# ECS7012P - Music and Audio Programming
# Final project: Pitch detection

Max Tamussino, 200579179

May 14, 2021

## Contents

# 1 Introduction

This report covers the design and implementation of a simple pitch detection device using the Bela Platform [1]. The frequency content of a pre-recorded file or of live microphone input is analysed using the fast fourier-transform (FFT) and the fundamental frequency is detected. A browser-based graphical user-interface (GUI) is used to present the calculated spectrum and show the detection result. Additionally, for the possible application of tuning instruments, useful information for tuning is shown. The spectrum graph axis may be edited by provided settings.

# 2 Background

There are many algorithms to detect the pitch of a signal. Most of these use the so-called autocorrelation function [2], less sophisticated methods just utilise the fourier transformation. Both mathematical concepts are briefly summarised.

## 2.1 Autocorrelation

The autocorrelation function (ACF) of any time-continuous real signal $x(t)$ is defined as the cross correlation with itself [3] at lag $\tau$, as given in Equation 1. Qualitatively, the value of the ACF is proportional to the degree of similarity [4] between the signal and itself shifted by lag $\tau$.

$$r_x(\tau) = \int_{-\infty}^{\infty} x(t) \cdot x(t+\tau)\, dt \tag{1}$$

The ACF shows a global maximum at $r_x(0)$ and local maxima at lags $\tau$ which correspond to the time period $P$ of a frequency component of $x(t)$ [5, 6]. To be able to set static thresholds, the normalised ACF $r'_x(\tau) = r_x(\tau)/r_x(0)$ $P$ may then be used to calculate the dominant frequency $f_0 = 1/P$ of the given signal. In the case of sampled discrete-time signals, which are given as blocks of finite length $N$, the autocorrelation is given in Equation 2.

$$r_x[k] = \sum_{n=0}^{N-1} x[n+k] \cdot x[n] \tag{2}$$

## 2.2 Fourier transformation

The fourier transformation (FT) decomposes signals into their individual frequency components. The mathematical form of this integral transformation and its inverse (iFT) for continuous-time signals $x(t)$ is given in Equation 3.

$$X(f) = \int_{-\infty}^{\infty} x(t) \cdot e^{-j2\pi ft}\, dt \qquad\qquad x(t) = \int_{-\infty}^{\infty} X(f) \cdot e^{j2\pi ft}\, dt \tag{3}$$

The power spectral density $S_x(f)$ may then be extracted from the magnitude of $X(f)$ by $S_x(f) = |X(f)|^2 = X(f) \cdot X^*(f)$. The discrete-time equivalent of the fourier transformation (again for finite blocks with length $N$ of sampled data), often called the DFT, is

given in Equation 4. The DFT yields exactly $N$ unique frequency *bins* (for real signals actually only $N/2$ bins are unique), where each index $k$ korresponds to a certain frequency $f_k = k \cdot \Delta f$ of the originally continuous sampled signal $x(t)$, where $\Delta f = f_s/N$ and $f_s$ is the sampling frequency. This statisfies the nyquist criterion, as for real signals sampled with $f_s$, only frequencies up to $f_s/2$ can be resolved.

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-j2\pi \frac{n}{N} k} \qquad x[n] = \frac{1}{N} \cdot \sum_{n=0}^{N-1} F[k] \cdot e^{j2\pi \frac{n}{N} k} \tag{4}$$

There are very efficient implementations of the DFT for block sizes $N = 2^n$. These fast implementations are referred to as fast fourier transform or FFT and are complexity order $O(N \log N)$ instead of the orginal FT with $O(N^2)$. Because of these, the ACF for large block sizes is often calculated by using the Wiener-Khintchine Theorem $r_x(\tau) = iFFT\left[S_x(f)\right]$ [4].

# 3 Design

The method used in this project primarily uses the obtained spectrum of the FFT to detect and calculate the fundamental frequency also referred to as *spectral peak picking* [7].

## 3.1 Input

The input signal is either sampled from a microphone or retrieved from recordings. The latter relies on sound files recorded with a sampling rate of $f_s = 44.1\,\text{kHz}$, as the same is used for live microphone input. In the case of live input, a simple electret microphone is connected to one of the Bela audio input channels as shown in Figure 1. The input method can be switched by pressing a button, which is located on the same breadboard as the microphone.

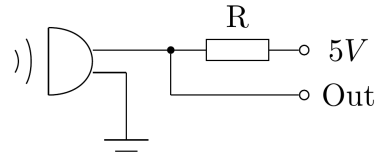


Figure 1: Wiring for the used electret microphone with $R = 10\,\text{k}\Omega$

## 3.2 Downsampling

As mentioned previously, a standard sampling rate of $f_s = 44.1\,\text{kHz}$ is used initially. For the block-based processing of this data, overlapping windows of length $N = 2048$ and a hop size of 512 were chosen. Applying the FFT directly to one window with $f_s$ would resolve frequencies up to $f_{max} = f_s/2 = 22.05\,\text{kHz}$. However, the frequency resolution calculates to $\Delta f = f_s/N = 21.5\,\text{Hz}$, which is unacceptable for tuning instruments. A

higher resolution could be reached by increasing $N$, however this is not feasible considering the amount of time these calculations would take for each window. The resolution was instead increased by downsampling the input signal by a factor of 16. This greatly improves frequency resolution to $\Delta f = f_s/(16 \cdot N) = 1.35\,\text{Hz}$. On the other hand, it leads to a significantly decreased maximum detectable frequency of $f_{max} = f_s/32 = 1.38\,\text{kHz}$, which is however sufficient for most instruments.

## 3.3 Maximum detection and fundamental check

If a full window of $N = 2048$ samples is assembled, the FFT is calculated. From the FFT, the discrete spectrum magnitude $S_x[k]$ is obtained and its maximum value detected. For some instruments however, the fundamental frequency is not the global maximum of the spectrum. Especially guitars often show the global maximum at the second or third harmonic [7]. This is why after the global maximum at the dominant frequency $f_d$ is found, the regions at one half and one third of $f_d$ are examined. Should one of these local maxima exceed 70 % of the global maximum, it is considered that $f_d$ is a harmonic of the fundamental frequency $f_0$. Otherwise, the dominant frequency is considered to be equal to the fundamental frequency.

## 3.4 Interpolation

The previously determined local maximum in the spectrum may be further refined. Because the maximum of the continuous spectrum is most likely not to coincide with the center of the frequency bin, parabolic interpolation as proposed in [8] is used to improve the FFTs accuracy beyond $\Delta f$. Using the two neighbour values to the local spectrum maximum, parabolic interpolation is performed by calculating the non-integer frequency index $k' = k_{max} + \Delta_k$ by Equation 5, which can then be used to calculate the refined frequency $f_0 = \Delta f \cdot k'$.

$$\Delta_k = \frac{S_x[k_{max}+1] - S_x[k_{max}-1]}{2 \cdot (2 \cdot S_x[k_{max}] - S_x[k_{max}+1] - S_x[k_{max}-1])} \tag{5}$$

## 3.5 User Interface

The user interface is displayed in the browser of a connected PC, as depicted in Figure 2. The spectrum of the given input is displayed in the graph, which also indicates where the current fundamental is detected by a vertical line. Below, both frequency and magnitude axis of the graph may be edited using the settings. Minimum, maximum and the label steps may be changed in the textboxes, pressing *Set* will then apply these changes. Pressing *Reset* will put the graph in its default state. Additionally, by pressing *Pause/Resume*, the graph may be stopped to have a closer look at the spectrum while the axis settings still are modifiable. Next to the settings, the results are presented - the fundamental frequency and its corresponding MIDI standard note number.

At the bottom right of Figure 2, the instrument tuning information is shown. The measured fundamental frequency will generally not perfectly match one note. In this case, the bar at the bottom will move out of its center. Should the instrument be far from calibrated, the bar moves in yellow and red areas respectively. The nearest note is usually

the tuning goal and is achieved when the bar moves directly into the center. Using english notation, the nearest note is therefore written above the center of the tuning bar. The next and previous notes are given at the sides.

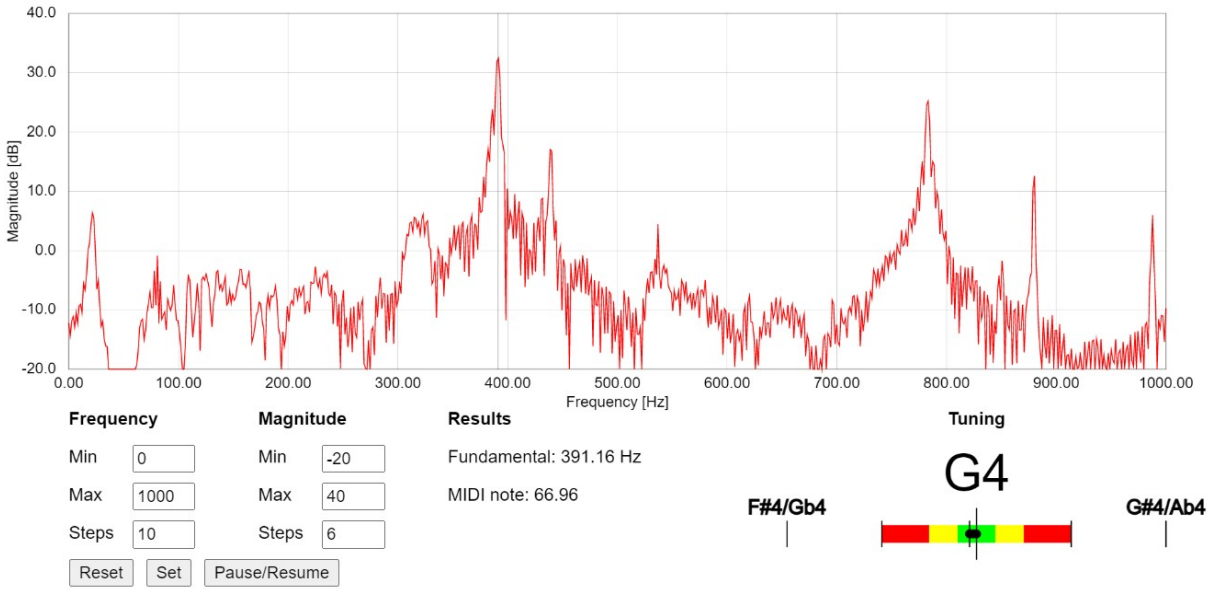


Figure 2: The browser-based GUI with spectrum graph, axis settings, results and instrument tuning information

# 4 Evaluation

The device was only tested using a reference tone generator, no real instrument was available. For every tested reference frequency, the device showed an error lower than 0.1 %. However, it must be said that the only noise present was a peak at 50 Hz, which was most likely due to the power connection of the microphone showing grid frequency. Recordings of pianos and guitars were tested and yielded the correct notes, however the tuning of these instruments is not known and the slight detuning which was measured could originate both in an error of the designed device or the actual instrument.

String instruments, like the guitar, which show stronger harmonics than fundamental frequencies in their spectrum, are considered to be the weakness of the spectrum peak-picking method. Allthough the wrong detection of second and third harmonics as fundamental frequencies was circumvented, the design is will certainly encounter difficulties with instruments that were not tested. Other approaches using the autocorrelation function could yield better results for those instruments. However, calculating the FFT twice is computationally highly expensive and did not yield statisfying image refresh rates for the GUI in this project. The PRAAT pitch detection algorithm proposed in [9] using advanced autocorrelation techniques would be an interesting alternative.

Additionally, other methons of picking up the instruments sound could be used. Mechanical vibrations, for example, could be measured with an accelerometer. This approach would yield better results in noisy environments, as acoustic noise would not interfere with measurements.

# References


[1] A. McPherson and V. Zappi, "An environment for submillisecond-latency audio and sensor processing on beaglebone black," *Journal of the audio engineering society*, May 2015.

[2] R. G. Amado and J. V. Filho, "Pitch detection algorithms based on zero-cross rate and autocorrelation function for musical notes," in *2008 International Conference on Audio, Language and Image Processing*, IEEE, Jul. 2008.

[3] P. S. Rao, S. Khoushikh, S. Ravishankar, R. A. Ananthkrishnan, and K. Balachandra, "A comparative study of various pitch detection algorithms," in *2020 5th International Conference on Computing, Communication and Security (ICCCS)*, IEEE, Oct. 2020.

[4] B. Apicella, A. Bruno, X. Wang, and N. Spinelli, "Fast fourier transform and autocorrelation function for the analysis of complex mass spectra," *International Journal of Mass Spectrometry*, vol. 338, pp. 30–38, Mar. 2013.

[5] L. Rabiner, "On the use of autocorrelation analysis for pitch detection," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 25, no. 1, pp. 24–33, Feb. 1977.

[6] M. Staudacher, V. Steixner, A. Griessner, and C. Zierhofer, "Fast fundamental frequency determination via adaptive autocorrelation," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2016, no. 1, Oct. 2016.

[7] A. Von Dem Knesebeck and U. Zölzer, "Comparison of pitch trackers for real-time guitar effects," in *Proc. of the 13th Int. Conference on Digital Audio Effects*, 2010.

[8] M. Gasior, "Improving FFT frequency measurement resolution by parabolic and gaussian spectrum interpolation," in *AIP Conference Proceedings*, AIP, 2004.

[9] P. Boersma, "Accurate short-term analysis of the fundamental frequency and the harmonics-to-noise ratio of a sampled sound," in *IFA Proceedings 17*, 1993, pp. 97–110.