborate and complex BCI applications, similar to how BCPy2000 uses various python scripts to communicate with BCI2000.

B. Unity

Unity is one of the most popular game engines, with over 700 million users [17], and is used primarily to create video games for numerous platforms including mobile devices, internet browsers, PCs and Macs. Other than game creation, recently Unity is being utilized to create videos, educational material, and neuroscientific applications [18]. Besides being useful in traditional screen-based video game design, native virtual and augmented reality support have been added to communicate with various VR hardware devices. Additionally, it uses C# as a dynamic scripting language for control of its various assets. Regarding routing the signals from BCI2000 into Unity, the scripting component allows us to create user datagram protocol (UDP) network connections to read and write data streams. Likewise, we can take the signals we receive from BCI2000 and port them to other IoT devices connected to a local or remote network. As mentioned in the previous section regarding the telnet connection on the operator layer, when the Unity application is properly configured it will open BCI2000's operator module and will pass in a user-defined argument on what IP/port to connect to (Fig. 1). These connections remain open throughout the duration of the experiment and allow Unity to pass information into the operator module such as which signal source and processing modules to use, as well as various parameters. Two of BCI2000's parameters that are critical for using a 3rd party application layer are the *ConnectorInputAddress* and *ConnectorOutputAddress,* which takes as an argument an additional IP address and port number. By setting these arguments to a specific IP/port combination, reading and writing state variable values to and from BCI2000 for use by the signal processing module can be accomplished.

Unity utilizes what are called gameobjects, scenes, and components within its game engine. A gameobject can be any object (shape, model, user interface element) that holds values and components related to a scene. A scene is a collection of gameobjects that is presented to a user to interact with (Fig. 2). Components can be added to gameobjects to provide functionality, such as C# scripts, position/rotation values, or pre-built functions, to name but a few. The creation of a Unity scene can therefore have gameobjects which have C# script components that provide a network connection to BCI2000, allowing for BCI control of virtual objects.

When C# scripts are attached to interactable elements, such as buttons, additional functionality is provided. For instance, when a button in the GUI is referenced by a script, logic-based functions can be enacted, such as opening a new scene, or beginning a BCI-related task. When the corresponding dropdown menus and buttons are interacted with in the GUI (Fig. 3), BCI2000 launches with its respective parameter files. After BCI2000 is configured, a new scene opens based upon the 'Signal Application" chosen from the GUI. If, for instance, a 1-dimensional (1D) or 2-dimensional (2D) motor imagery based scene was selected, game objects and scripts associated with that type of task will be presented. This includes configuration files to load the correct state variables for BCI2000 for this type of task, targets and a cursor to provide feedback to the user on the MI task, as well as a virtual camera for the user to view the scene and to move around within it. To customize the appearance of the scene the user can create their own gameobjects or models using software packages such as Blender or Maya. Additionally, numerous

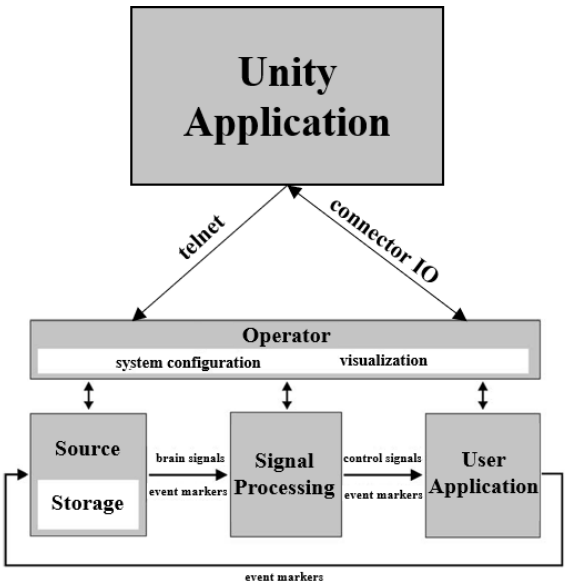

Fig. 1. Unity/BCI2000 integration. Adapted from [16].

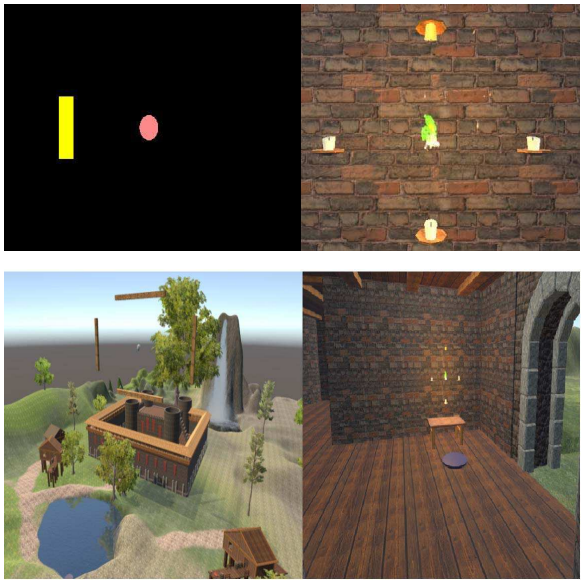

Fig. 2. (Top) Side by side comparison of BCI2000 native application and Unity application layer, 2D perspective. (Bottom) Different views of a MI task taking place in Unity.

types of assets can be included in the scene such as trees, buildings, etc, to provide a more immersive experience, and many of these are free to use and available in the Unity Asset Store.

C. Implementation

Like BCI2000, the user can select which core modules and what parameter files to use, or they can select a preconfigured batch file. These commands are all written to BCI2000's operator layer as would be the case if using BCI2000's native GUI. Integrating communication from Unity to BCI2000's operator layer was done so that the user did not need to switch between the two software packages when configuring

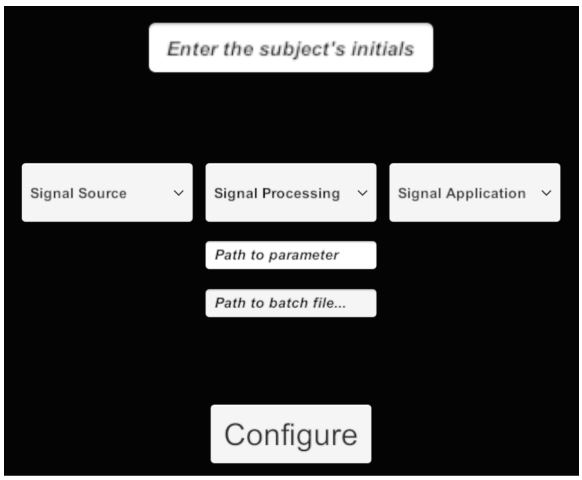

Fig. 3. Unity-based GUI to communicate with BCI2000

the operator and application layer. Therefore BCI2000 simply runs in the background with the user communicating with it via Unity. When the user presses the "Configure" button the commands are routed to BCI2000, BCI2000 loads the selected configuration, a new scene is loaded in Unity where the experiment takes place, and control is handed to the end-user. During this step BCI2000 is put into a resting state, ready to receive additional parameters. If the user selects a signal application template from the GUI (such as the VR_LR_Candle scene shown in Fig. 2) then the respective scene will load, and the experiment will begin whenever the user is ready.

Capability for the end-user to select when the application begins is enabled via keyboard or game-controller input. After the scene is loaded the user will be able to move around the environment at their pace. If the user is wearing a supported virtual reality device this will be by the corresponding controllers (Fig. 4). If using the computer monitor for visual feedback, the user can move around via keyboard control or gamepad. This type of experimental setup is rare, if not unique, because the user is no longer presented with a static screen in with which to interact, but with a virtual environment that may contain elements not directly corresponding to the BCI task at hand.

For a simple motor imagery-based application we use the *ConnectorOutputAddress* to write the variables "TargetCode", "ResultCode", and "Feedback." By then setting the signal processing layer to look for these state conditions, we can use the *ConnectorInputAddress* to extract information pertaining to target hits, misses, aborts, and cursor position. This type of information is all that is required from BCI2000 to create any type of BCI-based application in Unity. By routing the signals from BCI2000 into Unity we can perform a one-to-one mapping between the BCI2000 control signal and the Unity cursor position.

D. Features

This software package was designed (but not limited) to be utilized in two ways. As all scenes, gameobjects, scripts,

Fig. 4. User controlling a Unity-based BCI application in VR.

and other assets can be saved as assets of a project, assets can be created and shared amongst groups and individuals to be used as open-source templates. This is fairly common in open-source projects with online repositories of Unity-based projects. Users are encouraged to make forks of these projects and make additions and customized elements. For instance, a basic SMR-based BCI task may include a cursor, two targets, and a script that communicates with the core BCI2000 software. If a user makes an addition to a particular template, or customizes the appearance of the targets/cursor, it can be repackaged and shared amongst others. Secondly, if a user just wishes to use the provided templates, little additional configuration is required. By providing templates and open-source licensing, users will ideally customize and create variations of their own BCI scenes.

To make this software applicable to as broad an audience as possible, support for VR and several IoT devices has been included as templates. These include the Leap Motion packages for hand tracking in Unity, Philips Hue API and TPLink to control supported lightbulbs/outlets, as well as the Roku API to control smart televisions. As long as a commercial IoT device has a publicly available API, it can be integrated within Unity using it's C# scripts and asynchronous calls. Currently the VR plugin used is SteamVR, which allows this application to run using the HTC Vive, Oculus Rift, and Windows Mixed Reality devices. By not focusing on a single type of application or a particular physical device, and offering basic support for multiple devices, it is hoped that more individuals build upon and share their contributions. This useful so others can easily reproduce experiments, as well expand upon them.

1) Templates: For user's that simply want to use traditional BCI2000 tasks but with Unity's feedback stimuli, several templates have been included. These operate identically to traditional BCI2000 applications, the only difference being the user physically looks at Unity instead of BCI2000. Currently these templates include an SMR-based left vs right paradigm, SMR-based up vs down paradigm, a SMR-based two-dimensional (up vs down vs left vs right) paradigm, and a P300-based speller. Furthermore, within the SMR-based paradigms, output can be routed to IoT devices and/or a VR headset. With a little knowledge of using Unity to add elements to a scene, visually aesthetic scenes can be overlaid on top of the core BCI scene.

2) IoT: With the advent of the Internet of Things, various home devices can be connected to the internet and controlled via a set of API calls provided by the manufacturer. These devices range anywhere from toasters and lightbulbs to televisions and thermostats. By connecting to a network the IoT device is assigned an IP address or given a token that allows users to interact with it via the manufacturer's official application (typically via smartphone applications), or a custom designed application. This is done by accessing the devices API. Users can interact with the API using hypertext transfer protocol (HTTP) requests to read and write data, typically provided in a JavaScript Object Notation (JSON) payload. For instance, if there are several IoT lightbulbs connected a home network a user can choose which one to connect to, turn it on and off, and set the brightness/hue values. For individuals with physical disabilities ranging from mild to severe, tasks which may appear simple to healthy individuals provide insurmountable obstacle towards independence and these types of services can be largely beneficial. What may seem like a small act to most people, turning a light on or adjusting the temperature, could make a world of difference for someone who only has the ability to type a command on their phone. For those that can't accomplish even that task, however, brain computer interfacing of things (BCIOT), no matter how simple, could make a huge impact, improving the quality of life and providing further independence from caregivers.

Three IoT devices have been integrated into this software package. Roku, for smart TV control, TP Link for lights/outlets, and the Philips Hue lightbulbs. As network security is a major concern for IoT devices, the IP/token parameters can, and must, be set by the individual administrators of their personal network. In order to control the IoT devices from within Unity the user opens the "IoT" scene provided as a template. After toggling a virtual switch to indicate which type of device is to be controlled, the user inputs the corresponding IP address/token, and the device is found on the network and automatically connects.

All testing of BCI controlled IoT devices was completed offline using a subset of real-time SMR-based BCI data collected during the experiment protocol outlined in section II. To simulate simple, asynchronous, control of IoT devices a one to one mapping was done between the control signal utilized in the up/down task of the VR study. To move the cursor up the user imagined opening/closing both hands simultaneously, and in order to move the cursor down the user was in a rested

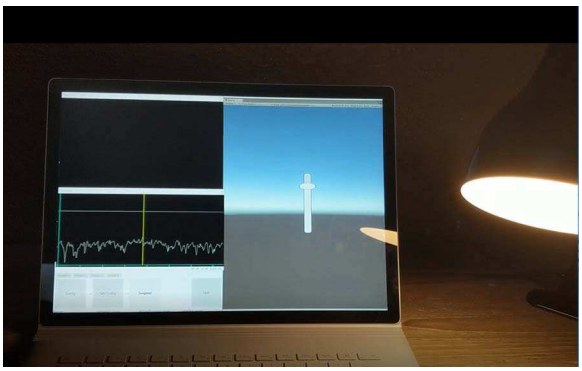

Fig. 5. BCI controlled lightbulb (right) via Philips Hue interface with Unity routing the signals over the network.

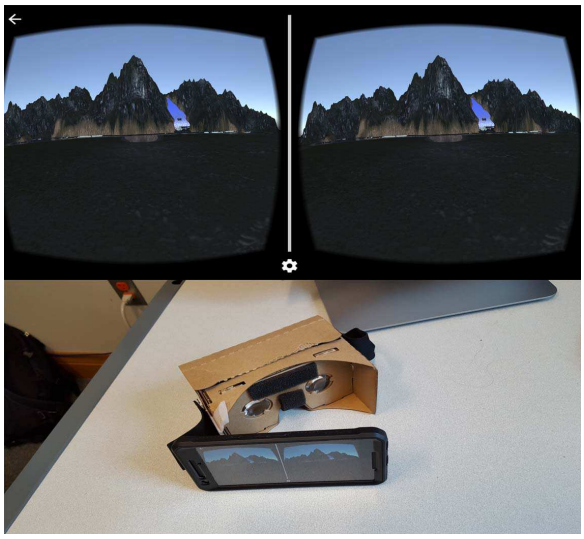

Fig. 6. Mobile-based SMR BCI application built in Unity to be used as a Google Cardboard application.

state, not imagining their hands moving at all. For control of a coffeepot via the TP Link smart switch a threshold was experimentally determined and if the user performed bilateral motor imagery of the hands the coffeepot would turn on, and if they were in a rested state it would turn off. Likewise, for control of a lightbulb via the Philips Hue bridge there was a mapping between cursor position and the light's brightness intensity (Fig. 5). The more the user performed bilateral motor imagery the brighter the light became. Lastly, for control of the volume using a TCL Roku-enabled smart TV there was a mapping between cursor control and volume. If the cursor position was greater than the value in the previous time point, the volume increased, if the cursor began to move down, the volume decreased. Other Unity gameobjects can be introducted into the scene such as buttons to provide on/off functionality to compliment the asynchronous control of the BCIOT device.

3) Mobile-use: In a traditional BCI experiment the computer serves two purposes: 1) processing and saving the user's data by physically interfacing with the neural acquisition hardware over a series of cables and 2) presenting visual stimuli via the computer monitor. With the advent of mobile EEG devices with onboard amplifiers that communicate to the computer via Bluetooth or Wi-Fi we can remove the first case [21]. By presenting the visual stimuli on a similar wireless device, such as a smartphone, we can achieve a purely mobile-based BCI application. Fig. 6 shows a mobile based BCI application with the option to use a VR headset to compliment it. It runs no differently than a computer-based BCI task except the visual stimuli is a Unity-based application built for a mobile device. The quality of the graphics must also be reduced due to the processing constraints of the mobile device.

4) Virtual Reality: Unity's contribution as an application layer are evident with the inclusion for support of virtual/augmented reality libraries. Specifically, the SteamVR library for Unity allows for communication with multiple VR headsets. If the intended BCI application utilizes VR, the game camera will change various settings and allow the user to navigate the space with a virtual reality controller. In an immersive VR application, users have shown increased levels of embodiment [22]. Coupling this type of interaction with a BCI could grant the user a higher level of control. In addition to the added level of embodiment, cognitive-control applications that are gamified [23] have the potential to improve attention and multitasking [24]. With the inclusion of the Leap Motion library for Unity user's can see a virtual representation of their physical hands. While this is useful for MI-based experiments, it can also be useful for psychological experiments akin to mirror treatments for phantom limb patients [25].

III. Experimental Evaluation

To validate that the Unity-based BCI2000 application layer would provide adequate feedback for a user to control aspects of a virtual environment, 31 subjects were recruited to test and validate the implementation of a virtual reality based, SMR-controlled, BCI experiment under a protocol approved by the IRB of the University of Minnesota. The participant population was split into two groups: those who experience only a single session, and those exposed to multiple sessions. The first study compared the performance of users familiar with SMR-based BCI and users that were naïve in both a VR environment and a traditional SMR-based BCI task in a single session. The second study compared the learning rates of naïve subjects in a control group (traditional SMR-based BCI) and a VR group throughout 6 sessions.

The feedback hardware used in this study was the HTC Vive for the VR group and a 15-inch computer monitor for the control group. During the VR portion of the experiment users donned the VR headset (Fig. 4), did a quick calibration of the lenses, and then spent 5 minutes traversing the virtual environment via the tracked controllers to become accustomed to the experience and to determine whether they would have any adverse reactions to the visual stimuli. When the users showed an adequate ability in traversing the 3D space and showed no signs of motion sickness, they were instructed to move into the experiment area (Fig 2, bottom). In order to give the subjects a level of autonomy not normally seen in BCI

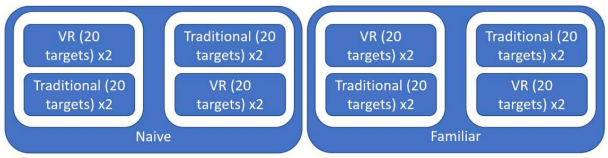

Fig. 7. Study 1 pipeline.

experiments, they were given control of how to position their virtual avatars which allowed them to view the motor imagery (MI) task from whatever perspective they chose (some chose to be less than 1 meter from the task while others chose to be several meters away). They were also given the ability to begin the experiment whenever they felt comfortable (instead of the experimenter dictating when the experiment would begin) by squeezing a button on the tracked controller, and then placing the controller on the table in front of them. The control group simply indicated whenever they were ready to begin. The metric of performance utilized in this study was the percent valid correct (PVC) which is the number of hits divided by the total number of valid (non-timed out) attempts.

IV. Data Collection

All participants were seated comfortably in front of a computer monitor, regardless of whether their group utilized it, for all sessions. All data was collected with a 64 channel EEG cap using a Neuroscan Synamps 2 amplifier, at a sampling rate of 1000Hz and filtered from 0.1-30Hz. A Laplacian spatial filter was used centered around electrodes C3 and C4. The Laplacian filter takes as input 10 electrodes, (C3/C4 and the 4 electrodes directly neighboring them) and sets a weight of 1 to C3/C3 and .25 to all others. All signal acquisition and processing were identical in the two studies. For these motor imagery-based tasks, a target was presented on the screen/headset (as seen in Fig. 2, top) and it was the participants' goal to imagine opening and closing their respective hand(s) to move the cursor.

A. Single Session

In the first study (Fig. 7) 22 subjects were recruited, 15 having already participated in at least one BCI experiment, and 7 naive subjects. This study consisted of a single session primarily designed to determine whether or not there was a significant difference between the type of visual stimuli in non-naïve subjects. Each subject performed 2 2D runs (20 targets/run that were randomly distributed between top, down, left, and right targets, with roughly 3 minutes per run) while receiving the visual stimulus in either VR or via the computer monitor. The subject had a 6 second period where the target was presented. If no target was hit during this period, the trial was aborted; if a target was hit, the trial would end upon the successful hit. Regardless of a hit, miss, or aborted trial, the participant was then given a 2 second inter-trial rest period before the next target was presented.

After the 2 initial runs, subjects completed an additional 2 runs in the alternate paradigm (switch between VR and computer monitor). Subjects were then given a 5-minute rest

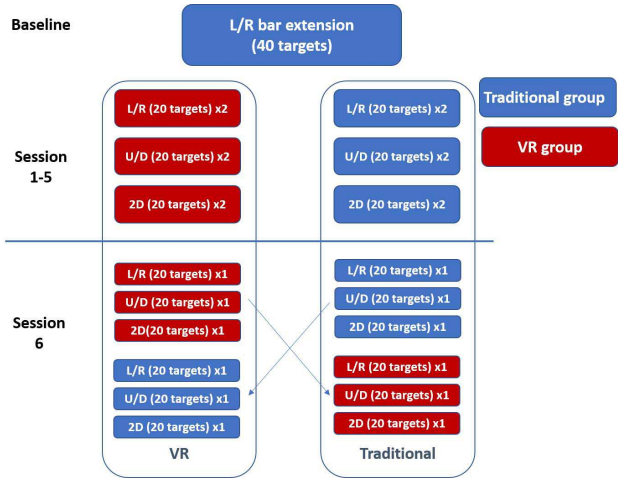

Fig. 8. Study 2 pipeline.

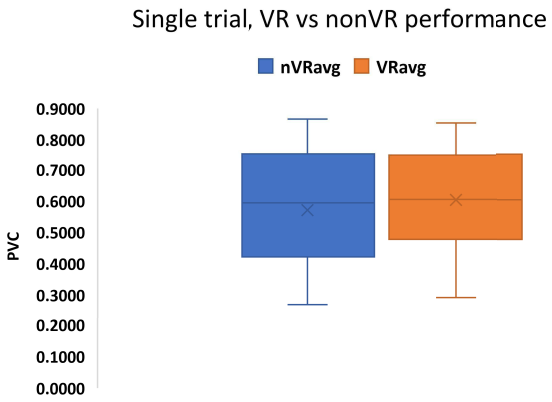

Fig. 9. Study 1: No change in performance between VR and control group.

period prior to repeating both tasks in the previous order (if they started with a VR stimulus in the first phase, they again started with a VR stimulus in the second phase). The starting stimuli (VR or traditional) was randomized so that each group had an equal distribution of individuals who began with both the VR and computer monitor stimuli. In each group the filtering pipeline was identical, with the only difference being the visual stimuli that was presented.

B. Learning effects

In the next study (Fig. 8) 9 naive subjects were recruited and participated in 6 sessions each. This study was designed to test the performance/learning rates between naïve subjects receiving a visual stimulus of targets presented via the computer monitor or VR headset. The first task presented to the users was a motor-imagery feedback task. Users were presented with a black screen on the computer monitor and asked either to physically open/close their respective hands or to imagine the act. This was to measure their baseline EEG activity prior to the BCI task.

In the second study, each session consisted of 3 left vs right runs (20 targets per run, roughly 3 minutes per run) followed by 3 up versus down runs, and finally 3 2D runs. After this

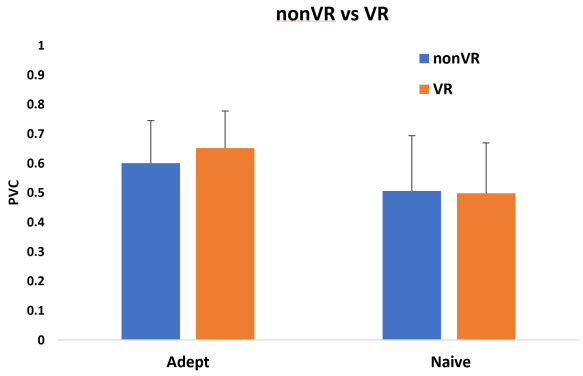

Fig. 10. Study 1: Small increase in performance in adept subjects switching from a control stimulus to a VR stimulus.

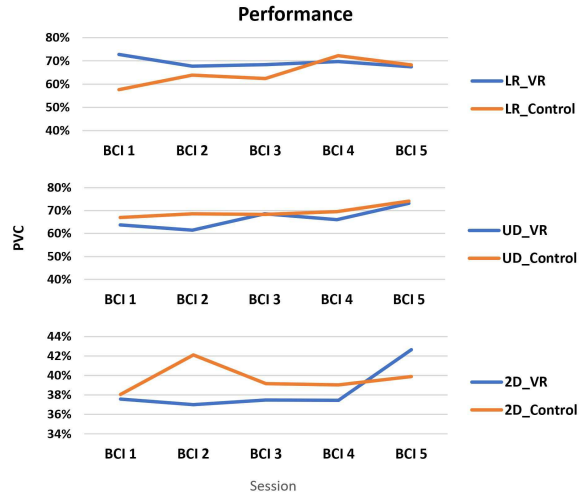

Fig. 11. Study 2: No significant changes in performance between groups.

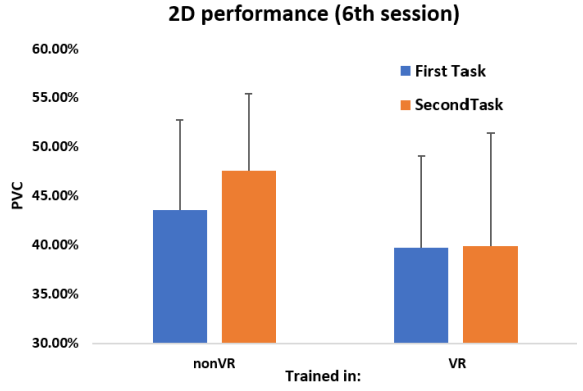

Fig. 12. Study 2: Small increase in performance in adept subjects switching from a control stimulus to a VR stimulus.

initial phase the subjects were given a 5-minute rest period before repeating the task a second time. This pipeline was used for the first 5 of 6 sessions (Fig. 8). When the participants returned for the 6th and final session they were once again presented with an identical protocol to the first 5 sessions, but after the 5-minute rest period and completion of the first half of the session, they switched to the alternate paradigm. If during the first 5 sessions the participants were presented with targets via a computer monitor, for the second half of the 6th session they would be presented with targets via a VR headset. The opposite was true of the other group (if original stimuli was VR, novel stimulus was provided via the computer monitor). This test was designed to assume that after 5 sessions of a MI-based BCI task, all participants were relatively proficient in the task. With a novel stimulus introduced it would show whether or not either stimulus had any affect over the other.

V. Results

As can be seen in the figures below, performing MI-based BCI tasks in VR does not affect one's ability to perform the task. Across both studies and 31 subjects assessed, mean differences were not seen between groups, indicating 1) signal transmission from BCI2000 to Unity and back again did not introduce any noticeable latency, and 2) that users performed no worse when in an immersive, virtual reality, BCI experiment, indicating that that the immersion effect of a virtual environment does not impede performance.

While not statistically significant, an interesting trend that may offer additional information is in Fig. 10. There appears to be an increase in performance while switching from the control stimulus to the VR stimulus in the group that has already had some experience with the SMR-based BCI task in the past. This can be compared with Fig. 12 in study 2. Once again, switching from the control stimulus (of which this group was trained in for 5.5 sessions) to the VR stimulus shows a similar increase in performance, not seen in both study 1 and study 2's participants going from a VR to a control stimulus. While not significant, it appears to be consistent throughout both studies.

Additionally, there was no significant difference in between groups switching from a VR stimulus to a control stimulus, further indicating the visual stimuli alone does not impede performance.

VI. Discussion

With the pace of technological advancement exponentially increasing, amazing strides in virtual/augmented reality and the internet of things have been seen in the past several years. Just as advances in these fields required prerequisite technological improvements in network infrastructure and high-end consumer graphics, so to does the continued improvement of BCI use rely on a prerequisite set of technologies. While research-grade neural acquisition devices have enabled a vast trove of reliable datasets to be analyzed, consumer-grade EEG/BCI devices are just now beginning to enter the market. Using commercial-grade virtual and augmented reality devices with applications created with game engines such as Unity and Unreal Engine could bridge the gap between research and personal use of these devices.

While this application layer has been built for use with BCI2000, there are no technical limitations as to why it could not be used in conjunction with other BCI systems (OpenVibe, OpenBCI, etc) since the reading and writing of data is done using simple TCP/UDP communication. Likewise, example scenes include support for the Philips Hue and Roku API,

but there are no technical limitations to expand beyond this as they only require HTTP requests.

The accompanying software and documentation can be found at, *http://www.github.com/bfinl/UnityBCI,* (http://doi.org/10.5281/zenodo.1181738) under a MIT License. Users are encouraged to download, distribute, and make contributions. While the current state of this release does require a fair amount of manual tuning to begin creating virtual scenes to integrate with BCI2000, it does support all acquisition and processing modules and includes a set of core BCI2000 modules. Due to the open source nature of this application a user can fork the code, create an application, push the updated code, and grant other users access to it. Because Unity can run in Editor-mode, by importing the newly created assets from a different user, no compilation is necessary, so experiment/scenes can be shared and used instantly. It is hoped that by offering an easy-to-use tool to create BCI applications, the field will grow faster.

In the source code several different folders can be found as is standard with Unity projects. The 'scenes' folder will include several different templates such as IoT or VR-based BCI paradigms. Under the 'scripts' folder numerous C# scripts will be found. These scripts are what creates the networking and communication between both BCI2000 and Unity, as well as Unity to the IoT devices. There is a core script 'BCI_Class.cs' that works behind the scene in configuring the networking protocol, with other scripts such as 'BCI_SMR.cs' that configure the specific BCI task. Documentation for these assets are included in the repositories Readme. In addition several videos and images will be provided as supplementary material as a detailed tutorial.

While these newer technologies could provide freedom to explore advances in BCI research, several issues may prevent it from becoming wholly adopted. For instance, VR use greatly expands on the level of embodiment a user feels during their session, however it yet to be seen whether this embodiment will be more motivation, or distracting. Similarly, while the majority of subjects reported no sense of motion sickness or vision impairments during prolong VR use, one subject was excluded from the study due to discomfort in the first session. Because both a VR headset and EEG electrodes are placed directly on top of the head, these two can physically interfere with each other. As can be seen in Fig. 4, the HTC Vive's straps directly press on electrodes C3/C4, the primary electrodes used in SMR-based BCI control. Lastly, the VR device adds an additional weight upon the user's head and may cause fatigue not seen in other groups. All user's in these studies were given a 5-minute break halfway through the session, regardless of group, to minimize this external factor.

This work primarily focused on taking a traditional 2D BCI task presented on a computer monitor and recreating it in Unity3D. From the user's perspective the only change is what they physically looked at and what they were able to interact with. This, however, should only be the first stage of using a tool such as Unity. Because VR allows the exploration of a 3D environment, further research into utilizing a 3D BCI task should be explored.

VII. Conclusion

Using widely available software packages and network calls it is possible to route BCI control signals into various applications such as game engines, virtual reality devices, and personal home devices. These devices can be controlled, and environments traversed, using a combination of BCI control and physical input. This greatly expands the potential for creating complex BCI applications and creating a standard to communicate with virtual and external objects/devices. While user performance did not increase during the immersive VR task, users within the group showed a comparable learning rate relative to their counterparts in the control group, indicating that there are no detrimental effects due to latency of network transmission or cognitive aspects such as motivation or distraction.

The work presented here describes the creation, implementation, and evaluation of a Unity game engine-based application layer for BCI2000. Supplanting the native application layer allows for the integration of advanced 3D graphics, VR, and IoT devices. This software package includes several template scenes for rapidly implementing various BCI paradigms along with their use in VR or IoT applications. If user's wish to build there own BCI applications, or build upon published templates, an easy interface allows them the freedom to edit each individual component from the visual stimuli to the control signals of interest.

References

[1] He B, Gao S, Yuan H, Wolpaw J. "Brain-computer interface," pp: 87-152, In (He B, ed.): *Neural Engineering, 2nd edition*. Springer, 2013.

[2] Millán, J. D. R., et al. "Combining brain–computer interfaces and assistive technologies: state-of-the-art and challenges." *Frontiers in neuroscience* vol. 4, no. 161, Mar. 2010, DOI: 10.1109/ICCME.2012.6275712.

[3] Leeb, Robert, et al. "Combining BCI and Virtual Reality: Scouting Virtual Worlds" in *Toward brain-computer interfacing,* 1st ed., Springer-Verlag Berlin Heidelberg 2007, pp. 197-220.

[4] Faller, Josef, et al. "An application framework for controlling an avatar in a desktop-based virtual environment via a software SSVEP brain–computer interface." *Presence: teleoperators and virtual environments* vol. 19, no. 1, pp. 25-34, Jan. 2010, DOI: https://doi.org/10.1162/pres.19.1.25 .

[5] Yuan, H., and B. He. "Brain Computer Interfaces Using Sensorimotor Rhythms: Current State and Future Perspectives." *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 5, May 2014, pp. 1425–35. *IEEE Xplore*, doi:10.1109/TBME.2014.2312397.

[6] Machado, Sergio, et al. "EEG-based brain-computer interfaces: an overview of basic concepts and clinical applications in neurorehabilitation." *Reviews in the neurosciences* vol. 21, no. 6, pp. 451-468, Jan. 2010, DOI: 10.1515/REVNEURO.2010.21.6.451.

[7] Leeb, Robert, et al. "Self-paced (asynchronous) BCI control of a wheelchair in virtual environments: a case study with a tetraplegic." *Computational intelligence and neuroscience* vol. 2007, no. 7, Apr. 2007, pp. 1-8, DOI: 10.1155/2007/79642

[8] Yuan, H., et al. "Cortical Imaging of Event-Related (de)Synchronization During Online Control of Brain-Computer Interface Using Minimum-Norm Estimates in Frequency Domain." *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 16, no. 5, Oct. 2008, pp. 425–31. *IEEE Xplore*, doi:10.1109/TNSRE.2008.2003384.

[9] Royer, A. S., et al. "EEG Control of a Virtual Helicopter in 3-Dimensional Space Using Intelligent Control Strategies." *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 18, no. 6, Dec. 2010, pp. 581–89. *IEEE Xplore*, doi:10.1109/TNSRE.2010.2077654.

[10] Meng, Jianjun, et al. "Noninvasive Electroencephalogram Based Control of a Robotic Arm for Reach and Grasp Tasks." *Scientific Reports*, vol. 6, Dec. 2016. *PubMed Central*, doi:10.1038/srep38565.

[11] Edelman, B. J., et al. “EEG Source Imaging Enhances the Decoding of Complex Right-Hand Motor Imagery Tasks.” *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 1, Jan. 2016, pp. 4–14. *IEEE Xplore*, DOI: 10.1109/TBME.2015.2467312.
[12] He, B., et al. “Noninvasive Brain-Computer Interfaces Based on Sensorimotor Rhythms.” *Proceedings of the IEEE*, vol. 103, no. 6, Jun. 2015, pp. 907–25. *IEEE Xplore*, doi:10.1109/JPROC.2015.2407272.
[13] Lécuyer, Anatole, et al. "Brain-computer interfaces, virtual reality, and videogames." *Computer,* vol. 41, no. 10, Oct. 2008, pp. 66-72. *IEEE Xplore*, doi: 10.1109/MC.2008.410.
[14] Ron-Angevin, Ricardo, and Antonio Díaz-Estrella. "Brain–computer interface: Changes in performance using virtual reality techniques." *Neuroscience letters* vol. 449, no. 2, Jan. 2009, pp. 123-127, https://doi.org/10.1016/j.neulet.2008.10.099.
[15] Wu, Hong, et al. "Evaluation of Motor Training Performance in 3D Virtual Environment via Combining Brain-computer Interface and Haptic Feedback." *Procedia Computer Science* vol. 107, Mar. 2017, pp. 256-261, DOI: https://doi.org/10.1016/j.procs.2017.03.096.
[16] Schalk, Gerwin, et al. "BCI2000: a general-purpose brain-computer interface (BCI) system." *IEEE Transactions on biomedical engineering* vol. 51, no. 6, Jun. 2004, pp. 1034-1043, DOI: 10.1109/TBME.2004.827072.
[17] Unity3D. https://unity3d.com/public-relations
[18] Mullen, T., Kothe, C. Chi, Y.M., Ojeda, A., Kerth, T., Makeig, S., Cauwenberghs, G., Jung, T-P. July 2013 Real-Time Modeling and 3D Visualization of Source Dynamics and Connectivity Using Wearable EEG. Presented at 35th Annual International Conference of the IEEE Engineering in Biology and Medicine Society, doi: DOI: 10.1109/EMBC.2013.6609968
[19] Edlinger, Günter, Gunther Krausz, and Christoph Guger. July 2012, A dry electrode concept for SMR, P300 and SSVEP based BCIs. Presented at Complex Medical Engineering.
[20] Lalor, Edmund C., et al. "Steady-state VEP-based brain-computer interface control in an immersive 3D gaming environment." *EURASIP journal on applied signal processing* vol. 2005, no. 19, Dec. 2005, pp. 3156-3164, doi: 10.1155/ASP.2005.3156.
[21] Navarro, Karla Felix. Apr. 2004 Wearable, wireless brain computer interfaces in augmented reality environments. Presented at *Information Technology: Coding and Computing.*
[22] Kilteni, Konstantina, Raphaela Groten, and Mel Slater. "The sense of embodiment in virtual reality." *Presence: Teleoperators and Virtual Environments* vol. 21, no. 4, Nov. 2012, pp. 373-387, doi: DOI: 10.1162/PRES_a_00124
[23] Bayliss, Jessica D., and Dana H. Ballard. "A virtual reality testbed for brain-computer interface research." *IEEE Transactions on Rehabilitation Engineering* vol. 8, no. 2, Jun. 2000, pp. 188-190, doi: 10.1109/86.847811
[24] Anguera, Joaquin A., et al. "Video game training enhances cognitive control in older adults." *Nature* vol. 501, no. 7465, Sep. 2013 pp. 97-101, doi:10.1038/nature12486.
[25] Chan, Brenda L., et al. "Mirror therapy for phantom limb pain." *New England Journal of Medicine* vol. 357, no. 21 Nov. 2007, pp. 2206-2207, doi: 10.3344/kjp.2012.25.4.272.

Christopher Coogan received the B.S. degree in biomedical engineering from the State University of New York at Binghamton in 2016 and the M.S. degree in biomedical engineering from the University of Minnesota, Twin Cities in 2018. He is currently a software developer at Johns Hopkins University. His research interests include brain computer interfaces, human-machine design, and virtual reality.

Bin He is Professor and Head of the Department of Biomedical Engineering at Carnegie Mellon University (CMU), Pittsburgh. He also holds joint faculty appointments in the Department of Electrical and Computer Engineering and Center for Neural Basis of Cognition at CMU. Dr. He has made significant contributions to brain-computer interface technology and biomedical imaging, including electrophysiological source imaging and tissue electrical property imaging. Dr. He has received a number of recognitions, including the IEEE Biomedical Engineering Award, the Academic Career Achievement Award and Distinguished Service Award from the IEEE Engineering in Medicine and Biology Society, the Established Investigator Award from the American Heart Association, and the CARRER Award from the National Science Foundation, among others. He was President of the IEEE Engineering in Medicine and Biology Society and Chair of Publications Committee of the American Institute of Medical and Biological Engineering, and is Chair-Elect of the International Academy of Medical and Biological Engineering. Dr. He is an elected Fellow of International Academy of Medical and Biological Engineering, IEEE, American Institute of Medical and Biological Engineering, Biomedical Engineering Society, and is the Editor-in-Chief of IEEE Transactions on Biomedical Engineering.