Hochschule Esslingen
University of Applied Sciences

Department of Computer Engineering

# Classification of Large Biological Image Sets within the Browser

Bachelor Thesis

Christoph Friedrich

Summer Term 2018

First Advisor: Prof. Dr. Jürgen Koch
Second Advisor: David Dao ETH Zürich

# Declaration

Hereby I certify that I did this work by my own. Literally or analogously used intellectual property was declared as such.

| | |
|---|---|
| Esslingen, May 17, 2018 | |
| Place, Date | Christoph Friedrich |

# Contents

# 1. Motivation

These days a lot of data is generated to improve the way cars are used or phones are unlocked. Autonomous driving and face recognition are only two examples where a lot of data is needed to improve algorithms and with that the devices these algorithms are used for.

The most interesting area large collections of data are used is health care. Modern technologies give possibilities that have not been there before. Artificial intelligence makes it possible to identify a malignant melanoma by discriminating it from a harmless mole with a normal cellphone camera.

However, there is still a lot of high potential data unused. One reason for that is computer laymen don't know how to use the data they have. It seems to be a privilege to Computer Scientists and Computer Enthusiasts to take advantage of that data and the potential it holds.

For example biologists and doctors going through Hundreds and Thousands of samples, giving them a tool to automatize there daily work flow would be a huge ease.

Goal of this bachelor thesis is to evaluate on how to give normal users, users without knowledge of how to install a Python environment or dealing with databases, the possibility to use all advantages of machine learning. By that this projects follows the modest aim to democratize artificial intelligence.

# 2. Basics

## 2.1. JavaScript

There are a few big advantages offered by **JavaScript (JS)**.
Maybe the biggest advantage is that it comes with the web browser, so everybody can run it by simply typing in an Uniform Resource Locator (URL). This offers great possibilities, since the users don't have to install any software package. The provided application is immediately ready to use once it was loaded.

Today JS is the most used programming language world wide (Picture: 2.6). JS first appeared in 1995, originally developed at Netscape by Brendan Eich it had a simple purpose, dynamically manipulating the HTML DOM in the browser.
At the same time a company called Sun worked on a programming language called Java, Netscape and Sun decided to work together and followed the idea of making LiveScript using Java Applets. Netscape noticed that it would be good marketing to rename LiveScript to JS. Since Java already had a growing community and was liked by many people, so Netscape took the chance and JS was born.

However both language don't share a lot of things together, Java is a normal, static, and highly typed programming language, it runs on a virtual machine and needs to be compiled, whereas the single threaded JS only runs in the browser and is script. Nevertheless they share the C related syntax and some similarities regarding naming conventions, both support object oriented programming.

The following analysis shall show why Java Script is in many ways superior for web development. Jeff Attwood the co-founder of the computer programming question-and-answer website Stack Overflow and Stack Exchange once said "Any application that can be written in JS, will eventually be written in JavaScript". [1] [3] [4]

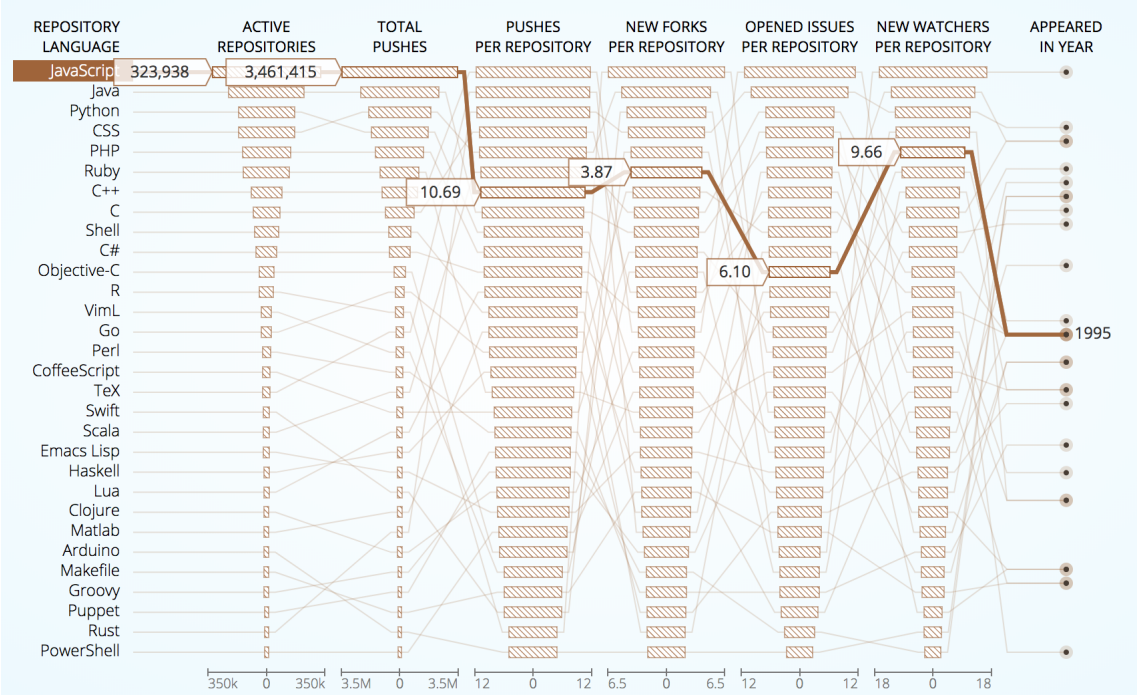


Figure 2.1.: Github Repositories sorted after used language
source:[2]

### 2.1.1. JavaScript Distributed Architectures

Most modern web designs rely on they three-tier architecture. On the server side computing is done and data is stored in databases. The client fetches data from server and makes it accessible. Often the user is able to change the data on client side.

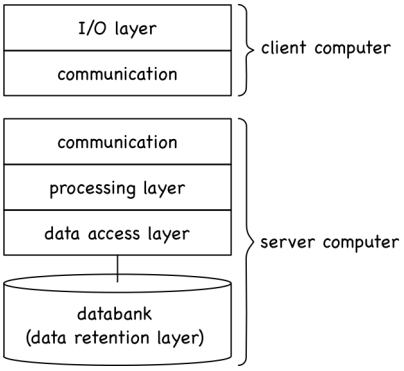


Figure 2.2.: Github Repositories sorted after used language
source:[2]

For example deleting or adding a user, a request is sent to the server and a database change is done. For client side it facto mandatory to use JS, but on server side a large spectrum of languages is available, Java, CSharp, PHP, Ruby these are just the most famous one.

Once a user is added on client side, code is needed there in JavaScript. Same action needs to be implemented in a different language on server, therefore a second implementation is needed for the same action. But the JavaScript ecosystem is undergoing rapid growth, with Node.js JS is coming to the server and developers are now able to do whole applications in JS.

That's big step, distributed systems that can use shared modules. It even affects the job market, front-end developers and back-end developers have been two separated jobs for over decades but now are one. People do not need to learn two languages anymore if they want to do full-stack web development, everything is covered with JavaScript.

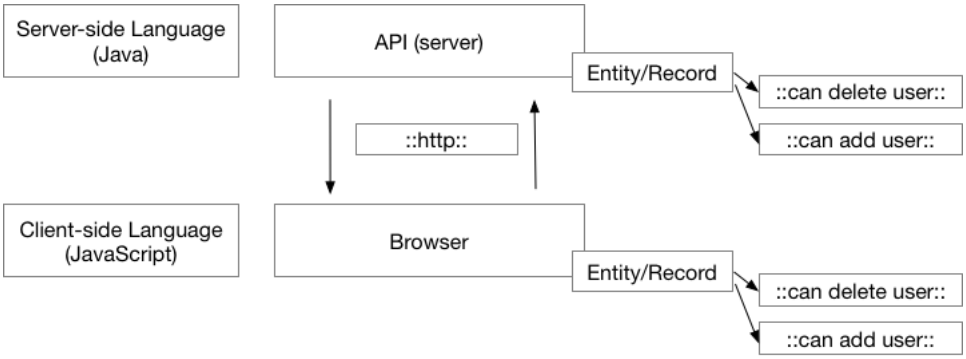


Figure 2.3.: Github Repositories sorted after used language
source:[2]

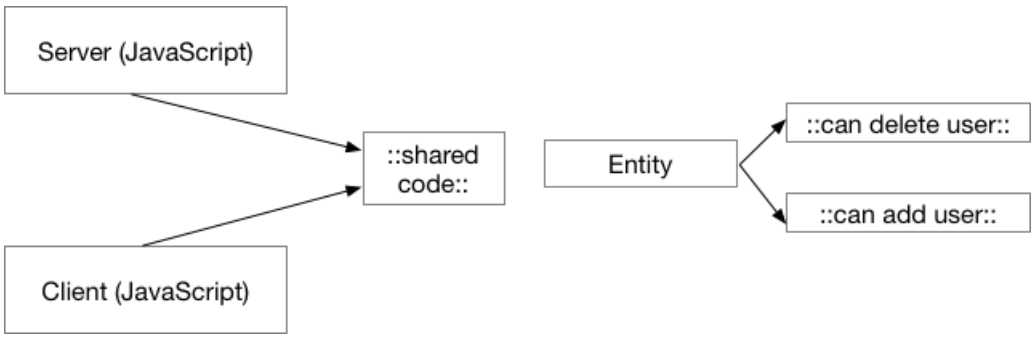


Figure 2.4.: Github Repositories sorted after used language
source:[2]

### 2.1.2. Multi-Platform Support

Deploying an application for the web browser is only one aspect of modern web development, often a separate implementation is needed for mobile devices or even desktop implementations. Two major operating systems are currently dominating the smart-phone market Android and IOS.

Both support native App developing but that brings back the problem of having to implement the same functionality twice. Sure smart-phone Apps are different from browser based applications, starting with the touch event and ending up with totally different layouts.

With JavaScript developers can take advantage of modern frameworks like React Native or Electron, which give the possibility to code in JS, the JavaScript is communicating natively with implemented components written in Java on Android, Objective C on iOS, CSharp on Windows via an abstraction called bridge. Everytime JS is called it forwards the call to the native code implementation, the response is passed back to the implemented abstraction JS.

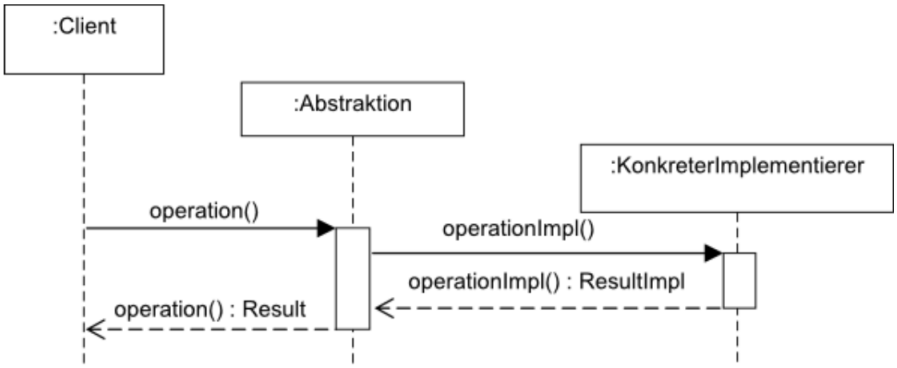


Figure 2.5.: Market share operating systems based on sold phones
source:[2]

### 2.1.3. Single Threading and Parallelism

The single treaded paradigm and non blocking architecture JS is bond to, can be considered a weakness. Developing with JavaScript will generate errors and behavior that is perplexing . This section is about analyzing why JS is relying on single threading and how it is still possible to add concurrency.

JS was developed to manipulate the DOM, the DOM is a limited resource. Executing two pieces of code, asynchronously changing the same part of DOM would not be good. Given this single purpose it was decided to make JavaScript single threaded.

Actions that do not change the DOM should be able to run asynchronously. That makes sense, since fetching data from the server or reading a large list of files can take a lot of time and would block the browser, buttons would not be clickable anymore and any rendering would stop.

How is it possible that JavaScript offers the opportunity to run code concurrently, even though it is single threaded? The following example how multi-threading is implemented in JavaScript.

Every time a function in JS is called, it is put on the call stack. The result of that function is popped out from stack.

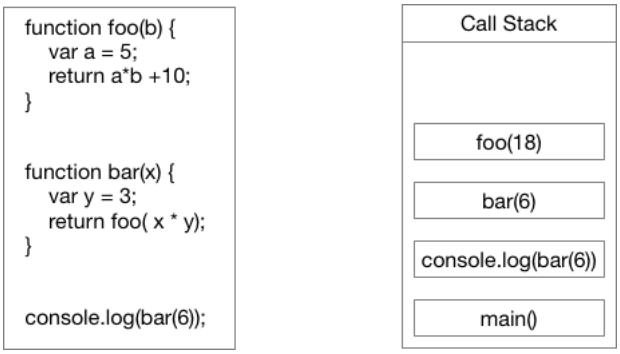


Figure 2.6.: Github Repositories sorted after used language
source:[2]

The browser offer APIS to execute code concurrently, it can do that because web browsers are written in C++, which offers full multi-threading support. The concrete implementation of concurrency in JavaScript shall be explained in the following.

For executing concurrently JavaScript takes adavantage of the Web API, it calles the API with the function to be executed

For executing code asynchronously JavaScript provides so called callbacks. A callback is executed once the function, the callback is attached to, finished executing. The following example shall further demonstrate how concurrency in JavaScript works.

As the name is already revealing, a callback is called once an asynchronous function finished executing. This passed back callback will be put on the stack and normally be called in a synchronous way. What happens exactly is that JavaScript passes the function and the regarding callback to Web APi after running it the callback with the result is passed to the task que, the event loop then checks if the stack is free and puts the callback on top of stack.The decoupling of the caller from the response allows for the JavaScript runtime to do other things while waiting for your asynchronous

operation to complete and their callbacks to fire.

# 3. Systementwurf

## 3.1. Abtastung

Idee ist es, das aufgenommene Bild in vier verschiedene Regionen einzuteilen. (Siehe Bild 3.1)

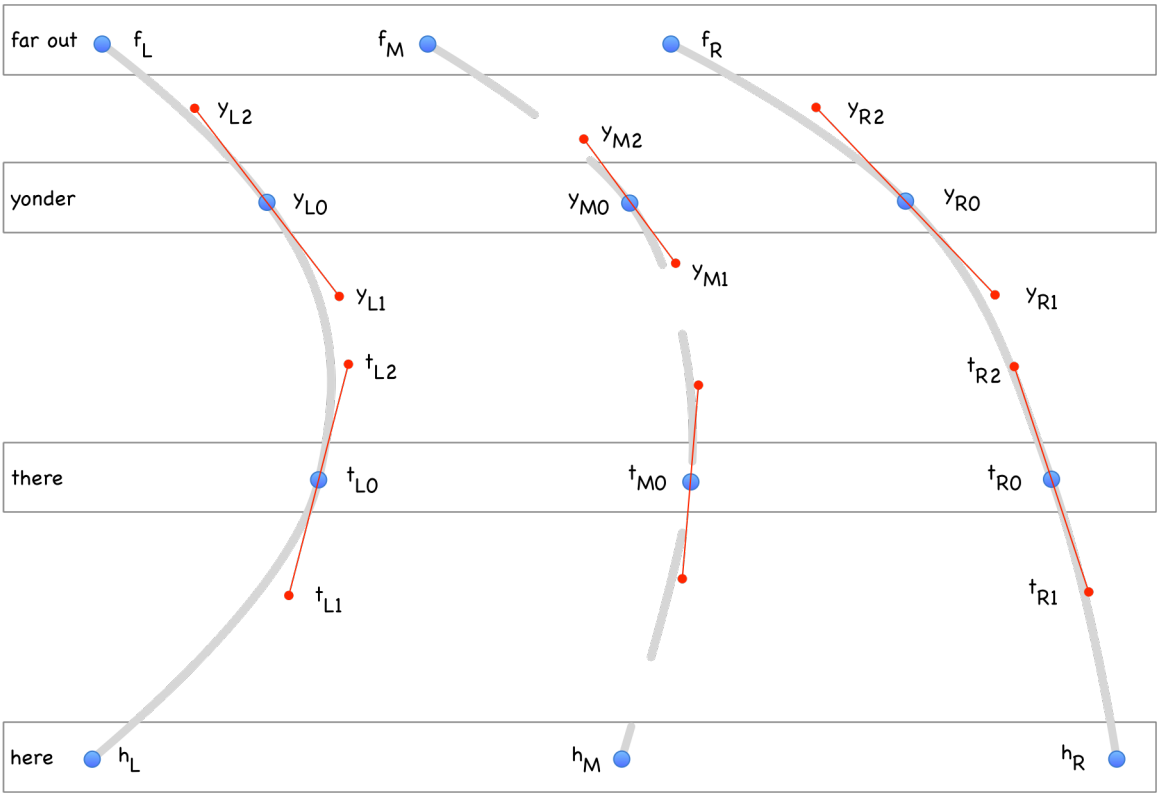


Figure 3.1.: Verschiedene Regionen

Für jede dieser Regionen wird jeweils eine Zeile abgetastet. Als Ergebnis erhält man ein Histogram (Siehe Bild 3.1). Man kann erkennen, dass die Spurmarkierungen durch Peaks mit einer bestimmten Breite und einem bestimmten Abstand (in Grün) zueinander erkennbar sind. In Rot zu erkennen sind Artefakte, welche sich durch eine kleine Breite(1-2px) auszeichnen. Sie entstehen zum Beispiel durch Reflexionen auf der Fahrbahn.

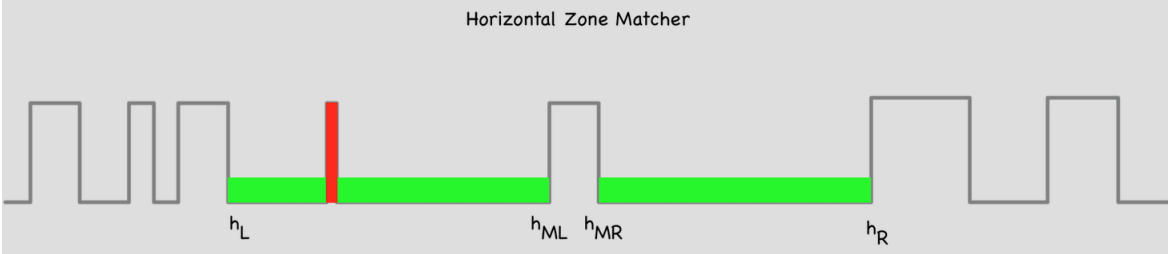


Figure 3.2.: Histogramm

### 3.1.1. Eleminiere Artefakte

Um Artefakte und Objekte ohne Kante zu eliminieren werden die abgetasteten Regionen mit der Canny Kantenerkennung gefiltert. Übrig bleiben Punkte, die zu einer Kante gehören und deutlich weniger Artefakte.

## 3.2. Wahrscheinlichkeitsverteilung

Für jeden gefunden Punkt in der Region soll bestimmt werden mit welcher Wahrscheinlichkeit dieser zu einer Spurmarkierung der rechten Fahrspur gehört. Verschiedene Filter werden dafür eingesetzt, jeder dieser Filter addiert eine Wahrscheinlichkeit zu jedem der gefundenen Punkte. Initial erhalten alle Punkte die Wahrscheinlichkeit null.

#### 3.2.1. Filter ROI

#### 3.2.2. Filter Spurbreite

#### 3.2.3. Filter Streifenbreite

#### 3.2.4. Filter Tiefpass

#### 3.2.5. Trafo

### 3.3. Szenarien

#### 3.3.1. Ausbleibende Spur

##### Verhalten

#### 3.3.2. Zebrastreifen

##### Zebrastreifen Erkennung

##### Verhalten

#### 3.3.3. Kreuzung

##### Stopplinien Erkennung

##### Verhalten

#### 3.3.4. Weitere denkbare Szenarien

### 3.4. Ausblick

#### 3.4.1. Vortasten

#### 3.4.2. Kalman Filter

# 4. Systemanalyse

Abbildung 4.1 zeigt den prinzipiell Ablauf der Systemanalyse.

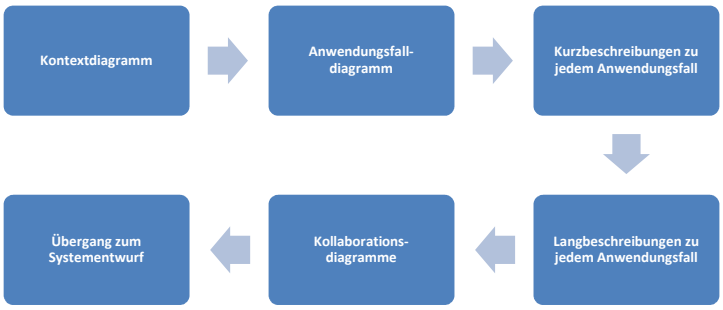


Figure 4.1.: Ablauf der Systemanalyse

## 4.1. Kontextdiagramm

...

## 4.2. Anwendungsfalldiagramm

...

## 4.3. Kurzbeschreibungen zu jeden Anwendungsfall

...

# 5. Systementwurf

## 5.1. Klassendiagramm

## 5.2. Logisches Datenmodell

## 5.3. Physikalisches Datenmodell

## 5.4. Entwurf der grafischen Oberfläche

# 6. Zusammenfassung

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

# Bibliography

[1] A brief history of JavaScript ? Ben Aston ? Medium.

[2] GitHut - Programming Languages and GitHub. Last accesed on 2018-05-15.

[3] JavaScript is the Future of Enterprise Application Development.

[4] Wie unterscheidet sich JavaScript von Java?

[5] Ekbert Hering, Rolf Martin und Martin Stohrer. *Physik für Ingenieure*. Springer Verlag, 2004.

[6] RASPBERRY FOUNDATION. Cam module v2, 2016. Quelle: https://www.raspberrypi.org/products/camera-module-v2/, zugegriffen am 15.12.2017.

[7] RASPBERRY FOUNDATION. Raspberry 3 modell b, 2016. Quelle: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/, zugegriffen am 15.12.2017.

[8] RASPBERRY FOUNDATION. Raspberry 3 modell b, 2016. Quelle: https://www.raspberrypi.org/app/uploads/2017/05/Raspberry-Pi-3-462x322.jpg, zugegriffen am 15.12.2017.

[9] Joachim Goll, Cornelia Heinisch und Frank Müller. *Java als erste Programmiersprache*. Teubner Verlag, 2005.

[10] Lothar Papula. *Mathematische Formelsammlung für Wissenschaftler und Ingenieure*. Vieweg Verlag, 2003.

[11] Manfred Dausmann, Ulrich Bröckl und Joachim Goll. *C als erste Programmiersprache*. Teubner Verlag, 2005.

# List of Figures

# List of Tables

# Listings

# A. Anhang zum Systementwurf

Allgemeine Beschreibung des Anhangs

## A.1. Diagramme

Hier werden Diagramme platziert, die in den Textkapitel zuviel Platz beanspruchen.

## A.2. Tabellen

Hier werden Tabellen platziert, die in den Textkapitel zuviel Platz beanspruchen.

## A.3. Quellcodelistings

Hier werden Tabllen platziert, die in den Textkapitel zuviel Platz beanspruchen.