Cyberglove

Winter Term 2017/2018

Documentation

supervised by Michael Schröder

Lukas Bach	1939135
Shant Gananian	2132387
Frieder Haizmann	1995626
Fabian Nussberger	1721271
Tanja Stumpf	1664311
Johanna Thiemich	1928207
Ran Xu	1913117

Virtual Reality Project
Institute for Information Management in Engineering, Karlsruhe Institute of Technology

February 8, 2018

Contents

1 Introduction

1.1 Motivation and Objective

Virtual Reality technology to immerse in a computer generated scenario simulating a realistic experience because of a created environment similar to the real world is continuously improving and potential users become more and more attracted by the possibilities offered through virtual reality.

Current technology enables the user to look around the artificial world, move in it, and interact with virtual features or items. Therefore continuous development of virtual reality items and features is necessary to improve their functionality as well as expand their possibilities of application. Because of that, our objective is to adjust the actual Cyber Glove and design new use cases to offer the user a better experience and expand the application possibilities of the Cyber Glove in the virtual reality.

Fulfilling this objective, we defined our project plan and our strategy and divided the main aspects into three sub aspects. Therefore all team members were able to specialize better in their chosen sub aspects and progress was guaranteed. The three aspects were defined as followed:

Hardware development: Designing a new Cyber Glove with a new board and its housing as well as implementing three additional bending sensors and a joystick. Renewing the tracking system and changing the battery to improve functionality of the glove and make it lighter and more comfortable to use.

Software development: Revising and redesigning of the WLAN transmission protocol as well as modifying, customizing and implementing more features by adapting the Arduino code to transmit the sensor values and enable the usage of the Cyber Glove and its new functions in the cave.

Development and design of new use cases: Our renewals and development of software and hardware were tested by development and implementation of new use cases. Those use cases simplify it for users to understand the functionality of the Cyber Glove and a new design was implemented to make it easier to immerse in the virtual world using the Cyber Glove in the cave.

Thus, definition of individual goals in the main aspects represent the overall task. The implementation of the overall task improves immersion in the virtual world with the Cyber Glove. Furthermore, structuring of the three main aspects helps to simplify the management and organization within the team.

1.2 Team and Project Plan

Interdisciplinary skills should ensure optimal development of the Cyber Glove. Therefore our team consists of 7 members of different disciplines who did contribute their individual knowledge to the project. We divided the group of three Computer Scientists, two Civil

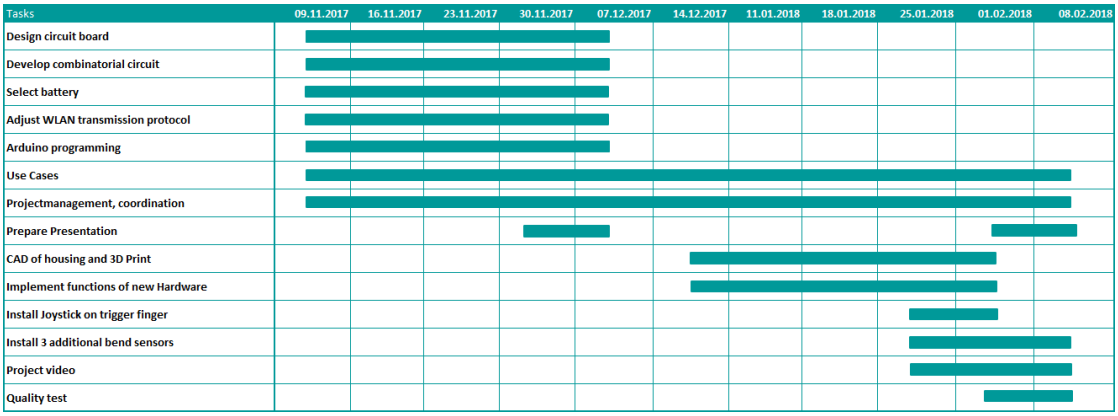


Figure 1: Gantt chart describing our tasks

Engineers and two Industrial Engineers into smaller project groups. Thereby each group was resolving another challenge of the objectives set.

We have formed the following groups in order to divide the tasks:

Hardware team: Frieder, Fabian

Software team: Johanna, Lukas

Use case team: Ran, Shant, Johanna, Lukas

Coordination and management: Tanja

The Gantt chart in Figure 1 shows the main tasks, together with the scheduled time to fulfill each task.

First of all we analyzed the current Cyber Glove and discussed challenges and difficulties of further development. It was important to get an insight in the technical state of the Cyber Glove and further explanation regarding hardware and software from our project coordinator. After getting a complete overview we defined the objectives as well as the tasks and divided our team in expert groups. Afterwards we scheduled the tasks and estimated the time necessary to fulfill each task.

In the second phase the hardware group designed the circuit board, developed the combinatorial circuit and selected a battery. Our software group adjusted the WLAN transmission protocol and started Arduino programming. The user case team started to focus on understanding Poly VR software to afterwards start programming new use cases for the Cyber Glove.

Weekly team meetings were coordinated and held to keep all team members up to date on current progress of the other expert groups. Furthermore those team meetings were a possibility for each sub group to get input and ideas by other sub groups so even though there were existing expert groups, development was still discussed and supported through the whole team. The fact of having team members of very different specializations made the input across groups very profitable and versatile. The second phase closed with the

midterm presentations where we presented the current state and developments of the project.

Hereafter the third phase started in which our software team joined the use case group to generate use cases meanwhile they kept on working on the software. The hardware team was responsible to set up the CAD of housing and transmit it to project coordinator for 3D printing. Furthermore the started implementing functions of new hardware. To the same time the new Cyber Glove was designed and to afterwards install joystick and 3 additional bend sensors. All together was connected through soldering of the wires.

The task "prepare presentation" was fulfilled two times, each one week before the mid and final presentation by all team members. Management, coordination and communication, designing the new Cyber Glove by sewing and wiring as well as the project video were the tasks of the project coordinator.

A more detailed description of the individual tasks as well as the identification of problems during the process and the learning effect are discussed below.

2 Task Overview

2.1 Our Approach

The different project subgroups are represented by the tables listing their tasks, the time planned and needed to actually fulfill each task, task description, difficulties which came up while performing the tasks and the final result of each task.

General efforts of the VR internship for example attendance during the lectures and preparatory work, like VR homework is shown below the group tables. The hours are cumulated hours over all group members.

2.2 Tasks

2.2.1 Hardware tasks

Hardware: Circuit board wiring		
	Fabian, Frieder. 09.11.17 - 30.11.17	
Time in hours:	Estimated: 15 hours each	Actually: 21 hours each
Task description:	Getting information about the components of the board.	
Issues:	Information overload and learning the basic skills first.	
Result:	Complete wiring for our circuit board.	

Hardware: Design Board

	Frieder. 30.11.17 - 07.12.17, 20.12.17	
Time in hours:	Estimated: 20 hours	Actually: 38 hours (5 hours first desing, 20 hours reducing size, 10 hours troubleshooting 3 hours setting labels)
Task description:	With the schematics of the board, drawn in the step before, now a pcb layout as small as possible was needed. With manual placement and Using of the autorouter, this was done in a few hours but to reduce the size of the board to a size by which it can easily fit on the back of someones hand replacing and manually routing was required.	
Issues:	Too many connection coming from the pin header with too narrow space between them to connect. Soultion: Trial and Error and connecting one Side of the chip to VCC and the other side to GND. Some airwires (not not connected parts which should be connected) near parts which could not be resolved. Solution: Replace faulty parts with the right ones.	
Result:	PCB wich was ready to solder after ordering.	

Hardware: Solder the board (two times)		
	Fabian. 24.01.2018, 25.01.2018, 31.01.2018 - 02.02.2018	
Time in hours:	Estimated: 0 hours	Actually: 30 hours
Task description:	Soldering the board with its components	
Issues:	We had trouble getting the board to work, problem was a missing wire connection. Then we accidently destroyed the first board and had to make another one without the direct help of Michael.	
Result:	Working board.	

Hardware: Burn bootloader to second board

	Fabian, Frieder. 02.02.18	
Time in hours:	Estimated: 0 hours	Actually: 8 hours each
Task description:	To burn the bootloader to the chip we had to direcly solder cables to the board which we then connected to a programmer device. Also exessive testingng with a multimeter, that there are no bridges.	
Issues:	We did not have a programmer. Solution: we bought one. It was very difficult to solder the wires which were so near to each other, approach use a magnifying glass. We got errors while trying to burn the bootloader. We used Amtel Stuio instead of the Arduino GUI, used a programm called zardig to install the right drivers to the computer and reconnected some of the cables to the board.	
Result:	The board now behaves like a Arduino Leoanrdo and can be programmed via USB.	

Hardware: Select battery		
	Fabian, Frieder. 09.11.17 - 07.12.17	
Time in hours:	Estimated: 2 hours each	Actually: 1 hour each
Task description:	Selct a battery which fits to the board, regarding size, voltage and capacity.	
Issues:	Flat battery would be bigger than board, and to be orderd which would also take time. Solution: Use a small round battery which Michael alredy has	
Result:	Small round battery which can be attached to the side of the board.	

Hardware: Design of case for hardware

	Frieder. 14.12.17 - 18.01.18	
Time in hours:	Estimated: 5 hours	Actually: 36 hours (5 hours case for board, 5 hours later discarded battery housing, 12 hours discussing desired/needed changes and implement them, 8 hours inspecting test print and adjusting model, 6 hours mend 2nd print)
Task description:	Make a case which houses the board, carries tracking, a battery and can be easily mounted to the glove.	
Issues:	The tracking used on the old glove was too big for our goal of minimizing the size of the glove. Also even though it seemed 3D printed, no 3D models were avaiable, another problem was that the tracking cannot be printed directly on the case because that would require too much support matertial. So as a solution a new tracking had to be moddeled, printed and attached to the rest of the case. Adding a Battery underneath the board would make the case around 4 to 5 cm tall. It would also hinder the already difficult mounting mechanism. To solve these issues, a smaller, round battery will be attached to the side of the case, and instead of screwing the case to the glove the case will be held in place by magnets attached to the case and metal strips attached to the glove. When evalutating the test print the board did not fit in the case and the tracking had severe printing error, so the position of many holes could not be correcly evaluated. Because the first real print crashed the printer, there was only time left for a second print. Approach: Tthe measures which could be evaluated were corrected and the others were ment after the second succesful print.	
Result:	A 3D printed case with attachable tracking and battery.	

Hardware: Assemble Case

	Frieder. 28.01.18 - 05.02.18	
Time in hours:	Estimated: 3 hours	Actually: 14 hours (3 hours connecting battery case, 2 hours magnets, 3 hour tracking, 2 hours going to the hardware-store, 4 hours trying to fit a switch to the board)
Task description:	Solder Cables to Baterry case, screw battery case and tracking to the rest of the case. Also place and screw the board in the case, add tracking points to the tracking holders and glue magnets in their place.	
Issues:	Cables of battery did fit through their hole in the case but were to thick with shrinking tubes. Solution: enlarge the holes. The support materials from printing did not come out properly out the wells for the magnets, so they didn't stayed in place. Solution: Carefull remove glue and support material and reattach magnets. Could not allocate the right screws. Approach: Go to two different hardware stores, buy the rest of the internet. The first switch was way too high. Approchach: Second switch, it was lower but very difficult to fit on the board. Because the switch was soldered on the fist board and too would only have fit in the case if shortened, the second board was not equipped with a switch at all and is turned off by removing the easy removable battery.	
Result:	A fully assembled case.	

Hardware: Connect sensors and joystick to a plug strip

	Frieder. 28.01.18 - 03.01.18	
Time in hours:	Estimated: 2 hours	Actually: 10 hours
Task description:	To connect the bending, fingertipsensors and joystick easily to the board, they all have to be selodered to a plug which then can be connected with the pin socket of the board.	
Issues:	We used a very thin wire wich was, before connecting, sewed to the glove. So it was difficult to connect on the one side to the sensors and on the other side to the plug also the wire did break very easily. Solution approach: Use longer shrinking tubes and be very carefull.	
Result:	Glove with attached sensors and one plug to connect them all to the case.	

Hardware: New Glove Design		
	Tanja. 14.12.17 - 25.01.18	
Time in hours:	Estimated: 5 hours	Actually: 6 hours
Task description:	Selection of new Cyber Glove, sewing, cabling, soldering.	
Issues:	None.	
Result:	New glove for CyberGlove	

Hardware: 3D printing		
	Tanja. 01.02.18	
Time in hours:	Estimated: 1 hour	Actually: 1 hour
Task description:	3D Printing of the housing and tracking system privately; the first case (printed at IMI) did not fit and while printing the second case the printer broke, so we had to find a possibility to print the case somewhere else.	
Issues:	None.	
Result:	Successfully printed and suitable housing and tracking system.	

2.2.2 Software tasks

Software: Virtual machine setup		
	Johanna, Lukas. 09.11.17 - 07.12.17	
Time in hours:	Estimated: 5 hours each	Actually: 5 hours each
Task description:	We wanted set up a virtual machine with PolyVR, VRPN and git installed to be able to work from home and to simulate the Glove. The virtual machine could also be used by other VR Project groups in following semesters.	
Issues:	The building process of PolyVR took much longer than expected in the VM environment. At first the simulation of the Cyberglove did not exactly work out the way we wanted it to, but then we discovered the tool `netcat` by which the data sent by the Cyberglove can be simulated easily.	
Result:	The result is a working VM which makes working more flexible and more convenient.	

Software: Revise WLAN transmission protocol		
	Johanna, Lukas. 09.11.17 - 16.11.17	
Time in hours:	Estimated: 2 hours each	Actually: 1 hour each
Task description:	Our task was to decide whether we want the Cyberglove to communicate via ZigBee or Wifi (VRPN Protocol).	
Issues:	None.	
Result:	We decided to use Wifi. The reasons for that are that the team from last semester already used Wifi so we knew it would work. Furthermore Wifi offers the opportunity to deliver more data in a single data package than ZigBee. With ZigBee we were not sure whether we could transfer enough data (4 bend sensors, 4 buttons and 2 channels for the joystick), so we chose Wifi.	

Software: Implement VRPN library on Arduino

	Johanna, Lukas. 09.11.17 - 16.11.17	
Time in hours:	Estimated: 10 hours each	Actually: 5 hours each
Task description:	Our task was to decide whether it is possible to implement the VRPN library completely on the Arduino and if possible, to do so. The current solution was to run a server application on the Cave computer to transfer data between the Cyberglove and PolyVR. In order to evaluate this problem correctly we had to become familiar with the VRPN protocol.	
Issues:	Since the Cyberglove team from last semester already wrote down in their documentation that the Arduino is not compatible with VRPN since the documentation of VRPN is not that good and due to the fact that it would have taken a massive effort to rewrite the VRPN library for our chip we decided to use the server application instead, as did the team in the previous semester. So we decided not to implement the VRPN library. Instead we inserted a command in the PolyVR applications that use the Cyberglove so that the server is started automatically every time the application is started. Something future teams could fix is to shorten the time it takes for the Cyberglove to build up a connection to the server (this currently takes about a minute).	
Result:	Our result is a more or less elegant workaround for the problem.	

Software: Implement software for new hardware		
	Johanna, Lukas. 14.12.17 - 25.01.18	
Time in hours:	Estimated: 10 hours each	Actually: 5 hours each
Task description:	Since we were going to have a new circuit board the software had to be adapted. For example, pin numbers changed and some bending sensors will be added. In order to make use of those our task was to adapt the hardware software.	
Issues:	None.	
Result:	A working board that is able to address all the new bending sensors and to pass on those values to other applications.	

2.2.3 Use case tasks

Use-Case: Getting used to PolyVR		
	Ran, Shant. 09.11.18 - 23.11.2018	
Time in hours:	Estimated: Ran: 30 hours Shant: 5 hours	Actually: Ran: 35 hours Shant: 5 hours
Task description:	Learning how to use PolyVR.	
Issues:	When building the basic environment and adding physic elements, I did not know how to realize it in PolyVR. Then I tried to ask Viktor and Polina for help (Ran). The Marsrover example was not so helpful in my case to get familiar with the PolyVR. Afterwards things were more clarified through the examples and through Victor's and Polina's explanations whenever I asked them. (Shant)	
Result:	Basic konwledge of the use case development in PolyVR.	

Use-Case: First steps with blender		
	Tanja, Johanna, Frieder. 11.01.18 - 25.01.18	
Time in hours:	Estimated: Tanja: 3 hours, Johanna: 2 hours, Frieder: 0 hours	Actually: Tanja: 3 hours, Johanna: 1 hour, Frieder: 1 hour
Task description:	Getting to know blender (with tutorials) and looking for furniture made in blender (Tanja). Adapting hand poses and position (Frieder, Johanna).	
Issues:	None.	
Result:	CAD of furniture and basic knowledge blender (Tanja), appropriate model of virtual hand for Line Drag use case.	

Use-Case: Line Drag

	Johanna, Lukas. 02.12.17 - 08.02.18	
Time in hours:	Estimated: 40 hours each	Actually: 45 hours each
Task description:	The goal was to implement a game using all buttons and sensors of the Cyberglove. The idea was to develop an application similar to the game “Hot Wire” where the player has to drag a ball along a given line.	
Issues:	We developed the game using PolyVR. Some issues that occured were that we needed the Cyberglove itself to test the full functionality and the fact that the glove only connects to the PC in the Cave since the PC’s IP Adress has been hardcoded in the Arduino Code. But there were no bigger issues, the implementation in general just took us much time.	
Result:	We were able to implement what we wanted to: a fully functional game which uses all the buttons of the Cyberglove. We implemented features like showing penalties and time to complete the level.	

Use-Case: Model hand poses for Line Drag		
	Frieder. 23.01.18	
Time in hours:	Estimated: 6 hours	Actually: 6 hours
Task description:	Model a 3D hand in the given (by photos) poses, using a template in blender.	
Issues:	Fingers touching each other was not possible using the controlls given by the template. Solution: A deeper modification inside the model and meticulous adjustments so that the hand still looks normal were needed. Exporting the model was also exporting "bones" which should not be seen in the final export. Soultion: Find an option to only export the selected objects.	
Result:	8 hands in different poses in the collada format	

Use-Case: The towers of Hanoi

	Fabian, Johanna, Lukas. 09.11.17 - 08.02.18	
Time in hours:	Estimated: Fabian: 20 hours, Johanna: 0 hours, Lukas: 0 hours	Actually: Fabian 20 hours, Johanna: 2 hours, Lukas: 2 hours
Task description:	Getting into working with Blender and PolyVR and create the towers of hanoi usecase (Fabian). Lukas and Johanna only contributed with some issues with the glove.	
Issues:	Working with PolyVR's physics engine, importing models from Blender.	
Result:	The towers of hanoi usecase.	

Use-Case: Ball play

	Ran, Johanna, Lukas. 01.01.2018 - 01.02.2018	
Time in hours:	Estimated: Ran: 40 hours Johanna: 0 hours Lukas: 0 hours	Actually: Ran: 35 hours Johanna: 0.5 hours Lukas: 0.5 hours
Task description:	Getting into working with Blender and PolyVR and create the ball in sliding usecase. (Ran) Johanna and Lukas just helped with some issues integrating the Cyberglove.	
Issues:	When I finished the 3D Models, they could not be imported correctly in PolyVR (crashed all the time). After adding the UVMap to the models in Blender, it worked. Also I wanted to enable the user to throw the ball, but there is no existing physics acceleration model in PolyVR. It is too hard for me to create one. So I made a sliding board to realize the movement for ball. (Ran)	
Result:	The ball play usecase.	

Use-Case: Glove Menu

	Shant. 23.11.17 - 05.02.18	
Time in hours:	Estimated: 20 hours	Actually: 50 hours
Task description:	Building a use-case for surfing on the web in VR.	
Issues:	To build any use-cases it was necessary to get familiar with PolyVR, Blender, Python, HTML and Javascript. I have learned some Python, HTML and Javascript with online tutorials for beginners. I went through online examples to learn how to use Blender for making 3D objects with Texture and exporting them as Collada, in a way that can be used in PolyVR. The unsolved problem in this use-case was using a virtual keyboard to write in the opened websites. This was not successful. A new virtual keyboard was built that has a textarea in its own window, and text is written there. The text could be copied by the help of a button on the virtual keyboard but could not be pasted in another window without the help of a physical keyboard.	
Result:	This use-case is ready and surfing through websites is working well.	

Use-Case: Wire and Loop		
	Shant. 23.11.17 - 05.02.18	
Time in hours:	Estimated: 10 hours	Actually: 20 hours
Task description:	Building a 3D game.	
Issues:	To build this use-cases it was necessary to get familiar with game physics. I have read about game physics, collision detection, rigid and solid bodies, degrees of freedom etc. Most of the physics engines online have their software and they use functions that work on their sofware, which can not be applied in PolyVR directly. PolyVR has its own functions that one has to get familiar with by asking and trying. This use-case did not reach its final goal because we could not find a way to make it work as was hoped. Could not move a 3D object (ring) along a path (curved tube) with simultaneous rotation around its own center.	
Result:	This use-case was left out.	

Use-Case: Breakout		
	Shant. 23.11.17 - 05.02.18	
Time in hours:	Estimated: 10 hours	Actually: 10 hours
Task description:	Building a javascript game.	
Issues:	Javascript runs in PolyVR quite smoothly. Some minimal errors appeared and were fixed.	
Result:	This use-case works well.	

Use-Case: Maze		
	Shant. 23.11.17 - 05.02.18	
Time in hours:	Estimated: 10 hours	Actually: 15 hours
Task description:	To create another 3D game, where a board could be tilted to make a ball move within a maze and be directed toward an exit.	
Issues:	To make the tilting possible a cone was situated below the board and was programmed so that its pointed head functions s a joint, which allows the board to tilt freely. This joint did not function as planned and the game could not be played, despite many tries and follow-ups.	
Result:	This use-case was left aside.	

Use-Case: Fruit versus Fastfood

	Shant. 23.11.17 - 05.02.18	
Time in hours:	Estimated: 20 hours	Actually: 25 hours
Task description:	Building a javascript use-case that exhibits the functionality of the Cyberglove in grabbing things and moving them around. The game was created according to a tutorial from a book. A sling should be grabbed and pulled and directed toward the target and then released.	
Issues:	The issue here was that the game ran on Mozilla browser, but not in the browser within the PolyVR. The problem was solved by changing the expression "let" with "var" within the Javascript, and by disabling the audio lines within the code.	
Result:	We have a functional game.	

2.2.4 General effort

General: Group Meetings		
	All members. 09.11.17 - 08.02.18	
Time in hours:	Estimated: 13 hours per person	Actually: 19 hours per person
Task description:	Weekly group meeting to discuss questions and results of each sub group.	
Issues:	None.	
Result:	The group meetings helped us improve the organization and achieve teamwork and communication skills.	

General: Lectures

	All members. 19.10.17 - 08.02.18	
Time in hours:	Estimated: 11 hours per person	Actually: 19 hours per person
Task description:	Organizational information, demonstrations in cave and of all VR projects, presentations of homework, group project plan, mid term presentations and final project presentation.	
Issues:	None.	
Result:	Teamwork skills, communication, overview of current project state.	

General: PolyVR Lab / Tutorial		
	All team members. 19.10.17 - 02.11.17	
Time in hours:	Estimated: 3 hours per person	Actually: 3 hours per person
Task description:	Introduction in PolyVR with exercises.	
Issues:	None.	
Result:	Basic skills in PolyVR.	

General: Seminar with Michael and preparation of presentation and Gantt chart		
	All team members. 19.10.17 - 02.11.17	
Time in hours:	Estimated: 3 hours per person	Actually: 3 hours per person
Task description:	Understanding the project tasks.	
Issues:	None.	
Result:	Clear task definition for the project, method and working steps, decision for WLAN (VRPN) as transmission protocol, structure project plan presentation, Gantt chart.	

General: Preparation of homework presentation

	All team members. 19.10.17 - 26.10.17	
Time in hours:	Estimated: Tanja: 4 hours, Johanna: 5 hours, Lukas: 5 hours, Fabian: 3 hours, Frieder: 3 hours, Ran: 4 hours, Shant: 15 hours	Actually: Tanja: 8 hours, Johanna: 8 hours, Lukas: 8 hours, Fabian: 2 hours, Frieder: 8 hours, Ran: 5 hours, Shant: 15 hours
Task description:	Each team member prepared his or her presentation about the individually assigned topic.	
Issues:	None.	
Result:	Presentation about the respective topic.	

General: Preparation of 1st presentation

	All team members. 02.11.17 - 09.11.17	
Time in hours:	Estimated: Tanja: 5 hours, Johanna: 3 hours, Lukas: 2 hours, Fabian: 1 hour, Frieder: 2 hours, Ran: 1 hour, Shant: 1 hour	Actually: Tanja: 5 hours, Johanna: 1 hour, Lukas: 2 hours, Fabian: 1 hour, Frieder: 2 hours, Ran: 1 hour, Shant: 1 hour
Task description:	Development the objectives for the project term, dividing the objectives in sub groups and scheduling the task as Gantt chart.	
Issues:	None.	
Result:	Presentation.	

General: Preparation of midterm presentation

	All team members.. 07.12.17 - 14.12.17	
Time in hours:	Estimated: Tanja: 4 hours, Johanna: 3 hours, Lukas: 2 hours, Fabian: 2 hours, Frieder: 2 hours, Ran: 10 hours, Shant: 2 hours	Actually: Tanja: 4 hours, Johanna: 2 hours, Lukas: 4 hours, Fabian: 2 hours, Frieder: 2 hours, Ran: 12 hours, Shant: 7 hours
Task description:	Preparation the mid term presentation with a summary of fulfilled tasks refering to time schedule (Gantt chart) and giving an outlook of the next steps and tasks.	
Issues:	None.	
Result:	Presentation.	

General: Preparation of final presentation		
	All team members.. 01.02.18 - 08.02.18	
Time in hours:	Estimated: Tanja: 3 hours, Johanna: 2 hours, Lukas: 2 hours, Fabian: 3 hours, Frieder: 2 hours, Ran: 5 hours, Shant: 15 hours	Actually: Tanja: 3 hours, Johanna: 3 hours, Lukas: 2 hours, Fabian: 3 hours, Frieder: 2 hours, Ran: 5 hours, Shant: 15 hours
Task description:	Preparation of final presentation, accomplished goals, outlook of future development possibilities of the CyberGlove. Preparation of Use Case presentation in the Cave.	
Issues:	None.	
Result:	Presentation.	

General: Documentation of our work

	All team members.. 25.01.18 - 08.02.18	
Time in hours:	Estimated: Tanja: 8 hours, Johanna: 10 hours, Lukas: 10 hours, Fabian: 10 hours, Frieder: 5 hours, Ran: 9 hours, Shant: 5 hours	Actually: Tanja: 8 hours, Johanna: 15 hours, Lukas: 15 hours, Fabian: 8 hours, Frieder: 6 hours, Ran: 7 hours, Shant: 5 hours
Task description:	Documentation of all of our tasks that we worked on during the semester. We also described our learning outcomes, issues and project results.	
Issues:	None.	
Result:	Full documentation to hand in.	

2.2.5 Coordination and management

Coordination: Project Management and Coordination		
	Tanja. 09.11.17 - 08.02.18	
Time in hours:	Estimated: 5 hours	Actually: 5 hours
Task description:	Communication, organization, coordination.	
Issues:	None.	
Result:	Fulfilling of tasks in time, good communication.	

Coordination: Gantt chart		
	Tanja. 02.11.17 - 08.02.18	
Time in hours:	Estimated: 4 hours	Actually: 6 hours
Task description:	Creating and updating Gantt chart.	
Issues:	None.	
Result:	Updated Gantt chart.	

Coordination: Buying new glove		
	Tanja. 25.01.18	
Time in hours:	Estimated: 1 hour	Actually: 1.5 hours
Task description:	Buying new glove.	
Issues:	None.	
Result:	New glove.	

Coordination: Preparation of project video		
	Tanja. 14.12.17 - 08.02.18	
Time in hours:	Estimated: 15 hours	Actually: 25 hours
Task description:	Getting used to work with Adobe Prime, video shoot, editing.	
Issues:	None.	
Result:	Finished Project video.	

Coordination: Creating poster and flyer		
	Tanja. 14.12.17 - 08.02.18	
Time in hours:	Estimated: 6 hours	Actually: 8 hours
Task description:	Creating flyer and poster with pictures and information about the Cyberglove.	
Issues:	None.	
Result:	Flyer and poster of CyberGlove project.	

3 Hardware

The following chapter deals with the hardware used for the cyberglove. It will explain how the new developed circuit board works and why certain decisions were made in this way.

3.1 Hardware selection and board design

The old version of the cyberglove used an Arduino Uno board with a breakout chip for wireless data transmission. It also has an external battery in form of a typical smartphone power bank. Wiring is realized by a breadboard which is equipped with resistors and connects the glove with its components to the GPIO pins of the Arduino Uno. This setup needs a lot of space and is also heavy and unwieldy. In order to get a smaller and more light processing unit for the cyberglove, we need to develop a new hardware basis. The basic functions of the glove should be preserved, so we orient ourselves to the development of the original version.

The Arduino Uno in the previous cyberglove uses a ATmega328 microprocessor which provides 16 MHz and a total amout of 20 GPIO pins, six of them are analog pins and the rest are for digital input and output. We decide to use the ATmega32U4 since it's similar to the ATmega328 and can be programmed in the same way with the Arduino IDE. Just like the ATmega328, the ATmega32U4 has a clock frequency of 16 MHz and provides up to 32 I/O pins.

We use an Li-Io battery cell with 1200 mAh to power the circuit board. The battery provides 3.7V, so we need a TPS63031 converter which converts the voltage to a constant 3.3V supply voltage for the ATmega32U4. Because we also want to charge the battery over our USB connector, we use the BQ24090 microchip to adjust the charging current. According to the following image, we set the maximum charging current to 500mA by installing a 1kOhm resistor.

After we soldered the board, we noticed that the battery was not charging. The measured charging curret is 150mA instead of our calculated 500mA. We were not able to track down the source of this error so this function will unfortunately be not available.

For the Wi-Fi connection we use the same ESP-12E module as the previous cyberglove used. This way we could bui lt up on the previous groups work and had the chance to improve it, instead of reinventing the data transfer, wasting resources.

The Wi-Fi module connects the hardware with the CAVE on startup and sends the evaluated output signal of the glove constantly to the CAVE server application.

After selecting the main hardware elements, we created a sketch in Eagle and wired the corresponding connections. Unfortunately, we made a spelling mistake with the USB connection of the microcontroller. That's why there needs to be an additional connection between pin 7 "VBUS" and the USB connector "VUSB".

The final wiring can be seen in the images 3, 4 and 5.

We accidently destroyed the first board because the battery was turned upside down, which caused the TPS63031 to break. That's why we had to solder another board. Using a diode would have avoided this problem because it only lets through the current in a predetermined direction and blocks the flow of current in case of accidental reverse polarity.

We decided to use a bar of connector pins to connect the bendsensors and the fingercontacts with the circuit board. This allows the user to connect the glove with the board easily and disconnect both parts from each other for e.g. reprogramming the board without taking the glove itself with you. Image 6 shows the pin assignment of the pin

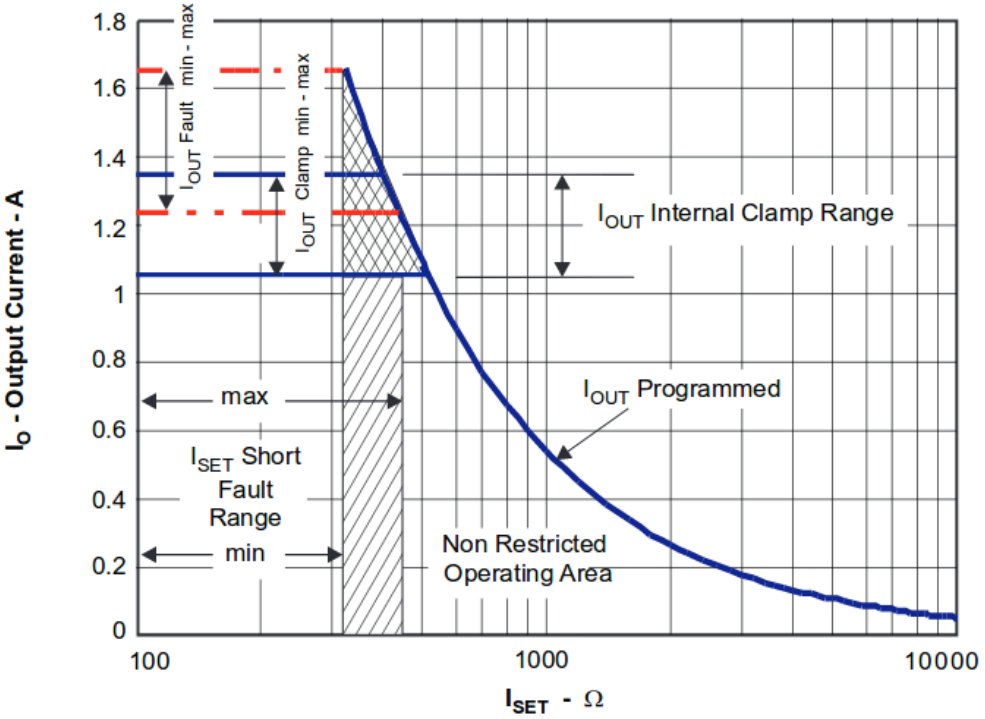


Figure 2: Programmed/Clamped Out Current

bar.

3.2 The Cyberglove

After soldering the hardware together, we could not communicate with the board yet, we first had to burn the bootloader in the Atmega chip. To do so we used a so-called programmer, soldered cables to the corresponding pins of the board, plugged these in the cables in the programmer and the programmer in the computer. With Amtel Studio we then could burn the bootloader and set the fuses, so that after that we could communicate with our board via USB as if it was an Arduino Leonardo.

The glove itself contains the following elements:

- Fingercontacts on all fingers
- Four bendsensors
- A two-axis joystick
- Steel sheets for mounting the hardware case

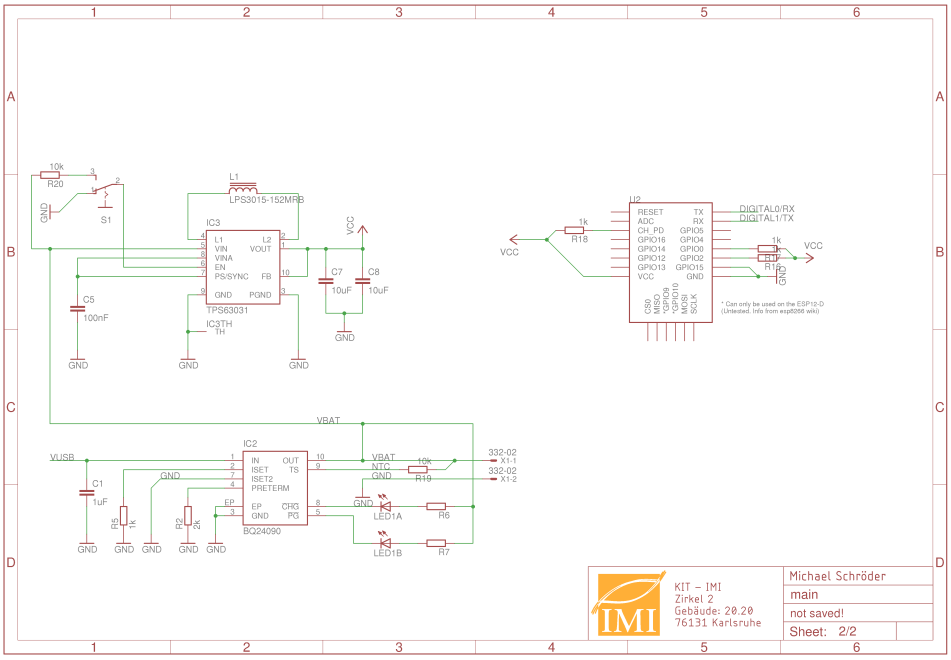


Figure 3: Final wiring of the board.

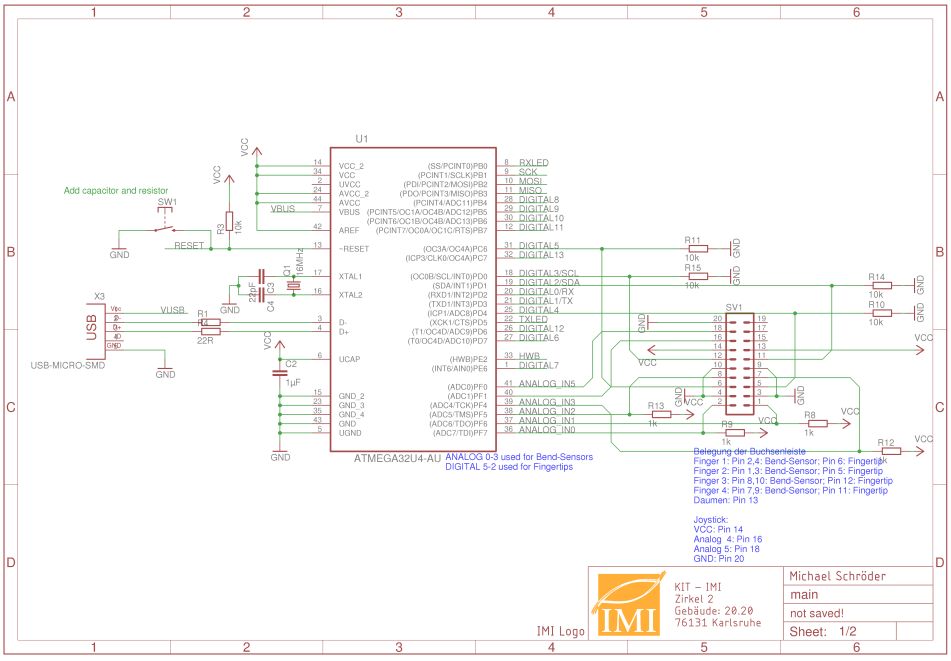


Figure 4: Final wiring of the board.

Putting the fingercontacts together with thumb closes the electric circuit and sends a "high" signal to the microcontroller. These signals can be used by the software to function

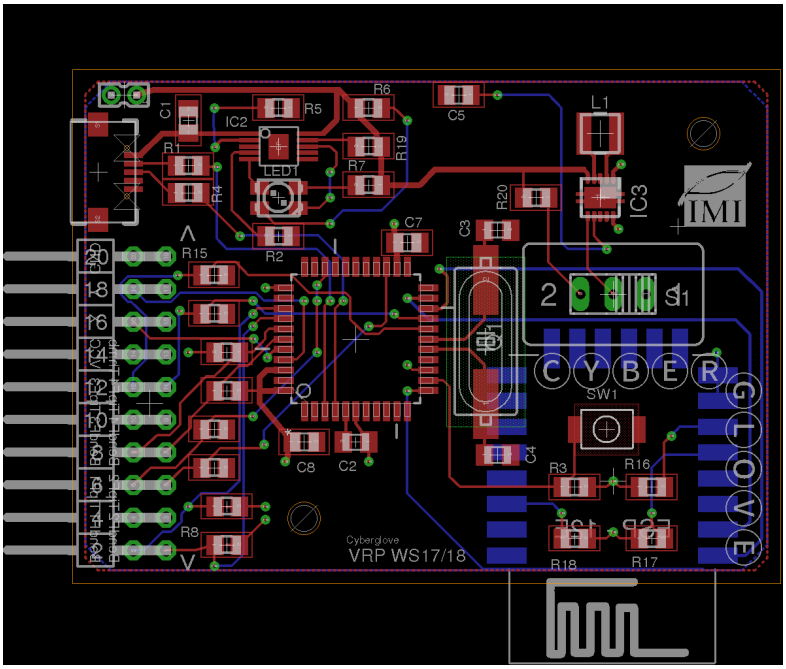
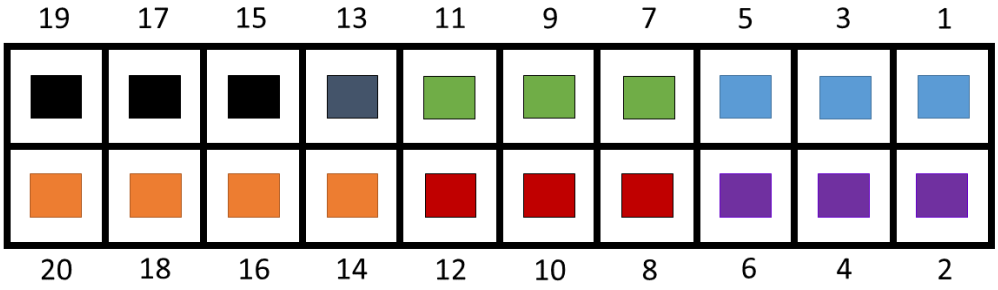


Figure 5: Final wiring of the board



1. Finger 1: Bendsensor GND
2. Finger 2: Bendsensor VCC
3. Finger 1: Bendsensor VCC
4. Finger 2: Bendsensor GND
5. Finger 1: Finger contact
6. Finger 2: Finger contact
7. Finger 3: Bendsensor VCC
8. Finger 4: Bendsensor VCC
9. Finger 3: Bendsensor GND
10. Finger 4: Bendsensor GND
11. Finger 3: Finger contact
12. Finger 4: Fingercontact
13. Thumb finger contact
14. Joystick VCC
15. UNUSED
16. Joystick: Analog X
17. UNUSED
18. Joystick: Analog Y
19. UNUSED
20. Joystick: GND

Figure 6: Assignment of the pin bar.

as four buttons, one for each finger. If one of the bendsensors is bent this increases its resistance, which can be evaluated by the software. This makes it possible to recognize

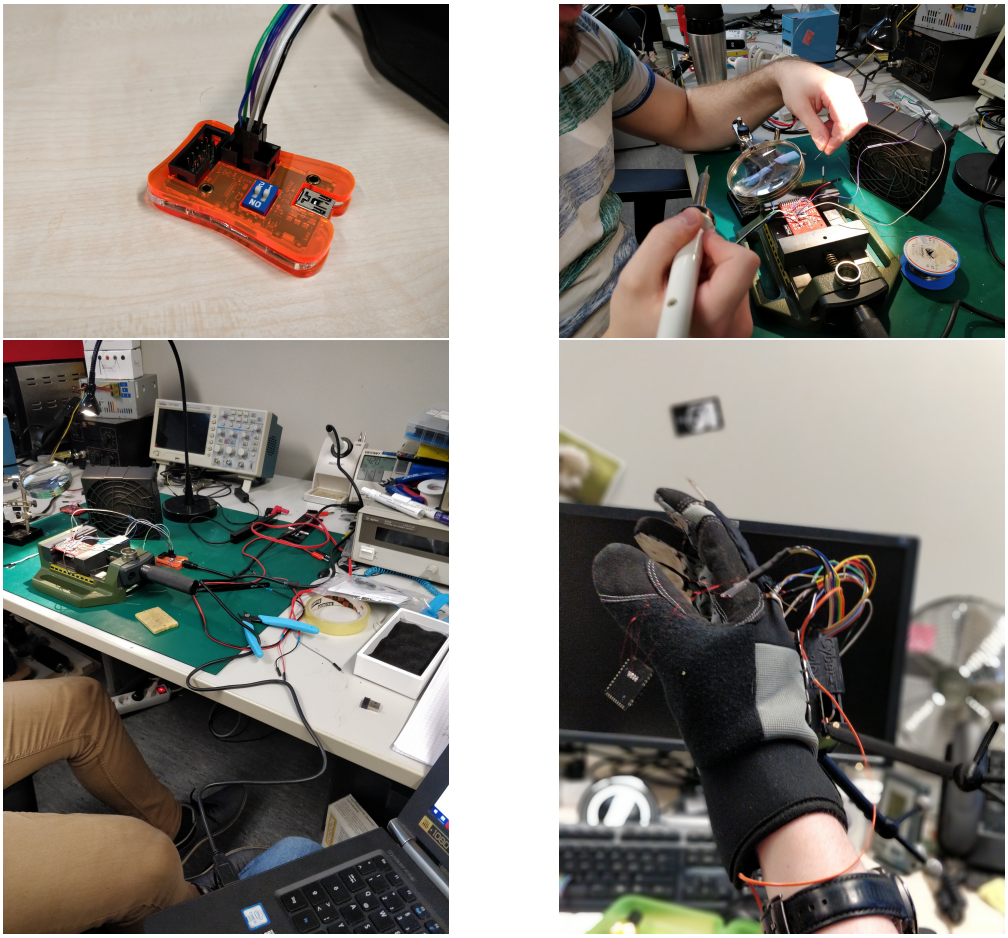


Figure 7: Work on the hardware.

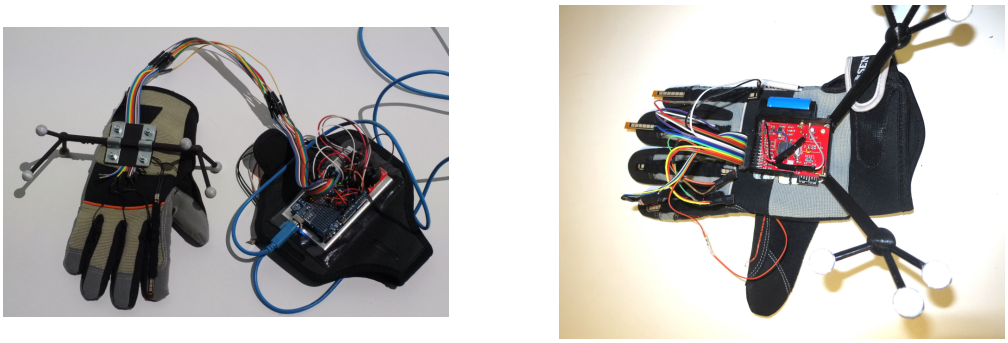


Figure 8: Comparison of the old and the new glove.

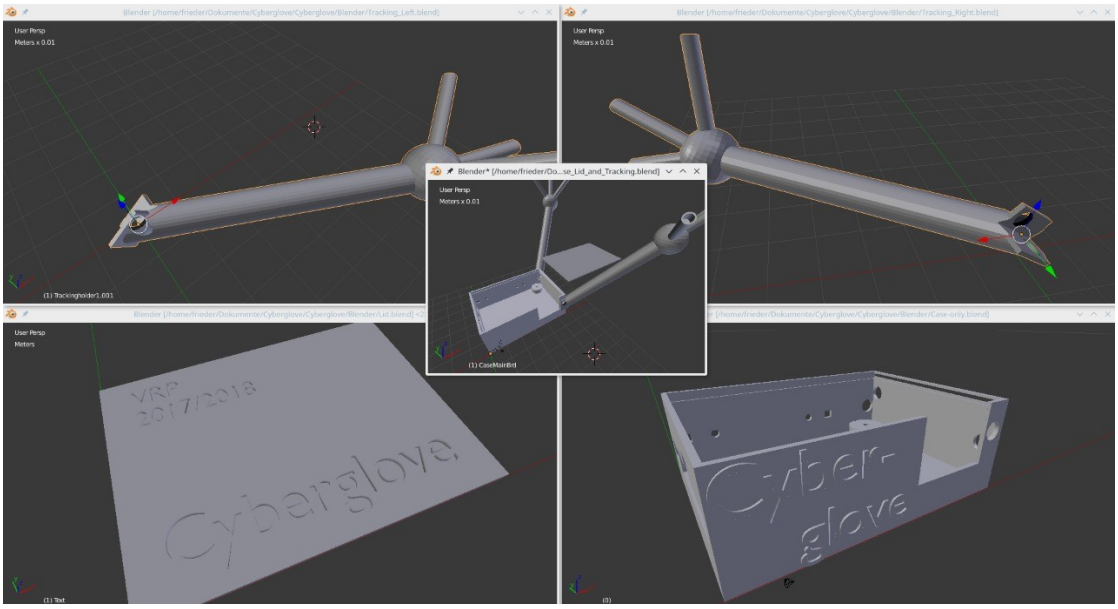


Figure 9: The case and its parts

different gestures made by the glove. We decided to put steel sheets on the back of the glove, so the case of the board can be mounted with the help of magnets.

3.3 Case Design

To design the case for the glove we used Blender, because it is easy to learn and to export. We started with a solid box of what size the final case should be, then made it hollow by subtracting a box the size of the board and and kept using the "boolean" operator to make holes for the pin header, wifi chip, etc. This way we also made a lid, which once fully inserted falls a millimetre to lock itself in place. After that we designed a housing for the flat battery which then could be inserted under the board. After discussing this design with the rest of the group we decided to discard this concept and instead attach a battery clip, which Michael provided, to the side of the case. For that we measured the exact position for the holes for screws and the pins on the clip and added appropriate holes to the model. We also found out we could not use the tracking of the old glove whith the new case because we could not mount it to the case. For this reason we designed two tracking arms which could be attached to the case and reused the tracking points of the old tracking mount. Other improvements to the case were pillars on which the board can rest and deepengings in which magnets can be placed to hold the case on the glove.

Size adjustmends had to be made, which were rather difficult to achieve in Blender and we learned that while blender makes modeling of pleasing looking 3D objects easy, it is not designed to make precision models of parts and we should rather have used CAD. We did end up with a case which fits our needs but probably could have saved a lot of work in the process by using the right software.

4 Software

4.1 Arduino platform and server code

Most of the hardware related programming code was implemented by the team from last semester, we have left much of it unchanged. The team has implemented the platform code for the Arduino controller, the server application which runs on the cave computer and forwards the TCP packages from the arduino to PolyVR via the VRPN protocol.

We have slightly modified the Arduino code to adapt the hardware changes that we have made to the new glove. The new bending sensors are being tracked and their data is properly being sent in the way the previous team has designed. The sensor data is collected on the Arduino and mapped into a TCP package, which contains the state of all sensors at a certain point in time. Those TCP packages are then being sent repeatedly as long as the glove runs.

The TCP packages are constructed as an eleven character long string, where each sensor information is encoded as one character. Boolean sensor values are just encoded as 0 (zero) or 1 (one), all other floating sensor values are just cast into characters. The packages have the following form:

Character 0 Touch sensor: Thumb, index finger

Character 1 Touch sensor: Thumb, middle finger

Character 2 Touch sensor: Thumb, ring finder

Character 3 Touch sensor: Thumb, little finger

Character 4 Virtual button

Character 5 Bending sensor: Index finger

Character 6 Bending sensor: Middle finger

Character 7 Bending sensor: Ring finger

Character 8 Bending sensor: Little finger

Character 9 Joystick in x-direction

Character 10 Joystick in y-direction

On the computer which runs the PolyVR application (which usually is the cave computer), the server application has to run to receive those TCP packages and forward them to PolyVR. PolyVR uses VRPN, a protocol designed for communication between virtual reality devices, to receive input data. The initial idea was to make the Arduino code use the VRPN protocol by itself and directly send the data to the VRPN interface of PolyVR. However, has the team from last semester has pointed out in their documentation, VRPN is not available for Arduino, and as we have found out it is nearly

impossible to develop a custom VRPN library which works on the Arduino, especially due to the way VRPN works. We have settled for using the server application the way it was designed by the last team, however added code to the use cases which at last automatically runs the server on start up of the use cases.

The touch sensors supply values of 0 or 1, corresponding to wether the thouch sensor is pressed or not. The bending sensors supply floating point values which are around zero when not bending the sensor, and go up to around 20 when bending the finger. We have used 10 as threshold to determine wether the sensor is bent or not. The joystick unfortunatly does not supply correct values at the moment. We are not sure if this is caused by incorrect wiring, a defect of the joystick or a problem in the arduino software, and could not find the causing of the error. This could be a follow-up for the next team.

In order to make a use case automatically run the server application, the following code has to be added at the beginning of the initialization script. Make sure that the server application has been built sucessfully and the path to the executable is correct.

```
1 os.system("cd ../../Rechner/Rechner/build && ./server")
```

4.2 Emulating the glove

Because the arduino controller uses a low level TCP connection to send the sensor data to the VRPN server application, we have found that it is very easy to emulate the glove for testing purposes in enviroments outside the cave, where the glove can not be used during developement.

For emulation, the linux tool `netcat` can be used. It is a command line tool which exposes many features regarding TCP commands. To start emulation, enter the following command into terminal while the server application runs in another terminal window:

```
1 $ netcat localhost 3490
```

This will initiate a TCP connection to the same computer (localhost) on port 3490, which is the port that the server application listens for. Then type `69` and hit enter. This is a hard coded message that the team from last semester has defined and for which the server application waits to make sure that the talking device really is the arduino application and not a random device which coincidentally also uses port 3490.

Then, strings with a length of exactly 10 can be typed and sent using the enter key, in order to send virtual sensor data to the server application. The sensor data encoded in the strings follow the format defined in section 4.1.

4.3 Virtual Machine

For the developement of the use cases in PolyVR we have set up a virtual machine, so that we could work on the code at home. The virtual machine uses Lubuntu as operating

system, a variant of Ubuntu capable of running all required software including PolyVR, but more sleek and smaller in size than Ubuntu. The virtual machine was then exported to a `.vdi` file so that other people could download and use it.

The fully configured virtual machine is available for download at `https://goo.gl/Pg1WCJ`. In Windows, the machine can be run using Oracles VirtualBox (`https://www.virtualbox.org/wiki/Downloads`). The login username and password are both `glovevm`.

The machine comes with a terminal and filebrowser preinstalled by the OS, as well as installations of CodeBlocks, PolyVR, the VRPN library, our project repository and GitKraken for managing the git repository, although we have used mostly terminal commands.

We have also written a seperate documentation for using the virtual machine in German. This document especially explains how to setup Oracles Virtual Box and how to import the `.vdi` image file. This documentation is available at the GitLab repository under "documentation/VMDocumentation".

4.4 User Manual

The following sections describe how to download the repository and get started with the use case applications.

4.4.1 Downloading the repository

The first step is to download the git repository from gitlab, which can easily be achieved using the `git clone` command. Then, change into the repositories directory using `cd`. All following guides assume that the repository is already downloaded and that the user has changed to the directory.

```
$ git clone http://imi-dev.imi.kit.edu:443/vrp1718/Cyberglove.git
$ cd Cyberglove
```

4.4.2 Repository directory structure

After downloading, the directory should have the following structure.

```
Cyberglove/
├── arduino-software
│   └── newcode
│       └── oldcode
├── Blender
└── documentation
    ├── assets
    ├── content
    └── VMDocumentation
```

```
├── Eagle
├── server
│   └── build
└── usecases
    ├── Ball
    ├── Breakout
    ├── Hanoi
    ├── linedrag
    ├── Maze
    ├── Memory
    ├── Menu
    ├── ObstgegenFastfood
    ├── skytest
    └── WireLoop
```

The directory `arduino-software/` contains the arduino code for the controller. It contains the subdirectories `oldcode`, which is the unmodified code from last semester, as well as `newcode`, which contains our modified code that has adapted to the hardware changes. The code files can be modified using the official Arduino IDE. A detailed documentation on the arduino software can be found in section 4.1.

The directory `Blender/` contains all Blender sketches that we have used during the project, including the hand models used in the linedrag use case and the case models that we have 3D printed.

`documentation/` contains the source LaTeX files for this document. The `.tex` files with the sections contents are put inside `documentation/content`, while `documentation/assets` contains files declaring LaTeX variables and dependencies. The subdirectory
`documentation/VMDocumentation` contains an explanation on how to use our virutal machine.

`Eagle/` contains the circuit board models which can be edited in AutoDesk Eagle.

Under `server/` the server application can be found. Its functionality is documented in section 4.1, a guide on how to use it is in section 4.4.3.

`usecases/` contains all use cases that we have implemented. Each subdirectory includes exactly one use case with its corresponding `.xml` or `.pvr` file, which can be opened in PolyVR.

4.4.3 Setting up and running the server application

When downloaded, the source code for the server application lies in the directory `Cyberglove/server`. Change to this directory and then into its build directory.

Then, run `cmake ..` and `make` to build the program and create an executable. This executable will be called `server` and can then be found inside the `Cyberglove/server/build` directory. It can be run by calling it by its name.

```
1 $ cd server/build
2 $ cmake ..
```

```
$ make
$ ./server
```

If the Arduino is not running yet, start it now. After the server application started running, press the Arduinos reset button to restart Arduinos program. The Arduino will then connect to the server via WIFI.

4.4.4 Running the use case applications

For all of the use case applications the user simply has to start the specific application in PolyVR. Then the server will be started automatically. Afterwards the user only has to press the reset button of the microcontroller and a connection with PolyVR will be established. If the reset button is out of reach for the user for example because of the cover of the case the user alternatively can take out the battery pack and put it back in again in order to reset the microcontroller. For the use case "Line Drag" the user has to run the `init` script manually one more time before the application runs properly.

5 Use Cases

5.1 Line Drag

The first use case we implemented was the game "Line Drag". The game idea is based on the game of skill "Hot Wire".

The goal of "Line Drag" is to drag a ball along a given path using the Cyberglove. On this path there are challenges. In order to pass those the player has to mimic the pose of the hand shown on the display.

The player can grab the virtual ball by bending pointing at the ball and bending the index finger. As long as the player manages to keep the ball close to or on the path the ball will have a green color. As soon as the distance between the path and the ball gets too large, the color of the ball changes to red. Whenever this happens the player's penalty points increase. Also while the player is dragging the ball along the line the time already passed since starting the game as well as the penalty points will be displayed.

The earlier mentioned challenges are spread across the path. They are marked as blue planes. As soon as the ball touches one of those planes, a challenge is being triggered. A virtual hand is shown on the display and the player has a certain amount of time to mimic the pose of this hand by using the Cyberglove. While the time runs, the color of the ball fades gradually from green to red. When the ball is red, the time is up and the player lost the challenge. For every lost challenge the player will receive penalties.

The game is over when the ball reaches the end of the path. Then the time needed and the fault points will be displayed.

Right now this game only has one level. An idea would be to create more levels with different difficulties in the future. Also a database could be attached to the game (if that is possible in PolyVR). Then the scores of different players could be compared and

a ranking could be created.

To determine wether the ball is touching the line or not, a set of position vectors defining the line are extracted and each position is compared with the position of the ball. If there is at least one position on the line, which has a distance to the ball with a value lower than a set epsilon value, the ball is considered to be touching the line.

The line is generated from an array of position vectors, which can be easily modified in order to change the line and therefore modify the level. This makes the creation of new levels very easy. This line defining array is instantiated in `Line.py`.

The challenges are also predefined in a array so that they can be easily modified. This array is instantiated in `Plane.py` and defines for each challenge its position as well as a set of hand configurations, which have to be imitated by the player in order to progress to the next part of the challenge, or in order to complete the challenge if they are on the final part.

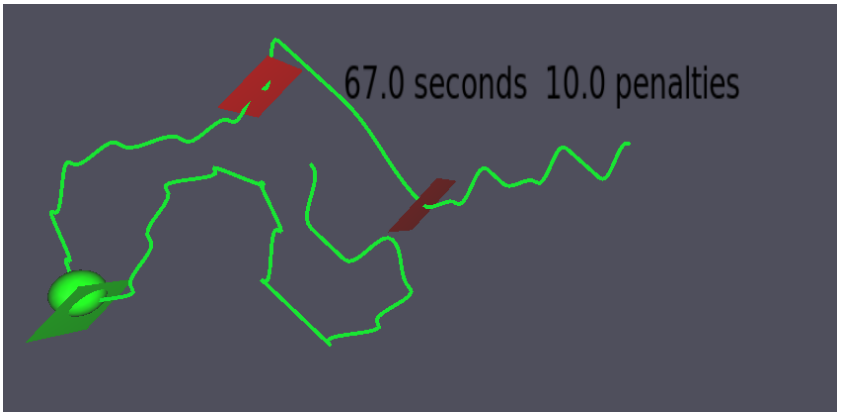
5.2 The towers of Hanoi

The tower of Hanoi is a mathematical puzzle which contains three pillars and a number of disks. When the game begins, the disks are in ascending order of size stacked on the outer left pillar. The goal of the puzzle is to move the stack from the left to the right pillar only by moving one disk at a time and no disk can be placed on top of a smaller disk. All three pillars can be used to achieve the objective.

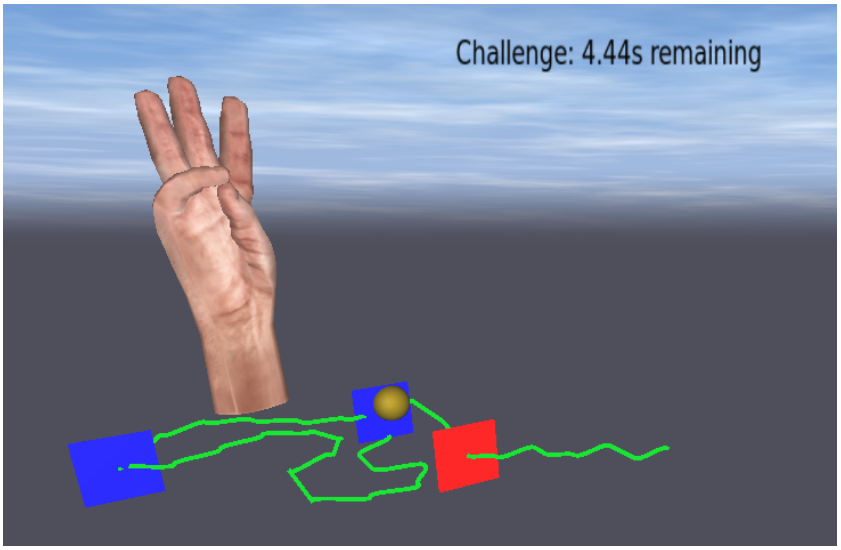
The number of implemented disks in the usecase is three. When the usecase starts the user finds himself in a small room with a table in front of him. The pillars with the disks stand on top of the table. The table itself was downloaded from `https://www.turbosquid.com`. The pillars and disks were designed in Blender. At first there was a problem with importing the model into PolyVR because there was no material assigned to the single parts of the models. The second challenge was to apply the physics engine to the objects. When we tried to use the standard `object.physicalize()` method with the parameters "convex" and "concave" there was a collision box rendered around the disks but it did not cut out the hole in the middle of the disks. Due to this missing hole the disks could not be put onto the pillars. With the help of Victor we could solve the problem by setting the physicalize parameter to "ConvexDecomposed" and changing the ConvexDecomposed parameters via trial and error to working values. The disks can be grabbed by the cyberglove. Since we already had the code for handling the cyberglove in PolyVR for the linedrag usecase, we just use the same code for the tower of Hanoi. In the end I wanted to implement a number of user interfaces for interaction but had no time to finish them because of the problems with the circuit board. In the future there could be functions to select the number of disks to play with or an algorithm which detects when you finish the game or broke the rules.

5.3 Menu

This use-case consists of several screens, designed with Blender, and arranged on two rows. The ones in the upper row are where web contents are presented and the bottom



(a) Line drag use case. The green ball is the handle, red planes are failed challenges and green planes are sucessfully completed challenges.



(b) During a challenge, the remaining time is shown, the handle changes color according to the remaining time (and fades from green to red), and the hand, which has to be mimiced is being shown.

(c) Before the game starts, an intro message is shown.

(d) Upon sucessfull completion of the game, a end message is shown.

Figure 10: Line Drag use case.

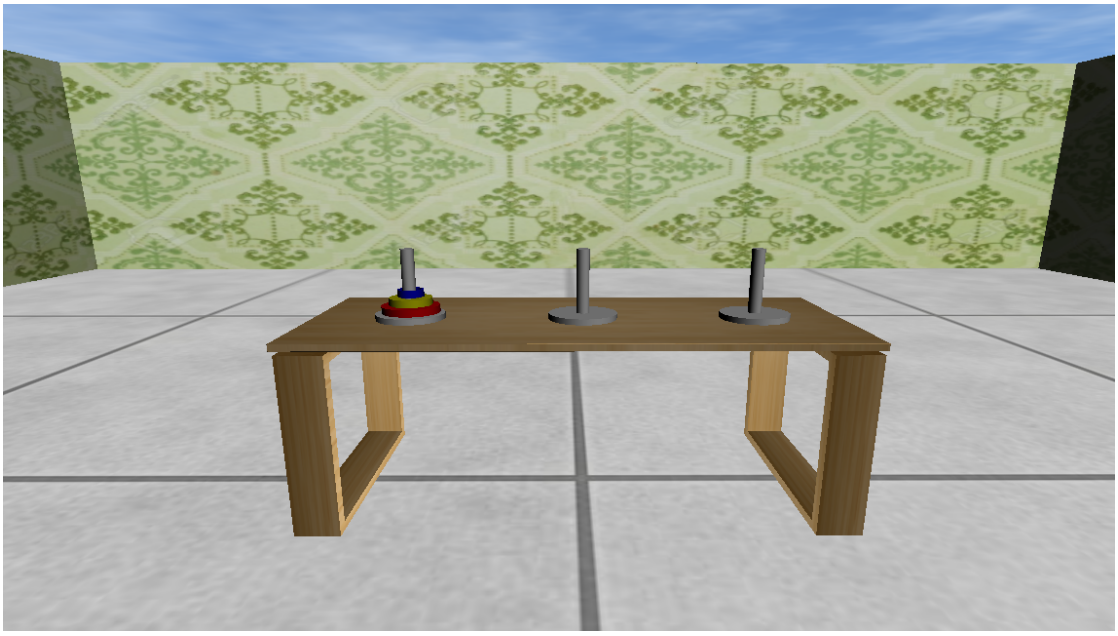


Figure 11: The towers of Hanoi use case

ones contain buttons to help the user switch between websites. There is also a virtual keyboard added to the scene to help the user type in text editing sections of the webpages, by merely using the Cyberglove.

The screens have both a cover and a face. The webpages and the buttons are designed and presented on the faces. This is done through a localhost. The buttons are designed by HTML as websites and then presented on the faces of the screens through a local server. These buttons are called and recognized by the initial code and accordingly they order the respective page to open on the respective page.

The virtual keyboard was in the beginning designed to write and function within another window, called Textarea. Despite of the many tries, allowing the keyboard to write on the webpages was not met with success. So the keyboard was changed to a simpler one, which contains a text area within its window. It is possible to copy the text written in this text area by means of a simple HTML code which could be applied to a button named „Copy Text“ on the keyboard itself. Again, to go to another webpage and to paste that text without the physical keyboard was not possible. Therefore, the virtual keyboard in this use-case is merely to write a text and exhibit the cyberglove's ability to fulfill this purpose.

5.4 Fruit versus Fastfood

This is javascript-based use-case that exhibits the functionality of the Cyberglove in grabbing things and moving them around. The game was built according to a tutorial from a book. A sling is grabbed, pulled, directed toward the target, and then released.

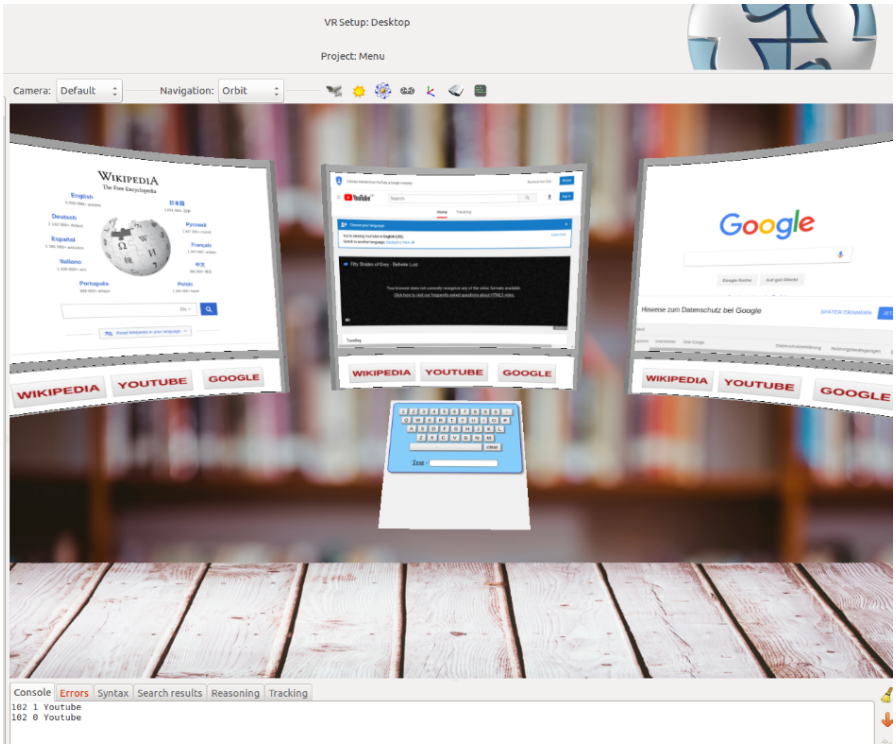


Figure 12: Menu use case

There are two levels in the game. It is very similar to the famous game “Angry Birds”, but here fruits are fighting the fast food items. One should throw a fruit by means of a sling and bring down the construction that holds the fast food items within. The game level is won when all the fast food items are disappeared from the screen. The game is lost when the fruits have disappeared before the fast food items.

The issue here was that the game ran on Mozilla browser, but not in the browser within PolyVR. The problem was solved by changing the expression "let" with "var" within the Javascript, and by disabling the audio lines within the code.

5.5 Maze

This was supposed to be 3D game, where a board could be tilted to make a ball move within a maze and be directed toward the exit.

The object was designed in Blender and exported and integrated within PolyVR, then physicalized. To make the tilting possible a cone was situated below the board, also via Blender, and was programmed so that its pointed head functions as a joint, which allows the board to tilt freely. This joint did not function as planned and the game could not be played, despite many tries and follow-ups.

Figure 13: “Fruit vs Fastfood” use case

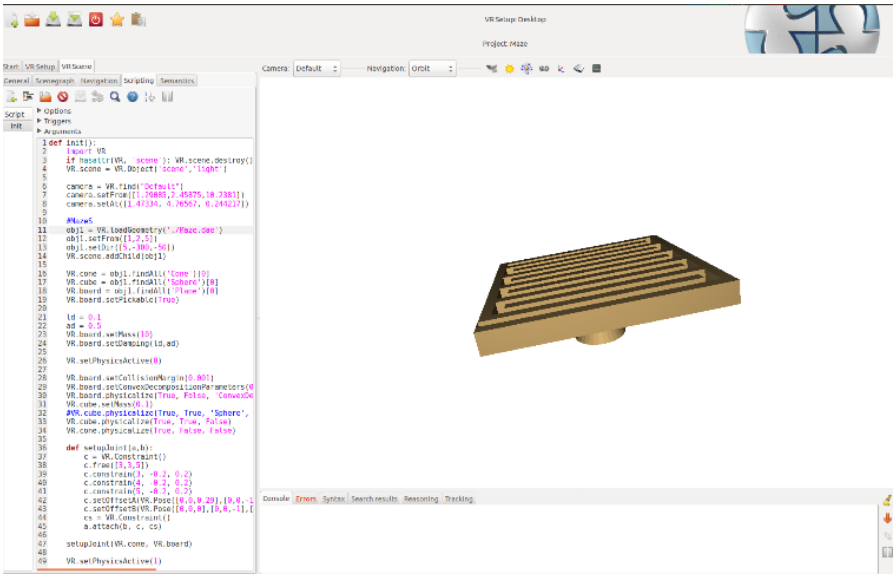


Figure 14: Maze use case

5.6 Breakout

A JavaScript game called Breakout, that uses collision detection to find out when the ball hits the walls or the bricks, and a paddle is moved through the screen by following

the mouse pointer or the Cyberglove location in the Cave.

This was built by following a JavaScript tutorial. The ball movement was programmed so that in every instance it draws a new ball and removes the old, so that shadows will not remain behind. The bricks will disappear once collided with the ball. The mouse movement was integrated so that the paddle follows the mouse cursor, without pressing any buttons. And one has 3 tries before the game is over.

Some minimal errors appeared and were fixed.

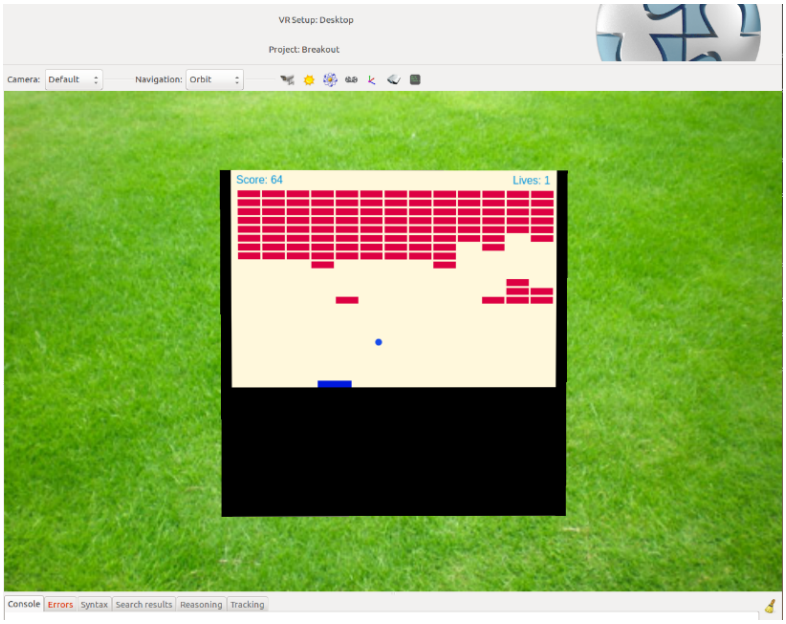


Figure 15: Breakout use case

5.7 WireLoop

To build this use-case it was necessary to get familiar with game physics. I have read about game physics, collision detection, rigid and solid bodies, degrees of freedom etc. Most of the physics engines online have their software and they use functions that work on their software, which cannot be applied in PolyVR directly. PolyVR has its own functions that one has to get familiar with by asking and trying. The drawings were done in Blender and exported to Collada, then integrated in PolyVR and were physicalized. However, this use-case did not reach its final goal because we could not find a way to make it work as hoped. We could not make a 3D object (ring) move along a path with simultaneous roation around its own center.

5.8 Ball play

The ball play is a tiny physics game. It has a ball, a slidingboard and number of cans. The ball is pickable, the user can pick it and drop it upon the slidingboard. This movement

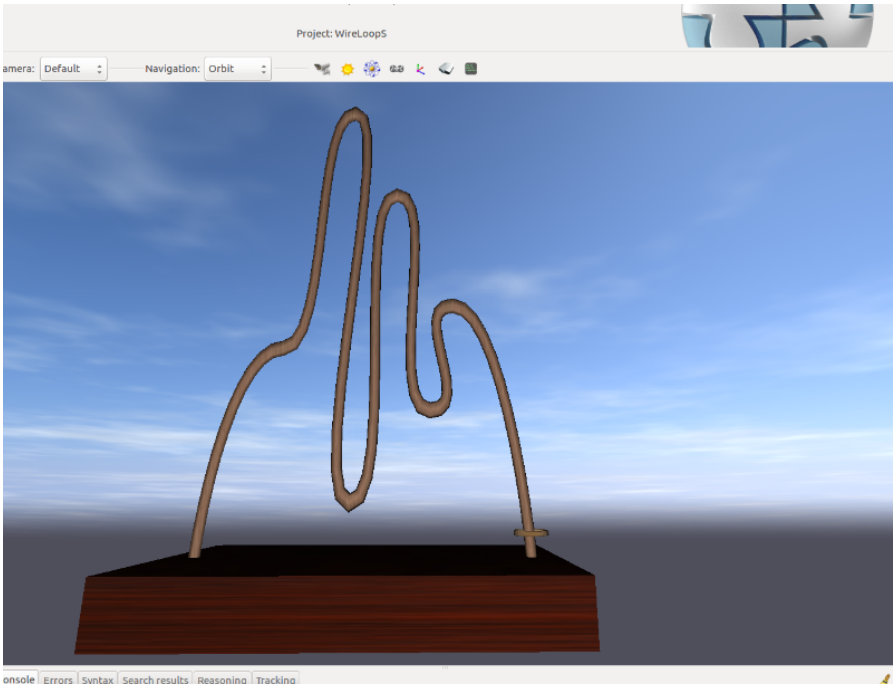


Figure 16: WireLoop use case

can give the ball a speed in the 3D space, which can be used to throw down the cans. The goal of this game is to throw down as many cans as possible.

The number of implemented cans in the usecase is ten, there is also one sliding board, one ball, and a table. When the usecase starts the user finds himself in a small room with a table in front of him. All of them have been setted in a good sight, which can be easily picked and dropped. All of the models were designed in Blender. Because it was pretty complicated, Victor helped to design the sliding board in Blender. At first there was a problem with importing the model into PolyVR, because the models did not have a UVMap, so each time PolyVR crashed. After adding the UVMap, it finally worked.

The second challenge is about how to let the ball fly in the 3D Space. At first I just wanted to throw the ball using a hand, which is the Glove in PolyVR. But therefore an extension to PolyVR's physics model would have had to be implemented, but that was too difficult. So instead, it was decided to add a sliding board to the ucecase.

The third challenge is adjusting the hight of the sliding board, because at first the ball did not get enough speed to hit the cans. So different weights were added to the ball and cans, making the objects correctly physicalized without friction. Then the ball can hit the cans. At the end, a reset option was implemented and the Cyberglove has been integrated in the use case.

The difficulty of the usecase is to determine how high the ball should be dropped. With different heights the ball will get different speeds, although it can hit the cans with different position and force.

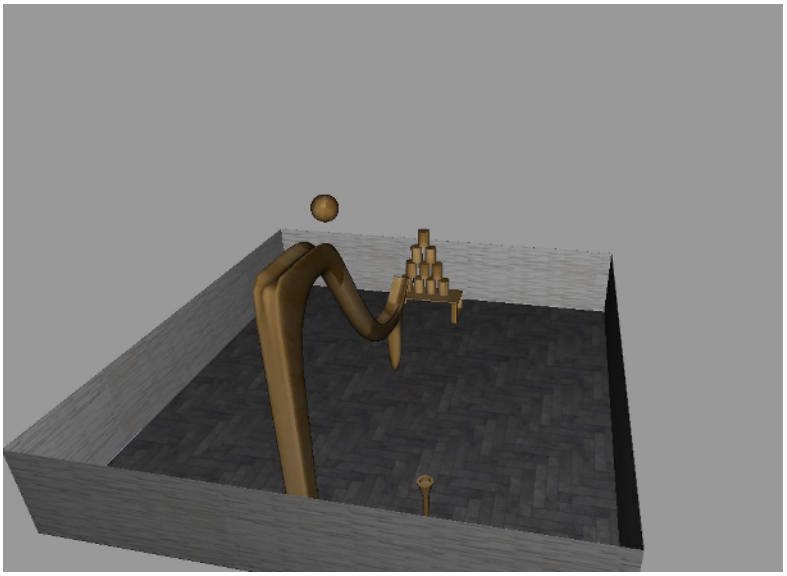


Figure 17: Ball play use case

5.9 Glove dragging code

When creating a new use case, it is very easy to incorperate the glove in order to use it for dragging objects. The following code has to be added as a script into PolyVR which runs on a loop. Objects in PolyVR which are marked as `Pickable` can then be dragged using the glove.

```
def controlHandleDrag():
    import VR

    # must run on loop trigger

    if not hasattr(VR, 'isDragging'):
        VR.isDragging = False

    glove = VR.find('vrpn_device')
    if not glove: return # in testing enviroments, dont run the script

    b = VR.find('ART_tracker.2')
    b.findAll('av_ray')[0].show()

    def startDrag():
        if glove.intersect(VR.getRoot()):
            VR.isDragging = True
            print "Attempting to start dragging, found intersecting object(s)."
            obj = glove.getIntersected()
            print "Intersecting Object:" + str(obj)
            glove.drag(obj)
        else:
            print "Attempting to start dragging, but no intersecting object."

    def stopDrag():
        VR.isDragging = False
        glove.drop()

    if hasattr(glove, 'getSlider') and hasattr(glove, 'setBeacon'):
        num = glove.getSlider(100) # bend sensor data
        glove.setBeacon(VR.find('ART_tracker.2'))
    else:
        num = 0


    # If gripping while not dragging the handle, start dragging it.
    if num >= 10 and not VR.isDragging:
```

```
38          VR.fGlove = 0
39          VR.touchedHandle = True
40          startDrag()
41
42      # If loosening the grip while dragging the handle, stop dragging it.
43      # Also stio dragging if a challenge is running
44      if (num < 10 and VR.isDragging) or VR.isChallenge:
45          VR.fGlove = 1
46          stopDrag()
```

assets/glovedragging.py

6 Project coordination and Management

The following chapter deals with the tasks of project coordination and management and explains the difficulties, problems as well as their solution, which influenced the work flow through the Cyber Glove project.

6.1 Tasks, problems, and difficulties

First of all it was necessary to find a way to combine the knowledge and skills of students from different disciplines, to create openness to learn together and from each other, and to transform the shared resources into an interesting and goal fulfilling project outcome.

Forming a group out of students from different disciplines but also from different countries with different culture and language backgrounds made communication and management within this interdisciplinary group framework sometimes even more difficult. Learning from each other and approaching each other when problems arise is not a prerequisite in every culture. After realizing those difficulties the focus was set on improving communication and understanding within the group.

Individual motivation, increased communication effort as well as control was necessary and helpful. Furthermore, the approach to the given tasks was different in many aspects due to the different disciplines, characters and cultures. Again it was important to first analyze the circumstances and then make decisions to merge.

In addition it was necessary to frequently communicate the current project state, problems as well as uncertainties with the supervisor. The fact that our supervisor was often not available in person and communication by mail took longer than calculated made fulfilling the tasks long-winded and difficult. After realizing, communication and meetings were scheduled with lead time and we tried to fulfill tasks with the resources we could organize ourselves.

Apart of facing those tasks and difficulties, weekly meetings were organized and held, fulfillment of tasks was discussed and new tasks as well as goals were defined. While keeping an eye on the project schedule it was necessary to focus on an equal and fair division of work within the group.

In addition to the tasks named above 3D printing as well as designing the basic parts of the new Cyber Glove among other things by sewing, soldering and cabling were parts of the Project Management.

7 Summary and outlook

In the previous semester a Cyber Glove was developed to enable the integration of intuitive finger and hand gestures into a virtual reality scene. The bending and the contact of the fingers as well as the hand position in the scene were tracked.

In the Virtual Reality Practical Course in the winter semester 2017/2018 our team achieved designing and developing a new Cyber Glove to improve the functionality and usage of it and to make it more comfortable to wear. Hereby we designed an advanced version of the board, housing, wiring and tracking system as well as implemented three additional bending sensors and finger sensors. Furthermore we created an one plug solution and adapted the Aduino code to the new hardware.

The new connection system of the housing with the glove, based on magnets, enables disconnecting one from the other. This simplifies working on the housing, board and wiring or exchanging components during future development of the Cyber Glove.

Another innovation were the new Use Cases our group created and programmed with PolyVR. Now users can chose in-between the cases 'Line drag', 'Tower of Hanoi', 'Ball playing', 'Menu' and others.

Looking at the future development opportunities of the Cyber Glove one recommendation is adding an attachment for the joystick which would improve the handling. Another one is shortening the cables. In addition we suggest shortening the time the Glove needs to connect with the server application to exchange data between the Glove and PolyVR. Moreover charging the battery via the microcontroller is recommended to be fixed.

Further development in the area of use cases could be realized by creating new use cases which are more complex. For example referring to the use case 'line drag': more levels of increasing difficulty could be created and a database to save the user scores to compare them with each other could be connected.