OVR Stylus: Designing Pen-Based 3D Input Devices for Virtual Reality

Bret Jackson*

Macalester College
Saint Paul, Minnesota, USA

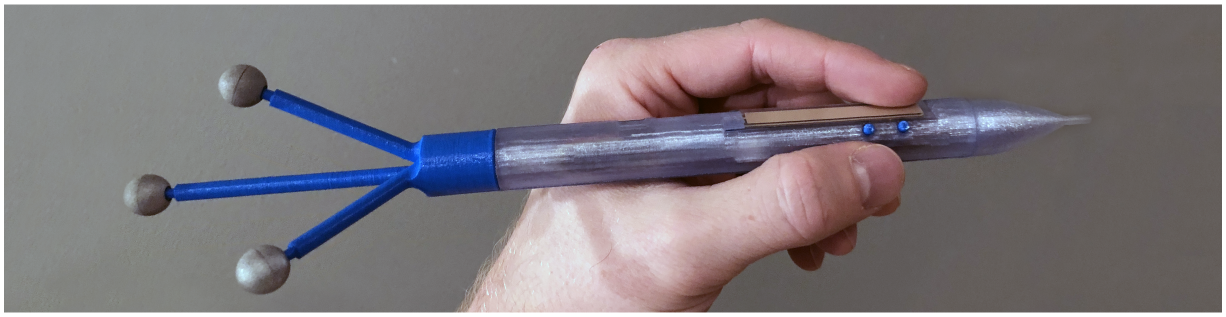

Figure 1: The OVR Stylus is a pen-shaped 3D input device for VR/AR environments. Wireless communication transmits button clicks and values from a sliding touch pad. A haptic motor provides vibrotactile feedback.

Abstract

We present the OVR Stylus, an open-source, tangible, haptic prop for 3D input in virtual environments. The design offers an alternative to commercial 3D pen input devices, and represents a second-generation hardware design based on six years of use with a previous prototype. With a touch pad, two input buttons, LRA-based vibrotactile haptic feedback, and Bluetooth communication, the 3D-printed prop's light-weight design (35g) enables creative and precise interaction. Through a discussion of design considerations and motivations, components, manufacturing processes, software, and lessons learned we enable others to create their own OVR Stylus or develop similar designs.

Index Terms: Human-centered computing—Human computer interaction—Interaction devices—Pointing devices; Human-centered computing—Human computer interaction—Interaction paradigms—Virtual reality

1 Introduction

In virtual reality (VR) and augmented reality (AR) users typically interact with a hand-held device (e.g. HTC Vive Controllers) to perform common operations like selection, manipulation, and system control. Driven primarily by the gaming industry, these devices are commonly held using a power grip, similar to how a user might hold a handle or lift a coffee mug. For many games, this type of grip and controller makes sense. It closely mimics how someone might hold a tool, and predominantly uses the force of the hand rather than the strength of individual fingers.

However, holding a controller using a power grip limits precision [9]. Indeed, recent work by Pham and Stuerzlinger [28] found that in a Fitts' Law task, a pen-shaped prop using a precision grip outperformed a standard VR controller in pointing speed, mouse control, errors, and throughput; it reached comparable input to a mouse. It is no surprise that with increasing use of VR/AR technologies for consumer applications beyond gaming, commercial companies are developing pen-shaped 3D input devices. Examples include the Logitech VR Ink [22], Massless Pen [24], and Holo-Stylus [16].

In this paper, we present the OVR Stylus, a pen-shaped, tangible prop for 3D input. The stylus features two buttons and a touch pad slider for input that communicate wirelessly with a host computer. A linear resonant actuator (LRA) motor enables vibrotactile haptic feedback. The design presented here is a second-generation prototype, refining an initial model that has been used in our lab with VR applications for the last six years.

Although pen-shaped 3D input devices are not in themselves novel, to our knowledge, no other fully-realized, open source implementation exists using modern hardware. Our primary contribution is the hardware and software implementation, which other researchers can use to produce their own OVR Stylus. Furthermore, the hardware design can serve as a building block for modifications to create other formats of novel 3D input devices. In addition, we provide design criteria, discussion, and lessons learned to guide this future work.

2 Related Work

Many stylus input devices have been created, but they are used predominantly for surface computing or touch displays (e.g Wacom Pen [15], Elastylus [23], Haptylus [27], ImpAct [35]). Here, we limit our discussion to those relevant to mid-air 3D input for use with VR or AR displays.

Notable examples include work by Teather and Stuerzlinger [33] and Pham and Stuerzlinger [28] that uses an un-powered pen-like device fitted with optical tracking markers to explore 3D pointing. Similarly, Brown et al. [8] use a chopstick tracked with a Leap Motion [34] for pointing. SymbiosisSketch [3] uses a traditional writing pen fitted with motion capture markers to support mid-air sketching. Each of these examples demonstrates the utility of pen-shaped input devices for VR, but their focus is on studying 3D pointing performance or application design rather than creating a functional input device for everyday use. With the exception of Pham and Stuerzlinger [28], which includes a short description of the VR/AR pen design space, these references provide further motivation for the need for pen-shaped VR input devices, but little in the way of concrete designs.

Most existing functional 3D stylus designs are tethered with a cord to a host computer to facilitate communication of button presses

*e-mail: bjackson@macalester.edu

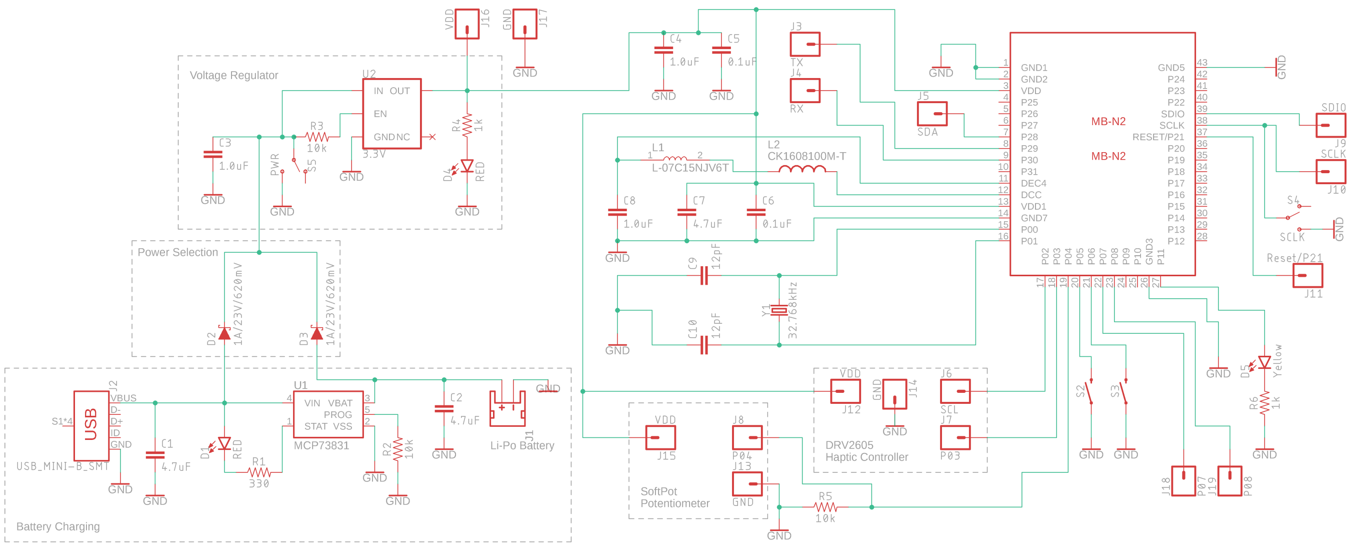

Figure 2: OVR Stylus circuit schematic. The hardware contains a battery charging circuit, variable selection of the power source, and a voltage regulator to output 3.3v power. The Redbear MB-N2 module houses a Nordic nRF52832 BLE chip and antenna.

and provide power. Examples include the Fasttrak Polyhemus Digitizer [11], and the zSpace stylus [36]. The Tactylus stylus [21] and the stylus presented by Kamuro et al. [19] are similar, but add additional haptic feedback to the devices. However, the tether limits mobility for use in room-scale displays (e.g. a VR CAVE), and can cause increased fatigue due to balance issues from the cord [21]. This prior work influenced the design decision to make the OVR Stylus fully wireless.

The SenStylus [12] is perhaps the most similar to the OVR Stylus design and holds comparable motivations with our prior work in developing pen-based input devices for immersive 3D modeling [17] and other VR applications. The SenStylus consists of a stylus containing buttons and analog input linked to a wireless Logitech Rumblepad game controller worn on the user's arm. In contrast to this 15 year old design, the OVR Stylus provides a modern implementation. It does not require the cumbersome wrist-worn device and is both lighter and thinner.

3 Design Considerations

Our design considerations were informed by six years of use with a first generation stylus prototype used for input in a VR CAVE with applications for scientific visualization and 3D modeling. This experience instructed our design goals for both the physical characteristics of the device and functional characteristics.

One of the main advantages of the first prototype is its light weight. We have used it in multiple 2–5 hour VR sessions with relatively little discomfort compared to using traditional and heavier wand input devices, like the HTC VIVE Controllers. This is particularly important for 3D input devices that are used for large mid-air sweeping input to avoid the common gorilla arm effect [7]. Managing weight was a key requirement for the new design. Prior work has also found that the weight distribution of a stylus can impact fatigue with use [28]. As a result, the weight distribution in the OVR Stylus is centered on the user's hand as much as possible. The total assembled weight is about 35 grams.

Six years ago, our first prototype was also quite thick, with a diameter of 25mm. At the time, the smallest off-the-shelf micro-controller development boards available were 18mm wide. Although the first prototype was cylindrical in the shape of a pen, some users would initially hold it like a baton with a power grip. A primary goal of the redesign is to keep the OVR Stylus as thin as possible. This more closely mimics the shape of a pen or marker and avoids ambiguity in how the device should primarily be held using a precision grip (while still allowing for a power grip if needed).

The wireless connectivity is a primary functional characteristic. Without a tether providing power, this design decision necessitates that the electronic components have low consumption to avoid the weight and size impact of heavier batteries. Along these lines, the first prototype required the user to disassemble the stylus to remove the battery for recharging. To improve the utility of the OVR Stylus, the current design needs to include a battery charging circuit, allowing the device to be recharged through a USB port without disassembly.

For 3D tracking, we considered adding an inertia measurement unit (IMU) to calculate orientation. However, this approach would require an external way to measure the absolute spatial position of the stylus. Instead, the design integrates optical tracking markers for use with common motion capture systems (e.g. NaturalPoint OptiTrack).

The last functional consideration is to limit the number of buttons on the device. Although it is a common practice in user interface design to map each new function to a separate button; this approach does not scale. Instead, we follow the recommendation of Jackson et al. [18] to minimize the number of buttons and instead use context-based interaction to interpret button input. Our experience with the first prototype, which also included only two buttons, has been that this characteristic improves learnability of the interface and makes it more intuitive to use.

4 OVR Stylus

The OVR Stylus is made from the components shown in Figure 3. A custom-designed printed circuit board (PCB) houses the electronics and micro-controller with a Bluetooth Low-Energy (BLE) module and antenna. For user input, there are two buttons and a touch pad created by a sliding soft membrane potentiometer that gives analog values. A LRA haptic motor and controller board is mounted at the front of the stylus for vibrotactile feedback. The 3D printed case allows interchangeable tips and optical tracking configurations. In the sections below, we describe the hardware and software implementations in more detail.

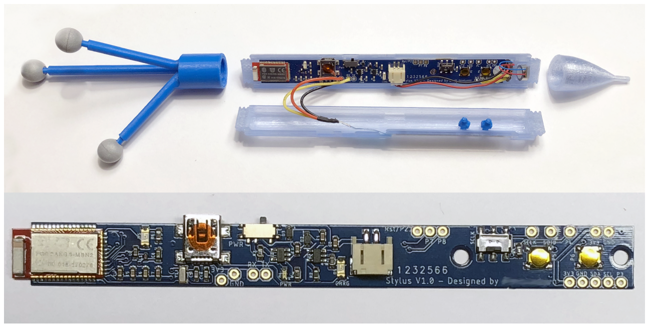
Figure 3: Top: The custom PCB, Lithium-polymer battery, LRA haptic motor and controller components fit inside a 3D printed case with interchangeable ends. Bottom: The custom PCB enables the thin form-factor.

4.1 Hardware

To meet the minimum thickness requirement necessitated developing a custom printed circuit board. This design process includes choosing components and creating a schematic for the electronic circuits. This step is followed by laying out the placement of the components and connections to create the board design, and finally submitting the design for manufacture and assembly.

The circuit schematic, shown in Figure 2, contains several primary components (summarized in Table 1). The primary component choice is the micro-controller and BLE module. At the start of this project the smallest BLE system-on-a-chip (SoC) was the Redbear MB-N2 BLE module [29] that combines a Nordic nRF52832 BLE chip [30] and on-board antenna. Its 10mm width sets the standard for the rest of the PCB layout.

To support on-board battery charging, a MCP73831 Li-Polymer charging manager chip [25] is used. This low-cost chip manages preconditioning and charging rate. A 110mAh Li-Polymer battery is used for its small size and high energy density.

Two Schottky diodes (D2 and D3 in Figure 2) are blocking diodes to broker power. When the USB port is plugged in, its higher voltage charges the battery through the MCP73831 chip and also powers the stylus. When it is not present the power runs through D3 to power the stylus. An AP2112k linear voltage regulator [10] maintains a constant 3.3v required by the MB-N2 module. An external 32kHZ crystal and DC/DC converter circuit lowers the power usage of the nRF52832 chip.

The PCB was designed using Autodesk Eagle [4]. This layout software is fairly easy to learn and use (the author has no prior PCB design experience). The circuit schematic is created first by importing the components and creating the electrical connections. This is followed by the PCB design where the user lays out the primary modules and routes the wires between them. The final PCB design is shown in Figure 3.

Two surface-mount momentary push-button switches are mounted near the front of the stylus to serve as input. The switches require 160gf to activate, giving a tactile feel of a click, while not requiring so much force that the user accidentally re-positions the stylus in space.

In addition to the buttons, the stylus integrates a soft membrane potentiometer (SoftPot). Sliding a finger along the sensor membrane produces variable resistance which can be read using an analog input on the micro-controller. VR application developers can use this input to adjust sliders or other scaled input. Additionally, its output could be discretized to produce additional input buttons by pressing different locations along the slider.

To support haptic effects, the stylus integrates a Samsung 8mm Linear Resonant Actuator (LRA) motor that is controlled by a

Table 1: Summary of primary hardware components.

Component	Part Description
Micro-Controller & BLE	RedBear Labs MB-N2 nRF52832 Module. Other nRF52832 alternatives: U-Blox NINA-B112, Seeed MDBT42Q
Haptic Controller	Fyber Labs LRA Haptic Flex Module - DRV2605L
LRA Motor	Samsung LRA 8mm
SoftPot Potentiometer	Spectra Symbol 50mm ThinPot
Battery Manager	MCP73831T Li-Ion, Li-Pol Controller
Voltage Regulator	AP2112 - 600mA CMOS LDO Regulator
Battery	Sparkfun 110mAh Lithium Polymer

DRV2605L LRA driver module produced by Fyber Labs [14]. The LRA uses magnetic fields to move a mass linearly in one dimension to produce a haptic feeling. The DRV2605L supports multiple standard effects from different types of clicks to pulses and buzzes. An LRA was chosen over the more commonly available Eccentric Rotating Mass (ERM) motors, which rotate an off-center mass to produce a force, because LRAs enable forces along the pointing direction of the stylus rather than perpendicularly. This helps maintain precision because the forces will predominantly move the stylus in depth, which users already have less precision doing [33].

The case to hold the electronic components was modeled using Autodesk Fusion 360 [5] and 3D printed using a Makerbot Replicator printer. 3D printing allows for quick iterations on the design. The plastic case supports the light-weight design goal while still being robust for everyday use.

The top and bottom of the case are connected using interchangeable threaded end-caps. The front end-cap narrows to a point for precise input and to further replicate the use of a pen. The back end-cap contains antlers to mount reflective markers used for 3D tracking. Multiple marker configurations can be printed and interchanged if multiple styluses are used to enable bi-manual input or multiple users. The LRA motor is glued to the front of the stylus to facilitate transfer of the forces. The battery is mounted in the middle to balance the weight. Additional 3D printing infill is added to the front, minimally increasing the weight, while offsetting the additional material used at the back of the stylus in the tracking antlers to provide balance. The assembled stylus weighs 35g.

4.2 Software

The software running on the OVR Stylus is primarily organized around the Bluetooth low energy stack. As part of the BLE protocol, the Generic Access Profile (GAP) defines the roles that a BLE device can have [6]. The two relevant roles for the stylus communication are:

- Peripheral – Usually small, low-power devices that advertise a set of data features and can connect to more powerful central devices to output data.
- Central – A device that scans for advertising peripherals and creates a connection to receive data.

4.2.1 Peripheral

The stylus acts as the peripheral device for the BLE communication. It advertises that it supports the Nordic Universal Asynchronous Receiver/Transmitter (UART) service [31], used to send and receive serial data over the BLE radio. This allows central devices to create a connection to the stylus to receive button or analog input states and to instruct it to play specific haptic effects.

For ease of programming, the stylus uses the commonly available Arduino programming environment. The arduino-nrf5 library created by Sandeep Mistry [26] ports the core Arduino libraries to be available for programming nRF52832 boards. We use an additional modification of the library created by Matthew Ford [13] that adds low-powered serial communication functions and sleep timers. These libraries require that you first flash a SoftDevice, precompiled binaries that run the BLE protocol stack, onto the stylus hardware. The stylus uses the Nordic S132 v2.0.1 SoftDevice. The haptic effects are provided by the Adafruit DRV2605 library [1].

A simplified version of the stylus code is shown in Listing 1. Fully documented code is available from https://github.com/bretjackson/OVRStylus. The `setup()` method, which Arduino runs when powered up, initializes the pins connecting the buttons to the micro-controller as inputs with internal pullup resistors. These internal pullup resistors pull the input to the high state and prevent fluctuations in the current from inadvertently being read as button presses. It initializes the DRV2605 library, starts the BLE stack advertising for UART connections, and registers callback handlers for when a central device connects or disconnects to the stylus peripheral.

Listing 1: Simplified pseudo-code implementation of the stylus code. Full code is available at https://github.com/bretjackson/OVRStylus

```
void setup() {
  // Set the micro-controller pins and IO

  // Initialize the haptic controller module here

  // Initialize and start advertising for connections.
  ble.setName("OVR␣Stylus"); // Advertised name
  ble.setConnectedHandler(handleConnection);
  ble.setDisconnectedHandler(handleDisconnection);
  ble.begin();
}

void loop() {
  // Sleep in low-power mode until data is received
  sleep();
  while (ble.available() > 0) {
    // read message and play haptic effect
  }
}

void handleConnection(BLECentral& central) {
  sleepTimer.start(handleSleepTimer);
}

void handleDisconnection(BLECentral& central) {
  sleepTimer.stop();
}

void handleSleepTimer() {
  // Update current state of the buttons and analog
  // input and sendStateUpdate if changed
}

void sendStateUpdate() {
  byte states = 0b00000000;
  if (button0 == LOW) {
    bitSet(states, 7);
  }
  if (button1 == LOW) {
    bitSet(states, 6);
  }

  for(int i = 0; i < 6; i++) {
    if (bitRead(softpotState, i)) {
      bitSet(states, i);
    }
  }
  ble.write(MESSAGE_DELIMITER); ble.write(states);
  ble.flush();
}
```

The `loop()` method is then repeatedly called. It sleeps the processor in a low-powered mode to save energy until a message is received over BLE instructing the stylus to play a haptic effect. The BLE radio interrupts the processor, waking it from the sleep state. The message is read and the indicated haptic effect is triggered on the DRV2605 controller.

Button input is handled when a controller connects to the peripheral device. A sleep timer is started that polls the `handleSleepTimer()` callback every 2ms to determine if the buttons or analog SoftPot state have changed. This interval can be adjusted if needed. Setting it less frequently saves battery life but potentially adds additional lag to the input. If the input state changes, the controller sends a serial message to the central device indicating the new state.

4.2.2 Central

The implementation of a central device to connect with the OVR Stylus and respond to input is somewhat arbitrary. The Bluetooth Low Energy protocol is available on most major operating system platforms, and any could be used to write a central program to connect with the stylus. In this section we describe our specific implementation to make a complete system, although other options could easily be substituted without changing the stylus design or software running on it.

Each OS defines their own BLE stack with associated library calls, making it challenging to write a cross platform central device. For example, Apple's iOS allows a programmer to access the built in BLE calls using Swift, but this same code will not run on Windows. Our approach avoids this cross-platform compatibility issue by using a second BLE micro-controller device plugged into a USB port on the host machine to act as an intermediary. The device acts as a BLE-to-serial repeater, receiving Bluetooth messages from the stylus and repeating them on a serial connection to the host machine.

The specific device used as the central is an Arduino Nano 33 BLE development board, that can be programmed as a central BLE device using the ArduinoBLE library [2]. The central code starts by scanning for peripherals in range that advertise the Nordic UART service. If it finds one with a matching name "OVR Stylus", it initiates a connection to the stylus peripheral.

While the central device is connected to the stylus, it forwards messages from the BLE radio to the serial port and vis-a-versa. This code is shown in Listing 2.

Listing 2: The central device forwards messages received from the stylus peripheral to the serial port of the host computer.

```
while (peripheral.connected()) {
  if (Serial.available() > 0) {
    byte hostByte = Serial.read();
    rxCharacteristic.writeValue(hostByte);
  }

  if (txCharac.valueUpdated()) {
    // lp_BLESerial.h buffers in sets of 20 bytes
    byte bleBytes[20];
    int readLen = txCharac.readValue(bleBytes, 20);
    Serial.write(bleBytes, readLen);
  }
}
```

Forwarding messages to the host computer's serial port enables the OVR Stylus input to integrate with VR applications using the VRPN library [32]. We have written a new device class for VRPN to support the OVR Stylus. The class inherits from `vrpn_analog_serial` and `vrpn_button_filter`. The relevant code for the `get_report()` method used to parse the button states from the serial input is show in Listing 3.

Listing 3: The simplified report method for the vrpn_OVRStylus VRPN device.

```
void vrpn_OVRStylus::get_report()
{
  int i;
  if (status == vrpn_ANALOG_SYNCING) {
    if (1 == vrpn_read_available_characters(serial_fd,
                                            buffer, 1)) {
      // if not a message start, we have an error
      if (buffer[0] != _messageStartByte) {
        return;
      }
      _num_read = 0;
      status = vrpn_ANALOG_PARTIAL;
    }
  }
  if (status == vrpn_ANALOG_PARTIAL){
    int result = vrpn_read_available_characters(
                          serial_fd,
                          &buffer[_num_read],
                          _expectedNumChars-_num_read);

    _num_read += result;
    if (_num_read < _expectedNumChars) {
      return;
    }

    for (i = 0; i < 2; i++) {
      lastbuttons[i] = buttons[i];
    }

    buttons[0] = static_cast<unsigned char>((buffer[0]
      & (1 << 7)) ? VRPN_BUTTON_ON : VRPN_BUTTON_OFF);
    buttons[1] = static_cast<unsigned char>((buffer[0]
      & (1 << 6)) ? VRPN_BUTTON_ON : VRPN_BUTTON_OFF);

    vrpn_Analog::last[0] = vrpn_Analog::channel[0];
    // Softpot value is in lowest six bits
    vrpn_Analog::channel[0]=buffer[0] & ((1<<6) - 1);

    status = vrpn_ANALOG_REPORT_READY;
    report_changes();
      status = vrpn_ANALOG_SYNCING;
  }
}
```

4.2.3 Communication Protocol

The communication protocol between the OVR Stylus and host computer is designed to be as simple as possible to minimize the number of bytes that need to be sent, reducing lag. Messages from the stylus to the host follow a two byte format.

1. The first byte holds a message start character, '!' in our implementation.

2. The second byte encodes the button and analog states.

Because the states are encoded only in one byte, the message start byte is nonessential. However, including it adds minimal latency with additional protection for identifying corrupt messages.

The states are encoded in the second byte using the following scheme:

- The highest bit represents the state of the first button. A value of one indicates that it is currently pressed.

- The second highest bit represents the state of the second button.

- The six remaining bits encode the analog potentiometer value from 0–63.

This design does involve a trade-off. The SoftPot potentiometer is able to output values from 0–1023 but only able to transmit from 0–63. We found that in general the full output was not very stable when using a finger as a slider on the surface. Even keeping the finger still resulted in small fluctuations. Remapping the input to the smaller range enabled more stable results, while 64 possible values still meets the needs for most of our applications. If an application needs the larger range, this code can easily be modified to transmit the value with more bytes. The SoftPot input is further stabilized using exponential filtering of the raw value.

Messages from the host computer to the stylus follow a similar format of a message start byte, '!', followed by an integer from 1–123 that corresponds with haptic effect IDs pre-programmed into the DVR2605 LRA controller.

5 Discussion

We gained several insights during the process of designing and prototyping the OVR Stylus. Below, we reflect on these lessons learned with suggestions for future designers of novel input devices for VR applications.

Custom PCB design allows for flexibility. The primary difference between the first version of the stylus and the current is the use of a custom designed PCB electronics board. This allows for the thin case design that is critical for the stylus to perform like a pen in 3D space. It also allows for flexibility. Using a tool like Autodesk Eagle, we could lay out the components of the circuit in a different configuration to make other novel input devices. For example bracelets could be made to explore body-based interactions [20], or using a square layout configuration rather than rectangular could be used to make a wearable ring input device. The OVR Stylus' electronic circuit schematic is generalizable enough to support creation of these other types of devices with little modification other than the layout of components.

Linked design tools boost accuracy and speed. Multiple 3D modeling systems are available to design the stylus case. However, the chosen tool, Autodesk Fusion 360, directly integrates with the Eagle PCB design tool. This enabled us to automatically import a 3D model of the circuit board, complete with modeled components, enabling adjustment of the case design for taller components and clearances without having to wait for the case to be 3D printed and tested.

3D printing supports iteration. Using 3D printing to build the case components enables rapid iteration of the design. Our process required several iterations on the design of the tracking antler configuration for the optical tracking. Initial designs were constrained in size, however they did not spread the tracking markers far enough apart that they could easily be distinguished by our optical tracking system.

Design for modification. We specifically designed the stylus to use end-caps that thread onto the main case so that they would be modifiable in the future. This allows for adjustment to different tracking configurations by changing the rear end-cap. Additionally, the stylus function can be changed in the future by adjusting the front cap. For example, in a VR surgery training simulator the pointed tip could be replaced with a mock scalpel shape, or a dentistry simulator could replace it with a dentist's pick. An augmented reality application could replace the tip with an actual marker to enable real drawing while tracking the digital twin.

In addition, we recommend supporting more input/output pin locations on the PCB than are needed for the number of buttons and other inputs actually used. Our design includes four additional pin locations that provide opportunities to add additional sensors to the design in the future.

6 Conclusion

We have presented the OVR Stylus as an open-source alternative to commercial 3D VR pen devices. From our experience, the thin, light-weight design supports extended use. We have shared the hardware and software implementations and design files to spur others to quickly and cheaply develop their own version, and our discussion of the design process and lessons learned will hopefully inspire more creative custom development of novel input devices for VR.

Acknowledgments

This work was supported in part by a Wallace scholarly activities grant from Macalester College.

References

[1] Adafruit. DRV2605. (github.com/adafruit/Adafruit_DRV2605_Library). Accessed: 2020-01-29.

[2] Arduino. Arduinoble library. (arduino.cc/en/Reference/ArduinoBLE). Accessed: 2020-01-29.

[3] R. Arora, R. Habib Kazi, T. Grossman, G. Fitzmaurice, and K. Singh. Symbiosissketch: Combining 2D & 3Dsketching for designing detailed 3D objects in situ. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pp. 1–15, 2018.

[4] Autodesk Inc. Eagle. (autodesk.com/products/eagle/overview). Accessed: 2020-01-29.

[5] Autodesk Inc. Fusion 360. (https://www.autodesk.com/products/fusion-360/overview). Accessed: 2020-01-29.

[6] Bluetooth SIG, inc. Bluetooth 5 core specification. (https://www.bluetooth.com/specifications/bluetooth-core-specification/). Accessed: 2020-01-29.

[7] S. Boring, M. Jurmu, and A. Butz. Scroll, tilt or move it: using mobile phones to continuously control pointers on large public displays. In *Proceedings of the 21st Annual Conference of the Australian Computer-Human Interaction Special Interest Group: Design: Open 24/7*, pp. 161–168, 2009.

[8] M. A. Brown and W. Stuerzlinger. Exploring the throughput potential of in-air pointing. In *International Conference on Human-Computer Interaction*, pp. 13–24. Springer, 2016.

[9] J. Clarkson. 6 - human capability and product design. In H. N. Schifferstein and P. Hekkert, eds., *Product Experience*, pp. 165–198. Elsevier, San Diego, 2008. doi: 10.1016/B978-008045089-6.50009-5

[10] Diodes Inc. AP2112k linear voltage regulator. (https://www.diodes.com/assets/Datasheets/AP2112.pdf). Accessed: 2020-01-29.

[11] Fasttrak. Polyhemus digitizer. (https://polhemus.com/). Accessed: 2020-01-29.

[12] M. Fiorentino, A. E. Uva, and G. Monno. The Senstylus: a novel rumble-feedback pen device for cad application in virtual reality. *WSCG*, 2005.

[13] M. Ford. Easy very low power BLE in arduino. (https://www.forward.com.au/pfod/BLE/LowPower/index.html). Accessed: 2020-01-29.

[14] Fyber Labs Inc. LRA haptic flex module. (https://www.tindie.com/products/fyberlabs/lra-haptic-flex-module/). Accessed: 2020-01-29.

[15] W. Global. Wacom digitizer stylus.

[16] Holo-Light inc. Holo-stylus. (https://www.holo-stylus.com/). Accessed: 2020-01-29.

[17] B. Jackson and D. F. Keefe. Lift-off: Using reference imagery and freehand sketching to create 3D models in VR. *IEEE Transactions on Visualization and Computer Graphics*, 22(4):1442–1451, 2016.

[18] B. Jackson and D. F. Keefe. From painting to widgets, 6-DOF stylus input beyond the pointing metaphor. In W. Sherman, ed., *VR Developer Gems*, chap. 14, pp. 243–268. CRC Press, 2019.

[19] S. Kamuro, K. Minamizawa, N. Kawakami, and S. Tachi. Ungrounded kinesthetic pen for haptic interaction with virtual environments. In *RO-MAN 2009-The 18th IEEE International Symposium on Robot and Human Interactive Communication*, pp. 436–441. IEEE, 2009.

[20] R. Khadka and A. Banic. Body-prop interaction: Evaluation of augmented open discs and egocentric body-based interaction. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 1705–1710. IEEE, 2019.

[21] E. Kruijff, G. Wesche, K. Riege, G. Goebbels, M. Kunstman, and D. Schmalstieg. Tactylus, a pen-input device exploring audiotactile sensory binding. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pp. 312–315. ACM, 2006.

[22] Logitech. VR ink stylus. (https://www.logitech.com/en-us/promo/vr-ink.html). Accessed: 2020-01-29.

[23] R. Lyu, H. Hao, W. Chen, Y. Liu, F. Wang, and A. C. Wu. Elastylus: flexible haptic painting stylus. In *SIGGRAPH Asia 2015 Emerging Technologies*, pp. 1–3. 2015. doi: 10.1145/2818466.2818475

[24] Massless Corp. Massless pen. (https://massless.io/). Accessed: 2020-01-29.

[25] Microchip. MCP73831 li-polymer battery charge manager chip. (https://www.microchip.com/wwwproducts/en/en024903). Accessed: 2020-01-29.

[26] S. Mistry. Arduino core library for nordic semiconductor nRF5 based boards. (https://github.com/sandeepmistry/arduino-nRF5). Accessed: 2020-01-29.

[27] S. Nagasaka, Y. Uranishi, S. Yoshimoto, M. Imura, and O. Oshiro. Haptylus: haptic stylus for interaction with virtual objects behind a touch screen. In *SIGGRAPH Asia 2014 Emerging Technologies*, pp. 1–3. 2014.

[28] D.-M. Pham and W. Stuerzlinger. Is the pen mightier than the controller? a comparison of input devices for selection in virtual and augmented reality. In *25th ACM Symposium on Virtual Reality Software and Technology*, pp. 1–11, 2019.

[29] Redbear Labs. MB-N2 bluetooth low energy module. (https://github.com/redbear/nRF5x/tree/master/nRF52832). Accessed: 2020-01-29.

[30] N. Semiconductor. nRF52832 bluetooth 5 chip. (https://www.nordicsemi.com/Products/Low-power-short-range-wireless/nRF52832). Accessed: 2020-01-29.

[31] N. Semiconductor. UART/serial emulation over BLE. (https://tinyurl.com/whohol7). Accessed: 2020-01-29.

[32] R. M. Taylor, T. C. Hudson, A. Seeger, H. Weber, J. Juliano, and A. T. Helser. VRPN: a device-independent, network-transparent VR peripheral system. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pp. 55–61, 2001.

[33] R. J. Teather and W. Stuerzlinger. Pointing at 3D targets in a stereo head-tracked virtual environment. In *2011 IEEE Symposium on 3D User Interfaces (3DUI)*, pp. 87–94. IEEE, 2011.

[34] Ultrahaptics Ltd. Leap motion. (https://www.leapmotion.com/). Accessed: 2020-01-29.

[35] A. Withana, M. Kondo, Y. Makino, G. Kakehi, M. Sugimoto, and M. Inami. ImpAct: Immersive haptic stylus to enable direct touch and manipulation for surface computing. *Computers in Entertainment (CIE)*, 8(2):1–16, 2010.

[36] zSpace. zSpace stylus. (https://zspace.com/technology/). Accessed: 2020-01-29.