Master of Science Thesis Project:

Implementation and Analysis of Pitch Tracking Algorithms

author:
Stefan Uppgård

IR-SB-EX-0117 2001-12-19

PREFACE

In this report, results are presented that forms the Master of Science thesis project in signal processing at KTH, Stockholm, Sweden. The author is studying at the electrotechnical program and the project has been performed at the Department of Signals, Sensors and Systems, S3. The thesis has been done commissioned and in the premises of Clavia DMI AB.

The project has consisted of a literature study, followed by computer simulations and finally the construction of a program for implementation in hardware. The work has been done in parallel with the Master of Science project by Leopold Roos at the department of Speech, Music and Hearing at KTH.

ABSTRACT

The purpose of the project has been to find a fundamental frequency tracker with good properties that can be implemented in the current target system.

Essential relevant mathematical signal theory is presented briefly, for the understanding of the algorithms that are presented.

Theories for the algorithms that have been found in an extensive literature study and information search is presented. The algorithms are introduced groupwise, divided in time and spectral domain algorithms. In this section, all algorithms are presented without consideration to the computational load it demands. Also presented are the reasons why some of these have not been further evaluated.

The results from the computer simulations that have been performed are presented along with a discussion of the benefits and disadvantages of the algorithm. Since the factor complexity has been very important during the analysis, this has been the reason why some algorithms have been sorted out at an early stage. Especially analyzed has been an algorithm written by *Cooper and Ng* [1], which has been modified and developed into a new algorithm that has been named *reduced ACF*.

In the study of algorithms, there is also a study of the so called pre- and postprocessing of data. It has been shown, that a proper preprocessing makes a great difference in the result that the algorithm presents. An approach has been chosen, using recursive setting of a lowpass filters cutoff frequency and recursive setting of the level for a method called center-clipping and compression.

The implementation of the developed algorithm in the present hardware, the target system, is described. An explanation for how the specification of the target system has influenced the choice of algorithm is given.

FÖRORD (Swedish Preface)

I denna rapport presenteras resultaten från det projekt som utgör ett examensarbete i signalbehandling vid KTH. Författaren läser vid elektroteknik linjen och examensarbetet har utförts vid institutionen för signaler, sensorer och system. Examensarbetet har utförts på uppdrag av och i lokaler hos Clavia DMI AB.

I arbetet har ingått en litteraturstudie, följt av datorsimuleringar och slutligen konstruktion av ett program för implementation i hårdvara. Arbetet har utförts parallellt med ett examensarbete utfört av Leopold Roos vid institutionen för tal, musik och hörsel vid KTH.

SAMMANFATTNING (Swedish Abstract)

Syftet med projektet har varit att finna en grundtonfrekvens följare med goda egenskaper som kan implementeras i det aktuella målsystemet.

Relevant grundläggande matematisk signalteori presenteras kortfattat, som grund för förståelsen av de algoritmer som presenteras.

Teorier för de algoritmer som har hittats i en omfattande litteraturstudie presenteras. Algoritmerna presenteras gruppvis uppdelade i tids- och spektral domäns algoritmer. I denna del presenteras alla algoritmer utan hänsyn till den beräkningsbörda de kräver. Här presenteras också anledningar till varför vissa av dessa inte har blivit fortsatt undersökta.

Resultat från de datorsimuleringar som har utförts presenteras tillsammans med en diskussion kring algoritmens för- och nackdelar. Då faktorn komplexitet har varit en viktig faktor vid analysen, har detta varit en anledning till att vissa algoritmer har sorterats bort på ett tidigt stadium. Speciellt analyseras en algoritm skriven av *Cooper and Ng* [1], vilken har modifierats och utvecklats till en ny algoritm som har namngivits *reducerad akf PDA*.

I studie av algoritmer ingår även studie av s.k. för- och efterbearbetning av data. Det har visat sig att en ordentlig förbearbetning gör stor skillnad på det resultat som algoritmen presenterar. Ett tillvägagångssätt med rekursiv bestämning av ett lågpassfilters skärfrekvens och rekursiv bestämning av nivån för en metod kallad centerklippning och kompression.

Implementationen av den framtagna algoritmen i den befintliga hårdvaran, målsystemet, beskrivs. En förklaring till hur målsystemets specifikationer har inverkat på valet av algoritm.

ACKNOWLEDGEMENTS

I would like to thank Leopold Roos who has been performing his Master of Science thesis also at Clavia DMI AB. Thank you for nice co-operation and discussions.

A thanks goes also to the tutors, Peter Händel at S3 KTH and Magnus Kjellander at Clavia DMI AB. Magnus has at all times, even when being occupied, been willing to answer any question.

Stefan Uppgård
2001-11-15

1. INTRODUCTION 8
1.1 BACKGROUND 8
1.2 PURPOSE 8
1.3 FUNDAMENTAL ABBREVIATIONS / BASIC TERMINOLOGY 8
2. THEORY OF MUSICAL SIGNALS 9
2.1 PROPERTIES OF MUSIC 9
2.2 MUSICAL INSTRUMENTS 9
2.3 THE HUMAN VOICE 9
2.4 FUNDAMENTAL FREQUENCY 9
2.5 FORMANTS, HARMONICS, SUBHARMONICS AND PARTIALS 9
2.6 HUMAN HEARING MECHANISM 9
2.7 PITCH AND PERCEPTION OF PITCH 10
3. THEORY OF SIGNAL PROCESSING 11
3.1 MATHEMATICAL MODEL FOR THE SIGNAL 11
3.1.1 Waveform 11
3.1.2 Time-Variant Systems 11
3.1.3 Windowing 11
3.2 SHORT TIME SIGNAL ANALYSIS TOOLS. TEMPORAL ANALYSIS 11
3.2.1 Autocorrelation 11
3.2.2 AMDF 12
3.2.3 ASDF 12
3.3 SHORT TIME SIGNAL ANALYSIS TOOLS. SPECTRAL ANALYSIS 12
3.3.1 DFT and FFT 12
3.3.2 DSTFT 12
3.3.3 The Constant Q Transform 12
3.3.4 Multiple Spectral Transform, Cepstrum 12
3.3.5 Wavelets 12
3.4 DIGITAL FILTERS 13
3.4.1 FIR and IIR Filters 13
3.4.2 Resonance Filter 13
3.4.3 Comb filtering 13
3.5 ANALOG TO DIGITAL CONVERSION (SAMPLING) 14
4. INTRODUCTION TO THE SCIENCE OF PITCH DETERMINATION 15
4.1 HISTORY OF PITCH DETERMINATION 15
4.2 PITCH DETERMINATION FOR SPEECH VS. PITCH DETERMINATION FOR MUSIC 15
4.3 ONLINE VS. OFFLINE / REALTIME VS. NOT REALTIME 15
4.4 PROPERTIES OF A PDA 15
4.5 TYPICAL DIFFICULTIES IN A SOUND SIGNAL FOR PITCH DETERMINATION 16
5. ALGORITHMS, THEORY AND EVALUATION 17
5.1 HOW THE ALGORITHMS ARE GROUPED 17
5.2 TIME DOMAIN PDA 17
5.2.1 Common features 17
5.2.2 Common drawbacks 17
5.2.3 PDA: LP and Threshold Crossing Analysis 17
5.2.4 PDA: Two Non-zero Threshold Crossing with Hysteresis 18
5.2.5 PDA: Simple Envelope follower 18
5.2.6 PDA: Extended Envelope follower 19
5.2.7 PDA: Peak Detector by Reddy 19
5.2.8 PDA: Rabiner and Gold 19
5.2.9 PDA: Simple PDA with filterbank preprocessing 20
5.2.10 PDA: eSFRD 20
5.2.11 PDA: Pisarenko method and Yule-Walker method 20

5.2.12 PDA: LPC Analysis 22
5.2.13 PDA: Adapting 2 pole notch filter with RLS or LMS 22
5.2.14 PDA: Autocorrelation 22
5.2.15 PDA: AMDF 23
5.2.16 PDA: Maximum-Likelihood 23
5.2.17 PDA: On-Line LS-fit 23
5.2.18 PDA: Cooper and Ng 23
5.2.19 PDA: The Reduced ACF 24
5.3 SPECTRAL DOMAIN PDA 24
5.3.1 Common features 24
5.3.2 Common drawbacks 24
5.3.3 PDA: FFT, Maximum of FFT, Division method 24
5.3.4 PDA: FFT, Distance of spectral peaks 25
5.3.5 PDA: FFT, Piszczalski and Galler 25
5.3.6 PDA: Spectral Compression and Harmonic summation 26
5.3.7 PDA: Constant Q transform 26
5.3.8 PDA: Phase Vocoder 27
5.3.9 PDA: Auto- and Cross-Correlation techniques in spectral domain 27
5.3.10 PDA: Cepstrum 28
5.3.11 PDA: Wavelets 28
5.3.12 PDA: Comb filters 29
5.3.13 PDA: Generalized Spectrum 29
5.4 COMBINED TIME AND SPECTRAL DOMAIN PDA 30
5.4.1 PDA: Rough FFT and LS-fitting 30
5.4.2 PDA: Loose-Harmonic Matching 30
5.5 OTHER PDA 30
5.5.1 PDA: Neural Network 30
5.5.2 PDA: Heterodyne filtering 31

6. RESULTS, SIMULATIONS AND EVALUATIONS 32

6.1 ERROR ANALYSIS 32
6.2 RESULTS TIME DOMAIN PDA 32
6.2.1 Results: Threshold Crossing 32
6.2.2 Results: Envelope Following PDA 33
6.2.3 Results: Peak Detecting PDA 34
6.2.4 Results: Simple Basic Extractor with Filterbank Preprocessing 34
6.2.5 Results: One pole notch adaptation with RLS and LMS 35
6.2.6 Results: Autocorrelation 35
6.2.7 Results: AMDF 36
6.2.8 Results: Cooper et al 36
6.2.9 Results: Reduced ACF 36
6.3 RESULTS SPECTRAL DOMAIN PDA 38
6.3.1 Results: FFT with different harmonic analysis 38
6.3.2 Results: Constant Q 39
6.3.3 Results: Phase Vocoder 39
6.3.4 Results: Cepstrum 39
6.4 RESULTS COMBINED PDA 40
6.4.1 Results: Rough FFT and sinusoid adaptation 40

7. PREPROCESSING 41

7.1 LOWPASS FILTERING 41
7.2 BANDPASS FILTERING 41
7.3 HIGHPASS FILTERING 41
7.4 FILTERING ADAPTIVELY 41
7.5 INCREASING RESOLUTION THROUGH INTERPOLATION 42
7.6 SPECTRAL FLATTENING 42
7.6.1 Cubic non-linear distortion 42
7.6.2 Center Clipping 42

7.6.3 Center Compression 42
7.6.4 Sign function 43
7.7 ADAPTIVE SPECTRAL FLATTENING 43
7.8 DOWNSAMPLING THE INPUT SIGNAL 43
7.9 INVERSE FILTERING 43

8. POSTPROCESSING 44

8.1 ONLINE AND OFFLINE POSTPROCESSING 44
8.2 TIME TO FREQUENCY CONVERSION 44
8.3 LINEAR SMOOTHING 44
8.4 NON-LINEAR SMOOTHING 44
8.5 DE-STEP FILTER 44
8.6 CONFIDENCE COUNTER 44
8.7 USING “RULES” 44

9. DEVELOPMENT AND IMPLEMENTATION OF AN EFFICIENT PDA IN A DSP 45

9.1 THE PROJECT SPECIFICATION 45
9.2 ABOUT THE NORD MODULAR SYSTEM 45
9.3 THEORY OF THE ALGORITHM 45
9.4 IMPLEMENTATION IN THE DSP 46
9.5 PDA-RELATED FUNCTIONS 46
9.5.1 Event Detection 46
9.5.2 Gate signal 46
9.5.3 Signal level 46
9.6 FUTURE IMPROVEMENTS 46

10. CONCLUSIONS 47

10.1 SPECTRAL AND TIME DOMAIN PDA 47
10.2 PRE- AND POSTPROCESSING 47
10.3 TODAY’S PROCESSORS AND MEMORY INFLUENCE OF HARDWARE WHEN CHOOSING PDA 47
10.4 THE FUTURE OF MUSICAL PITCH TRACKING 47

APPENDIX A 48

RESOURCES 51

BIBLIOGRAPHY AND REFERENCES 52

1. Introduction

The background and purpose of this project is presented.

1.1 Background

Pitch determination is a vast area, often discussed in speech analysis/synthesis applications. For these, a number of PDA's have been developed.

It also has a great interest in a musical context. For a musician playing his natural instrument, it would add a new dimension, having the possibility to control electronic instruments.

This was noticed by Clavia, who gave the author and Leopold Roos, the mission to find a PDA that would be possible to include in an existing musical synthesis system.

1.2 Purpose

The purpose of the project has been to find a fundamental frequency tracker with good properties that can be implemented in the current target system (ch.9). This has been done through an extensive information search, studying what techniques are used in similar systems.

The different algorithms found has been simulated and one of the algorithms have been selected for implementation in the target system (9.2).

Throughout the report the concentration and visual angle has been on low computational complexity. This will be reflected in the discussions of the different pitch determination algorithms.

1.3 Fundamental Abbreviations / Basic terminology

Abbreviations:

ACF	- Autocorrelation Function
PDA	- Pitch Determination Algorithm
DSP	- Digital Signal Processor
DFT	- Discrete Fourier Transform
FFT	- Fast Fourier Transform
STFT	- Short Time Fourier Transform
F_s	- The Sampling Frequency
F_0	- The True Fundamental Frequency
$\hat{F}_0$	- The Estimated F_0
T_0	- The True Fundamental Period
$\hat{T}_0$	- The Estimated T_0

Basic Terminology:

Basic Extractor - The fundamental part of the PDA. The block that actually performs the pitch detection, different from pre- and postprocessing.

Short-Term Analysis - The method of studying a sampled signal segmentwise.

2. Theory of Musical Signals

Here some basic theory of musical signals is explained and the properties that are interesting when doing pitch detection are discussed.

2.1 Properties of Music

Music is according to Olson [2], '*the art of producing pleasing, expressive, or intelligible combinations of tones. Noise is any undesired sound.*' The later was considered true in the 50's, but is not necessarily true today. What is true, is that noise lacks pitch.

A musical sound wave can be characterized by six physical variables: *Frequency, intensity, waveform, duration, growth and decay,* and *vibrato*.

Frequency is the number of waves per second. *Intensity* is the sound energy per unit of time through a unit area. *Waveform* is a periodic soundwave made up of the fundamental frequency and overtones. *Duration* is simply the length of time that the tone lasts. *Growth and decay* describes the amplitude variation of the tone. These characteristics are in general exponential functions. Duration, growth and decay are often replaced by the more common *ADSR*-function (*Attack, Decay, Sustain, Release*). Here attack, decay and release are time duration's and sustain is a level. *Vibrato* designates primarily a frequency modulation of the musical tone, but is also followed by an amplitude modulation. Pure amplitude modulation is called *tremolo*.

2.2 Musical Instruments

'*A musical instrument is a system for producing one or more pleasing tones.*' That's the beautiful definition by Olson [2].

The monophonic instrument plays only one tone at a time, while the polyphonic instruments can play several tones simultaneously.

The sound is generated when a resonant, or multiresonant system is excited by a musician.

2.3 The Human Voice

The periodic sound from the human voice is generated at the larynx by periodic vibrations of the vocal cords. These are voiced sounds such as {e} and {a}. Unvoiced sounds such as {s} and {f} are noisy, non-periodic signals, created in the mouth cavity.

2.4 Fundamental frequency

Olson [2] defines the fundamental frequency as '*the frequency component of the lowest frequency in a complex sound'*. Hess [3] only makes a definition for voice signals: *T_0 is defined as the time between two successive laryngeal pulses*.

Using a waveform approach, the author would like to define the fundamental period as: *The length of time between two successive repetitions of the sound waveform.*

2.5 Formants, Harmonics, Subharmonics and Partials

A harmonic is an overtone whose frequency is an integral multiple of the fundamental frequency. A subharmonic is an integral submultiple of the fundamental frequency. The harmonics are generated along with the fundamental frequency when the musical instrument's multiresonant[1] system is excited.

A formant is a component or a resonant frequency of the voice system or an instrument, that does not change despite a change of the pitch. An example is the acoustical guitar where, in better instruments, the lid is tuned to a certain resonance frequency [4].

The fundamental frequency and its harmonics, produced by the musical instrument, are normally situated in the nearness of the formants. If the formants are not exactly harmonically spaced, the instrument is inharmonic.

The actual overtones produced by an instrument are called partials, and hence, not always situated at the theoretical harmonics.

2.6 Human Hearing Mechanism

The human ear is the destination for all resynthesized music. A short description of its function is at place.

When the sound enters the ear canal, it hits the eardrum, which vibrates with a motion corresponding to the ripple of the sound wave. The motion of the eardrum is transported to cochlea[2], where there are 4000 nerve fibres running to the brain.

The cochlea is frequency-selective, and is in effect a sound analyzer with a very fast response. It's capable of distinguishing 1500 separate frequencies [2].

[1] The resonant frequency in a first-order acoustical system f_{ra} and for a electrical system f_{re} are given by

$$\left\{ f_{ra} = \frac{1}{2\pi\sqrt{MC_A}} \qquad f_{re} = \frac{1}{2\pi\sqrt{LC_E}} \right\}$$

where M = inertance, C_A = acoustical capacitance, L = inductance and C_E = electrical capacitance.

[2] Cochlea - The inner ear, where the mechanical force is transformed into hydraulic pressure.

2.7 Pitch and Perception of Pitch

Pitch is a psychological tonal characteristic of music, which arise from frequency, and may assign to a position in a musical scale. It is subjective in character and can only be found by the mean results from a number of tests [2].

The limits for pitch are given by the lowest and highest frequency that gives a sensation of tone, which is about 20 – 20000 Hz. The higher limit is decreased with age.

Pitch discrimination is the difference of pitch that an individual can detect. The ear is more sensitive to frequency changes at the higher frequencies [2].

In early research, the fundamental was regarded as playing a dominant role in pitch perception. Hess [3] says that '*recent theories and models, according to experimental evidence, postulate that pitch perception is performed by harmonic pattern matching. ... All the spectral pitches together contribute to the overall pitch perception*'. It is even so that a virtual pitch at the fundamental frequency can be percepted, even though the fundamental present in the signal is very weak.

PDA's that use spectral partial analysis, can imitate the way pitch perception works to resolve the fundamental. Brown stated that her algorithm [5], described in 5.3.9, was consistent with the pattern matching theory.

3. Theory of Signal Processing

Basic signal processing and mathematical theory is presented to facilitate the understanding of the following chapters.

3.1 Mathematical Model for the Signal

Periodic signals can be represented by the Fourier coefficients. For a discrete-time periodic signal *x(n),* its Fourier series representation can be expressed as

$$x(n) = \sum_{k=0}^{N-1} c_k e^{j2\pi kn/N} \qquad (3.1)$$

where the coefficients are given by

$$c_k = \frac{1}{N}\sum_{n=0}^{N-1} x(n) \cdot e^{j2\pi kn/N} \qquad (3.2)$$

A discrete-time signal with fundamental period N consists of frequency components separated by *2π/N* radians. Since the frequency range for discrete-time signals is unique in the interval (0,*2*π*)*, the Fourier series will contain at most *N* frequency components. The *N* values in equation (3.1) and (3.2) will therefore suffice to represent the periodic signal having fundamental period *N*.

The expected signal sampled in this project is in general non-stationary, but during a short time, it can be restricted to being stationary or at least quasi- stationary. A few number of fundamental periods can be considered a short time (See 3.1.2). This holds at least for musical signals.

Using this, the input signal *s(n)* to the PDA can during a short time be described by *x(n)* above, plus additive noise *w(n).*

$$s(n) = x(n) + w(n) \qquad (3.3)$$

The noise component *w(n)* is expected to contain white gaussian noise and rowdy elements such as key sounds from the instrument, breath noise and weak signals from interfering sound sources.

3.1.1 Waveform

A waveform is a two dimensional representation of one period of the sound signal. It is determined by any setting of the coefficients in eq. (3.2). The waveform can have the characteristic and appearance of e.g. a simple sinusoidal form, or more complex like a triangular shape, consisting of an infinite number of odd Fourier series components.

3.1.2 Time-Variant Systems

The musical signal change with time and is therefore time-variant, i.e. parameters such as pitch change with time. For short-term analysis the assumption is made that the parameters are constant or quasi-constant during a frame length *K*.

3.1.3 Windowing

A *window function* (or *weighting function*) is a function that is multiplied to the signal to get a short-time representation. The window has characteristic properties within a short interval, and values zero outside the interval. A basic example is the rectangular windowing function,

$w_r(k) = 1$; $k = -K/2, ..., K/2 - 1$ (3.4)
$w_r(k) = 0$; $k = otherwise$

Other used window functions is e.g. the Hann window

$w_h(k) = 0,5 * (1 + \cos (2\pi k/K))$ (3.5)

and Hamming window

$w_h(k) = 0,54 + 0,46 * \cos (2\pi k/K).$ (3.6)

Windowing signals for spectral analysis will distort the spectral estimate due to *leakage*[3] and it reduces the spectral resolution. The effect of leakage can be reduced by choosing another window function than the rectangular. Both (3.5) and (3.6) have lower sidelobes in the frequency domain and will therefore reduce the leakage. This is done at the cost of a loss of resolution.

Windowing signals for temporal analysis has been shown [3], at least for the AMDF (3.2.2), to deteriorate pitch determination results (i.e. other windows than the rectangular window). A windowed signal is obviously, in time domain, less periodic than an unwindowed one.

3.2 Short Time Signal Analysis Tools. Temporal Analysis

Functions used in temporal analysis is presented below.

3.2.1 Autocorrelation

Correlation is a measure of similarity. The crosscorrelation of two sampled signals *x(n)* and *y(n)* is given by

$$r_{xy}(k) = \lim_{N \to \infty} \frac{1}{2N+1} \sum_{n=-N}^{N} x(n)y(n+k) \qquad (3.7)$$

The maximum value of $r_{xy}(k)$ at k_{max} will indicate how much one of the signals must be shifted as to make *x(n)* and *y(n)* most similar.

The autocorrelation function (***acf***) is a special case of the crosscorrelation where the correlated signals are similar.

$$r_{xx}(k) = \lim_{N \to \infty} \frac{1}{2N+1} \sum_{n=-N}^{N} x(n)x(n+k) \qquad (3.8)$$

[3] Leakage it the phenomen that the signal power has "leaked out" to the entire frequency range.

If a stationary periodic signal is autocorrelated, the distance between the maximum peaks will be equal to the fundamental period.

The autocorrelation can efficiently be calculated using the *FFT* (3.3.1), using the fact that the acf is the inverse Fourier transform of the power spectrum of the signal. The power spectrum $S_{xx}(f)$ is also calculated using the Fourier transform. ($X(f)$ is the Fourier transform of the signal $x(n)$).

$$r_{xx}(k) = \sum_{-1/2}^{1/2} S_{xx}(f) e^{j2\pi fn} df \qquad (3.9)$$

$$S_{xx}(f) = |X(f)|^2 \qquad (3.10)$$

3.2.2 AMDF

Like autocorrelation, the AMDF *(Average Magnitude Difference Function)* is also a measure of similarity and periodicity. The function is expected to have strong minimum when the shifting variable k in 3.11 becomes equal to the period T_0 of a quasiperiodic signal $x(n)$ of length K.

$$a(k) = \frac{1}{K} \sum_{n=q}^{q+K-1} |x(n) - x(n+k)| \qquad (3.11)$$

The minimum would be zero in case the input signal x(n) was exactly periodic.

3.2.3 ASDF

The ASDF (*Average Squared Difference Function*), can be defined

$$ASDF(k) = \frac{1}{K} \sum_{n=0}^{K-1} |x(n) - x(n+k)|^2 \qquad (3.12)$$

3.3 Short Time Signal Analysis Tools. Spectral Analysis

Functions used in spectral analysis are considered here.

3.3.1 DFT and FFT

The discrete Fourier transform (***DFT***), defined in 3.13, plays an important role in many applications of digital signal processing, including linear filtering, correlation analysis and spectrum analysis.

$$X(f) = \sum_{n=-\infty}^{\infty} x(n) e^{-j2\pi fn} \qquad (3.13)$$

In practice, the spectrum can only be approximated from a finite data record. This finite observation interval puts a limit on the frequency resolution, the ability to distinguish two frequency components, to $f_{res} = 1 / LT_s$, where L is the number of samples in the DFT and T_s is the sampling period.

An important reason for the great use of DFT is the existence of efficient methods to calculate it. Often used fast Fourier transform (FFT) algorithms are radix-2 and radix-4 FFT algorithms where the number of data in the FFT is of power 2 or power 4 respectively.

3.3.2 DSTFT

The DSTFT (*Discrete Short-Time Fourier Transform*) is the Fourier transform used on a single frame of length N. Most often the length N is set so that the signal is approximately stationary throughout the whole window.

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2\pi kn/N} \quad k = 0,\ldots,N-1 \quad (3.14)$$

3.3.3 The Constant Q Transform

The constant Q transform is based on transforming the Fourier transform into log frequency domain. This is useful, since the tones of the western musical scale are geometrically spaced. Constant Q refers to constant frequency to resolution, $Q = f / df$. Method:

1. Choose minimal frequency f_0 and the number of bins per octave b. Let f_{max} be the maximal frequency.
2. $K = \left\lceil b \cdot \log_2\left(\frac{f_{\max}}{f_0}\right) \right\rceil$
3. $Q = (2^{1/b} - 1)^{-1}$
4. $N_k = \left\lceil Q \frac{f_s}{f_k} \right\rceil$
5. $X^{cq}(k) = \frac{1}{N_k} \sum_{n<N_k} x(n) w_{N_k}(n) e^{-j2\pi nQ/N_k}$

3.3.4 Multiple Spectral Transform, Cepstrum

The Cepstrum has been used to separate the spectral content from the pitch frequency of speech. It's calculated in three steps:

1. Analyzing the signal $x(n)$, calculate its DFT $x(e^{j2\pi k/N})$
2. Calculate its logarithm $\log\left|x(e^{j2\pi k/N})\right|$.
3. Calculate the inverse DFT, resulting in $c_p(n)$.

3.3.5 Wavelets

There are many variations to the wavelet transform, only mentioned here is the orthogonal wavelet, which is the one that have been used in PDA's.

The study of wavelets is the study of bases spanning the signal space, which have the property that all basis functions are self-similar, which means that they only differ by translation and change of scale from one another.

In [6] a wavelet is visualized as a damped sine wave whose amplitude is very small, perhaps zero, outside some bounded interval, and which is distorted some in its shape to guarantee the orthogonality conditions to hold.

The discrete wavelet transform is based on sampled two-channel perfect reconstruction filterbanks with halfband highpass and halfband lowpass branches. If the transform is iteratively applied to its lowpass branch, it is in each iteration additionally scaled. Figure 3.a describes the scheme for a multi-scale wavelet transform (here 2-scale).

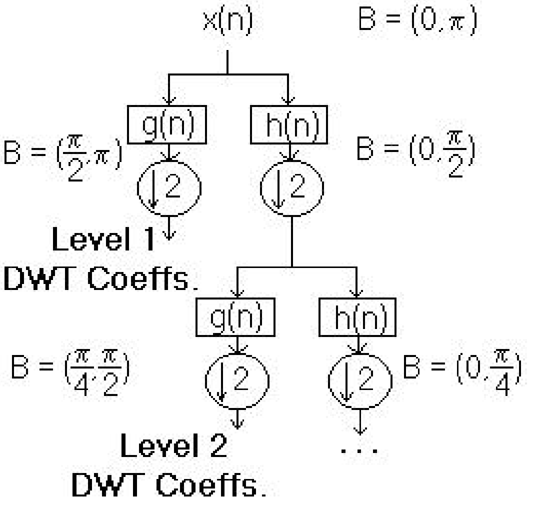


figure 3.a) The discrete wavelet transform

Here, *g(n)* is a highpass halfband wavelet filter and *h(n)* is the complementary lowpass wavelet filter. The output of *h(n)* is the lowpass residue for the level *x* filter branch. The highpass subband for the *g(n)* branch, represented by *Level x DWT Coeffs.*, is called the wavelet function. The result after desired number of iterations is then a coarse estimate of the original signal in the lowpass branch. The outputs of *g(n)* consists of successively finer details of the highpass channels. B is the bandwidth at each level.

The original signal can be perfectly reconstructed by inverse-filtering through the filterbank.

In summary, the wavelet analysis filterbank, derives the coefficients for the linear expansion of the signal with respect to the basis functions corresponding to the impulse responses of the synthesis filterbank. For an introduction to wavelet transforms, [7] is a nice resource.

3.4 Digital Filters

Any device that converts an input signal *x(n)* to an output signal *y(n)* is a digital filter. But discussed here, are the linear digital filters having a transfer function *H(ω)* that defines an analytical relation between the Fourier transforms of the input- and output signal, $H(\omega) = Y(\omega)/X(\omega)$.

3.4.1 FIR and IIR Filters

The low-, band- and highpass filters discussed in this report are of either finite or infinite impulse response type (FIR or IIR). If there is a requirement of linear-phase characteristics, FIR filters are most often used. If phase distortion is tolerable or unimportant a IIR filter is used, since its implementation involves fewer parameters, less memory and has lower computational complexity. A filter can be described by the difference equation

$$y(n) = -\sum_{k=1}^{N} a_k y(n-k) + \sum_{k=0}^{M} b_k x(n-k) \quad (3.15)$$

where the frequency response mentioned above is given by

$$H(\omega) = \frac{\sum_{k=0}^{M} b_k e^{-j\omega k}}{1 + \sum_{k=1}^{N} a_k e^{-j\omega k}} \quad (3.16)$$

For the FIR filter, $\{a_k\} = 0$.

3.4.2 Resonance Filter

The digital resonator is a two-pole filter with the pair of complex conjugate poles located near the unit circle. The magnitude of the frequency response is large (it resonates) in the vicinity of the pole location. The zeros can be located in the origin or commonly at $z = 1$ and $z = -1$. The later will set the response to zero at frequencies $\omega = 0$ and $\omega = \pi$.

3.4.3 Comb filtering

Taking a FIR filter with system function

$$H(z) = \sum_{k=0}^{M} h(k) z^{-k} \quad (3.17)$$

and replacing *z* by z^L, where *L* is a positive integer, results in the new FIR filter

$$H_L(z) = \sum_{k=0}^{M} h(k) z^{-kL} \quad (3.18)$$

If the frequency response of the original filter is H(ω), the frequency response of 3.18 will be

$$H_L(\omega) = \sum_{k=0}^{M} h(k)e^{-jkL\omega} = H(L\omega) \qquad (3.19)$$

Thus, the frequency response of $H_L(\omega)$ is an L-order repetition of $H(\omega)$ in the range $0 \le \omega \le 2\pi$.

3.5 Analog to Digital Conversion (Sampling)

If the highest frequency contained in an analog signal *x(t)* is f_{MAX}, and sampled at a rate $F_S > 2*f_{MAX}$, then the signal *x(t)* can be exactly recovered from its sample values using the sinc interpolation function

$$g(t) = \frac{\sin 2\pi f_{MAX} t}{2\pi f_{MAX} t} \qquad (3.20)$$

The sampling rate $F_S = 2*f_{MAX}$ is called the Nyquist rate.

4. Introduction to the science of Pitch Determination

Pitch determination is equivalent to fundamental frequency estimation, while pitch perception is connected to the subjective understanding by the human being (2.7).

Pitch analysis is often linked to a resynthesis step, as when used in voice coding. Thus, it normally has not only a analysing purpose, but is also followed by a system where the result in practical will be applied.

This subsequent system is then the director for what results the pitch determining system must achieve, e.g. what measuring range the system must handle and the necessary resolution.

4.1 History of Pitch Determination

The oldest techniques for pitch determination are auditory and manual pitch determination systems. The auditory system simply consists of the human ear and the brain, where the limit according to Hess [3], is the human mind. This is because the mind is unable to follow fast short-term variations of the pitch in a signal such as the voice. However, when there is a stationary behaviour, as in musical signals, the auditory pitch determination system performs better. The ability of a person to quantitatively determine the fundamental of a periodic sound without a known reference is called absolute pitch.

Time-domain manual pitch determination, simply means determining the pitch from a visual display of the signal. The first frequency-domain manual pitch determination, became possible with the first mechanical spectrograph in 1946 [3].

Most studies on pitch determination, has been made for voice signals. The goal has been to improve voice coding algorithms used in tele communications. Determining musical pitch for technical devices became more interesting with the introduction of the MIDI – standard[4]. Pitch-to-MIDI converters are by musicians well-known devices .

4.2 Pitch Determination for Speech vs. Pitch Determination for music

Fundamental frequency estimators for speech do not in general assume the strong harmonic structure that is present in a musical signal. The speech algorithms also operate in a much narrower frequency range. Furthermore, the demand for a fast response time is more important in a musical context.

4.3 Online vs. Offline / Realtime vs. not Realtime

An attempt to distinguish online processing versus realtime processing is proposed by Hess [3]. *Realtime processing* is said to take place, when the calculation of pitch for a signal segment takes less time than the segment itself. *Online processing* occurs when a result is presented in one step, without needing a considerable amount of future data. A PDA is said to be instantaneous when it operates both online and in realtime.

4.4 Properties of a PDA

A PDA used for musical signals should preferably cover a huge fundamental frequency range. An electric bass having the lower string tuned to H0, corresponds to 30,9 Hz. An acoustical piano could be expected to have tones played up to C7, corresponding to 3951,1 Hz. In other words, the measuring interval can at least be restricted to 30 - 4000 Hz.

Reducing the measuring interval will most often improve the performance of the PDA. According to H.F.Olson [2], the following fundamental frequency ranges can be expected:

• Singing Voice	: 80 – 1000 Hz
• Piano	: 30 – 5000 Hz
• Saxophone	: 50 – 1500 Hz
• Trumpet	: 150 – 1000 Hz
• Flute	: 300 – 3000 Hz
• Acoustic Guitar	: 70 – 700 Hz

These values are only approximates, and a singing voice can, e.g. when singing a piece by Mozart, demand a range of 50 – 1800 Hz [3].

The resolution, or measurement accuracy needed, is definitely all depending of what the pitch estimate will be used for in the subsequent system. However, for musical applications one must assume that an upper bound, should be set by half the distance between two adjacent semitones. This approximately corresponds to a relative difference of *2,5 %* in Hz. In Hess [3], it's been concluded that for "perfect pitch determination", an accuracy of *0,3 %* to *0,5 %* is needed.

Response time is crucial for demanding musicians. Delays of 30 ms are very noticeable. Depending on the PDA, the response time is either fixed, or within an interval. A response time of 5 ms is by the author considered desirable, but also extremely fast. A frequency of 30 Hz corresponds to a fundamental period of 33 ms. A fundamental period of 5 ms corresponds to a fundamental frequency of 200 Hz.

[4] The MIDI standard was introduced in Los Angeles 1983 [8], as a communication protocol for musical instruments.

The contradiction of low frequency range and fast response time is clear. A fast response time is also hindered by the fact that during the attack of a new tone, pitch determination is most difficult (4.5).

4.5 Typical difficulties in a sound signal for pitch Determination

In general, when talking of signals having a fundamental frequency, one normally assumes there are formants present, and it is no longer possible to speak of the signal as being simple.

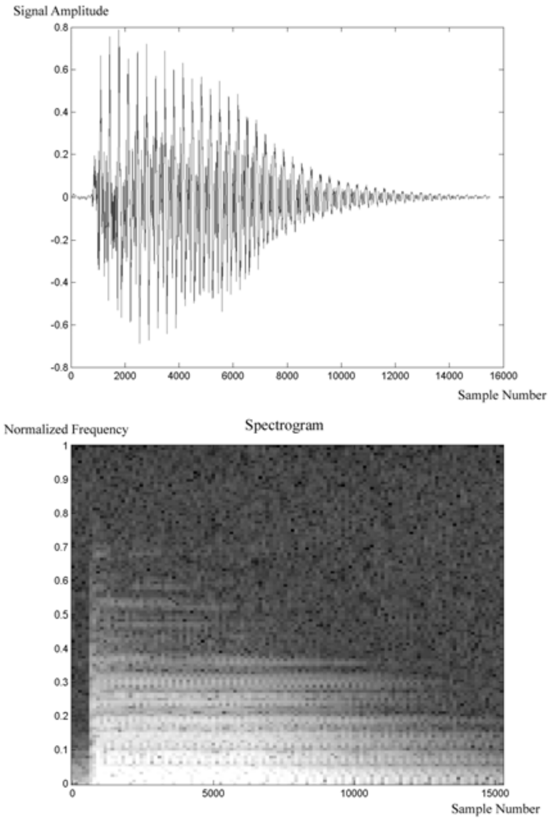


figure 4.a) A piano tone and its spectrogram.

As mentioned in the preceding section, the nature of the signal makes the fast response demand a tedious task. In figure 4.a, the signal of a tone played from a sampled acoustical piano is plotted along with its spectrogram.

At the attack, not only the fundamental at the normalized frequency 0.2 is present, but **almost the whole spectrum**. As the tone diminishes, the fundamental and its harmonics are made clear.

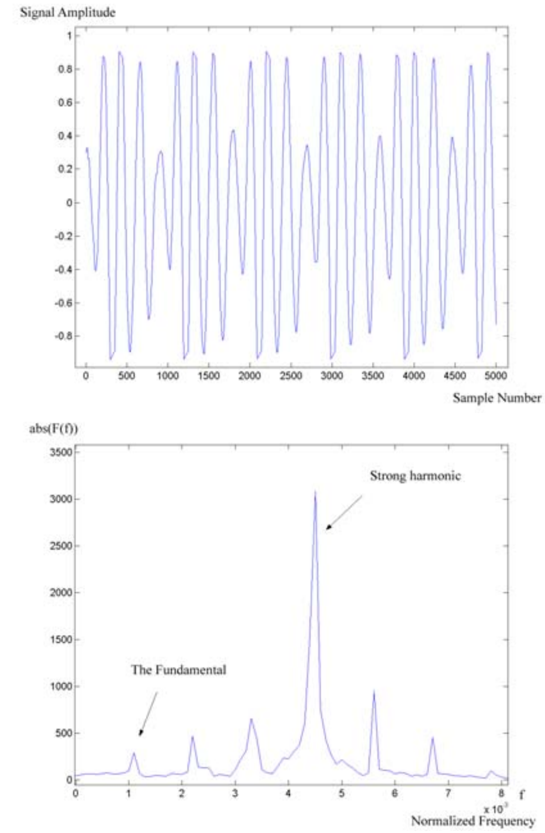


figure 4.b) A El.Bass tone with weak fundamental

The presence of higher harmonics is the main reason for pitch detection errors and it often results in octave detection errors. As can be seen in figure 4.b, the fundamental can sometimes be so weak that it's hard to tell if it's present at all. Here, the third harmonic is very strong and PDA's will have problems to separate this frequency from the fundamental frequency.

In played music, one note is after a changeover followed by another. During this transition both tones may be present and that makes pitch determination during attacks even harder. Two tones being played simultaneously, as at transitions, are said to be duophonic.

Furthermore, an obstacle is that a musical sound is non-stationary and may have a new appearance from period to period (3.1.2).

5. Algorithms, Theory and Evaluation

A discussion of the pitch determining algorithms that have been found in an extensive literature search are discussed.

5.1 How the algorithms are grouped

The algorithms described below are grouped in time domain PDA's, spectral domain PDA's, combined PDA's and other PDA's.

The classification of algorithms is difficult, since some algorithms use calculations having results that are valid for both temporal and spectral domain, e.g. AR-modeling of a signal. A common classification is time-domain and short-term PDA's [3]. However, classifying algorithms into time and spectral domain is to the author an intuitive approach.

Also, the algorithms are discussed as complete pitch determination algorithms more than basic extractors. This is because the simulations have been performed on complete algorithms. A basic extractor is here considered a fundamental function, and not being subject to dissection. As the algorithms are described, it will be clear which basic extractor has been used.

5.2 Time Domain PDA

In this section the PDA's that operate in time-domain, i.e. directly on the sampled data, are discussed.

5.2.1 Common features

Pitch tracking in time domain is by some seen as an old-fashioned method. When the techniques of doing transforms to other domains were developed, it was considered as if the answer to all musical signal processing problems would be solved there.

Still, for a problem such as pitch tracking, it is not crystal clear that frequency domain operations always are superior.

The benefit of studying the signal in time domain for pitch tracking, is that the analysis can be performed at sample basis instead of at buffered intervals. No transformation is needed, which is an advantage if the algorithm is restricted by computational load. Also, in almost all situations it is possible to know the fundamental period of a signal in time domain, just by studying it with the human eye. If algorithms are made that are so intelligent, time domain will do.

5.2.2 Common drawbacks

When there are strong harmonics in the signal and the fundamental is weak or even missing, classical time-domain functions such as peak-picking or the envelope follower will have a tedious task. There is no way of easily extracting the formant structure and benefiting this, as there could be in spectral domain.

A *DC – offset* will for some time domain algorithms render the result useless. This could of course be solved by high pass filtering.

5.2.3 PDA: LP and Threshold Crossing Analysis

The most elementary extractor of periodicity has since the first days of pitch determination been the study of polarity changes of a signal. This technique is generally referred to as *zero crossing analysis*, since the change of polarity in time domain is described by the signal curve as crossing the abscissa. Having undistorted simple signals like e.g. a sinusoidal signal, this basic extractor will be enough to find the period. The period time is then, the time distance between two subsequent positive zero crossings, where a ***positive zero crossing*** is defined as a change from negative to positive polarity. However, musical signals can not be generalized as having simple properties, and therefor this technique will undoubtedly fail.

A generalisation of the discussion above about having the signal cross the abscissa, would be to let the signal cross a threshold where its level is arbitrary set somewhere along the ordinate. The technique would then not only be restricted to being called zero crossing analysis, but instead *threshold crossing analysis*. Putting the threshold at precisely the correct level, the performance would be improved to handle more complicated waveforms as described in figure 5.a. Of course, since we are, as explained in 3.1.2, processing time-variant signals, it is impossible to know where to put the threshold level correctly.

This extractor is like practically all other extractors preprocessed by a lowpass filter.

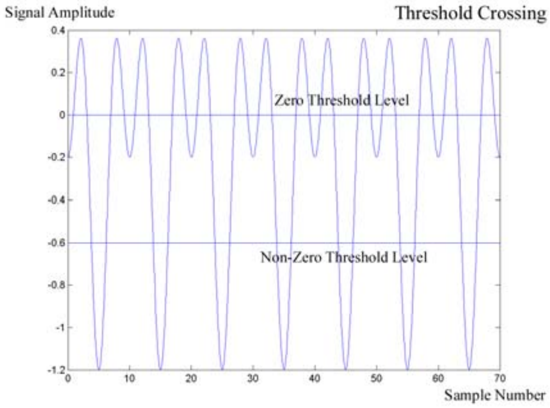


figure 5.a) Single Threshold Crossing Analysis

5.2.4 PDA: Two Non-zero Threshold Crossing with Hysteresis

From the approach described in 5.2.3, one could improve the performance by adding a second threshold. This second threshold would then be put at a negative level corresponding to the rate of the positive threshold or simply at zero level. The gain introduced by a second threshold is that each individual threshold can be crossed an infinite number of times, but only when the two thresholds are crossed successively in a defined sequence, an indication for a start of a new period is set. (figure 5.b) The method is said to have hysteresis behaviour, since it is a non-linear input-output system with memory.

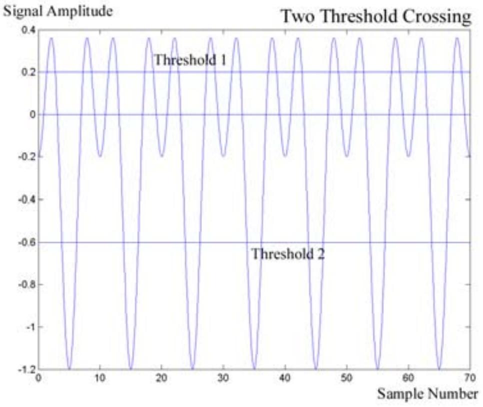


figure 5.b) Two Threshold Crossing Analysis

5.2.5 PDA: Simple Envelope follower

In 1954, Ladislav O. Dolansky presented a paper with the headline: “An Instantaneous Pitch-Period Indicator” [9]. This was the introduction of an algorithm that have had a grand influence of all pitch determination since then. Though introduced in analogue electronics, the technique is still used in today’s digital environment.

The basic extractor used in this method is said to be a “simple envelope follower”. Figure 5.c describes pretty well the function. The pitch period can be derived from the envelope by setting marks where the signal exceeds the envelope. Alternatively a peak finder will indicate period. A third way would be to use the zero crossings of the signal following a discontinuity of the envelope follower.

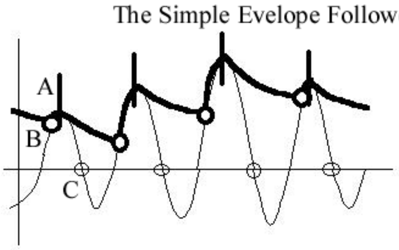


Figure 5.c) Envelope follower with periodicity cues

5.2.6 PDA: Extended Envelope follower

The result from one simple envelope follower can be improved by adding another envelope follower to another. But there in between, a *first-order highpass filter* is put. Filtering the envelope signal will have the result displayed in figure 5.d. Dolansky[5] made a circuit that repeated this procedure six times, ideally resulting in an output signal consisting of positive pulses at the beginning of each fundamental period.

The envelope follower is not restricted to being used only on the positive part of the signal. Another one could simultaneously be following the negative side. The problem arise then of knowing which of the two results to choose. J. Engdegård [10] implemented a system using cascade linked envelope followers at both positive and negative amplitudes. Each of the positive and negative curves resulted in individual pitch estimates. The decision of selecting the positive or negative results were solved by selecting the one having the smallest variance a number of estimates back.

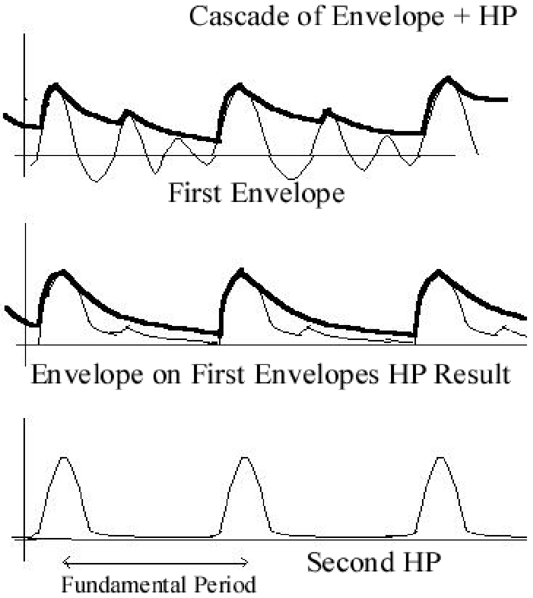


figure 5.d) Highpass filtered envelope

5.2.7 PDA: Peak Detector by Reddy

An early method using peaks of the sound signal as cues for calculating the fundamental period, is the algorithm made by Reddy in 1966. The method includes a simple basic extractor as well as a global correction routine. The global correction routine is discussed in 8.1. The basic extractor is typical for how peak detecting algorithms can be designed.

Small blocks of the signal, about 25 ms, are examined in each cycle of the algorithm. All local maxima and minima in this block are determined. After that the algorithm determines "significant" maxima from the local maxima following the rules[6]:

The peak
- is positive
- does not occur within 2,5 ms from the previous significant maximum
- is

  a) either greater than 0,9 times the absolute minimum

  b) greater than the linearly extrapolated value from the previous two significant maximum peaks

  or

  c) if neither a) nor b) is satisfied within 13,5 ms of speech from the previous significant maximum, then the maximum of all the local maxima in that 13,5 ms of speech

A significant minimum is defined similarly. The limitations in time (2,5 ms and 13,5 ms) would according to Hess [3] indicate a range of the fundamental frequency from 75 Hz to 400 Hz. Finally, a significant peak is defined as a significant max having a significant min within 3,5 ms of its occurrence. The significant peaks are further refined in the global correction algorithm, before being calculated into pitch estimates.

This algorithm is instructive as an example for how rules (8.7) can be used for improving the performance of a PDA.

5.2.8 PDA: Rabiner and Gold

Like Reddy's algorithm in 5.2.7, the PDA by Rabiner and Gold (1969) explore the structure of the waveform directly in the time domain. This algorithm has become very well known and has often been cited and referred to when analysing other algorithms. Figure 5.e. explains what characteristics of the waveform are examined.

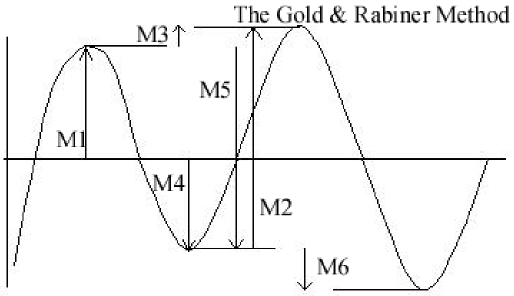
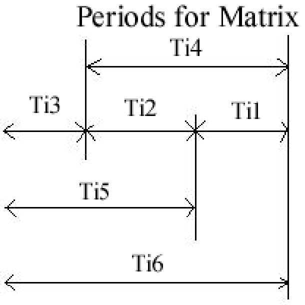


figure 5.e) The peak values M1 – M6

[5] Dolansky's paper is discussed in section 5.2.5.

[6] The algorithm is reproduced as it was summarized by Hess [3].

The six individual peak values M1-M6 are separately examined in a simple envelope-following like extractor. That way six different proposals for the pitch period are deduced from these six independent envelope followers. The six period values are put in a matrix according to figure 5f. Each of the elements in the first row is compared to the other 35 elements. The element having the most coincidences[7] is chosen as the fundamental frequency.



Period	*Matrix*				
T_{11}	T_{21}	T_{31}	T_{41}	T_{51}	T_{61}
T_{12}	T_{22}	T_{32}	T_{42}	T_{52}	T_{62}
T_{13}	T_{23}	T_{33}	T_{43}	T_{53}	T_{63}
T_{14}	T_{24}	T_{34}	T_{44}	T_{54}	T_{64}
T_{15}	T_{25}	T_{35}	T_{45}	T_{55}	T_{65}
T_{16}	T_{26}	T_{36}	T_{46}	T_{56}	T_{66}

figure 5.f) The periods T_{i1} to T_{i6} are the period values corresponding to extractor i (with characteristic value M_i. The most recent period is T_{i1} and T_{i2} and T_{i3} are past values. T_{i4} to T_{i6} are calculated according to the figure.

5.2.9 PDA: Simple PDA with filterbank preprocessing

An approach where the input signal has been separated into different channels through a bandpass filterbank has been examined. The filterbank are set with geometrically spaced center frequencies so that each filter corresponds to one musical octave. Each output has been linked to a basic extractor, and the result is chosen from the filter containing most energy.

The use of filterbanks can be compared to the discrete wavelet transformation approach in 5.3.11.

5.2.10 PDA: eSFRD

Enhanced super resolution F0 determinator [11]. The algorithm was developed by Bagshaw (1994) and is a modified and improved version of an algorithm SFRD, by Medan, Yair & Chazan (1991).

The method analyses the signal frame wise. The largest amount of samples needed for each frame will be decided by the frequency interval where the fundamental is looked for, say $[f_{min}, f_{max}]$. The number of samples needed will then be $N_{max} = F_s / f_{min}$, resulting in a frame length of $3N_{max}$ since the frame is defined as:

$$s_N = \{ s(i) \mid i \in [-N_{max}, 2N_{max}] \}.$$

Each frame is divided into three consecutive sequences each having a variable length n. The three sequences are defined as:

$$x_n = \{ x(i) = s(i-n) \mid i \in 1,\dots,n \}$$
$$y_n = \{ y(i) = s(i) \mid i \in 1,\dots,n \}$$
$$z_n = \{ z(i) = s(i+n) \mid i \in 1,\dots,n \}$$

A value for n is looked for, such that each segment x_n , y_n *and* z_n will contain the fundamental period. By calculating a normalized cross correlation coefficient (5.1) for each n in the interval $[N_{min}, N_{max}]$, candidates for the fundamental period will be taken as the local maxima exceeding a threshold level[8]. The same calculations as for $\rho_{x,y}(n)$ will be calculated for $\rho_{y,z}(n)$. Candidates appearing in both calculations are given 2 points, and the other candidates will be given 1 point. The winning candidates are further compared in a normalized cross correlation (5.2). These remaining candidates n_m, are listed in order of size of the fundamental period with n_1 as the shortest period and n_M the longest period. The greatest value of (5.2) will settle the winning period estimate.

$$\rho_{x,y}(n) = \frac{\sum_{j=1}^{[n/L]} x(jL) \cdot y(jL)}{\sqrt{\sum_{j=1}^{[n/L]} x^2(jL) \cdot \sum_{j=1}^{[n/L]} y^2(jL)}} \quad (5.1)$$

$$\rho\,(n_m) = \frac{\sum_{j=1}^{n_M} s(j-n_M) \cdot s(j+n_M)}{\sqrt{\sum_{j=1}^{n_M} s^2(j-n_M) \cdot \sum_{j=1}^{n_M} s^2(j+n_M)}} \quad (5.2)$$

5.2.11 PDA: Pisarenko method and Yule-Walker method

Non-parametric methods for estimating the power spectrum make no assumption for how the analysed data is generated. Moreover, they are relatively simple and easy to compute using the FFT algorithm. In the search for a PDA that does not

[7] A coincidence is given when the absolute difference between the elements under consideration is less than a given threshold.

[8] The threshold level is chosen empirically.

assume anything of the source signal this approach is superior. However, the drawback of these methods are that they need a long data record in order to obtain the necessary frequency resolution required. Furthermore, the methods suffer from spectral leakage effects due to windowing.

Contrary to the non-parametric methods, the parametric methods models the received data sequence as the output of a linear system characterized by a rational system function having corresponding difference equation

$$x(n) = -\sum_{k=1}^{p} a_k x(n-k) \quad + \sum_{k=0}^{q} b_k \, w(n-k) \quad (5.3)$$

where *w(n)* is the input sequence to the system and the observed data *x(n)* represents the output sequence. It is convenient to assume the input sequence w(n) as zero-mean white noise. The power spectrum of our observed data will then be

$$\Gamma_{xx}(f) = |H(f)|^2 \Gamma_{ww}(f) = \sigma_w^2 \frac{|B(f)|^2}{|A(f)|^2} \quad (5.4),$$

where σ_w^2 is the variance of *w(n)*. The model-based approach thus consists of two steps. First estimate the parameters *{a_k}* and *{b_k}* of the model. Then compute the power spectrum according to (5.4). [12]

A random process generated by a pole-zero transfer function is called an autoregressive-moving average (*ARMA*) process. If the parameters $\{b_k\}$ is of order zero (*q = 0* in (5.3)), The system is called an *AR* process. The *AR* model is widely used for two reasons. The *AR* model is suitable for representing spectra with narrow peaks (resonances). Moreover, the *AR* model results in very simple linear equations for the *AR* parameters.

The Yule-Walker method estimates the autocorrelation $r_{xx}(n)$ from the observed data and is used in the normal equations (5.5) to obtain the *AR*-model parameters. That is, the true correlation values $\gamma_{xx}(n)$ are replaced by the estimates $r_{xx}(n)$. The power spectrum is then calculated in (5.4).

$$\begin{bmatrix} \gamma_{xx}(0) & \gamma_{xx}(-1) & \cdots & \gamma_{xx}(-p) \\ \gamma_{xx}(1) & \gamma_{xx}(0) & \cdots & \gamma_{xx}(-p+1) \\ \vdots & \vdots & & \vdots \\ \gamma_{xx}(p) & \gamma_{xx}(p-1) & \cdots & \gamma_{xx}(0) \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} \sigma_w^2 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad (5.5)$$

The Pisarenko Harmonic Decomposition method is a method using eigenanalysis for estimating the spectrum. It is assumed that the spectrum consists of a number of sinusoids in white additive noise. In short, the method proceeds as follows. From the data, the autocorrelation matrix is formed[9]. The minimum eigenvalue is found and the corresponding eigenvector reveals the parameters of the *ARMA* model. The frequency of the sinusoids are determined by the roots of *A(z)* in 5.6.

$$A(z) = 1 + \sum_{m=1}^{2p} a_m z^{-m} \quad (5.6)$$

Xiao and Tadokoro [13] compared in 1994 the Pisarenko and the Constrained Yule-Walker (CYW) estimators. To their knowledge, such a simple and intuitive derivation as theirs of the CYW, had not yet appeared in the literature. The two frequency estimators used by Xiao/Tadokoro are derived as follows.

The problem statement is set to consider a single sinusoid in noise:

$$y(t) = A\sin(\omega t + \varphi) + e(t), \qquad t = 1,2,\ldots,N \quad (5.7)$$

where $\omega \in (0,\pi)$, the initial phase ϕ is uniformly distributed $[0,2\pi)$, *e(t)* is white gaussian noise sample sequence with zero mean and variance σ^2. *N* is the number of observed data.

The autocorrelation for *y(t)* at time *k* is

$$r_k = E\{y(t)y(t+k)\} = \frac{A^2}{2}\cos(k\omega) + \sigma^2\delta_k \quad (5.8)$$

An estimator for the frequency is derived if eq. (5.7) is assumed stationary during two time samples. Results from equation (5.8) will then reveal:

$$\begin{cases} r1 = \dfrac{A^2}{2}\cos(\omega) \\ r2 = \dfrac{A^2}{2}\cos(2\omega) = \dfrac{A^2}{2}(\cos^2(\omega) - 1) \end{cases} \quad (5.9)$$

which will result in the estimator

$$\omega = \arccos\left(\frac{r_2}{4r_1} + \frac{1}{2}\operatorname{sgn}(r_1)\sqrt{\frac{r_2^2}{4r_1^2} + 2}\right) \quad (5.10)$$

This is the result from the one-sinusoid model of the Pisarenko Harmonic Decomposition. The general result can be derived from (5.6).

If one to (5.9) also add the assumption of stationarity through a third time sample and uses the equations for r_3 as well, it will lead to a second frequency estimator

$$\omega = \arccos\left(\frac{r_1 + r_3}{2r_2}\right) \quad (5.11)$$

This estimator can be obtained by using the first equation for of the Yule-Walker equation set for a sinusoid in noise. Xiao/Tadokoro leaves it an open

[9] The autocorrelation matrix is formed in the same way as the matrix on the left in (5.5).

question whether or not the same derivation method can be extended to the multiple sinusoidal case.

5.2.12 PDA: LPC Analysis

Linear Predictive Coding, LPC, declares that a signal sample *x(n)*, is predictable from previous samples, except for an additive error signal, the LPC residual *e(n)*.

$$x(n)=a_1x(n-1)+a_2x(n-2)+a_kx(n-k)+e(n)$$

The filter coefficients a_i can be seen as a digital filter that can be calculated using the normal equations mentioned in 5.2.11. LPC analysis can according to [3] be seen as a method for short-time spectrum estimation and the residual signal will have a much flatter spectrum than the original signal *x(n)*. LPC analysis could therefore serve as an efficient preprocessing technique.

5.2.13 PDA: Adapting 2 pole notch filter with RLS or LMS

In this section an adaptive filtering approach is taken, where the pitch is calculated from the filter coefficients. Assuming the observed signal can be described as $x(n) = sin(2\pi(f/F_s)n)$, then the filter $A(q) = 1 - 2cos(2\pi(f/F_s))q^{-1} + q^{-2}$ will give $A(q)x(n) = 0$.[10] This is a notch filter. By observing the input signal *x(n)* and building an adaptive estimate of the desired notch filter coefficients, $A = [1,a_1, a_2]$, the filter coefficients will give an estimate of the angular frequency $2\pi f/F_s$ for *x(n)*. If the filter coefficient a_2 is close to one, then $arccos(-a_1/2)$ will be a good estimate for *f*. Otherwise the angular frequency can be estimated from the argument of one of the roots to *A(z)*, since ideally the roots should be $z = e^{(\pm j2\pi f/Fs)}$ [14].

The methods used in 5.2.11 are examples of off-line estimation of the *AR*-model parameters. That is when a whole batch of *N* data samples $Y(n) = [y(n),\ldots,y(n-N+1)]$ is available, and processed right away. Estimating the parameters in a recursive manner is called on-line estimation. In this case the estimate of the system parameters are updated every sample.

The *AR* system can be written as

$$y(n)+a_1y(n-1)+\cdots+a_Ny(n-N)=e(n) \qquad (5.12)$$

where *e(n)* is zero mean white noise and $\theta = \{a_1,\ldots,a_N\}$ are the unknown system parameters. Introducing $\varphi(n) = [-y(n-1), -y(n-2), \ldots, -y(n-N)]^T$, the system can be rewritten as

$$y(n)=\varphi^T(n)\theta+e(n) \qquad (5.13)$$

A one-step ahead predictor of the value is formed[11]

$$\hat{y}(n\,|\,n-1,\theta)=\varphi^T(n)\theta \qquad (5.14)$$

We're not really interested in the one-step ahead predicting, but what we use is the fact that the predictor algorithms will also deliver an estimate for the system parameters. The Least Mean Square (LMS)-algorithm that serves as a one-step ahead predictor and system estimator is given by

$$\begin{cases} \hat{y}_{LMS}(n\,|\,n-1)=\varphi^T(n)\hat{\theta}(n-1) \\ \hat{\theta}=\hat{\theta}(n-1)+\mu\varphi(n)(y(n)-\hat{y}_{LMS}(n\,|\,n-1)) \end{cases} \qquad (5.15)$$

where μ is the step size found empirically.

The stability of the LMS algorithm is dependent on the signal power of *y* (variance σ_y^2) which of course vary, this will make the selection of the step size unnecessarily small. By using a normalized step size (5.16) the algorithm is made insensitive to the signal power. This is the normalized LMS.

$$\mu(n)=\frac{\bar{\mu}}{c+\|y(n)\|^2} \qquad (5.16)$$

where *c* is a positive constant and for stability $0 < \bar{\mu} < 2$.

The recursive least squares (RLS) algorithm (5.17) is a method using a more sophisticated numerical optimisation procedure. It has faster convergence than the LMS and is less noise sensitive.

$$\begin{cases} \hat{y}_{RLS}(n\,|\,n-1)=\varphi^T(n)\hat{\theta}(n-1) \\ K(n)=\dfrac{P(n-1)\varphi(n)}{\lambda+\varphi^T(n)P(n-1)\varphi(n)} \\ P(n)=\dfrac{1}{\lambda}(P(n-1)-K(n)\varphi^T(n)P(n-1)) \\ \hat{\theta}=\hat{\theta}(n-1)+K(n)(y(n)-\hat{y}_{RLS}(n)) \end{cases} \qquad (5.17)$$

where λ is called the forgetting and set empirically. The step size μ in the LMS algorithm equates *1*-λ in RLS.

5.2.14 PDA: Autocorrelation

The autocorrelation function (*acf*) PDA's are popular and widely used. It is a short-term analysis method and the most intuitive one following that formula. The principle is simple and the theory is discussed in 3.2.1 in greater detail. The benefits of the autocorrelating PDA's are that they can be designed and implemented in a countless number of manners. They are easily modified for different measuring ranges, resolution and desired response time characteristics. Best of all is that the acf PDA

[10] The q shift operator behave as q x(n) = x(n+1) and q^{-1} x(n) = x(n-1).

[11] Given n-1 values of y and θ, the one-step ahead predictor computes a guess of the next value of y. It can be shown that this is the optimal one-step ahead predictor when e(n) is zero-mean.

does not require the presence of the fundamental harmonic.

The ordinary[12] acf PDA though has some shortcomings. A distinct formant structure is maintained in the acf, and notorious errors such as higher harmonic or subharmonic detection will occur. The conclusion that has been made is that acf PDA's need some sophisticated preprocessing to reduce the influence of the formants, especially from the first formant. Spectral flattening is done with methods involving non-linear or linear operations, e.g. through center clipping and filtering. (See Ch. 7) If the acf is calculated using discrete Fourier Transform, spectral flattening can be done in frequency domain.

5.2.15 PDA: AMDF

The AMDF function is described in 3.2.2. It is calculated as the agreement of the signal at a certain distance and could be seen as a cheap alternative to the acf. No multiplication's are needed[13]. There are other comparable "distance" comparing functions but the AMDF is the most used. The method can be realized in a similar fashion as the acf PDF, i.e. through a short-term analysis.

The method is phase-insensitive, since the harmonics are removed without regard to their phase. Unfortunately the function is sensitive to intensity variations and noise.

The function has sometimes been used as a refiner as to calculate the pitch in the vicinity of a crude estimate of the pitch.

5.2.16 PDA: Maximum-Likelihood

Maximum-likelihood (ML) pitch determination, in some literature called least-squares pitch determination, assumes nothing about the signal observed over *K* samples, except that it consists of a noise component *w(n)* and a periodic component *x(n)* (with a period less than *K*).

$$a(n) = x(n) + w(n) \qquad n = 0,...,K-1 \qquad (5.18)$$

The job for the ML algorithm is to mathematically reconstrunct *x(n)* and *w(n)*. This is done by finding an estimate $\hat{x}(n,p)$ that is most likely to represent the original wave *x(n)*. The estimate is found by maximising its energy as a function of the trial period p.

$$\sigma^2(p) = \frac{1}{K}\left[\sum_{n=0}^{K-1} a^2(n) - \sum_{n=0}^{K-1} \hat{x}^2(n,p)\right] \qquad (5.19)$$

The first term in (5.19) [3] is the energy of the signal within the frame and the second term is the energy of the periodic estimate. The problem is to minimize the variance of the noise σ^2, which is done when the energy of the periodic component, depending on the trial period *p*, becomes a maximum. For a given period p, the signal estimate is given by (5.20) [3]. P is the number of complete periods contained in the interval n = 0,…,K-1.

$$\hat{x}(n,p) = \begin{cases} \dfrac{1}{P+1}\displaystyle\sum_{k=0}^{P} a(n+kp), n = 0,\ldots,K-Pp-1 \\ \dfrac{1}{P}\displaystyle\sum_{k=0}^{P-1} a(n+kp), n = K-Pp,\ldots,p-1 \end{cases} \qquad (5.20)$$

5.2.17 PDA: On-Line LS-fit

As described in section 3.1, any periodic signal can, as presented in the theory of Fourier series, be represented by a signal consisting of many sinusoidal components.

Händel and Tichavsky [15] has designed an algorithm that uses an adaptive comb filter based on discounted least squares (LS) identification of a harmonic signal combined with estimation of frequencies from phase differences.[14]

The signal plus noise signal is parameterized in a state space model. To the LS criterion, which is set up, a forgetting factor $\lambda < 1$ is introduced. This is according to Händel/Tichavsky [15] a frequently used modification in on-line applications in order to track time-varying parameters.

5.2.18 PDA: Cooper and Ng

D. Cooper and K.C. Ng introduced an algorithm in 1994 [1]. The input signal is analysed and separated into segments, where each segment starts and ends at two consecutive positive zero crossings (as defined in 5.2.3).

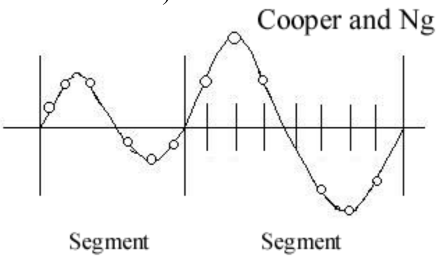


figure 5g) Two segments and characteristic values

[12] By ordinary Rabiner in 1977 meant that the incoming signal is more or less unprocessed except for some lowpass filtering.

[13] In the target system this does not reduce the complexity. One multiplication or one subtraction can be done in one processor cycle.

[14] This approach has revealed accurate performance and low-cost update schemes according to [8].

As depicted in figure 5g, each of these segments are divided into eight subsegments. The first three and the last three amplitude values of these subsegments form the landmarks for the current segment. The segment with the largest segment is then compared to all the others, calculating a similarity ratio using a normalized distance

$$\begin{cases} similarity\_ratio = \dfrac{a.b}{(a.a)+(b.b)-(a.b)} \\ a.b = \sum_{i=1}^{6} a_i b_i \end{cases} \qquad (5.21)$$

The two segments to be selected as similar, must fulfil the following:

- The difference in length of the two segments must be less than a defined threshold.
- The similarity-ratio should be high.[15]
- The similar segment should be as near to the largest segment as possible.

The fundamental period is finally calculated as the distance between the two similar segments.

5.2.19 PDA: The Reduced ACF

An approach similar to the Cooper and Ng method in 5.2.18 has been developed[16]. The advantages of that method, such as being cheap in computations in relation to the results, made the approach look promising. The main idea from Cooper and Ng, is to use the signal's zero crossings as cues for a calculation, where only a few signal characteristics are used when calculating the correlation.

As in 5.2.18, one segment is extracted between two positive zero crossings. The characteristical values are chosen as the segments maximum and minimum values and the segment length. The intersegment similarities are calculated according to equation (5.21).

A reference segment is picked as the maximum valued segment in the eight most recent segments. This segment is correlated to the others using the similarity calculation. A segment with a similarity exceeding a certain threshold, that is most adjacent to the reference segment, will be chosen for the distance corresponding to the fundamental period.

The accuracy is improved if two reference segments are used, suggested is the maximum segment and also the segment having the biggest minimum. A fundamental period estimate is calculated from the two reference segments, and a resulting period estimate is valid if the two estimates don't differ more than a certain value.

This PDA is proposed to use a recursive setting of the cut-off frequency for a LP filter and a recursively set center-clip and compression value.

The development and theory of the algorithm is further described in a later section (9.3).

5.3 Spectral Domain PDA

In this section the algorithms that operate in the frequency domain are discussed.

5.3.1 Common features

The frequency domain algorithms are in general dependent of a transformation from time to spectral domain, and therefore fairly demanding in terms of computational quantity. The FFT can of course be implemented quite effectively in today's digital signal processors.

In contrast to time domain PDA's, downsampling the signal does not decrease the accuracy of frequency domain PDA's.

Intuitively, pitch tracking should be done in the frequency domain, since it gives superior control of where the energy of the formants are situated. Not only the fundamental is then of interest but one can explore the relationship of harmonic spectral peaks.

5.3.2 Common drawbacks

A problem with the spectral PDA is that a simple study of the spectrum, e.g. the maximum resulting from the FFT, is not enough for determining the fundamental period. A simple spectrum peak picker will be erroneous since formants in musical signals will increase the magnitude of higher harmonics. Extra processing such as transformation to Cepstrum or some bright harmonic analysis of the spectral peak distribution is needed.

Another obstacle is that a normal DFT separates the bandwidth into equally spaced frequency bins. However, as discussed in 2.7 the relationship between pitch perception and frequency is logarithmic. This could be solved by using a constant-Q transform (5.3.7), which on the other hand lead to a considerable amount of calculations for the lower frequency parts.

5.3.3 PDA: FFT, Maximum of FFT, Division method

Transforming data to spectral domain is done in a short-term analysis manner. When using the DFT this way, it is usually referred to as the STDFT *(Short Time Discrete Fourier Transform).* Of course, an efficient FFT algorithm is used when implementing the DFT.

[15] Cooper and Ng set this threshold to 0.75.

[16] The development of this algorithm was done by Leopold Roos and Stefan Uppgård during this project. The name 'Reduced ACF' asserts that the proposed method is cheaper in computations than the ordinary ACF.

As discussed in 5.3.2, the most primitive form of spectral analysis is simply picking the maximum spectral peak, hoping this is the fundamental. When the fundamental harmonic is strong, and simultaneously other harmonics are weak, this should produce nice results. The spectral peak picker frankly traverses the data and remembers the position of the maximum peak. This method can be called the *maximum of FFT method* [16].

A simple way to improve the method mentioned above is the *division method* [16]. After finding the maximum peak at position *F*, it is investigated if there are peaks at *F/n*, where *n* is an integer. It is assumed that the largest *n*, still having a peak at its position, corresponds to the fundamental frequency. How to determine if the analysed peak is a “valid” peak, is not discussed in [16]. A simple way of doing this is to set a threshold magnitude that the peak must exceed.

5.3.4 PDA: FFT, Distance of spectral peaks

The methods in 5.3.3 will fail if the fundamental harmonic is missing. One way out of this is to explore the relationship between the harmonics of the signal.

It is sufficient to measure the distance between two adjacent spectral peaks to know the fundamental frequency. The method has since long been used when manually analysing spectrogram plots. The crude estimate of F_0, as derived from two adjacent peaks, can be refined by analysing the distance to higher (or lower) harmonic peaks.

This approach could be expected to have been investigated even more than what was found in the information search underlying this project. However, it seems to be an area that is still active for research, since many articles published (such as [17]) originate from recent years. Nevertheless, some of the following sections do indeed explore the spectral harmonic relationship.

5.3.5 PDA: FFT, Piszczalski and Galler

The Piszczalski and Galler (P & G) method [18] (1979) for predicting pitch from “component frequency ratios” is one example of algorithms evolved when the performance of digital signal processing was improved, and it was clear that processing digital data in frequency domain had some potential.

The method considers each peak in the spectrum a potential candidate for being connected to one or more harmonic numbers, ranging to a specified maximum[17]. By examining the magnitudes and relationship of the spectral peaks, weighting factors are calculated which reflect the likelihood of each peak being associated with each possible harmonic number.

The weighting factors, initially set to zero, are updated through a procedure which is applied to all peak pairs formed from the peaks of the spectrum. For every pair with frequencies f_1 and f_2 ($f_1 > f_2$), the frequency ratio f_1 / f_2 is computed. The ratio is then compared to “harmonic-number” ratios of the form i / j (integers $i > j$). If the absolute difference between the ratios are less than a threshold, a match is declared.

If a match has been declared the weighting factors of connecting f_1 to the harmonic number *i*, as well as the factor for connecting f_2 to *j*, is incremented by

$$[a_{\min} + 0.1 \cdot a_{\max}][1 - 0.03(i + j - 3)] \qquad (5.22)$$

where a_{min} and a_{max} are the respective smaller and larger of the magnitudes at f_1 and f_2. The derivation of (5.22) is explained and discussed in [17] and [18].

The highest weighting factor calculated determines a particular peak and a corresponding harmonic number, from which the fundamental frequency can be calculated.

Dorken and Nawab revisited the method in 1994 [17], introducing a technique called spectral conditioning for restraining interfering components of the signal that would deteriorate the result of the P&G process.

The method follows a fairly complicated procedure, which for this project would not be possible to implement, but likewise provide some interesting theory for how suppression of interference could be taken care of.

The technique operates on the spectrum from a constant Q transform (5.3.7) and according to [17], “essentially performs circular shifts on the various short-time spectra in order to convert the log-frequency excursions of the harmonic signal with greatest net energy into constant-frequency paths.”

This is done by an iterative eigen analysis for estimating the spectrum. (The Pisarenko method described in 5.2.11 is an algorithm originating from eigen analysis).

The first principal component $\varphi^{(m)}(f[k])$ can be calculated using the power method [19], which is an iterative method for finding eigenvectors of a square matrix. In this case the square matrix should consist of the autocorrelated input signal and look like the left matrix in equation (5.5). The eigenvector corresponding to the largest eigenvalue, is called the first principal eigenvector [12].

Then, a correlation measure for a particular frequency-shift k_0 is calculated using

[17] Fixed at 12 by Piszczalski & Galler.

$$r^{(m)}(nLT)=\sum_{k=0}^{N-1}\varphi^{(m)}(f[k])Q_x(nLT,\tilde{f}[k-k_0]) \quad (5.23)$$

where $Q_x(nLT,f[k])$ is the input spectrum at time nLT and L is a decimation factor.

The value k_0, where the correlation is maximized, and the corresponding output $r_{max}^{(m)}(nLT)$, are tested in a convergence test [17]. If the test fails, the power method is called for another iteration. If successful, $\varphi^{(m)}(f[k])$ and k_0 are the outputs of the iterative procedure.

The first principal component is run through the P & G method, where the resulting frequency is shifted k_0, and then divided by the resulting harmonic number.

5.3.6 PDA: Spectral Compression and Harmonic summation

Schroeder introduced in 1968, an algorithm demanding many computations, but still, one of the most reliable PDA's available. The technique used is called spectral compression and it can derive an estimate for F_0 from higher harmonics without demanding the corresponding harmonic numbers. Schroeder explores the fact that fundamental frequency should be the greatest common divisor of the frequencies of the individual harmonics.

Peaks present in the spectrum are found using a peak-picker traversing the data. [3] A histogram is built up having the frequencies of all the peaks at its entries. The frequencies are then divided by two and added to the histogram. The same procedure is repeated with compression factors *3, 4 etc.*[18] This will ideally result in a maximum of the histogram at the fundamental frequency. Schroeder also proposes an approach where the entries of the histogram are weighed by the magnitude of resp. peak.

Also introduced by Schroeder in 1968 were the Harmonic Product PDA and the Harmonic Summation PDA [3]. The later sum up the magnitude values of the spectrum at equidistant frequencies. The frequency that maximizes the sum is then F_0. The magnitude values are multiplied with a weighing function diminishing towards higher frequencies, so that the lower frequencies will have greater influence of the sum. Also, to reduce the influence of background noise, the spectra is set to zero except for at the peaks and their surrounding.

The Harmonic Product PDA works in a similar way, but the spectrum is assumed logarithmic.

5.3.7 PDA: Constant Q transform

The theory and the properties of the constant Q transform has been discussed in 3.3.3. It was concluded that it has properties appropriate for processing musical signals, since it can be executed so that the geometrically spaced bins correspond exactly to a musical semitones. The notes on the western musical scale are as a matter of a fact spaced geometrically. In addition, the tracking of frequency fluctuations at higher frequencies requires wider bandwidth.

Pitch tracking algorithms using the constant Q transform has been proposed by Judith C. Brown and Miller S. Puckette. In [20] they introduced a first algorithm that in following articles since, e.g. has been developed to be implemented by FFT [21].

A high resolution PDA based on the phase changes of the Fourier transform was introduced in 1994 [22]. The ambition was to escape the demand in earlier algorithms for the analysed music to stick to the equally tempered scale[19]. Introducing this algorithm, it would be possible to analyse such phenomena as glissando, vibrato or instruments being tuned to other scales than the equal tempered. For these cases, the resolution had to be improved.

The Hanning-windowed Fourier transform evaluated for a window beginning on sample n_0, for an input signal $x(n)$ can be denoted

$$X^H[k,n_0]=\sum_{n=0}^{N-1}x[n+n_0]w[n]e^{-j2\pi kn/N} \quad (5.24)$$

with

$$\begin{aligned} w[n] &= \frac{1}{2}-\frac{1}{2}\cos(2\pi n/N)= \\ &= \frac{1}{2}[1-(\frac{1}{2})e^{j2\pi n/N}-(\frac{1}{2})e^{-j2\pi n/N}] \end{aligned} \quad (5.25)$$

Eq. (5.25) substituted into (5.24) will lead to

$$X^H[k,n_0]=\frac{1}{2}\left\{X[k]-\frac{1}{2}X[k+1]-\frac{1}{2}X[k-1]\right\} \quad (5.26)$$

An approximation for the DFT after one sample is

$$\begin{aligned} X^H[k,n_0+1] &= \frac{1}{2}e^{j2\pi k/N}\cdot \\ &\cdot\left\{X[k]-\frac{1}{2}e^{j2\pi/N}X[k+1]-\frac{1}{2}e^{-j2\pi/N}X[k-1]\right\} \end{aligned} \quad (5.27)$$

The digital angular frequency in radians per sample for bin k is the phase difference for a time advance of one sample.

$$\omega(k,n_0)=\phi(k,n_0+1)-\phi(k,n_0) \quad (5.28)$$

[18] When to end the procedure is not defined, but a limit can be found empirically.

[19] Here the smallest frequency difference between notes is approximately 6 %.

where

$$\begin{cases} \phi(k,n_0+1) = \arctan\left\{\dfrac{\mathrm{Im}(X^H[k,n_0+1])}{\mathrm{Re}(X^H[k,n_0+1])}\right\} \\ \phi(k,n_0) = \arctan\left\{\dfrac{\mathrm{Im}(X^H[k,n_0])}{\mathrm{Re}(X^H[k,n_0])}\right\} \end{cases} \qquad (5.29)$$

The method is hence carried out by getting an initial frequency estimate, most simply by taking the maximum peak of the constant Q spectrum. A calculation is made to determine the corresponding bin number for the FFT[20]. The values already calculated in the FFT, are then used in (5.26) and (5.27), finally giving the resulting high resolution fundamental frequency in (5.28).

The method is equivalent to a phase vocoder (5.3.8) using the FFT method, with a hop size of one sample. The advantage here is the approximation in eq. (5.27), which avoids a second FFT.

Worth noting is that Brown and Puckette claim this method to be more accurate than the technique using a quadratic fit of the magnitude of the selected estimated fundamental bin and its neighbours. (Compare section 7.5)

5.3.8 PDA: Phase Vocoder

The phase vocoder was first introduced by Flanagan [23] as a time-domain technique, but the modern fast Fourier transform based implementation was displayed by Portnoff [24]. It has been used as an analysis/resynthesis tool in applications such as altering time scale or pitch scaling sound. The later application was recently implemented in real time, using the phase vocoder by a group at KTH with good results [25].

The analysis part of the phase vocoder can be used for pitch tracking. It is usually based on a DFT for calculating a first estimate telling in which frequency bin the fundamental is situated.

The phase propagation between two adjacent frames, overlapping H samples can be denoted

$$\varphi(M,k) = \arg X_w[M+H,k] - \arg X_w[M,k] \qquad (5.30)$$

where $X_w[M,k]$ is the short-time Fourier transform of the N long input signal at time-instant M. The factor $1/H$ is sometimes called the overlapping factor β.

The phase offset has to be wrapped, i.e. mapped into an interval from $-\pi$ to π. The deviation from the bin frequency can then be calculated

$$\Delta\omega_{est}(H,N,M) = \frac{\{\varphi(M,k)\}_{Mod2\pi} - \pi}{H} \qquad (5.31)$$

and the estimated fundamental frequency

$$\hat{F}_0 = \frac{k}{N} + \frac{\Delta\omega_{est}(H,N,M)}{2\pi} \qquad (5.32)$$

The method can be seen as an interpolation of two adjacent frames of the DFT calculation.

5.3.9 PDA: Auto- and Cross-Correlation techniques in spectral domain

Following the introduction of the constant Q transform, Judith C. Brown also presented the "pattern recognition method" [5]. This method has got it's inspiration from methods such as P&G and Schroeder (5.3.5 and 5.3.6).

The idea is to cross-correlate the spectrum with a pattern having 1's at the appropriate positions, at harmonic distances. The number of components in the pattern are matched to what is "ideal" for that instrument and should be seen as an adjustable parameter. In other words, the number of components should match the average number of non-zero Fourier components for a particular instrument.

As the spectrum has logarithmic spaced frequencies, the distance between two adjacent harmonics is not equal to the fundamental frequency. For example, the spacing between the fundamental and the second harmonic is *log(2)*, between the second and third components *log(3/2)*. That way the correlating pattern can be constant.

The result of the cross-correlation should result in a dominant peak at the fundamental frequency.

A recent article [26] presents an algorithm where the possibility of doing an auto-correlation of the power spectrum is mentioned. The algorithm implemented there however, makes a logarithmic power spectrum of a standard autocorrelation, which then again in spectral domain, is autocorrelated.

[20] This was in the simulation calculated as $bin_{dft} = (f_{min} / freq.resolution_{dft})\ 2^{(bincq/bins\_octave)}$, where f_{min} was the lowest bin's centerfrequency and bins_octave the number of bins per octave used in the constant Q transform.

5.3.10 PDA: Cepstrum

The theory of Cepstral processing has been presented in 3.3.4. PDA's employing this technique have mostly done it analysing speech signals.

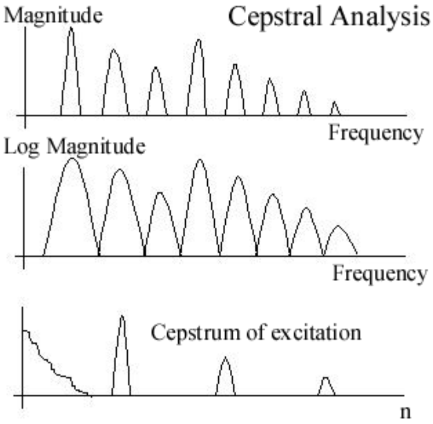


figure 5.h) The Cepstrum

Detecting the significant peak at $\hat{T}_0$ of the Cepstrum forms the principle idea of a Cepstrum PDA. Basically, the input signal is divided into windowed frames, with for example a size of 512 samples and a Hann window. The frame is Fourier transformed, logarithmized and then inverse Fourier transformed back to time domain. Once there in Cepstrum, the significant peak at cuefrency $\hat{T}_0$ is determined with a simple peak picking basic extractor, traversing the operating interval for the PDA.

It has been shown that Cepstrum algorithms are sensitive to noise [27]. A method have been employed in [27] where the MUSIC algorithm[21] is used for estimating the background noise characteristics, successfully improving the performance of the PDA.

5.3.11 PDA: Wavelets

(The basic theory of wavelets is discussed in 3.3.5.)

Tristan Jehan [28] implemented[22] an algorithm using Wavelet Transform, considering its nice non-linear properties. The WT uses short windows at high frequencies and long windows at low frequencies. This technique, using logarithmically spaced frequencies can be compared to the constant Q transform (5.3.7). He adopted a speech orientated algorithm, but showed that after some improvements, it gave nice results also for musical sounds.

The technique using the wavelet transform for pitch detection was first introduced by Kadambe/Boudreaux-Bartels [29]. They adopted results from Mallat [30], where it was showed that when analysing images, the use of wavelet functions with derivative characteristics produced maxima in the wavelet transform across many coincident scales along sharp edges.

The same maximum should occur, at the GCI[23] for a speech signal, when filtered through a derivative function [28]. That way the time between each maximum should correspond to T_0.

A filter function can be defined

$$\rho(t) = \psi_{k_a}(t) * \varphi_{k_b}(t) \tag{5.32}$$

where

$$\begin{aligned} \psi(t) &= 2^{k/2}\psi(2^k t) \\ \varphi(t) &= 2^{k/2}\varphi(2^k t) \end{aligned} \tag{5.33}$$

The righthand side in 5.33 is a lowpass function and the conjugate mirror filter of ψ(t), which is a highpass wavelet function. If the PDA's measuring range is between f_l and f_2, the final filtering function constructed should have a similar bandwidth. Therefore the lowpass scaling function is

$$2^{k_a} = \frac{F_s}{f_1} \tag{5.34}$$

and the highpass wavelet function

$$2^{k_b} = \frac{F_s}{f_2} \tag{5.35}$$

The filtering function in 5.32 will result in pseudo sinusoid, where the distance between two adjacent peaks is $\hat{T}_0$.

Choosing the *mother wavelet* is very important since it defines the behaviour of the wavelet transform. For voiced speech it is often modelled as a filtered impulse train, where the period between each pulse represents the pitch period [31]. The mother wavelet used by Tristan was a derivating function, a Daubechies filter[24].

[21] The Multiple Signal Classification (MUSIC) algorithm estimates the noise subspace using eigen analysis.

[22] Though only implemented in computer simulations using MATLAB®.

[23] GCI - Glottal Closure Instant

[24] Suggested by Ingrid Daubechies, filters that under certain conditions provide perfect reconstruction.

5.3.12 PDA: Comb filters

Comb filters have been used for measurements such as pitch strength ("pitchiness") [32] as well as surpressing harmonics, but there are also pitch determination algorithms applying the technique.

The principal in a method proposed by Miwa, Tadokoro and Saito [33] is to eliminate the pitch and its harmonic frequencies. This is done with comb filtering technique (3.4.3). The pitch is estimated by detecting zero outputs of cascaded comb-filters.

The algorithm was designed to detect the pitch in three octaves, from tone C_3 to B_5.

The transfer function of the comb filter can be written

$$H_{q,p}(z_p) = 1 - z_p^{-q}$$
$$p = 1,2,\mathrm{K},12 \qquad , \quad q = 1,2,4,8 \tag{5.36}$$

where *p* represents a semitone in one octave and *q* is the order of the comb filter. The filter in (5.36) can be implemented as $y_p(n) = x_p(n) - x_p(n-q)$.

It's necessary that each tone *p* has its own sampling frequency f_{sp}, to get the frequency response plotted in figure 5.i.b. That way, three different filters, having orders $q = 2,4$ and 8, can put out tone *p* in the three octaves 3, 4 and 5.

If the output of $H_{8,p}(z_p)$ is zero, the tone played is *p*. Furthermore if $H_{4,p}(z_p)$ is zero, tone *p* must be in octave 4 or 5. Finally if $H_{2,p}(z_p)$ is zero, it has been deduced that the tone number must be *p* in octave 5.

The comb filters are put in cascade as described in figure 5.i.a, but since each of the filters have different sampling frequencies through different unit delays z_p^{-1}, the system is oversampled using[25] (figure 5.i.c)

$$\begin{cases} n_p \cong 8 \cdot \dfrac{f_s}{f_{s_p}} \\ H_{8,p}(z_p) \cong 1 - z^{n_p} \end{cases} \tag{5.37}$$

Consequently, the tone number p is found as the zero output from output y_1 to y_{12}. Then the filters $H_{8,p}$ are next replaced with $H_{4,p}$ respectively $H_{2,p}$ to decide which octave the tone is in.

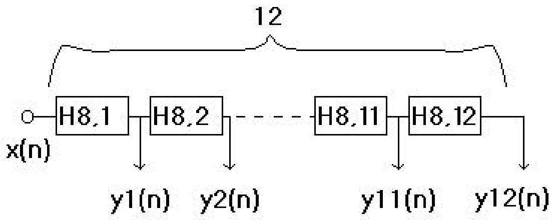


figure 5.i.a) Cascade connection of comb filter $H_{8,p}(z_p)$

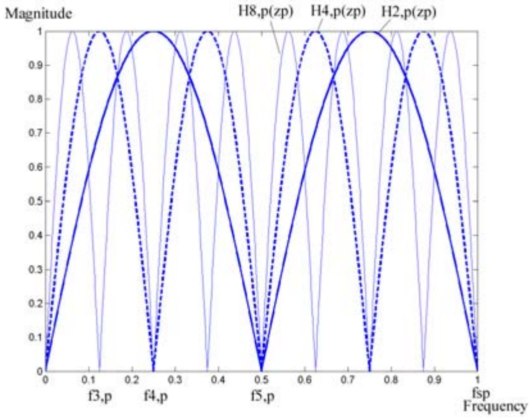


figure 5.i.b) Frequency response of comb filter $H_{q,p}(z_p)$ $(q = 2,4,8)$

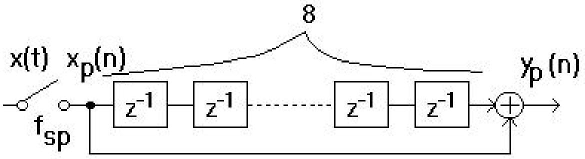


figure 5.i.c) Implementation of $H_{8,p}(z_p) = 1 - z_p^{-8}$

There have also been techniques using an inharmonic comb filter, where the center frequencies are not equidistant, but spaced according to a function. This could offer advantages when studying sources having inharmonic components (2.5). However, a relation between the partial frequencies, or at least some hypothesis for the relation, is needed in advance.

Galembo and Askenfelt [34] introduced a technique which was applied to the acoustical piano.

5.3.13 PDA: Generalized Spectrum

Black and Donohue propose a PDA based on the Generalized Spectrum [35].

The Generalized spectrum is defined as

$$GS(f_1, f_2) = E[X(f_1) \cdot X(f_2)] \tag{5.38}$$

where *X(f)* is the discrete Fourier transform (length *M*) of a discrete input signal *x(n)*.

The main diagonal of GS have real values along the main diagonal, corresponding to the PSD (Power Spectral Density) estimate of *x(n)*. The other elements are complex valued, and reflects the

[25] According to [28], where f_s = 54054 Hz was used, the maximum error of the approximate f_{sp} was 0,14 %.

coherence between different frequency components in the DFT.

A signal with pitch, could be called a cyclostationary process, which in GS introduce correlation and non-zero values in the off-main diagonal regions.

The collapsed average transforms the GS into one-dimensional, which makes it easy to identify diagonals with significant correlation values. The collapsed average can be denoted

$$CA(k) = \frac{diag(|G|,k)}{diag(|G|,l)} \quad \text{for k = 1,2,…,M} \qquad (5.39)$$

where *diag(G,k)* represents a sum along the k_{th} diagonal of *G*.

The pitch is extracted as the maximum value of CA, which hopefully corresponds to the pitch frequency, or in some cases a multiple of the pitch. Therefore Black/Donohue prefer to take the FFT of CA for a more consistent pitch period estimation.

$$\hat{T}_0 = \frac{M}{f_S} m_{\max} \qquad (5.40)$$

where m_{max} it the index of the maximum value of the FFT of CA. *M* is as mentioned above the number of elements in *X(f)*.

5.4 Combined Time and Spectral Domain PDA

Algorithms that clearly operate in both temporal and spectral domain are described herein after.

5.4.1 PDA: Rough FFT and LS-fitting

An algorithm has been evaluated that uses a very rough FFT, 128 points, to get a crude estimate of in which frequency bin the signal energy is situated.

That frequency estimate is then used to initiate a fit of a sinusoidal-curve to the input signal. The fit is done according to a IEEE standardized iteration[26], where the frequency resolution can be arbitrarily increased through each iteration.

5.4.2 PDA: Loose-Harmonic Matching

Quiros and Enriquez proposed a PDA, which they claimed could correctly estimate the fundamental frequency of practically any musical sound [36]. An estimate of the short-term spectrum $X_w(\omega)$ for the windowed sequence $x_w(n)$ is set up as

$$\hat{X}_w(\omega) = \sum_k A_k |W(\omega - 2\pi k / P)| \qquad (5.41)$$

given a candidate pitch *P* and the window *W(ω)*. The coefficient A_k is given by

$$A_k = \frac{\int_{-\pi}^{\pi} |X_w(\omega)| |W(\omega - 2\pi k / P)| d\omega}{\int_{-\pi}^{\pi} |W(\omega - 2\pi k / P)|^2 d\omega} \qquad (5.42)$$

For this approximation, the error is calculated

$$\varepsilon = \frac{1}{2\pi} \sum_k \int_{-\pi}^{\pi} \left[|X_w(\omega)| - \hat{X}_w(\omega - \omega_k) \right]^2 d\omega \qquad (5.43)$$

where

$$\omega_k = \frac{\int_a^b \omega |X_w(\omega)| d\omega}{\int_a^b |X_w(\omega)| d\omega} \qquad (5.44)$$

is a recentering value, that handles the problem of the harmonic *k* not being at exactly its harmonic position *2πk / P*.

Expression (5.43) is evaluated for every *P* under consideration and will provide a estimate for the pitch when it's minimized. The estimate for *P* was in [36] provided by an ordinary ACF calculation

$$r_x(m) = \sum_n x_w(n) x_w(n - m) \qquad (5.45)$$

The value of *m* where the first peak is found (*m = 0* not included) will serve as the first estimate for *P* and (5.43) is evaluated at the values *P,2P,3P,…,P/2,P/3…*

5.5 Other PDA

Algorithms that were not of the kind that they could be categorized as being completely of temporal or spectral kind, but using a somewhat different approach have been put here.

5.5.1 PDA: Neural Network

Neural Network PDA's can process data that are either temporal, spectral or both. In [37], the technique was said to have a promising future in speech analysis, since in such architectures, functions such as pitch detection, formant estimation, etc. can be implemented in parallel. Of course, the same would then also be valid for musical analysis.

Common for neural networks is that they extract characteristics of a signal, compare these to what is stored in the network and makes a classification.

The neural network has to be trained by training data, so that it learns how to distinguish and classify the information.

[26] The IEEE Standard (IEEE-STD-1057).

5.5.2 PDA: Heterodyne filtering

If the fundamental frequency is known, the heterodyne filtering technique can be used. It will serve as an indication for the deviation of the measured tone to a reference frequency. Electronic tuners uses heterodyne filtering.

Assume the measured signal *x(t)* to be a sinusoid of frequency ω. Multiplying the signal by $e^{j\omega rt}$, where ω_r is the reference frequency, results in

$$\begin{aligned} y(t) &= \sin(\omega t+\varphi)\cdot \\ &\left[\cos(\omega_r t+\varphi_r)+j\sin(\omega_r t+\varphi_r)\right] \end{aligned} \qquad (5.46)$$

Here, *y(t)* will contain the difference frequencies $\omega+\omega_r$ and $\omega-\omega_r$ from well known trigonometric formulas. The sum frequencies are removed by filtering and the derivative of the phase is then

$$\frac{d\angle y(t)}{dt}=\omega-\omega_r \qquad (5.47)$$

The slope of phase thus in fact shows the deviation of the input signal's frequency to the reference frequency. Like in 5.3.8, the phase offset has to be wrapped, i.e. mapped into an interval from -π to π.

The same method of studying phase can be applied to any bandpass filter.

6. Results, Simulations and Evaluations

In this section, some of the results from the many simulations that have been performed, are presented. Often the results are presented along with conclusions that have been drawn.

Not all of the algorithms discussed in the theory section have been simulated. The algorithms that have been simulated, either had potential for being implemented in the target system, or was simulated for improving the understanding of pitch algorithms in general.

The results have, in the cases where it was appropriate, been compared to the results from other PDA's. In other cases, the results are used for presenting benefits and drawbacks of the algorithm.

The algorithms have been tested on sound recordings that are listed in resources, at the end of the document. The recordings have been chosen as to represent signals form a variety of different musical instruments.

The simulations that have been made, have had the purpose of determining if the PDA is suitable in a musical context (compare section 9.1). It can therefore be easy to misinterpret the results. An algorithm that here is assigned a bad grade, can for a different purpose perform well.

The study has been done from an implementation point of view. Therefore the factor complexity has been important.

It is hard to compare algorithms in a fair way. Hess and Rabiner [3] have tried. Describing the results in a statistical manner is desirable, but in this project hardly meaningful. The sound recordings used for evaluation, have not been compared to correct pitch information. The correct pitch information could be extracted with an off-line method. This is left for future studies.

Hence, the evaluation has been made by listening to an oscillator whose pitch have set by the PDA. The original signal and the resynthesized oscillator signal have been compared by listening simultaneously in two different sound channels (left and right speakers). This is consistent with the fact that pitch is a psychological term.

6.1 Error Analysis

In the analysis of the PDA's, the following aspects are discussed:

Response time, accuracy, resolution and complexity.

Response time is the time from when a tone's attack is initialized to when the PDA delivers a first fundamental frequency estimate. Accuracy is a measure for how trustworthy the result is. Resolution describes the factors deciding which frequency resolution the pitch estimate has. The complexity of the algorithm has been a leading factor during the project. Complexity involves memory requirements, arithmetical operations needed and conditions.

6.2 Results Time Domain PDA

Results from algorithms being referred to being of time domain character is presented here.

6.2.1 Results: Threshold Crossing

(Algorithms 5.2.3 - 5.2.4)

Algorithms using threshold crossing for basic extraction, all have in common the fact that they need a "clean" signal for good performance. Of course, when a large amount of preprocessing is done, a threshold crossing extractor can for some applications be just about the only thing needed.

Figure 6.a shows the result of the zero crossing algorithm, used on a tone from the saxophone recording [R3]. The algorithm is seriously sensitive to any harmonic, causing additional polarity changes. It is obvious that moving the threshold from zero to a non-zero level, would improve the performance. In this case, a negative threshold would be useful. However, moving the level to a positive value would rather worsen the result.

Adding a second threshold, putting one at positive and one at negative level, definitely improves the result (figure 6.b).

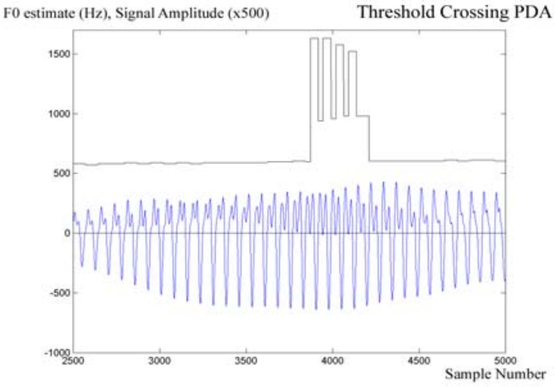


figure 6.a) Zero crossing results

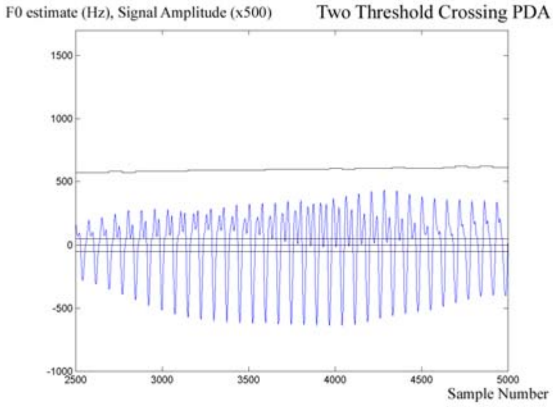


figure 6.b) Two threshold crossing results

Summary:
Response time: Good. Fast, one signal period, if no postprocessing added.
Accuracy: Poor. Very sensitive to the non-stationary feature of the signal.
Resolution: Good. Period resolution is $1/F_S$, and improved by interpolation.
Complexity: Very Good. Very low, is implemented very facile.

6.2.2 Results: Envelope Following PDA

(Algorithms 5.2.5 - 5.2.6)

Results from the envelope following PDA applied to the same saxophone tone as in preceding section results in an envelope plotted in figure 6.c. The improvement of highpass filtering the signal followed by a second envelope follower is obvious. The number of erroneous peaks, are reduced in the difficult interval from sample 3500 to 4500. Yet another iteration of highpass filtering + envelope follower should remove the one error remaining on the positive envelope.

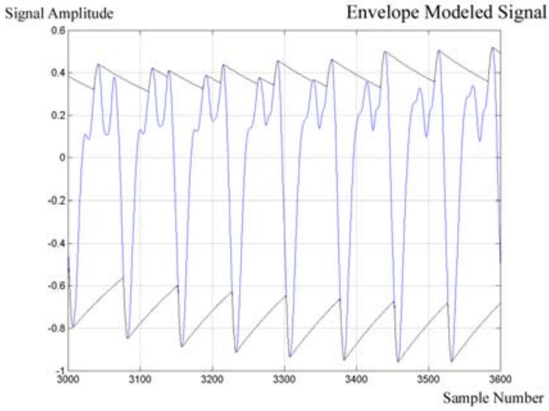


figure 6.c) Envelope following results

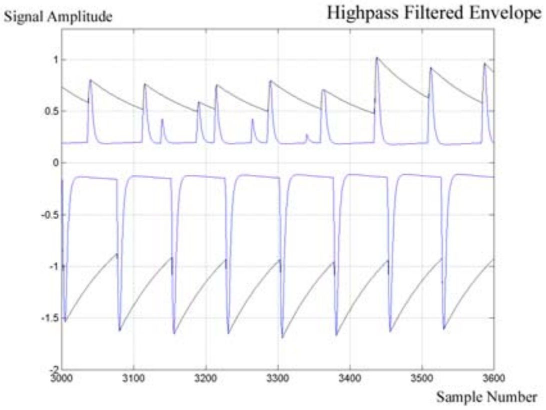


figure 6.d) Highpass filtered envelope

For more complex signals, such as the acoustic piano sound [R4], even a second highpass filtering and envelope won't improve the result. Figure 6.d. shows the same tone as analysed with the reduced ACF PDA (6.2.9). Only in the vicinity of sample 1,1 x 10^3 the pitch estimate is correct. (The true pitch in this segment is about 116 Hz, B^b2). When the signal is more complex and as the overtone content increase, it seems like the result is deteriorated.

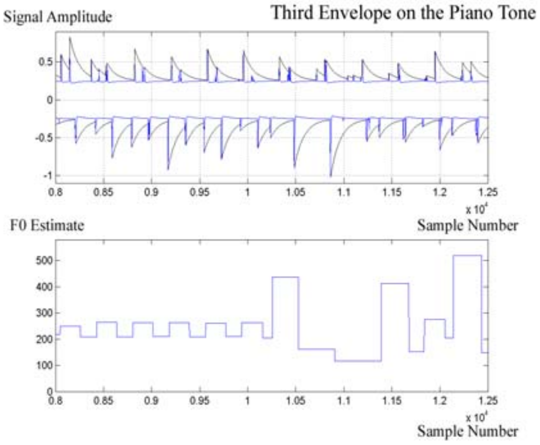


figure 6.e) Cascaded highpass filtered envelopes

Center clipping and compression is not considered to improve the performance, though a 'tighter' lowpass filtering probably would. Here however, different from the reduced ACF PDA, a source for a recursive setting of the lowpass cut-off frequency is not obvious and has not been tried out.

One problem in the simulations have been the issue of setting an appropriate value for the envelope decay constant.

A second problem has been to decide if the results from the positive or the negative envelope is most valid. The approach by Engdegård [10], selecting the result having the least variance in the last estimates have been tried. It has however during the simulations been hard to get good results from this measure.

Summary:

Response time: Good, Fast, one signal period, if no postprocessing added.

Accuracy: Ok. Most problems occur during the attack of tones, but also sudden unexpected pitch jumps as at sample 3200 in figure 6.d. for the positive envelope.

Resolution: Good. Period can be measured either from the zero crossing period or the peak positions of the envelope. Resolution is $1/F_s$, but can be improved by interpolation.

Complexity: Very Good, Low, one envelope follower and first-order highpass filtering is cheap. Adding up the order of envelopes increases the complexity. Variance calculation of past estimates adds additional calculations.

6.2.3 Results: Peak Detecting PDA

(Algorithms 5.2.7 - 5.2.8)

The algorithm by Reddy (5.2.7), has not been simulated in detail. Hess [3] stated that the algorithm was not suitable for real time implementation, since the postprocessing correction routine works on a longer signal segment. But, it is though enlightening to study it since it was one of the first in the peak detecting area.

The algorithm by Gold & Rabiner (5.2.8) gives an instantaneous response and the fundamental does not need to be present. The result is more stable than the envelope follower algorithm. In figure 6.f it is shown that the response is slower, but the estimate makes no sudden jumps. The bad resolution is from the fact that no interpolation has been performed.

The algorithm will for difficult tones like the piano tone analyzed in figure 6.e, have severe problems (figure 6.g). The estimate is stable, but wrong! The true pitch should be at about 116 Hz, one fourth of the result here!

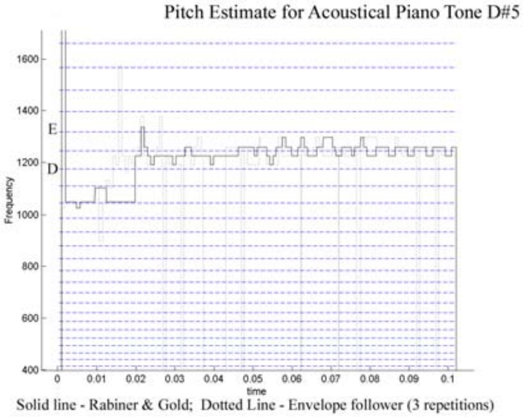


figure 6.f) Gold & Rabiner results

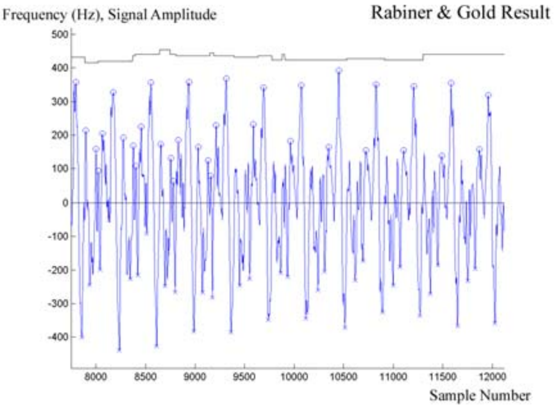


figure 6.g) The circles and crosses indicates the peaks found by the algorithm.

Center clipping can improve the performance of algorithms detecting peaks. The number of peaks will be reduced and facilitate the analyze process.

As for the envelope following algorithms, recursive lowpass filtering is tedious.

Summary:

Response time: Ok. A new estimate is calculated at each positive zero crossing.

Accuracy: Ok. Better than threshold crossing and envelope following algorithms, but many errors occur for difficult signals.

Resolution: Good. Can be improved by interpolation.

Complexity: Ok. Not cheap, includes both envelope following, zero crossing analysis and the candidate analysis. Analysing the candidates for the most likely period is tedious.

6.2.4 Results: Simple Basic Extractor with Filterbank Preprocessing

The method systematically fail when no fundamental is present. Selecting the filterbank channel having most energy often gives an estimate of type double- or triple octave error.

It was tested if selecting a lower filterbank channel could be done when the energy therein trespassed a certain threshold level. This method didn't prove to be accurate enough.

Summary:

Response time: Ok. Depends on the basic extractor used at the end of the filterbank.

Accuracy: Fairly Ok. The results were not accurate.

Resolution: Ok. Depends on the basic extractor used at the end of the filterbank.

Complexity: Ok. At least one bandpass filter per octave and one or more basic extractors.

6.2.5 Results: One pole notch adaptation with RLS and LMS

The parameters for a one-pole notch filter has been calculated using adaptive RLS algorithm. RLS resulted in faster and more accurate result than the LMS algorithm. The method suffers from serious problems such as its sensitivity to noise. Figure 6.h shows the result from a composition of a saxophone tone and a pure sinusoidal signal. The reference estimate is given from the reduced ACF PDA. The RLS algorithm can be set with a parameter λ which decides the behaviour of fast response versus stable estimate.

The noisy saxophone signal doesn't come close to the true estimate at any point, which it however does for the pure sinusoid. The estimate is adapted very slowly to the true sinusoidal fundamental, which could be changed by a different value of λ. This could then on the other hand cause the signal to become unstable.

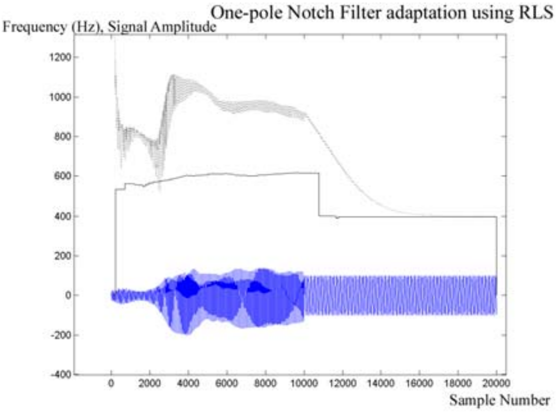


figure 6.h) One pole notch filter adaptation

Summary:
Response time: Fairly Ok. Can not be set too fast, since that could make the algorithm unstable.
Accuracy: Poor. Very noise sensitive.
Resolution: Good for a noise-free signal.
Complexity: Fairly Ok. Not cheap. Most effort is used to calculate the filter parameters.

6.2.6 Results: Autocorrelation

(Algorithm 5.2.14)

The autocorrelation PDA is very reliable and accurate. The same piano tone analyzed in 6.2.2 and 6.2.3 gives for the ACF the result in figure 6.i. After the attack, when the tone is somewhat stationary, the result is very exact. In this simulation the estimate is updated every sample.

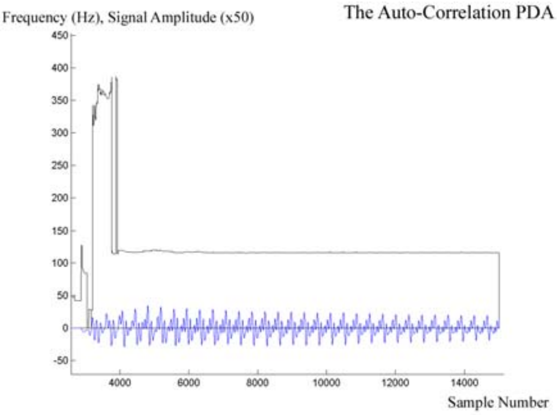


figure 6.i) Results from acf PDA

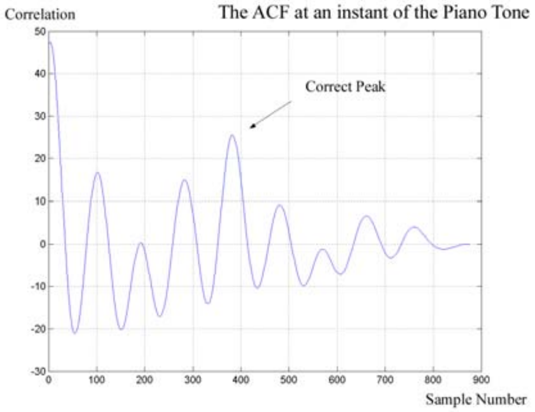


figure 6.j) The correct peak situated at the fundamental period sample

The algorithm picks the maximum peak of the acf and assumes this corresponds to the fundamental period (figure 6.j). The problem of only selecting the closest peak is that the resolution can be very bad. Using a larger correlation window will adjoin additional peaks and the distance can be averaged.

Here only lowpass filtering has been used in the pre-processor. Center clipping does improve the result.

Summary:
Response time: Very Good. Can be updated every sample if implemented so.
Accuracy: Very Good. Very accurate when calculated at every sample.
Resolution: Ok. Depends on the length of the correlation window. Averaging over several peaks is necessary.
Complexity: Fairly Ok. The main algorithm is an ordinary acf which will demand many calculations. The acf can be speeded up by using the FFT[27].

[27] According to [6], the number of real multiplications using the over-lap save method would cost (N = frame length, M = overlap data) $c_{FFT} = 4\ N\ log_2 2N\ /\ (N\text{-}M+1)$ calculations per

6.2.7 Results: AMDF

(Algorithm 5.2.15)

Hess [3] concluded about the AMDF PDA's:

- The autocorrelation and AMDF PDA's are comparable in performance. Highly correlated signals have a low minimum of the AMDF and vice versa.
- The significant minima of the AMDF are usually sharper than the corresponding peaks of the ACF.
- The computation of the AMDF is faster since no multiplication's are needed and it allows for a comparatively short frame.
- The AMDF is sensitive to intensity changes, whereas the ACF is fairly insensitive.
- Both ACF and AMDF are sensitive to a dominant formant structure, which is helped up by linear and/or non-linear preprocessing.

The AMDF was tested and compared to the ACF function, but showed inferior performance. This was probably caused by the sensitivity to intensity changes mentioned above.

Implementing the AMDF PDA, as well as the ACF PDA, would exceed the limitations given by the target system, and therefore the further studies concentrated on the cheap algorithm derived from the Cooper and Ng method.

Summary:
Response time: Good. Depends on the implementation. Could be updated every sample.
Accuracy: Good. Did show inferior performance to the ACF PDF in simulations.
Resolution: Good. Has a sharper peak than the ACF and could be improved by interpolation.
Complexity: Fairly Ok. Not cheap. Comparable to the ACF but no multiplication's is done, only subtractions. This is however not an advantage in the target system.

6.2.8 Results: Cooper et al

(Algorithm 5.2.18)

The PDA by Cooper and Ng was a very interesting algorithm because of its cheap calculation cost. Initial simulations of the algorithm gave results that were very promising. Figure 6.k shows a typical pitch contour resulting from the algorithm.

The algorithm has some design parameters that must be set in an appropriate way for optimum result. There are a number of thresholds that must be set. What is the least similarity ratio necessary for accepting the segments as similar? How equal in time must the segments be? How many segments should be saved an compared?

Trying different setups and signals, revealed that the appropriate similarity ratio threshold was dependent on the frequency of the tone played. A low frequency tone was best pitch tracked when the similarity ratio was high. For higher pitched tones, this demand was not as crucial.

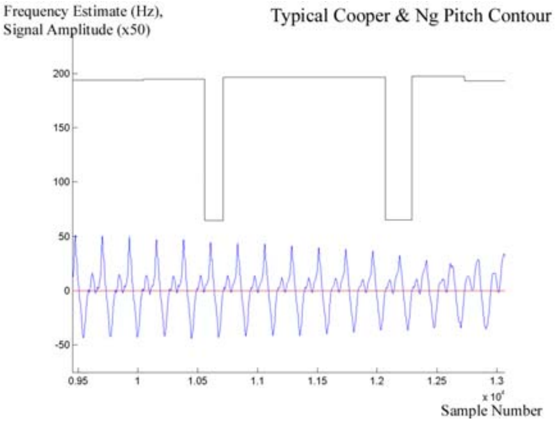


figure 6.k) Results from the Cooper and Ng PDA

Summary:
Response time: Ok. A new estimate is calculated at each positive zero crossing.
Accuracy: Ok. Better than threshold crossing and envelope following algorithms.
Resolution: Ok. Zero crossing resolution can be improved by interpolation.
Complexity: Very Good. Cheap. Involves a correlation calculation and signal feature extraction.

6.2.9 Results: Reduced ACF

(Algorithm 5.2.19)

The experiences made from the simulations of the Cooper and Ng PDA, led to the development of an algorithm that would be more suitable for real-time implementation. This algorithm has accordingly been developed by Leopold Roos and the author during this project.

The main problem with the Cooper and Ng PDA, is that the waveform landmarks, the six points, can not be chosen directly at the sample instant. One segment has to be saved, and then the six points are picked. The six points can be seen as describing essential characteristics of the waveform, would there be other ways of describing the characteristics? Preferably these characteristics should be possible to pick at the sample instant, with no memory necessary.

Characteristics that have been considered are:

- The area of the signal, one for the positive part and one for the negative.

output point, compared to c_{corr} = M for the direct cross correlation.

- The maximum value and the minimum value.
- The time instants at where the maximum and minimum occur.
- The length in time of the segment.
- Number of derivative sign changes during the segment.

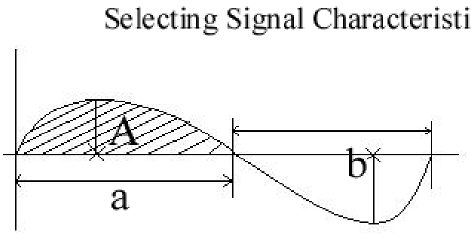


figure 6.l A - positive area, a - positive length, b - minimum instant occurrence

As a result, the characteristics that were most important and showed necessary for accurate correlation / similarity calculations were, the maximum- and minimum-value and the length of the segment.

For the improvement of the algorithm, it seemed that some kind of intelligent preprocessing was necessary. It was therefore investigated if the lowpass filter cutoff frequency could be set recursively and if the center compression level as well could be set recursively. Figure 6.m. shows part of the result when a fixed cutoff frequency of 3000 Hz is compared to the recursively set cutoff.

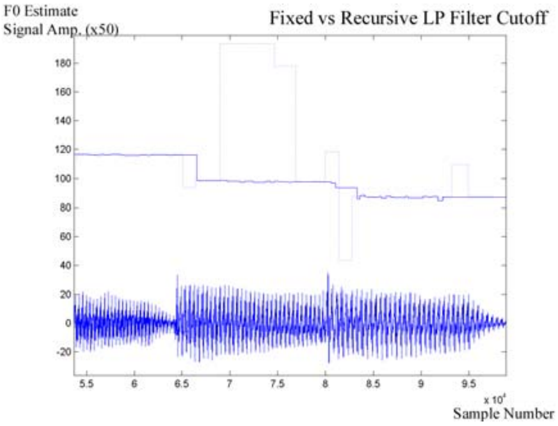


figure 6.m Solid - Recursive Cutoff
Dotted - Fixed Cutoff

Simulations have shown that the length of the segments saved, was a good measure for an upper limit of the fundamental frequency. The filter cutoff is set corresponding to the mean length of the four most recent segment lengths.

The "tight" lowpass filtering is one of the reasons why the few characteristical values, max, min and segment length suffice for good performance. Other characteristics of the signal, deriving from higher partials are removed by the lowpass filter.

The center compression level is set at a rate of 40% of the maximum value of the four most recent segments.

As mentioned in 6.2.8, the similarity ratio threshold, the means for deciding which segments belong together, had in simulations shown that low pitched tones require high similarity ratio threshold for best performance. Therefore the similarity ratio level is set as function of the lowpass filter cutoff frequency.

A simulation was run to compare the Cooper and Ng to the Reduced ACF (figure 6.n.). Comparing is difficult and the setup for the Cooper and Ng algorithm is done according to the authors interpretation from the document by Cooper and Ng [1]. In this simulation the signals were filtered with the same LP filter. The Reduced ACF uses no confidence counter (8.6), but Cooper & Ng needed a length of 5 to reduce the number of errors. At least it is clear that the Reduced ACF is not inferior to the Cooper and Ng algorithm.

Figure 6.o. illustrates a problem with the method often occurring. When the smaller peak in one of the segment just pass the zero level, and the segmentation is altered, half pitch errors can occur. One thing that reduce the effect of this is to some part center clipping. Also, since correlation is done both for the maximum and the minimum segment, and the periods must agree, the effect is reduced. This is also the main reason for the much more stable result than the Cooper and Ng PDA.

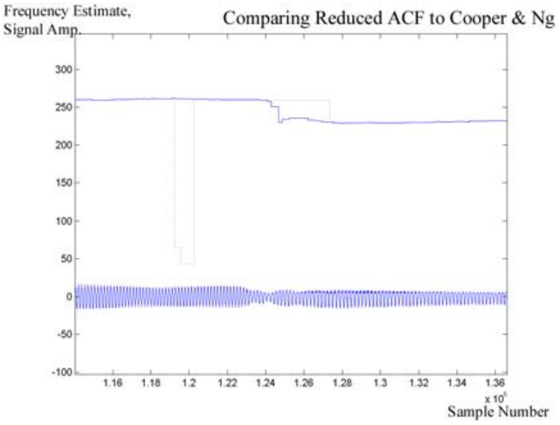


figure 6.n) Comparing the reduced ACF to the Cooper and Ng algorithm.

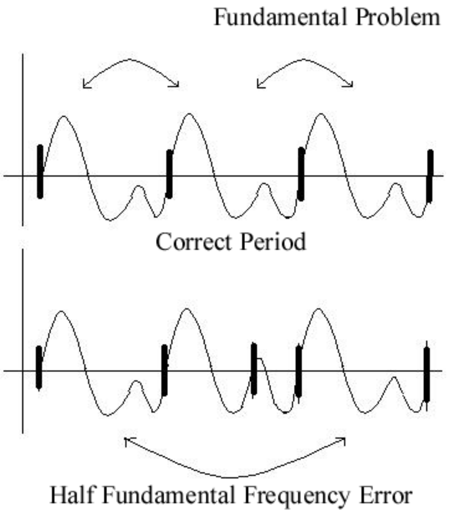


figure 6.o) A typical error when segmenting the signal.

Summary:
Response time: Ok. A new estimate is calculated at each positive zero crossing.
Accuracy: Ok. Better than threshold crossing and envelope following algorithms.
Resolution: Ok. Zero crossing resolution can be improved by interpolation.
Complexity: Good. Cheap. Involves a correlation calculation and signal feature extraction.

6.3 Results Spectral Domain PDA

Here, results from spectral domain PDA's are presented.

6.3.1 Results: FFT with different harmonic analysis

(Algorithm 5.3.3 – 5.3.6)

In figure 6.k., the same piano signal as analyzed in 6.2.6, has been analyzed using a Fourier transform. The dotted line shows the pitch estimate when simply picking the maximum peak of the spectrum, and the solid line is the estimate when to that peak, using the division method.

The result is very bad, it is obvious that studying individual peaks of the spectrum will fail. Harmonic overtones are chosen very often for this signal. The division method gives a better result, but does often choose a subharmonic estimate. It is also tedious finding a limit for how many sub peaks the method should look for and if the peak verified should be accepted (i.e. setting an appropriate threshold). For softly played signals with few overtones, the results are however better.

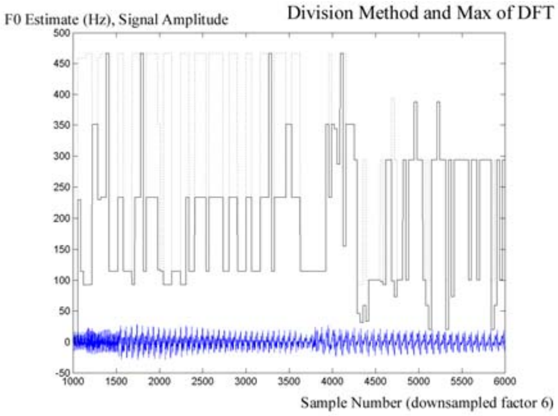


figure 6.p) Simple spectral peak extracting methods

The P & G method (5.3.5) explores the relationship of the peaks of the spectrum and figure 6.p. shows that the result is seriously improved. The method is however not accurate enough. There are still some parts where the method fail.

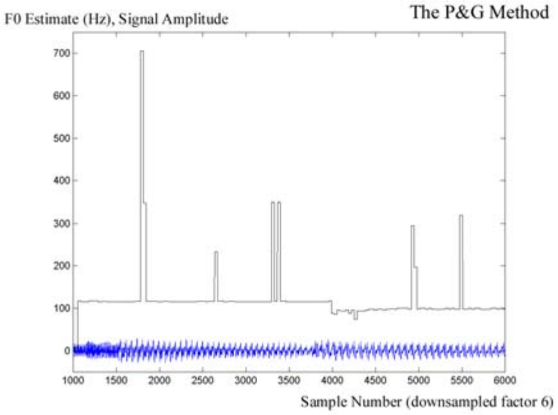


figure 6.q) Results from the P&G method

The resolution of the FFT must be quite high to accurately determine the peaks. When the FFT window of the simulation in figure 6.q. was halved to 512, the number of errors were doubled.

Summary:
Response time: Good. Depends on the window length and the update interval. The update interval could be done at every sample.
Accuracy: Ok. Depends on which spectral analysis method used.
Resolution: Good. Depends on the FFT size N, giving resolution 1/N. Could be improved by parabolic fitting to adjacent frequency bin values.
Complexity: Fairly Ok. Needs a Fourier Transform.

6.3.2 Results: Constant Q

(Algorithm 5.3.7)

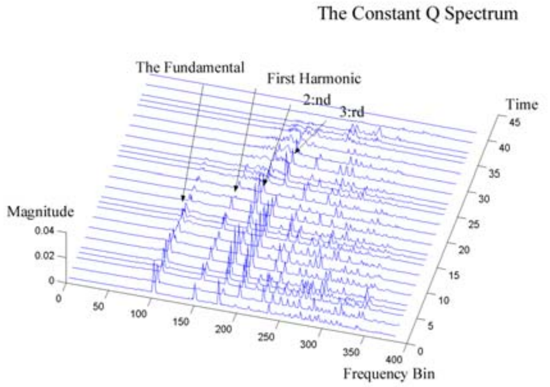


figure 6.r) The logarithmic spectrum from the constant Q transform

The spectral frequencies resulting from the Constant Q Transform, are logarithmically spaced. The results for all the harmonic analysis techniques presented in the two proceeding sections can also be applied this logarithmic spectrum. Of course they must be modified to handle the logarithmic spacing. e.g. the harmonic summation method corresponds to using the harmonic product method (5.3.6).

The high resolution PDA presented by Brown (5.3.7) has in the simulations proved to have high accuracy. However, the high resolution results are often spoiled when frequency bin selection errors occur, such as illustrated in figure 6.r., where the higher harmonics are stronger than the fundamental. In the simulations the Piszczalski and Galler method was used for bin picking. A second problem is that the high resolution is useless if too few frequency bins are used. The method can modify the frequency estimate about 2 % from the initial estimate.

The transform was successfully implemented using the FFT and calculations could also be saved by using the precalculated Kernel as proposed by [38].

Summary:

Response time: Good. Depends on the window length and the update interval. The update interval could be done at every sample.

Accuracy: Ok. Depends on which spectral analysis method used.

Resolution: Very Good. Constant frequency to resolution value Q, i.e. Better resolution at higher frequencies.

Complexity: Fairly Ok. Needs a Fourier Transform.

6.3.3 Results: Phase Vocoder

(Algorithm 5.3.8)

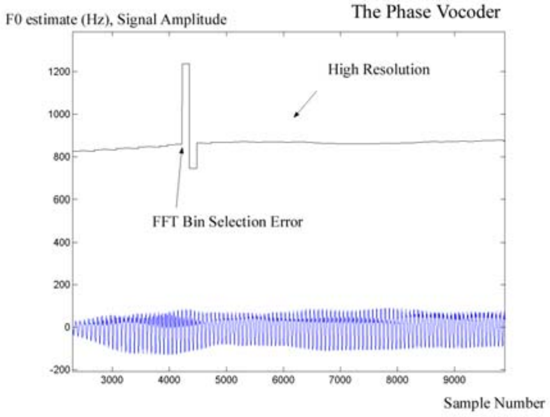


figure 6.s) Interpolation using the phase vocoder

The phase vocoder gives very high resolution of the frequency estimate. In this simulation (figure 6.s.) of the saxophone signal, the frequency estimate was determined at about 0.25 Hz resolution. As for the high resolution algorithm discussed in the previous section, the result is sensitive to finding the correct pitch estimate in the spectral bin search. In this simulation, only the maximum of FFT method was run, which in preceding sections has proven to be too poor.

Summary:

Response time: Ok. Depends on the window length and the update interval. The update interval could be done at every sample.

Accuracy: Ok. Depends on the initial frequency bin finding method.

Resolution: Very Good. Very high resolution.

Complexity: Poor. Needs a Fourier Transform and phase study.

6.3.4 Results: Cepstrum

It is known according to [27], that FFT based Cepstral methods are accurate and reliable for determining fundamental frequency in voice signals, but also that the method degrades severely in a noisy environment.

A way of coping with this, could be a pitch estimator that utilizes the MUSIC algorithm for estimating background noise characteristics (introduced in [27]) and compensates for this.

The Cepstrum has been simulated for musical signals, but the results have not been as good as expected. Since the Cepstrum needs a lot of processing, it has therefore not been considered interesting for further analysis.

The Cepstrum should, like the ACF, benefit from centerclipping.

6.4 Results Combined PDA

Algorithms using t both temporal and spectral techniques are presented here.

6.4.1 Results: Rough FFT and sinusoid adaptation

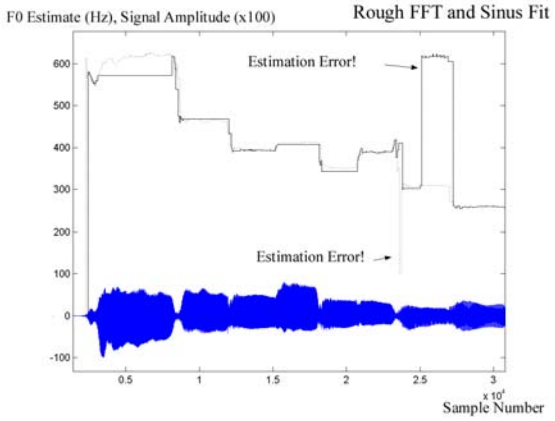


figure 6.t. *Solid line – LS Fit*
Dotted line – Reduced ACF

In the search for a cheap algorithm in spectral domain, the size of the FFT was made smaller. In this PDA a FFT of 128 points was used to establish a rough estimate, which was refined by least squares fitting the signal to a sinusoidal signal. The fitting was made in an iterative algorithm, where the estimate was improved in each cycle. Hence the resolution was arbitrary, only depending on the number of iteration cycles used.

Fitting the signal to a sinusoidal signal, will fail when the signal has characteristics like the bass signal in figure 4.b. at page 16. For more ‘simple’ signals like the saxophone signal, the algorithm should perform better. In figure 6.t. the result is compared to the result from the Reduced ACF PDA for the saxophone.

This PDA does however track an higher octave in the vicinity of sample 2,7 x 10^4. The rough frequency bin picking does often pick higher harmonics.

One great opportunity of fitting the signal to a reference is that a measure of the error will be available when comparing the original to the adapted signal. Thus, it is possible to know if the estimate is good enough. But, as mentioned the method suffers from the assumption of sinusoidal behaviour.

Summary:
Response time: Good. Depends on the window length and the update interval. The update interval could be done at every sample.
Accuracy: Poor. The sinusoidal fit is questionable. Not safe to tracking higher harmonics.
Resolution: Very Good. Very high resolution.
Complexity: Poor. Needs to compute a FFT and a following least squares calculation in at least three iterations.

7. Preprocessing

The term preprocessing means that this functional block is put first in the PDA, in front of a basic extractor.

The purpose of preprocessing the received data, is partly to reduce the amount of data and partly to extract the features of the fundamental frequency. The periodicity information should be augmented.

Preprocessing can strictly range from a simple lowpass filter to advanced Fourier transform. In this thesis, preprocessing functions have been considered to be simple operations that are used in a common way in many PDA's.

In this section, theory is discussed, but also some results from the simulations are put in.

7.1 Lowpass filtering

Coming across a PDA that does not apply lowpass filtering to the input data is very rare, but there are a few examples. A lowpass filter, when put at the appropriate cut off frequency, can do miracles to the performance of a PDA. Figure 7.a shows how the results may vary for the bass tone discussed in 4.5, as the cut off frequency is altered. (The theory of filtering is discussed in 3.4.)

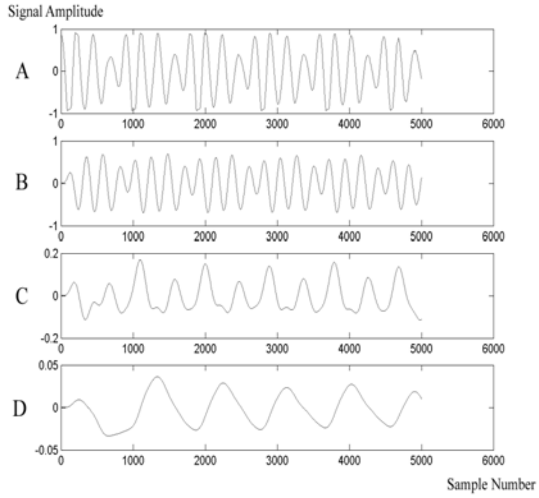


figure 7.a A – Original signal (fundamental 50 Hz)
B – Lowpass filtered at 200 Hz
C – Lowpass filtered at 100 Hz
D – Lowpass filtered at 50 Hz

7.2 Bandpass filtering

Bandpass filtering can be applied to eliminate frequencies outside the measuring range.

Bandpass filters are used in filterbanks, which are sometimes used to filter out the measuring range into subranges of about one octave.

7.3 Highpass filtering

Highpass filtering can be done to remove DC- or other lowfrequency-components. A very simple way of doing this is to differentiate the signal as $y(n) = x(n) - x(n-1)$.

In Dolansky's envelope following algorithm [9], a first order highpass filter is applied to the envelope, that way increasing the performance by removing lowfrequncy-components and only saving information of where there are abrupt changes.

7.4 Filtering adaptively

By adaptive filtering one usually means, either a setup where a system transfer function is identified, or a setup where a set of filter parameters are configured to minimize the difference between two signals.

The identification setup is usually performed by an iterative method like LMS, to determine a set of FIR parameters. This approach is equivalent to 5.2.13.

In this report, filtering adaptively, refers to a recursive way of setting the cut-off frequency for the preprocessing filter.

Finding a source for the decision of cut-off frequency can be tedious, but referring to the algorithm developed in this project (5.2.19), this was solved by assuming the mean of all segments saved, to be an upper limit for the fundamental frequency. Therefore, the preprocessing lowpass filter was set to cut at this upper limit frequency.

The advantage is that a very tight lowpass filtering can be achieved, and the pre-processor will remove some of the harmonics. As seen in 7.1, this is important for improved performance.

The risks of filtering adaptively, is that when abrupt changes of the input signal occur, the filter cut-off will not respond, resulting in an extinguished signal. However, the simulations made, have shown that the method can follow fast changes of pitch.

Abrupt variations of the cut-off frequency for a high-ordered filter, may cause the filter to become 'unstable', in the sense that it's amplitude will run toward a great value.

A solution to this is to let the complex filter pole pairs move slowly through the complex plane. It can also be solved by using a filter of sufficiently low-order, or having the cut-off frequency change in quantized steps where the state of the filter is pre-calculated for the new cut-off frequency. In this project, the problem was solved using a low-ordered filter.

7.5 Increasing resolution through interpolation

Interpolation of the signal samples, is necessary in most systems to improve resolution for pitch estimation. This is especially obvious at high pitched signals, where the number of samples per fundamental period can be very few. The resolution in time-domain is directly corresponding to the sampling frequency. If the period calculation is based on positive zero crossings, even a sampling rate of 96 kHz will result in a period estimation accuracy corresponding to 100 cent error for a semitone close to 4,0 kHz but only maximum 2 cent error for a tone at 100 Hz.

Zero crossing interpolation is most easily calculated using a linear interpolation between the two samples on each side of the zero level.

A way of improving the estimate for period time, and make it robust to noise was proposed by Johan Liljencrants [39], where each sample in two adjacent periods are compared. It needs more calculations but results in an improved estimate.

Peak interpolation is not only used for peaks in time-domain, but also at spectral peaks as a useful way of improving the resolution for a DFT.

A common way of doing this is to fit a parabola to the three values situated where the maximum point is. This can be done by a least squares approximation of the three points to the function $ax^2 + bx + c = y$, which will yield a maximum value index $x_{max} = -b / 2a$.

The phase vocoder can also be seen as a technique interpolating adjacent DFT frames (See 5.3.8).

7.6 Spectral flattening

Removing the formant structure from the signal is called *spectral flattening*. There are linear and non-linear methods.

The linear methods involve filtering methods. Sondhi proposed a method where the signal is run through a bandpass filterbank [3]. The output from each filter is divided by its mean-energy, and then summed up, in a resulting spectral flattened signal. Sondhi himself though claimed this method to be inferior to the center clipping technique (7.6.2). An alternative used in voice signal is inverse filtering (7.11)

The non-linear methods for spectral flattening are more common and originates from 1948 [3]. The following sections discuss the functions normally used.

The performance of the PDA is improved if the non-linear processing is performed prior to the lowpass filtering [40].

7.6.1 Cubic non-linear distortion

One approach for degrading the formant structure, is cubic distortion. Each signal sample is then raised to factor 3. The performance is comparable to center clipping [3], but has the disadvantage of being amplitude dependent.

An alternative is squaring the signal with maintained sign.

7.6.2 Center Clipping

The Center-Clipping function sets all absolute values under a threshold to zero. This is a common spectral flattening technique. What the threshold value should be set to, is not specified. A common approach is to adjust the level according to the maximum value discovered in the most recent part of the signal. Hess [3] propose a method where the last 30 ms of the signal is divided in three segments. A maximum value for each of the segments is found. The clipping level is then set to 80 % of the smallest of the maximum values found.

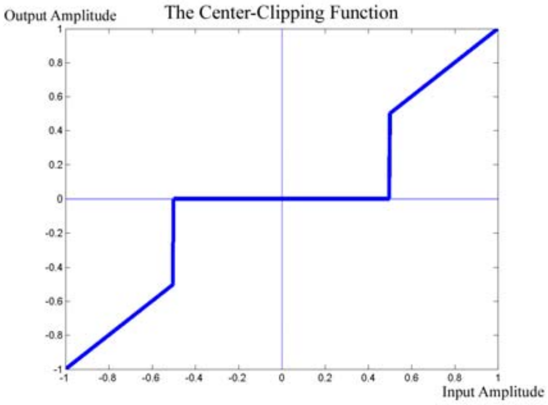


figure 7.b. Center Clipping Function

7.6.3 Center Compression

Letting the signal be both Center-Clipped and compressed results in a function according to figure. This is the method chosen for this project's implementation, based on a great comparative study by Rabiner in 1977 [3], where the Center Compression technique showed nice results. Simulations have been done to confirm the results. In the study, Rabiner compared peak-clipping, center-clipping, center-clipping and compression and the sign function.

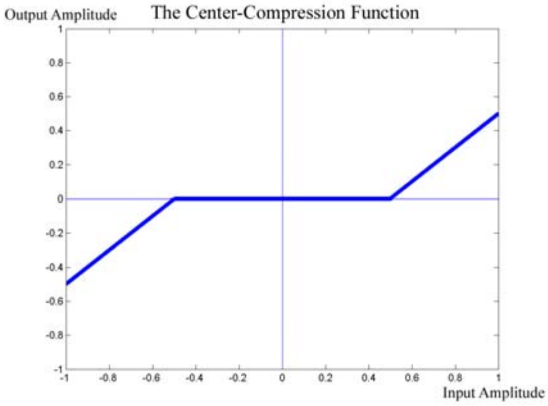


figure 7.c. The Center-Compression Function

7.6.4 Sign function

An alternative spectral flattening technique performs compression and only save the polarity information of the signal, as described in figure 7.

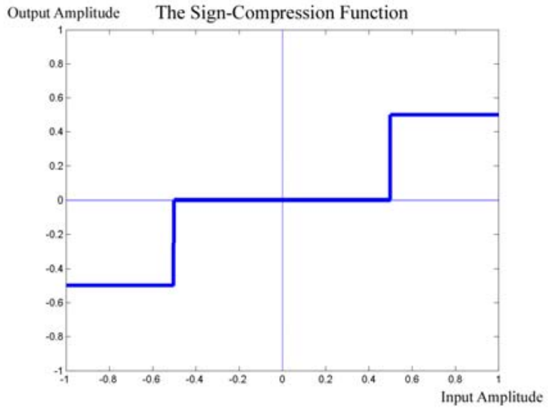


figure 7.d. The Sign Function

7.7 Adaptive Spectral Flattening

The spectral flattening techniques are normally using recursive methods. Setting the clipping level to a constant value will of course make the method very sensitive to the signal amplitude.

The level can be set similar to the approach described in 7.6.2. This is an elegant way of reducing the influence of amplitude-varying signals to the pitch estimation result.

This project's implementation sets the compress-level to 20 % of the maximum found in the last four segment's. (See 9)

A problem that can occur, is that when the processed signal does unexpected jumps, the compress level will be set unnecessarily high. This happened in the project when the signal became unstable at filter cut-off changes as discussed in 7.4.

7.8 Downsampling the Input Signal

Downsampling is an easy and efficient way of reducing data, as was the purpose of preprocessing. The Nyquist Criterion and the measuring range sets a limit for the amount of downsampling possible. Since we're interested in signals having frequencies up to 4000-5000 Hz, downsampling so the sampling rate fall short of 10 kHz, will be devastating. According to section 3.5, no information is lost if the sampling frequency is above the Nyquist rate.

Another problem is the loss of resolution at zero-crossings and at peaks, which will decrease the accuracy in corresponding period estimates. This can however be solved by keeping higher sample rate at zero crossings.

One advantage of the frequency domain PDA's, is that the spectral resolution and with it, the accuracy of the measurement, does not suffer from time domain downsampling.

7.9 Inverse filtering

Not analyzed in this project, but useful on voice signals. A filter is adapted to the vocal tract (2.3) and the signal is inverse filtered to reconstruct the glottal signal (2.3), which is easier pitch estimate.

A major drawback is that prior knowledge of the system frequency response is necessary.

8. Postprocessing

Postprocessing is the functional block put at the end of the pitch period determination algorithm. Its task is to remove obvious pitch detection errors and to refine the result. The most obvious errors occurring in a PDA are octave errors, since higher partials are tracked.

8.1 Online and Offline postprocessing

In offline postprocessing, all future and past data are available at the instant where the pitch estimate is calculated. This of course facilitates the search of occurred pitch errors.

Dynamic programming is one method of efficient offline postprocessing, which is possible to implement if a measure of the strength of each pitch estimate is available. The pitch contour is found as the pitch path with minimum error. Thus it can find an optimal pitch contour in a global sense, though not optimal at each estimate. The optimal path can also be found using the Viterbi algorithm [41].

In this project however, this area has not been investigated since the pitch must be presented instantaneous. The functions used here can be labelled online postprocessing functions.

8.2 Time to Frequency conversion

One obvious function in the postprocessor is converting the pitch period estimate $\hat{T}_0$ to pitch estimate $\hat{F}_0$, in case period is the output of the basic extractor.

8.3 Linear Smoothing

Linear smoothing is done through lowpass filtering the pitch estimate. The filter length depends on the update rate of the estimate, but should be kept short, avoiding delay of the response. The length is best found by experiment. A length of three was used in the project.

8.4 Non-Linear Smoothing

The median filter is non-linear. $\hat{F}_0$ values are stored in a vector which is sorted in augmenting order. The output of the filter is the center value of the vector. The length of the vector must be odd and be kept short as discussed in 8.3.

8.5 De-Step Filter

Bagshaw presented (1994) [42] a filter where the pitch is only allowed to change 75 % of an octave from one estimate to another if there is a continuous signal. Else a doubling or halving error has occurred.

Each pitch estimate value F_0 is put in a group G_x. If a doubling error occur, the new value is put in a higher group, i.e. x is increased. If a halving error occur, x is decreased.

The group that has most values is considered holding the correct estimate and for that group, x is set to zero.

Let f_i represent the i:th value of the current F_0 estimates, that is put in group $G_{x(i)}$. The group index x(i) is calculated according to

$$x(i)=\begin{cases} x(i-1)+\left[\log_2\left(\dfrac{4}{7}\cdot\dfrac{f_i}{f_{i-1}}\right)+1\right] & if \quad f_i \geq f_{i-1} \\ x(i-1)-\left[\log_2\left(\dfrac{4}{7}\cdot\dfrac{f_{i-1}}{f_i}\right)+1\right] & if \quad f_i < f_{i-1} \end{cases} \tag{8.1}$$

Thus G_0 represents the group having most F_0 values. The F_0 values are then corrected through a multiplication with $2^{-x(i)}$.

The method should include flushing of the group vectors, when a new note is detected.

8.6 Confidence Counter

Cooper and Ng used a confidence counter in their algorithm [1]. When a pitch estimate is found, a confidence counter is increased each time the same pitch is recognized at subsequent estimates, otherwise decreased. The system responds when the confidence count reaches a pre-defined maximum[28].

8.7 Using "rules"

When implementing a PDA, there are "rules" that can be used for increasing the performance of the algorithm. These rules can be used for recursive control of the algorithm. The control can be dependant on an accuracy measure of the estimate. Such a measure can in most algorithms be derived. Using the "rules" is a design matter which can be optimized by the PDA constructor.

The pitch estimate can be restricted to a predefined measuring range. If an estimate is found outside this interval, it is abandoned. The preprocessing BP filter (7.2) could take care of this, but the rules can be used as preventive measures.

If the energy of the signal is too low, below a certain threshold, it may be suitable to abandon the estimate or deactivate the gate signal (9.5.2).

When there is a sudden raise of the signal energy, it is likely that a new note is played and then also a new pitch. This could help when aiming for short response time.

[28] Cooper and Ng used a confidence maximum of 3.

9. Development and implementation of an efficient PDA in a DSP

The goal for the project was to implement one of the algorithms found in the study, in an existing musical synthesis system. The algorithm used and the system is described here.

9.1 The project specification

The specification for the project was given as follows:

- A consistent literature search and study of the pitch determination area, to obtain a number of algorithms and acquire knowledge of the complex of problems.
- Choose a number of promising algorithms, considering the target systems architecture, memory access, computational capacity and of course the characteristics of the expected material.
- Perform a first evaluation in MATLAB® or another highlevel language being appropriate.
- From the results and received experiences choose, alternatively develop a few variants for realtime implementation in the target system. The implementation is done in assembler for the signal processors.
- Perform evaluation and incorporate the result as a finished module in the product.

The assembler program is run in a Motorola 56303, 80 MHz processor with 4k + 2k + 2k word RAM. The pitch tracking module may use about 200 word program memory, 200 word variable memory and use about 100 cycles. (In this processor one multiplication can be performed in one cycle.)

9.2 About the Nord Modular System

The target system for implementation was the Nord Modular Synthesizer. This system uses virtual-analog modular synthesis, which means that digital technique is used to make a synthesizeer having an analog feeling.

A modular synthesizer is built up from interconnected basic modules, such as filters and oscillators. In the Nord Modular, the modules are linked together by the user in a software editor. There are a great number of modules, which makes great possibilities for generating varied sounds. At the instant a change is made to the patch[29], a new DSP program is compiled and uploaded to the synthesizer.

A pitch tracking module does not exist in the current system, but would be appreciated.

9.3 Theory of the Algorithm

The PDA that was decided to be tried for implementation was the algorithm already described in 5.2.19. The algorithm is described in figure 9.a.

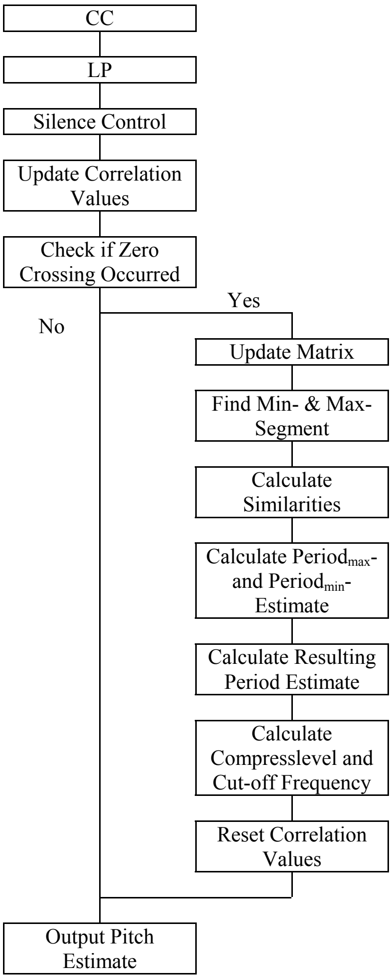

CC
LP
Silence Control
Update Correlation Values
Check if Zero Crossing Occurred
Yes
No
Update Matrix
Find Min- & Max-Segment
Calculate Similarities
Calculate Period$_{max}$- and Period$_{min}$-Estimate
Calculate Resulting Period Estimate
Calculate Compresslevel and Cut-off Frequency
Reset Correlation Values
Output Pitch Estimate

figure 9.a) The Reduced ACF algorithm

The details for the implementation can be found in a MATLAB®-script in appendix A. However some of the blocks are briefly presented here:

CC - Center Clipping and Compression, where the level is set at 40 % of the highest maximum in the four most recent segments.

LP - Lowpass filter, where the cutoff frequency is set as the mean value of the four most recent segment lengths.

Silence Control - Checks if the signal is beneath a certain threshold during a certain time.

Find Min- & Max- Segment - Picks the segment having the largest max value and also the segment having the largest min value.

Calculate Resulting Period Estimate - The maximum segment and the minimum segment are computed independently for a period. If the results

[29] Patch is the group of modules setup by the user.

only differ a small value, the mean of the two is presented as the fundamental period estimate.

9.4 Implementation in the DSP

The computational load on the DSP should be as low as possible. The implementation has been written in assembly code, for most efficient use of the processor cycles.

The resources available in the system for the pitch tracking algorithm is in program memory 200 words (24 bits / word), variable memory 200 words and 100 processor cycles. This is explained by the fact that the sampling frequency is 96 kHz and the processor runs at 80 MHz.

Some changes were made to the algorithm, to make it more suitable for implementation. Making divisions costs! For 24 bit resolution, one division costs 24 cycles. In the specification for the project, one module should preferably use only about 100 cycles (numbers are represented in 24 bits in the system).

When correlating value a in segment 1, to b in segment 2, the calculation is made from ***(ab / (aa+bb-ab)***, using the same method as the Cooper and Ng algorithm. This has to be calculated three times, since each segment is represented by three characteristical values.

A correlation could simply be calculated as: ***a / b***, if the denumerator is greater than the numerator. Simulations did not show any deterioration in performance. The total correlation of the values a to b, c to d and e to f, could then be calculated by only one division: ***(1/3) * (adf + bcf + bde) / (bdf)***. As shown in figure 9.b, this actually did not worsen the performance.

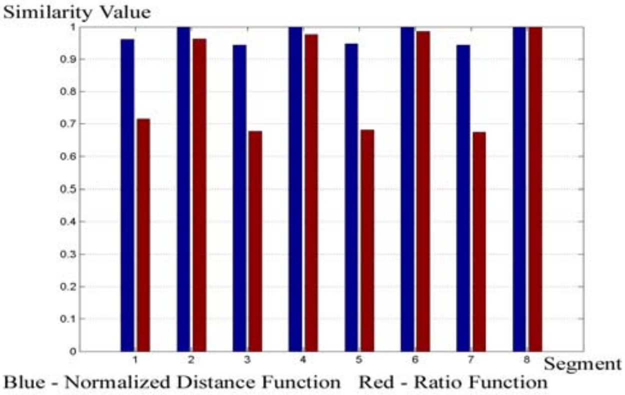


figure 9.b) An example where the simplified ratio function actually makes it easier to find a suitable threshold when comparing adjacent segment's. (Red on the right and Blue on the left.)

9.5 PDA-related *functions*

Other control signals than the pitch estimate itself are useful. A few are mentioned here.

9.5.1 Event Detection

In voice analysis it is useful to know if the signal is voiced or unvoiced. That matter has not been analyzed in this project, since the focus has been on musical signals.

Note detection is most simply done by observing the energy of the signal. When there is a sudden change of energy, a new note is assumed to be played.

The end of a tone is either a new tone or a pause. A pause is detected by calculating the mean energy during recent samples, and comparing this to a certain threshold.

9.5.2 Gate signal

A gate signal can be used as an output of the pitch detection unit, to indicate if the pitch signal is valid or not. The signal energy should be reflected in the gate signal, so that low energy cause a non-valid indication.

The algorithm itself can also have functions indicating if the pitch estimate is trustworthy, and setting the gate signal according to this.

9.5.3 Signal level

An output of the signallevel would make the unit a complete signal analyzer. This is not linked to the pitch, but definitely useful when resynthesizing the source signal.

9.6 Future Improvements

Making the algorithm even cheaper in calculations is not probable without worsen the performance severely.

If the number of calculations is not the limit, improvements of the performance is however possible.

The selection of segments could e.g. be improved by using a two-threshold method (compare 5.2.4), instead of picking it from zero crossings.

It would be possible to add additional correlation values, such as the curve area. However, in simulations it has been shown that there is only a small improvement by adding this correlation value, since the area measure is close related to length and height. Better would be to add an orthogonal measure, such as a curvature measure, counting the number of derivative change per segment.

The response time could be improved by giving an estimate by chance at the attack of tones, e.g. the having the period estimates given by the most recent segment length or given by an envelope following algorithm running in parallel.

10. Conclusions

The conclusions that can be drawn from the results of the simulations are discussed here.

10.1 Spectral and Time Domain PDA

The simulations have shown that spectral functions certainly can give good results, but there is more than only the transform to it. A fairly demanding analysis of the spectrum is needed for accurately finding the correct fundamental frequency estimate.

Time domain PDA's are inferior to the best spectral algorithms but still, they are able to produce quite good results, considering the often cheap computational load.

The PDA developed in this project is an example of a cheap implementation. The result for it is inferior to what would be reachable for an ordinary autocorrelation method. However, the results are surprisingly good.

Other successful PDA's use both spectral domain and time domain functions. There are examples where time domain is used for event detection and spectral domain is used for pitch detection, such as Davies/Etter's proposal from 1997 [43].

10.2 Pre- and Postprocessing

Preprocessing has shown to be an important issue for the PDA performance. This has been explored by recursively setting the lowpass filter cut-off frequency and the center clipping - compression level.

The perfect preprocessing method, would extinguish the harmonic overtones and strengthen the fundamental. This is however very difficult.

Postprocessing is difficult in an online situation. Here, it has been used for a short window pitch contour smoother and some rules for in which interval the pitch is allowed to be estimated.

10.3 Today's processors and memory influence of hardware when choosing PDA

For efficient real-time PDA's, with more intelligence, lots of calculations need to be done. This demands for even faster processors. More memory will of course also be necessary when implementing ACF functions and Fast Fourier Transforms.

The development of faster processors and more memory has continued, which is promising.

10.4 The future of Musical Pitch Tracking

The key for improved pitch determination algorithms is probably the combination of different existing methods. Hess [3] says: *"Rather than developing new principles of pitch determination, one should take the existing ones and combine them in an appropriate way to yield an overall improvement of performance."* He continues: *"When combining several principles one must take care that they perform in a complementary way so that the one works well where the other fails and vice versa."*

This study would have looked different if the goal was not to develop an algorithm for implementation in hardware, maybe being more focused on combining different methods.

However, also the reduced ACF PDA presented in this report, is a combination of different methods, namely polarity crossing analysis and correlation.

Preferably the PDA should be as intelligent as the human eye, in determining the fundamental period manually. The human eye and mind, can by looking at the time domain signal, quite accurately find the periodic behavior.

Appendix A

Sample MATLAB-code for the Reduced ACF PDA.

```
function [pitch_vec,Y] = reduced_acf (Y,FS)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% Master Thesis on pitch tracking SU and LR
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
%%%% Reduced acf algorithm.
%%%% Version 13.0, 2001-10-16
%%%% Correlates 3 values: min-, max- and periodlength of each
%%%% segment.
%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% Initiate values
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Antal_Segment       = 8;
Start_segm          = 8;

%%%% Algorithm-spec. variables:
max_pitch           = 5000;
min_pitch           = 30;
L                   = length(Y);
pitch_est           = 0;
pitch_vec           = zeros(1,L);
zero_crossed        = 0;
Y1                  = Y;                % used when filtering
MATRIX              = zeros( 5, Antal_Segment);
max_value           = 0;
previous_max_value= 0;
min_value           = 0;
s                   = 0;
p_tx_old            = 0;
area_ok_flag        = 0;
AREA1_proc          = 0.3;              % 0.2 = 20%.
AREA2_proc          = 0.3;              % 0.6 = 60%.
Similarity_threshold= 0.9; % 0.9 = 90%.
%%%%Percentage of how similar two segments are.

sim_index           = 8;
similarity_vector   = [0.8 0.7 0.7 0.7 0.6 0.6 0.5 0.5];
similarity_vec_save = zeros(1,L);
min_max_sim          = 0.05;
atleast_one_similarity= 0.9;
time_threshold      = 0.1;
%%%%
gate                = 0;% gate = 1 if pitch estimate ok, else 0.
gate_vec            = zeros(1,L);
silent_threshold    = 0.02;
silent_counter      = 0;
silent_flag         = 0;    % 0 if signal has been silent.
current_sign        = sign(Y(1));
do_zeroX            = 0;
lost_pitch          = 0;
compress_level      = 0.01;
compress_level_vec= zeros(1,L);
compress_rate       = 0.2;
cut_off_freq        = 5286;
cut_off_freq_lp     = zeros(1,L);
p_tx                = 0;
period              = 0;
period_old          = 0;
past_period          = zeros(1,3);
% Init for LP filter
x_n                 = 0;
x_n_1               = 0;
y_lp                = 0;
ylp_n_1             = 0;
w_n                 = 0;
w_n_1               = 0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% Run the algorithm
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for (p = 1: L-1)
  %%%% --PREPROCESSING:--
  cut_off_freq_lp(p) = cut_off_freq;
  compress_level    = 0.99 * compress_level;
  compress_level_vec(p) = compress_level;
  similarity_vec_save(p) = Similarity_threshold;

  %%%% --Clip and Compress--
  if( (Y1(p) < compress_level) & (Y1(p) > -compress_level) )
     Y1(p) = 0;
  else
    if(Y1(p) > 0)
       Y1(p) = Y1(p) - compress_level;
    else
       Y1(p) = Y1(p) + compress_level;
    end
  end

  %%%% -LP-filtering:- state filter a la NL.
  f_cut = cut_off_freq;
  q      = f_cut / FS;
  F      = 2*sin(pi*q);
  Q      = 0.5;
  R      = 1 / Q;

  x_n   = Y1(p);
  w_n   = F*x_n_1 - F*ylp_n_1 + (1-F*R)*w_n_1;
  ylp_n= ylp_n_1 + F*w_n;
  x_n_1   = x_n;
  ylp_n_1= ylp_n;
  w_n_1  = w_n;
  Y(p) = ylp_n;

  %%%% Silent signal?
  if(abs(Y(p)) < silent_threshold)
    silent_counter      = silent_counter + 1;
    if(silent_counter == 1200)
       gate          = 0;
       silent_flag    = 0;
       silent_counter  = 0;
       cut_off_freq    = 5280;
       lost_pitch      = 1;
       sim_index       = 8;
       Similarity_threshold        = similarity_vector(sim_index);
    end
  else
    silent_counter  = 0;
  end
%AREA2_proc = sign(max(0,s - period))*(AREA2_proc*0.95);
  if(s > 2*period)
    AREA2_proc = AREA2_proc*0.95;
    AREA1_proc = AREA1_proc*0.96;
  end

  %%%% --MAINPROCESSING:--
  %%%% --Setting signs and direction-variables:--
  previous_sign           = current_sign;
  current_sign            = sign(Y(p));
  zer_pos                         = 0;
  neg_pos                         = 0;

  if((current_sign==1)&(previous_sign==0));
    zer_pos          = 1;
  end
  if((current_sign==1)&(previous_sign==-1));
    neg_pos          = 1;
  end

  if( (current_sign < 0) & (abs(min_value) >
(AREA2_proc*max_value)) )

%%% Flag-setting because the current negative area-segment is
%%% big enough compared to previous pos.area-segm.:
    area_ok_flag = 1;
  end

  %%% --Setting the correlation-values:--
  s = s + 1;                  % Counter; numbers of samples
  if(Y(p) > max_value)
    max_value = Y(p); % corr-value
  end
  if(Y(p) < min_value)
    min_value = Y(p); % corr-value
  end

  %%% --Setting the "zero_crossed"-flag
  if ( do_zeroX )
    if( (max_value < AREA1_proc*previous_max_value |
          ~area_ok_flag));% & (peaks_passed == 0)) % |
      do_zeroX  = 0;
    else
      p_tx_old   = p_tx;
      %p_tx      = (p - 2) + ( 1 - (Y(p-1) / (Y(p) - Y(p-1))) );
      %%% Alternative interpolation
      y_3        = Y(p-1) - 0.5*(Y(p) - Y(p-1));
      x_3        = 0.5;
      y_4        = y_3 - (sign(y_3) * 0.25 * (Y(p) - Y(p-1)));
      x_4        = 0.5 + (sign(y_3) * 0.25);
      x_5        = x_4 + (sign(y_4) * 0.125);
      p_tx       = p - x_5;
      %%%
      do_zeroX          = 0;
      zero_crossed      = 1;

      MATRIX(1:3,1)      = MATRIX(1:3,2);
      MATRIX(1:3,2)      = MATRIX(1:3,3);
      MATRIX(1:3,3)      = MATRIX(1:3,4);
      MATRIX(1:3,4)      = MATRIX(1:3,5);
      MATRIX(1:3,5)      = MATRIX(1:3,6);
      MATRIX(1:3,6)      = MATRIX(1:3,7);
      MATRIX(1:3,7)      = MATRIX(1:3,8);
      MATRIX(1:3,8)      = [max_value min_value (p_tx –
                                       p_tx_old)]';
      MATRIX            = silent_flag * MATRIX;
      Start_segm = Start_segm + (1-silent_flag)*(9 –
                  Start_segm);
    end
  else
    zero_crossed = 0;
  end
  if ( (neg_pos | zer_pos) & area_ok_flag )
    do_zeroX = 1;
  end
  %%% --Calculations when zero is crossed from neg to pos.:---
  if( zero_crossed )
    %%% -Calculation-algorithm starts
    gate             = 1;

    AREA2_proc = 0.3;
    AREA1_proc = 0.3;

    Start_segm = max(1,Start_segm - 1);

    [max_place_value, index_max] = max(MATRIX(1,:));
    [min_place_value, index_min]  = max(abs(MATRIX(2,:)));
    if( index_max == index_min )
      temp_vec                  = MATRIX(2,:);
      temp_vec(index_min)    = 0;
      [max_place_valu2, index_min]   = max(abs(temp_vec));
    end

    %%%
    if (index_max == 8 | pitch_est == 0)
      atleast_one_similarity              = 0.6;
      min_max_sim                         = 0.7;
      time_threshold                      = 0.3;
    else
      min_max_sim                         = 0.05;
      time_threshold                      = 0.1;
    end

    MATRIX(4:5,:) = 0;
    i_max           = index_max;
    period_max      = MATRIX(3,index_max);
    jumping_index = max(Start_segm,index_max);
    jump_direction = -1;
    jumping_FLAG= 0;
    test=0;
    stop_FLAG_max= 0;

    i_max           = index_max;
    for(d           = 1 : Antal_Segment-Start_segm)

      if(jumping_index == 8 & test == 0)
        jumping_FLAG = 1;
        jump_direction =-1;
        jumping_index = jumping_index - d;
        test=1;
      end
      if(jumping_index == Start_segm & test == 0)
        jumping_FLAG = 1;
        jump_direction =1;
        jumping_index = jumping_index + d;
        test=1;
      end
      test=test+test;
      if(jumping_FLAG == 1 & test > 2)
        jumping_index = jumping_index + jump_direction;
      elseif(test==0)
        jumping_index = jumping_index + jump_direction*d;
        jump_direction  = -1*jump_direction;
      end

      asim   = min([MATRIX(1, index_max)
                  MATRIX(1,jumping_index)]);
      bsim   = max([MATRIX(1, index_max)
                  MATRIX(1,jumping_index)]);
      csim   = max([MATRIX(2, index_max)
                  MATRIX(2,jumping_index)]);
      dsim   = min([MATRIX(2, index_max)
                  MATRIX(2,jumping_index)]);
      esim   = min([MATRIX(3, index_max)
                  MATRIX(3,jumping_index)]);
      fsim   = max([MATRIX(3, index_max)
                  MATRIX(3,jumping_index)]);

      sim_ratio_alternative =
                (1/3) * (asim*dsim*fsim +
                bsim*csim*fsim + bsim*dsim*esim) /
                (bsim*dsim*fsim);
      MATRIX(4,jumping_index) = sim_ratio_alternative;
      if(stop_FLAG_max ~= 1
                & MATRIX(4,jumping_index) >
                MATRIX(4,i_max))
        i_max = jumping_index;
      end
      if(MATRIX(4,jumping_index)>Similarity_threshold
                & stop_FLAG_max ~= 1)
        i_max                 = jumping_index;
        stop_FLAG_max = 1;
      end
   end
  i_min            = index_min;
  period_min       = MATRIX(3,index_min);
  period_max       = MATRIX(3,index_min);
  jumping_index = max(Start_segm,index_min);
  jump_direction = -1;
  jumping_FLAG =0;
  test=0;
  stop_FLAG_min=0;
  period_A         = MATRIX(3,index_min);
  period_B         = MATRIX(3,index_min);
  i_min            = index_min;
  for(d = 1 : Antal_Segment-Start_segm)
        if(jumping_index == 8 & test == 0)
        jumping_FLAG = 1;
        jump_direction=-1;
        jumping_index = jumping_index - d;
        test=1;
     end
     if(jumping_index == Start_segm & test == 0)
        jumping_FLAG = 1;
        jump_direction=1;
        jumping_index = jumping_index + d;
        test=1;
     end
     test=test+test;
     if(jumping_FLAG == 1 & test > 2)
        jumping_index = jumping_index + jump_direction;
     elseif(test==0)
        jumping_index = jumping_index + jump_direction*d;
        jump_direction  = -1*jump_direction;
     end
     asim   = min([MATRIX(1, index_min)
                    MATRIX(1,jumping_index)]);
     bsim   = max([MATRIX(1, index_min)
                    MATRIX(1,jumping_index)]);
     csim   = max([MATRIX(2, index_min)
                    MATRIX(2,jumping_index)]);
     dsim   = min([MATRIX(2, index_min)
                    MATRIX(2,jumping_index)]);
     esim   = min([MATRIX(3, index_min)
                    MATRIX(3,jumping_index)]);
     fsim   = max([MATRIX(3, index_min)
                    MATRIX(3,jumping_index)]);
     sim_ratio_alternative = (1/3) *
                    (asim*dsim*fsim +
                    bsim*csim*fsim + bsim*dsim*esim) /
                    (bsim*dsim*fsim);
     MATRIX(5,jumping_index) = sim_ratio_alternative;
     if(stop_FLAG_min ~= 1 &
                    MATRIX(5,jumping_index) >
                    MATRIX(5,i_min))
        i_min = jumping_index;
     end
     if(MATRIX(5,jumping_index)>Similarity_threshold
                    & stop_FLAG_min ~= 1)
        i_min = jumping_index;
        stop_FLAG_min = 1;
     end
  end
  if(i_min<index_min)
     period_min=sum(MATRIX(3,i_min + 1:index_min));
  else
      period_min=sum(MATRIX(3,index_min:i_min-1));
  end
  if(i_max<index_max)
     period_max=sum(MATRIX(3,i_max + 1:index_max));
  else
      period_max=sum(MATRIX(3,index_max:i_max-1));
  end
  period_min=sign(abs(i_min-index_min))*period_min;
  period_max=sign(abs(i_max-index_max))*period_max;
  if(abs(1-period_max/max(1,period_min))  < min_max_sim
      & (MATRIX(4, i_max) > atleast_one_similarity
       | MATRIX(5, i_min) > atleast_one_similarity))
     period = (period_max + period_min )*0.5;
  else
     period = period_old;
     lost_pitch = 1;
  end
  if(abs(MATRIX(3, i_max) - MATRIX(3, index_max)) >
       ( MATRIX(3, index_max) * time_threshold ) | ...
         abs(MATRIX(3, i_min) - MATRIX(3, index_min)) >
       ( MATRIX(3, index_min) * time_threshold ) )
     period = period_old;
     lost_pitch = 1;
  end
  compress_level = compress_rate*( max(MATRIX(1,5:8)) );

  %%% Calculate new cutoff frequency.
  cut_off_freq          = FS / max(1,max(MATRIX(3,:)));
  if(pitch_est > cut_off_freq)
     temp_pi = find(MATRIX(3,:));
     if(isempty(temp_pi))
        cut_off_freq          = 5280;
     else
        cut_off_freq          = FS /
                      max(1,min(MATRIX(3,temp_pi)));
     end
  end
  cut_off_freq            = 200+max(50,cut_off_freq);
  cut_off_freq            = min(5000,cut_off_freq);

  if(cut_off_freq > 700 & pitch_est ~= 0),
  Similarity_threshold = 0.5;
  else, Similarity_threshold = 0.8; end
  period_old  = period;
  %%% Reset the silent flag and setting other variables:
  silent_flag        = 1;
  previous_max_value  = max_value;
  max_value          = 0;
  s                  = 0;
  zero_crossed       = 0;
  min_value          = 0;
  area_ok_flag       = 0;
  if(lost_pitch)
     if(Start_segm == 1)
        MATRIX(1:5,1:5) = 0 * MATRIX(1:5,1:5);
        Start_segm = 6;
        lost_pitch=0;
     end
  end
  pitch_est   = FS/max(1,period);
  if( pitch_est > max_pitch | pitch_est < min_pitch )
     pitch_est = pitch_vec(max(1,p-1));
     period    = past_period(1);
  end
  past_period(2:end)  = past_period(1:end-1);
  past_period(1)       = period;
  period               = mean(past_period);
end
gate_vec(p)        = gate;
if(~gate)
   pitch_vec(p)   = 0;
   pitch_est      = 0;
   period         = 0;
   period_old     = 0;
else
   pitch_vec(p)   = pitch_est;
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% End of algorithm
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Resources

Software and hardware: The simulations of algorithms has been performed using MATLAB(R). Software development for the Nord Modular has been done in Codewright.

Sound recordings analyzed and used in this report:
[R1] Kristina at Clavia, singing "Fly me to the moon" by Howard
[R2] Rasmus at Clavia, playing electric bass.
[R3] Björn at Clavia, playing saxophone and acoustic guitar.
[R4] Stefan Uppgård playing acoustical piano and the Rhodes electronical piano sounds from the Nord Electro keyboard.

Other sounds analyzed during this project:
Björn . playing the acoustical guitar.
Stefan Uppgård singing, talking and playing other piano sounds from the Nord Electro keyboard.
Leo Roos whistling, singing and talking.

Bibliography and References

[1] D. Cooper and K.C. Ng
"A Monophonic Pitch Tracking Algorithm",
May 10 1994, The University of Leeds
Leeds, United Kingdom

[2] Harry.F.Olson
"Musical Engineering"
© 1952
s.203

[3] Wolfgang Hess
"Pitch Determination of Speech Signals",
1983
ISBN 3-540-11933-7
ISBN 0-387-11933-7

[4] www.instrument-online.nu
2001-11-08

[5] Judith C. Brown
"Musical Fundamental Frequency Tracking using a Pattern Recognition Method",
J. Acoust. Soc. Am 92 (3), September 1992
Massachusetts

[6] Peter M Clarkson and Henry Stark
"Signal Processing Methods for Audio, Images and Telecommunications", © 1995
ISBN 0-12-175790-0

[7] http://engineering.rowan.edu/~polikar/WAVELETS/WTtutorial.html
2001-11-18

[8] http://www.midiworld.com/basics.htm#intro
2001-11-06

[9] Ladislav O. Dolansky
"An Instantaneous Pitch-Period Indicator"
The Journal of the Acoustical Society of America, volume 27, nr 1, Jan 1955

[10] Jonas Engdegård
"Signal Processing for a Real-Time Phonetograph"
Tal, musik och hörsel KTH, Stockholm

[11] Jonas Malmkvist
"Grundtonsextraktion – en jämförande studie av algoritmer", September 1996
IR-SB-EX-9617, exjobb vid s3 KTH.
p22, **(5.2.10)**

[12] John G. Proakis and Dimitris G. Manolakis
"Digital Signal Processing. Principles, Algorithms and Applications",
IEICE TRANS. FUNDAMENTALS
vol. E77-A, No.8, August 1994 p.1404-1406
p.922 **(5.2.11)**
p.950-951 **(5.3.5)**

[13] Yegui XIAO and Yoshiaki TADOKORO
"On Pisarenko and Constrained Yule-Walker Estimators of Tone Frequency",
IEICE TRANS. FUNDAMENTALS
vol. E77-A, No.8, August 1994 p.1404-1406
(5.2.11)

[14] Håkan Hjalmarsson and Mats Bengtsson
"Collection of Computer Exercises in Adaptive Signal Processing",
2E1350, 2000-12-28, s3, KTH

[15] Peter Händel and Petr Tichavsky
"Adaptive Estimation for Periodic Signal Enhancement and Tracking",
International Journal of Adaptive Control and Signal Processing, vol.8, p.447-456 (1994)

[16] Bengtsson, Dahlqvist, Eriksson, Lundahl and Olofsson
"Automatic Note Printer",
May 27 1999, Project Course at S3
KTH, Sweden

[17] Erkan Dorken and S. Hamid Nawab
"Improved Musical Pitch Tracking Using Principal Decomposition Analysis",
In International Conference on Acoustics, Speech and Signal Processing, volume II, pages 217--220. IEEE, 1994.
Boston University, Boston, MA

[18] Martin Piszczalski and Bernard A. Galler
"Predicting Musical Pitch from Component Frequency Ratios",
Journal of the Acoustical Soceity of America
1979 Sept, 66(3)
University of Michigan, Michigan

[19] Lennart Råde and Bertil Westergren
"BETA, Mathematics Handbook for Science and Engineering",
ISBN/Studentlitteratur 91-44-25053-3
Third edition
p.369 **(5.3.5)**

[20] Judith C. Brown
"Calculation of a Constant Q Spectral Transform ",
J. Acoust. Soc. Am. {\bf 89} 1991 425-434.

[21] Judith C. Brown and Miller S. Puckette
"An efficient algorithm for the calculation of a constant Q transform",
J. Acoust. Soc. Am. 92, vol.5 2698-2701

[22] Judith C. Brown and Miller S. Puckette
"A high resolution fundamental frequency determination based on phase changes of the Fourier transform",
J. Acoust. Soc. Am. 94, vol.2 662-667

[23] J. L. Flanagan and R. M. Golden
"Phase Vocoder",
Bell Syst. Tech. J., vol 45, p 1493-1509

[24] M. R. Portnoff
"Implementation of the digital phase vocoder using the fast Fourier transform",
IEEE Trans. Acoust., Speech, Signal proc., vol. ASSP-29, p. 374-387 June 1981

[25] Carlson, Einarsson, Kim, Nyström, Renefeldt and Wadenberg
"Real Time Pitch Scaling",
May 28 2001, Project Course at S3
KTH, Sweden

[26] Yuki Tabata and Tetsuya Shimamura
"Noise Robust Pitch Extraction Based on Auto-Correlation Analysis in the Frequency Domain",
Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing, May 2-4 2001 Hong Kong
Saitama University, Japan

[27] M. Scott Andrews, Joseph Picone and Ronald D. Degroat
"Robust Pitch Determination via SVD based Cepstral Methods",
IEEE 1990, vol 2/90/ p.253-256
Saitama University, Japan

[28] Tristan Jehan
"Musical Signal Parameter Estimation"
http://cnmat.cnmat.berkeley.edu/~tristan
CNMAT, Berkeley, USA and
Université de Rennes, Rennes, France

[29] Shubha Kadambe and
G. Faye Boudreaux-Bartels
"Application of the wavelet transform for pitch detection of speech signals."
IEEE Transactions on Information Theory, vol.38, no.2, March 1992, p.917-924

[30] S.G. Mallat and S. Zhong
"Characterization of signals from multiscale edges "
IEEE Transactions of Patt. Analy. and Mach. Intell., vol.14, p.710-732, July 1992

[31] M. Scott Andrews, Joseph Picone and
Ronald D. Degroat
"Robust Pitch Determination via SVD based Cepstral Methods",
IEEE 1990, vol 2/90/ p.253-256
Saitama University, Japan

[32] Alexander Galembo
"A method for objective evaluation of timbre in the piano treble", 1979
Proc. 18th Acoustical Conf. Czechoslovakia, Sept. 10-14, 1979, p. 45-48

[33] Taeko Miwa, Yoshiaki Tadokoro and
Tsutomu Saito
"Musical Pitch Estimation and Discrimination of Musical Instruments using Comb Filters for Transcription", 1999
0-7803-5491-5/99 © 1999 IEEE

[34] Alexander Galembo and Anders Askenfelt
"Signal Representation and Estimation of Spectral Parameters by Inharmonic Comb Filters with Application to the Piano", 1999
IEEE Transactions on speech and audio processing, vol.7, no.2, March 1999

[35] Tim R. Black and Kevin D. Donohue
University of Kentucky, Electrical Engineering Department
"Pitch Determination of Music Signals Using the Generalized Spectrum", 2000
(0-7803-6312-) 4 / 00 , IEEE

[36] Francisco J. Casajus Quiros and
Pablo Fernandez-Cid Enriquez
Ciudad Universitaria, Madrid, Spain
"Real-Time, Loose-Harmonic Matching Fundamental Frequency Estimation for Musical Signals", 1994
(0-7803-1775-) 0 / 94 , 1994 IEEE

[37] Etienne Barnard, Ronald A. Cole, Mathew P. Vea and Filena A. Alleva
"Pitch detection with a Neural-Net Classifier",
IEEE Transactions on Signal Processing, vol.39, no.2, February 1991

[38] Judith C. Brown and Miller S. Puckette
"An efficient algorithm for the calculation of a constant Q transform",
J. Acoust. Soc. Am 92 (5), November 1992

[39] Johan Liljencrants
"Algorithm to find a period time, interpolated between the sampling instants",
STL – QPSR 1 / 1991, KTH

[40] T.F.Quatieri
"Discrete-Time Speech Signal Processing: Principles and Practice",
Prentice Hall Inc., 2001
s.69 **(7.6)**

[41] John G. Proakis and Masoud Salehi
"Communication Systems Engineering",
© 1994, Section 10.7.2

[42] Bagshaw, P.C.
"Automatic prosodic analysis for computer aided pronounciation teaching",
PhD Thesis 1994, University of Edinburgh

[43] Stephen C. Davies and Delores M. Etter
Dept. Of Electrical/Computer Engineering
University of Colorado
"An Adaptive Technique for Automated Recognition of Musical Tones",
Asilomar Conference 1996
IEEE © 1997 1058-6393/97