Künstliche Grundfrequenzkontur für Elektro-Larynx-Sprache in Echtzeit-Simulation

Florian Pokorny
(flopok@sbox.tugraz.at)

Bachelorarbeit

Graz, am 9. Juni 2009

Institut für Signalverarbeitung und Sprachkommunikation
Technische Universität Graz

Institutsleiter: Univ.-Prof. Dipl.-Ing. Dr.techn. Gernot Kubin

Betreuer: Dipl.-Ing. Martin Hagmüller

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel[1] nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am ....
(Unterschrift)

[1] Die vorliegende Bachelorarbeit wurde unter Verwendung von LaTeX angefertigt. Alle Abbildungen ohne Quellenangabe wurden entweder mit CorelDRAW 12 erstellt, oder aus MATLAB 7.0.4 bzw. Simulink exportiert.

Inhaltsverzeichnis

1 Einleitung

1.1 Grundlagen der Sprachproduktion

Die Erzeugung von Sprachlauten durch den menschlichen Stimmapparat setzt die Lenkung eines Luftstroms voraus. Die Lunge als Luftquelle, die Luftröhre für den Lufttransport, der Kehlkopf mit den Stimmbändern[2] als Signalquelle und der Vokaltrakt als akustisches Filter bilden die physiologischen Hauptkomponenten zur Sprachproduktion. [VM06, S.6]

Die Abbildungen 1 und 2 veranschaulichen die anatomischen Gegebenheiten.

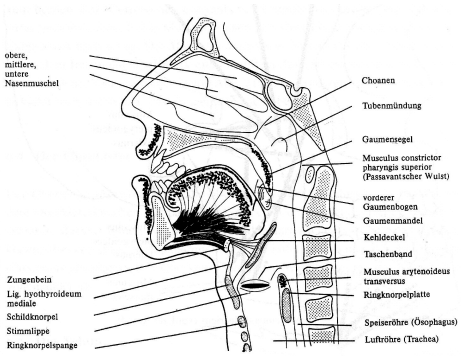

Abbildung 1: Sagittalschnitt[3] durch Schädel und Hals [PN02, S.77]

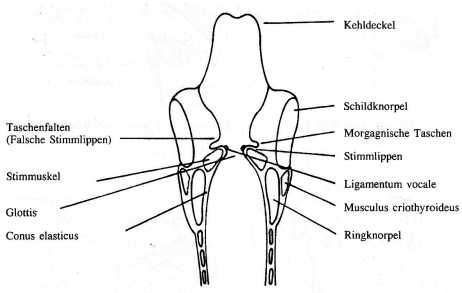

Abbildung 2: Dorsaler[4] Querschnitt durch den Kehlkopf [PN02, S.65]

Stimmhafte Laute werden erzeugt, wenn der von der Lunge stammende Luftstrom durch die Vibrationen der Stimmbänder periodisch unterbrochen wird. Am Beginn einer jeden Periode ist die Glottis, der Bereich zwischen den Stimmbändern mittels deren Spannung weitgehend geschlossen. In weiterer Folge steigt der Luftdruck unterhalb der Glottis und zwingt diese zur Öffnung. In dem Moment, in dem die Stimmbänder durch

[2] klinische Bezeichnung für die Stimmlippen [Tut81, S.556]
[3] Schnittebene parallel zur Medianebene und normal auf die Frontalebene [Tut81, S.506]
[4] von der Rückseite [Tut81, S.120]

Muskelerschlaffung auseinanderklaffen, steigt die Geschwindigkeit der Luft durch die Glottis und der lokale Luftdruck sinkt. Dieses Phänomen wird in der Strömungslehre als Bernoulli-Effekt bezeichnet. Anschließend können sich die Stimmbänder wieder spannen und der nächste Zyklus beginnt, vorausgesetzt der Luftstrom bleibt gegeben. Aufgrund der abrupten, periodischen Unterbrechungen des Luftstroms kann man für die resultierende, sehr obertonreiche Druckwelle dieses sich selbst aufrechterhaltenden Oszillators, die das Anregungssignal des Vokaltrakts darstellt, eine Grundfrequenz $f_0 = 1/T_0$ definieren, wobei T_0 die Periodendauer beschreibt. Die Obertöne werden nun entsprechend des Frequenzgangs des Vokaltrakts spektral geformt.[VM06, S.7]

Diese Art der Schallerzeugung nennt man Phonation. Die Variationsbreite der einzelnen, fungierenden Komponenten sorgt für eine große Vielseitigkeit des menschlichen Stimmapparats. So ist neben den durch ihre beweglichen Bestandteile Unterkiefer, Zunge, Gaumen und Lippen in Form und Abschluss veränderlichen Hohlräumen Rachen, Mund und Nase weiters die Stärke des Luftstroms, die sowohl Lautstärke als auch Tonhöhe des erzeugten Stimmklangs beeinflusst, weitgehend variabel. Die Tonhöhe wird außerdem von der Spannung der Stimmlippen bestimmt und kann auf diese Weise auch für verschieden starke Anblasdrücke gehalten werden.[Kei75, Kap.18 S.21]

1.2 Alternative Methoden zur Spracherzeugung

Nun ist im Fall von Kehlkopfkrebs im fortgeschrittenen Stadium die gänzliche Entfernung des Kehlkopfs meist die einzige und letzte Möglichkeit, das Fortschreiten des Krankheitsverlaufs zu stoppen. Der Verlust des herkömmlichen, menschlichen Spracherzeugungsapparats, der auf der Vibration der Stimmlippen beruht, folgt als Konsequenz. Zur Rückgewinnung der akustischen Verständigungsmöglichkeit sind betroffene Patienten auf alternative Spracherzeugungsmethoden angewiesen, von denen heute drei bevorzugt zur Anwendung kommen. [Hag07a]

Die Technik der Ruktusstimme beruht auf geschluckter Luft, die in kontrollierter Weise wieder durch die Speiseröhre ausgestoßen wird. Dadurch kommt es zur Anregung der falschen Stimmbänder, auch Taschenbänder genannt, die im Allgemeinen den Kehlkopfeingang unten abschließen [Tut81, S.555]. Sie erzeugen so den zur Spracherzeugung benötigten Wechselluftstrom. Die falschen Stimmbänder, deren viele Schleimdrüsen Elastizität und Feuchtigkeit der wahren Stimmbänder gewährleisten [HG93, S.566], spielen bei der herkömmlichen Lautbildung keine Rolle, was das Erlernen der Ruktusstimme erschwert. Allerdings ist hier der fehlende Bedarf an technischen Hilfsmitteln als Vorteil anzusehen. [Hag07a]

Der Gebrauch einer Stimmprothese basiert auf einem Ventil, das Patienten zwischen Luft- und Speiseröhre eingesetzt wird und wiederum eine Anregung der falschen Stimmbänder im Rachen durch das Ausatmen der Luft aus der Lunge ermöglicht. [Hag07a]

Neben der Ruktusstimme und der Stimmprothese repräsentiert die Verwendung eines Elektro-Larynx (EL) die dritte gängige Technik zur alternativen Spracherzeugung. Wie in Abbildung 3 zu sehen, ist ein EL ein kleines, tragbares Gerät, welches, an den Hals gehalten, den Vokaltrakt auf Knopfdruck mit seiner begrenzt einstellbaren Grundfrequenz anregt und somit den auf Lungenluftstrom und Stimmbandvibrationen basieren-

den Oszillator der natürlichen Phonation, in Variabilität stark eingeschränkt, funktionell ersetzt. Stimmhafte Laute können nun durch die gewohnten, dieses Quellsignal spektral filternden Artikulationsbewegungen erzeugt werden. Zur Bildung stimmloser Laute wird bei der EL-Sprache auf den vorhandenen Luftvorrat im Mund zurückgegriffen. [Hag07a]

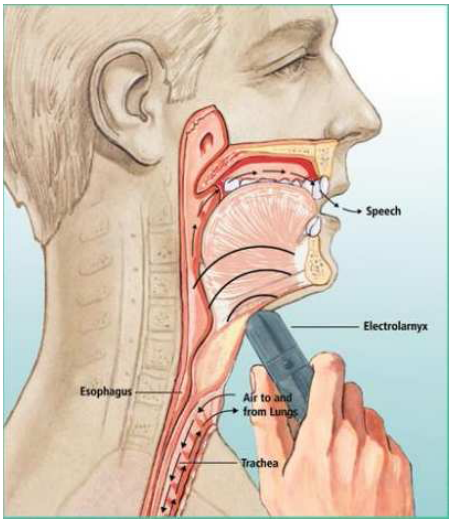

Abbildung 3: Funktionsprinzip eines EL [Hag07b]

Allerdings sorgt die konstante Anregungsfrequenz am Hals für das sehr monotone und mechanisch wirkende Klangbild der EL-Sprache [Hag07a]. Zur Steigerung der Sprachnatürlichkeit bestand die Aufgabe nun darin, mittels computergestützter Echtzeit-Simulation eine künstliche Grundfrequenzkontur für EL-Sprache in Abhängigkeit des gegenwärtig artikulierten Lauts zu berechnen und zur variablen Vokaltraktanregung zu nutzen.

2 Simulationsaufbau

Die erste Komponente des Simulationsaufbaus bildet ein Mikrofon, das über den Mikrofoneingang an einen Computer angeschlossen ist und diesem das aktuelle Lautgeschehen als Eingangssignal liefert. Das Mikrofon sollte in Mundnähe des Sprechers positioniert werden. Mit Hilfe des Computers findet dann in MATLAB-Simulink die Berechnung der momentanen, vom Eingangssignal abhängigen Frequenzkontur statt. Eine durch sie getriggerte, erzeugte Pulsfolge liefert das Ausgangssignal des Computers, das am Audioausgang anliegt. Zur Ansteuerung des, als EL-Prototyp fungierenden Mini-Schwingerregers Typ 4810 von Brüel & Kjær, in weiterer Folge auch Shaker genannt, muss die Pulsfolge verstärkt werden. Dazu wird hier der vierkanalige 4x60 Watt *Power Amplifier* PA4060 von APart verwendet. Um ein kontinuierliches Übersprechen des vom Shaker während des Schwingens abgestrahlten Störgeräuschs auf das Mikrofon zu verhindern, trennt ein selbst gebauter Drucktaster den Signalweg zwischen Computer und Verstärker. So muss der EL-Sprecher den Knopf zur Phonation konstant betätigen und kann das Ansteuern

des Shakers zum Beispiel während der Sprechpausen, gleich wie beim herkömmlichen EL durch Loslassen unterbrechen. Der beschriebene Simulationsaufbau, bestehend aus den verwendeten Hardware-Komponenten, ist in Abbildung 4 veranschaulicht.

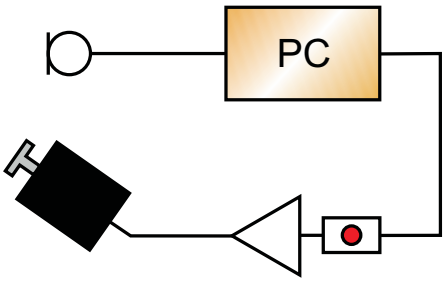

Abbildung 4: Simulationsaufbau

3 Simulink-Modell

Den wesentlichen Schritt innerhalb der Simulation bildet die computergestützte Verarbeitung des Mikrofonsignals zur Pulsfolge. Das für diese Aufgabe entwickelte MATLAB-Simulink-Modell ist über die für Windows zur Verfügung stehenden *Signal Processing Blockset I/Os From Wave Device* und *To Wave Device* mit Audio-Ein- und Ausgang des PCs verbunden und die Einbindung in die Hardware-Peripherie somit gegeben. Abbildung 5 zeigt seinen abstrahierten Aufbau.

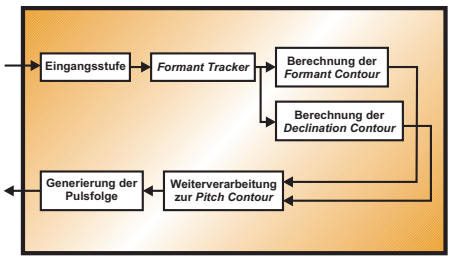

Abbildung 5: Abstrahiertes Simulink-Modell

Im Laufe der Erstellung des Modells und der Programmierung mancher darin enthaltener Funktionen wurde die *MATLAB Help* (siehe [The05]) unterstützend verwendet.

3.1 Eingangsstufe

Das Eingangssignal wird nun also über den *From Wave Device*-Block eingelesen und mit einer benutzerdefinierten Sample-Rate fs von 8000 Hz abgetastet. Die *Frame Size* beträgt 512 Samples (64 ms), die *Queue Duration* nach Möglichkeit der Soundkarte des Computers 0 s.

Die selbsterstellte und über den *MATLAB Fcn*-Block in die Simulation eingeschleuste Funktion `noise.m` vergleicht nun das gesampelte Eingangssignal mit einem einstellbaren *Threshold*. Der Programmcode ist in Listing 1 zu sehen. Wenn sich mehr als 4/5 der Samples eines Frames betragsmäßig unterhalb dieses Schwellwerts befinden, so wird das gesamte Frame auf Null gesetzt, anderenfalls unverändert durchgelassen. Ziel des Blocks ist es, dass das Grundrauschen des Mikrofons, sowie leise Stör- und Hintergrundgeräusche während der Sprechpausen für die Weiterverarbeitung unterdrückt werden und einen definierten Zustand ergeben.

Listing 1: `noise.m`

```
1  function [ out ] = noise( in )
2
3  global threshold;
4
5  noise_indices = find(abs(in) < threshold);
6
7  if length(noise_indices)>4*length(in)/5
8      out = in*0;
9
10 else
11     out = in;
12 end
```

Noch vor dem *Noise Filter* dient ein *Spectrum Scope* zur optionalen Visualisierung des Eingangsspektrums.

3.2 *Formant Tracker*

Zur Berechnung der neuen Grundfrequenzkontur für die Anregung werden hier die Formanten des Eingangssignals herangezogen. Das sind bekanntlich jene Frequenzbereiche, innerhalb derer alle Teiltöne unabhängig vom Grundton der Stimmlippen durch Resonanz verstärkt werden [Dic97, S.61], bzw. die Maxima der spektralen Einhüllenden eines Signals [Puc07, S.149]. Ihre Ermittlung erfolgt über die Methode der linearen Prädiktion.

3.2.1 Prinzip der linearen Prädiktion

In Annäherung kann die Übertragungsfunktion des Vokaltrakts für Vokale und nichtnasale, stimmhafte Konsonanten als Allpol-Übertragungsfunktion angesehen werden, der in der digitalen Realisierung ein rein rekursives Filter entspricht [Hes06b, S.1]. Die lineare Prädiktion modelliert nun ein solches Filter, welches Frequenzanteile mit hoher Energie, also spektrale Maxima, im Gegensatz zu jenen niedriger Energie, recht gut nachbildet, während es sich der Einhüllenden des Kurzzeit-Spektrums anpasst [Hes06b, S.5].

Für gewöhnlich wird mit Hilfe der Zustandsgleichung des rekursiven Filters bei gegebenem Eingangssignal $x(n)$, Filtergrad k und bekannten Filterkoeffizienten, abhängig von den vergangenen k Abtastwerten $y(n-i)$ $(i = 1 \ldots k)$, das Ausgangssignal $y(n)$ berechnet. Hier in diesem Fall ist das Filter jedoch, bis auf die Bedingung, dass es sich um ein rekursives handeln muss, unbekannt. Es findet daher eine gewichtete Mittelung

über die vergangenen k Abtastwerte statt. Aus dieser lässt sich ein neuer Abtastwert $y(n)$ des bekannten Ausgangssignals vorhersagen und die Prädiktorkoeffizienten können bestimmt werden. Allerdings verursacht dabei das ebenso unbekannte Eingangssignal $x(n)$ den Prädiktionsfehler $e(n) = x(n)$. Man bezeichnet ihn als Residualsignal. [Hes06a, S.1] Dieses Residualsignal sollte im gesamten Frequenzbereich ein annähernd flaches Spektrum aufweisen. Eine Impulsfolge, bzw. weißes Rauschen wäre ideal. Nachgebildet werden nun jene Pole der Übertragungsfunktion, die im inversen Filter die Energie des Fehlersignals weitest möglich minimieren. [Hes06b, S.5]

Für die Modellierung eines Formanten werden zwei Pole benötigt, somit ergibt sich ein Filtergrad von $k_{min} = 2$. *Anzahl der Formanten*. Das entspricht der Abtastfrequenz in kHz. Im Allgemeinen wählt man k allerdings um zwei oder drei höher als k_{min}. [Hes06b, S.5]

3.2.2 Realisierung

Die Berechnung der Prädiktorkoeffizienten für das Vokaltraktmodell erfolgt in der Simulation mit Hilfe des *Autocorrelation LPC*-Blocks (LPC – *Linear Predictive Coding* [VM06, S.265]), der im *Signal Processing Blockset* unter *Estimation* zu finden ist. Für die stabile Detektion der ersten drei Formanten beträgt die *Prediction Order* acht.

Da bei der menschlichen Sprachproduktion das durch die Stimmbandschwingungen erzeugte Anregungssignal des Vokaltrakts in der Regel mit ungefähr 6 dB pro Oktave abfällt, dem LPC-Algorithmus jedoch ein weißes Residualsignal zugrunde liegt, kann dessen Effizienz gesteigert werden, wenn das Signal nach dem *Noise Filter*, noch vor der Formantanalyse, im Bereich hoher Frequenzen um diesen Beitrag angehoben wird. Dazu dient hier ein Digitalfilter erster Ordnung mit dem zumeist üblichen *Pre-Emphasis Factor* von 0,9. [Hes06b, S.5]

Weiters sind Sprachsignale nur für relativ kurze Zeitintervalle zwischen 20 ms und 400 ms stationär. Die lineare Prädiktion muss also für entsprechend kurze Ausschnitte des Sprachsignals durchgeführt werden. [VM06, S.163] Zu diesem Zweck findet in der Simulation eine Pufferung mittels *Overlap Analysis Window* und anschließender Fensterung durch den *Window Function*-Block statt. Als Fensterform dient ein *Hamming Window*. Die Größe des Puffers beträgt 160 Samples mit einem *Overlap* von 80 Samples. Bei der Sample-Rate von 8000 Hz entspricht dies einer Fensterlänge von 10 ms.

Im Anschluss an die LPC-Analyse erfolgt die Ermittlung der Formantfrequenzen in einem *Subsystem*. Es ist in Abbildung 6 zu sehen.

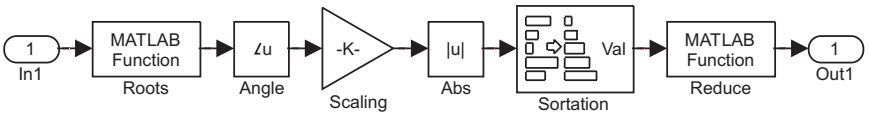

Abbildung 6: *Subsystem* zur Berechnung der Formantfrequenzen

Hier werden vorerst die, wohl gemerkt konjugiert komplexen Polstellen des vom LPC-Block in Koeffizientenform ausgegebenen Polynoms mit Hilfe der erstellten MATLAB-

Funktion `complexroots.m` berechnet, jedoch nur jene, die sich in der positiven komplexen Ebene befinden, am Ausgang ausgegeben. Listing 2 zeigt den Code.

Listing 2: `complexroots.m`

```
function [ out ] = complexroots( in )

all_roots = roots(in);
upper_half_roots = all_roots(find((angle(all_roots)>0)&(angle(all_roots)<pi)));
out = [upper_half_roots;-1*ones(length(all_roots)-length(upper_half_roots),1)];
```

Die Formantfrequenzen erhält man nun durch die Normierung der Winkel dieser Nullstellen auf die Kreisfrequenz, also durch eine Multiplikation mit $fs/(2.\pi)$. Die danach folgende Betragsbildung und aufsteigende Sortierung sorgt für die Ausgabe der ersten vier Formanten des Sprachsignals in geordneter Reihenfolge. Die Funktion `reduce.m`, in Listing 3 zu sehen, lässt für die Weiterverarbeitung nur die ersten drei passieren.

Listing 3: `reduce.m`

```
function [ out ] = reduce( in )

out = in(1:3);

if out == 4000
    out = zeros(1,length(out));
end
```

An dieser Stelle sei gesagt, dass es sich hier nicht um die bestmögliche Realisierung eines *Formant Tracker* handelt, was auf die geforderte Echtzeit-Tauglichkeit der Simulation zurückzuführen ist.

Die drei Formanten werden zur Visualisierung über *Scopes* mit Hilfe eines *Demux* aus dem Signalfluss separiert. Abbildung 7 zeigt die bisher beschriebenen, der Eingangsstufe und dem *Formant Tracker* zugehörigen Verarbeitungsstufen im Modell.

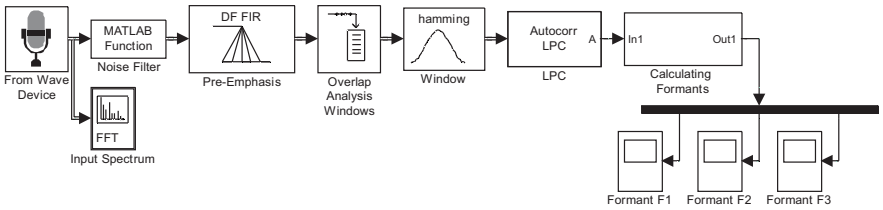

Abbildung 7: *Formant Tracker*

3.3 Berechnung der *Formant Contour*

Die Basis für die angestrebte künstliche Grundfrequenzkontur bildet jener Frequenzverlauf, der sich aus einer Linearkombination aus den ersten drei Formanten zusam-

mensetzt. Diese benutzerdefiniert einstellbare Linearkombination wird durch Matrix-Multiplikation des Ausgangssignals des *Formant Tracker* mit einem dreispaltigen Vektor realisiert. Nach einer ungewichteten Glättung des Signals mit Hilfe eines *Weighted Moving Average*-Filters über 20 Samples (200 ms) und einer Skalierung des Betrags mittels Division durch den *Scaling Factor*, erhält man nun die *Formant Contour* [Hag07a]. Der *Scaling Factor* beträgt hier 100. Die grafische Ausgabe erfolgt über ein *Scope*.

In Abbildung 8 sind die Erweiterungen des *Formant Tracker* um die Verarbeitungsblöcke zur Erzeugung der *Formant Contour* zu sehen.

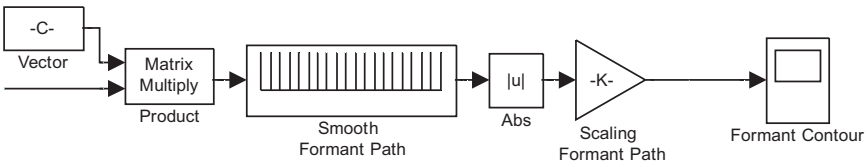

Abbildung 8: Zweig zur Berechnung der *Formant Contour*

3.4 Berechnung der *Declination Contour*

Aufgrund des nachlassenden Ausatmungsdrucks ist bei herkömmlicher menschlicher Sprache im Verlauf einer Äußerung ein exponentieller Abfall in der Grundfrequenzkontur festzustellen [Sch01, S.170]. Eine zusätzliche Verbesserung des Anpassungsversuchs der EL-Sprache an natürliche Sprache soll nun also die Einbeziehung einer *Declination Contour* erzielen. Sie wird mit Hilfe der Funktion `declination.m` erzeugt, für deren Ansteuerung das bereits über den *Demux* separierte Signal eines der drei Formanten, hier des dritten, dient. Listing 4 zeigt den Programmcode der Funktion.

Listing 4: `declination.m`

```
1  function [ out ] = declination( in )
2
3  global declination_counter;
4  global fs;
5  global output_buffer_size;
6  global buffer_overlap;
7  global current_value1;
8  global current_value2;
9  global simulation_time;
10 global zero_counter;
11 global t_zero_counter;
12
13 tau = -5/log(0.1);
14 maximum_value = 15;
15
16 if in == 0
17     zero_counter = zero_counter + 1;
18     t_zero_counter = zero_counter*1/(fs/(output_buffer_size-buffer_overlap));
19     current_value1 = current_value2 + (maximum_value - current_value2)*(1-exp(-t_zero_counter/tau));
```

```
20
21      out = 0;
22      declination_counter = 0;
23
24  else
25      declination_counter = declination_counter+1;
26      t_declination_counter = declination_counter*1/(fs/(output_buffer_size-buffer_overlap));
27      out = current_value1*exp(-t_declination_counter/tau);
28      current_value2 = current_value1*exp(-t_declination_counter/tau);
29      zero_counter = 0;
30  end
```

In Abhängigkeit davon ob am Eingang ein Signal anliegt oder nicht, sorgt die Funktion für die Berechnung einer, von einem anfangs definierten Maximalwert exponentiell abfallenden Frequenzkurve während einer Sprech-Phase und für eine exponentiell steigende Kurve während einer Sprech-Pause zur Bestimmung des neuen Startwerts für den nächsten Sprech-Einsatz. Zweitere wird im Gegensatz zur abfallenden nicht an den Ausgang übertragen, sondern stattdessen eine Folge von Nullen ausgegeben. Die Exponentialfunktionen $f(t) = f_o.e^{-t/\tau}$ für den Abfall und $f(t) = f_o.(1 - e^{-t/\tau})$ für den Anstieg liegen der Berechnung zu Grunde. Die abfallende Kurve nähert sich Null, die ansteigende hier dem Maximalwert an, wobei die Zeitkonstante τ dabei die Abfalls- und Anstiegsgeschwindigkeit festlegt.

Noch vor der Funktion findet, wie in Abbildung 9 ersichtlich, eine ungewichtete Glättung des Signals über fünf Samples (50 ms) mittels *Weighted Moving Average*-Filter statt. Ein *Scope* sorgt erneut für die Visualisierung der Kurve.

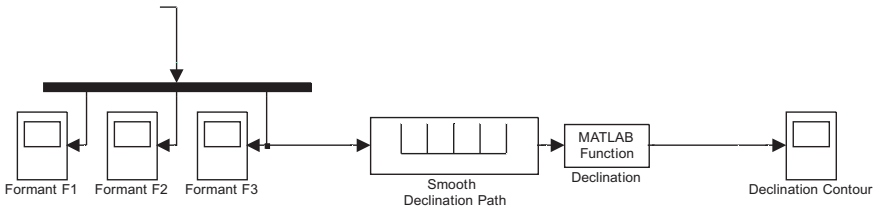

Abbildung 9: Zweig zur Berechnung der *Declination Contour*

3.5 Weiterverarbeitung zur *Pitch Contour*

Nach der Berechnung der *Formant Contour* sowie der *Declination Contour* muss nun lediglich zu deren Summe ein *Offset* addiert werden, um die endgültige *Pitch Contour* zu erhalten. An dieser Stelle kann zwischen zwei möglichen *Offset*-Modi gewählt werden. Die Umschaltung erfolgt über einen *Multiport Switch*. Der eine, nicht standardmäßig verwendete, trivialere Modus sorgt für die Addition eines konstanten, einstellbaren *Frequency Offset*. Der zweite bewirkt eine gleichmäßig, um eine beliebige Grundfrequenz ausgelenkte *Pitch Contour*. Dazu muss zur Summe aus *Formant Contour* und *Declination Contour* ein *Offset* addiert werden, der sich während der laufenden Simulation

ständig an neue Maxima und Minima der Kurve anpasst. Diese Adaption geschieht über die Funktion `automatic_offset.m`, die in Listing 5 zu sehen ist.

Listing 5: `automatic_offset.m`

```
function [ out ] = automatic_offset( in )

global save_max;
global save_min;
global offset;
global fundamental_frequency;

if in > save_max
    save_max = in;
end

if (in < save_min) && (in > 0)
    save_min = in;
end

out = fundamental_frequency - (save_max - save_min)/2 - save_min;
```

Parallel zu den Pfaden der zwei *Offset*-Modi verläuft unterhalb noch ein weiterer Zweig im Modell. In ihm wird die Summe aus *Formant Contour* und *Declination Contour* mit Null multipliziert und anschließend die eingestellte Grundfrequenz addiert. Ein *Switch*, getriggert vom separierten Signal des ersten Formanten, schaltet nun für die endgültige *Pitch Contour* diesen Zweig durch, wenn vom *Noise Filter* am Modell-Eingang Null ausgegeben wird, anderenfalls den ausgewählten der beiden oberen Pfade.

Während der Simulation ist die grafische Ausgabe der Kurve wiederum über ein *Scope* möglich. Die Verarbeitungsstufen des Signals zur Ermittlung der *Pitch Contour* aus *Formant Contour* und *Declination Contour* können in Abbildung 10 betrachtet werden.

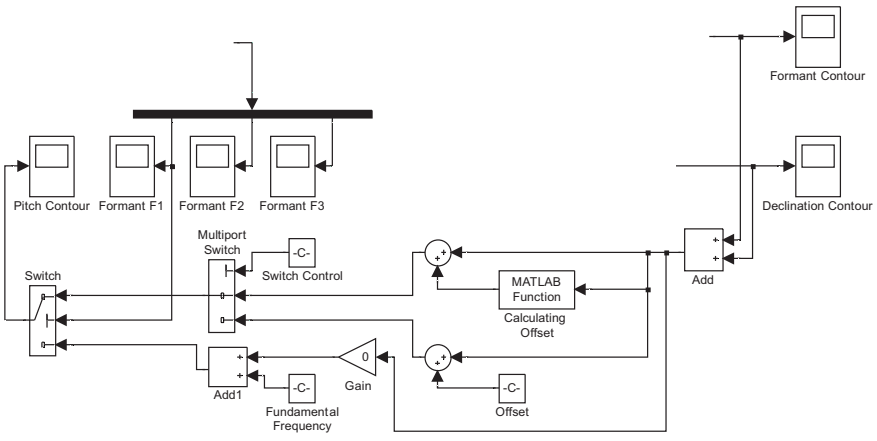

Abbildung 10: Verarbeitungsschritte zur Berechnung der *Pitch Contour*

3.6 Generierung der Pulsfolge

Der Shaker, der für die Simulation als EL-Prototyp agiert, soll nun mit einer, aus der *Pitch Contour* abgeleiteten Pulsfolge angesteuert werden. Durch Kehrwertbildung des *Pitch*-Signals ergibt sich für jeden abgetasteten Zeitpunkt die Periodendauer, die den für die Generierung der Pulsfolge entscheidenden Zeitabstand zwischen zwei Flanken darstellt. Die Erzeugung der Pulse findet innerhalb der Funktion `pulse.m` statt. Ihren Quellcode zeigt Listing 6.

Listing 6: `pulse.m`

```
function [ out ] = pulse( in )

global pulse_counter;
global fs;
global output_buffer_size;
global buffer_overlap;
global width;

if pulse_counter <= 0
    pulse_counter = round(in*fs);
    width = round(in*fs)*0.0000001;
end

pulse_counter = pulse_counter - 1;

if pulse_counter > width
    out = 0;

else
    out = 1;
end
```

Es erfolgt hier zunächst eine Umrechnung der aktuellen Periodendauer in eine Anzahl an Samples. Ausgehend von dieser Anzahl wird dann für jeden weiteren Funktionsaufruf jeweils um ein Sample rückwärts bis Null gezählt. Die Funktion gibt dabei laufend Nullen, nur während der letzten paar Samples Einsen aus. Sie bilden den Puls dieser Periode. Die Pulsbreite, also jene Anzahl an Samples, ab der beim Rückwärtszählen mit der Ausgabe der Einsen begonnen wird, kann über die Variable `width` eingestellt werden. Hier beläuft sie sich auf 0.00001% des Startwerts, also in dem, für diese EL-Simulation sinnvollen Frequenzbereich, genau ein Sample. Nach Ablauf einer Periode wiederholt sich der Vorgang für die neue, nun aktuelle Periodendauer.

Da das *Overlap Analysis Window* vor der Formantanalyse ein *Down-Sampling* des Signals auf 100 Hz bewirkt hat, die Funktion `pulse.m` jedoch zur Generierung der Pulsfolge für den rückwärtigen Zählvorgang oft genug aufgerufen werden muss, findet vorher noch ein *Up-Sampling* auf die ursprünglichen 8000 Hz des Eingangssignals statt. Die Funktion `refill.m`, deren Programmcode in Listing 7 zu sehen ist, ersetzt die durch das *Up-Sampling* entstandenen Nullen jeweils durch den letzten vorangegangenen Wert, der nicht Null entsprach.

Listing 7: `refill.m`

```
function [ out ] = refill( in )

global save_value;

if in ~= 0
    out = in;
    save_value = in;

else out = save_value;
end
```

Ein Ausgangspuffer mit der Größe von 160 Samples (20 ms), diesmal ohne *Overlap*, sowie eine *Queue Duration* im *To Wave Device*-Block von 1 s, die jedoch je nach Computer- und Soundkartenleistung nicht beliebig klein gewählt werden kann, sorgen für eine stabile Ausgabe der Pulsfolge an die Soundkarte und in weiterer Folge an den Shaker.

Abbildung 11 zeigt die beschriebenen Schritte im Modell.

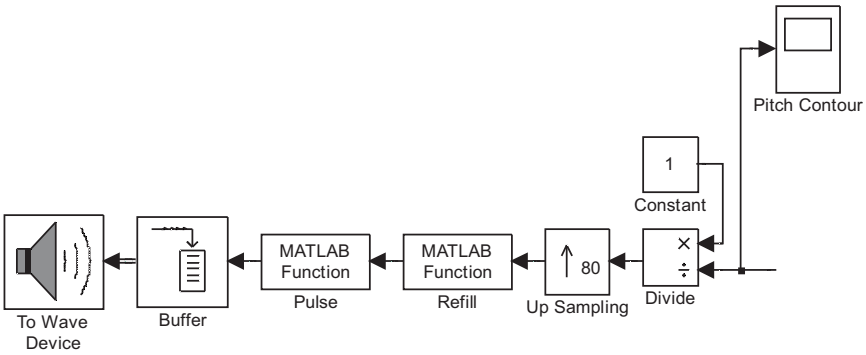

Abbildung 11: Zweig zur Generierung der Pulsfolge

Sämtliche benutzerdefiniert implementierten Einstellungsgrößen für die einzelnen Blöcke im Modell finden sich in der Datei `initial_data.mat`. Diese muss vor dem Simulationsstart ausgeführt werden, um ein Laden der Variablen in den *Workspace* zu bewirken. Hier im *Workspace*, er ist in Abbildung 12 zu sehen, könnte man die Grundeinstellungen beliebig verändern und abspeichern.

Abschließend ist in Abbildung 13 noch ein Überblick über das gesamte Simulink-Modell gegeben. Für die fehlerfreie Ausführung der Simulation, sowie auch die der im nächsten Kapitel beschriebenen grafischen Benutzeroberfläche, wird davon ausgegangen, dass sich alle in dieser Arbeit erwähnten, selbst erstellten MATLAB-Dateien in einem gemeinsamen Ordner befinden.

Name	Value	Class
buffer_overlap	80	double (global)
combination_vector	[1 1 0]	double (global)
current_value1	10	double (global)
current_value2	10	double (global)
declination_counter	0	double (global)
frame_size	512	double
fs	8000	double (global)
fundamental_freque...	100	double (global)
maximum_value	15	double (global)
offset	80	double (global)
output_buffer_size	160	double (global)
pre_emphasis_factor	0.9	double
prediction_order	8	double
pulse_counter	0	double (global)
save_max	0	double (global)
save_min	Inf	double (global)
save_value	0	double (global)
scaling_factor	100	double (global)
simulation_time	Inf	double (global)
switch_control	1	double (global)
threshold	0.01	double (global)
width	40	double (global)
zero_counter	0	double (global)

Abbildung 12: *Workspace* nach Ausführen der Initialisierungsdatei

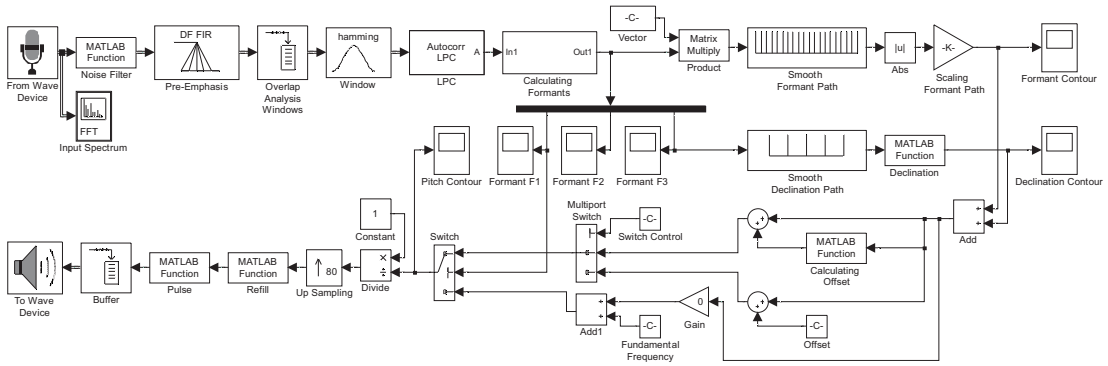

Abbildung 13: Gesamtmodell

4 Grafische Benutzeroberfläche

Zur benutzerfreundlichen Bedienung der Simulation dient ein MATLAB-GUI (GUI – *Graphical User Interface* [The05]), das wiederum unter Zuhilfenahme der *MATLAB Help* (siehe [The05]) programmiert wurde. Es kann im *Command Window* durch Ausführen der Datei `el_real_time_run.m` über den Befehl `run` gestartet werden.

Die grafische Oberfläche bietet innerhalb eines *Settings Panel* eine begrenzte Anzahl an Eingabefeldern über die der *Threshold* des *Noise Filter* zur flexiblen Anpassung der Simulation an unterschiedliche Geräuschumgebungen, die Grundfrequenz zur Berechnung der *Pitch Contour* und der *Scaling Factor* für die *Formant Contour* eingestellt werden können. Als Startwert ist der *Noise Input Threshold* auf 0.01, der *Scaling Factor* auf 100 gesetzt. Die Grundfrequenz beträgt 100 Hz. Die Wahl des gewünschten *Offset*-Modus erfolgt über *Radio Buttons*. Für den zweiten, standardmäßig nicht verwendeten,

trivialeren Modus, ist ein *Offset* von 80 Hz voreingestellt. Auch er kann jedoch über ein Eingabefeld beliebig verändert werden. Zu guter Letzt dienen drei *Sliders* als Eingabeelemente zur Festlegung der gewünschten Formant-Linearkombination, aus der sich in weiterer Folge die *Formant Contour* ergibt. Initialisiert wird dieser dreispaltige *Combination Vector* mit $(1\ -1\ 0)$. Die *Range* der *Sliders* ist durch die Extremwerte $+2$ und -2 begrenzt.

Die Simulation startet nach Betätigen des grünen *Push Button Start/Stop Simulation* und muss aufgrund der in `initial_data.mat` mit unendlich voreingestellten Simulationszeit wieder durch einen weiteren Knopfdruck angehalten werden. Während der laufenden Simulation ist ein Verändern der Einstellungen nicht möglich. Lediglich die einzelnen *Scopes* des Simulink-Modells können durch das Anhaken von *Check Boxes*, die innerhalb eines *Visualization Panel* angeordnet sind, optional angezeigt werden. Ein erneutes Anwählen einer *Check Box* schließt das entsprechende *Scope* wieder. Das *Spectrum Scope* hingegen befindet sich konstant im Vordergrund. Die Kurven der einzelnen *Scopes* bleiben auch nach dem Anhalten der Simulation bis zum nächsten Start erhalten. Das GUI selbst kann nur bei nicht laufender Simulation über den roten *Exit Push Button*, inklusive aller offenen *Scopes* geschlossen werden.

In Abbildung 14 ist das GUI bei laufender Simulation und Visualisierung aller *Scopes* veranschaulicht.

Abbildung 14: *Graphical User Interface* bei laufender Simulation

5 Evaluierung

Die Simulation bietet, vor allem aufgrund der grafischen Benutzeroberfläche, eine flexible und einfach handzuhabende Möglichkeit, künstliche Grundfrequenzkonturen für EL-Sprache variabel über die ersten drei Sprachformanten zu erzeugen und sofort auszuprobieren. Dabei können die Auswirkungen unterschiedlicher Einstellungsparameter direkt miteinander verglichen werden. Was die Linearkombination aus den Formanten anbelangt, ist festzuhalten, dass eine verstärkte Einbeziehung des ersten eine statischere *Pitch Contour* bei kleiner *Frequency Range* zur Folge hat. Der dritte Formant bewirkt im Allgemeinen Ähnliches. Ein höherer Anteil des zweiten Formanten sorgt für größere Auslenkungen und beeinflusst somit die Lebendigkeit der EL-Sprache am nachhaltigsten. Über den *Scaling Factor* kann der für die Anregung sinnvolle Frequenzbereich um ca. 100 Hz, in dem auch ein echter EL arbeitet, sehr leicht verlassen werden. Interessant ist auch der Vergleich zwischen den beiden *Offset*-Modi. Wo hingegen bei der gleichmäßig um die Grundfrequenz ausgelenkten *Pitch Contour* Sprechpassagen kurze Pausen in ihrer *Range* umschließen und somit das Anregungsgeräusch in diesen Phasen des Nicht-Sprechens in der Wahrnehmung eher in den Hintergrund drängen, können im *Manual Offset*-Modus Sprechphasen und Pausen durch entsprechende Einstellung voneinander frequenzmäßig entkoppelt werden. Eine verbesserte Anpassung der EL-Sprache an natürliche Sprache erreicht man so allerdings aufgrund der Frequenzsprünge an den Übergängen zwischen Sprechen und Nicht-Sprechen nicht gerade.

Schließlich ist noch zu erwähnen, dass der Shaker aufgrund seines metallenen Stößels und schweren Gehäuses verglichen mit einem handelsüblichen EL nicht für eine optimale Vokaltraktanregung geeignet ist. Sein Gewicht von 1,1 kg [Brü90, S.2], der separate Drucktaster und die vorausgesetzte Positionierung des Mikrofons in Mundnähe stellen ein kleines Manko für den handlichen Simulationsbetrieb dar.

In Abhängigkeit vom verwendeten Computer, sowie von dessen Soundkarte, ist für die Echtzeit-Berechnung der Pulsfolge und deren Ausgabe mit unterschiedlichen Latenzzeiten zu rechnen.

6 Ausblick

Mit Hilfe der in dieser Arbeit dokumentierten Simulation wäre es möglich, Vor- und Nachteile der über verschiedene Formantkombinationen berechneten, künstlichen Grundfrequenzkonturen für EL-Sprache auszuloten, um in Zukunft für jeden EL-Sprecher optimierte Grundeinstellungen treffen zu können. Eine wichtige Aufgabe würde auch eine Weiterführung, bzw. eine erneute Implementierung des bestehenden Berechnungsmodells für einen plattformunabhängigen Simulationsbetrieb darstellen.

Zur Verwirklichung eines alltagstauglichen EL-Prototypen mit künstlicher, variabel berechenbarer Grundfrequenzkontur, müssten alle Komponenten des Simulationsaufbaus gemeinsam in einem handlichen Gehäuse Platz finden und sämtliche Berechnungen von einem Mikroprozessor durchgeführt werden. Die genaue Abstimmung der hierfür verwendeten Bauteile aufeinander könnte dabei bestimmt eine große Rolle spielen.

Abbildungsverzeichnis

Listings

Literatur

[Brü90] Brüel & Kjær: *Technische Dokumentation: Schwingerreger Typ 4810*, 1990

[Dic97] Dickreiter, Michael: *Handbuch der Tonstudiotechnik: Band 1.* 6.Auflage. K. G. Saur Verlag KG, 1997. – ISBN 3–598–11321–8

[Hag07a] Hagmüller, Martin: Pitch Contour from Formants for Alaryngeal Speech. In: *Procedings of MAVEBA 2007*, Firenze University Press, 2007, S. 205–208

[Hag07b] Hagmüller, Martin: Review of Disordered Voice Enhancement: Engineering Approaches, 2007. – 1st COST 2103 Workshop on Advanced Voice Function Assessment, Heraklion

[Hes06a] Hess, Wolfgang: *Skript zur Veranstaltung „Sprachsignalverarbeitung“: 3.Lineare Prädiktion.* Version: 2006. `www.ifk.uni-bonn.de/lehre/magister/informationen-und-materialien-kopho/materialien-1/hess/sprachsignalverarbeitung/Kap.3.pdf`

[Hes06b] Hess, Wolfgang: *Skript zur Veranstaltung „Sprachsignalverarbeitung“: 5.Analyse der Vokaltraktparameter.* Version: 2006. `www.ifk.uni-bonn.de/lehre/magister/informationen-und-materialien-kopho/materialien-1/hess/sprachsignalverarbeitung/Kap.5.pdf`

[HG93] Hammerschmid-Gollwitzer, Josef: *Wörterbuch der medizinischen Fachausdrücke: Aktualisierte und erweiterte Neuausgabe.* Wilhelm Goldmann Verlag, 1993. – Buch-Nr. 03291 2

[Kei75] Keidel, Wolf D.: *Kurzgefasstes Lehrbuch der Physiologie.* 4.Auflage. Georg Thieme Verlag Stuttgart, 1975. – ISBN 3–13–358604–1

[PN02] Petursson, Magnús ; Neppert, Joachim M. H.: *Elementarbuch der Phonetik.* 3.Auflage. Helmut Buske Verlag, 2002. – ISBN 3–87548–318–9

[Puc07] Puckette, Miller: *The Theory an Technique of Electronic Music.* World Scientific Publishing Co. Pte. Ltd., 2007. – ISBN 981–270–077–3

[Sch01] Schmidt, Jürgen E.: *Germanistische Linguistik 157-158 2001 - Neue Wege der Intonationsforschung.* Georg Olms Verlag, 2001. – ISBN 978–3–487–11326–5

[The05] The MathWorks, Inc.: *MATLAB Help: Version 7.0.4.365 (R14) Service Pack 2*, 2005

[Tut81] Tutsch, Dagobert: *Taschenlexikon der Medizin.* 3.Auflage. Urban & Schwarzenberg, 1981. – ISBN 3–541–03013–5

[VM06] Vary, Peter ; Martin, Rainer: *Digital Speech Transmission: Enhancement, Coding and Error Concealment.* John Wiley & Sons Ltd, 2006. – ISBN 0–471–56018–9