Query By Humming

Murat Uzun, Ronja Deisel

Aufbau

- Einführung
- Methoden
  - String
    - String-Alignment
    - Hidden Markov Modelle
  - Tonal
- Fazit

Einführung

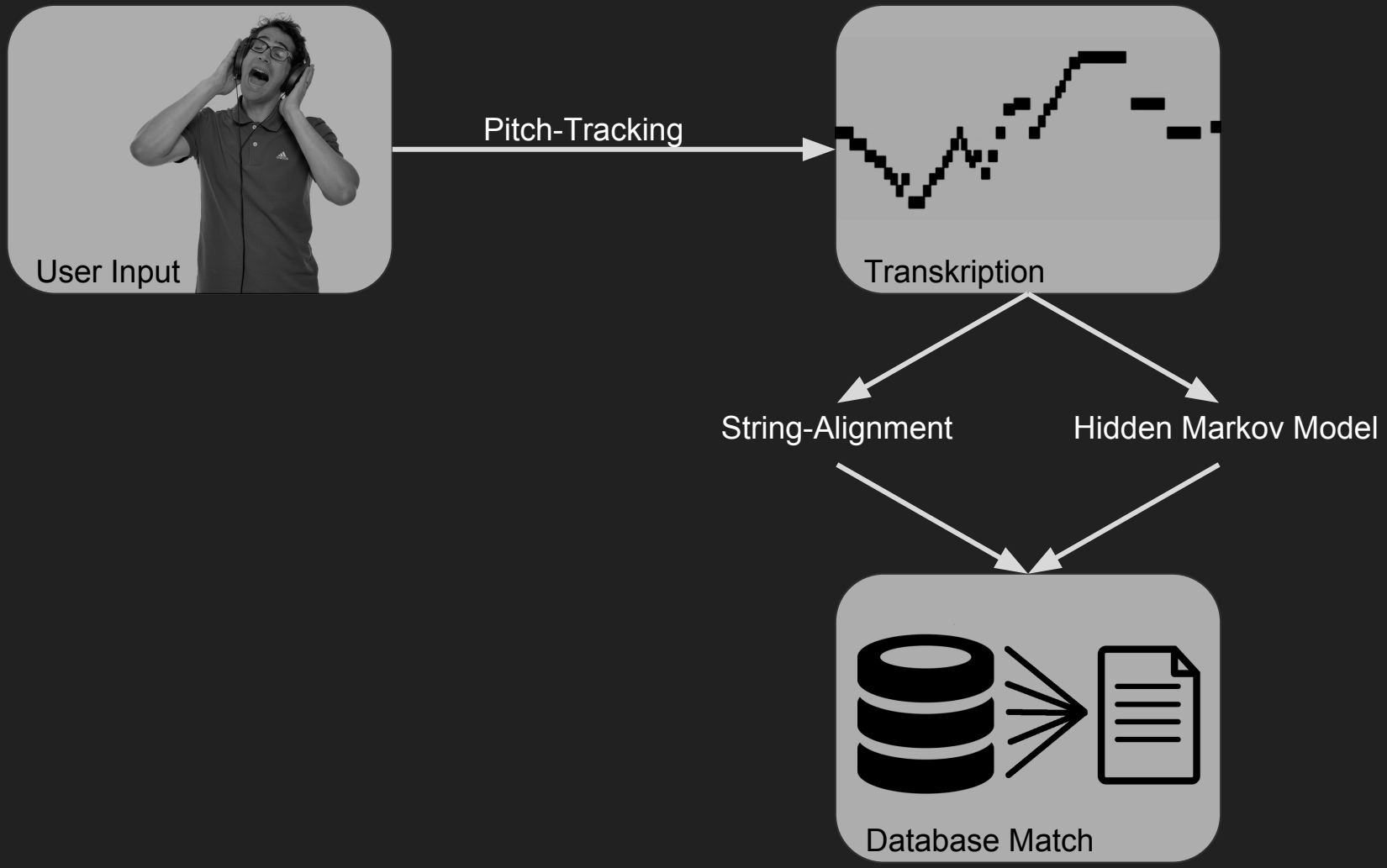
- Content-Based Ansatz
- Datenbanken mit Motiven/Themen/Songs
- Spezialfall von Cover Song Identifikation
- Motivation:
  - Intuitive Query durch Singen/Summen
  - → große Nachfrage durch Endnutzer
  - → wachsendes Angebot

Methoden

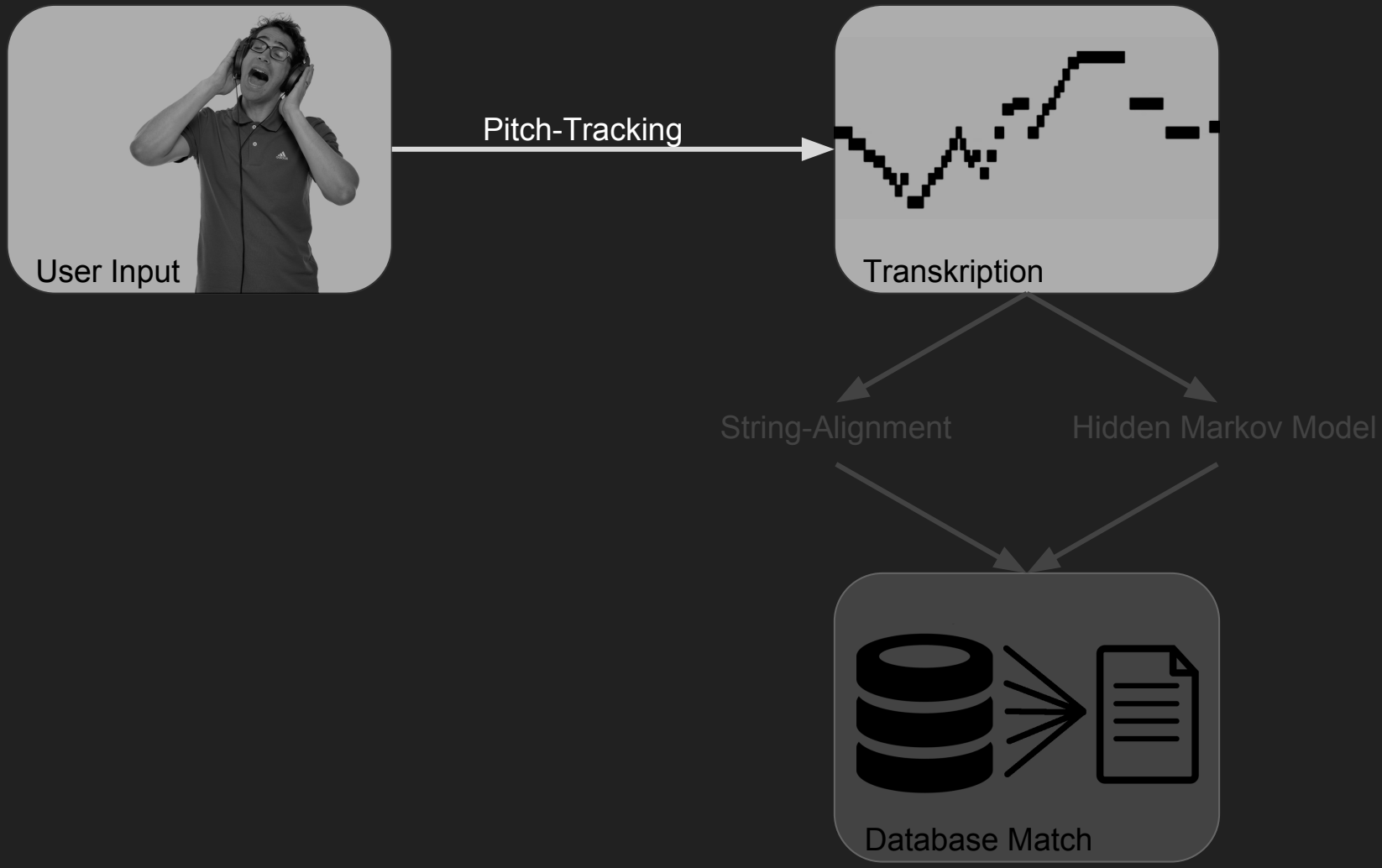
- String-Approach:
  - Interpretation von Query und Target als String
  - Suche nach bester Übereinstimmung von Query und Target
- Tonal-Approach:
  - Extraktion von Deskriptoren aus Query und Target
  - Vergleich von Kennwerten der Deskriptoren

String-Approach

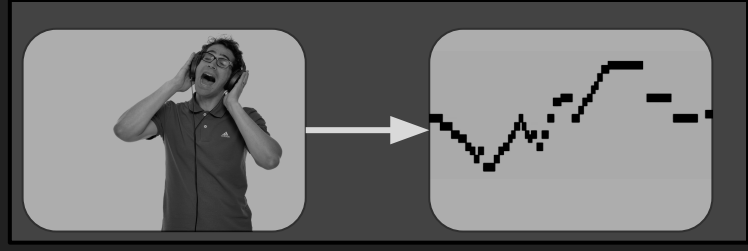
String-Approach



String-Approach: Transkription

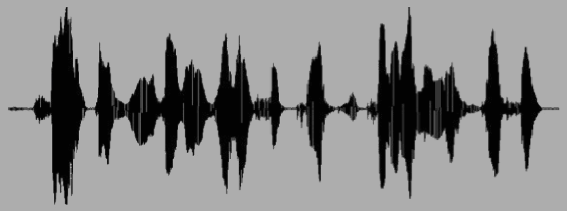


Transkription



Schallwellen

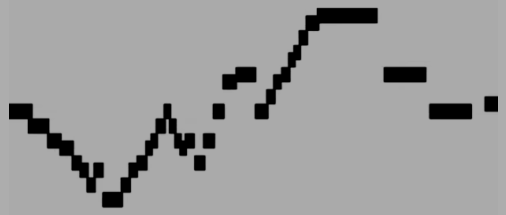
.wav



Pitch-Tracking

Noten

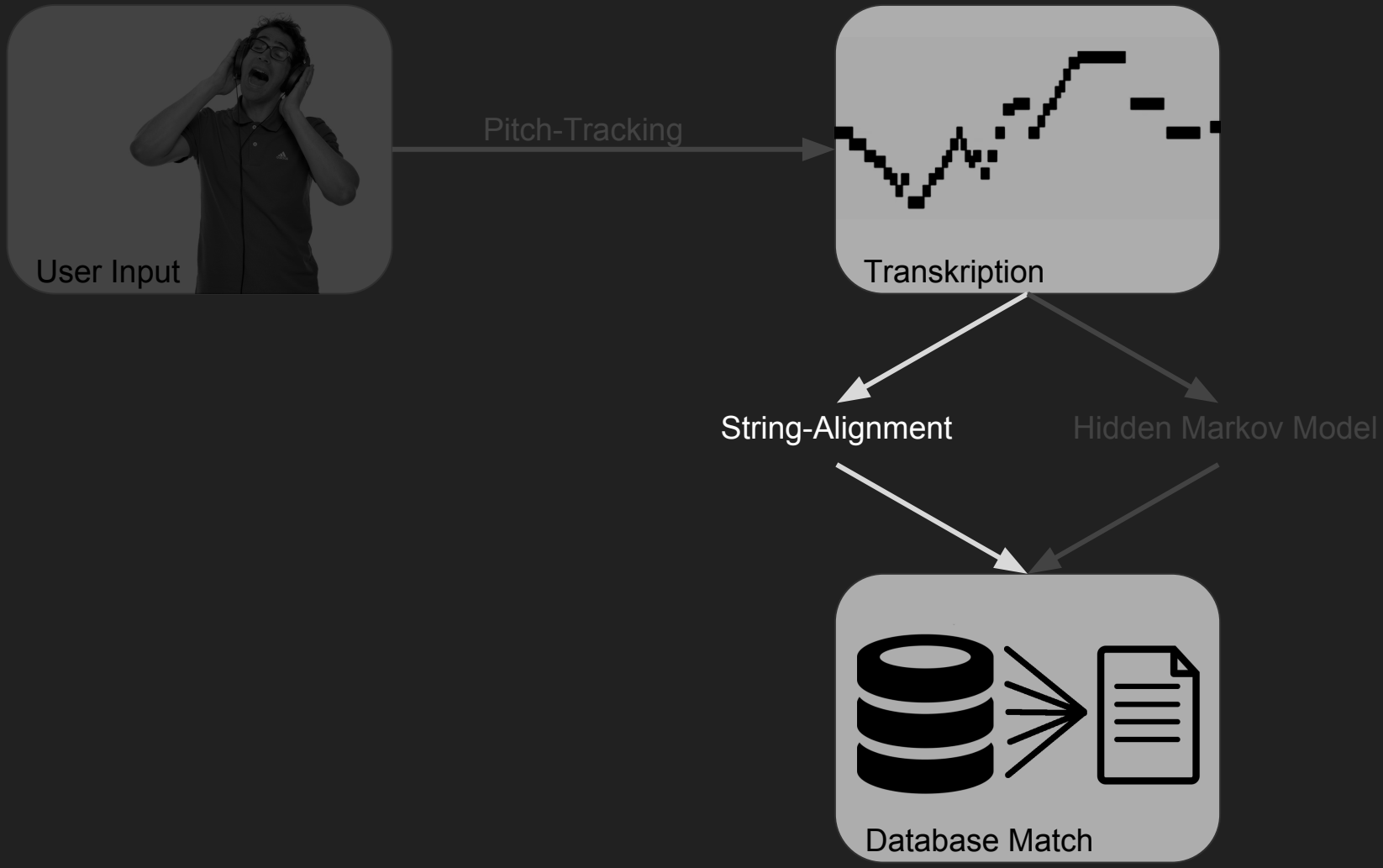
.midi



Liste von Notenübergängen

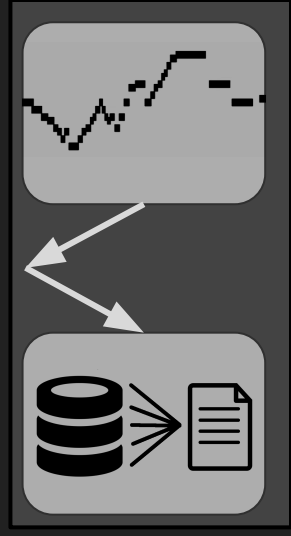
Pitch Interval	2	2	0	...
IOI Ratio	2.75	1	1	...

String-Approach: String-Alignment



String-Approach

String-Alignment



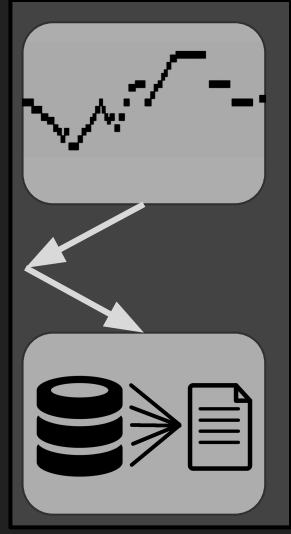
- Alphabet: Tonübergänge
- Query String: Q
- Menge von Target Strings: $\{T_1,...,T_n\}$
- Alignment Algorithmus: Global/Local Alignment Algorithmus

➔ Anwenden des Algorithmus auf alle (Q, T_i)
➔ Ranking nach 1/Rückgabewert ~ Similarity

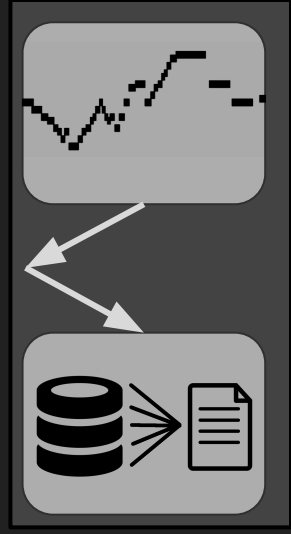
Global Alignment Algorithmus

Align Score	Target	α	γ	ε	ε
Query					
α					
β					
γ					
δ					
ε					

- Matrix: Score des besten Alignments zwischen Q und T



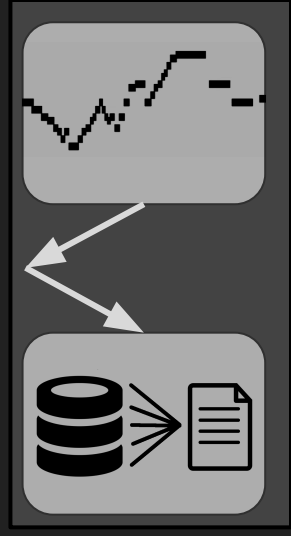
Global Alignment Algorithmus



Align Score	Target	α	γ	ε	ε
Query	0	-∞	-∞	-∞	-∞
α	-∞				
β	-∞				
γ	-∞				
δ	-∞				
ε	-∞				

- Matrix: Score des besten Alignments zwischen Q und T
- Initialisieren

Global Alignment Algorithmus

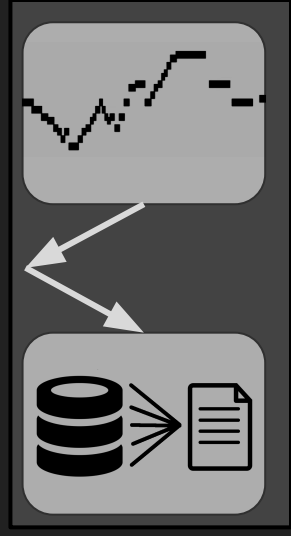


Align Score	Target	α	γ	ε	ε
Query	0	-∞	-∞	-∞	-∞
α	-∞	2			
β	-∞				
γ	-∞				
δ	-∞				
ε	-∞				

- Matrix: Score des besten Alignments zwischen Q und T
- Initialisieren
- Score(i,j) = max(
  - Score(i-1,j-1) + matchScore(Q_i,T_j)
  - Score(i-1,j) - $skipPenalty_{Target}(T_j)$
  - Score(i,j-1) - $skipPenalty_{Query}(Q_i)$

)
- matchScore(A,B) =
  - 2 , if A == B
  - - 2 , otherwise
- $skipPenalty_{Target}(A)$= $skipPenalty_{Query}(A) = 1$

Global Alignment Algorithmus

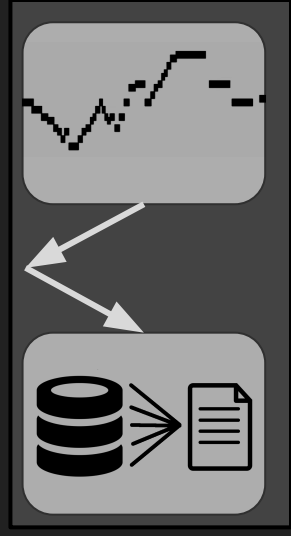


Align Score	Target	α	γ	ε	ε
Query	0	-∞	-∞	-∞	-∞
α	-∞	2 →	1		
β	-∞	1			
γ	-∞				
δ	-∞				
ε	-∞				

- Matrix: Score des besten Alignments zwischen Q und T
- Initialisieren
- Score(i,j) = max(
  - Score(i-1,j-1) + matchScore(Q_i,T_j)
  - Score(i-1,j) - $skipPenalty_{Target}(T_j)$
  - Score(i,j-1) - $skipPenalty_{Query}(Q_i)$

)
- matchScore(A,B) =
  - 2 , if A == B
  - - 2 , otherwise
- $skipPenalty_{Target}(A)$= $skipPenalty_{Query}(A) = 1$

Global Alignment Algorithmus

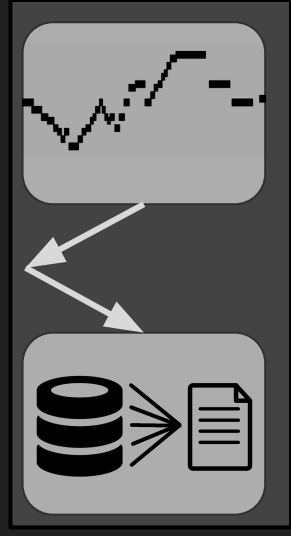


Align Score	Target	α	γ	ε	ε
Query	0	-∞	-∞	-∞	-∞
α	-∞	2	1	0	
β	-∞	1	0		
γ	-∞	0			
δ	-∞				
ε	-∞				

- Matrix: Score des besten Alignments zwischen Q und T
- Initialisieren
- Score(i,j) = max(
  - Score(i-1,j-1) + matchScore(Q_i,T_j)
  - Score(i-1,j) - $skipPenalty_{Target}(T_j)$
  - Score(i,j-1) - $skipPenalty_{Query}(Q_i)$

)
- matchScore(A,B) =
  - 2 , if A == B
  - - 2 , otherwise
- $skipPenalty_{Target}(A)$= $skipPenalty_{Query}(A) = 1$

Global Alignment Algorithmus

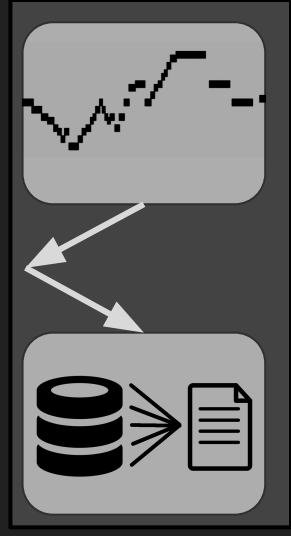


Align Score	Target	α	γ	ε	ε
Query	0	-∞	-∞	-∞	-∞
α	-∞	2	1	0	-1
β	-∞	1	0	-1	
γ	-∞	0	3		
δ	-∞	-1			
ε	-∞				

- Matrix: Score des besten Alignments zwischen Q und T
- Initialisieren
- Score(i,j) = max(
  - Score(i-1,j-1) + matchScore(Q_i,T_j)
  - Score(i-1,j) - $skipPenalty_{Target}(T_j)$
  - Score(i,j-1) - $skipPenalty_{Query}(Q_i)$

)
- matchScore(A,B) =
  - 2 , if A == B
  - - 2 , otherwise
- $skipPenalty_{Target}(A)$= $skipPenalty_{Query}(A) = 1$

Global Alignment Algorithmus

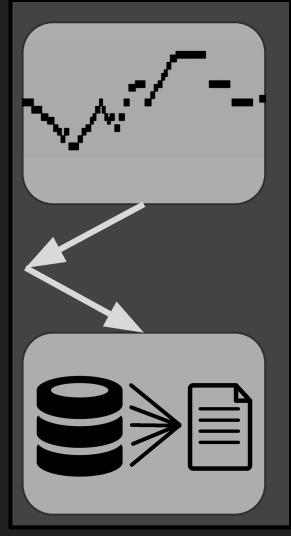


Align Score	Target	α	γ	ε	ε
Query	0	-∞	-∞	-∞	-∞
α	-∞	2	1	0	-1
β	-∞	1	0	-1	-2
γ	-∞	0	3	2	
δ	-∞	-1	2		
ε	-∞	-2			

- Matrix: Score des besten Alignments zwischen Q und T
- Initialisieren
- Score(i,j) = max(
  - Score(i-1,j-1) + matchScore(Q_i,T_j)
  - Score(i-1,j) - $skipPenalty_{Target}(T_j)$
  - Score(i,j-1) - $skipPenalty_{Query}(Q_i)$

)
- matchScore(A,B) =
  - 2 , if A == B
  - - 2 , otherwise
- $skipPenalty_{Target}(A)$= $skipPenalty_{Query}(A) = 1$

Global Alignment Algorithmus

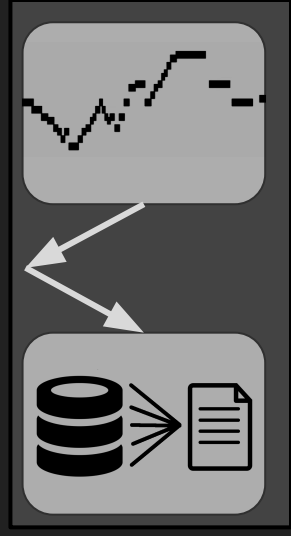


Align Score	Target	α	γ	ε	ε
Query	0	-∞	-∞	-∞	-∞
α	-∞	2	1	0	-1
β	-∞	1	0	-1	-2
γ	-∞	0	3	2 →	1
δ	-∞	-1	2 →	1	
ε	-∞	-2	1		

- Matrix: Score des besten Alignments zwischen Q und T
- Initialisieren
- Score(i,j) = max(
  - Score(i-1,j-1) + matchScore(Q_i,T_j)
  - Score(i-1,j) - $skipPenalty_{Target}(T_j)$
  - Score(i,j-1) - $skipPenalty_{Query}(Q_i)$

)
- matchScore(A,B) =
  - 2 , if A == B
  - - 2 , otherwise
- $skipPenalty_{Target}(A)$= $skipPenalty_{Query}(A)$ = 1

Global Alignment Algorithmus

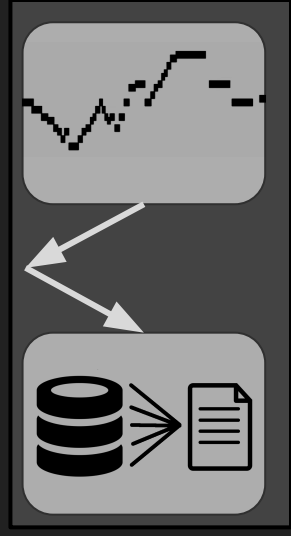


Align Score	Target	α	γ	ε	ε
Query	0	-∞	-∞	-∞	-∞
α	-∞	2	1	0	-1
β	-∞	1	0	-1	-2
γ	-∞	0	3	2	1
δ	-∞	-1	2	1	0
ε	-∞	-2	1	4	

- Matrix: Score des besten Alignments zwischen Q und T
- Initialisieren
- Score(i,j) = max(
  - Score(i-1,j-1) + matchScore(Q_i,T_j)
  - Score(i-1,j) - $skipPenalty_{Target}(T_j)$
  - Score(i,j-1) - $skipPenalty_{Query}(Q_i)$

)
- matchScore(A,B) =
  - 2 , if A == B
  - - 2 , otherwise
- $skipPenalty_{Target}(A)$= $skipPenalty_{Query}(A) = 1$

Global Alignment Algorithmus



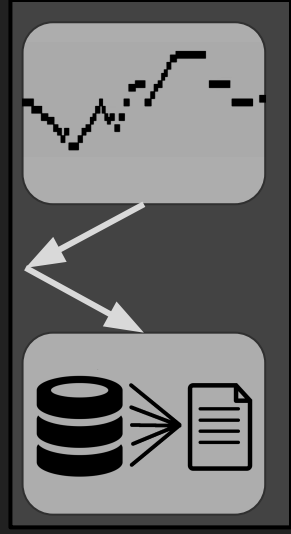
Align Score	Target	α	γ	ε	ε
Query	0	-∞	-∞	-∞	-∞
α	-∞	2	1	0	-1
β	-∞	1	0	-1	-2
γ	-∞	0	3	2	1
δ	-∞	-1	2	1	0
ε	-∞	-2	1	4	**3**

- Matrix: Score des besten Alignments zwischen Q und T
- Initialisieren
- Score(i,j) = max(
  - Score(i-1,j-1) + matchScore(Q_i,T_j)
  - Score(i-1,j) - $skipPenalty_{Target}(T_j)$
  - Score(i,j-1) - $skipPenalty_{Query}(Q_i)$

)
- matchScore(A,B) =
  - 2 , if A == B
  - - 2 , otherwise
- $skipPenalty_{Target}(A)$= $skipPenalty_{Query}(A) = 1$

String-Approach: String-Alignment

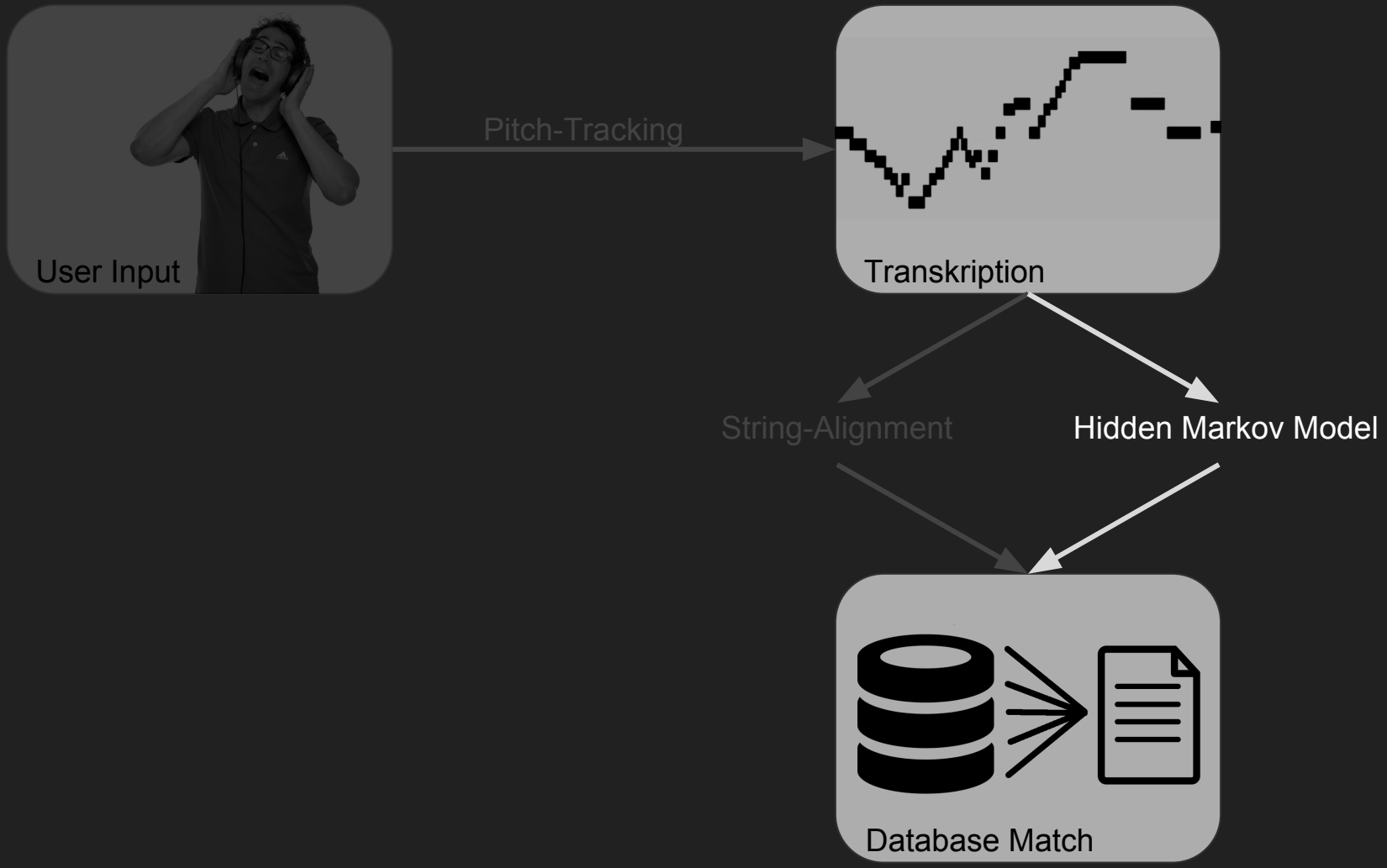
Optimierungen für QbH



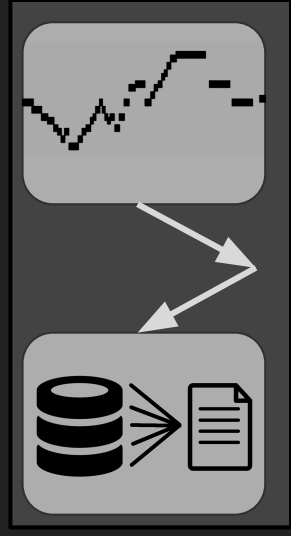
- Problem: unvollständige Queries
- → Lokaler Alignment Algorithmus

- Problem: Systematische Fehler in Transkription
  - Fehler des Pitch-Tracker
    - Oktavierungen
    - Tracken von Obertönen
    - Halbton Fehler durch Quantifizierung
  - Benutzer-Fehler
- → Berücksichtigung systematischer Fehler in matchScore:
  - $matchScore(Q_i,T_j) = \log(P_{match}(Q_i,T_j)/P(Q_i,T_j))$
- → Ermittlung der Wahrscheinlichkeiten durch Nachschlagen in generierter Wahrscheinlichkeitsmatrix

String-Approach: Hidden Markov Modelle

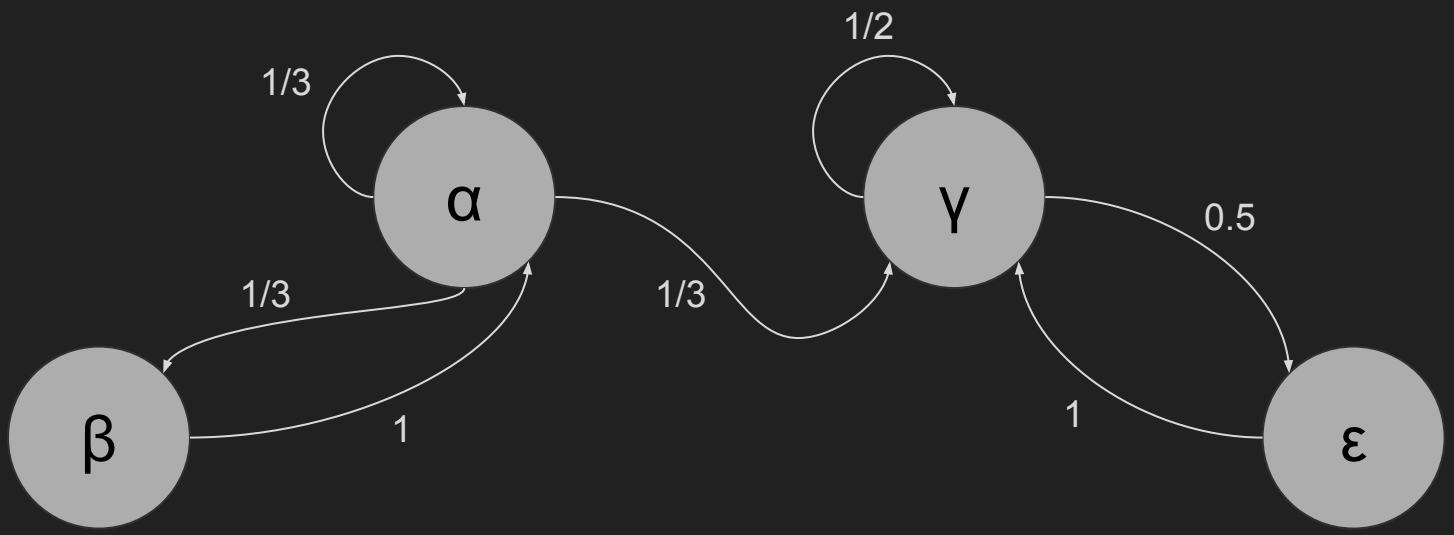


Targets als Markow Ketten

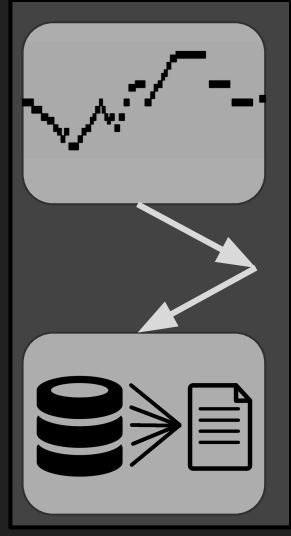


- Generierung von Markow Ketten aus MIDI-Dateien der Targets:
- Zustände (hier: Tonübergänge) $S = \{s_1, ..., s_n\}$
- Übergangswahrscheinlichkeiten $t_{ij} = P(s_i, s_j)$

- Problem: Zero Probability Sequenzen
- → Low Probability Übergänge

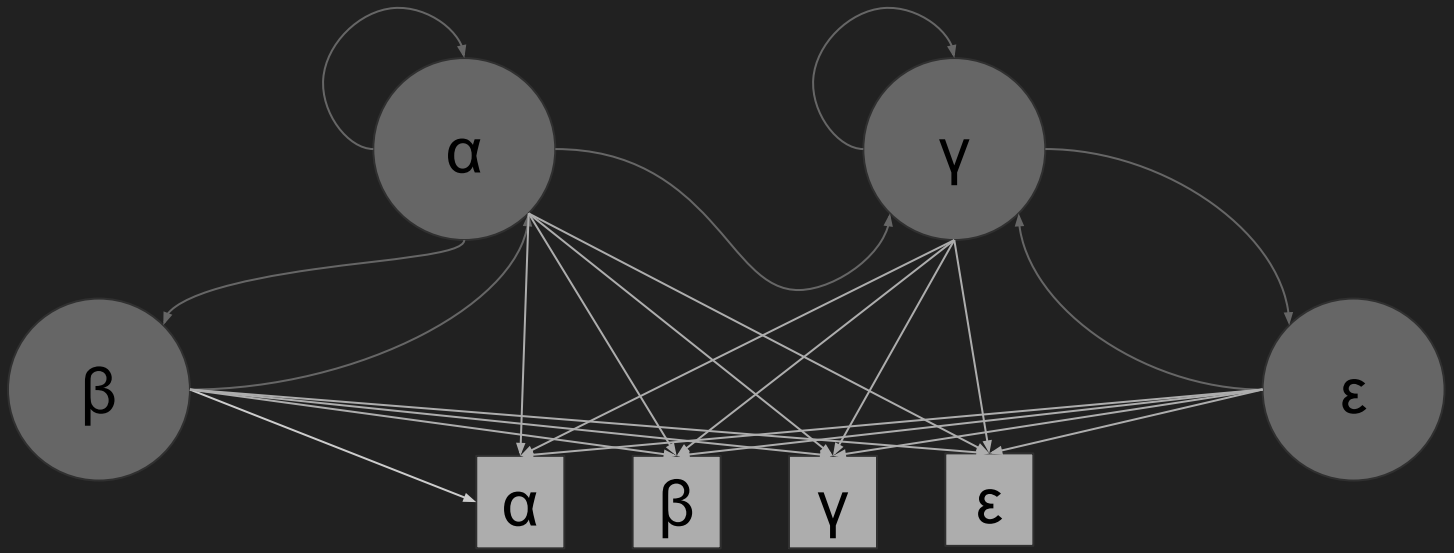


Erweiterung als Hidden Markov Modelle

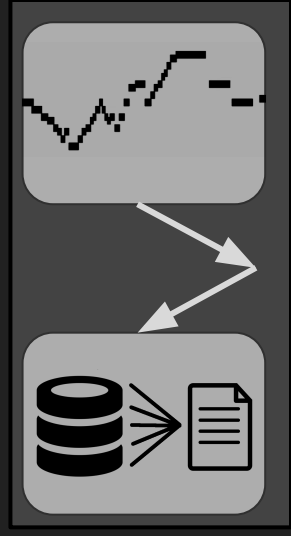


- Anpassungen:
  - Versteckte Zustände
  - Mögliche Beobachtungen: A = $\{o_1, ..., o_n\}$
  - Wahrscheinlichkeiten o_i in Zustand s_j zu beobachten

- Ranking nach Ergebnissen des Forward Algorithmus:
  - Dynamische Filter-Methode
  - Berechnet Wahrscheinlichkeit Sequenz von Beobachtungen durch HMM zu generieren



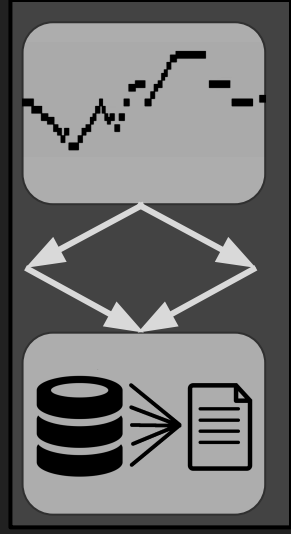
Optimierungen für QbH: Generierung einer Wahrscheinlichkeitsmatrix



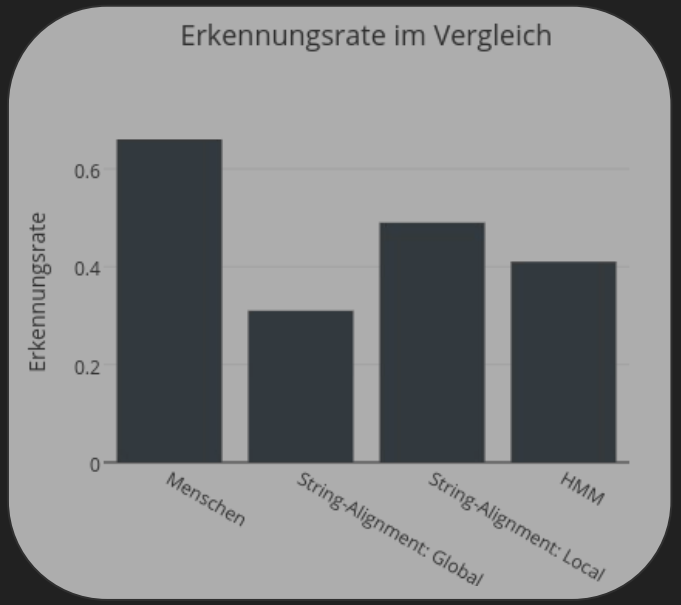
- Generierung durch Experiment:
  - $P_{sj}(o_i) = count(o_i, s_j)/count(s_j)$
- Problem: Größe der Wahrscheinlichkeitsmatrix
  - |IOI Ratio| = 5
  - |Pitch Interval| = 25

  } 125 Zustände je 125 Beobachtungen } 125^2 Wahrscheinlichkeiten
  - → Annahme von stochastischer Unabhängigkeit:
    - 2 Matrizen mit Mächtigkeit 25^2 und 5^2
    - $P_s(o)=P_{pitchInterval\ s}(pitchInterval_o)$ x $P_{IOIRatio\ s}(IOIRatio_o)$
- Ermittlung der Wahrscheinlichkeitswerte:
  - Auswahl von repräsentativen Sängern
  - Nachsingen von repräsentativen Intervallen und Rhythmen
  - Update beider Wahrscheinlichkeitstabelle

Experimenteller Vergleich



- Setup:
  - Library von Beatles Songs
  - Sets of Test Queries gesungen von 3 Personen
- Ergebnisse:
  - String Alignment performanter und zuverlässiger
  - Generalisierte Wahrscheinlichkeitsmatrizen besser als personalisierte
  - String-Alignment:
    - Brute Force Suche nach Skip Penalty
    - Local Algorithmus besser als Global
  - HMM:
    - Rhythmische Informationen oft nachteilig



Tonal-Approach

Tonal-Approach

- Identifikation einer Version
- Berechnung von Deskriptoren (Schlagwort) durch Benutzung von modernen Algorithmen
- Deskriptoren werden verwendet, um verschiedene Versionen desselben Musikstücks wiederzugewinnen

Tonal-Approach

- Genauigkeit der Deskriptoren evaluieren
- Möglichkeit einer Deskriptorfusion
- Erhöhung der Versionerkennungsgenauigkeit durch Kombination

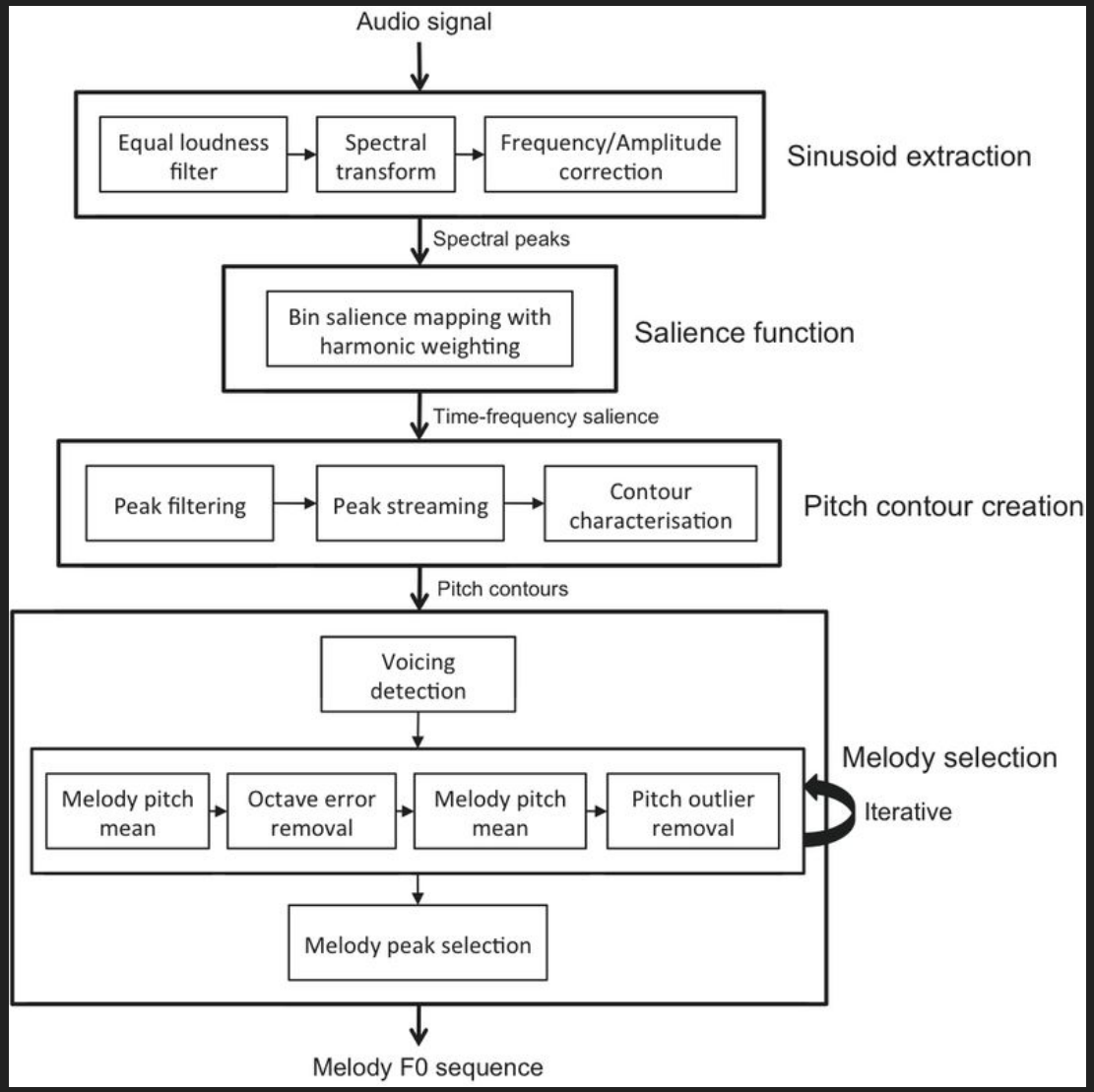
Arten von Deskriptoren

- Melodie
- Bassline
- Harmonie

Melodie

- Verwendung von modernen Melodie-Extraktionsalgorithmen
- Ziel: Erzeugung einer Folge von Frequenz-Werten entsprechend der Tonhöhe der Melodie
- Keine Quellentrennung
- Erzielte in der MIREX-Evaluationskampagne höchste mittlere Gesamtgenauigkeit

Melodie

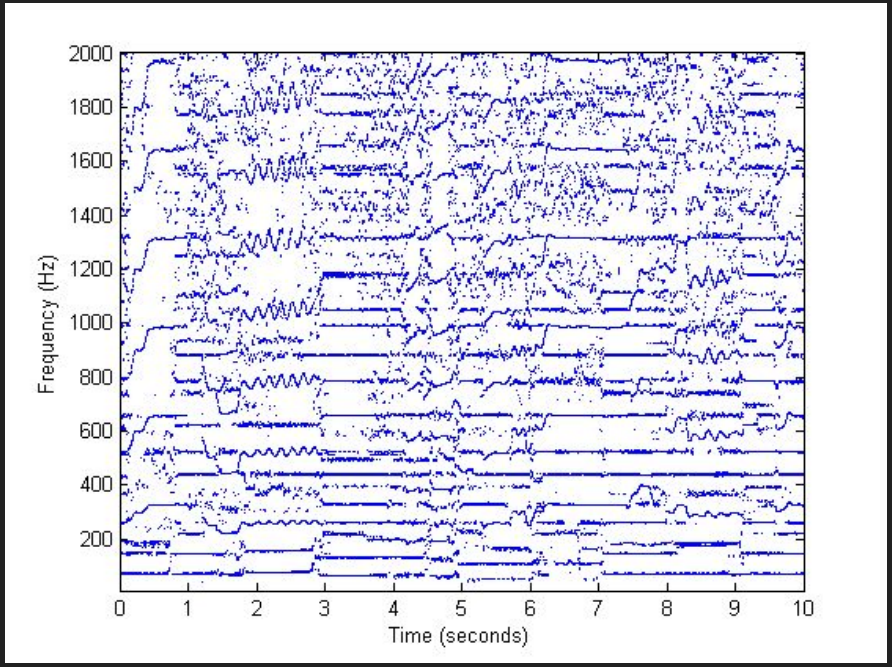

Audio signal
Equal loudness filter
Spectral transform
Frequency/Amplitude correction
Sinusoid extraction
Spectral peaks
Bin salience mapping with harmonic weighting
Salience function
Time-frequency salience
Peak filtering
Peak streaming
Contour characterisation
Pitch contour creation
Pitch contours
Voicing detection
Melody pitch mean
Octave error removal
Melody pitch mean
Pitch outlier removal
Iterative
Melody selection
Melody peak selection
Melody F0 sequence

Melodie

- Sinusoid Extraction:
  - Anwendung von Equal Loudness-Filter auf Audiosignal
  - Filter verstärkt Frequenzbereich (Melodie häufig gefunden)
  - Filter dämpft Frequenzbereich (unwahrscheinlich die Melodie zu finden)
  - Signal in kleine Blöcke für weitere Verarbeitung

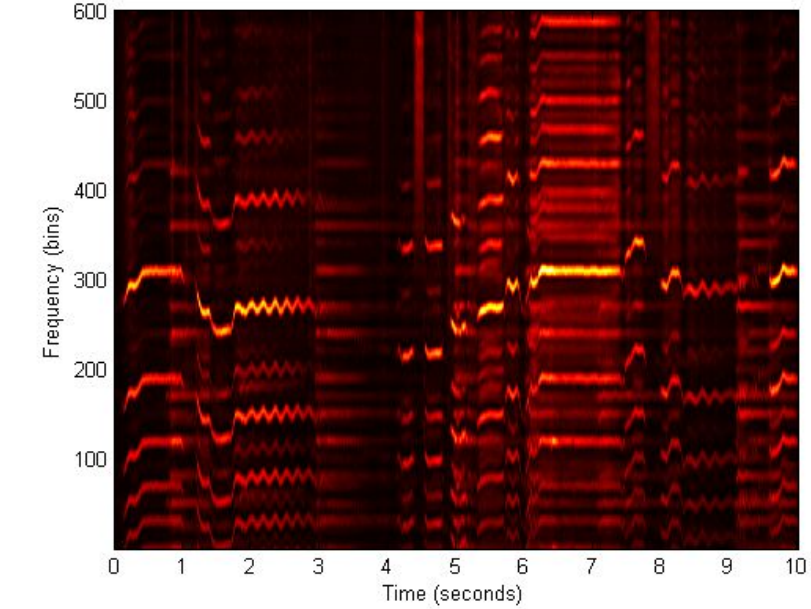
Melodie

- Sinusoid Extraction:
  - Diskrete Fourier-Transformation (DFT) auf jeden Block
  - Spektrale Spitzen (Sinusoide) sind am stärksten energetischen Frequenzen
  - Halten der spitzen Frequenzen und verwerfen der anderen Frequenzen



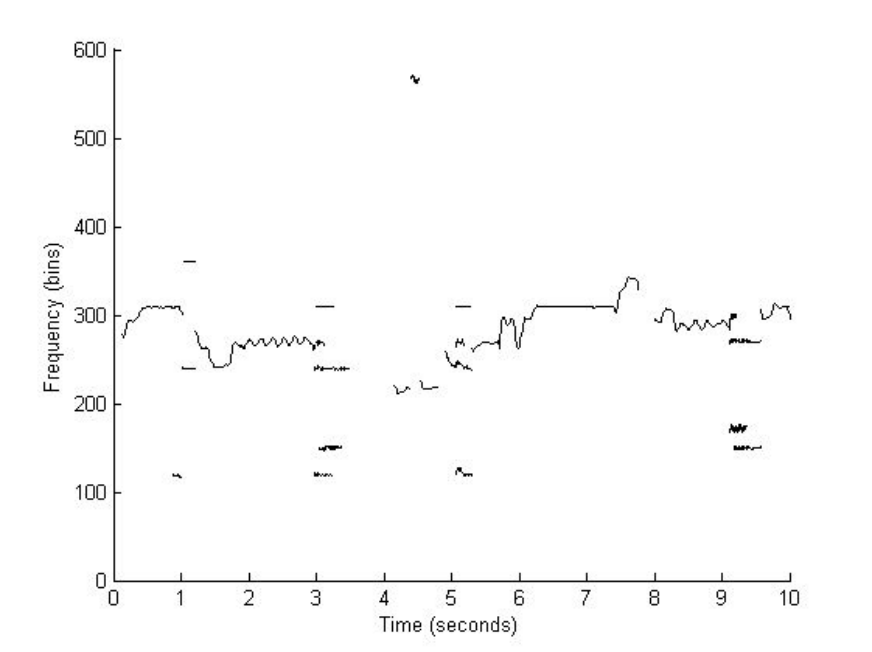
Melodie

- Salience Funktion:
  - Verwendung einer harmonischen Summenbildung
  - Summe der Energie gilt als Salience dieser Tonhöhe



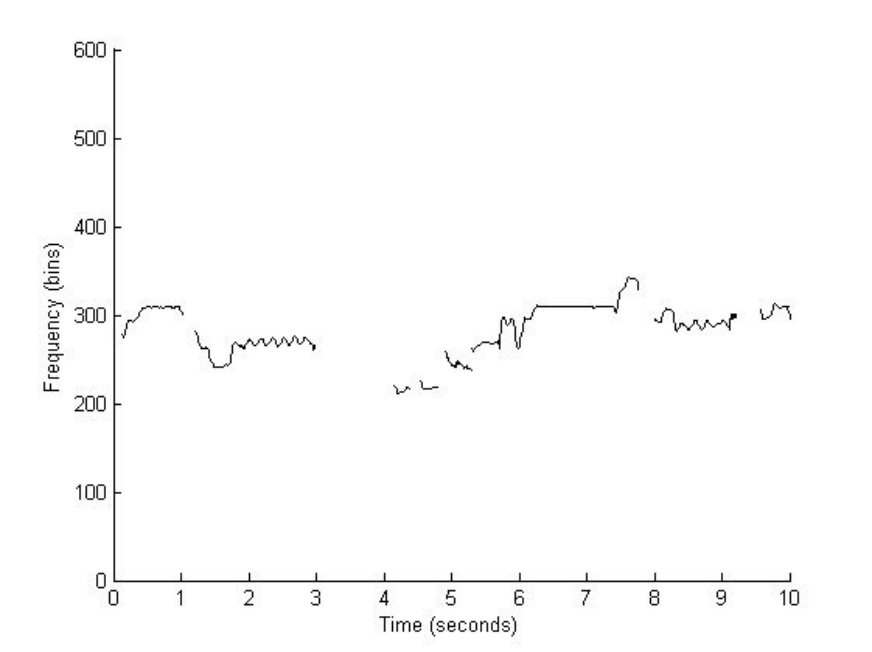
Melodie

- Contour Creation
  - Aus der Salience-Funktion werden Tonhöhenkonturen (Pitch) verfolgt
  - Dauer einer Tonhöhenkontur von einer einzelnen Note bis zu einer kurzen Phrase
  - Spitzen der Salience-Funktion von jedem Block werden genommen



Melodie

- Melody Selection
  - Berechnung von Konturmerkmalen
  - Ausfilterung der nicht-melodischen Konturen



Melodie

- Demo

  http://www.justinsalamon.com/melody-extraction.html#demo

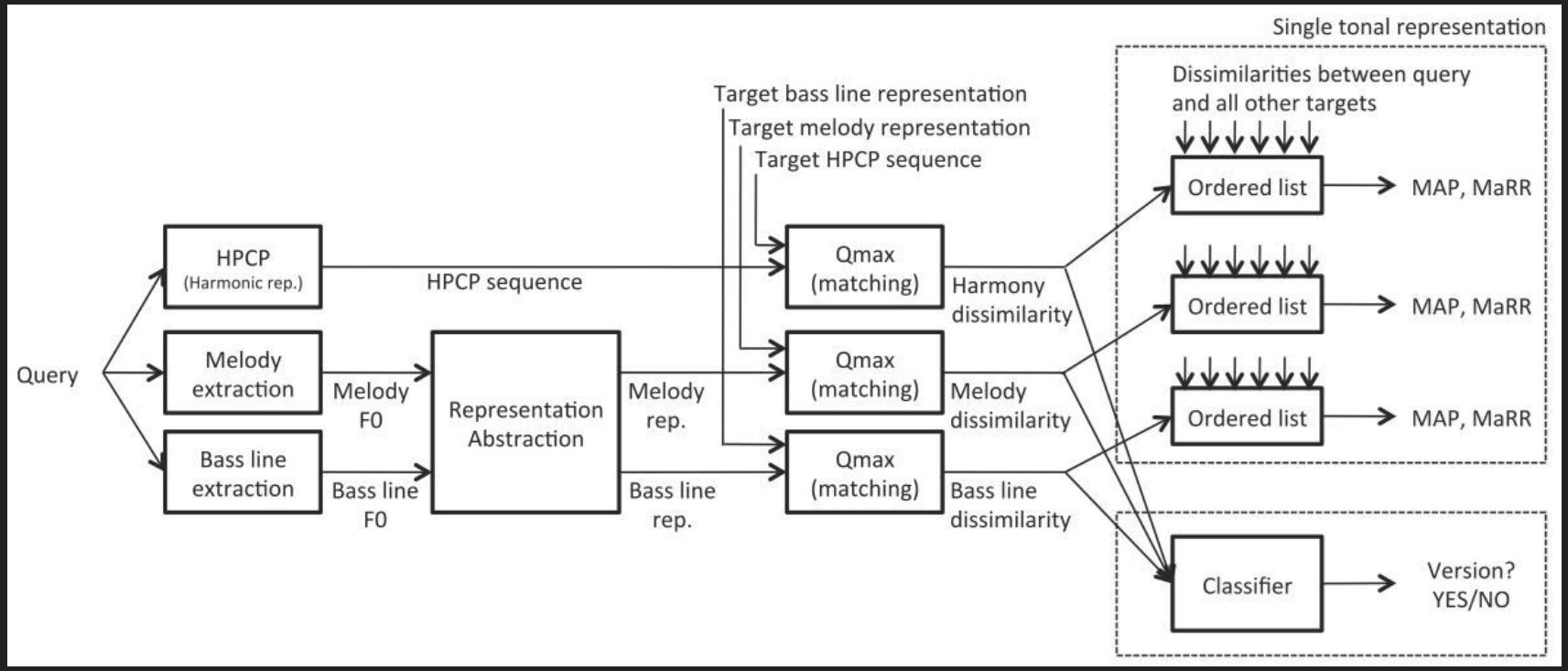
Bassline

- Extrahierung durch Anpassen des Melodie-Extraktionsalgorithmus
- Anwendung eines Tiefpaßfilters mit einer Grenzfrequenz von 261,6 Hz
- Erhöhung der Fenstergröße, da für den Bass eine höhere Frequenzauflösung erforderlich ist

Harmonie

- Berechnung einer Abfolge der harmonischen Klangklassenprofile (HPCP) – spezielle Chroma-Merkmalsimplementierung
- CMI: Leistungsstarkes Werkzeug für die Analyse von Musik (12 Kategorien) {C, C♯, D, D♯, E ,F, F♯, G, G♯, A, A♯, B}
- Vorher: Vorverarbeitungsschritte durchführen

Matching-Prozess



Bewertungsmaßstab

- Verwendung der vom Deskriptor erzeugte Martix, um geordnete Liste von Ergebnissen für jede Abfrage zu erzeugen
- Auswertung der Relevanz der Ergebnisse mit Hilfe von Standardinformationsabfrage-Metriken:
  - MAP = mean average precision
  - MRR = mean reciprocal rank

Bewertungsmaßstab

- Definition des MaRR = mean averaged reciprocal rank
- Beispiel: Abfrage hat 3 Zielversionen in der Auflistung:
  Höchste Mögliche MARR ist (1/1 + 1/2 + 1/3) / 3 = 0,61
  0,61 als grobe Obergrenze für die MaRR
- MAP- und MARR-Maßnahmen sind eine gemeinsame Wahl für die Beurteilung der Genauigkeit von Versionsidentifikationssystemen, die auf einer einzigen Informationsquelle basieren

Bewertungsmaßstab

- Ergebnisse für Single-Tonal Darstellung

Feature	MAP	MaRR
Melody	0.732	0.422
Bass line	0.667	0.387
Harmony	0.829	0.458

Feature	MAP	MaRR
Melody	0.483	0.332
Bass line	0.528	0.355
Harmony	0.698	0.444

Fusion von Musikdarstellungen

Feature	Random Forest	SMO (PolyKernel)	Simple Logistic	KStar	Bayes Net
M	69.84	76.73	75.29	77.98	77.90
B	73.34	81.03	78.98	81.31	81.03
H	82.04	87.69	86.42	87.74	87.58
M+B	79.80	82.05	80.91	84.62	84.46
H+M	84.29	87.73	86.51	88.01	87.81
H+B	84.72	87.80	86.77	88.32	88.14
H+M+B	86.15	87.80	86.83	88.46	88.24

Beispiele von QBH-Systemen

- ACRCloud (https://www.youtube.com/watch?v=JCqcFWwCEcw)
- SoundHound (https://youtu.be/iE6mpKkqcJI)
- Musipedia
- Tunebot

Fazit

- Forschung geht in die Richtung der Verbesserung der Identifikationsgenauigkeit
- Melodie- und Basslinien-Deskriptoren tragen nützliche Information zur Versionsidentifikation
- String-Alignment erzielt bessere Ergebnisse als HMM
- Genauigkeit stark abhängig von Art der Fehlermodellierung
- Erhöhung der Genauigkeit durch Fusion

Danke!

Referenzen

- Bryan Pardo, Jonah Shifrin and William Birmingham, “Name That Tune: A Pilot Study in Finding a Melody From a Sung Query”, 2004.
- Justin Salamon, Joan Serrà and Emilia Gómez, “Tonal Representations for Music Retrieval: From Version Identification to Query-by-Humming”, 2012.
- William Birmingham and Bryan Pardo, “The MusArt music-retrieval system: An overview”, 2002.

- https://www.acrcloud.com/blog/what-is-query-by-humming
- https://www.flickr.com/photos/ter-burg/8127279660
- https://www.youtube.com/watch?v=ATbMw6X3T40 @~1:43
- http://www.clipartkid.com/images/854/sound-wave-bleeding-cool-comic-book-movie-tv-news-IPM0it-clipart.png

Referenzen

- Tonal Respresentations for Music Retrieval: From Version Identification to Query-by-Humming; Salamon, Justin; Serrà, Joan; Gómez, Emilia
- Query by Humming – Music Retrieval Technique; Katkar, Shital
- Melody Extraction from Polyphonic Music Signals using Pitch Contour Characteristics; Salamin, Justin; Gómez, Emilia
- Melody, Bass Line, and Harmony Representations for Music Version Identification; Salamon, Justin, Serrà, Joan; Gómez, Emilia

- Justin Salamon Homepage: http://www.justinsalamon.com/
- The Tunebot Dataset: http://music.cs.northwestern.edu/data/tunebot/
- ACRCloud Blog: https://www.acrcloud.com/blog/what-is-query-by-humming
- Query by humming Youtube: https://www.youtube.com/results?q=query+by+humming