Usability Evaluation of a VibroTactile API for Web-based Virtual Reality Experiences*

José Canelha$^{1[0000-0003-3049-1675]}$, Jorge C. S. Cardoso$^{2[0000-0002-0196-2821]}$, and André Perrotta$^{3[0000-0003-2953-0585]}$

[1] University of Coimbra, DEI `zecanelha@gmail.com`
[2] University of Coimbra, CISUC, DEI `jorgecardoso@dei.uc.pt`
[3] Catholic University of Portugal, CITAR `avperrotta@dei.uc.pt`

Abstract. This paper presents the Vibrotactile Editor System for designing and programming Vibrotactile (VT) feedback for Virtual Reality (VR) experiences. We describe the system and a usability evaluation of the programmatic component for the A-Frame web-based VR framework through a set of programming tasks. The resolution of the tasks showed some difficulty in understanding the state of the component when developing without the VT hardware present. The results show a number of potential improvements that we discuss.

Keywords: Vibrotactile feedback · Virtual Reality · Usability evaluation · API

1 Introduction

Providing Vibrotactile (VT) feedback for VR experiences is often complex, involving custom hardware with several actuators. An inherent challenge in the use of VT in this context is the technical expertise in designing and programming the different VT patterns to be used in VR experiences [1]. Different combinations of vibration patterns across the various actuators and across time must first be designed and iterated, and then the final sequences must be programmed into the VR framework.

Motivated by the technical requirements of the Thertact-Exo project[4], whose objective is to improve a neurorehabilitation training protocol that induces partial neurological recovery in paraplegic patients through long-term training with a brain-machine interfaced exoskeleton and an immersive VR system enhanced with visual-tactile feedback, the aim of this work was to develop a Graphical User Interface (GUI) for editing VT patterns and a high-level VR component that allows easier programming of VT feedback.

* Funded by national funds through the FCT - Foundation for Science and Technology, I.P., within the scope of the project CISUC - UID/CEC/00326/2020 and by European Social Fund, through the Regional Operational Program Centro 2020.

[4] *Thertact-Exo: A brain controlled exoskeleton for spinal cord regeneration*, funded by Prêmio Melo e Castro - Santa Casa da Misericórdia de Lisboa, reference: MC-12-18.

In this paper, we describe the proposed system, an initial evaluation of the Application Programming Interface (API) of the VR framework component, and the insights we gathered from this evaluation.

2 Vibrotactile Editor System

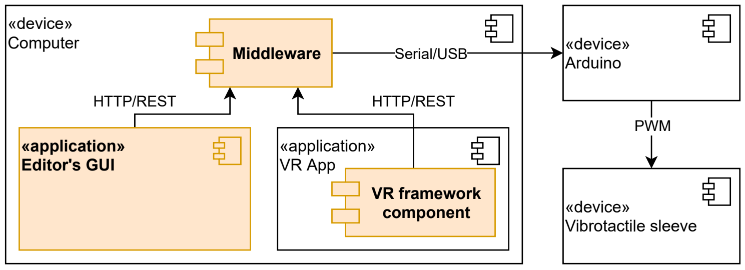


Fig. 1: Architecture of the Vibrotactile Editor System

The Vibrotactile Editor system (Fig. 1) is composed of 1) middleware, 2) editor's GUI for editing, and 3) VR framework components.

The middleware component is responsible for interpreting the commands issued from either the GUI or from the VR framework components and translating them into Arduino commands to control the VT sleeve. The middleware exchanges information with Arduino via a serial connection, controlling the vibration motors through Pulse Width Modulation (PWM). It exposes an HyperText Transfer Protocol (HTTP) API that is used by the VR framework components and the editor's GUI. Additionally, the middleware stores user's configurations as well as custom and default vibration patterns.

The editor's GUI is a desktop application for creating and testing vibration patterns across a number of actuators. The editor allows playing the timeline directly on the VT device for testing purposes, but its main output is a configuration file ($.json$) that can be used by the VR components.

The VR framework components provide high-level programming abstractions to programmers of VR applications. We currently support the web-based VR framework A-Frame (http://aframe.io) and Unity (http:/unity3d.com). The VR components also expose a number of JavaScript functions that can be used directly by programmers such as $sin()$ to send a sinusoidal waveform vibration, $ramp()$ to send a ramp vibration, $sendVibrations()$ to vibrations defined in a configuration file, and $customVibrations()$ to send a custom timeline.

3 Vibrotactile Component API Usability Testing

We evaluated the A-Frame implementation of the VR framework component through a remote, API task-based usability test approach.

Eight participants completed the test, all were students of Informatics Engineering in our department, with 3+ years of programming experience. we sent out the test tasks through email, as a shared Google Doc. We asked participants to leave comments in their code, pointing out unclear issues with the API's documentation, and suggestions to improve the API. We also asked them to fill in a post-test questionnaire after completing the tasks and requested users to send their solutions through email. The test consisted of four programming tasks (https://bit.ly/3fykqye), with increasing difficulty. The questionnaire was based on the structured interview presented by [2]. The complete set of questions can be seen in Fig. 2.

3.1 Results

The source code of each participant was analysed to verify the correctness of each solution. Table 1 shows the main issues that were found. The main issues were related to the entity that the component was attached to, the event associated with triggering the vibration, the use of an incorrect vibration file, and extra or incorrect parameters in calling the vibration functions. The results from the post-test questionnaire are shown in Fig. 2. In general, the results for each question are positive, or neutral.

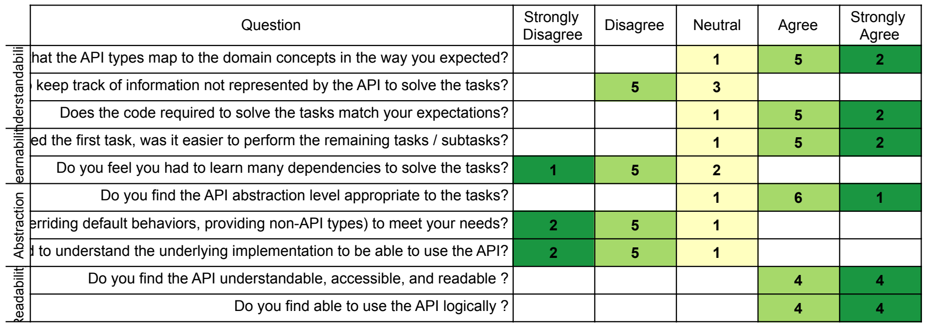
Table 1: A-Frame Component API usability tests results

Task	**Issues**
#1 Attach the vibrotactile component to an entity	Attaching component to wrong entity; Not associating event; Associating wrong event
#2 Vibrations after animation	Using wrong animation event; Not associating event; Associating wrong event; Incorrect vibration file
#3 Use JavaScript methods	Sending extra vibrations; Incorrect ramp; Associating wrong event
#4 Vibrotactile GPS	Vibrations in area transition only

4 Discussion

Given the mistakes that we identified in participants' code, we identified opportunities for improving the usability of the VR framework component:

- Provide more complete log messages. In particular, when vibrations are triggered from events, the log message is not explicit enough. This may have caused some of the mistakes we found in the source code from the various participants. For example, they may have assumed they had configured the correct event.



	Question	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
nderstandabili	hat the API types map to the domain concepts in the way you expected?			1	5	2
	keep track of information not represented by the API to solve the tasks?		5	3		
	Does the code required to solve the tasks match your expectations?			1	5	2
earnabili	ed the first task, was it easier to perform the remaining tasks / subtasks?			1	5	2
	Do you feel you had to learn many dependencies to solve the tasks?	1	5	2		
Abstraction	Do you find the API abstraction level appropriate to the tasks?			1	6	1
	erriding default behaviors, providing non-API types) to meet your needs?	2	5	1		
	to understand the underlying implementation to be able to use the API?	2	5	1		
eadabilit	Do you find the API understandable, accessible, and readable ?				4	4
	Do you find able to use the API logically ?				4	4

Fig. 2: A-Frame component API usability questionnaire results

- Provide a visual debug mode where the entities to which the component is attached are visually highlighted and visually represent (e.g., through animation) when a vibration is triggered. This would make it easier to detect when the vibrations are attached to the wrong entities.
- Provide an audio debug mode where vibrations are automatically rendered as audio. This would make it easier to understand the vibrations being triggered even in case they are attached to non-visible entities. This would make it obvious, for example in the last task, whether the vibrations were being triggered continuously as the user moved through the virtual environment. Audio representations can also make it easier to notice when wrong (or more than expected) vibrations are being triggered.

5 Conclusion

Having a VT system for designing and programming vibration patterns is an important aspect for the creation of rich VR experiences that rely on VT feedback.

Our system provides not only a GUI for editing the VT patterns, but also programmatic components for the A-Frame and Unity VR frameworks. We have evaluated the VR framework component and are in the process of revising its implementation according to the findings.

References

1. Burdea, G.C.: Virtual rehabilitation–benefits and challenges. Methods of information in medicine **42**(5), 519–23 (2003). https://doi.org/10.1267/METH03050519, http://www.ncbi.nlm.nih.gov/pubmed/14654886
2. Piccioni, M., Furia, C.A., Meyer, B.: An empirical study of API usability. International Symposium on Empirical Software Engineering and Measurement pp. 5–14 (2013). https://doi.org/10.1109/ESEM.2013.14