Praktikumsprotokoll

Mikrorechentechnik 2

Versuch: INT-5 Akustischer Schalter

Betreuer: S. Kürbis

Gruppe: 17

Datum der Praktikumsdurchführung: 06.06.2023

**Teilnehmer:**

**Martin Blümel 4842445**

**Theo Röhl 4845481**

**Kleon Dingeldein 4923572**

**Justus Krenkel 4878752**

Informationen zur Abgabe des Protokolls**:**

Abgabetermin: 20.06.2023

Abgabeart: als PDF per E-Mail an: Steffen.Kuerbis [ a ] tu-dresden.de

Inhaltsverzeichnis

[1 Einführung 2](#_Toc137037084)

[2 Projektablauf 2](#_Toc137037085)

[3 Vorstellung des Programms 2](#_Toc137037086)

[3.1 Detektion 2](#_Toc137037087)

[3.2 Nulldurchgangsdichte 2](#_Toc137037088)

[3.3 Nulldurchgangsdichtehistogramm 2](#_Toc137037089)

[3.4 Main 3](#_Toc137037090)

[3.5 Einlesen der Daten 4](#_Toc137037091)

[3.6 Berechnung des nächsten Animationsschrittes 7](#_Toc137037092)

[3.7 Ausgabe des Bildes 9](#_Toc137037093)

[3.8 Benutzereingabe 10](#_Toc137037094)

[4 Probleme und Schwierigkeiten 11](#_Toc137037095)

# Einführung

Im Rahmen des MRT-Programmierpraktikums soll eine Signalverarbeitung als Problem der Objekterkennung in Sprache C erarbeitet werden. Dabei soll ein zeitdiskretes Lautsignal ausgewertet werden um die Laute „a“ und „i“ zu erkennen und zu unterscheiden. Die Aufgabe umfasst die Implementierung von drei Softwaremodulen und deren Einarbeitung in die vorhandenen Programmteile.

# Projektablauf

Um die Aufgabe zu lösen haben wir uns mit der Problematik der Objekterkennung vertraut gemacht. Danach haben wir die drei von uns zu implementierenden Aufgaben diskutiert und aufgeteilt und diese dann in Zweier-Gruppen bearbeitet. In diesen Gruppen haben wir dann die Algorithmen für die Lösung des jeweiligen Problems entwickelt. Um die Funktionalität des Codes zu testen erwies es sich als hilfreich, vorrübergehend eine main-Funktion zu programmieren. Über diese konnten die jeweiligen Funktionen mit einem Beispielsignal aufgerufen werden.

Die gefundenen Lösungsansätze wurden anschließend umgesetzt und gemeinsam besprochen.

Am Praktikumstermin wurden dann die erarbeiteten Module in vorgefertigte Dateien eingefügt und angepasst. Erst zu diesem Zeitpunkt wurden Programmfehler gefunden und behoben.

# Vorstellung des Programms

## Detektion

Die Funktion *signal\_detekt* soll aus dem aufgenommenen Signal den zur weiteren Verarbeitung relevanten Teil finden und für das Programm kenntlich machen.

Als Schnittstelle zum Hauptprogramm werden zwei Parameter verwendet:

* *short \*\*sample\_anfang*: Dieser zeigt auf den Pointer, welcher zu Beginn des Programmes auf den ersten aufgenommenen Signalwert zeigt. Nach Durchlauf der Funktion soll der Pointer auf den Beginn des relevanten Abschnitts zeigen.
* *int \*sample\_anzahl*: Übergeben wird die Adresse der Speicherzelle, welche die Länge des relevanten Abschnittes angibt. Diese soll mit dem passenden Wert überschrieben werden.

Der Programmablauf umfasst vier Schritte:

1. **Energieverlauf bestimmen:** Grundlage für die Detektion des relevanten Abschnittes ist der Energieverlauf. Dafür wird das Signal in 100 Abschnitte unterteilt und für jeden dieser Abschnitte der Energiegehalt über das Aufsummieren der Quadrierten Signalwerte bestimmt.

Im Code wird über das Array der aufgenommenen Werte iteriert und die Energie berechnet. Ist das Ende eines Abschnittes erreicht, liefert *(i+1)%laenge\_zeitintervalle=0* und der errechnete Energiewert wird im Array *energieverlauf* abgespeichert.

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

1. **Schwellwert bestimmen:** Der relevante Teil der Aufnahme beginnt, wenn der Energiewert einen einzustellenden Schwellwert überschreitet. Um diesen berechnen zu können muss zunächst der maximale Energiewert *energie\_max* im Verlauf der Aufnahme bestimmt werden. Dafür wird über das Array *energieverlauf* iteriert.

Anschließend wird der Schwellwert *schwelle* durch Multiplikation des maximalen Energiewertes mit einem einzustellenden Faktor (0<*einstellung\_schwelle*<1) ermittelt.

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

1. **Startwert bestimmen:** Aufbauend auf die vorherigen Schritte wird nun der Beginn des relevanten Abschnittes ermittelt. Dafür wird ein weiteres mal über *energieverlauf* iteriert und der erste Abschnitt ermittelt, dessen Energiegehalt den Schwellwert übertrifft.

Ist dieser gefunden, so muss der Pointer, welcher zunächst noch auf den ersten Wert der Aufnahme zeigt, auf den ersten Wert des relevanten Abschnittes geändert werden. Dieser befindet sich im Array an der Stelle *i\*laenge\_zeitintervalle*, wobei *i* den Abschnitt darstellt, ab dem die Energie den Schwellwert überschritten hat.

Für den kommenden Schritt wird die ermittelte Position im Energieverlauf-Array benötigt. Daher wird *index\_Beginn\_Signal=i* gesetzt.

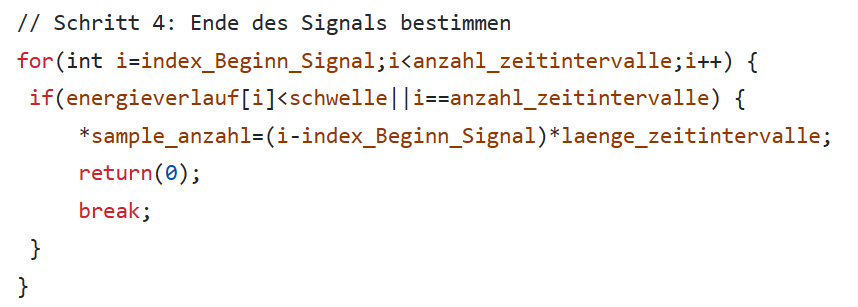
Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

1. **Ende des relevanten Abschnittes bestimmen:** Der relevante Abschnitt endet, wenn die Energie den Schwellwert wieder unterschreitet. In einer weiteren Schleife wird, beginnend von der vorherig ermittelten Position, weiter über den Zeitverlauf iteriert und nach dem Unterschreiten des Energiewertes zum Schwellenwert gesucht.

Ist dieser gefunden oder das Ende des Arrays *zeitverlauf* erreicht, wird die Anzahl der zu verarbeitenden Signalwerte errechnet. Diese ergibt sich aus der Anzahl der zusammenhängenden Abschnitte, welche einen höheren Energiewert als der Schwellwert haben (=*i-index\_Beginn\_Signal*), multipliziert mit der Länge der Zeitintervalle.

Das Signal ist verarbeitbar, wenn das Programm bis zum Ende dieses Schrittes gelaufen ist. Folglich wird der Rückgabewert auf =0 gesetzt.



## Nulldurchgangsdichte

Die Funktion ndg\_dichte bestimmt die Nulldurchgangsdichte im relevanten Signalabschnitt des originalen sowie des differenzierten Signals.

Die Schnittstelle zum Hauptprogramm sind die folgenden Parameter:

* *short \*feld\_ptr*: Übergeben wird die Adresse des Signalwertes, bei dem der zu untersuchende Abschnitt beginnt.
* unsigned int anzahl\_atw: Anzahl der Signalwerte im relevanten Signalabschnitt.
* *float \*dichte\_or*i: Übergeben wird die Adresse der Speicherzelle, in welche der Wert für die originale Dichte der Nulldurchgänge abgelegt werden soll.
* *float \*dichte\_diff*: Übergeben wird die Adresse der Speicherzelle, in welche der Wert für die originale Dichte der Nulldurchgänge abgelegt werden soll.

Das Programm durchläuft im Wesentlichen drei Schritte:

1. **Bestimmen der Nulldurchgangsdichte des Originalsignals:**

Für die Bestimmung der Dichte muss zunächst die Anzahl der Nulldurchgänge im Signalabschnitt bestimmt werden. Dafür wird über den relevanten Abschnitt des Signals iteriert und nach Nulldurchgängen gesucht.

Beispielhaft wird hier auf ein Nulldurchgang vom Positiven ins Negative eingegangen:

Der Nulldurchgang ist erfolgt, wenn ein Signalwert zunächst größer als ein einzustellender Schwellwert *schwelle* ist. Anschließend wird weiter über den Signalabschnitt iteriert und nach einem Wert unterhalb der Schwelle gesucht. Ist dieser gefunden hat ein Nulldurchgang stattgefunden. Die absolute Anzahl der Nulldurchgänge ist zunächst in der Speicherzelle der originalen Nulldurchgangdichte abgelegt und um Eins erhöht.

Ein Bild, das Text, Schrift, Handschrift, Screenshot enthält.

Automatisch generierte Beschreibung

Ein Nulldurchgang in die andere Richtung wird analog behandelt, jedoch mit umgekehrten Ungleichheitszeichen in den Bedingungen.

Vor allem die ersten Signalwerte können betragsmäßig unterhalb der Schwelle liegen. Für diesen Fall muss der Schleifenindex *i* um +1 erhöht werden, damit das Programm fehlerfrei ablaufen kann.

Nach Durchlauf des Signalabschnittes wird die relative Dichte bestimm. Dafür werden die gezählten Nulldurchgänge durch die Anzahl der zu verarbeitenden Signalwerte geteilt.

1. **Bestimmen der Nulldurchgangsdichte des differenzierten Signals**

Der Ablauf ist hier analog zur Bestimmung der Nulldurchgangsdichte des originalen Signals. Unterschiedlich sind nur die zu verarbeitende Werte. Da das Signal differenziert betrachtet werden soll, muss jeweils vom Wert des *i*-ten Signales der (*i*-*1*)-te Wert abgezogen werden. Daher muss der Schleifenindex *i* zu Beginn auf =1 gesetzt werden.

1. **Rückgabe des Funktionswertes an das Hauptprogramm**

Falls das aufgenommene Signal sehr schwach ist, kann es dazu kommen dass keiner der Signalwerte den Schwellwert betragsmäßig überschreitet. Folglich ist das Signal nicht verarbeitbar und die Funktion liefert =1 als Rückgabewert. Andernfalls wird =0 zurückgegeben.

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

Automatisch generierte Beschreibung

## Nulldurchgangsdichtehistogramm

Die Funktion ndg\_histogramm baut auf der Logik der Funktion ndg\_dichte auf, unterscheidet sich aber dahingehend, dass nicht die Anzahl der Nulldurchgänge, sondern die Signallänge bzw. Anzahl der Abtastwerte zwischen zwei Nulldurchgängen gezählt wird und in entsprechenden Histogrammkanälen aufsummiert wird. Die Funktion benutzt folgende Parameter im Aufruf:

* *short \*feld\_ptr*: Übergeben wird die Adresse des Signalwertes, bei dem der zu untersuchende Abschnitt beginnt.
* unsigned int anzahl\_atw: Anzahl der Signalwerte im relevanten Signalabschnitt.
* *float \*hist\_ori*: Zeiger auf ein Feld von float-Werten, in das die Histogrammwerte der originalen Zeitfunktion geschrieben werden sollen.
* *float \*hist\_diff*: Zeiger auf ein Feld von float-Werten, in das die Histogrammwerte der differenzierten Zeitfunktion geschrieben werden sollen.

Die Logik lässt sich exemplarisch für alle Nulldurchgänge (positiv oder negativ), sowie für das originale und das differentiale Signal beschreiben:

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

Zuerst wird geprüft, ob der Schwellwert erreicht ist (bei negativen Nulldurchgängen selbstverständlich unterschritten). Ist dies der Fall, dann wird die „Startposition“ des Intervalls gesetzt **pos\_1 = i;**

Anschließend wird mit einer while-Abfrage so lange hochgezählt bis der Schwellwert nicht mehr erreicht wird. Dann wird die „Stopposition“

**pos\_2 = i;** auf dem aktuellen Zählerwert gesetzt. Anschließend wird die Differenz **laenge = pos\_2 - pos\_1;** der beider Positionen errechnet und in den passenden Histogrammkanal zugeordnet.

Sollte es passieren, dass keine Werte in den Kanälen erfasst wurden, wird ein return-Wert 1 an die Hauptfunktion zurückgegeben, der meldet, dass keine Werte erfasst werden konnten. Andernfalls wird der Wert 0 zurückgegeben. Dieser Signalisiert dem Programm, dass alles funktioniert hat.

Leider konnte die Funktion nicht im Zeitrahmen des Praktikums implementiert und getestet werden, sodass die Beschreibung oben nur die Logik wiederspiegelt aber nicht auf Funktionalität geprüft werden konnte.

# Probleme und Schwierigkeiten

* Finden des passenden Variablentyps für die Energieverlaufsberechnung um Speicherüberlauf bei zunächst verwendeten Integer Variablen zu vermeiden
* Anpassungen der Verpointerung für Adressübergabe des Startpunktes vom auszuwertenden Signalabschnitt
* Setzen eines Schwellwertes mithilfe der Betragsfunktion aus stdlib.h ergab negative Ergebnisse (Lösung durch Entfernen der Funktion möglich da beide Faktoren positiv sind (energie\_max und einstellung\_schwelle)
* Printf (aus stdio.h) Funktion lieferte andere Werte als der Debugger, wodurch die Fehlersuche und Behebung sehr mühsam verlief
* Einarbeitung in die Programmierung mit Pointern

**To-dos:**

* **Fehlermeldungen**
* **Alles auf float geändert?**