Design and implementation of a computer aided ergotherapy framework

Georg Grab

Advisor: Dirk Reichardt, Prof. Dr.

STUDENT RESEARCH PROJECT

created as part of the
Bachelor study program

Applied Computer Science

in Stuttgart

June 2018

Declaration

I hereby declare and confirm that this thesis is entirely the result of my own original work. Where other sources of information have been used, they have been indicated as such and properly acknowledged. I further declare that this or similar work has not been submitted for credit elsewhere.

Stuttgart, June 4, 2018

Georg Grab

Contents

Abstract

After carrying out hand surgeries, the patient often has to undergo a lengthy recovery period in order to get the hand mobility back to the original, healthy state. The recovery phase is usually accompanied by a dedicated ergo therapist in various therapy sessions, requiring the physical presence of both the patient and an ergo therapist.

In a joint venture of the DHBW Stuttgart and the Katharinenhospital Stuttgart, the possibility of computer aided recovery is explored. The long term goal is for the patient to be able to complete some of the recovery exercises at home, saving time and resources for both the patient and the clinic.

This student research project is exploring one particular possibility of achieving this: combining low cost hand tracking devices with the modern web. Hand tracking devices are small hardware devices containing various sensors, capable of producing a virtual representation of the hand. Hand tracking devices are normally used in the field of VR and Augmented Reality, but can arguably also be used to track the post surgery recovery progress. A well known and relatively inexpensive Hand tracking device is the Leap Motion Device Platform. This is the Tracking device primarily used for this project, although the project architecture allows for the possibility to implement support for other tracking devices.

The essence of the project is to gamify the recovery exercises: the patient should be able to play games through a web interface, controlled by Hand Gestures (for example, spreading the thumb to make a spaceship shoot). The Hand Gestures correspond roughly to recovery exercises that would normally have been done together with a therapist. The therapist should be able to configure gestures for a patient that he or she has to get better at in order to aid in recovery. These gestures must then be used by the patient in order to correctly navigate the game. The gameplay should finally be producing monitoring information for the therapist to review, and thus provide evidence for the recovery progress of the patient.

This work presents a possible Architecure and Minimal Viable Product (MVP) implementation for such a system. The core system components are identified and implemented. Furthermore, advise on extending the system and the recommended next steps are given. Finally, concrete Usage Manuals are provided for both the potential end users and future developers.

Chapter 1

Introduction

1.1 Problem description

hand therapy from home, refer to previous works

Chapter 2

Solution Design

2.1 Available Alternatives

"dumb" web platform for visualizing, main work happening in server processes running locally

fully featured web platform, doing everything

GUI Application

2.2 Elected Alternative

fully featured web platform, because modern web technologies allow it, GUI Application does not fit the requirements, and separating main work into server thread is, while potentially more performant, both potentially insecure and unnecessarily complex.

Chapter 3

Project Scope

framework for others to build upon. minimum viable product covering as much of the overall required architecture as possible.

3.1 System Architecture

Chapter 4

Development Pipeline

Webpack 4
- VueJS 2, Vue Router, VueX, inversify Dependency Injection
- Karma Unit Tests
- THREE.js

Chapter 5

Implemented Subsystems

5.1 Device Driver Interface

Describe Generalized Device Driver Interface. Point out that adding different devices is possible with this architecture.

5.2 Device Facade

Justify for a need of a Facade in Front of the raw Device Driver: Seperation of Concerns and Recording (Mock data) Functionality. Ease of Testing.

5.3 Device Debug Interface

Describe Component: Raw Device Logger, Device Status Log, Device Graphical Log

5.4 Graphical Hand Logger

THREE.JS, OrbitControls, Smoothing, Prop Configuration

5.5 Device Recorder

Record Segments of hand movements in order to aid in development, make classification errors reproducible

5.6 Persistence Provider

Describe Abstract Persistence Provider Interface

Describe Concrete Persistence Provider Interface Implementation: IndexedDB

5.7 Preprocessing Framework

Justify need for Preprocessing: Lots of useless data coming from the device. Preliminary clean up of data may be relevant for all classifiers

5.8 Classification Framework

Describe how Classifiers receive the preprocessed data frame stream, and transform the stream in order to emit another stream of classifications, along with relevant metadata

5.9 Game Execution Engine

Describe how Games are receiving the Classification Stream and using that in order to drive the gameplay.

Chapter 6

Recommended Future Works

6.1 Proposed Subsystems

6.1.1 Data Postprocessing Framework

Generic interface that does something with classification data / game data. For example logging it to a remote location, like a backend

auth -> classification data -> backend

6.1.2 Progress Analysis Dashboard

Component that gets patients postprocessed classification / game results, and visualizes progress

auth -> backend -> select patient -> patient view

6.1.3 Messaging Platform

Therapist Requirement. Ability to send messages to / from patients

6.2 Proposed Enhancements

6.2.1 Classification Metadata

Anti Cheat (Therapist requirement)

Log Classification Specific relevant Metadata for Therapist Analysis

Chapter 7

Conclusion

Possibility of Project Confirmed, Modern Web is evolved enough to tackle this task. But lots of extensions should be made in order to make the project actually useful

Appendix A

User Manual

A.1 Installing the Hardware Device Driver

A.2 Verifying the Installation Success

A.3 Configuration

A.3.1 Preprocessors

A.3.2 Classificators

A.4 Gameplay

Appendix B

Developer Manual

B.1 Building and Running the Project in Development Mode

B.2 Executing Unit Tests

B.3 Extending the Framework

B.3.1 Adding a Preprocessor

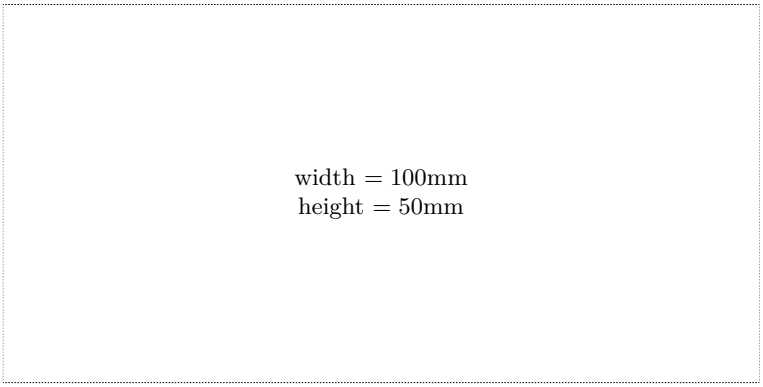
B.3.2 Adding a Classifier

B.3.3 Adding a Game

References

Check Final Print Size

— Check final print size! —



— Remove this page after printing! —