

*Tabgenerator: Gitarrennoten*

*Visualisieren*

Ben Kleinschnittger

Code: <https://github.com/SuperPirate-ai/TabGenerator>

**Inhaltsverzeichnis:**

Inhalt

[Einleitung 3](#_Toc157359248)

[Problemstellung 3](#_Toc157359249)

[Generelle Idee der Umsetzung 3](#_Toc157359250)

[Aufbau 4](#_Toc157359251)

[Open Source 4](#_Toc157359252)

[Grundidee 4](#_Toc157359253)

[Die Fast Fourier Transformation 5](#_Toc157359254)

[Das Userinterface 5](#_Toc157359255)

[Die Umsetzung 6](#_Toc157359256)

[Die Aufnahme einer Note 6](#_Toc157359257)

[Die Audioanalyse 6](#_Toc157359258)

[Berechnung der Notenfrequenz 6](#_Toc157359259)

[Findung der nächstliegenden Notenfrequenz 8](#_Toc157359260)

[Die Visualisierung 8](#_Toc157359261)

[Aufbau der Gitarre und der Tabs 8](#_Toc157359262)

[Bestimmung der Notenposition im visuellen Bereich 9](#_Toc157359263)

[Berechnung der tatsächlich gespielten Notenposition 9](#_Toc157359264)

[Anzeigen der Noten 10](#_Toc157359265)

[Das Speichern und Laden 10](#_Toc157359266)

# Einleitung

## Problemstellung

Bei einer Gitarre lernt man Lieder wie bei vielen Instrumenten mit Noten. Jedoch gibt es bei der Gitarre ein vereinfachtes System, welches speziell auf sie angepasst ist. Es nennt sich Gitarrentabs oder kurz einfach nur **Tabs**. Um einen Song in Tabs zu notieren, muss man, wie beim normalen Notensystem auch, jede Note **einzeln** eintragen. Ein Zeitaufwendiger Prozess der viele Nerven und Geduld kostet. Viel praktikabler wäre es diesen Prozess zu automatisieren. Und genau darum geht es auch in meinem Projekt. Ich möchte ein Programm entwickeln, dass Gitarrenspielern und Musikproduzenten hilft Songs in Tabs zu notieren. Dies soll automatisch passieren um das Songschreiben praktikabler und einfacher zu gestallten.

## Generelle Idee der Umsetzung

Dieses Projekt beschäftigt sich mit der **automatischen** Visualisierung dieser Tabs mithilfe eines selbstgeschriebenen **Computerprogrammes**. Durch dieses Programm soll das Eintragen der Noten automatisiert werden, **ohne** dass **spezielles Equipmen**t benötigt wird. Dieses Projekt widmet sich auch der Audioanalyse, einem zentralen Thema mit wachsender Bedeutung in der heutigen Industrie. Für mich ist dies das erste Projekt, das seinen Fokus auf die Audioanalyse legt und ich hatte beim Start dieses Projektes keine ausschlaggebenden Kenntnisse von diesem Fachgebiet der Informatik.

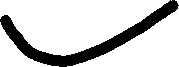
# Aufbau

Zur Umsetzung dieses Projektes nutze ich die **Spiele-Engine Unity** mit der Programmiersprache **C#**. Ich nutze hier eine Spiele-Engine, da ich sehr vertraut mit ihr bin und ich sie für diesen Zweck als sehr praktisch ansehe.

Der analoge Aufbau besteht aus einer **Gitarre** und einem **Audiointerface**. Dieses dient als Schnittstelle zwischen Gitarre und Computer.









# Open Source

Das Projekt wird nach der Entwicklung offen auf GitHub1 zur Verfügung stehen. Dadurch ist eine volle Transparenz bei der Entwicklung und ein freier Zugriff für jeden gegeben. Somit sollen möglichst viele Menschen erreicht werden.

# Grundidee

Die Grundidee besteht daraus mithilfe der **Fast Fourier Transformation** (FFT) das Ausgangssignal, in diesem Fall eine gespielte Note, zu analysieren, und somit die **stärkste Frequenz** in einem bestimmten Zeitintervall auszumachen. Dieser Frequenz, welche der gespielten Notenfrequenz entspricht, wird dann einer **Position im virtuellen Notensystem** zugeordnet und für den Nutzer dargestellt. Es besteht die Möglichkeit die gespielten Noten in einer Datei zu **speichern** und diese dann auch wieder zu **laden**.

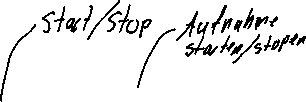
# Die Fast Fourier Transformation

Die **Fast Fourier Transformation**, kurz FFT, ist eine effiziente mathematische Operation zur Berechnung der **Diskreten Fourier-Transformation**. Die FFT ist wesentlich schneller als die herkömmliche Methode, da es mit dem **Teile-und-Hersche** Verfahren arbeitet. Dieses Verfahren teilt im Grunde ein Problem so lange rekursiv in kleine Teile, bis es im Teilproblem lösbar („beherrschbar“) ist. Anschließend wird die Teillösung genutzt, um die Lösung des Gesamtproblems zu rekonstruieren. Im großen Ganzen wandelt die FFT das Ausgangssignal vom **Zeitbereich** in den **Frequenzbereich** um. Wenn man das Signal in einem Graphen zeichnen würde, wäre vor der Kalkulation der FFT die X-Achse die **Zeit** und die Y-Achse die Amplitude, also die Lautstärke, des Signals. Nach der Kalkulation wäre die X-Achse die **Frequenz** und die Y-Achse die Amplitude. In diesem Projekt ist diese Funktion zur Frequenzbestimmung der einzelnen Noten von großem Nutzen. Für die Implementierung dieser Operation besteht die Möglichkeit sie entweder komplett selbst zu implementieren. Da dies aber mit sehr großem Aufwand verbunden ist und nicht dem Ziel dieses Projektes entspricht, wird in diesem Projekt eine Libary genutzt. Genauer gesagt die Accord.Math Libary. Die Implementierung der FFT findet man im AudioComponents Skript.

# Das Userinterface

A screenshot of a record

Description automatically generated



# Die Umsetzung

## Die Aufnahme einer Note

Zur Aufnahme einer Note wird das Signal, welches von der Gitarre „ausgesendet“ wird, zunächst mithilfe des Audiointerfaces auf den Computer übertragen. Da das ankommende Signal endlos weiter geht, also weder einen klar definierten Anfang noch Ende hat, wird es in kleine Samples unterteilt. Diese haben eine Buffersize von 8192 Sample. Bei einer zu kleinen Buffersize besteht die höhere Wahrscheinlichkeit, dass die Frequenz der Note auf zwei Samples aufgespalten ist. Die Aufnahme des Signals wird im MicrophoneInput Skript geregelt.

## Die Audioanalyse

Die Audioanalyse findet hauptsächlich im AudioAnalyser und im AudioComponent Skript statt.

### Berechnung der Notenfrequenz

#### Berechnung des Frequenzspektrums

Nachdem das Signal in kleine Samples aufgeteilt wurde, muss nun jedes einzelne Sample analysieren werden, um die Frequenz, der in dem Abschnitt gespielten Note, zu erfassen. Dazu ist es am effizientesten die oben beschriebenen Fast Fourier Transformation zu nutzen.

Ein Bild, das Text, Screenshot, Diagramm, Reihe enthält.

Automatisch generierte Beschreibung



Hier wurde ein D3 mit 145 Hz gespielt. Die gelbe Markierung entspricht der Notenfrequenz

#### Bestimmung der Hochpunkte

Wie man in der obigen Abbildung erkennt, kann man die gespielte Note anhand ihrer Amplitude (Lautstärke) bestimmen. Dazu schaut man sich alle Hochpunkte der FFTs an. Um Noise zu ignorieren, setzt man einen minimal Threshold für die Amplitude. Der tiefste Hochpunkt entspricht der fundamentalen Frequenz, also der Frequenz der gespielten Note.

#### FFT-Error

Nun ist es leider so, dass die FFT eine gewisse Ungenauigkeit bei der Frequenzbestimmung hat. Diese hat vor allem mit der Größe der Buffersize zu tun. Je größer, desto genauer. Allerdings bleibt immer ein gewisse Fehlerquote. Diese ist vor allem bei tiefen Noten ein Problem, da hier die Frequenzen sehr nah beieinander liegen. Um diese Abweichungen von der eigentlichen Frequenz auszugleichen, sind die Obertöne von Relevanz.

#### Obertöne und die genaue Frequenzbestimmung

Die Obertöne sind diejenigen Frequenzen, die mitschwingen, wenn eine Note, also eine bestimmte Frequenz gespielt wird. Sie sind immer ein Vielfaches der fundamentalen Frequenz. Wenn wir also ein tiefes A spielen, ist die fundamentale Frequenz 110 Hz, der erste Oberton 220 Hz, der zweite 330 Hz, der dritte 440 Hz und so weiter.

Nun ist es wichtig anzumerken, dass je höher die Frequenz einer Note ist, desto größer ist ihr Abstand zur nächstliegenden Note. Folglich ist es einfacher Noten mit hohen Frequenzen zu unterscheiden, als Noten mit niedriger.

Wenn wir nun den obigen Gedanken aufgreifen und ihn auf unser Problem mit der Ungenauigkeit der FFT anwenden, können wir uns die jeweiligen Obertöne anschauen, deren Frequenz bestimmen und diese dann auf die fundamentale Frequenz runterrechnen. Somit würden wir die Genauigkeit um einiges verbessern. Die Obertöne sind in der obigen Abbildung grün markiert.

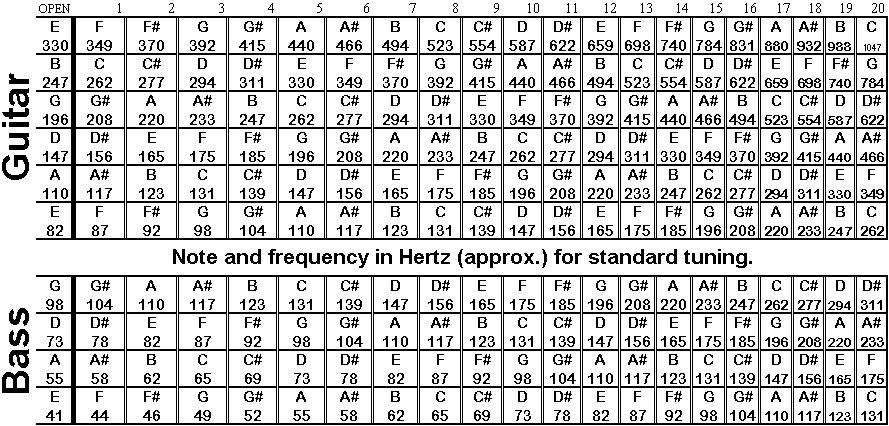
Allerdings haben wir etwas wichtiges außerachtgelassen. Um die Hochpunkte zu bestimmen haben wir einen Threshold gesetzt. Was ist aber nun, wenn wir uns die Obertöne anschauen und einer oder mehrere unter dem Threshold liegen? Dann würden wir nicht alle erfassen und da wir auch nicht wissen, welche Obertöne wir haben und welche nicht, könnten wir auch nicht sagen den wievielten Oberton wir haben. Das heißt wir können die fundamentale Frequenz nicht berechnen.

Um dieses Problem zu lösen, erhöhen wir iterativ den Index und teilen dann die Obertonfrequenz durch den Index und anschließend durch die bis jetzt ungenaue fundamentale Frequenz. Der Index, bei dem das Ergebnis am kleinsten ist, wird als Index für den Oberton verwendet und die vom Oberton resultierende fundamentale Frequenz als neue fundamentale Frequenz. Dies machen wir für jeden Oberton bis wir das genauste mögliche Ergebnis haben.

#### Saitenanschlagen erkennen

Bis jetzt ist es möglich die Grundfrequenz zu bestimmen. Allerdings besteht nun immer noch das Problem, dass man eben nicht immer exakt alle 8192 Samples eine neue Note spielt. Sondern eben nur dann, wenn eine Saite angeschlagen wird. Dafür muss man sich überlegen, wann genau eine neue Note angezeigt bzw. gespielt wird. Dies geschieht einmal, wenn sich die Frequenz

### Erschließung der nächstliegenden Notenfrequenz

Nachdem die Hauptfrequenz des jeweiligen Samples bestimmt und der Saitenanschlag erkannt wurde, muss jetzt die Hauptfrequenz, einer Frequenz auf dem Gitarrenspektrum zugeordnet werden.

[Bild1]

Hierbei wird analysiert, welche Frequenz auf dem Gitarrenspektrum der Hauptfrequenz am nächsten, unter Einberechnung des FFT-Fehlers, liegt. Dieser berechnet sich mit folgender Formel:

FFTError = Samplerate / Buffersize

Die Samplerate beträgt 44,1 kHz.

## Die Visualisierung

### Dean Guitars - WikipediaAufbau der Gitarre und der Tabs



[Bild2]

Der Gitarrenhals ist in sogenannte Bünde aufgeteilt. Diese sind jeweils mit Bundstäbchen (rot) getrennt. Außerdem hat er noch sechs Saiten (grün). Jeder Bund hat einen Halbton Abstand zum nächstgelegenen Bund. Also auf der obersten Saite, welche leer gespielt einem E entspricht, haben wir dann im ersten Bund ein F und im zweiten ein F# und so weiter.

Ein Bild, das Reihe, Zahl, Schrift, Text enthält.

Automatisch generierte Beschreibung

[Bild3]

Im Tabs-System gibt es ebenfalls diese sechs Saiten, als Striche dargestellt. Allerdings sind diese genau andersherum als auf der Gitarre. Also die höchste Saite ist oben und die tiefste unten. Die Bünde werden als Nummern auf der jeweiligen Saite dargestellt.

### Bestimmung der Notenposition im visuellen Bereich

Mithilfe der vorher ermittelten Frequenz ist es nun möglich die dazu passenden Positionen auf der Gitarre zu bestimmen. Da die ermittelte Frequenz häufig mehr als einmal vorkommt, werden mehrere mögliche Positionen ermittelt, die auf der Gitarre gespielt worden sein könnten. Zum Beispiel kann man das G4 auf der G-Saite im 12. Bund oder auch auf der D-Saite im 17. Bund gespielt werden.

### Berechnung der tatsächlich gespielten Notenposition

Nun herauszufinden, wo tatsächlich die Note gespielt worden ist, gestaltet sich etwas schwierig. Denn wir können ohne eine zweite Quelle, wie zum Beispiel eine Kamera, nicht so einfach sagen, wo die Note gespielt worden ist, da jede Notenposition exakt derselben Note und somit derselben Frequenz entspricht. Eine praktikable und einfache Möglichkeit besteht darin, von allen möglichen Notenpositionen diejenige zu wählen, die räumlich gesehen am nächsten zur Vorherigen steht. Allerdings können sich so sehr leicht Fehler einschleichen, da die nächstliegende Position nicht unbedingt immer die am einfachsten zu spielende Position für den Gitarristen sein muss. Wenn man es sehr genau machen möchte, besteht außerdem die Möglichkeit, saitenspezifische Audiomerkmale, wie zum Beispiel bestimmte Frequenzen, die nur auf einer bestimmten Saite vorkommen, zu identifizieren und anhand derer, die eigentlich identischen Noten voneinander zu differenzieren.

### Anzeigen der Noten

Nun wird die passende Notenposition als Notenobjekt dargestellt und mit einer bestimmten Geschwindigkeit abgespielt. Diese ist abhängig von den Beats per Minute (kurz: BPM).

## Das Speichern und Laden

Zusätzlich besteht die Option, die gespielten Noten in einer Datei zu speichern und diese dann zu einem anderen Zeitpunkt wieder abzurufen. Momentan ist dies nur in einem programmeigenen Format möglich. Dies soll noch so weit ausgeweitet werden, dass die Noten in der gebräuchlichen Form als .gp5 Datei abgespeichert werden können und somit mit den regulären Programmen, wie GuitarPro und Songsterr, nutzbar sind.

# Quellen

[Bild1] https://i.pinimg.com/originals/6c/3e/7b/6c3e7bb6076c2c9eff8918a28bd6e67e.jpg

[Bild2] http://www.edroman.com/guitars/dean/dimebag/dean\_from\_hell\_cfh\_enlarged.jpg

[Bild3] https://www.songsterr.com/a/wsa/george-benson-song-for-my-father-tab-s417389