Projektbeleg VL Virtuelle Realität

Hannes Bischoff

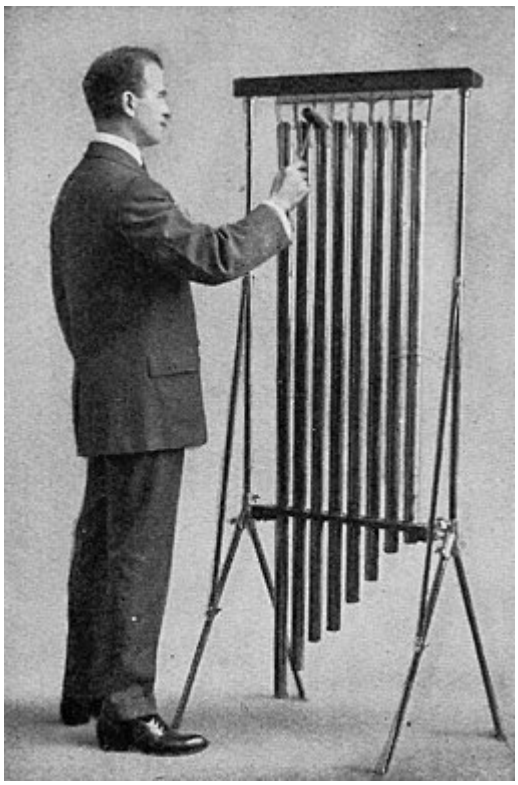
Inhaltsverzeichnis

Beleg

Motivation

Projektidee war Eine App für ein Android Smartphone zu entwickeln die Röhrenglocken abbildet. Das Instrument der Röhrenglocken gehört zu den Idiophonen wird seit Ende des 19. Jahrhunderts auch in Orchestern verwendet. Um es zu spielen schlägt der Schlagwerker mit einem kleinen hämmerchen gegen eines der Röhren. Diese sind so aufgehangen das diese frei schwingen können und dabei klingen. Die meisten Röhrenglocken bestehen a 8, 12 oder 20 Röhren. Ziel diese Projektes ist es eine einfache, günstige Simulation dieses Spielverhaltens zu erzeugen. Die Verschiedenen Sensoren sollen dabei als Trigger dienen. Imitiert man mit dem Smartphone einen schlag, so soll analog zum anschlag mit dem hammer ein Ton erklingen. Wird eine andere röhre angeschlagen ändert sich die Tonhöhe, auch das soll abgebeildet werden die wird so gelößt das abhängig von der richtung in der das smartphone bim schlag gehalten wird sich die Tonhöhe ändert.

Ein Schlagwerker an den Röhrenglocken 1917
https://de.wikipedia.org/wiki/R%C3%B6hrenglocken



Technologie

Grundlagen

Zentrales element der Simulation ist das Smartphone. Dank seiner kompakten größe, handlichkeit und der eingebauten sensorik lässt es sich vielseitig verwenden und ist hierfür ideal geeignet. Außerdem ist keine Anschaffung spezieller hardware nötig wenn man handelsübliches Android smartphone besitzt.

Die Wahlfiel auf die Android platform, da diese im kern ein open source source betriebssystem ist und ess unzählig freie software und tools für die entwicklung von apps gibt.

Der Programmcode dieses Projektes ist in Java geschrieben nutzt die als Entwicklungsumgebeung wurde das freie Android studio mit den dazugehörigen Android developer kits genutzt welchee federführend google entwickelt werden. Layoutdatein und design des projektes sind android typisch in xml.

Um die Notwendigen und recht umfngreichen Vorarbeiten einer Android app zu verkürzen habe ich auf ein bereits exisiterendes projekt mit offenem Quellcode zur auswertung der verschiedenene sensoren zurück gegriffen. Das Projekt Lemur auf github, shien dafür ideal geeignet zu sein. Es bedarf natürlich einiger anpassungen aber verkürzte die Arbeitsaufwand immens. Layout, grundeinstellungen, App interne navigation und berechtigungen systemseitig anzufragen können mitunter bei einer app die von grund auf neu entwickelt wird mehrere Tage arbeitsaufwand bedeuten befor dass auch nur eine Zeile des gewünschten Algorithmuses geschrieben werden kann.

Kernaufgaben

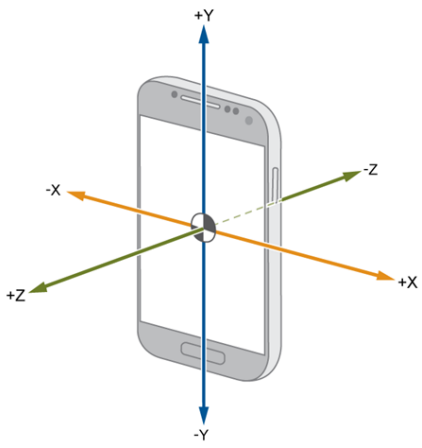
Von allem überbau jeglicher Android projekte abgesehen lassen sich zwei Schwerpunkte heruskristalisieren.

Das verwendete basis projekt ließt dieverse sensorwerte über die systemschnittstellen aus und kann diese grafisch anschaulich plotten.

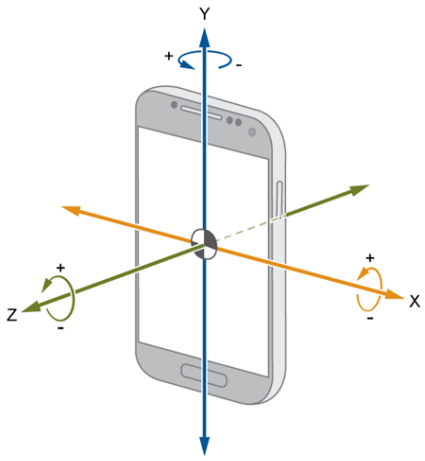
Die grafische darstellung der werte ist besonders nützlich um Charakteristik und Wertebereich eines ereignisses zu erkennen.

Die meisten heuszutage verkauften Smartphones ferfügen über mehrere eingebaute Bewegungs sensoren. In der Android Entwicklerdokumentation https://developer.android.com/guide/topics/sensors/sensors_motion sind diese beschrieben. Einige sensoren liefern physikalische messwerte wie z.B. Beschleunigungssensor und gyroskop. Andere sensoren existiere nur virtuell. der Roatationsvektor oder schritzäher sind software implementierungen und nutzen die daten der anderensensoren und kobinieren diese und vorverarbeiten die daten.

Beschleunigungssensor



Rotationsvektor-Sensor

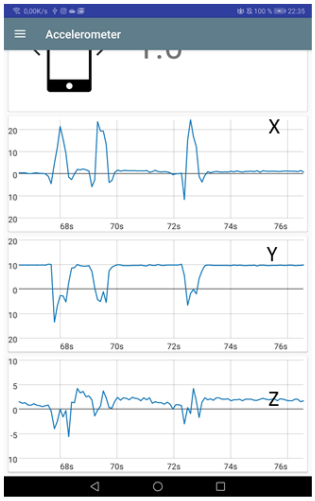


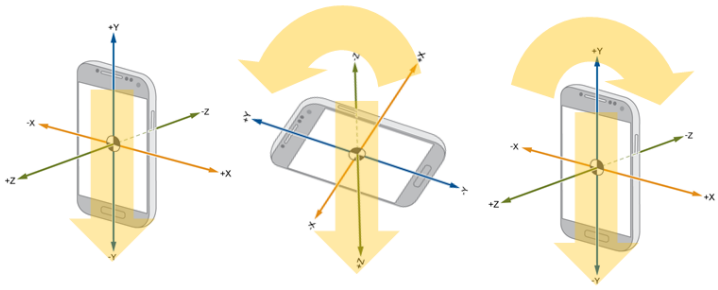
1. Erkennen des Anschlags

Dazu wird das telefon in einer kippbewegung um die Z-Achse bzw in der von der Y und X Achsen aufgespannten display ebene nach links bewegt. Dieses ereigniss kann mit den im Smartphone eingebauten beschleunigungs sensoren erfasst werden. Der sensor liefert werte für Beschleunigung in der X,Y, und Z achse. Um die Ausgelesenen werte zu validieren genügt ein simpler test. steht das Gerät senkrecht auf mit der unterkante auf einem tisch und wird nich bewegt liefert der sensor im idealfall für die Y achse 9,8m/s^2 entsprechend der lokalen erdbeschleunigung und die sensoren für die X und Z achse je 0 m/s^2.

Dokumentation Beschleunigungssensor:

Beschleunigungssensor



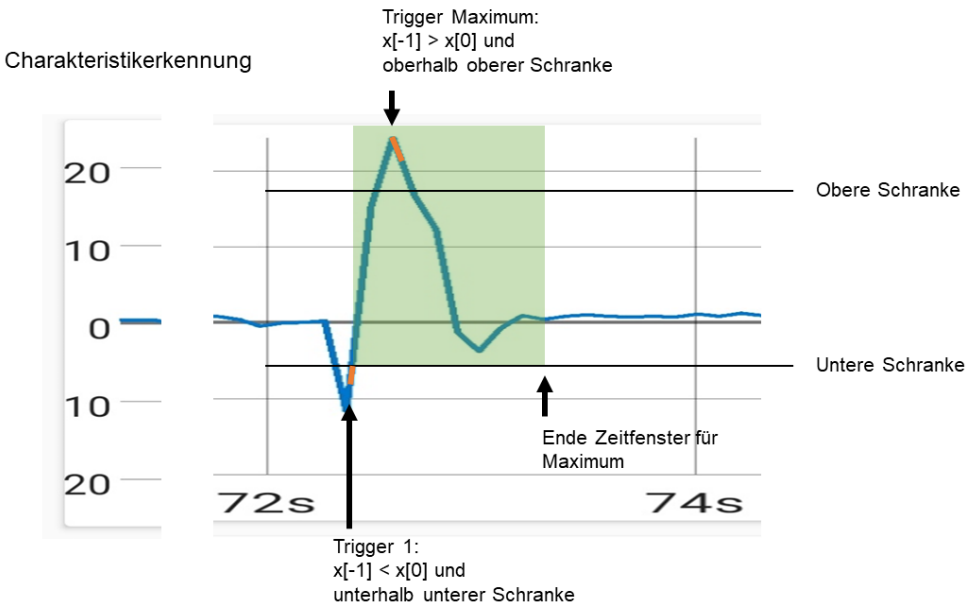


Gut zu sehen: die Erdbeschleunigung auf der Y-Achse in der Ruhelage und deren Einbruch im Moment der Rotation sowie der Charakteristische Ausschlag auf der X-Achse.

Graphish ist der Ausschlag sehr gut zu sehen. Im nächsten schritt muss eine geeignete methode entwickelt und im Algorithmus umgesetzt werden um aus der Zeitreihe das Muster zuverlässig zu erkennen. Das vom sensor gelieferte Signal ist Zeitdiskret mit konstanter abtastzeit und wertediskret. Die Quantisierungsschritte aber sehr klein als fließkommazahl im Java datentyp double.

Als praktikabel und zuverlässig hat sich folgende Methode herausgestellt.

Der Signalwert der X-Achse wird bei jeder iteration mit dem wert der vorherigen iteration verglichen. Wenn der aktuelle größer ist, also die werte ansteigen, und dennoch unterhalb der unteren schranke liegen. wird trigger 1 ausgelöst. Dieser trigger öffnet in 300ms langes zeitfenster (grün). Wird trigger 1 mehrfach ausgelößt spielt das keine rolle, dieser überschreibt den vorherigen. Der Trigger 1 ist notwendig aber noch nicht hinreichend um den Anschlag zu erkennen. Im geöffneten zeitfenster, wird das signal nun auf eine fallende Flanke untersucht. sobald das signal fällt also der vorherige signalwert größer als der aktuelle ist und sich die werte oberhalb der oberen schranke befinden wird dies als Maximum detektiert. Die festlegung der Schranken dienen dazu sehr kleine ausschläge zu ignorieren.

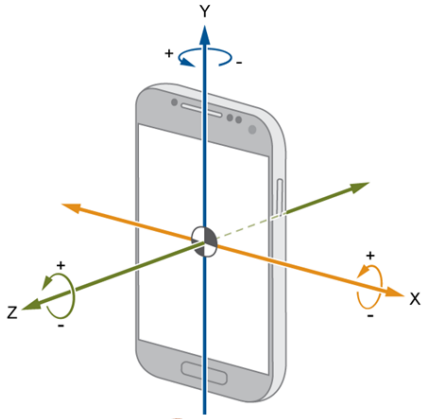


2. Richtungserkennung

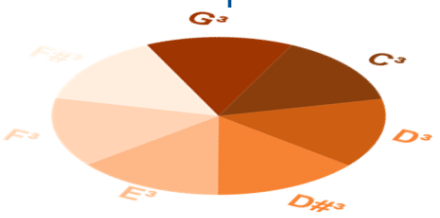
Mithilfe des virtuellen Richtungsvektor sensors ist eine Bestimmung der orientierung des Gerätes im raum möglich. Wird das smartphone gedreht und verharrt dann in dieser position wird dies trozdem erkannt und es wird eine angegeben in welchem winkel sich das smarthpone befindet. Relativ zur angenommenen erdoberfläche und der magnetischen orientierung. Der softwaresensor bezieht daten aus verschiedenen quellen wie: Beschleunigungsmesser, Magnetfeltdetektor und Gyroskop. Fehlt einer dieser sensoren am Gerät, fällt die genauigkeit der werte dementsprehend deutlich schlechter aus.

Für sie simulation wird die X-Z-Ebene in 7 gleichgroße Felder zu je 51,4° eingeteilt die jeweils einen ton repräsentieren. Je nachdem welchen winkel der Sensor für die dreheung um die Y-Achse liefert wird der ton gespielt.

Rotationsvektor-Sensor



Virtueller Sensor kombiniert Daten aus Beschleunigungssensor, Magnetfelddetektor und Gyroskop. Gibt Orientierung des Gerätes relativ zur Umgebung an. Je nach Richtung werden verschiedene Töne angeschlagen.



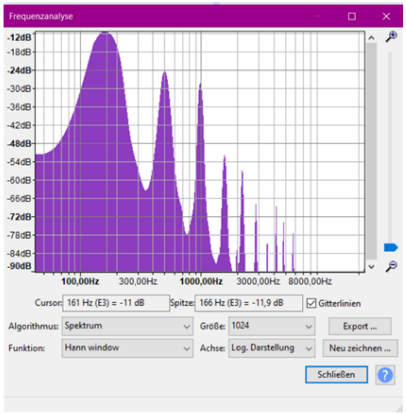
Wird ein maximum detektiert wird Je nachdem welchen winkel der Richtungsvektor Sensor für die dreheung um die Y-Achse zur gleichen zeit liefert wird der dazugehörige ton gespielt.

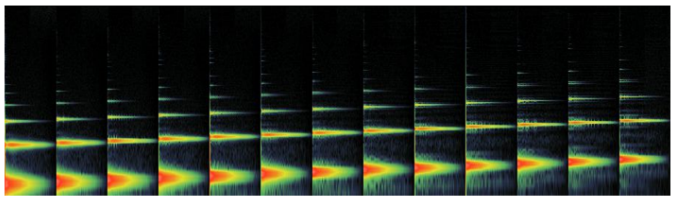
Sound

Für die Klang wird mangels vorhandensein von röhrenglokecn und der schwierigkeit ein frei verwendbares audio file einer tonliter des besagten instuements zu finden auf klangschalen ausgewichen. Das verwendete audio file ist stereo und von der website freesound. https://freesound.org/people/mooncubedesign/sounds/347975/

Die aufnahme der 12-Teiligen Tonleiter geht in halbtonschritten von $C\#^3$ bis $C\#^4$ und weist eine sehr gute qualität auf. Im Frewuenzspektrum sind grund und obertöne gut zu erkennen. Es trit fast kein rauschen auf, wie im Spektrum zu sehen ist.

Frequenzanalyse

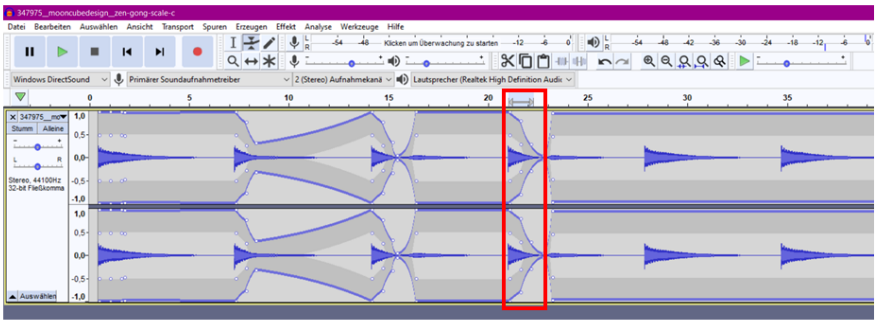




C-Dur Tonleiter der verwendeten Klangschale mit 13 halbtonschritten von C#³ bis C#⁴.
Und das beispielhafte Spektrum für E³.

Die Klangschale klingt für ca 12 sekunden, was hier für die App nicht besonders nicht hilfreich war. in audacity wurde das Audiosignal getrennt und die einzelnene Töne gerausgeshnitten in einzelnen foundfiles gepackt um diese seperat abspielen zu können und die Hüllkurve der amplitude so moduliert, dass der ton nach 2,5 sek auskling ohne aprupt zu enden.

Amplitudenanpassung



Vorverarbeiten des Soundmaterials in Audacity; z.B. Verkürzen der Nachhallzeit von 7sek auf 2,5sek durch Anpassen der Hüllkurven

Da es durch die genauigkeit der Richtungsbestimmung durch den Sensor nicht sinnvoll möglich ist eine umdrehung von 360° in 12 Sektoren zu teilen können in dieser version leider nicht alle Töne der tonleiter untergebracht werden. Die verwendeten töne im aktuellen projekt sind c#³, d³, d#³, e³, f³, f#³, g³. 7 erwies sich als gerade noch praktikable umsetztung um gezielt einen gewünschten sektor zu treffen und zu spielen. Eine weitere unterteilung für mehr töne würde

evntuell mit der unterscheidung ob ein anschlag nach ober oder unten geschiet unterschieden werden können, nicht jedoch durchweitere sektoren.

App layout

Die Activity mit der Röhrenglocken Simulation ist in das Konstrukt der Basis app integriert. Das layout ist schlicht gehalten und zeigt je nach orientierung des Smartphones an welcher ton gespielt würde, wenn das gerät jetzt zum schlag angesetzt wird. Dabei hilft zusätzlich die farbliche unterstützung in verschiedenen orange tönen. Die Gradangabe ist basis für die unterteilung der Sektoren, jedoch nicht mit der ausrichtung nach Norden übereinstimmen. Das diagram im unteren bereich dient nur der information und zu debugging zwecken. es veranschaulicht die Beschleunigungskräfte die Auf das telefon im moment des anschlages reagieren. Mitunter können beschleunigungen von 3 bis 4 g gemessen werden.

App Layout

Je nach Richtung in der das Smartphone gehalten wird werden verschiedene Töne angespielt. Zur besseren Unterscheidung der jeweiligen Bereiche für einen Ton wird ändert sich die Farbe. Das Diagramm im unteren Bereich zeigt zur Kontrolle die Beschleunigungswerte der X-Achse an.

Ausblick

Das sotationssymmetrische erweistsich mitunter als unpraktisch, da zum teil große drehbewegungen nötig sind um von einem zum andern Ton zu wechseln.

in einer weiteren version könnten mehr Töne durch einführung einer virtuellen ebene geschaffen werden. z.B. anschlag nach oben, anschlag nach unten.

Referenz

Projektverzeichnis der Simulation auf Github:
https://github.com/hablix/Lemur

Quellen

Basis projekt welches die grundlage für die simulation ist auf Github:
https://github.com/Roslund/Lemur

verwendeter soundfile: https://freesound.org/people/mooncubedesign/sounds/347975/

Entwicklerdoku Senroren: https://developer.android.com/guide/topics/sensors/sensors_motion

Entwicklerdoku Kordinatensystem Sensoren:
https://developer.android.com/guide/topics/sensors/sensors_overview.html#sensors-coords

Entwicklingsumgebung Android studio: https://developer.android.com/studio

Audio editor Audacity: https://www.audacityteam.org/

Bildnachweis

Achsen smartphone: https://jp.mathworks.com/help/supportpkg/android/ref/gyroscope.html

Röhrenglocken:
https://de.wikipedia.org/wiki/R%C3%B6hrenglocken#/media/Datei:Chimes_(IMSO_pp55).jpg