EveryDrum

Autoren:Tom Ziegler[1], **Lukas Peschel**[1]
Betreuer: Dr. Sebastian Merchel[1]

[1]Technische Universität Dresden

Erstellt am 11. Juli, 2018

Wird eine Oberfläche durch einen Anschlag angeregt, so beginnt die Oberfläche zu schwingen. Diese Schwingung kann mittels Beschleunigungssensoren auf der Oberfläche aufgenommen werden. Diese Schwingungen sollen genutzt werden, um verschiedene Anschlagspositionen voneinander unterscheiden zu können.
Verwendet wurde ein Arduino Nano mit einem Kombisensor MPU-6050. Die Auswertung der Sensordaten fand auf einem Android-Smartphone statt. Zur Unterscheidung der Anschlagpositionen wurden aus den aufgenommenen Schwingungen mittels FFT-Analyse die dominantesten Frequenzen ermittelt und der Frequenzabstand zu den erlernten Positionen bestimmt. Nach erfolgreicher Zuordnung zu einer bekannten Position wurde ein entsprechender Ton abgespielt.
Ziel des Projektes war es, mittels des entstandenen Systems eine beliebige Oberfläche als virtuell erweitertes Schlaginstrument nutzen zu können.

Drums | Android | Arduino | PureData |

1. Einleitung

Diese Projektarbeit entstand im Rahmen des Moduls "Virtuelle Realitätïm Sommersemester 2018. Die Zielstellung war die Entwicklung eines Systems, bestehend aus Hard- und Software, mithilfe dessen eine beliebige Oberfläche als Schlagzeug verwendet werden kann. Eine angeschlagene Position soll dafür in Echtzeit klassifiziert und einem der vorher erlernten Schlaginstrumente zugeordnet werden.

Die Vergabe der Projektthemen und erste Besprechung dieser erfolgte am 16.04.2018. Am 28.05. erfolgte eine Zwischenpräsentation, im Rahmen dieser die bisherigen Konzepte der verschiedenen Teams gegenseitig vorgestellt und diskutiert wurden. Die Abgabe und abschließende Präsentation der Ergebnisse erfolgte am 09.07.2018. Der Zeitraum der Bearbeitung erstreckte sich damit über 12 Wochen.

Im Rahmen einer ersten Konzeption wurden folgende Maßgaben für die Umsetzung aufgestellt:

- Das entstandene System soll portabel, einfach anzuwenden und wenig spezielle Hardware erfordern
- Um ein realistisches Spielgefühl zu erzeugen, soll die maximale Latenz zwischen Anschlag und Abspielen des Sounds 100ms betragen
- Das System soll robust funktionieren und dem Anwender ein Feedback über die Zuverlässigkeit der Erkennung liefern, sodass dieser eventuell Verbesserungsmaßnahmen durchführen kann (z.Bsp. Reduzierung von Hintergrundrauschen)

Um die Realisierung der Projektidee in diesem Zeitrahmen möglich zu machen, wurden zu Beginn einige Vereinbarungen getroffen:

- Als Oberfläche sollen vorerst nur Tische mit Holz- oder Kunststoffplatten verwendet werden
- Maximal sollen drei verschiedene Positionen unterschieden werden

- Im Sinne einer Prototypentwicklung sollen Funktionalität und Beweis der Konzeptplausibilität höhere Priorität als Benutzerfreundlichkeit haben
- In diesem Projekt soll die Eigenschwingung der angeschlagenen Oberfläche zur Auswertung genutzt werden.

Fragestellungen

Zur Untersuchung der Anwendbarkeit, wurden folgende Fragestellungen untersucht:

- Welche Oberflächen eignen sich am besten?
- Wie groß muss das Zeitfenster für einen Schlag gewählt werden, um eine robuste Erkennung sowie eine geringe Latenz zu erreichen?
- Welche Informationen können aus den Beschleunigungsdaten gewonnen werden, und welche eignen sich zur Echtzeit-Klassifizierung?
- Wie kann die Überlagerung zweier Anschläge verhindert oder verringert werden?

2. Konzept

Ein naiver Ansatz wäre, mehrere Sensoren auf der Oberfläche zu verteilen und eine ein- oder zweidimensionale Lokalisation mittels Bestimmung der Laufzeitunterschiede durchzuführen. Dies setzt allerdings die Verwendung mindestens zweier (eindimensional, max. 3 Positionen) bzw. dreier (zweidimensional) Sensoren. Die Verwendung mehrerer Sensoren bedeutet jedoch auch steigende Anforderungen an die verarbeitende Hardware, höherer Realisierungsaufwand, höhere Anforderungen an den Anwender und nicht zuletzt höheren Hardwareaufwand und -abhängigkeit, was die Portabilität und Verwendbarbarkeit deutlich einschränkt.

Daher soll lediglich ein Sensor verwendet werden. Dies bedeutet jedoch, dass die Unterscheidung der Anschlagpositionen lediglich anhand der Schwingungscharakteristiken erfolgen muss.

Wird eine Tischplatte an einer Position angeschlagen, führt diese eine Eigenschwingung aus. Je nach Material, Lagerung, Obejekte auf der Oberfläche, etc., unterscheiden sich verschiedene Tischplatten in ihrer Impulsantwort. Außerdem unterscheidet sich die ausgeführte Schwingung je nach Position und Art der Erregung. Ein Schlag mit den Fingerknochen erzeugt eine andere Schwingung, als ein Schlag mit der flachen Hand. Dies lässt sich anhand des entstehenden Geräusches verifizieren. Ebenso erzeugt ein Schlag auf die Tischkante eine andere Schwingung, als ein Schlag in die Tischmitte, welches sich vor allem in der unterschiedlichen Nachhallzeit zeigt.

Gewisse charakteristische Eigenschaften sind bauartbedingt durch den Tisch vorgegeben. Diese können vom Anwender nicht beeinflusst werden, folglich muss die Software in der Lage sein, diese Unterschiede in der Verarbeitung zu berücksichtigen. Trotzdem hat der Anwender gewisse Möglichkeiten, die Schwingung der Tischplatte zu beeinflussen, beispielsweise durch Anbringen von Massen, Dämpfern, etc. Dies kann und soll bewusst eingesetzt werden, um die Schwingungscharakteristiken der Anschlagspositionen möglichst unterschiedlich zu gestalten.

Um eine hohe Flexibilität in der Anwendung zu liefern, soll auf die Verwednung eines PCs/Laptops verzichtet werden. Stattdessen soll die softwareseitige Umsetzung mittels eines Smartphones erfolgen.

Zur Aufnahme der Schwingung wurden verschiedene Möglichkeiten erprobt:

- Die Verwendung eines Piezzosensors zur Aufnahme von Schwingungen.
- Messen der erzeugten Beschleunigung, mit Hilfe eines internen Beschleunigungssensors des Smartphones

- Das Messen mittels externer Beschleunigungssensoren eines Arduino Mikrocontrollers

3. Umsetzung

Für die Umsetzung wurde ein Android-Smartphone, ein Lenovo K6 mit einem Qualcomm Snapdragon 430 Prozessor [QT18] verwendet. Hingegen zu älteren Smartphones unterstützen Neuere den USB On-The-Go (OTG) Modus, womit die Stromversorgung des Arduinos über USB-Anschluss gewährleistet wird.

Desweiteren wurde ein Arduino CH340 Chip verwendet und als Beschleunigungssensoren das Modell MPU-6050. Zur Kommunikation mit dem Arduino und abgreifen der Beschleunigungssensoren des Sensors wurden die Bibliotheken *usb-serial-for-android* [mwm18] und *UsbSerial* [Her18] verwendet.

Die synthetisch erzeugten Schlagzeuggeräusche für Hihat, Snare und Bass wurden synthetisch mit PureData [uIoEMA18] erzeugt. Zur Nutzung der PureData-Patches auf dem Android Gerät, wurde die Bibliothek *pd-for-android* [lu18] von *libpd* verwendet. Diese unterstützt keine erweiterten Funktionen, wie sie in PureData Extended vorkommen.

Die Android-App wurde in Java geschrieben und unter der Laufzeitumgebung Android-Version 7.0 getestet. Abbildung 1 zeigt den Aufbau der Anwendung mit dem Arduino und den beiden MPU-6050 Sensoren. Die Sensoren werden bei Testzwecken am jeweiligen Objekt fixiert, so dass die Z-Achse des Beschleunigungssensors senkrecht zur Tischoberfläche steht.

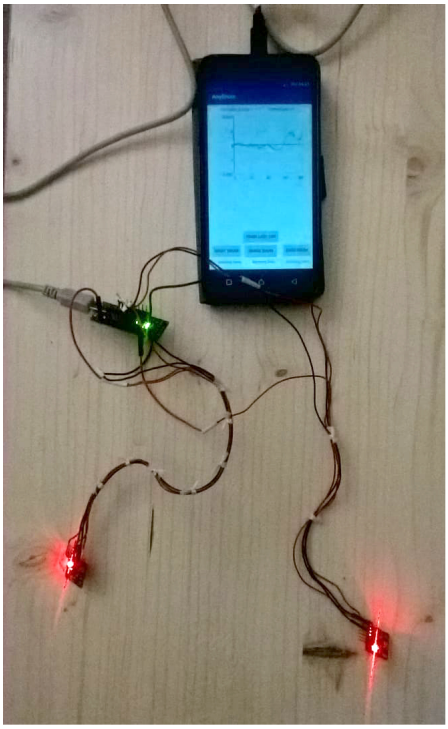


Fig. 1. Aufbau mit Arduino auf Holzplatte

A Arduino

A.1 Arduino Mikrocontroller

Verwendet wurde ein „Arduino Nano“Mikrocontroller mit Prozessor ATmega328P. Dieser führt selbst keine Datenverarbeitung aus, sondern bietet lediglich die Kommunikationsschnittstelle zwischen Smartphone und

Sensoren. Die Kommunikation zum Smartphone erfolgt über die Serielle Schnittstelle, die Kommunikation zum Sensor über den I^2C-Bus.

A.2 CH340

Der Arduino Nano ist mit dem Seriell-zu-USB-Chip CH340 ausgestattet. Laut Datenblatt [htt18] werden sowohl 5V, als auch 3.3V Spannungsquellen zum Betrieb unterstützt. Die Baudraten reichen von 50bps bis zu 2Mbps.

A.3 MPU-6050

Nach dem Datenblatt [Inc] besitzt der MPU-6050 Sensor ein Gyroskop und Beschleunigungssensoren für die X-, Y- und Z-Achse. Auf einer Oberfläche angebracht kann der Beschleunigungssensor verwendet werden, um Schwingungen der Oberfläche aufzunehmen. Die Abtastrate des Beschleunigungssensors beträgt max. 1kHz, der Messbereich je nach Konfiguration +-2g, +-4g, +-8g, +-16g. Für unser Projekt verwendeten wir den Beschleunigungssensor der Z-Achse im Wertebreich +-2g mit einer Auflösung von $16384\frac{LSB}{g}$.

B Signalanalyse

B.1 Anschlagserkennung

Zur Erkennung von Anschlägen und Abgrenzung von Hintergrundrauschen werden die Messdaten in Echtzeit geprüft. Hierzu werden die jeweils letzten beiden Messwerte miteinander verglichen. Ein Anschlag wird erkannt, wenn der aktuellere Wert den vorherigen Wert um ein bestimmtes Vielfaches der Standartabweichung übersteigt. Der Faktor sollte dabei auf das zu erwartende Rauschen abgestimmt werden. Höhere Werte neigen weniger dazu, Ausschläge im Rauschen bzw. unabsichtliche Berührungen der Oberfläche als Anschlag zu erkennen. Niedrigere Werte erlauben dagegen, auch sanfte Anschläge zu erkennen. In unseren Versuchen hat sich ein Faktor von sechs als stabil erwiesen.

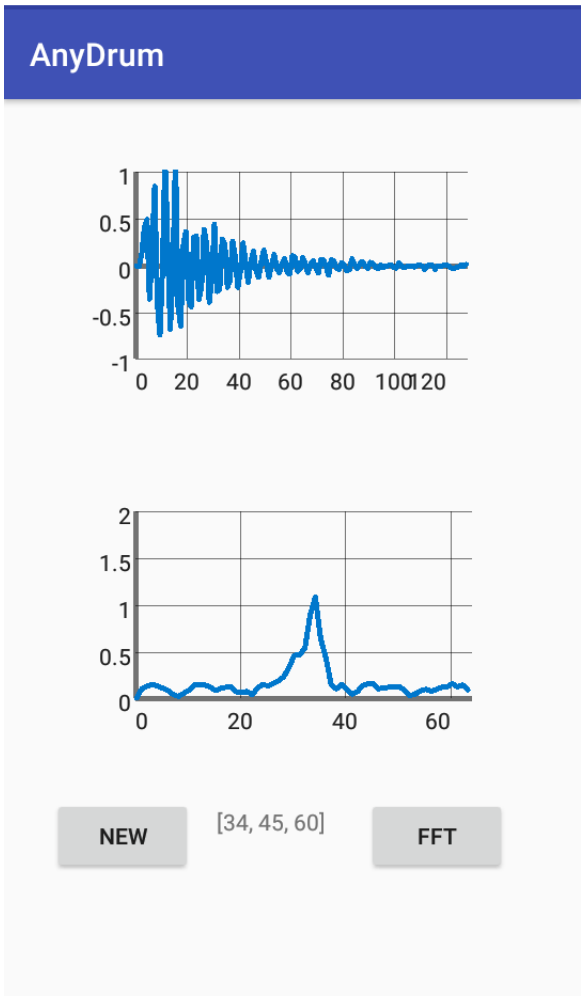
Wurde ein Anschlag erkannt, so beginnt ein Zeitfenster von ca. 500ms, in dem eingehende Messwerte gepuffert werden. Nach Ablauf des Zeitfensters wird dem Programm signalisiert, dass ein Schlag aufgenommen wurde. Zusätzlich werden die Messwerte bis ca. 20ms vor dem erkannten Anschlag in die Zeitreihe aufgenommen.

B.2 Analyse der Beschleunigungsdaten

Die Klassifizierung aufgenommener Schläge erfolgt mittels Auswertung der stärksten Frequenzen. Hierzu werden die Messwerte zunächst einer diskreten Fouriertransformation unterzogen, und die Spektrale Leistungssdichte der Zeitreihe bestimmt. In der aktuellen Implementierung konnten Abtastraten von ca. 200Hz erreicht werden. Die größte detektierbare Frequenz f_{max} liegt somit bei 100Hz. Für ein Zeitfenster von ca. 500ms ergeben sich 128 Messwerte. Nach Anwendung der Fouriertransformation erhält man somit 64 diskrete Frequenzgruppen mit einer Trennschärfe von ca. 2Hz.

Aus diesen Daten werden die vier Frequenzanteile mit der höchsten Amplitude ermittelt zur weiteren Verwendung gespeichert.

Abbildung 2 zeigt die Benutzeroberfläche und eine Darstellung der aufgenommen Schwingungen des Beschleunigungssensors. Dabei gibt die X-Achse den zeitlichen Verlauf in Abhängigkeit von der gewählten Abtastrate an und die Y-Achse die wahrgenommene Beschleunigung des Sensors in Y-Richtung.



Lernphase

Fig. 2. Analyse eines aufgenommenen Schlages mit FFT-Analyse und größten Maxima. Die Abbildung entstand mit einer frühen Version der Android-App, die noch den Beschleunigungssensor des Smartphones verwendete. Aufgrund der niedrigen Samplerate ist die Schwingung bereits nach 80 Messwerten abgeklungen. Weiterhin wurden zu diesem Zeitpunkt nur die drei stärksten Maxima ermittelt.

B.3 Lernphase

Zum Erlernen einer Schlagposition werden zwölf Schläge aufgezeichnet und wie oben in Abschnitt B.2 beschrieben analysiert. Anschließend wird der Durchschnitt der jeweils stärksten Frequenzgruppe, zweitstärksten Frequenzgruppe, usw. ermittelt, wie in Abb. 3 zu sehen ist. Diese werden als charakteristisches Muster gespeichert.

Tabelle 1 zeigt die zwölf Schläge zum Lernen für Hihat und Bass. Die Schläge wurden auf einer Tischplatte ausgeführt mit unterschiedlicher Position für HiHat und Bass. Es ist zu erkennen, dass die Unterschiede zwischen den Arithm. Mittelwerten deutlich größer sind, als zwischen den Median-Werte in Abb. 4. Aufgrund dieser Beobachtung wurde das Arithm. Mittel gewählt, um das charakteristische Muster zu erzeugen.

Tabelle 1. Stärkste Frequenzgruppen aus jeweils zwölf Schlägen für HiHat und Bass

HiHat					Bass			
Frequenzgruppe	F_1	F_2	F_3	F_4	F_1	F_2	F_3	F_4
	6	13	8	18	6	52	58	41
	15	19	7	10	6	14	18	10
	5	24	15	11	17	23	11	29
	14	18	7	26	6	27	11	31
	8	16	12	23	7	11	21	15
	5	12	16	21	6	27	15	10
	14	7	20	4	10	45	60	21
	8	30	41	14	6	10	15	27
	21	24	31	7	9	10	14	30
	12	18	23	35	9	11	5	18
	11	23	6	18	7	14	31	20
	17	11	7	4	6	29	36	58
Arithm. Mittel	11.3	17.9	16.1	15.9	7.9	22.8	24.6	25.8
Median	11.5	18	13.5	16	6.5	18.5	16.5	24

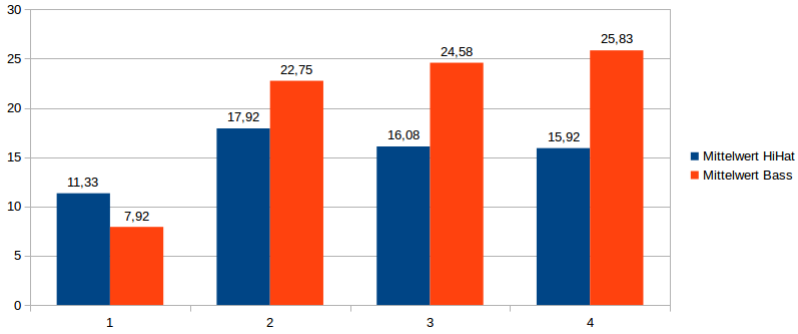


Fig. 3. Arithm. Mittel von Hihat und Bass für Frequenzen F1 - F4

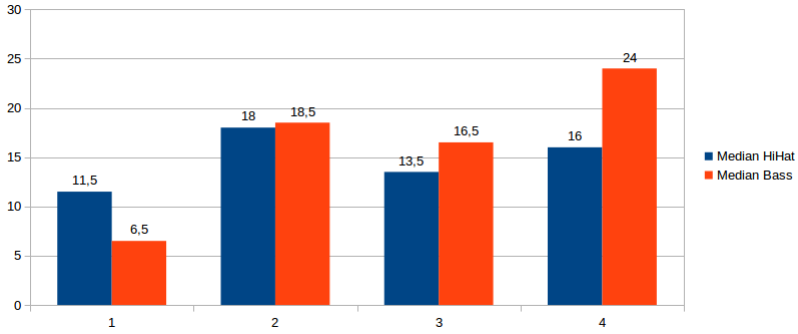


Fig. 4. Median von Hihat und Bass für Frequenzen F1 - F4

B.4 Echtzeit-Klassifizierung

Ist die Echtzeit-Klassifizierung aktiviert, so werden nach erfolgter Anschlagserkennung und Schlagaufzeichnung die Messwerte analysiert und mit denen der gelernten Positionen verglichen. Für diesen Vergleich werden die Frequenzabstände der jeweils stärkste Frequenzgruppen, zweitstärksten Frequenzgruppen, usw. ermittelt und die Teilergebnisse zu einer totalen Abweichung aufsummiert.

Nachfolgend ist so eine Klassifizierung mittels der im Abschnitt B.3 eingeführten Beispieldaten dargestellt. Nach erfolgter Lernphase wurde ein Anschlag aufgenommen und analysiert:

Dominateste Frequenzgruppen des Anschlages:

F_1	F_2	F_3	F_4
13	18	7	28

Berechnung der Frequenzabstände:

$$Error_{position} = \sum_{i=1}^{4} |F_{i,schlag} - F_{i,position}|$$

ergibt folgende Fehlerwerte:

$$\begin{aligned} Error_{HiHat} &= \sum_{i=1}^{4} |F_{i,schlag} - F_{i,HiHat}| \\ Error_{HiHat} &= |(13 - 11.3)| + |(18 - 17.9)| + |(7 - 16.1)| + |(28 - 15.9)| \\ Error_{HiHat} &= 23 \end{aligned}$$

$$\begin{aligned} Error_{Base} &= \sum_{i=1}^{4} |F_{i,schlag} - F_{i,Base}| \\ Error_{Base} &= |(13 - 7.9)| + |(18 - 22.8)| + |(7 - 24.6)| + |(28 - 25.8)| \\ Error_{Base} &= 30,7 \end{aligned}$$

Der Fehlerwert für HiHat ist geringer. Demzufolge wird die angeschlagene Position als HiHat-Position klassifiziert.

Erkennbar hierbei ist, dass in jedem Falle eine Zuordnung erfolgt, auch wenn die Abweichung sehr groß ist. Dies könnte jedoch leicht geändert werden, indem eine maximale Abweichung festgelegt wird und alle berechneten Abweichungen, die diesen Wert übersteigen, als nicht klassifizierbar eingestuft werden. Alternativ könnte, ähnlich zu anderen Clustering-Verfahren, eine Nachbarschaftsbeziehung zur Klassifikation verwendet werden.

B.5 Signalaufbereitung und Filterung

In der aktuellen Implementierung wird keine Filterung der Messdaten vorgenommen. Um die Genauigkeit der Frequenzanalyse zu erhöhen, sollten die Messwerte vor Anwendung der FFT mit einem Tiefpassfilter mit der Grenzfrequenz f_{max} gefiltert werden.

Eine Möglichkeit, um Rauschen zu unterdrücken wäre die Nutzung eines Grenzwertes. Auf diese Art können schwache Erschütterungen und leichte Vibrationen unterdrückt werden. Die Sensordaten könnten weiter geglättet werden durch Verwendung eines Box-Filters (Moving-Average) oder Gauss-Filters.

4. Auswertung

A Oberfläche

Welche Oberflächen eignen sich am besten?

Im Laufe des Projektes wurde eine Reihe von Testdurchläufen auf verschiedenen Oberflächen ausgeführt. Dazu wurden zwei Positionen mit jeweils 10 Schlägen angelernt und anschließend 100 Anschläge druchgeführt.

Die Dämpfung erfolgte durch eine flach auf den Tisch gelegte Hand realisiert. Zur Bewertung der Oberfläche wurde die effektive Erkennungsrate herangezogen.

Tabelle 2. Erkennungsrate verschiedener Oberflächen

Oberfläche	Erkennungsrate
Holztisch,	65%
Holztisch, mit Dämpfern	68%
Kunststofftisch, leer	60%
Kunststofftisch mit laufenden PCs	53%

In Tabelle 2 ist erkennbar, dass zwischen gedämpfter und ungedämpfert Holztischplatte kaum signifikante Unterschiede bestehen. Die Dämpfung könnte jedoch relevant werden, wenn ein schnelleres Spieltempo angestrebt wird, und Schwingungen aus Anschlägen sich überlagern. In unserem Versuch führten wir den nächsten Anschlag erst aus, wenn die Schwingung der Oberfläche durch den vorherigen Anschlag abgeklungen war.

Weiterhin ist erkennbar, dass die Erkennungsrate bei der Kunststofftischplatte geringer war als bei der Holzplatte.

B Zeitfenster

Wie groß muss das Zeitfenster für einen Schlag gewählt werden, um eine robuste Erkennung sowie eine geringe Latenz zu erreichen?

Zu Beginn des Projektes wurde zwischen Anschlag und Soundausgabe eine maximale Latenz von 100ms angestrebt. Im Laufe der Entwicklung mussten jedoch festgestellt werden, dass mit den verwendeten Methoden diese Latenzzeit nicht erreicht werden kann.

Um die Anschläge zuverlässig unterscheiden zu können, muss nach der FFT-Analyse eine ausreichend hohe Frequenzauflösung erreicht werden. Bei anfänglichen Versuchen mit 32 und 64 Messwerten pro Schlag zeigte sich, dass bei unterschiedlichen Anschlagspositionen die dominantesten Frequenzgruppen größtenteils identisch waren. Erst durch eine weitere Verfeinerung auf 128 Messwerte pro Schlag konnten zuverlässig Unterschiede in den dominantesten Frequenzen ermittelt werden.

Mit Verwendung des MPU6050 konnten Sampleraten von ca. 200Hz erreicht werden. Um 128 Messwerte aufzunehmen, muss somit ein Zeitfenster von ca. 500ms aufgenommen werden. Da während dieses Zeitfensters noch keine Analyse ausgeführt werden kann, ergibt sich die in der aktuellen Implementierung vorhandene, enorm hohe Latenz von ca. 550ms.

Um das ursprüngliche Ziel einer Latenz von <100ms einzuhalten, müsste eine deutlich höhrere Samplerate erreicht werden. Wir vermuten, dass bei einer Samplingrate von ca. 2.5kHz und einem Zeitfenster von 50ms zur Messwertaufnahme eine Latenz von <100ms gelichzeitig zu einer robusten Erkennung erreicht werden könnte.

C Analyse

Welche Informationen können aus den Beschleunigungsdaten gewonnen werden, und welche eignen sich zur Echtzeit-Klassifizierung?

Wir wählten den Ansatz der Klassifizierung anhand der dominantesten Frequenzen. Wie in der Projektarbeit "Virtuelles Schlagzeug"(Peetz/Ankermann/Ridder, Modul Virtuelle Realität, TU Dresden, 2017) beschrieben, ist damit eine zuverlässige Unterscheidung mind. zweier Anschlagspositionen möglich. Vorraussetzung ist jedoch eine Ausreichend große Abtastrate des Sensors, wie im Abschnitt „Zeitfenster“ beschrieben.

5. Diskussion

Literaturnachweis

[Her18] Felipe (felHR85) Herranz. Usbserial. https://github.com/felHR85/UsbSerial, 2018.

[htt18] http://wch.cn/. Usb to serial chip ch340. https://www.olimex.com/Products/Breadboarding/BB-CH340T/resources/CH340DS1.PDF, 2018.

[Inc] TDK Inc. Mpu 6000 and mpu 6050 product specification revision 3.4. https://www.invensense.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf.

[lu18] libpd (http://libpd.cc/). pd-for-android. https://github.com/libpd/pd-for-android, 2018.

[mwm18] mike w (mik3y). usb-serial-for-android. https://github.com/mik3y/usb-serial-for-android, 2018.

[QT18] Inc. and/or its affiliated companies Qualcomm Technologies. Snapdragon 430 mobile platform. https://www.qualcomm.com/products/snapdragon/processors/430, 2018.

[uIoEMA18] IEM (http://iem.at/) Institute of Electronic Music and Acoustics. Pure data. https://puredata.info/, 2018.