Brandenburgische Technische Universität Cottbus
Institut für Informatik
Lehrstuhl Datenbank- und Informationssysteme

Bachelorarbeit

Entwurf einer GUI zum Lernen von Gewichten beim visuellen Multimedia-Retrieval

Bianca Böckelmann
MatrikelNr.: xxxxxxx
Informations- und Medientechnik

Datum der Ausgabe: 05.05.09
Datum der Abgabe: 15.09.09

1. Betreuer: Prof. Dr.-Ing. Ingo Schmitt
2. Betreuer: Dipl.-Inf. (FH) David Zellhöfer M.Sc.

Inhaltsverzeichnis

1 Einführung

In den letzten Jahren stieg die Anzahl an digitalen Multimedia-Objekten rasant an. Mit dem immer besser werdenden Preis-Leistungs-Verhältnis von digitalen Fotoapparaten und Camcordern, aber auch DVD-Brennern und Receivern mit Festplatte, stiegen deren Absatzzahlen. Die Vorteile der digitalen Aufnahmetechnik führen zu einer stetig größerwerdenden privaten Sammlung an Fotos und Videos, die zunehmend auch mit der Internetgemeinschaft geteilt wird.

Um aus der großen Menge das richtige Bild oder Video zu finden, ist aus zeitlichem Aspekt betrachtet eine automatische Suche unabdingbar. Das Einsortieren in eine nach Kategorien aufgestellte Ordnerstruktur ist meistens aufwendig. Der Dateiname, der meist in der von der Kamera gewählten Kombination aus Zahlen und Buchstaben belassen wird, ist bei der Suche nach einem bestimmten Begriff nicht hilfreich. Eine manuelle Annotation der Bilder mit beispielsweise Aufnahmeort und abgebildetem Inhalt ist im privaten Gebrauch vielleicht noch vertretbar, jedoch für einen großen mehreren Menschen zugänglichen Datenbestand zum einen zu zeitaufwendig und zum anderen subjektiv und unvollständig. Der Inhalt eines Bildes wird durch einen Menschen, und damit nicht objektiv, z.B. nur mit dem Begriff *BMW* beschrieben. Sucht eine andere Person in dieser Bilderkollektion nach einem *Auto*, so taucht in seinem Ergebnis nicht das *BMW*-Bild auf. Das Anfragesystem vergleicht beide Begriffe und weiß jedoch nicht, dass *BMW* eine Teilmenge von *Auto* ist. Desweiteren wären die Metadaten sprachabhängig und einige unbeabsichtigt mit Rechtschreibfehlern versehen. Eine exakte Übereinstimmung einer Bildannotation mit einer Anfrage ist dadurch ohne weiteres nicht immer möglich.

Im World Wide Web können beispielsweise mithilfe der *Google*[1] Bildsuche oder in der Bilddatenbank *Flickr*[2], wie es die Abbildung 1.1 zeigt, unter der Angabe von Begriffen Bilder gesucht werden. Im Ergebnis sind meist auch einge Fotos enthalten, die nicht der Anfrage entsprechen.

[1] `http://www.google.de/`
[2] `http://www.flickr.com/`

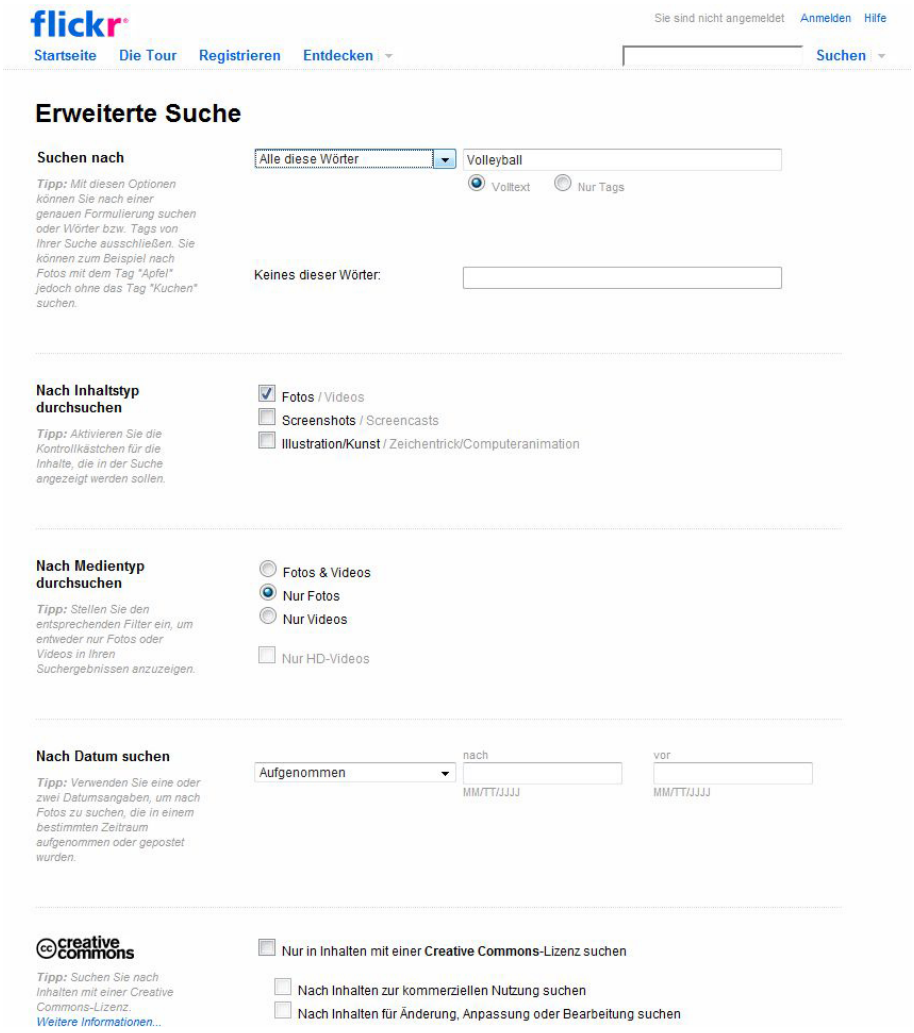


Abbildung 1.1: Erweiterte Suche unter Flickr [Fli09]

Das liegt u.a. daran, dass *Google* und *Flickr* die Anfrage nur mit den manuell erzeugten Informationen des Bildes, den so genannten Tags, vergleichen. Dazu zählen exemplarisch der Dateiname und eine kurze Beschreibung des Dargestellten. Klassische Datenbanksysteme ermöglichen eine Suche nur anhand von Metadaten, jedoch nicht anhand des Abgebildeten. Die tatsächlich dargestellte Semantik kann nicht in die Anfrage einbezogen werden.

Die Lösung für das Problem sind Retrieval-Systeme für Multimedia-Objekte. In diesen wird inhaltlich nach Objekten gesucht, die die Anfrage erfüllen. Abhängig vom Medientyp wird eine automatische Extraktion der Informationen, genannt Features, nach verschiedenen Gesichtspunkten vorgenommen. Bei Bildern kann das u.a. die Farbverteilung oder der Umriss der dargestellten Objekte sein. Lautet eine Anfrage des Nutzers „Finde alle Bilder,

auf denen ein Strandkorb am Meer abgebildet ist“, so wird nach einem Bild mit einem großem Blauanteil in der oberen und einem großen Beigeanteil in der unteren Bildhälfte gesucht, auf dem ein rechteckiges Objekt in der Mitte dargestellt ist. Der Nutzer kann aber auch ein Bild als Beispiel für seine Suche „Finde alle Bilder, die diesem Bild ähnlich sind“ angeben. Hilfreich ist diese Herangehensweise beispielsweise bei der polizeilichen Ermittlung eines Täters anhand eines Phantombildes. Überwachungsvideos und Telefongespräche könnten hierbei auch herangezogen werden.

Multimedia-Retrieval-Systeme liefern aufgrund unpräziser Anfragen und vager Objektbeschreibungen eine Liste von Ergebnissen zurück, die absteigend nach der Relevanz bezüglich der Anfrage sortiert ist. Im Gegensatz zu traditionellen Datenbanksystemen, in denen das Ergebnis die Anfrage entweder exakt erfüllt oder nicht, kann hier zusätzlich eine partielle Erfüllung auftreten. Auf das obige Beispiel beziehend, wäre ein Bild, welches einen Strandkorb an der Ostsee zeigt, im Ergebnis vor jenen Fotos platziert, die nur einen Strandkorb auf einer Wiese oder die Ostseeküste ohne Strandkörbe darstellen. Ähnlichkeit wird von vielen Menschen unterschiedlich wahrgenommen. Die ersten Ergebnisse der Suche sind deshalb meist nicht zufriedenstellend. Um das subjektive Ähnlichkeitsempfinden in die Anfrage einfließen zu lassen, werden einzelne Terme der Bedingung gewichtet [SZ09]. Dem Nutzer fällt es jedoch schwer Angaben über die numerischen Gewichtswerte zu machen, da er u.a. deren Bedeutung im System nicht kennt. Aus diesem Grund gibt er stattdessen Präferenzen auf der Ergebnisliste an [ZL08]. Er vergleicht jeweils zwei Objekte und entscheidet, welches von beiden seinem Informationswunsch besser entspricht. Intern werden die Präferenzen in Gewichte umgewandelt, die die Anfrage modifizieren. Anschließend wird ein erneuter Suchdurchlauf gestartet. Nicht relevante Objekte werden aus der Liste entfernt und durch neue ersetzt. Dieser iterative Prozess, als Relevance-Feedback bezeichnet, lässt das System Gewichte lernen. Es hat sich gezeigt, dass die Bewertung der Relevanz der Ergebnisobjekte durch den Nutzer verbunden mit einer iterativen Suche meist bessere Ergebnisse erzielt als ohne diese [Sch06].

Das Spektrum der Anwendungsgebiete von Multimedia-Retrieval-Systemen ist breit gefächert. Privatpersonen, Journalisten, Fernseh- und Radiosender, Ärzte und Polizei können hier exemplarisch als Nutzer genannt werden. Es ist deshalb wichtig eine einfach zu bedienende Nutzerschnittstelle bereitzustellen. Diese Bachelorarbeit beschäftigt sich aus diesem Grund mit dem Entwurf einer grafischen Benutzeroberfläche für visuelle Multimedia-Retrieval-Systeme, die dem Anwender eine einfache Gewichtung der Ergebnisobjekte ermöglicht und diese durch das System lernen lässt.

Gliederung der Arbeit

Im Kapitel 2 wird in die Thematik des Multimedia-Retrievals eingeführt. Die Begriffe Multimedia und Information-Retrieval werden geklärt, wobei eine Abgrenzung zum Daten-Retrieval erfolgt. Der grundlegende Suchablauf beim Multimedia-Retrieval wird beschrieben. Im Anschluss werden die Konzepte der Anfragesprache CQQL vorgestellt, insbesondere das Relevance-Feedback.

Im Kapitel 3 werden einige bereits bestehende Benutzeroberflächen von Retrieval-Systemen für die Suche von Bildern bzw. Videos bezüglich der Interaktion mit diesen beschrieben und bewertet.

Anforderungen an die in dieser Arbeit zu entwickelnde GUI für das visuelle Multimedia-Retrieval werden im Kapitel 4 aufgestellt. Sowohl grundlegende Punkte als auch die Anforderungen für die Suche, die Darstellung der Ergebnisse und für das Relevance-Feedback werden betrachtet. Es werden an dieser Stelle die Kriterien aufgezeigt, die sich durch die Verwendung von CQQL als Anfragesprache ergeben.

Im folgenden Kapitel 5 werden anhand der aufgestellten Anforderungen verschiedene GUI-Prototypen für die inhaltsbasierte Bildsuche präsentiert. Bei der Suche werden Methoden für das Browsing und Navigieren von denen der Anfrage separiert. Es folgen Entwürfe, wie die Fotos dem Anwender angezeigt werden sollen. Im Rahmen des Relevance-Feedbacks werden Ideen für die Angabe von Nutzerbewertungen vorgelegt. An entsprechender Stelle werden die unterschiedlichen Prototypen in Bezug auf Benutzerfreundlichkeit, Erfüllung der CQQL-Prinzipien und auftretenden Problemen bewertet. Am Ende des Kapitels wird eine zusammenfassende Beurteilung der vorgestellten Entwürfe vorgenommen.

Die Integrationsmöglichkiet der GUI-Prototypen in bestehende Arbeiten am Lehrstuhl wird im Kapitel 6 untersucht. Im letzten Kapitel werden die in dieser Arbeit gewonnenen Erkenntnisse zusammengefasst und ein Ausblick für zukünftige Weiterentwicklungen gegeben.

2 Multimedia-Retrieval

In diesem Kapitel wird ein Überblick über den Ablauf der Suche in Multimedia-Retrieval-Systemen gegeben. Eine Abgrenzung zu den klassischen Datenbanksystemen findet statt. Im Hinblick auf die zu konstruierende GUI wird die am Lehrstuhl Datenbank- und Informationssysteme entworfene Anfragesprache CQQL näher beschrieben.

2.1 Einführung

Der Begriff *Multimedia* ist aus dem alltäglichen Sprachgebrauch nicht mehr wegzudenken. In [Sch06] wird ein *Multimedia-Objekt* als die Kombination von Daten verschiedener digitaler Medientypen verstanden, wobei mindestens ein Medientyp nicht alphanumerisch sein darf. Als Medientypen seien Text, Grafik, Bild, Audio und Video zu nennen. Mit Grafiken sind hier Vektor- und mit Bildern Pixelgrafiken gemeint. Diese Medien lassen sich u.a. anhand ihres Zeitbezugs klassifizieren. Statische Medien wie Bilder sind im Gegensatz zu dynamischen Medien wie Videos zeitunabhängig. In [Hen08] wird sich der Definition von Ralf Steinmetz bedient:

> *„Ein Multimediasystem ist durch die rechnergesteuerte, integrierte Erzeugung, Manipulation, Darstellung, Speicherung und Kommunikation von unabhängigen Informationen gekennzeichnet, die in mindestens einem kontinuierlichen (zeitabhängigen) und einem diskreten (zeitunabhängigen) Medium kodiert sind.“*

In dieser Arbeit wird die Definition von Multimedia nach [Sch06] verwendet. Ziel ist es, für das Multimedia-Retrieval (MMR) grafische Nutzerschnittstellen zu entwerfen, wobei der Schwerpunkt auf Bildern mit Hinblick auf Videoobjekten liegen soll.

Relationale Datenbanksysteme, deren zugrunde liegendes Relationenmodell vom amerikanischen Wissenschaftler Edgar F. Codd Ende der 1960er Jahre aufgestellt worden ist [Kud07], sind ursprünglich nicht für die Verwaltung und Suche von Texten, Fotos, Videos und Tonaufnahmen konzipiert worden. Für die Suche nach Dokumenten haben sich indessen Information-Retrieval-Systeme (IRS) entwickelt, die anfänglich vor allem im Bibliothekswesen genutzt wurden. In den letzten Jahren wurden aufgrund der ansteigenden

Datenflut von Multimedia-Objekten Multimedia-Retrieval-Systeme (MMRS) entworfen. Diese werden in einigen Quellen wie in [Hen08] auch als Multimedia-Information-Retrieval-Systeme bezeichnet, da sie auf den Konzepten des Information-Retrieval aufbauen und die Verwaltung und Suche um Multimedia-Objekte erweitern. Die Unterschiede zwischen klassischen Datenbanksystemen und Information-Retrieval-Systemen werden im Folgenden näher erläutert.

Zunächst soll jedoch der Begriff Information-Retrieval, in der Fachliteratur mit IR abgekürzt, definiert werden. Das Langenscheidt Handwörterbuch Englisch [Mes94] bietet für das Verb *to retrieve* nachstehende Übersetzungen an:

1. *hunt.* apportieren
2. wiederfinden, -bekommen
3. (sich *et.*) zurückholen
4. *et.* herausholen, -fischen (from aus)
5. *fig.* wiedergewinnen, -erlangen; *Fehler* wiedergutmachen; *Verlust* wettmachen
6. *j-n* retten (from aus)
7. *et.* der Vergangenheit entreißen

Andreas Henrich [Hen08] kommt, die Bedeutung des Wortes *Information* einbeziehend, zu folgender Definition von **Information-Retrieval**:

> *„(Wieder-)Gewinnung von Informationen, die von jemandem in einer konkreten Situation zur Lösung von Problemen benötigt werden.“*

Die Abbildung 2.1 vergleicht Daten- und Information-Retrieval miteinander. Unter Daten-Retrieval ist der Zugriff auf klassische relationale Datenbanken zu verstehen. Während beim Daten-Retrieval die Informationen strukturiert in einer expliziten Form vorliegen, müssen diese beim Information-Retrieval meist aus dem Objekt extrahiert werden. Dieser Prozess nennt sich Feature-Extraktion. In einem Text wird beispielsweise vereinfacht formuliert die Häufigkeit des Auftretens aller Wörter in einem Dokument gezählt. Beim Daten-Retrieval werden nur Ergebnisse zurückgeliefert, die die Anfrage vollständig erfüllen. Alle Ergebnisobjekte sind untereinander gleichrangig, sie werden also als Menge angegeben. Eine in SQL formulierte Anfrage wie `SELECT * FROM studenten WHERE studiengang = 'IMT' AND matrikel > 2600000;` liefert nur die Datensätze von Studenten zurück, die den Studiengang IMT gewählt haben und eine größere Matrikelnummer als 2600000 besitzen. Anders ist es beim Information-Retrieval. Aufgrund vager Anfragen, die in einer natürlichen Sprache formuliert sind, und ungenauen Objektbeschreibungen werden auch

Ergebnisse präsentiert, die mit der Anfrage nur teilweise übereinstimmen. Absteigend sortiert werden die Ergebnisse anhand ihrer Relevanz zur Anfrage in einer Liste aufgeführt. Die Größe der Liste wird meist mittels eines Relevanzschwellwertes oder der maximalen Anzahl von Ergebnissen eingeschränkt. Nicht nur das reine Auftreten der Begriffe, sondern auch deren Häufigkeit innerhalb des Dokuments ist ausschlaggebend für den Rang in der Liste. Treten die gesuchten Vokabeln oft in einem Abschnitt auf, so sind diese wichtiger anzusehen als selten vorkommende Bezeichnungen. Neben identischen Dokumenten werden auch ähnliche ermittelt. Aufgrund des Ähnlichkeitsbegriffs ist eine gewisse Fehlertoleranz bei Information-Retrieval-Systemen vorhanden. Für den Laien ist es im Allgemeinen schwer eine Ähnlichkeitsanfrage genau seinen Bedürfnissen entprechend aufzustellen. Es ist deshalb meist notwendig die Suche iterativ durchzuführen, um bessere Ergebnisse zu erzielen. Anders sieht es bei klassischen Datenbanksystemen aus, wo es nur eine gültige Ergebnismenge geben kann und somit ein erneuter Suchdurchlauf unnötig ist. Fehler werden hierbei nicht toleriert. In [Sch06,Hen08] können diese Kriterien der Abgrenzung beider Systeme nachgelesen werden.

Merkmal	Daten Retrieval	Information Retrieval
Information	explizit	implizit
Ergebnisse	exakt	unscharf
Anfrage	einmalig	iterativ verfeinernd
Fehlertoleranz	keine	vorhanden
Ergebniskollektion	Menge	Liste

Abbildung 2.1: Vergleich Daten- und Information-Retrieval [Sch06]

Die Unterschiede zwischen Multimedia-Retrieval-Systemen zu Datenbanksystemen verhalten sich analog zu den hier aufgeführten Vergleichen. Hervorzuheben sei am Schluss, dass beim Information- bzw. Multimedia-Retrieval die Daten unstrukturiert vorliegen, so dass für eine Suche anhand der tatsächlichen Semantik eine Extraktion des Wissens nötig ist. Auf diesen Prozess wird in 2.2.1 genauer eingegangen.

2.2 Prinzipien des Multimedia-Retrievals

Um die Anforderungen an grafische Benutzeroberflächen für Multimedia-Retrieval-Systeme zu spezifizieren, ist es wichtig die Konzepte des MMR zu kennen. In diesem Abschnitt werden deshalb die Verwaltung von Multimedia-Objekten und der Suchablauf genauer erklärt. Desweiteren wird auf CQQL, eine am Lehrstuhl Datenbank- und Informationssysteme entworfene Anfragesprache, eingegangen. Um die Ergebnismenge den Wünschen des

Nutzers besser anzupassen, findet der Relevance-Feedback-Prozess Anwendung. Beschrieben wird dieser in der Anwendung in einem CQQL-System.

2.2.1 Ablauf

Die Verwaltung und die Suche wird in 5 Schritte unterteilt [Sch06], die in Abbildung 2.2 schematisch dargestellt sind:

1. **Einfügen in die Multimedia-Datenbank:** Multimedia-Objekte und relationale Daten werden entsprechend ihres Typs in die Datenbank eingefügt. Relationale Daten werden in einer relationalen Datenbank abgelegt. Beispiele für solche strukturierten Daten sind Metadaten wie Auflösung und Speicherformat eines Bildes, Nutzerprofile und Strukturdaten. Multimedia-Objekte hingegen müssen vorverarbeitet werden, bevor sie in Form von Rohdaten und relationalen Daten abgelegt werden können. Zuerst werden komplexe Multimedia-Objekte in Strukturdaten und nicht komplexe Medienobjekte zerlegt. Ein Medienobjekt beinhaltet Daten eines beliebigen Medientyps, anders bei Multimedia-Objekten, die mindestens zwei Medientypen kombinieren. In einer Power-Point-Präsentation werden so etwa die Bilder und Animationen von den Textteilen getrennt. Der zweite Vorverarbeitungsschritt ist die Normalisierung, bei der die Medienobjekte in ein einheitliches Format gebracht werden um Störfaktoren auszuschalten. Eine verzerrte Architekturfotografie wird so perspektivisch zu einer unverzerrten Frontalaufnahme transformiert. Zuletzt werden diese Objekte segmentiert. Ein Segment stellt dabei einen eigenständigen Teil des Objekts dar. Mögliche Segmente für die einzelnen Medientypen sind in Abbildung 2.3 aufgeführt. Die Segmentierung kann manuell, aber auch automatisch bzw. teilautomatisch durchgeführt werden. Der Benutzer kann zusätzlich zu den einzelnen Teilen Informationen angeben, die dann relational in der Datenbank abgespeichert werden. Nach diesen drei Schritten der Vorverarbeitung von Multimedia-Objekten gelangen die Rohdaten getrennt von den manuell hinzugefügten Metadaten in die Datenbank.

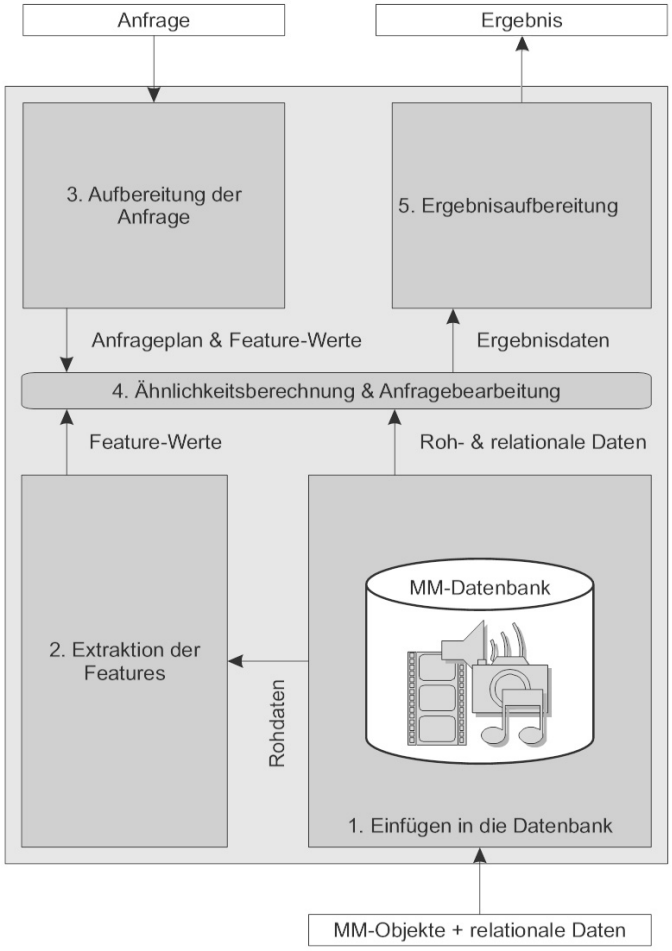


Abbildung 2.2: Verwaltung und Retrieval von Multimedia-Daten - grober Ablauf [Sch06]

2. **Extraktion der Feature-Werte:** Um eine inhaltsbasierte Ähnlichkeitssuche zu ermöglichen, werden aus den Rohdaten für jedes Segment automatisch Feature-Werte extrahiert. Diese charakterisieren die Semantik der Segmente bzw. des gesamten Objekts. Zuerst werden die Features in Abhängigkeit vom Medientyp, der beabsichtigten Ähnlichkeitsberechnung und den bereitgestellten Extraktionsalgorithmen erkannt. Abbildung 2.3 listet einige Features der verschiedenen Medientypen auf. Für Bilder sind u.a. die Farb-, Textur- und Formeigenschaften signifikant. Hilfreich zur Ähnlichkeitsbestimmung zweier Bilder auf Grundlage von Farben ist das Farbhistogramm. Dieses stellt den Anteil der zu jedem Farbbereich eines Farbmodells zugehörigen Pixel im Bild [Hen08] dar. Bei Videoobjekten kommen zusätzliche Feature-Werte hinzu, die die zeitliche und räumliche Abhängigkeit beschreiben. Bewegungsvekto-

ren geben die Richtung und Geschwindigkeit von sich zeitlich bewegenden Teilen in Einzelbildern an [Sch06].

	Feature-Extraktion	Segmentierung	Konvertierung
Textdaten	Statistiken zu Wortvorkommen, ...	Gliederung in Kapitel, Abschnitte, Paragraphen, Sätze, ...	als Bild drucken, vorlesen, ...
Bilddaten	Farbhistogramme, Textur, ...	regionenbasierte Segmentierung, kantenbasierte Segmentierung, ...	Optical Character Recognition (OCR), ...
Audiodaten	Sprechererkennung, Erkennung von Musikinstrumenten, Tonhöhen, ...	Erkennung von Pausen, ...	Spracherkennung, ...
Videodaten	Bewegungserkennung, ...	Szenenerkennung, ...	Extraktion von Schlüsselbildern, Erkennen von Untertiteln, ...

Abbildung 2.3: Beispieloperationen für verschiedene Medientypen [Hen08]

Diese automatisch extrahierten Features werden als Low-Level-Features bezeichnet [BVBF07]. Im Gegensatz dazu fallen die menschlichen Annotationen der Medienobjekte unter den Begriff der High-Level-Features, da solche für den Nutzer aussagekräftig und verständlich sind. Ein Farbhistogramm gibt nur Auskunft über die Farbanteile im Bild, die Bedeutung derer für den Menschen kann vielseitig sein. Ein hoher Blauanteil kann auf die Darstellung eines Himmels, eines Meeres oder einer Nahaufnahme der bekannten Pflanze Enzian hindeuten. Dieser Abstand zwischen beiden Feature-Typen wird semantische Lücke genannt.

Nach der Feature-Erkennung folgt die Feature-Aufbereitung, die Abhängigkeiten zwischen den verschiedenen Feature-Werten beseitigt. Um eine schnelle Suche auf den Feature-Werten zu realisieren und die entsprechenden ähnlichsten Medienobjekte sofort zurückliefern zu können, werden Feature-Indizes verwendet.

3. **Aufbereitung der Anfrage:** Multimedia-Retrieval-Systeme können sowohl klassische Datenbankanfragen und Ähnlichkeitsanfragen als auch deren kombinierte Variante verarbeiten. Bei der herkömmlichen Datenbankanfrage wird versucht die Größe von Zwischenergebnissen zu minimieren. Als Resultat der Optimierung entsteht ein

Anfrageplan. Bei der Ähnlichkeitsanfrage wird anders vorgegangen, da hier ein Multimedia-Objekt die Anfrage darstellt. Die im Schritt 1 und 2 erläuterten Verfahren der Vorverarbeitung und Feature-Extraktion finden hier erneut Anwendung. Das Ergebnis sind die entsprechenden Feature-Werte, die zusammen mit dem Anfrageplan für die Anfragebearbeitung benötigt werden.

4. **Ähnlichkeitsberechnung und Anfragebearbeitung:** Eine Suche kann, wie bereits unter Schritt 3 erwähnt, klassische Datenbankanfragen mit Ähnlichkeitsanfragen vereinen. Anhand des erstellten Anfrageplans und den unter Schritt 1 eingefügten relationalen Daten wird das Ergebnis der Datenbankanfrage berechnet. Bei einer Ähnlichkeitsberechnung werden mittels einer bestimmten Ähnlichkeitsfunktion auf Grundlage der Feature-Werte der in die Datenbank eingefügten Multimedia-Objekte und der Feature-Werte der Anfrageobjekte die ähnlichsten Datenbankobjekte herausgefunden. Bei einer kombinierten Anfrage tritt das Problem auf, dass die Ergebnisse zweier unterschiedlicher Systeme, deren grundlegende Konzepte in der Einführung dieses Kapitels verglichen worden sind, miteinander verknüpft werden müssen. Die exakten und gleichrangigen Ergebnisse des Daten-Retrievals müssen mit der nach Relevanz sortierten Ergebnisliste der Ähnlichkeitssuche zusammengeführt werden. Welche Anfragesprache dieses Problem lösen kann, ist unter 2.2.2 zu lesen.

5. **Ergebnisaufbereitung:** Im letzten Schritt müssen die Ergebnisse den Anforderungen des Nutzers angepasst werden. Das kann eine Medienumsetzung, also die Umwandlung der Informationen eines Medientyps in einen anderen, eine Formatumwandlung oder die Wiederherstellung eines komplexen Medienobjekts bedeuten. Die Art der Präsentation ist dabei abhängig vom Medientyp der darzustellenden Ergebnisse.

2.2.2 Anfragesprache CQQL

In relationalen und objektrelationalen Datenbanksystemen wird u.a. die Anfragesprache *Structured Query Language* (SQL) verwendet. Eine SQL-Anfrage besteht aus Bedingungen, die mittels boolescher Junktoren miteinander verbunden sind. Die Ergebnismenge enthält aufgrund der Booleschen Logik nur Datenbankobjekte, die die Anfrage exakt erfüllen. Wie bereits in 2.1 beschrieben wurde, liefert eine inhaltsbasierte Suche in MMRS hingegen eine Liste von Ergebnissen zurück, die sortiert ist nach der Relevanz bezüglich der Anfrage. Zwei signifikante Probleme treten bei der Verwendung von SQL als Anfragesprache in Multimedia-Retrieval-Systemen auf [SZ09]. Zum einen werden keine unscharfen Anfragen ermöglicht. Das ist jedoch erforderlich, da bei einer Ähnlichkeitsanfrage der Nutzer

unpräzise Angaben macht. MMRS werden für viele verschiedene Anwendergruppen konzipiert, von denen im Allgemeinen nicht erwartet werden kann, dass sie der Syntax von SQL kundig sind und die zugrunde liegende Datenstruktur kennen. Die Suche in einem MMRS wird durch Laien initiiert, die sich der natürlichen Sprache bedienen. Eine Anfrage für eine Bildersuche könnte so etwa die Bedingungen „größtenteils rot“ und „sehr eckige Konturen“ enthalten. Zum anderen realisiert SQL keine Gewichtung von einzelnen Anfragetermen. Experimente haben ergeben, dass Ähnlichkeit subjektiv empfunden wird und eine Gewichtung von Teilen der Anfrage die Nutzerzufriedenheit bezüglich der gelieferten Ergebnisse steigert [SZ09]. Welche Objekte als Ergebnis bevorzugt werden, wird mittels Präferenzen durch den Nutzer angegeben. Zwar kann SQL für die klassischen Datenbankanfragen in einem MMRS verwendet werden, jedoch eignet es sich nicht für Ähnlichkeitsanfragen und die Gewichtung von Anfragetermen in solchen Systemen.

Es existieren eine Reihe von Multimedia-Anfragesprachen. Eine genaue Beschreibung dieser würde den Rahmen der Bachelorarbeit sprengen. Es sei hier auf [Sch04] verwiesen. In der vorliegenden Arbeit wird speziell die Anfragesprache CQQL behandelt, da die zu erstellende Benutzeroberfläche auf ein System aufsetzen soll, welches sich dieser Anfragesprache bedient. CQQL steht für *Commuting Quantum Query Language* und wurde am Lehrstuhl Datenbank- und Informationssysteme an der BTU Cottbus entwickelt. Im Gegensatz zu SQL ermöglicht es sowohl die Verwendung vager Prädikate als auch die Gewichtung einzelner Anfrageterme [Sch08]. Um eine Anfrage an das subjektive Ähnlichkeitsempfinden anzupassen, werden Präferenzen angegeben. Die Angabe dieser wird in zwei Klassen unterteilt: qualitative und quantitative Ansätze [Sch04]. Bei den qualitativen Ansätzen legt der Nutzer seine Präferenzen durch die Angabe von Halbordnungen fest. Eine Halbordnung ist eine Relation, die reflexiv, antisymmetrisch und transitiv ist [Hag04]. Steht bei einer Halbordnung jedes Paar von Elementen in Relation, so liegt eine Totalordnung vor. Hervorzuheben sei, dass der Anwender seine Präferenzen bereits zum Zeitpunkt der Anfrageformulierung kennen muss. Dieses ist in vielen Fällen für den Laien jedoch schwierig. Beispielsweise kann er bei einer Bildersuche nicht konkret angeben, wie wichtig ihm einzelne Feature-Werte sind [SZ09]. In quantitativen Ansätzen wird durch die Verwendung einer numerischen Scoring-Funktion eine Totalordnung der Ergebnisobjekte erzielt. Hier werden die Präferenzen indirekt in der Regel durch numerische Gewichtungswerte über Anfragebedingungen ausgedrückt [Zel09].

Beispiel 1 *Gesucht werden Bilder von Autos, wobei rote Ferraris bevorzugt werden.*

qualitativer Ansatz: *Es entsteht eine Halbordnung der Ergebnisse. Nach dem Ceteris-paribus-Prinzip stehen die Objekte, die die Präferenz erfüllen, vor allen übrigen,*

die als gleichrangig bewertet werden [Zel09]. Ein Bild eines roten Ferraris wird gegenüber einem Bild mit einem schwarzen Ferrari, einem blauen BMW oder einem roten Ford bevorzugt. Eine Ordnung innerhalb der letzten drei aufgezählten Bilder existiert nicht.

quantitativer Ansatz: *Es entsteht eine Totalordnung der Ergebnisse. Im Gegensatz zum qualitativen Ansatz existiert zusätzlich eine Ordnung innerhalb der Bilder, die die Präferenz nicht vollständig erfüllen.Für Automarke=Ferrari$\wedge_{0.6 0.4}$Lackierung=rot, wobei 0.6 die Gewichtung der Bedingung Automarke und 0.4 die der Lackierung ist, würde sich folgende Ergebnisreihenfolge ergeben: Das Bild des roten Ferraris steht vor dem Bild des schwarzen Ferraris, welches wiederum besser ist als das des roten Fords. Am schlechtesten wird das Bild des blauen BMWs eingestuft, das keines der beiden Bedingungen erfüllen kann.*

Die Ergebnismenge des quantitativen Ansatzes ist gegenüber der des qualitativen Ansatzes ausdifferenzierter [Zel09], da eine Aussage getroffen werden kann, um wie viel ein Ergebnis besser als ein anderes ist. In MMRS hat sich der quantitative Ansatz durchgesetzt [Sch04]. Die Angabe von numerischen Gewichtswerten für Anfrageterme stellt jedoch besonders bei komplexen Anfragen ein Hindernis für den Nutzer dar. Ein CQQL-System kombiniert deshalb die Vorzüge beider Ansätze [Zel09]. Intern wird mit CQQL eine Gewichtung der Anfrageteile vorgenommen, welche dem Anwender in diesem Verfahren verborgen bleibt. Der Nutzer gibt über die Nutzerschnittstelle seine Präferenzen $\geq$ paarweise über der Ergebnismenge an. Im Gegensatz zu traditionellen qualitativen Verfahren werden hier keine abstrakten Präferenzen verwendet. Unter einer abstrakten Präferenz sei in diesem Zusammenhang eine Präferenz zu verstehen, die nicht nur für einzelne Objektpaare, sondern für alle Objekte eine Ordnung herstellt. Der grundlegende Ablauf geht wie folgt vonstatten. Die vom Nutzer erstellte CQQL-Anfrage wird nach einem bestimmten Schema initial gewichtet, z.B. wird jeder Term als gleichwertig angesehen. Das System ermittelt daraus eine total geordnete Ergebnismenge, die zu einem Anteil dem Anwender präsentiert wird. Auf diesem Bereich des Ranks gibt der Nutzer seine Präferenzen an, welche intern in Gewichte umgewandelt werden und die Anfrage für einen erneuten Suchdurchlauf modifizieren. Abbildung 2.4 zeigt dieses Verfahren schemenhaft, welches in [SZ09] erläutert wird. Das System lernt durch diesen iterativen Prozess die Gewichte, die das Ähnlichkeitsempfinden des Nutzers ausdrücken bzw. sich diesem annähern. Das Relevance-Feedback-Verfahren wird später detaillierter beschrieben.

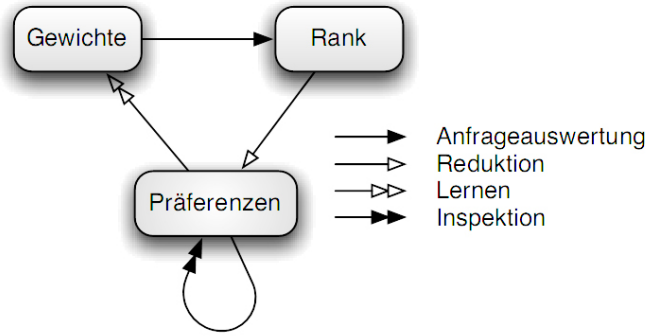


Abbildung 2.4: Zustandsdiagramm der Datenstrukturen [SZ09]

CQQL basiert auf den Gesetzen der Quantenlogik. Dadurch ist es möglich die unterschiedlichen Suchverfahren des Daten- und Information-Retrievals zu vereinen [SZ09]. Somit können die klassischen Datenbankanfragen, die auf der Booleschen Logik beruhen, mit Ähnlichkeitsanfragen kombiniert werden. Eine CQQL-Anfrage kann so neben Booleschen Bedingungen, die nur den Wert 0 oder 1 zulassen, auch Score-basierte Bedingungen enthalten, welche jeweils einen Wert aus dem Intervall [0,1] abhängig von der Nähe des Attributwerts zum Sollwert zurückgeben. Desweiteren können die Operanden, und zwar die Disjunktion und Konjunktion, gewichtet werden. Syntaktisch lehnt sich die Anfragesprache an das relationale Bereichskalkül an.

In [SZ09] stellt q^{Θ} eine gewichtete CQQL-Anfrage mit den Gewichtsvariablen $\Theta = \{\theta_i\}$ dar. Die Gewichtungsfunktion w ordnet jeder Gewichtsvariablen θ_i einen Wert aus dem Intervall [0,1] zu. w^I ist die initiale Gewichtungsfunktion, die z.B. alle Gewichte auf den Wert 1 setzt. In diesem Fall verhält sich eine Anfrage wie eine ungewichtete Anfrage, da alle Gewichte den gleichen Wert haben. Eine andere mögliche Variante wäre eine initiale Gewichtung mittels Zufallszahlen. Wird eine Bedingung mit Null gewichtet $\theta_i = 0$, so hat sie keinen Einfluss auf das Gesamtergebnis. Die Auswertung der Anfrage q^{Θ} wird durch die Funktion $eval\left(q^{\Theta}, o, w\right)$ realisiert, welche jedem Anfrageobjekt aus der Menge $O = \{o_1, o_2, \ldots, o_n\}$ einen Score aus [0,1] zuordnet. Der Wert 1 wird bei maximaler und der Wert 0 bei minimaler Erfüllung der Anfrage vergeben. Die Berechnung des Scores für jedes Objekt $o \in O$ mittels $eval\left(q^{\Theta}, o, w^I\right)$ ergibt eine Totalordnung der Ergebnismenge, die Permutation $p\left(w^I\right) : \{1, \ldots, n\} \rightarrow O$. Der Rank, der mit der initialen Gewichtungsfunktion w^I erzeugt wird, entspricht meist nicht dem Ähnlichkeitsempfinden des Nutzers. Ziel des Relevance-Feedbacks ist es deshalb, die Gewichtungsfunktion w^t zu finden bzw. sich dieser anzunähern, die den Rank erzeugt, der die Erwartungen des Nutzers an das Ergebnis optimal erfüllt.

Beispiel 2 *Ein Anwender sucht nach einem Notebook. Dieses soll eine lange Akkubetriebsdauer aufweisen (a). Bei einem Mini-Notebook, einem Netbook (net), fordert er eine gute Bedienbarkeit der Tastatur (k) und einen günstigen Preis (p). Findet sich kein Notebook in kleiner Größe, dann muss der Prozessor eine hohe Taktfrequenz (f) haben und der Festplattenspeicher über eine große Kapazität (m) verfügen. Abbildung 2.5 zeigt auf der linken Seite den dazugehörigen Bedingungsbaum mit den entsprechenden Gewichtsvariablen für die Bedingungen k, p, f und m (nach [SZ09]).*

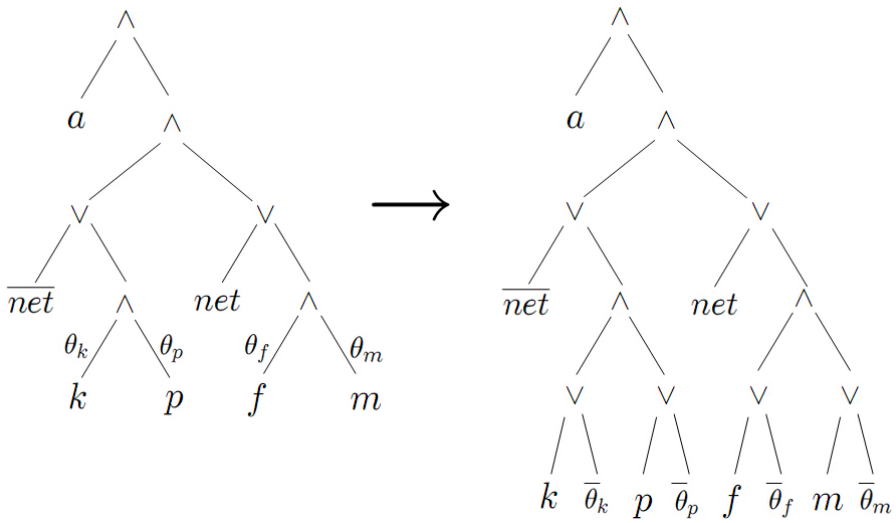


Abbildung 2.5: gewichtete Anfrage als Baum (links) und Transformation in ihr logisches Äquivalent (rechts)

Das Beispiel 2 enthält neben einer klassischen Datenbankbedingung *net* auch Score-basierte Bedingungen wie $a \approx high$ und gewichtete Operanden, etwa $\wedge_{\theta_k,\theta_p}$. Die Anfrage für dieses Beispiel wird wie folgt formuliert:

$$\{name | \exists a, net, k, p, f, m : notebook\,(name, a, net, k, p, f, m) \wedge a \approx high \wedge \\ \left(\neg net \vee \left(k \approx good \wedge_{\theta_k,\theta_p} p \approx low\right)\right) \wedge \left(net \vee \left(f \approx high \wedge_{\theta_f,\theta_m} m \approx high\right)\right)\}$$

CQQL baut, wie bereits erwähnt, auf der Quantenlogik auf. Die Konjunktion, Disjunktion und Negation werden somit nach den Gesetzen der Quantenlogik ausgewertet. In [SZ09] wird darauf verwiesen, dass CQQL das relationale Bereichskalkül verallgemeinert und die Gesetze der Booleschen Algebra erfüllt. Nachdem eine CQQL-Bedingung syntaktisch normalisiert wurde, kann sie mittels $eval\,(\varphi, o)$ für ein Objekt o, wobei φ die Bedingung darstellt, nach den folgenden rekursiven Regeln ausgewertet werden:

Definition 1 *[SZ09]*

$$
\begin{aligned}
eval\,(\varphi_1 \wedge \varphi_2, o) \quad &= eval\,(\varphi_1, o) * eval\,(\varphi_2, o) \\
eval\,(\varphi_1 \vee \varphi_2, o) \quad &= eval\,(\varphi_1, o) + eval\,(\varphi_2, o) - eval\,(\varphi_1, o) * eval\,(\varphi_2, o) \\
&\quad \textit{(wenn } \varphi_1 \textit{ und } \varphi_2 \textit{ nicht exklusiv)} \\
eval\,(\varphi_1 \vee \varphi_2, o) \quad &= eval\,(\varphi_1, o) + eval\,(\varphi_2, o) \\
&\quad \textit{(wenn } \varphi_1 \textit{ und } \varphi_2 \textit{ exklusiv)} \\
eval\,(\neg\varphi, o) \quad &= 1 - eval\,(\varphi, o)
\end{aligned}
$$

Eine Disjunktion ist exklusiv, wenn sie die Form $(\varphi \wedge \ldots) \vee (\neg\varphi \wedge \ldots)$ hat.

Gewichtung von Anfragebedingungen

Bei einer gewichteten Konjunktion $\varphi_1 \wedge_{\theta_1,\theta_2} \varphi_2$ oder Disjunktion $\varphi_1 \vee_{\theta_1,\theta_2} \varphi_2$ bestimmt die Gewichtsvariable θ_i den Einfluss der Bedingung φ_i. In CQQL wird die Summe der Operandengewichte θ_i nicht auf 1 eingeschränkt [SZ09]. Die genannten gewichteten Junktoren können mit der Definition 2 in logische Ausdrücke ohne Gewichte transformiert werden, wobei die Gewichtswerte in konstante Score-Werte umgesetzt werden. Die Gewichtung erfolgt aufgrund der linearen Auswertung der Bedingungen (siehe Definition 1) auch linear. Das Neuartige dieses Gewichtungsansatzes liegt in der ausschließlichen Verwendung der Logikregeln [SZ09]. Im Gegensatz zu bereits existierenden Ansätzen zur Gewichtung, wie z.B. von Fagin und Wimmers [FW00], wird die arithmetische Auswertung nicht oberhalb der Anfragesprache ausgeführt. Dadurch ist eine Optimierung durch die Verwendung von beispielsweise Assoziativitäts- und Distributivitätsregeln überhaupt möglich [Sch07]. Die Abbildung 2.5 zeigt den Transformationsschritt anhand des Beispiels 2 mit dem Notebook.

Definition 2 *[SZ09]*

$$
\begin{aligned}
\varphi_1 \wedge_{\theta_1,\theta_2} \varphi_2 &:= \quad (\varphi_1 \vee \neg\theta_1) \wedge (\varphi_2 \vee \neg\theta_2) \\
\varphi_1 \vee_{\theta_1,\theta_2} \varphi_2 &:= \quad\quad (\varphi_1 \wedge \theta_1) \vee (\varphi_2 \wedge \theta_2)
\end{aligned}
$$

Nach der Normalisierung und der Anwendung der Regeln aus den Definitionen 1 und 2 ergibt die arithmetische Auswertung des Beispiels 2:

$$
\begin{aligned}
eval\left(q^{\Theta}, o\right) = a\left(net * kp + \overline{net} * fm\right) &\text{ mit} \\
kp = \left(k + \overline{\theta_k} - k\overline{\theta_k}\right)&\left(p + \overline{\theta_p} - p\overline{\theta_p}\right) \\
fm = \left(f + \overline{\theta_f} - f\overline{\theta_f}\right)&\left(m + \overline{\theta_m} - m\overline{\theta_m}\right)
\end{aligned}
$$

Präferenzen

Für den Nutzer ist das Angeben von Präferenzen einfacher als das konkrete Setzen von Gewichten in der Anfrage. In diesem Ansatz, wie bereits beschrieben, gibt der Anwender deshalb für jeweils zwei Objekte $o_1, o_2 \in O$ seine Präferenz $o_1 \geq o_2$ bezüglich der Anfrage q^{Θ} an. Er entscheidet also, welches Objekt o_1 die Anfrage besser erfüllt als ein anderes Objekt o_2. Eine Gewichtungsfunktion w erfüllt eine Präferenz $p \in P$, wenn

$$eval\left(q^{\Theta}, o_1, w\right) - eval\left(q^{\Theta}, o_2, w\right) = d \geq 0$$

gilt. Die Präferenzmenge P für q^{Θ} heißt *konsistent*, wenn ihr reflexiver und transitiver Abschluss antisymmetrisch ist, also keine Zyklen wie $o_1 \geq o_2$ und $o_2 \geq o_1$ mit $o_1 \neq o_2$ enthält. Eine Präferenz kann in eine der folgenden drei Kategorien eingeordnet werden. Abbildung 2.6 zeigt dieses schematisch anhand zweier Gewichte θ_1, θ_2, die den Hyper-Einheitswürfel aufspannen. Die Präferenz ist in diesem Fall als Einheitsebene eingezeichnet.

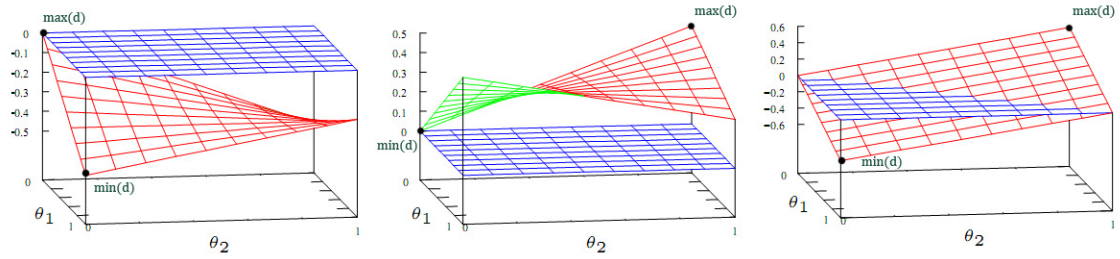


Abbildung 2.6: Kategorien der Präferenz $o_1 \geq o_2$ bezogen auf die 0-Hyperebene, nach [SZ09]

nicht erfüllbare Präferenz: Die Präferenz kann durch keine Gewichtung von q^{Θ} erfüllt werden. Es gilt $max(d) < 0$. (Abbildung 2.6 links)

nutzlose Präferenz: Die Präferenz kann nicht für das Lernen von Gewichten verwendet werden, da unabhängig von den Gewichtswerten für q^{Θ} die Präferenz immer erfüllt ist. Es gilt $min(d) \geq 0$. (Abbildung 2.6 mittig)

nützliche Präferenz: In den übrigen Fällen kann die Präferenz für den Lernalgorithmus von Gewichten, also zum Modifizieren von w, genutzt werden. Es gilt $min(d) < 0$ und $max(d) \geq 0$. (Abbildung 2.6 rechts)

Ein Beispiel für eine nutzlose Präferenz $o_1 \geq o_2$ wäre eine Anfrage, die aus einer gewichteten Konjunktion mit zwei Bedingungen besteht. Objekt o_1 besitzt in beiden Bedingungen

höhere Score-Werte als Objekt o_2. Egal, wie die Gewichtung gesetzt wird, o_1 ist immer besser als o_2. Die Angabe der Präferenz $o_1 \geq o_2$ ist somit unbrauchbar für das Lernen von Gewichten. Würde in diesem Beispiel hingegen die Präferenz $o_2 \geq o_1$ angegeben werden, so kann diese nie erfüllt werden.

Nutzlose oder nicht erfüllbare Präferenzen sind für den Lernalgorithmus von Gewichten nicht hilfreich bzw. erzeugen Widersprüche. Aus diesem Grund sollen sie während der Nutzerinteraktion entfernt werden. Um unterscheiden zu können, welche Präferenzen nützlich sind oder nicht, werden $max(d)$ und $min(d)$ gesucht. In [SZ09] kann der Algorithmus nachgelesen werden, der für eine ungeschachtelte Anfrage für eine Präferenz diese Werte ermittelt.

Umwandlung von Präferenzen in Gewichte

Der Lernalgorithmus *prefsToWeight* bildet mit der Eingabe von einer gewichteten Anfrage q^{Θ}, einer konsistenten Menge von Präferenzen, die in Gewichte umgewandelt werden sollen, und der Anfrageobjektmenge O eine entsprechende angepasste gewichtete Anfrage [SZ09]. Die verschiedenen Präferenzen werden so zusammengefasst:

$$\min_{(o_i \geq o_j) \in P} \left(eval\left(q^{\Theta}, o_i, w\right) - eval\left(q^{\Theta}, o_j, w\right) \right) \geq 0$$

Es soll die Gewichtungsfunktion w gefunden werden, die die obige Zielfunktion maximiert [SZ09]. Aufgrund der schweren Berechenbarkeit eines nichtlinearen Optimierungsfalls, kann hier abgeschwächt werden, dass nicht unbedingt das Maximum gefunden werden muss, sondern auch ein w genügt, das alle Präferenzen beachtet. Anwendung findet hier der Downhill-Simplex-Algorithmus von Nelder und Mead [NM65], mit dem sich durch die Verwendung unterschiedlicher, zufälliger Startwerte dem Maximum der Zielfunktion angenähert wird. Aufgrund der endlichen Ausführung des Algorithmus erhält man nur eine Approximation des Maximums. Abbildung 2.7 stellt die gültigen Gewichtswerte als Schnitt von Präferenzen dar. Um w möglichst eindeutig zu bestimmen und den Nutzer so wenig wie nötig zu belasten, ist die minimale Anzahl an Präferenzen gesucht, die die kleinste nichtleere Schnittfläche hervorrufen.

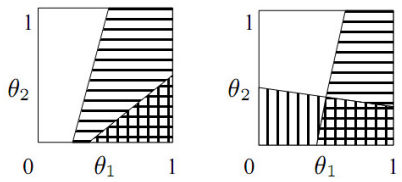


Abbildung 2.7: Implikation (links) und Überlappung (rechts) von Präferenzen [SZ09]

2.2.3 Relevance-Feedback in CQQL-Systemen

Eine Anfrage wird in CQQL mithilfe von Gewichten auf Bedingungen dem subjektiven Ähnlichkeitsempfinden des Nutzers angepasst. Die Gewichte können jedoch aus den folgenden Gründen nicht vollständig vom Anwender gesetzt werden [ZL08]:

- **Unklarheit über das Suchziel:** Der Nutzer weiß nicht, welche Bedingungen ihm bei der Anfrage besonders wichtig sind. Der ihm nicht bekannte Datenbestand kann u.a. ein Grund dafür sein.
- **Fehlannahmen über das konzeptionelle Modell:** Der Nutzer hat über die Systemauswertung der Gewichte eine falsche Vorstellung. Das mentale Modell, wie der Nutzer sich die Funktionsweise des Systems vorstellt, ist hier irrtümlich vom konzeptionellem Modell, wie das System wirklich reagiert, verschieden.
- **Komplexität der Gewichte:** Bei komplexen Anfragen ist das Setzen von Gewichten kaum mehr überschaubar.
- **unerwartete Wechselwirkungen:** Zwischen den einzelnen Bedingungen können sich Wechselwirkungen ergeben, bei denen sich die Ergebnisse der Suche von dem Änhlichkeitsempfinden des Nutzers entfernen.

Aufgrund dessen wird dem Nutzer das Setzen von Gewichten durch das System abgenommen. Es wird zu Beginn der Suche eine initiale Gewichtungsfunktion verwendet, die meist jedoch nicht die vom Nutzer gewünschten Resultate liefert. Die allgemeine Reaktion eines Anwenders auf ein nicht zufriedenstellendes Ergebnis könnte das Browsing, also das sequentielle Durchsuchen der Ergebnisse, sein. Die Anfrage könnte auch manuell modifiziert werden, was, wie bereits oben beschrieben, eine schwierige Angelegenheit für einen Laien sein kann. Schließlich kann auch die Anfrage automatisch angepasst werden. Der Nutzer gibt hierfür dem System dem eigenen Empfinden entsprechend die Relevanz der Ergebnisse für seine Suchanfrage an. Aufgrund der Rückkopplung der Benutzerrelevanz zum System wird dieser Prozess als Relevance-Feedback bezeichnet. Es gibt allgemein verschiedene Varianten, wie die Anfrage modifiziert werden kann. In [Sch04] werden u.a. die Verschiebung des Anfragepunktes und die Modifzierung der Gewichte von Anfragetermen als Möglichkeiten genannt. Rocchio, auf den das Verfahren des Relevance-Feedback zurückzuführen ist, entwickelte im IR das Verfahren für das Verschieben des Anfragepunktes im Vektorraum in Richtung der relevanten Ergebnisse und weg von den als irrelevant markierten Dokumenten [Roc71]. Der Einfluss der beiden Dokumentgruppen auf die Verschiebung des Anfragevektors wird hierbei gewichtet. In einem CQQL-System wird hingegen im Zuge des Relevance-Feedbacks eine Anpassung der Gewichte der einzelnen Bedingungen vorgenommen [SZ09]. Wie bereits beschrieben, bleiben dem Nutzer die Gewichte verborgen,

stattdessen arbeitet dieser auf Präferenzen. Die Abbildungen 2.4 und 2.8 stellen diesen Ansatz vereinfacht dar.

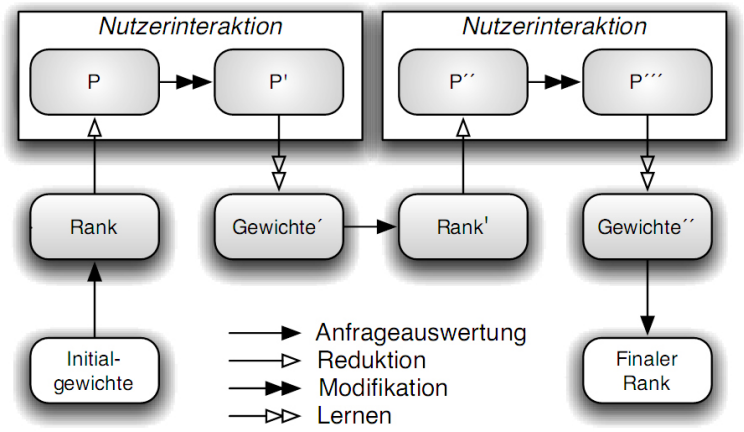


Abbildung 2.8: Verfeinerung von Gewichten durch Nutzerinteraktion mittels Präferenzen P [SZ09]

Eine Anfrage, die zu Beginn der Suche die initiale Gewichtungsfunktion w^I verwendet, wird ausgewertet. Es entsteht eine total geordnete Menge der Ergebnisse (Rank), die dem Nutzer jedoch nicht vollständig präsentiert wird. Eine charakteristische Präferenzmenge P wird aus den ersten k Objekten des Ranks abgeleitet. Diese Menge enthält keine reflexiven und transitiven sowie keine nutzlosen und nicht erfüllbaren Präferenzen. Außerdem ist sie auf das Minimum reduziert. Mit der Präferenzmenge P sind die Top-k Elemente eindeutig bestimmt. Auf diesen Systempräferenzen gibt der Nutzer seine eigenen Präferenzen P' an. Der Anwender kann für eine vom System angegebene Präferenz $p \in P : o_1 \geq o_2$ *für* $o_1, o_2 \in O$ zwischen den folgenden Varianten wählen [SZ09]:

- **Bestätigung:** $o_1 \geq o_2$ muss nicht modifiziert werden, die Systempräferenz entspricht hier der Nutzerpräferenz.
- **Umkehrung:** Präferenz wird in $o_2 \geq o_1$ umgekehrt, weil sie der Anfrage widerspricht.
- **Entfernung:** Wenn zwischen zwei Objekten keine Auswahl getroffen werden kann $(o_1 = o_2)$, welches Objekt besser ist, dann kann die Präferenz entfernt werden.
- **Erstellung:** $o_2 \geq o_3$, *wobei* $o_2 \geq o_3, o_3 \geq o_2 \notin P$. Es wird eine neue Präferenz vom Nutzer hinzugefügt.

Während der Nutzerinteraktion werden die Präferenzen auf Widersprüche und Nützlichkeit geprüft (siehe weiter oben). Tritt ein Zyklus $o_1 \geq o_2$ und $o_2 \geq o_1$ mit $o_1 \neq o_2$ in

der Halbordnung auf, so ist eine automatische Entfernung der Präferenzen nicht ratsam, da das Ergebnis eventuell stark verzerrt werden kann [ZL08]. Dem Nutzer soll stattdessen der Konflikt angezeigt werden, sodass dieser ihn manuell entfernen kann. Die veränderten Präferenzen werden dann in Gewichte umgesetzt, die einen neuen Rank $Rank'$ erzeugen. Aus diesem werden dann wieder die Präferenzen P'' abgeleitet, die den Top-k-Rank erzeugen. Dieser beschriebene Prozess wird solange iterativ fortgesetzt bis der Nutzer zufrieden ist mit den abgeleiteten Präferenzen P^n oder die Suche einfach abbricht [SZ09].

Wichtig bei der Angabe von Präferenzen, die aus dem Rank ermittelt werden, ist die Größe von k [SZ09]. Um eine Anfrage so schnell und so gut wie möglich durch das System anpassen zu lassen, ist eine große Ergebnismenge nötig. Der Nutzer müsste dann aber auch viele Präferenzen bewerten. Das kann ihm nicht zugemutet werden. Die Wahl von k muss deswegen gut überlegt sein.

3 Bewertung von GUIs existierender visueller MMR-Systeme

In dieser Bachelorarbeit soll eine grafische Benutzeroberfläche (GUI) für visuelle MMRS entwickelt werden. Der Schwerpunkt soll in der inhaltsbasierten Suche von Bildern liegen, eine wesentlich geringere Bedeutung soll hier dem Video-Retrieval zukommen. Aus diesem Grund werden in diesem Abschnitt einige GUIs bereits existierender Content-Based-Image-Retrieval-Systeme (CBIRS) vorgestellt und beurteilt. Im Anschluss wird auch die Oberfläche eines Systems für die Videosuche bewertet. Es wird von einer bestehenden Datenbasis von Bildern bzw. Videos ausgegangen, die bereits beim Einfügen in die Datenbank vorverarbeitet und aus denen anschließend die Features extrahiert wurden. Eine Evaluierung der Suchergebnisse und der Laufzeit der Suchalgorithmen der Systeme ist nicht Thema dieser Arbeit.

Seit den frühen 1990er Jahren ist die Entwicklung von CBIRS ein aktives Forschungsfeld. Viele kommerzielle und nicht kommerzielle Systeme für die Bildersuche sind entstanden. Die meisten von ihnen unterstützen eine Suche nach einigen oder mehreren der folgenden Optionen [RHC99]:

- Browsen
- Suche anhand eines Beispielbildes, genannt Query by Example (QBE)
- Suche anhand einer Skizze, genannt Query by Sketch
- Suche anhand von Text (beinhaltet Schlüsselwörter)
- Navigation mittels Bildkategorien

In [VT02] werden einige CBIRS vorgestellt, leider existieren viele der Internetreferenzen der Systeme aus dem Jahr 2002 nicht mehr. Seitdem sind viele neue Programme entworfen, jedoch nur einige wenige existierende Systeme verbessert und die meisten nicht mehr weiter entwickelt worden [Kos09]. Es folgt eine Beurteilung der Umsetzung der Suchmöglichkeiten, der Ergebnispräsentation und des Relevance-Feedbacks einiger heutzutage existierender Systeme.

3.1 QBIC

QBIC (Query by Image Content) ist das erste kommerzielle Content-Based-Image-Retrieval-System [Käs05]. Es entstand am IBM Almaden Research Center in San Jose, begonnen wurde mit der Entwicklung 1992. Als so genannter Image Extender wurde es in das Datenbanksystem DB2 integriert [IBM09]. Es ermöglicht die Suche anhand eines Textes oder Beispielbildes. Der Nutzer kann auch Skizzen konstruieren, Farben, Texturen und deren Position als Anfrage angeben. Eine automatische Anfrageverfeierung, die auf Bildbewertungen des Nutzers beruht, wird nicht unterstützt [FSN+95].

Einen kleinen Einblick in die Funktionalität von QBIC erhält man bei der Suche von Kunstwerken des State Hermitage Museum unter [Sta03]. Es werden hier drei Varianten der Suche bereitgestellt: *Browse*, *QBIC Colour and Layout Searches* und *Advanced Search*. Beim Browsen werden 12 Kategorien (Paintings, Sculpture, Archaeological Artifacts, ...) übersichtlich in zwei 6er Spalten angeordnet. Die Einordnung des zu Suchenden in eine der Gruppen verläuft schneller aufgrund der Skizzen, die neben dem Kategorienamen stehen. Nachdem eine Unterkategorie über Radio-Buttons selektiert wurde, öffnet sich ein Popup-Fenster. Aus einer alphabetisch sortierten Liste, die z.B. Herkunftsländer enthält, muss wieder eine Auswahl getroffen werden. Die Ergebnisse werden als kleine Vorschaubilder, so genannte Thumbnails, präsentiert. Das ermöglicht einen Überblick über viele Ergebnisse. Desweiteren stehen Titel und die zeitliche Einordnung des Werkes rechts neben dem Bild. Nach einem Klick auf ein ausgewähltes Bild der Ergebnisliste oder dessen Titel wird das Bild vergrößert angezeigt. Dieses zeigt Abbildung 3.1. Will man einen bestimmten Bereich des Bildes nochmals vergrößern, so klickt man auf das Zentrum der zu vergrößernden Fläche. Ein weiterer Klick auf den Zoom-Button ist nötig, damit das mit grün umrahmte Gebiet vergrößert angezeigt wird.

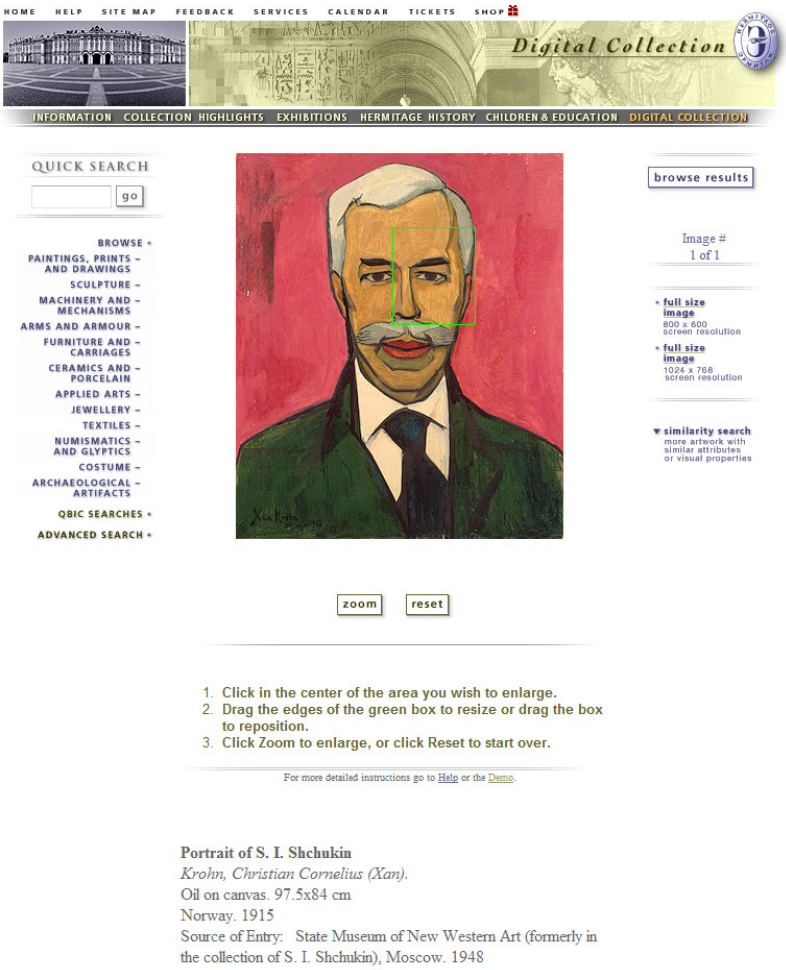
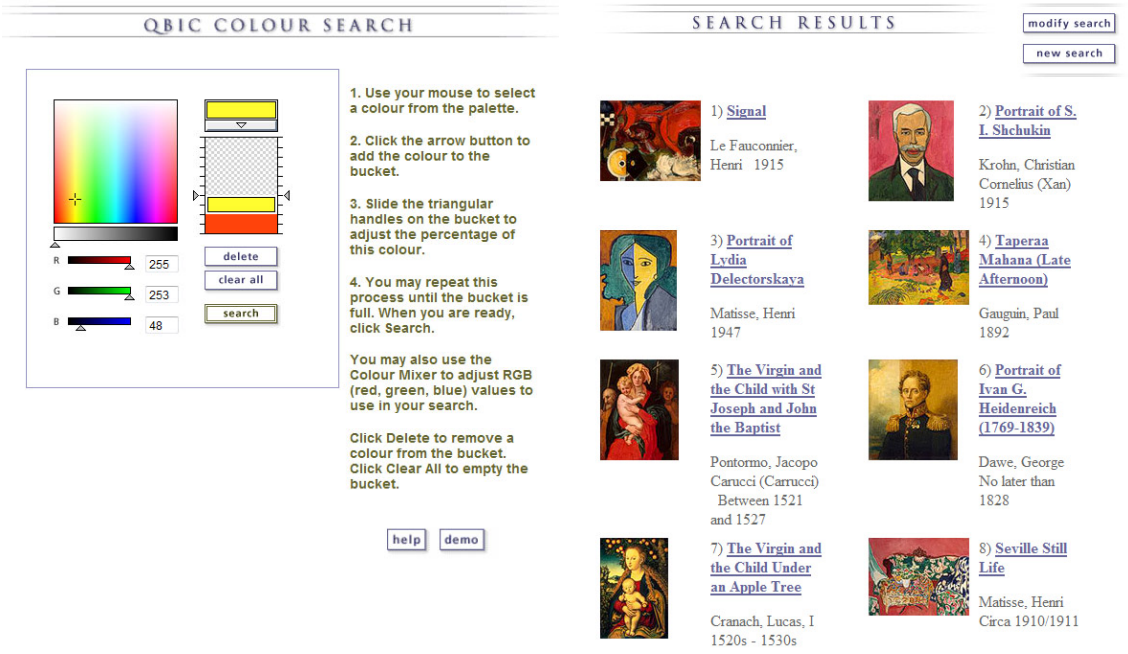


Abbildung 3.1: Ansicht eines Ergebnisses der Suche unter Browse, grüner Rahmen zur Anzeige des zu zoomenden Bereichs

Im Advanced Search-Modus kann der Nutzer zusätzlich zu den im *Browse*-Modus bereitgestellten Suchkriterien weitere spezifizieren wie beispielsweise Künstler, Titel oder Kunstrichtung.

Das Innovative der Suche in der digitalen Kunstsammlung des State Hermitage Museum ist das von IBM bereitgestellte QBIC Colour and Layout Searches. In der Abbildung 3.2 ist die Oberfläche der inhaltsbasierten Bildersuche von QBIC nach Farben dargestellt. Um den Anwender in der Benutzung zu unterstützen wird ihm zusätzlich zu den Bedienungshinweisen in Form von Text auch eine Animation bereitgestellt. Farben können entweder aus einer Farbpalette oder bei der Suche nach einer ganz bestimmten Farbe mittels der drei Schieberegler aus dem RGB-Farbraum ausgewählt werden. Es können bis zu 5 Farben



(a) Suche anhand dominanter Farben (b) Ausschnitt der Ergebnisliste der Suche

Abbildung 3.2: QBIC Colour Search in der digitalen Kunstsammlung des State Hermitage Museum

angegeben werden, die dominant in dem Anfragebild auftreten sollen. Die ausgewählten Farben sind in einer, einer Pegelstandanzeige ähnlichen, Palette aufgelistet. Die andere Variante der inhaltsbasierten Suche ist QBICs Layout Search. Darunter ist die Suche anhand einer benutzerspezifizierten Eingabe von Farben und Formen, die zueinander lokal in Relation stehen, zu verstehen. Abbildung 3.3 verdeutlicht dieses. In gleicher Weise, wie zuvor beschrieben, können hier die Farben selektiert werden. Der Nutzer kann mit Rechtecken und Ellipsen sein Anfragebild konstruieren. Ist ein Element durch einen Klick markiert, so kann es verschoben werden. Mittels zweier Button kann außerdem die Reihenfolge der Darstellung der Figuren verändert werden.

Bewertung von QBIC

Allgemein kann gesagt werden, dass die Suche von Kunstwerken des State Hermitage Museums, insbesondere QBIC Colour und Layout Searches, leicht verständlich ist. Animationen und zusätzliche textuelle Beschreibungen stehen dem Suchenden als Hilfe bereit.

(a) Suche anhand eines gezeichneten Bildes

(b) Ausschnitt der Ergebnisliste der Suche

Abbildung 3.3: QBIC Layout Search in der digitalen Kunstsammlung des State Hermitage Museum

Fehleingaben des Nutzers treten kaum auf, da Metadaten als Suchkriterien hauptsächlich nur aus Vorgaben gewählt werden können. Um den Anwender nicht mit vielen Auswahlschemen zu konfrontieren, wäre beim Browsen und der fortgeschrittenen Suche anstelle von Popup-Fenstern eine Integration der Auswahlmöglichkeiten in die Webseite besser gewesen. Als verbesserungswürdig sei auch zu nennen, das unter der Suche nach dem Layout nicht erkennbar ist, ob Rechteck oder Ellipse momentan eingestellt ist. Desweiteren ist der Nutzer in seiner Anfrage sehr eingeschränkt. Ihm stehen nur zwei Formen zum Zeichnen zur Verfügung. Ein bereits existierendes Bild als Anfrage kann nicht zur Suche genutzt werden und die Kombination exakter und inhaltsbasierter Bedingungen ist nicht möglich. Die Ergebnisse der Anfrage werden übersichtlich in einer Liste anhand der Ähnlichkeit geordnet präsentiert, jedoch ist nicht der Grad der Anfrageerfüllung gegeben. Das gesamte Bild oder ein Teil kann vergrößert werden. Die Zoom-Funktion, die nur für einen Ausschnitt vorgegebener Größe angewendet werden kann, bedarf zwei Schritte. Der Nutzer

möchte, um sein Ziel zu erreichen, möglichst wenige Schritte abarbeiten [Gal07]. Aus diesem Grund wäre das Vergrößern auch in einem Schritt denkbar, wobei dem Nutzer schon beim Bewegen der Maus über das Bild der Rahmen angezeigt werden könnte. Um benachbarte Bereiche zu vergrößern, muss der Nutzer zunächst den Reset-Button betätigen, um anschließend den naheliegenden Bereich zoomen zu können. Das hätte in weniger Schritten gelöst werden können. Sind die Bilder nicht zufriedenstellend, so könnte der Suchende durch einen Klick auf den modify search-Button seine Anfrage modifizieren. Diese Funktionalität scheint jedoch nicht implementiert zu sein, da man auf der Ergebnisseite verbleibt. Eine automatische Anfrageverfeinerung wird allgemein bei der Suche nicht bereitgestellt.

3.2 FIRE

FIRE (Flexible Image Retrieval Engine) wurde von Thomas Deselaers in Kooperation mit anderen Personen der Human Language Technology and Pattern Recognition Group der RWTH Aachen Universität entwickelt [Des09]. Im Folgenden wird sich auf die Online-Demonstration von FIRE bezogen, die unter [Des05] zu finden ist.

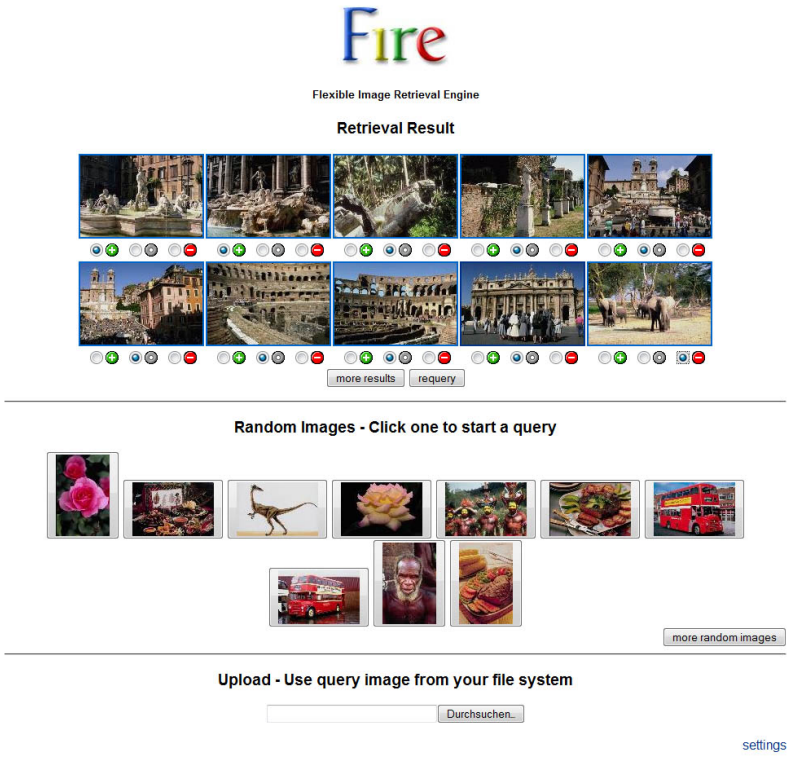


Abbildung 3.4: FIRE - Demo, Ergebnisliste der Suche anhand des Bildes links oben

Die Abbildung 3.4 zeigt das Webinterface von FIRE. Gesucht werden kann anhand eines Bildes, das aus einer zufällig generierten Menge von Fotos einer bereitgestellten Bilderkollektion stammt oder selbst hochgeladen wurde. Die als ähnlich eingestuften Bilder werden als Thumbnails in einer Liste präsentiert. Durch einen Klick auf das entsprechende Foto öffnet sich in einem neuen Browserreiter eine vergrößerte Bildansicht sowie ein RGB Farb- und Tamura-Texturhistogramm. Die Anzahl der gezeigten Ergebnisse kann unter settings verändert werden. FIRE unterstützt Relevance-Feedback. Der Nutzer kann mithilfe von Radiobuttons ein Bild als relevant (+), neutral oder nicht relevant (-) bewerten. Standardmäßig sind alle Ergebnisse als neutral ausgewählt, nur das oberste linke Bild ist relevant, da dieses das Anfragebild darstellt. Der Nutzer sucht römische Brunnen, er bewertet deshalb das Bild mit den Elefanten als nicht relevant. Mit einem Klick auf den requery-Button wird die Anfrage automatisch den Benutzerbewertungen angepasst. Eine neue Ergebnisliste wird präsentiert.

Bewertung von FIRE

FIRE bietet eine einfache Webbedienoberfläche zur Suche anhand eines Bildes. Um das Programm zu testen, steht eine Bilddatenbank zur Verfügung. Die angebotene Funktionalität des Hochladens eines eigenen Suchbildes funktioniert jedoch nicht. Negativ zu bewerten ist auch die fehlende Unterstützung einer Suche anhand von Metadaten wie z.B. Fotograf, Bildgröße oder Schlüsselwörter. Die Suchfunktion ist auf die Variante Query by Example eingeschränkt. Die berechneten ähnlichen Bilder werden in einer Liste dargestellt, jedoch bei unterschiedlich auftretenden Formaten sind diese etwas unübersichtlich angeordnet. Dem Nutzer werden auch keine für ihn interessanten Details über das Bild geliefert. Das Farbhistogramm oder die Tamura-Werte sind für den Laien nicht relevant. Desweiteren ist das Anfragebild in der Ergebnisliste auf den ersten Blick nicht als solches erkennbar. Die Reihenfolge der Bilder kann aufgrund der westlichen Leserichtung erahnt werden. Inwieweit ein Bild dem Anfragebild entspricht, wird jedoch nicht deutlich. Positiv zu nennen ist die Möglichkeit der Bewertung der Ergebnisse, die aufgrund der Symbole und der Farbuntermalung selbsterklärend ist. Eine gleiche Bewertung mehrerer Bilder ist nicht in wenigen Schritten realisierbar. Das kann dazu führen, dass der Nutzer nur wenige Evaluierungen vornimmt, da ihm der Aufwand sonst zu groß ist. Ein Video über die Interaktion mit FIRE wird auf [Des09] bereitgestellt. Eine Hilfe auf der Demonstrationsseite wird dem Anwender jedoch nicht angeboten. Eine Verlinkung zu dem Video und zusätzliche Informationen in Form von Text wären möglich.

3.3 GIFT/VIPER

The GNU Image Finding Tool (GIFT), welches den Suchmechanismus Visual Information Processing for Enhanced Retrieval (VIPER) als Plugin integriert [Fre05], ist eine Forschungsentwicklung an der Universität Genf. Eine Online-Demonstration steht unter [Cha09] zur Verfügung, die in diesem Absatz beurteilt wird.

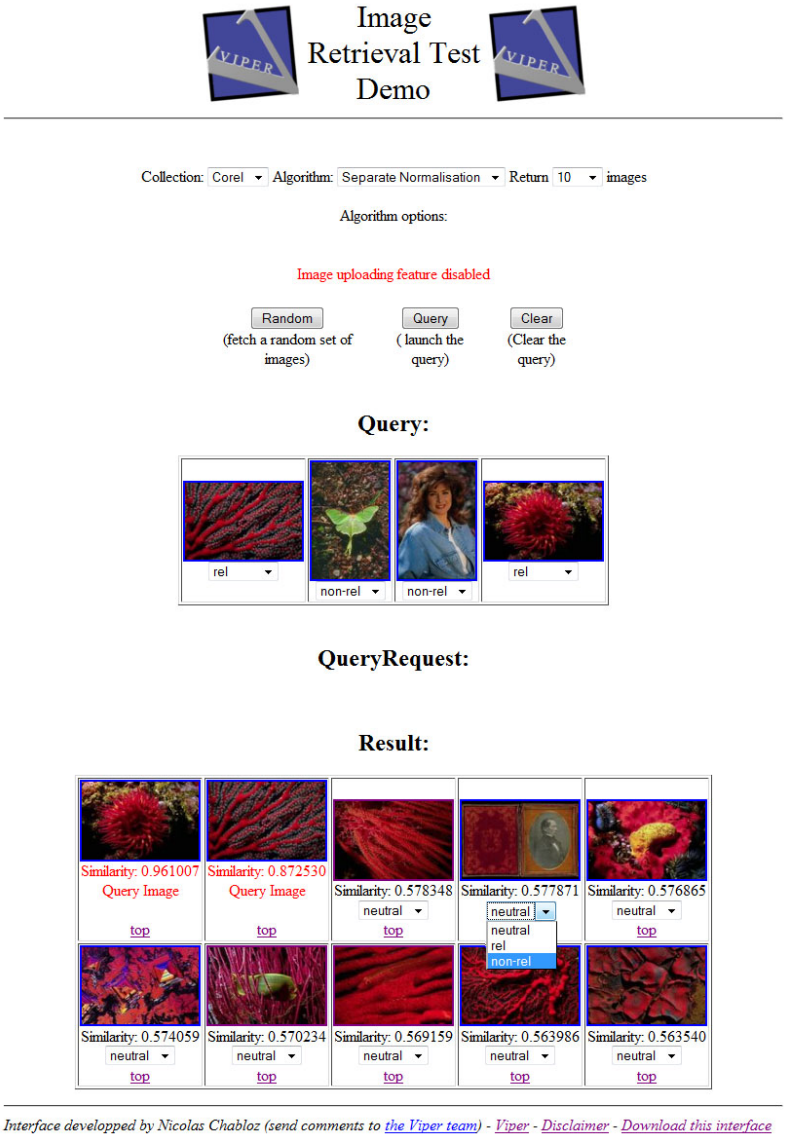


Abbildung 3.5: GIFT - Demo

Die Suche ist anhand eines oder mehrerer Beispielbilder möglich, welche sich aus der bereitgestellten Bilddatenbank auswählen lassen. Sie müssen als relevant eingestuft werden, damit beim Klick auf den Query-Button diese als Anfragebilder übernommen werden. Der Nutzer kann zwischen *Classical IDF* und *Separate Normalisation* als Verfahren für die Feature-Gewichtung und Ähnlichkeitsberechnung wählen. Classical IDF wird auch im Information-Retrieval angewendet, dort werden alle Features zusammen gewichtet und normalisiert. Beim Separate Normalisation werden hingegen die verschiedenen Feature-Gruppen wie z.B. Farb- und Texturhistogramm getrennnt gewichtet und normalisiert und die Ergebnisse anschließend zusammengefasst. Es hat sich gezeigt, dass die zweite Gewichtungsvariante bessere Resultate erzielt als die andere [Kos09]. Die Ergebnisbilder werden verkleinert in einer Liste dargestellt. Der Grad der Anfrageerfüllung wird über eine Zahl angegeben, wobei eine große Zahl auf eine starke Ähnlichkeit hinweist. Die Anfragebilder befinden sich am Anfang der Ergebnisliste. Ein Bild wird vergrößert in einem neuen Reiter des Browsers angezeigt, nachdem es angeklickt wurde. Die Anzahl der zurückgelieferten Bilder kann der Nutzer im oberen Bereich der Seite aus einer Combobox auswählen. GIFT ermöglicht dem Benutzer eine automatische Anfrageverfeinerung. Es gibt drei Bewertungsstufen: relevant, neutral und nicht relevant. Die Anfrage wird durch einen Klick auf den Query-Button modifiziert. Die als relevant oder nicht relevant bewerteten Fotos werden oberhalb der neuen Ergebnisse positioniert. Klickt man auf den Random-Button, so werden die bisherigen Bewertungen gelöscht und eine neues Anfragebild muss ausgesucht werden.

Bewertung von GIFT

GIFT in der Online-Demo-Version realisiert nur eine Suche anhand von Beispielbildern aus der vorgegebenen Datenbank. Im Gegensatz zu anderen Programmen können hier mehrere Bilder die Anfrage darstellen, indem sie als relevant beurteilt werden. Diese sind für den Benutzer aufgrund der roten Schrift deutlich von den anderen Fotos unterscheidbar. Die Thumbnails können aufgrund der einheitlichen Ausrichtung nach Spalten und Zeilen schnell überblickt werden. Zusätzliche Informationen zum Bild neben dem Ähnlichkeitsgrad wären wünschenswert. Eine Vergrößerung der Ansicht, die jedes Mal einen neuen Browserreiter öffnet, ist nicht komfortabel für die Suche. Das Programm stellt dem Nutzer Relevance-Feedback zur Verfügung. Die Auswahlmöglichkeiten in der Combobox unterhalb des Bildes sind jedoch nicht für jeden verständlich. Die Abkürzungen *rel* und *non-rel* hätten als Vermerk einmalig aufgeführt werden können. Positiv ist die räumliche Abgrenzung der bereits bewerteten Bilder von der Ergebnisliste. Eine Anpassung im Nachhinein ist so noch möglich. Nach der Modifizierung der Anfrage kann das Anfragebild einen Ähnlichkeitswert haben, der größer oder kleiner 1 ist. Der Nutzer vermutet, dass das

Anfragebild aufgrund der absoluten Übereinstimmung den maximalen Ähnlichkeitswert erhalten müsste. Der maximale Wert scheint jedoch nicht festgelegt zu sein. Es fehlt außerdem eine Hilfe, die beispielweise die zusätzlichen Optionen bei Auswahl von *Classical IDF* beschreibt.

3.4 Emir

Die Entwicklung von Experimental Metadata Based Image Retrieval (Emir) begann Mathias Lux 2002 am Institut für Informationssysteme und Computer Medien an der Technischen Universität Graz [Lux07]. Werner Klieber und Margit Lang entwickelten später an dem Projekt mit. Mathias Lux hat außerdem das zu Emir zugehörige Programm Caliph (Common And Light-Weight Photo Annotation) entworfen. Die Software kann unter [Sou09] heruntergeladen werden.

Mit Emir können Bilder aus einem selbst gewählten Verzeichnis gesucht werden. Um eine Suche überhaupt zu ermöglichen, muss in diesem Ordner zu jedem Foto die entsprechende MPEG-7 Datei liegen, die neben den manuell eingegebenen und von der Kamera erstellten Metadaten auch die automatisch extrahierten Features enthält. MPEG-7 als Multimedia Content Description Interface wurde von der Moving Picture Experts Group (MPEG) entwickelt. Im Gegensatz zu MPEG-1, MPEG-2 und MPEG-4 ist MPEG-7 nicht für die Kompression von Video- und Audiodaten ausgelegt, es ist stattdessen ein Standard zur Beschreibung von Multimedia-Inhalten.

Mit Caliph werden die für die Suche in Emir nötigen MPEG-7 Dateien erzeugt. Es können vorhandene Metadaten aus dem EXIF-, IPTC IIM- und XMP-Format extrahiert und in eine MPEG-7 Datei konvertiert werden [Lux05b]. Im Exchangeable Image File Format (EXIF) werden die von der Digitalkamera erstellten Informationen zum Bild wie z.B. die Blendeneinstellung, die Belichtungszeit und die Brennweite abgespeichert. Der von der International Press Telecommunications Council (IPTC) spezifizierte Multimedia-Standard Information Change Model (IIM) wird seit 1997 nicht mehr weiterentwickelt [Int09], stattdessen wird heutzutage eher auf den Standard Extensible Metadata Platform (XMP) gesetzt. Desweiteren kann der Nutzer bei Caliph selbst Metadaten hinzufügen. Neben der Eingabe eines Freitextes oder strukturierter Daten ist auch das Erstellen von gerichteten Graphen möglich. Es werden außerdem auch die Dominant Color-, Color Layout-, Scalable Color- und Edge Histogram-Deskriptoren des MPEG-7-Standards aus den Bildern extrahiert [Lux07]. Mit dem Dominant Color-Deskriptor werden die repräsentativen Farben im Bild beschrieben. Der Color Layout-Deskriptor charakterisiert die Positionierung von Farben im Bild und Scalable Color ist durch ein Farbhistogramm im HSV-Farbraum gekenn-

zeichnet. Das Edge Histogram beschreibt die räumliche Anordnung von Kanten [MSS02].

Emir unterstützt 3 Suchverfahren, die durch die Tabs *Index*, *Graph* und *Image* voneinander getrennt werden. Um eine schnelle Suche zu ermöglichen, ist ein Index zu erstellen. Unter dem Tab Index kann der Nutzer Bilder anhand von Schlüsselwörtern suchen. Werden mehrere Begriffe eingegeben, unabhängig davon, ob diese mit AND, OR oder NOT verbunden sind, so wird die Anfrage intern als eine OR-Verknüpfung verarbeitet. Eine erweiterte Suche kann durch das Öffnen des Keyword-Tabs über den Menüeintrag *View* vorgenommen werden. Die Abbildung 3.6 zeigt den Tab Keywords. Dort kann u.a. ausgewählt werden, wo die Suchwörter vorkommen müssen, ob diese mit AND- oder OR verknüpft werden, welche Bildqualität gefordert wird und wie genau gesucht werden soll, also exaktes Matching, Minimum- oder Maximum-Erfüllung.

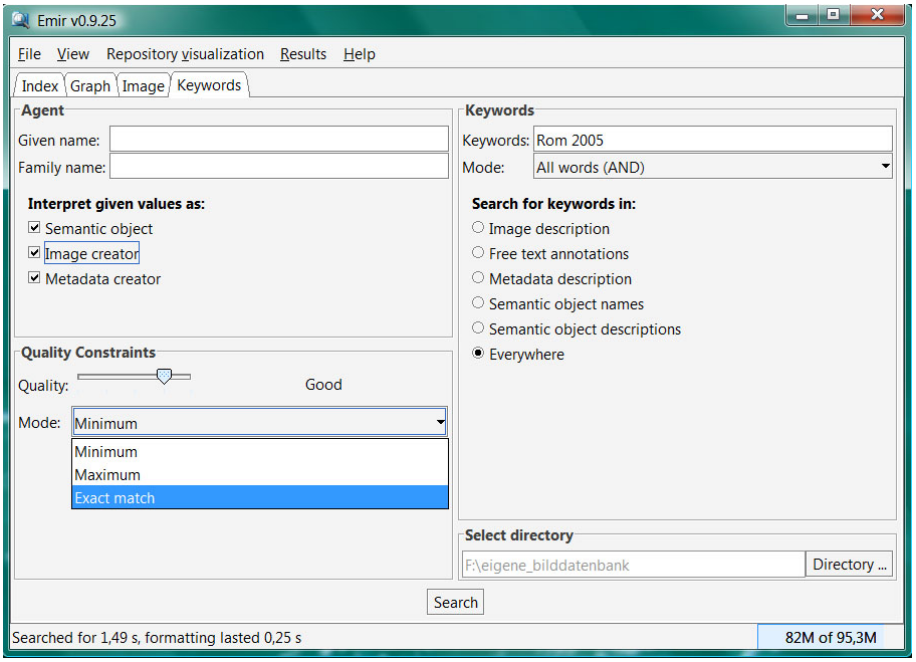


Abbildung 3.6: Emir - Suche anhand von Schlüsselwörtern

Unter dem Tab Graph kann der Nutzer einen Anfragegraphen A im Editor erstellen, nach dem alle Graphen G gesucht werden, für die A ein Teilgraph ist [LG05]. Die Knoten stellen semantische Objekte dar und die Kanten semantische Relationen. Im Textfeld wird der Knotennamen eingegeben, bei Verwendung der Tilde $\sim$ im Namen werden auch ähnliche Begriffe bei der Suche beachtet. Durch die Enter-Eingabe oder durch einen Klick auf den Create node-Button wird der Knoten im unteren Bereich erzeugt. Die Zahl in Klammern neben dem Knotennamen gibt die Anzahl der zutreffenden Knoten in der Datenbank

an. Die Knoten werden automatisch aneinander ausgerichtet, können aber auch manuell bewegt werden. Gerichtete Kanten zwischen Knoten werden mit der mittleren Maustaste durch Ziehen gezeichnet. Anschließend muss der Typ der Relation bestimmt werden. Abbildung 3.7 zeigt einen beispielhaften Anfragegraphen.

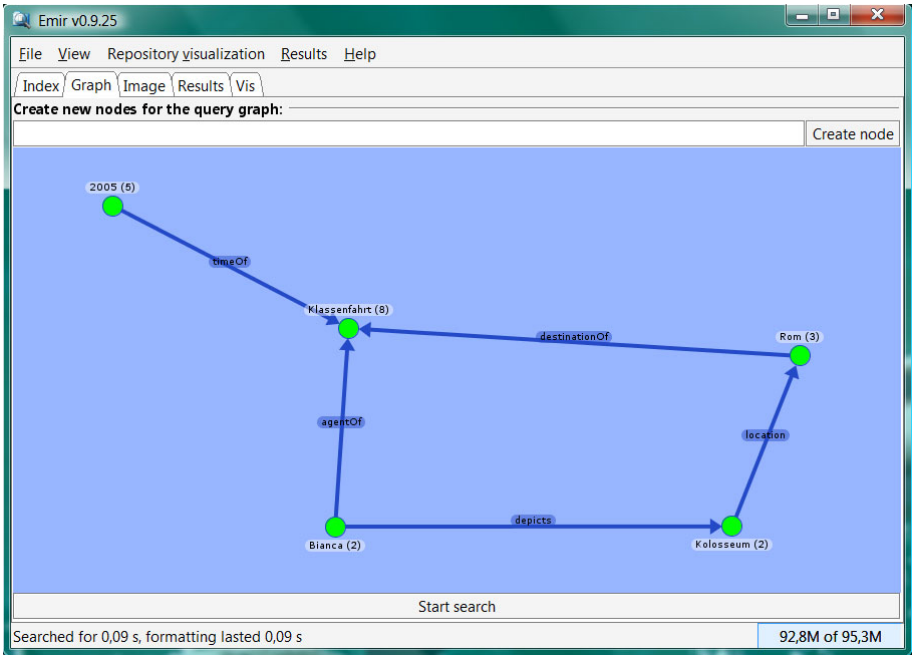


Abbildung 3.7: Emir - Suche anhand eines Graphen

Die letzte Suchvariante ist unter dem Tab Image vorzufinden. Abbildung 3.8 veranschaulicht dieses. Ein Anfragebild wird aus einem Verzeichnis ausgesucht und erscheint als Vorschau im linken unteren Bereich. Direkt darüber wird eine Voransicht des entsprechenden Color Layouts gezeigt. Der Nutzer kann aus einer Kombination der MPEG-7 Deskriptoren Color Layout, Scalable Color und Edge Histogram den Suchmodus bestimmen. Im Fall von Query by Example kann auch ohne bereits existierende MPEG-7 Dateien gesucht werden. Diese werden automatisch erzeugt, wenn in den Konfigurationen, unter File/Configuration aufrufbar, die Option *use Derby DB for content based image retrieval* ausgewählt ist. Der Search-Button befindet sich wie bei den Tabs Graph und Keywords in der Mitte des unteren Bereichs.

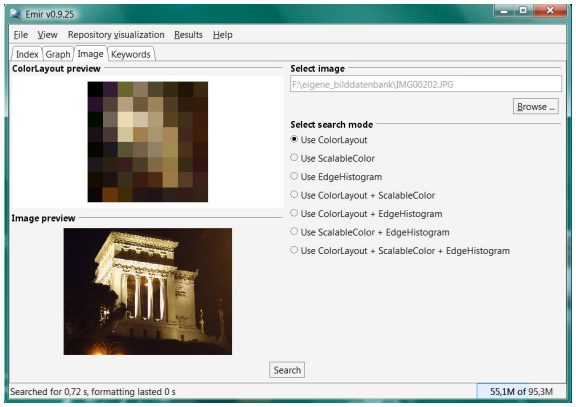


Abbildung 3.8: Emir - Query by Example

Die Ergebnisse werden als Thumbnails standardmäßig in einem neuen Tab namens Results in einer einspaltigen Liste präsentiert. Informationen wie z.B. Dateiname, Bildgröße, Bildbeschreibungen und Relevanz zur Anfrage werden rechts neben den Bildern angezeigt. Eine Vergrößerung der Bilder ist durch einen Klick auf das Foto möglich. Erwähnenswert ist neben der Listendarstellung die 2D-Visualisierung der Ergebnisse. Diese kann im Menü unter Results/Visualize results gestartet werden. Abbildung 3.9 zeigt die räumliche Anordnung der Ergebnisse, die durch den FastMap-Algorithmus auf Basis der drei bereits genannten visuellen Deskriptoren und der Ähnlichkeit von semantischen Beschreibungen berechnet wird [Lux07]. Beim FastMap-Algorithmus werden basierend auf einer Distanzfunktion beliebige Medienobjekte in einen k-dimensionalen Raum abgebildet, wo die euklidische Distanz eine Annäherung der ursprünglichen Distanzwerte ermöglicht [FL95]. Zoomen der Ansicht wird über das Scrollrad der Maus realisiert. Eine Umsortierung mittels Force Directed Placement (FDP) ist über <ALT> und rechter Maustaste möglich.

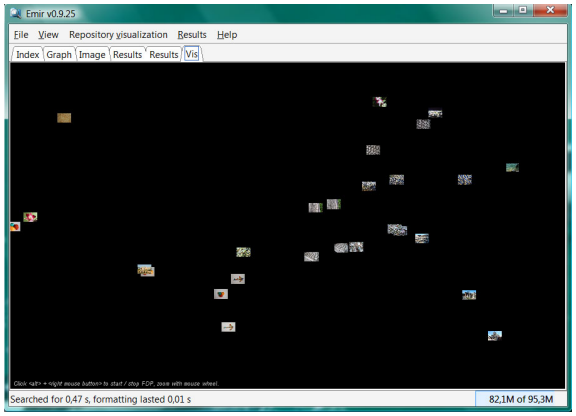


Abbildung 3.9: Emir - 2D-Visualisierung der Ergebnisse

Bewertung von Emir

Emir bietet einen gut strukturierten Aufbau. Die grundlegenden Suchfunktionen werden über die drei Standardtabs Index, Graph und Image abgedeckt. Die Optionen bei der Suche anhand eines Beispielbildes sind für den Laien nicht verständlich, eine Erklärung der drei Deskriptoren wäre sinnvoll. Im Gegensatz zu anderen CBIRS kann hier auch mit semantischen Graphen gesucht werden. Negativ anzumerken ist jedoch der Fakt, dass der Nutzer bei der Erstellung eines Graphen nicht weiß, wie er Kanten hinzufügen kann. Die Existenz der Hilfe über die rechte Maustaste ist für den Anfänger nicht der erste Gedanke. Ein erklärender Text am unteren Bildrand könnte hilfreich sein. Außerdem wurde die Bereitstellung der Hilfe über die rechte Maustaste nicht konsequent für das gesamte Programm durchgesetzt. Ist der Anwender erfahrener in der Benutzung, so enthält das Menü fortgeschrittene Suchoptionen und über Shortcuts ist eine schnellere Interaktion möglich. Emir ist sowohl für den Anfänger als auch für den fortgeschrittenen Anwender konzipiert worden. Ein weiterer positiver Aspekt ist die neben der Liste zusätzlich bereitgestellte 2D-Visualisierung der Ergebnisse. Das Zoomen ist intuitiv, jedoch kann die Ansicht nicht soweit vergrößert werden, dass das Abgebildete auf den Fotos gut zu erkennen ist. Verbesserungswürdig sei auch die angezeigte Größe eines Bildes bei einem Klick in der Listendarstellung. Bei jeder Suchanforderung wird ein neuer Result-Tab geöffnet. Das Schließen eines Tabs über einen Close-Button wäre schneller als die bisherige Lösung über den Menüpunkt View/Close active Tab. Um die Funktionalitäten von Emir besser kennenzulernen, existiert neben der Online-Dokumentation mit Screenshots unter [Lux07] auch eine Flash-Demonstration [Lux05a].

In den letzten Jahren entstanden neben Image-Retrieval-Systemen auch Systeme für die inhaltsbasierte Suche nach Videos. Da das Hauptaugenmerk dieser Arbeit auf der Entwicklung einer GUI für die Bildersuche liegt, wird zum Ende dieses Unterkapitels nur ein Einblick in ein Video-Retrieval-System gegeben. Ein Video ist ein dynamisches Medium. Aufgrund dessen muss bei der Suche neben anderen Kriterien auch der zeitliche Aspekt beachtet werden. Videos können Bilder, ein oder mehrere Audiospuren sowie textuelle Informationen wie Untertitel enthalten. Bei der Ähnlichkeitssuche werden deshalb auch Verfahren des Image- und Audio-Retrievals angewendet. Die optische Zeichenerkennung (Video Optical Character Recognition - OCR) zur Identifizierung des Textes, der im Video erscheint, die automatische Gesichts- oder Spracherkennung sind einige Beispiele hierfür [HJN03]. Im Folgenden wird das Video-Retrieval-System VideoQ vorgestellt.

3.5 VideoQ

VideoQ ist ein Web-basiertes System zur Videosuche, das innerhalb des ADVENT Projekts an der Columbia University 1997 zu entwickeln angefangen wurde [ADV97a]. Die Suche mit VideoQ kann unter [ADV97b] gestartet werden. Es werden drei Suchmodi unterstützt:

- Video Navigator
- Text Search
- Visual Search

Die Ergebnisse bei allen drei Suchmodi werden nicht angezeigt, da die angeforderte URL nicht auf dem Server gefunden werden kann. Das lässt sich höchstwahrscheinlich auf die nicht mehr aktualisierte Webseite zurückführen. Trotz fehlender Ergebnispräsentation wird die Benutzeroberfläche von VideoQ hier vorgestellt, weil dem Nutzer viele nennenswerte Möglichkeiten zur Spezifizierung der gesuchten Videos angeboten werden.

Der Video Navigator ermöglicht das Browsen durch die Kategorien Nature, Sports, Technology, Transportation und Travel. Diese besitzen wiederum zwei weitere Unterkategorieschichten. Jedes Video der Datenbank wurde mit Text annotiert, der den Inhalt beschreibt und die wesentlichen Objekte identifiziert. Unter dem Modus Text Search können Videos anhand dieser Beschreibungen gesucht werden. Abbildung 3.10 zeigt eine bespielhafte Eingabe. Begriffe werden mittels Boolescher Operatoren verknüpft. Unter dem Eingabefeld stehen einige Beispielanfragen, die dem Benutzer das Prinzip der Eingabe verständlicher machen.

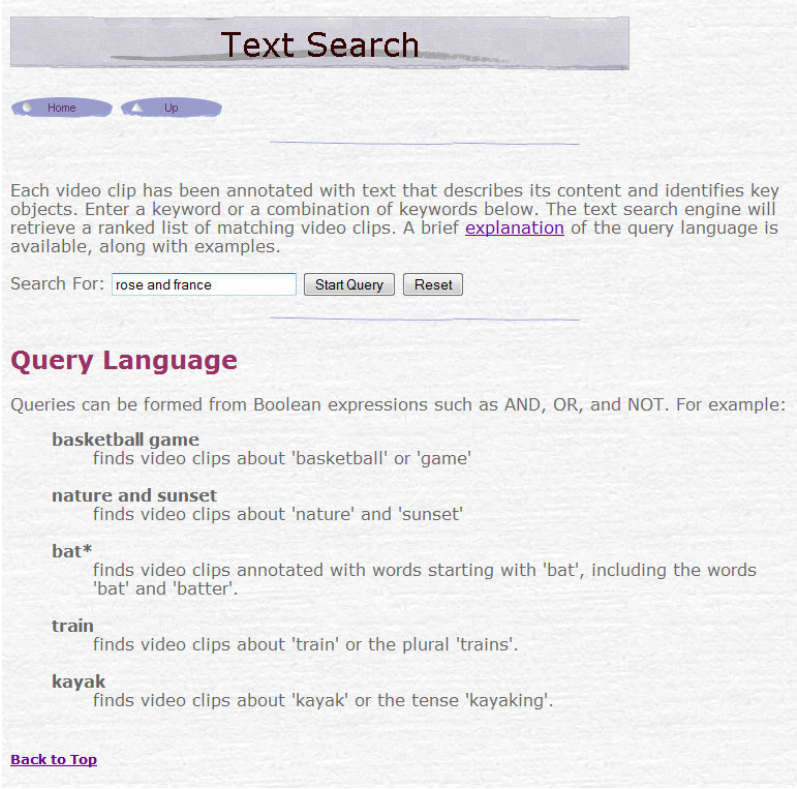


Abbildung 3.10: VideoQ - Suche im Modus Text Search

Unter Visual Search kann der Nutzer, wie es auch bei einigen Image-Retrieval-Systemen möglich ist, eine eigene Skizze als Anfrage verwenden. In einem Java Applet Window befindet sich die Arbeitsoberfläche zum Zeichnen. Videos können anhand von Formen, Farben, Texturen und Bewegungen gesucht werden [CCM+97]. Rechts neben der Zeichenfläche befindet sich eine Werkzeugleiste, die die wichtigen Funktionen für die Angabe der Features bereitstellt. Die Abbildung 3.11 zeigt die Benutzeroberfläche der visuellen Suche. Die Skizze im Zeichenbereich entspricht der Anfrage „Finde Videos, in denen eine Person einen Berg besteigt“. Die graue Ellipse soll vereinfacht den Wanderer darstellen. Im Folgenden werden die Funktionalitäten zur Erstellung dieser Skizze genauer beschrieben.

Als Form kann zwischen Rechteck (Abb. 3.11 Button 2), Ellipse (Abb. 3.11 Button 3) und Polygon (Abb. 3.11 Button 4) gewählt werden. Bei einem Polygon fügt man durch jeden Klick einen neuen Eckpunkt hinzu, mit einem Doppelklick wird das Polygon als fertiggestellt übernommen. Eine Markierung eines Objekts bzw. dessen Aufhebung wird mit einem Doppelklick auf dieses vorgenommen. Markierte Objekte können mit gedrückter linker Maustaste verschoben werden. Die vertikale und horizontale Größe einer Figur lässt sich durch Ziehen des rechten unteren Eckpunktes verändern. Der Mauszeiger wird

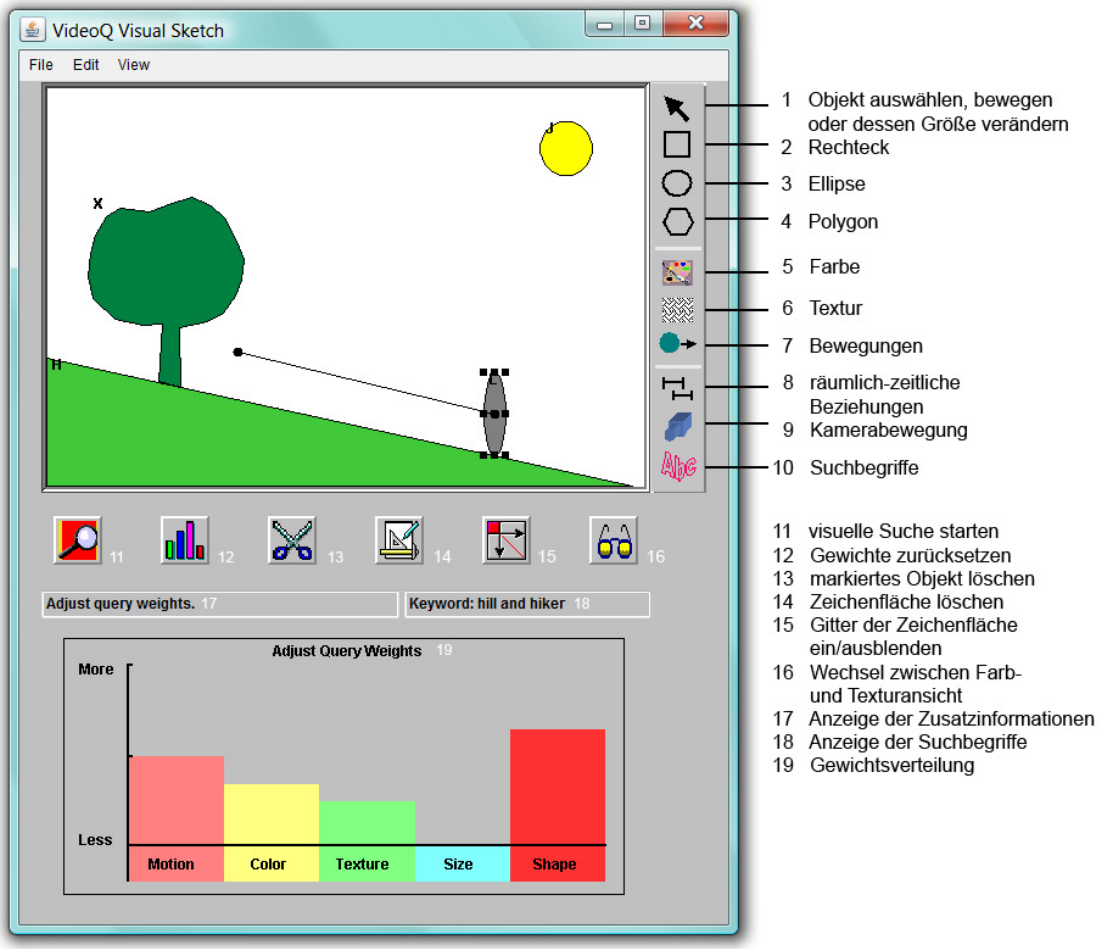


Abbildung 3.11: VideoQ - Suche im Modus Visual Search

während dieser Aktion als ein Pfeil mit zwei in entgegengesetzte Richtungen angeordneten Spitzen angezeigt. Unterhalb der Skizze befinden sich weitere Buttons, wie z.B. der Schere-Button (Abb. 3.11 Button 13), mit dem ein Element gelöscht werden kann. Die ganze Arbeitsfläche auf den Anfangszustand zurücksetzen, erfolgt durch Anklicken des Buttons 14. Soll eine Figur einer Farbe zugeordnet werden, klickt man auf den Mischpalette-Button (Abb. 3.11 Button 5). Es öffnet sich ein neues Fenster, welches die Auswahl einer Farbe aus dem HSV-Farbraum ermöglicht. Dieses ist unter (a) der Abbildung 3.12 zu sehen. Falls bei den 48 vordefinierten Farben nicht die passende enthalten ist, so kann im Custom Colors-Bereich aus dem Kreis die Sättigung (Saturation S), der Farbton (Hue H) und aus

dem darunter befindlichen Rechteck die Helligkeit (Value V) ausgewählt werden. Neben dem Zeichnen der Formen und Festlegen der Farben können Figuren auch einer Textur zugewiesen werden. Bei einem Klick auf den Button 6 in der Abbildung 3.11 öffnet sich eine Sammlung von Texturen, die von der MIT Media Lab Vision Texture Group bereitgestellt wurden. Der Berg, der von einer Blumenwiese bedeckt ist, erhält die Textur aus Abbildung 3.12 (c). Die Ansicht der Skizze wechselt automatisch von der Farbebene zur Texturebene. Die Metapher der Brille wird für den manuellen Wechsel zwischen beiden Sichten verwendet (Abb. 3.11 Button 16). Um die Schrittrichtung des Wanderers als Suchkriterium anzugeben, muss die graue Ellipse markiert sein und auf den Button 7 in der Werkzeugleiste geklickt werden. Es öffnet sich das unter Abbildung 3.12 (b) gezeigte Fenster. Jedes Objekt der Skizze wird mithilfe der Buchstabenlabels H, J, L und X eindeutig identifiziert. Für jede Figur kann angegeben werden, ob sie sich im Vordergrund, Hintergrund oder dazwischen befindet. Außerdem wird mittels eines Schiebereglers die gesamte Zeitdauer der Anzeige des Objekts in der Aufnahme (Shot) festgelegt. Nach dem Bestätigen durch Klick auf den Add Motion-Button gibt der Nutzer die Bewegungsbahn durch ein Polygon an. Der erste Punkt des Polygons muss innerhalb des markierten Objekts liegen, das sich bewegen soll. Der Wanderer geht den Berg hinauf. Die Bewegungsrichtung wird in diesem Fall als eine Linie von Objekt L ausgehend gezeichnet, Abbildung 3.11 verdeutlicht dieses. Neben den Bewegungen der Objekte kann sich auch die Kamera selbst bewegen. Die Abbildung 3.12 (d) zeigt das entsprechende Fenster, welches sich über den Button 9 öffnen lässt. Dort können die Schwenkrichtung der Kamera mittels Himmelsrichtungen und der Zoom festgelegt werden. VideoQ bietet auch eine Kombination der visuellen Suche und der Textsuche an. Im Fenster, welches dem Button 10 zuzuordnen ist, können Begriffe mit Booleschen Operatoren verbunden werden. Für das Beispiel mit dem Berg und Wanderer werden als Schlüsselwörter *hill and hiker* angegeben. Diese Begriffe erscheinen auch im Feld 18. Die Textsuche wirkt hier als Präfilter, der die für die visuelle Suche in Frage kommenden Videos einschränkt [ADV97d].

Nachdem die zu suchenden Formen und deren Größe, Farbe, Textur und Bewegung eingegrenzt wurden, können Gewichte für diese vergeben werden. Die Säulen im Bereich 19 der Abbildung 3.11 zeigen die Wichtigkeit eines Features innerhalb der Anfrage an. Um den Gewichtswert zu setzen, muss im Bereich des entsprechenden Merkmals in der Höhe des Wertes geklickt werden. Die Suche startet durch einen Klick auf den Lupe-Button (Abb. 3.11 Button 11).

Da die Ergebnisse unter [ADV97c] nicht präsentiert werden, wird auf die Aussagen unter [ADV97b] verwiesen. Es wird eine Liste von Video-Keyframes zurückgeliefert, welche am Anfang des jeweiligen Videoclips extrahiert wurden. Ein weißer Rahmen zeigt den

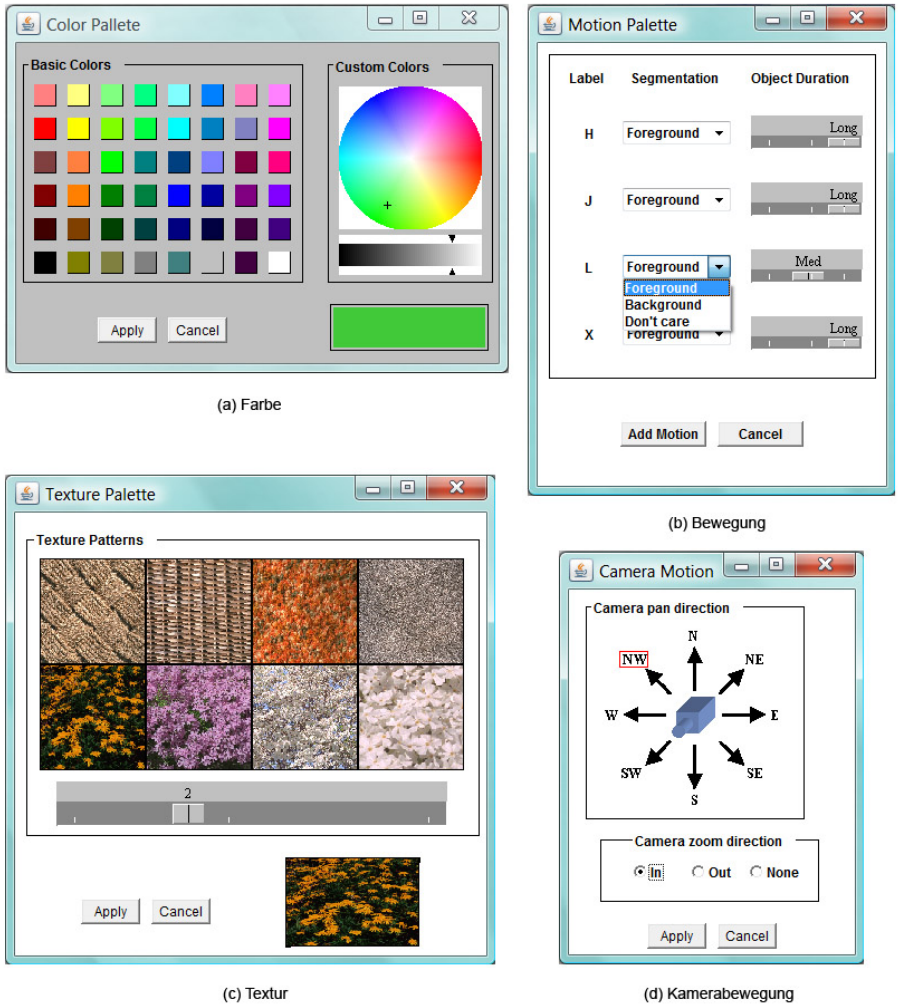


Abbildung 3.12: Festlegen der Feature-Werte im Modus Visual Search

Bereich an, der mit der Anfrage übereinstimmt. Ein gestrichelter Rahmen zeigt die Region an, welche nicht im gezeigten Keyframe, aber in anderen Bildern des Videoclips zu der Anfrage passt. Neben der ID und Dauer des Videos in Echtzeit wird ein Link zu dem Provider des Originalvideos angegeben. Das Video steht als MPEG in der Bitrate zur Verfügung, die aus einem der drei möglichen Bitraten bereits durch den Nutzer vor der Suche ausgewählt wurde.

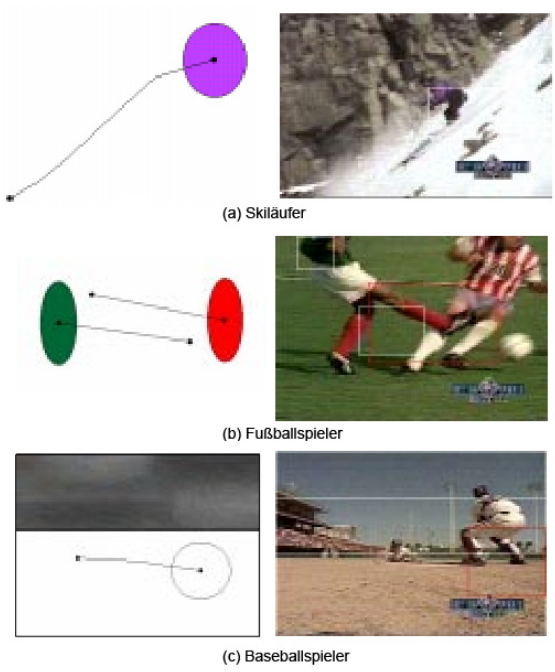


Abbildung 3.13: Beispielanfragen mit VideoQ im Modus Visual Search [CCM+97]

Abbildung 3.13 zeigt drei Beispielanfragen. Auf der linken Seite ist die visuelle Anfrage, die als bestes Ergebnis das Video mit dem Keyframe auf der rechten Seite liefert. Bei dem Baseballspieler in (c) ist hervorzuheben, dass die himmelsähnliche Textur der Anfrage mit der des Himmels beim Video übereinstimmt.

Bewertung von VideoQ

VideoQ bietet für die Videosuche viele Möglichkeiten um das gewünschte Resultat zu finden. Neben der Navigation über Kategorien beim Video Navigator oder der Suche über Annotationen beim Text Search ist die Kombination von visuellen und textuellen Bedingungen beim Visual Search möglich. Exakte und ähnlichkeitsbasierte, unscharfe Bedingungen können hier in einer Anfrage durch den Benutzer angegeben werden. Intuitiv ist auch das Setzen der Gewichte einzelner Anfrageterme, die die Subjektivität in die Suchanfrage einfließen lässt. Die Ergebnisse können nicht durch den Nutzer bewertet werden, sodass die Gewichte der einzelnen Anfrageterme an die Ähnlichkeitswahrnehmung des Anwenders angepasst werden könnten. Beim Visual Search wird anhand selbst entworfener Skizzen gesucht. Die Funktionen werden übersichtlich in einer Werkzeugleiste vereint. Viele Symbole sind selbsterklärend wie z.B. die Farbpalette (Abb. 3.11 Button 5), die Formenauswahl (Abb. 3.11 Buttons 2, 3 und 4) und die Stichworteingabe (Abb. 3.11 Button 10). Verbesserungswürdig ist dennoch das Symbol für die Kameraeinstellungen und das Festlegen

der Objektbewegungen. Die Kamera sollte auf Anhieb als diese erkennbar sein. Desweiteren befinden sich unterhalb des Zeichenbereichs Buttons. Die Metapher der Lupe (Abb. 3.11 Button 11) als Initiator der Suche oder die Brille (Abb. 3.11 Button 16) zum Wechseln zwischen verschiedenen Ansichten erleichtert die Benutzung der Software. Die Lupe kann aber auch in anderen Kontexten als Vergrößerungswerkzeug verstanden werden. Das Symbol der Schere (Abb. 3.11 Button 13) kann zum einen die Trennung eines Objekts in mehrere gleichwertige Teile bedeuten und zum anderen das Entfernen von etwas Unerwünschtem aus einem Objekt. In diesem Programm entspricht die Schere dem zweiten Interpretationsansatz. Das Symbol zum Löschen aller Objekte (Abb. 3.11 Button 14) ist nicht so aussagekräftig wie die Schere. Es ist zwar anhand der Zeichenutensilien wie Dreieck, Stift und Papier verständlich, dass das Symbol die Zeichenfläche darstellen soll, jedoch fehlt noch der Aspekt des Löschens bzw. Entfernens. Das ähnliche Prinzip besteht bei dem Symbol für das Zurücksetzen der Gewichte (Abb. 3.11 Button 12). Die Gewichte werden als Säulen dargestellt in Analogie zu dem Feld unterhalb der eigenen Skizze, das Löschen der Gewichte wird durch das Symbol jedoch nicht deutlich. Die Funktionalität einiger Buttons ist nicht auf den ersten Blick erkennbar. Im Gegenzug sei aber auch positiv zu betonen, dass der Nutzer Informationen im Feld 17 erhält, auf welchem sich die Maus momentan befindet. VideoQ bietet kein freies Zeichnen an, jedoch ist der Nutzer nicht sehr eingeschränkt, da er Polygone verwenden kann. Die Maus verändert sich dann zu einer Hand, mit der aber nicht das Zeichnen, sondern eher das Fassen und Verschieben von Objekten in Verbindung gebracht wird. Ein Polygonzug wird mit einem Doppelklick abgeschlossen. Das ist bei der ersten Verwendung ohne Anleitung unklar. Weiterführende Informationen nach Auswahl eines Werkzeugs wären deshalb hilfreich. Bei der Erstellung eines Bewegungspfades eines Objekts hat der Nutzer das gleiche Problem. Nach dem Bestätigen durch Klick auf Add Motion im Fenster Motion Palette, weiß der Anwender nicht, wie weiter vorzugehen ist. Desweiteren ist auch zu kritisieren, dass die Größe einer Figur nur über die rechte untere Ecke einstellbar ist. Aus vielen Bildbearbeitungsprogrammen kennt der Nutzer die Möglichkeit die Größe über Verschieben einer beliebigen Ecke zu verändern. Außerdem verwandelt sich der Mauszeiger zu einem Pfeil erst nach erfolgtem Verschieben der rechten Ecke und nicht bereits beim Bewegen der Maus über die Ecke. Dadurch ist dem Nutzer anfangs nicht bewusst, dass er die Größe überhaupt verändern kann. Das Medium Video hat neben räumlichen Abhängigkeiten auch zeitliche. Um diese bei der Suche zu beachten, kann der Nutzer sowohl die Bewegung des Objekts als auch die der Kamera angeben. Die räumlich-zeitlichen Beziehungen der Objekte können nicht festgelegt werden, obwohl bereits der entsprechende Button (Abb. 3.11 Button 8) existiert. Operationen auf Dateien und Bearbeitungsschritten sowie der Wechsel der Ansichten

können über das Menü ausgewählt werden, jedoch ist nur der Menüpunkt View vollständig implementiert. Allgemein kann gesagt werden, dass dem Nutzer viele Möglichkeiten gegeben werden seine Anfrage genau zu formulieren. Die Auswahlmöglichkeiten innerhalb der einzelnen Fenster aus Abbildung 3.12 sind auch gut verständlich. Eine Hilfe wird bei jeder der drei Suchoptionen über einen Help-Button angeboten. Bilder und Demonstrationen fehlen jedoch hier. Die Ergebnispräsentation kann hier nicht beurteilt werden, da die Ergebnisliste aufgrund veralteter Referenzen nicht generiert werden konnte.

Übersicht der vorgestellten Systeme

In der Abbildung 3.14 ist ein Vergleich der in diesem Unterkapitel vorgestellten Content-Based-Image and Video-Retrieval-Systeme (CBIVRS) aufgeführt. Welche Features bei der Ähnlichkeitsberechnung in FIRE und GIFT verwendet werden, ist aus [Kos09] entnommen. Es werden die Suche, die Ergebnispräsentation, das Relevance-Feedback und die Bereitstellung von Hilfe der einzelnen Programme gegenübergestellt.

	State Hermitage Museum mit QBIC-Suchfunktionen	Online-Demo FIRE	Online-Demo VIPER/GIFT	Emir	VideoQ
Suche					
Browsing	✓	✓ (random)	✓ (random)	×	×
Navigation über Kategorien	✓	×	×	×	✓
benutzerspezifizierte Anfrage					
exakte Bedingungen	✓	×	×	✓	✓
ähnlichkeitsbasierte, unscharfe Bedingungen	Farbe, Layout	Farbe, Textur	Farbe, Textur	ColorLayout, ScalableColor, Edge-Histogram, Textur	Farbe, Textur, Form, Bewegung
Query by Sketch	✓	×	×	×	✓
Query by Example	×	✓	✓	✓	×
weitere Möglichkeiten	×	×	×	semantischer Graph	×
Kombination exakter und unscharfer Bedingungen	×	×	×	✓	✓
explizite Gewichtung der Bedingungen durch Nutzer	×	×	×	×	✓
Ergebnispräsentation					
Präsentationsmöglichkeiten	Liste (2 Spalten)	Liste (5 Spalten)	Liste (5 Spalten)	Liste (1 Spalte) 2D-Visualisierung	Liste (1 Spalte)
Grad der Anfrageerfüllung	×	×	✓	✓	✓
Detailansicht der Ergebnisse	Vergrößerung Gesamtbild, fester Zoom für Bildbereich	nur Vergrößerung Gesambild	nur Vergrößerung Gesamtbild	nur Vergrößerung Gesamtbild	Auswahl aus drei möglichen Bitraten
Metadaten der Ergebnisse	✓	×	×	✓	✓
Relevance Feedback	×	✓	✓	×	×
Art der Bewertung	×	Radiobuttons (+, • , -)	Combobox (neutral, rel, non-rel)	×	×
Anzahl Bewertungen	×	beliebig	beliebig	×	×
Bereitstellung von Hilfe	gut (textuelle Hilfe, Animationen)	mangelhaft	nicht vorhanden	gut (Flash-Demo, Online-Dokumentation)	befriedigend (Info-Feld, Online-Dokumentation)

× nicht möglich bzw. nicht angegeben
✓ möglich bzw. angegeben

Abbildung 3.14: Übersicht der vorgestellten CBIVR

4 Analyse der Anforderungen an eine GUI für eine visuelle Mediensuche

Aufbauend auf den Bewertungen existierender grafischer Benutzeroberflächen von inhaltsbasierten Image- und Video-Retrieval-Systemen im Kapitel 3 werden in diesem Kapitel die Anforderungen an eine zu entwerfende GUI für das visuelle Multimedia-Retrieval konkretisiert. Der Schwerpunkt soll dabei auf die Suche von Bildern gesetzt werden. Auf Grundlage der zu erfüllenden Kriterien werden verschiedene Prototypen im anschließenden Kapitel vorgestellt.

Neben den allgemeinen Anforderungen an die grafische Benutzeroberfläche muss beim Entwerfen der GUI die Anfragesprache CQQL einbezogen werden. Dessen grundlegende Regeln zur Formulierung einer Anfrage, zum Erstellen der Evaluierungsfunktion und zur Abbildung zwischen Präferenzen und Gewichten wurden im Abschnitt 2.2.2 genauer beschrieben. Im Folgenden werden die darauf beruhenden Kriterien für die Suche, die Ergebnispräsentation und die Einbeziehung des Nutzers in den iterativen Suchprozess zusammengestellt. Desweiteren soll die Reaktion des Systems auf Fehleingaben des Nutzers genauer untersucht werden.

4.1 Allgemeine Betrachtung

Ben Shneiderman, der als einer der ersten Informatiker die Wichtigkeit des grafischen Interface-Designs erkannt hat, beschreibt in [Shn92a], was ein erfolgreiches Computersystem ausmacht:

> *„Well designed, effective computer systems generate positive feelings of success, competence, mastery and clarity in the user community. When an interactive interface is well-designed, the interface almost disappears, enabling users to concentrate on their work, exploration, or pleasure.“*

4.1.1 Zielgruppe der GUI

Die zu entwerfende GUI soll für Laien konzipiert werden, Menschen die nicht in den Entwicklungen von Konzepten für das Multimedia-Retrieval involviert sind. Privatpersonen, Journalisten, Fernseh- und Radiosender können als eine kleine Auswahl aus der großen Zielgruppe genannt werden. Der Nutzer möchte sich nicht wissenschaftlich mit der Thematik der ähnlichkeitsbasierten Suche von Multimedia-Objekten auseinandersetzen. Daraus resultiert die Notwendigkeit die Benutzeroberfläche nicht implementierungsnah zu gestalten, also keine Oberfläche anzubieten, die auf dem Verständnis basiert, wie der Suchprozess intern abläuft.

In unterschiedlichen Kulturen können die Leserichtung, die Bedeutung von Symbolen und Gesten und die Formate beispielsweise der Zeit und des Datums variieren. Die GUI wird deshalb hauptsächlich für deutsche Anwender mit dem Bezug zur länderübergeifenden europäischen Kultur entworfen. Um die Weiterentwicklung der GUI in Hinblick auf andere Medientypen wie z.B. Audio zu vereinfachen und eine Anwendung der Software über den deutschen Sprachraum hinaus zu ermöglichen, erfolgt die Interaktion mit dem Nutzer auf Englisch.

4.1.2 Arten von Benutzerschnittstellen

Alan Cooper unterteilt die Benutzerschnittstellen in drei Klassen. Die erste ist die implementierungszentrierte Schnittstelle, die zweite die metaphorische und die dritte die idiomatische [CRC07]. Die erste sei für den vorliegenden Fall aus bereits genannten Gründen abzulehnen.

Die zweite, die metaphorische Schnittstelle, entfernt sich von der Voraussetzung, das System verstehen zu müssen, um die Schnittstelle erfolgreich verwenden zu können. Es wird sich Metaphern bedient, die nicht die wörtliche, sondern die übertragende Bedeutung repräsentieren. Diese sind intuitiv verständlich, da der Nutzer diese mental mit bereits gelernten Sachverhalten verbindet und aufgrund der Ähnlichkeiten auf die Funktion der Metapher schlussfolgert. Die grafische Nutzeroberfläche sollte visuelle Metaphern integrieren, sodass dem Nutzer der Einstieg in die Bedienung erleichtert wird. Diese können z.B. Buttons in der Werkzeugleiste sein, wie die Schere zum Ausschneiden bei VideoQ. Ein weiteres Beispiel sind die Werkzeuge Pinsel und Stift. In der Realität ermöglichen sie das Zeichnen mit Farben auf Leinwand oder Papier. Diese Funktionalität wird auf die Maus als Zeichenmittel auf der digitalen Arbeitsfläche portiert. Metaphern haben aber auch ihre Grenzen. Wenn der Nutzer aufgrund seiner kulturellen Herkunft eine andere Assoziation hat als der Designer, dann werden die Bilder falsch verstanden. Desweiteren muss in Hin-

blick auf die Erweiterung der GUI zur Suche anderer Medientypen wie Audio beachtet werden, dass die Metaphern auch dann noch gut arbeiten sollen.

Aus diesen Gründen soll die GUI eher im Bereich der idiomatischen Schnittstellen anzusiedeln sein. Diese basieren auf dem Lernen und Anwenden von Idiomen. Mit Idiomen sind die Primitiven gemeint wie z.B. Mausklick, Ziehen mit der Maus, Tastendruck oder deren Kombination wie Doppelklick und Drag & Drop, denen in einer Applikation eine bestimmte Funktionalität zugeordnet wird. Eine bestehende Bedeutung im Alltag existiert nicht. Wie Redewendungen können Idiome nur durch vorheriges Lernen verstanden werden, wobei gute einmal gelernt und nie wieder vergessen werden. Das menschliche Gehirn kann eine Menge von Idiomen leicht aufnehmen und auf diese schnell zugreifen. Darunter zählen beispielweise das Minimieren und Maximieren von Fenstern und das Auswählen aus Drop-Down-Menüs.

4.1.3 Die 8 goldenen Regeln des Dialog-Designs

Shneiderman stellte auf Grundlage von heuristischen Verfahren die folgenden 8 Regeln des Interface-Designs auf [Shn92a]:

1. **Konsistenz:** Die Benutzeroberfläche soll konsistent sein. In ähnlichen Situationen sollen konsistente Aktionsfolgen auftreten. Ein Begriff muss überall die gleiche Aktion hervorrufen. Desweiteren soll für eine Aktion überall das gleiche Wort verwendet werden, da verschiedene Bezeichnungen für das identische schwer einprägsam sind.
2. **Shortcuts:** Um Fortgeschrittenen ein schnelleres Arbeiten mit dem System zu ermöglichen, sollen Shortcuts, Abkürzungen, versteckte Befehle und Makros zur Verfügung stehen.
3. **Informatives Feedback:** Für jede Operation soll das System dem Nutzer eine Rückmeldung geben. Informationen für kleine oder häufige Aktionen können bescheiden ausfallen im Gegensatz zum Feedback bei großen oder seltenen Operationen.
4. **Abgeschlossenheit von Aktionssequenzen:** Aktionsfolgen sollen in die drei Gruppen Beginn, Mitte und Ende eingeteilt werden, die für den Nutzer klar erkennbar sein sollen. Aus diesem Grund erhält der Anwender nach dem Ende einer Interaktion eine Systemmitteilung. Dieses signalisiert die Möglichkeit der Abarbeitung einer neuen Gruppe.
5. **Einfache Fehlerbehandlung:** Das System soll so entworfen werden, dass der Nutzer keine schwerwiegenden Fehler machen kann. Bei dem Auftreten eines Fehlers, soll dieser erkannt werden und einfache Methoden zur Behebung bereitstehen. Nur der fehlerhafte Teil soll korrigiert werden, nicht die gesamte Eingabe.

6. **Reversible Eingaben:** Aktionen sollen umkehrbar sein. Mit dem Gedanken beim Auftreten von Problemen Operationen rückgängig machen zu können, ist der Anwender eher ermutigt unbekannte Optionen auszuprobieren. Reversibilität soll auch für eine Menge von Aktionen möglich sein.
7. **Benutzerkontrolle:** Um dem Nutzer das Gefühl der Kontrolle über das Programm zu geben, soll dieser Aktionen auslösen anstatt auf sie zu reagieren. Die Ursache einer Systemreaktion soll für ihn nachvollziehbar sein, da er sonst Angst verspürt und unzufrieden ist.
8. **Entlastung des Kurzzeitgedächtnisses:** Der Psychologe George Miller stellte 1956 die Theorie auf, dass der Mensch nur 7 ± 2 Informationsbrocken, wie z.B. Nummern, Buchstaben oder Worte, im Kurzzeitgedächtnis halten kann [Mil56]. Dementsprechend müssen Anzeigen einfach gehalten, mehrere Anzeigeseiten zusammengelegt und die Frequenz der Fensterbewegungen minimiert werden. Dem Nutzer soll ausreichend Trainingszeit gegeben werden, um sich Aktionsfolgen und Mnemonics[1] einzuprägen. Außerdem sollen Informationen online als Hilfe zur Verfügung stehen.

4.1.4 Mentales und konzeptionelles Modell

Ziel beim Entwurf der GUI ist es, dass sich das mentale Modell dem konzeptionellem Modell annähert bzw. diese im Optimalfall übereinstimmen. Das mentale Modell ist das Verständnis des Nutzers über die Funktionsweise des Systems. Dieses entwickelt sich aufgrund von Erfahrung und Training, um Systemvorgänge zu verstehen und um Entscheidungen bei der Interaktion mit dem Programm zu treffen. Der Nutzer versucht mit seinem Modell Aktionen vorherzusehen, falls er diese vergessen hat oder diese noch nie auftraten. Das konzeptionelle Modell hingegen basiert auf den Implementierungen des Systems, wie das System also tatsächlich funktioniert. Wenn das entwickelte mentale Modell mit dem konzeptionellen Modell übereinstimmt, dann ist die Benutzung des Systems intuitiv. Im anderen Fall kann der Nutzer unzufrieden werden, wenn nicht das Vermutete eintritt. Ein Beispiel, bei dem sich beide Modelle nicht gleichen, ist das mehrmalige Betätigen des Schalters um einen Fahrstuhl anzufordern. Der Nutzer denkt fälschlicherweise, dass dadurch der Lift schneller eintrifft.

4.1.5 Multi-Layer-Interface-Design

Bei dem Entwurf der GUI soll sowohl an die Einsteiger als auch an die Fortgeschrittenen sowie Experten gedacht werden. Werden viele Optionen und Funktionen angeboten, so ist

[1] Gedächtnisstützen

der Anfänger meist überfordert. Um die Komplexität der Funktionen zu handhaben, können die Funktionen in Module aufgeteilt werden. Microsoft Word 2007 hat beispielsweise die verschiedenen Einheiten in Werkzeugleisten unterteilt. Unter Start und Einfügen sind die zwei wichtigsten Toolbars zu finden. Neben diesen existieren Formel-, Bild-, Zeichen-, Diagrammtools und viele weitere. Der Anfänger weiß jedoch damit immer noch nicht, wie er in das Programm einsteigen soll.

Ein Konzept, um dieses Problem zu lösen, wurde in [Shn03] unter dem Begriff Multi-Layer-Interface-Design vorgestellt. Durch das Bereitstellen einer vielschichtigen Design-Architektur kann eine universelle Benutzerfreundlichkeit erzielt werden. Der Nutzer selbst hat die Kontrolle, welche Funktionen angezeigt werden. Mittels Stufen wird der Nutzer an die einzelnen Funktionen in einer bestimmten Reihenfolge herangeführt. Auf der untersten Stufe befindet sich der Anfänger. Dort werden dem Nutzer Hilfestellungen geboten, welche die grundsätzlichen Ziele und Hauptfunktionen vorstellen. Das Programm ist auf dieser Ebene in seiner Funktionalität sehr eingegrenzt. Fehler des Nutzers werden dadurch verhindert. Das so genannte Stützrad-Prinzip führt zu schnellerem Lernen und steigert die Zufriedenheit des Nutzers [CC84]. Denkbar ist die Einschränkung der Suchanfrage, sodass der Anfänger nicht mit den vielen Möglichkeiten der Anfragestellung konfrontiert wird. Die Suche in Multimedia-Datenbeständen ist für ihn ein neues Gebiet.

Wenn der Anwender sicherer in der Bedienung des Programms ist, steigt er eine Stufe höher. Hier werden mehr Funktionen angeboten und die aus Stufe 1 bekannten Hilfsanweisungen ausgeblendet, die der Fortgeschritte nicht mehr benötigt. Der durchschnittliche Nutzer kennt die Arbeitsweise des Systems, jedoch hat er z.B. noch nicht alle Bedeutungen der Symbole im Kopf. Tooltips[2] sind hierfür gut geeignet. Sie nennen die Bedeutung der Funktion in Kürze und beschreiben nicht die Anwendung und das Ziel dieser, wie das bei den Hilfestellungen in der ersten Ebene der Fall ist. Pull-Down-Menüs und mehr Buttons wären eine mögliche Erweiterung in der zweiten Ebene. Die Anzeige der Suchergebnisse in einer anderen Darstellungsform wäre hier vorstellbar. Auf höheren Stufen kann der versierte Anwender z.B. Shortcuts und Makros zur schnelleren Arbeit anwenden.

Bei dem Entwurf der GUI müssen die Fragen gestellt werden, wie viele Schichten die GUI aufweisen soll und welche Funktion wo eingeordnet wird. Desweiteren muss auch geklärt werden, wie der Nutzer erkennen kann, auf welcher Ebene er sich befindet und wie er auf die nächste steigen kann. Das Umsetzen eines Multi-Layer-Interface-Designs kann somit die Lücke zwischen Einsteigern und Experten schließen.

[2] kleines Fenster mit Kurzinformationen zu einem Element, das bei Positionierung des Zeigers auf diesem erscheint

4.1.6 Generelle Anforderungen an die GUI

Dem Nutzer soll eine gut strukturierte und übersichtliche GUI vorliegen, bei der die Suche und die Ergebnispräsentation voneinander getrennt werden. Funktionen sollen in Gruppen eingeteilt werden, welches u.a durch eine Menüstruktur und das Bereitstellen von Toolbars realisiert werden kann. In der 8. goldenen Regel des Dialog-Designs von Shneiderman wird gefordert, dass die Last für das Kurzzeitgedächtnis reduziert werden soll. Diese Theorie bedeutet jedoch nicht für den Entwurf, dass nur 7 Optionen in einem Menü oder nur 7 Einträge in einer Liste oder Pull-Down-Menü platziert werden dürfen [SRP07]. Das liegt daran, dass diese Elemente nicht aus dem Kurzzeitgedächtnis abgefragt werden müssen, weil sie nicht vom Bildschirm verschwinden. Ist etwas einzigartig oder auf eine bestimmte Weise anders, so steigt die Wahrscheinlichkeit, dass sich der Nutzer daran erinnert. Dieses kann ausgenutzt werden, indem wichtige Elemente hervorgehoben werden, um diese im Gedächtnis zu behalten. Hoch gesättigte oder warme Farben, intensive Kontraste und große Symbole lenken die Aufmerksamkeit des Nutzers. Diese Mittel sollen jedoch sparsam eingesetzt werden, sonst wirken sie nicht mehr. Große Flächen wie der Hintergrund können in neutralen Farben gehalten werden und nur wichtige Objekte werden mittels auffälliger Farbe ausgezeichnet. Neben der Konsistenz von Aktionen und deren Begriffe, die Shneiderman in seiner ersten Regel fordert, sollen auch die Farben, das Layout und die Schriftart einheitlich verwendet werden. Die Position von Objekten soll gut überlegt sein. Eine gut entworfene GUI führt den Blick des Nutzers über alle relevanten Objekte in der vom Designer erzielten Reihenfolge.

Die Benutzerschnittstelle soll, wie bereits erwähnt, hauptsächlich idiomatisch ausgelegt sein. Es sollen Symbole auftreten bzw. Aktionsfolgen bereitstehen, die, nachdem diese vom Nutzer gelernt wurden, die Interaktion mit dem System effektiver gestalten. Die Symbole sollen dabei stets einfach gehalten werden, damit diese für den Nutzer schneller erkennbar sind. Im Allgemeinen werden Symbole schneller und präziser erfasst als Text. Existiert bereits ein Symbol oder ein Wort, so soll es in der gebräuchlichen Variante verwendet werden, um den Nutzer nicht zum Umdenken zu zwingen. Bereits Gelerntes ist schwer umzulernen. Idiome müssen entworfen werden, die u.a. die Eingabe von Suchanfragen ermöglichen, die Suche starten, dem Nutzer die Ergebnisse nach seinen Bedürfnissen detaillierter anzeigen und Bewertungen dieser vornehmen. Um das Verständnis zu erhöhen, sollen für Funktionen sowohl das Objekt als auch die Aktion deutlich werden. VideoQ bietet hier ein Negativbeispiel. Dort stellt ein Button, auf dem nur Zeichenutensilien erkennbar sind, das Zurücksetzen der gesamten Zeichenfläche dar.

Neben den Deutschen sind auch allgemein die Europäer als Zielgruppe anzusehen. Um

eine einfache Bedienbarkeit auch für nicht englische Muttersprachler zu erzielen, sollen einfache englische Begriffe sowie eine einfache Satzstruktur verwendet werden. Abkürzung von Wörtern sollen vermieden werden. Die Leserichtung für die GUI sei von links nach rechts bzw. von oben nach unten.

Zusammenfassend lässt sich sagen, dass die GUI übersichtlich gestaltet und auf den einzelnen Stufen der mehrschichtigen Design-Architektur einfach bedienbar sein soll. Das Aufstellen der Suchanfrage und das Darstellen der Ergebnisse sollen erkennbar voneinander getrennt werden. Dem Nutzer muss der Ablauf der Suche mit dem Prinzip des Relevance-Feedback verständlich werden, sodass das mentale Modell dem konzeptionellen Modell gleicht.

4.2 Suchformulierung

Für die Suche von Bildern und Videos sollen dem Nutzer drei Arten der Suche bereitgestellt werden:

- Browsing
- Navigation
- anfragebasierte Suche

Diese werden im Folgenden näher vorgestellt. Der Schwerpunkt der Betrachtung liegt dabei auf der anfragebasierten Suche. Wichtig ist es, dass der Nutzer angeben kann, nach welchem Medientyp bzw. -typen er sucht. Entsprechend der Auswahl soll sich die Anzeige für die Suche anpassen.

4.2.1 Browsing und Navigation

Beim Browsen und Navigieren sucht der Nutzer manuell in der Datenbank. Während das Browsing eine eher ziellose Suche ist, ist die Navigation eine zielorientierte [Her06]. *To browse* wird übersetzt als blättern [Mes94]. Die Datenbank wird linear durchlaufen. Der Nutzer verfolgt gefundene Pfade aus der gegenwärtigen Suchposition heraus. Dieses ist jedoch nur dann sinnvoll, wenn der Nutzer keine konkrete Vorstellung von dem zu suchenden Medienobjekt hat und die Datenbasis relativ klein ist. Die GUI muss hierfür eine andere Variante als das zeitaufwendige Blättern von einem Medienobjekt zu einem anderen bereitstellen.

Beim Navigieren sucht der Anwender nichtlinear innerhalb der Sammlung sein Zielobjekt. Den Einstieg gewinnt der Nutzer meist mithilfe einer Übersichtskarte. In der Datenbank sucht er anhand von Kategorien und deren Unterkategorien. Eine Klassifizierung

kann nach abgebildeten Themen aufgestellt werden oder auch anhand von Metadaten, die nicht direkt etwas über den Inhalt aussagen, wie z.B. der Aufnahmeort. Neben einer linearen Auflistung der Gruppen können diese auch visualisiert werden. Auf das Thema Visualisierung wird im Abschnitt 4.3 näher eingegangen.

4.2.2 Anfragebasierte Suche

4.2.2.1 Klassische Datenbankbedingung

Für die Suche anhand einer Anfrage muss dem Nutzer eine Eingabe von klassischen Datenbankbedingungen ermöglicht werden. Um den Anwender dabei zu unterstützen, können die für eine exakte Bedingung in Frage kommenden inhaltsunabhängigen Metadaten, wie beispielsweise die Auflösung eines Bildes bzw. die Bitrate eines Videos, der Medientyp, das Speicherformat, das Aufnahmegerät oder der Dateiname, als Auswahl angegeben werden. Zu jedem Multimedia-Objekt werden die Metadaten und extrahierten Feature-Werte MPEG-7 konform in einer XML-Datei abgelegt. XML steht für Extensible Markup Language, es ist eine Metasprache zur Definition beliebiger Auszeichnungssprachen. Um eine fehlerhafte Eingabe des Nutzers zu vermeiden, kann bei jedem Attribut der Wertebereich aufgeführt werden. Das muss jedoch abgewägt werden, wenn der Wertebereich wie bei dem Dateinameattribut für eine Anzeige zu groß wird. Außerdem sollen neben $=$ auch die Vergleichsoperatoren $\neq, <, >, \leq$ und $\geq$ Einsatz finden, da beispielsweise bei der Abspielzeit eines Videos der Nutzer meist nur eine ungefähre Laufzeit festsetzen kann.

4.2.2.2 Ähnlichkeitsbasierte Bedingungen

Neben den scharfen Bedingungen sollen bei einer Anfrage im Multimedia-Retrieval-System ähnlichkeitsbasierte Bedingungen eingebbar sein. Solche Bedingungen beziehen sich auf die Low- und High-Level-Features.

High-Level-Features Die High-Level-Features, also die inhaltsbeschreibenden Metadaten, wie u.a. die abgebildeten Personen und Gegenstände, die dargestellten Aktivitäten und der Aufnahmeort, werden heutzutage überwiegend durch Menschen vergeben. Der derzeitige Forschungsstand schafft noch nicht die Voraussetzung für eine automatische Extraktion. Die Suche in den manuell erzeugten Annotationen ist Thematik des Information-Retrievals. Um einen solchen Anfrageterm auswerten zu können, wird die Vagheit der Sprache beachtet, indem u.a. für die Suche irrelevante Wörter entfernt, Zeichenketten auf ihren Wortstamm zurückgeführt und Wörter durch ihre Synonyme ersetzt werden. Für eine tiefgehende Beschäftigung mit dieser Materie sei auf [Hen08] verwiesen.

Der Nutzer soll das Abgebildete anhand von freien Texteingaben in natürlicher Sprache suchen können, so genanntes Query by Keyword. Bei der Eingabe mehrerer Wörter soll eine Verknüpfung dieser, z.B. durch die Angabe Boolescher Operatoren, möglich sein.

Low-Level-Features Unter den Low-Level-Features, im späteren Auftreten in dieser Arbeit nur noch als Features bezeichnet, sind die inhaltsbezogenen Metadaten [Sch06] zu verstehen, die automatisch aus dem Medienobjekt extrahiert werden können. Bei Bildern sind das u.a. die Farbe, Form und Textur. Der Laie kann zu einem Feature keine konkreten Angaben machen, da er die interne Verarbeitung von Bild- bzw. Videodaten nicht kennt. Er weiß nicht, welche Features vorhanden sind bzw. welche Werte sie annehmen können. Für ein System wird beispielsweise der zu verwendende Farbraum festgelegt, sodass z.B. die Farbwerte aus dem RGB-Farbraum oder aus dem der menschlichen Wahrnehmung besser angepassten HSV-Farbraum stammen können. Deshalb ist es nötig für den Laien Varianten bereitzustellen, die ihm eine einfache Angabe der Features ermöglichen und dessen interne Darstellung verbergen. Der Nutzer soll sich nicht in die Thematik einarbeiten müssen, um nach Bildern oder Videos suchen zu können. Aus diesem Grund soll die Benutzeroberfläche Query by Sketch und Query by Example zur Anfrageformulierung zur Verfügung stellen.

Query by Sketch Mittels Query by Sketch kann der Nutzer seine Vorstellungen vom Ergebnis als Skizze angeben. Es werden hierfür Bildbearbeitungswerkzeuge benötigt. Der Benutzer soll aus vorgefertigten Formen wie Rechtecken, Ellipsen und Polygonen wählen können, wie es in VideoQ unter 3.5 bereits Anwendung findet. Außerdem soll auch freies Zeichnen mit einem Pinsel in auswählbarer Größe möglich sein. Löschen von einzelnen Figuren oder der gesamten Zeichenfläche soll in wenigen Schritten realisierbar sein. Der Nutzer sollte die Größe und die räumliche Beziehung dieser Formen durch Verschieben, sowie die Vordergrund- und Hintergrundebenen, wie es in QBIC unter 3.1 zu sehen ist, verändern können. Die Auswahl der Farben der einzelnen Objekte muss intuitiv sein, z.B. durch Klick auf einen Punkt in der Farbpalette. Eine exakte Angabe der einzelnen Farbwerte sollte zusätzlich angeboten werden. Ein Benutzer könnte nach Bildern suchen, deren Abgebildetes er nicht spezifizieren, sondern nur deren dominante Farben er angeben kann. Aus diesem Grund wird eine Suche anhand der überwiegend auftretenden Farben in Query by Sketch integriert. Gut umgesetzt wurde das bei QBIC Colour Search. Die Zeichenwerkzeuge müssen übersichtlich nah aneinander angeordnet werden, funktional abgetrennt von der Zeichenfläche und den Eingabefeldern für die einzelnen Attribute. Exemplarisch sei hier die Bildbearbeitungssoftware Photoshop zu nennen, wo die Zeichenwerkzeuge in einer

mit Symbolen versehenen vertikalen Leiste dargestellt werden. Die Symbole der Werkzeuge sollen die Funktionalität auf Anhieb deutlich machen. Einfache Symbole und die Verwendung von Metaphern für die Werkzeuge helfen dem Nutzer einen schnellen Einstieg in das Zeichnen der Skizze zu bekommen. Neben Farben kann der Nutzer den Figuren auch Texturen zuordnen, die in einer Palette zur Auswahl stehen. Daraus resultiert die Notwendigkeit zwischen der Sicht für die Farben und der für die Texturen der Objekte wechseln zu können. In VideoQ wurde diese Funktionalität gut umgesetzt. Vorstellbar ist auch eine Annotierung einzelner Bereiche der Skizze. So kann die Zeichnung in Segmente unterteilt werden, denen jeweils ein Begriff zugeordnet wird. Das würde dann die textuelle Suche anhand inhaltsbeschreibender Metadaten ersetzen.

Im Vergleich zu Bildern kommen bei Videos neben den räumlichen Abhängigkeiten auch die zeitlichen hinzu. Der Einsatz von Bewegungsvektoren sowohl für die Objekte innerhalb der Skizze als auch für die Kamera ist notwendig. Diese sollen Auskunft über die Bewegungsrichtung und -geschwindigkeit geben. VideoQ bietet hierfür einige gute Ideen, auf die für den Entwurf in Ansätzen zurückgegriffen werden kann. Die Umsetzung dieser sollte einfacher als dort gestaltet werden.

Query by Example In einigen Situationen ist es schwer seine Vorstellungen in Worte zu fassen oder in einer Skizze zu vereinen. Deswegen muss die GUI auch Query by Example unterstützen. Das Anfragebild, das dem Ergebnis ähnlich sehen soll, muss aus einem Verzeichnis ausgewählt werden. Die Dateinamen sind für den Nutzer meist nicht aussagekräftig. Eine kleine Vorschau des angeklickten Bildes ist hilfreich bei dem Finden des richtigen Beispielbildes. Das Anfragebild sollte als solches gegenüber den später angezeigten Ergebnissen erkennbar sein. In FIRE in 3.2 wurde das nicht beachtet, wodurch der Nutzer keinen Überblick über seine Anfrage hat. Falls nicht das gesamte Foto, sondern nur ein Teil für die Suche wichtig ist, so kann der entsprechende Bereich selektiert werden. Für die Auswahl eines zu suchenden Bildbereichs kann ein Button entworfen oder die Maus zum Aufziehen des Ausschnitts verwendet werden. Um den Anfänger nicht zu überfordern, wird erst auf einer höheren Schicht der Benutzeroberfläche die Option angeboten, mehrere Beispielbilder anzugeben. Bei diesen können auch einzelne Teilgebiete als wichtig markiert werden, wobei zusätzlich auch das Merkmal hervorgehoben werden kann, welches für die Anfrage besonders relevant ist.

Bei Videos kann ein Bild angegeben werden, nach dem ähnliche Bilder innerhalb des Videos gesucht werden sollen. Die Eingabe eines Videos als Beispielvideo sollte auch möglich sein.

Suche von Videos anhand der Audioinformationen Videos sollten sowohl anhand visueller Merkmale, wie bisher beschrieben, als auch anhand der Audiospuren gesucht werden können. Eine kombinierte Betrachtung beider Medien ist auch sinnvoll. Um nach einer Audiospur zu suchen, kann der Laie mithilfe eines Mikrofons eine Melodie summen und diese als Anfrage verwenden. Diese Methode wird als Query by Humming bezeichnet. Die Aufnahme der Klänge eines Musikinstruments oder einer Radiosendung seien auch möglich. Es muss deshalb die Option des Imports einer Audiodatei zur Verwendung als Anfrage in Query by Example gegeben sein. Für fortgeschrittene Musikliebhaber sollte es auch möglich sein, die Notenfolge eines Audiostücks einzugeben, sei es über die Tastatur oder über eine grafische Simulation eines Musikinstruments. Eine Charakterisierung der Lautstärke und Tonhöhe kann mittels sprachlicher Begriffe wie laut und leise, hoch und tief vorgenommen werden. Desweiteren soll der Nutzer auch gesondert nach Sprechszenen und nach Szenen mit Musikuntermalung suchen können. Um anhand einer Kombination von einer Skizze, die Angaben über zeitliche und räumliche Bezüge der Objekte im Video liefert, und den Musikspuren suchen zu können, muss auch deren zeitliche Abhängigkeit zueinander definiert werden. Es müssen also Prädikate geliefert werden, die festlegen, ob die Musik zeitgleich zur animierten Skizze ertönt oder diese innerhalb des Videos unabhängig voneinander auftreten.

4.2.2.3 Gewichtete Komposition der Anfragebedingungen

Neben der Suche anhand genau einer Suchbedingung, soll auch die Kombination verschiedener Bedingungen innerhalb einer Suchanfrage ermöglicht werden. Die GUI muss hierfür eine Variante anbieten, über die die einzelnen Forderungen miteinander über Boolesche Operatoren verknüpft werden. Bei einem Anfänger kann diese Funktionalität ausgeblendet werden, indem intern eine gewichtete AND-Verknüpfung aller Einzelbedingungen vorgenommen wird. In den meisten Fällen sucht der Nutzer Objekte, die alle angegebenen Anforderungen erfüllen. In CQQL werden verknüpfte Bedingungen intern für die Auswertung entsprechend der Definition 1 umgewandelt. Um die einzelnen Bedingungen in einer Anfrage zu vereinen, müssen sie eindeutig identifizierbar sein. Das kann z.B. über Farben, textuelle Bezeichner oder Zahlen realisiert werden. In CQQL können die Konjunktion und Disjunktion gewichtet werden, um die Wichtigkeit einer Bedingung gegenüber einer anderen festzulegen. Das bedeutet, dass immer ein Term relativ zu einem anderen Term gewichtet wird. Initial werden die einzelnen Gewichte θ_i durch die Gewichtungsfunktion w^I festgelegt. Auf der untersten Ebene der Multilayer-Oberfläche werden Gewichte automatisch gesetzt. Es ist vorstellbar allen Gewichte den Wert 1 zu geben, sodass alle Einzelbedingun-

gen gleichrangig sind, oder über eine Zufallsfunktion diese setzen zu lassen. Desweiteren kann auch auf Nutzerprofile zugegriffen werden. Diese enthalten Informationen über die eigenen Interessen, welche durch die manuellen Angaben eines Nutzers in einem Fragebogen ermittelt werden oder aus den eigenen früheren Suchanfragen geschlussfolgert werden. Außerdem können auch die Profile anderer Nutzer einbezogen werden, die ähnliche Anfragen gestellt oder gleiche Interessen haben. Ein Profil enthält u.a. die durch Relevance-Feedback gelernten Gewichte für gestellte Anfragen eines Nutzers. Diese können für einen anderen Anwender mit einer ähnlichen Anfrage als initiale Gewichte genutzt bzw. empfohlen werden. Das entspricht dem Verkaufsprinzip von einigen Shopping-Portalen im Internet wie *Amazon*[3] und *Otto*[4], die dem Kunden anhand des eigenen Kaufverhaltens neue Produkte vorstellen, die andere Kunden mit ähnlichen Interessen bestellt haben. Anzumerken sei jedoch, dass trotz vergleichbarer Anfragen und ähnlicher Interessen, die Setzung der Gewichte nicht immer zu einem zufriedenstellenden Ergebnis führen muss, da jede Person individuell ist. Für fortgeschrittene Anwender, die sich auf einer höheren Schicht der GUI befinden, soll neben dem automatischen Setzen der Gewichte auch eine manuelle Angabe dieser ermöglicht werden. Den zwei Termen eines Operators können natürlichsprachliche Begriffe wie *very important, important, neutral, less important, not important* zugeordnet werden, sodass deren relative Wichtigkeit abgeleitet werden kann. Eine andere Möglichkeit wäre die Angabe eines Wertes aus dem Bereich [0,1], wobei ein Term mit 1 am wichtigsten und mit 0 irrelevant ist. Das entspricht der internen Realisierung der Gewichte.

4.3 Ergebnispräsentation

In Abhängigkeit von der ausgewählten Suchmethode soll dem Nutzer die Ergebnismenge adäquat angezeigt werden. Es tritt hierbei das Problem auf, dass mehrdimensionale Informationen, gemeint sind Informationen mit mehr als 2 Variablen, auf einer zweidimensionalen Oberfläche darzustellen sind. Desweiteren ist die Anzeigefläche des Computermonitors begrenzt, sodass nur eine gewisse Anzahl von Objekten gleichzeitig präsentiert werden kann. Es wird außerdem für den Nutzer schwieriger einen Überblick über die Informationsmenge zu behalten, je größer diese wird.

Aus diesem Grund sollen Visualisierungsansätze entwickelt werden, die dem Nutzer viele Informationen gleichzeitig in einer verständlichen Form präsentieren. Der Überblick über eine sehr große Informationsmenge soll geschaffen werden, indem die für ein konkretes Ziel irrelevanten Aspekte weggelassen werden. Die Ansätze sollen unsichtbare Fakten durch

[3] `http://www.amazon.de/`
[4] `http://www.otto.de/shop-de_home`

neue Perspektiven sichtbar machen, Gemeinsamkeiten und Unterschiede können schnell erfasst werden. Die Reduzierung auf das Wesentliche ermöglicht dem Nutzer eine schnelle Orientierung in der großen Ergebniskollektion, sodass dieser das gewünschte Ergebnis in kurzer Zeit findet.

4.3.1 Präsentationsvarianten

Für die einzelnen Suchmethoden müssen Präsentationsmöglichkeiten entworfen werden, dessen Anforderungen an dieser Stelle zusammengefasst werden.

Browsing und Navigation: Beim Browsing hat der Nutzer kein konkretes Ziel vor Augen. Er folgt seinen Assoziationen. Es ist wichtig, dass das Durchstöbern auf einer großen Menge von Bildern bzw. Videos möglich ist. Viele Objekte sollen gleichzeitig angezeigt werden, jedoch in einer überschaubaren Größe. Das einzelne Durchblättern wäre zeitaufwendig und verhilft nicht zu einer Erfassung der zugrunde liegenden Struktur.

Bei der Navigation muss ein Klassifikationsschema entworfen werden, nach dem die einzelnen Medienobjekte eingeteilt werden. Nach welchem Merkmal die Menge strukturiert wird, soll für den Nutzer sofort erkennbar sein. Eine Variante der Einteilung wäre eine Unterscheidung aufgrund des abgebildeten Inhalts. Das erfordert jedoch allerdings einen hohen manuellen Aufwand, da sowohl die Aufstellung einer Einteilung als auch die Vergabe der High-Level-Features für jedes Objekt nur durch Menschen bisher zufriedenstellend bewerkstelligt wird. Bei einer Vergrößerung der Objektmenge kann eine Modifizierung der Klassifikation nötig werden, da das bisherige Schema höchstwahrscheinlich nicht alle Aspekte der Einteilung abdeckt oder eine feinere Untergliederung nötig wird. Problematisch kann auch die eindeutige Zuordnung zu einer Kategorie sein, da die Klassen nicht immer disjunkt sind. Das stellt jedoch keine Hürde dar, weil in der digitalen Kollektion ein Objekt auch mehrfach vorliegen kann im Gegensatz zu dem begrenzten Literaturbestand einer Bibliothek. Bei dem Entwurf eines Navigationsschemas ist es wichtig, dass die Kategorien übersichtlich gegenübergestellt werden. Vorteilhaft ist ein Überblick über sämtliche Ebenen. Ein möglicher Wechsel in einem Schritt von einer Unterkategorie zu einer anderen als der eigenen Oberkategorie erspart so dem Anwender Aufwand.

anfragebasierte Suche: Im Gegensatz zum Browsen und Navigieren hat der Nutzer vorher eine Anfrage formuliert. Diese wird intern in ein CQQL-Äquivalent transformiert und den Objekten der Datenbank entsprechend den Auswertungs- und Gewichtungs-

regeln aus Definition 1 und 2 ein Score aus [0,1] zugeordnet. Es werden hierbei nur die charakteristischen Präferenzen aus den ersten k Objekten des Ranks dem Nutzer angezeigt. Die Relevanz der Ergebnisse muss in der Präsentation ersichtlich sein. Die Score-Werte sind für den Laien uninteressant, für den Profi jedoch hilfreich. Die unterschiedlichen Bedürfnisse können mithilfe der mehrschichtigen Design-Architektur beachtet werden. Wichtig ist vor allem das Verhältnis der Objekte zueinander. Welches relevanter als das andere ist, kann mittels Farbe, Größe, Form, räumlicher Position oder einer bekannten Rangfolge deutlich werden. Bei der Position ist zu beachten, dass in der europäischen Kultur die Leserichtung von links nach rechts bzw. von oben nach unten verläuft. Die Elemente oben links werden vom Benutzer zuerst beachtet. Eine intuitive Abstufung der Ergebnisrelevanz bei einer Auflistung findet demzufolge nach rechts und nach unten statt. Der Blick des Nutzers soll auf die relevanten Objekte gelenkt werden. Das kann durch die bereits erwähnten Gestaltungsmittel realisiert werden. Die gesuchten Objekte zur Anfrage müssen schnell lokalisierbar sein. Der Nutzer muss das Prinzip der Ergebnisanordnung verstehen.

Allgemein muss abgewägt werden, ob ein ein-, zwei- oder mehrdimensionaler Visualisierungsansatz die Ergebnisse übersichtlicher darstellt. Die Wahrscheinlichkeit einer Informationsanhäufung oder -ausdünnung, die die Lesbarkeit erschweren, muss klein gehalten werden. Wichtig für den Nutzer ist eine gute Orientierung und eine einfache und schnelle Bewegung innerhalb der visualisierten Ergebnismenge. Die Idiome für das Navigieren innerhalb des Informationsraumes müssen einfach erlernbar sein. Die Gründe für die Positionierung und Gestaltung der Resultate müssen für den Nutzer ersichtlich sein.

Der Wechsel zwischen den verschiedenen Varianten der Ergebnisdarstellung soll möglich sein. Das Konzept von Brushing und Linking soll hier eingesetzt werden. Linking ist das Verknüpfen von verschiedenen graphischen Darstellungen, wobei diverse Sichten auf die Informationen erzeugt und in einen Zusammenhang gestellt werden [SM00]. Brushing bezeichnet das Selektieren von dargestellten Informationen bzw. Daten, die in anderen Bildern automatisch markiert werden. Brushing in Verbindung mit Linking soll Einsatz finden, damit selektierte Objekte nach dem Wechsel in eine andere Ergebnisvisualisierung schnell wiedergefunden werden können. Ein Wechsel in die Navigationsansicht ist beispielsweise vorstellbar um neue Objekte zu finden, die in der gleichen Kategorie wie das favorisierte Ergebnis sind.

4.3.2 Informationsebenen

In den verschiedenen Ergebnispräsentationen soll der Nutzer abhängig von der Informationsebene weniger oder mehr Informationen über ein Resultat erhalten. Es sollen drei Ebenen bereitgestellt werden:

- Übersichtsebene
- Übergangsebene
- Detailebene

Die Übersichtsebene vermittelt einen Überblick über viele Objekte. Dort werden keine bzw. nur wenige wesentliche Metadaten angezeigt, um eine gut überschaubare Ordnung der Resultate zu wahren. Auf der nächsten Stufe, der Übergangsebene, werden mehr Zusatzinformationen sichtbar. Das kann z.B. der Dateiname sein. Der Nutzer soll nach seinen Bedürfnissen diese Vorgabe verändern können. Eine etwas vergrößerte Darstellung des Objekts innerhalb der Gesamtübersicht verhilft dem Nutzer zu einer schnellen Auswahl des gesuchten Bildes. Eine Variante das umzusetzen ist eine Fokus-Plus-Kontext-Visualisierung. Es werden mehr Details über die fokusierten Objekte und weniger über die restlichen Resultate angezeigt, ohne das diese komplett verborgen werden [Man02]. Einzelheiten eines Objekts werden mit dem Blick auf den Kontext kombiniert. Um noch mehr Metadaten über ein Ergebnis zu erhalten, wechselt man in die Detailansicht. Dort können alle bereitgestellten Informationen abgerufen werden. Darunter zählen u.a. die Auflösung des Bildes bzw. die Bitrate des Videos, Uhrzeit und Datum der Aufnahme und Belichtungseinstellungen der Kamera. Der Kontext wird hier nicht mehr angezeigt, das einzelne Ergebnis wird nur betrachtet. Für den Wechsel zwischen den Informationsebenen müssen Idiome entwickelt werden, die nur wenige Benutzerinteraktionen abverlangen.

4.3.3 Medientyp- und hardwareabhängige Präsentation

In Abhängigheit von den Sucheinstellungen können auch verschiedene Medientypen als Ergebnis zurückgeliefert werden. Diese müssen anhand eines Icons oder aufgrund anderer Merkmale voneinander unterscheidbar sein.

Verschiedene Medientypen benötigen unterschiedliche Darstellungsweisen. Auf der Übersichtsebene können Bilder als Thumbnails auftreten. Im Gegensatz zu Bildern haben Videos einen Zeitbezug, der bei der Präsentation beachtet werden muss. Dort kann ein Keyframe als Repräsentant dienen. Es stellt sich die Frage, welches von den vielen Keyframes hierfür verwendet werden soll. Eine einfache Variante wäre die Auswahl des ersten

Keyframes des Videos. Möglich wäre auch ein Keyframe aus der zeitlich längsten Szene anzuzeigen, da diese wahrscheinlich charakteristisch für den Film ist.

Auf der Übergangsebene kann ein Foto bzw. eine Grafik vergrößert werden, sodass der Kontext noch gut zu erkennen ist. Um ein Video auf dieser Stufe aussagekräftiger zu präsentieren, können mehrere etwas vergrößerte Keyframes, z.B. eins von jeder Szene, in zeitlicher Folge hintereinander wie ein Daumenkino zu sehen sein.

In der Detailansicht soll das Vergrößern und Verkleinern eines Bildausschnittes über diverse Zoomstufen sowie frei wählbar möglich sein. Die Größe soll schnell auf die ursprüngliche zurückgesetzt werden können. Das Verschieben des Ausschnitts, so genanntes Panning, erspart dem Nutzer in einer vergrößerten Ansicht das Heraus- und das erneute Heranzoomen. Bei einem Video müssen Funktionen bereitgestellt werden, die das Video starten, stoppen, anhalten, im Zeitraffer oder Zeitlupe sowohl vor- als auch rückwärts abspielen. Ein Springen zu der nächsten Szene oder zum nächsten Kapitel soll einfach gehandhabt werden. Der Nutzer soll auch die Abspielstartposition innerhalb des Videos festlegen können. Falls das Video Audiospuren enthält, dann soll der Ton ein- und ausschaltbar sein. Neben diesen Kriterien soll das System auch automatisch die Qualität des Films durch die Veränderung der Bitrate einstellen können. Das ist nötig, da bei einer Verbindung zur Datenbank über eine langsame Netzwerkverbindung die Zeit des Herunterladens und das störungsfreie Abspielen eines Videos für den Nutzer wichtig ist. Das gleiche Prinzip gilt für ein Bild, bei dem die Auflösung heruntergesetzt werden kann.

Falls ein Ergebnis aufgrund der Hardware nicht dargestellt werden kann, kann ein Medientyp in einen anderen umgewandelt werden. Man spricht dann von einer Medienumsetzung [Sch06]. Beispielsweise kann der Nutzer keine Lautsprecher besitzen. Das Umwandeln der Sprache eines Videos kann in Text erfolgen, der als Untertitel zu sehen ist. Bei einem Softwareproblem kann zum einen das Speicherformat in ein anderes umgewandelt werden.

Um körperlich beeinträchtigten Menschen bei der Interaktion zu unterstützen, sollen erweiterte Auswahlmöglichkeiten bereitstehen. Farbenblinde Menschen können die in der GUI eingesetzten Farben auf einen anderen Farbbereich wechseln lassen. Anwender mit einer Schwäche im Nahsehen haben die Möglichkeit einer Vergrößerung der gesamten Anzeige, vor allem der Schriftgröße.

4.4 Relevance-Feedback

Bei der anfragebasierten Suche entsteht durch die Auswertung einer gewichteten Anfrage mithilfe der *eval*-Funktion eine Totalordnung der Ergebnisse. Es wird dem Nutzer jedoch nur die charakteristische Präferenzmenge P präsentiert, die aus dem Rank ermittelt wird.

Das bedeutet, dass das System nur die minimale Präferenzmenge anzeigt, mit der die ersten k Objekte des Ranks eindeutig bestimmt werden können. Wie unter 2.2.3 beschrieben, gibt der Nutzer im Zuge des Relevance-Feedback auf diesen Systempräferenzen P seine eigenen Präferenzen P' an. Eine Präferenz bezieht sich jeweils auf zwei Ergebnisobjekte, die miteinander bezüglich der Relevanz zur Anfrage verglichen werden.

Möglichkeiten der Präferenzangabe

Es gibt verschiedene Möglichkeiten Präferenzen durch den Anwender angeben zu lassen. Zum einen können Ergebnisse explizit verglichen werden: $o_1 \geq o_2$ *für* $o_1, o_2 \in O_{Rank}$. Zum anderen kann dieses implizit erfolgen. Eine Einteilung der Ergebnisse in Kategorien wie beispielsweise relevant und irrelevant ist vorstellbar. Wichtig bei der letzten Variante ist das Bereitstellen von Algorithmen, die aus den Nutzerbewertungen Präferenzpaare ableiten, sodass eine Halbordnung erzeugt wird, weil diese für den Lernalgorithmus *prefsToWeight* benötigt wird.

Desweiteren muss entschieden werden, ob die Ergebnisbewertung direkt in der Ergebnispräsentation stattfindet, wie es z.B. in FIRE der Fall ist, oder ob dafür zusätzliche Fläche benötigt wird, die von der Ergebnisanzeige separiert ist. Wie viel Platz im zweiten Fall zusätzlich benötigt wird, hängt vor allem von der durchschnittlichen Bewertungsanzahl des Nutzers pro Iterationsschritt ab. Falls die Fläche für eine überdurchschnittliche Zahl an Evaluierungen nicht ausreichend ist, so muss die Anzeige dynamisch angepasst werden. Vorstellbar ist statt einer Gesamtansicht einen Ausschnitt darzustellen, dessen Einordnung in das Ganze schnell möglich ist.

Fitts' Law

Ein wichtiger Faktor bei der Auswahl zwischen den genannten Möglichkeiten ist die benötigte Zeit, welche für eine gewisse Anzahl von Ergebnisbewertungen aufgewendet werden muss. Hier kann Fitts' Law aus dem Jahr 1954 herangezogen werden. Das Gesetz besagt, dass die Zeit für das Erreichen eines Zielobjekts von der Entfernung zum Ziel und von dessen Größe abhängig ist [DFAB03]:

$$T = a + b * log_2 (D/S + 1)$$

T ist die Zeit, die vergeht, bis das Zeigegerät auf das Ziel bewegt wird, wobei unter einem Zeigegerät z.B. eine Maus und unter dem Ziel z.B. ein Button oder ein Icon zu verstehen ist. D ist die Distanz zwischen momentaner Position des Zeigegeräts und Ziel, S ist die Größe des Ziels. Die Konstanten a und b werden empirisch ermittelt. Sie hängen von dem speziellen Gerät sowie von der Fähigkeit des Anwenders ab, mit diesem umzugehen.

In [DFAB03] wird beispielhaft für das Ziehen mit der Maus für a ein Wert von 135 ms und für b = 249 ms/bit angegeben. Aus der Formel ist zu schließen, dass das Ziel mit der Maus umso schneller erreicht wird, je größer das Ziel und je geringer der Abstand zu diesem ist. Um die Interaktionszeit des Nutzers bei der Bewertung gering zu halten, müssen die zurückzulegenden Wege mit der Maus klein gehalten werden und für die Evaluierung größere GUI-Elemente bzw. -Bereiche vorhanden sein. Für identische Bewertungen mehrerer Objekte sollen Idiome bereitstehen, die einer Gruppe von Ergebnissen einen Wert zuweisen können anstatt dieses seriell vornehmen zu müssen. Der Nutzer möchte im Allgemeinen den Aufwand für die Bewertungen so gering wie möglich halten.

Reversible Präferenzangaben

Nachdem der Anwender mit der Angabe der Präferenzen fertig ist, muss dieser das dem System mithilfe eines Idioms mitteilen. Das System wandelt anschließend die Nutzerpräferenzen P' in Gewichte um. Die Auswertung der angepassten gewichteten Anfrage liefert eine veränderte Ergebnismenge, in der der Nutzer erneut seine Präferenzen festlegen muss. Eine gleichzeitige Anzeige aller Evaluierungen vorhergehender Iterationsschritte, um diese im Nachhinein ändern zu können, würde zulasten der Übersichtlichkeit gehen und ist deshalb nicht ratsam. Stattdessen wird die in der 6. goldenen Regel aus 4.1.3 aufgestellte Forderung nach Reversibilität erfüllt, indem die Ansicht auf eine ältere eines bestimmten Iterationsschrittes zurückgesetzt werden kann. Bisher werden Gewichte, nicht aber Präferenzen, im Nutzerprofil gespeichert. Der Nutzer wäre irritiert, wenn vorhergehende Ansichten nicht die Präferenzen enthalten würden, die er gesetzt hatte. Durch die aktuelle Ergebnismenge und das Sichern von Gewichten jedes Schrittes ist nicht gewährleistet, dass die früheren Präferenzen bestimmt werden können. Unter 2.2.3 wird darauf hingewiesen, dass aus dem Rank die nutzlosen, reflexiven und transitiven Präferenzen entfernt werden und nur die minimale Präferenzmenge, die k-äquivalent ist, angezeigt wird. Aus der Ergebnismenge können im Nachhinein deshalb nur die Präferenzen ermittelt werden, die Top-k-äquivalent sind. Der Nutzer konnte jedoch andere angegeben haben. Die Abbildung *prefsToWeight* ist nicht bijektiv. Aus diesem Grund müssen zusätzlich zu den Gewichten auch die Präferenzen im Profil gespeichert werden. Veränderungen älterer Präferenzen sind somit durch das Bereitstellen einer History-Funktion möglich. Mithilfe einer Undo-Funktion kann in einen früheren Systemzustand zurückgekehrt werden. Wenn der Nutzer dann zu der Entscheidung kommen sollte, dass er seine älteren Präferenzen doch nicht ändern will, so kann er die durch Undo zurückgenommenen Aktionen mit einer Redo-Funktion wieder ausführen lassen. Er gelangt somit zur aktuellen Ansicht.

Empfehlung von Präferenzen

Der Nutzer möchte im Umgang mit der GUI so wenig wie möglich Aufwand betreiben, um sein Ziel-Rank zu erhalten. Um die Anzahl der abzugebenden Beurteilungen zu verringern, kann das System dem Anwender gute Präferenzen vorschlagen. Das können zum einen Präferenzen sein, die zwischen weit entfernten Ergebnisobjekten vergeben werden. Damit sind Ergebnisse gemeint, deren Score-Werte eine annähernd maximale Differenz erzeugen. Zum anderen können es auch Präferenzen sein, die die vorhandenen Präferenzregionen halbieren. In der Abbildung 2.7 auf der rechten Seite ist der zweite Ansatz bei einer Anfrage mit zwei Gewichten veranschaulicht. Die Schnittmenge der beiden Präferenzregionen entspricht der gültigen Gewichtswerte, die beide Präferenzen erfüllen. Die Gewichtsfunktion w, die erst bei einem kleinen Schnitt möglichst eindeutig aufgestellt werden kann, kann so mit einer geringeren Anzahl an Präferenzen ermittelt werden. Neben dem Vorteil des geringeren Nutzeraufwands kann der Lernalgorihtmus dadurch schneller ausgeführt werden [SZ09]. Beim Entwurf muss eine Lösung gefunden werden, wie dieses umgesetzt werden soll. Fraglich ist, ob der Nutzer auf die Vorschläge eingehen wird oder selbst andere Präferenzpaare auswählen will.

Manuelle Modifikation der Gewichte

Beim Relevance-Feedback werden dem Nutzer die Gewichte verborgen, er arbeitet nur mit Präferenzen. Mit diesen werden die Gewichte der Anfrage automatisch an die Nutzerrelevanz angepasst. Fortgeschrittenen Anwendern soll jedoch auch zusätzlich das manuelle Modifizieren der Gewichte ermöglicht werden. Dieses soll auf höheren Ebenen der Multi-Layer-Architektur zur Verfügung stehen.

4.5 Feedback, Fehlerbehandlung und Hilfestellung

Feedback

Damit der Nutzer die richtige Vorstellung von dem Systemverhalten bekommt, soll, Bezug nehmend auf die 3. goldene Regel aus 4.1.3, bei einer Aktion der Nutzer nützliches Feedback vom System erhalten. Eine Unterteilung der Nachrichten nach Wichtigkeit ist sinnvoll:

- Information
- Warnung
- kritische Nachricht

Die Nachrichtentypen sollen grafisch leicht unterscheidbar sein. Eine Bestätigung bei einer erfolgreichen Übernahme der vom Nutzer angegebenen Präferenzen ist nötig. Wenn die Durchführung einer Aktion mehr als 2 Sekunden braucht, muss der Benutzer informiert werden, dass diese Aktion gerade in Bearbeitung ist. Er vermutet sonst einen Fehler. Für kleine Operationen kann ein animiertes Objekt das anzeigen. Bei Wartezeiten über 10 Sekunden ist eine Information über den aktuell erreichten Fortschritt über einen Balken oder die benötigte Restzeit zur Vollendung der Aufgabe notwendig. Der Nutzer kann währendessen seine Zeit sinnvoll gestalten. Der Suchprozess in der Multimedia-Datenbank ist ein solches Beispiel. Ein Signalton oder eine blinkende Anzeige in der Taskleiste kann die Beendigung der Suche anzeigen, wenn das Fenster durch andere Anwendungen verdeckt oder minimiert wurde, siehe Regel 4 aus 4.1.3.

Fehlerbehandlung

Die Internationale Organisation für Normung hat Richtlinien für die Ergonomie der Mensch-System-Interaktion in der ISO 9241 niedergelegt. U.a. wird dort gefordert, dass Dialoge die vom Nutzer bekannten Begriffe verwenden sollen [Eur95]. Desweiteren kann man dort lesen, dass Fehler dem Nutzer verständlich erklärt werden sollen, damit dieser eine Korrektur vornehmen kann. Mit der Meldung, dass ein Syntaxfehler vorliegt, weiß der Nutzer nicht umzugehen. Eine Nachricht mit der Angabe, was genau falsch geschrieben wurde, ist verständlicher. Aus der 5. Forderung aus 4.1.3 ist zu entnehmen, dass der Nutzer nicht zu einer erneuten vollständigen Eingabe gebeten werden soll, sondern nur eine Korrektur des fehlerhaften Teils vornehmen muss. Falls eine automatische Verbesserung möglich ist, soll dem Nutzer die Fehlerquelle gezeigt und ihm Korrekturvorschläge unterbreitet werden. Beispielsweise können bei der Vergabe von Präferenzen durch den Nutzer Zyklen entstehen. Diese werden durch topologisches Sortieren vom System erkannt. Eine automatische Korrektur ist, wie bereits in 2.2.3 beschrieben, nicht ratsam. Stattdessen soll der Nutzer auf den Widerspruch verständlich hingewiesen werden und eine Präferenz des Zyklus einfach entfernen können. Falls ein Medientyp nicht darstellbar ist, wie es in 4.3.3 erläutert wird, so soll der entsprechende Hinweis gegeben werden. Das kann sowohl über ein Symbol als auch über eine Dialogmeldung passieren. Wenn beispielsweise die Downloadgeschwindigkeit für das Abspielen eines Videos in Realzeit nicht ausreichend ist, dann soll das System die Bitrate heruntersetzen bzw. bei Verwendung von Pufferalgorithmen die Wartezeit verlängern, bis der Puffer einen geeigneten Füllstand hat.

Allgemein sollen Fehler vermieden werden, jedoch darf das System nicht so restriktiv sein, dass der Nutzer in seiner Interaktion stark eingeschränkt wird. Stattdessen soll das System mit den Fehlern des Nutzers rechnen und Eingaben rückgängig machen können.

Dabei sollen auch mehrere Aktionsfolgen zurückzunehmen sein, siehe Regel 6 aus 4.1.3.

Hilfestellung

Dem Nutzer soll Hilfe angeboten werden, um einen Einstieg in das Programm zu bekommen, neue Funktionalitäten kennenzulernen und um Lösungsmöglichkeiten für auftretende Herausforderungen bzw. Probleme zu erhalten.

Tutorials, die die grundlegende Arbeitsweise mit dem Programm ausführlich erklären, sowie häufig gestellte Fragen (FAQ) sollen unter der Hilfe zu finden sein. Diese kann über den Menüpunkt *Help* angesteuert werden. Das ist den meisten Nutzern aus anderen Programmen bekannt. Dadurch ist kein langes Suchen nötig. Die Tutorials können neben der Beschreibung der allgemeinen Interaktionprozesse auch Hinweise geben, wie eine Suchanfrage besser gestellt werden kann, um das gewünschte Ergebnis zu erhalten. Das kann z.B. bedeuten, dass Skizzen als Anfrage eher einfach gehalten werden sollen. Das liegt daran, dass Features nur zu einem gewissen Grad aus den Bildern und Videos extrahiert werden und Informationen dadurch verloren gehen bzw. Analyseverfahren für den Vergleich der Anfrage mit Datenbankobjekten noch nicht die semantische Lücke überwunden haben.

Desweiteren soll auch eine Kontexthilfe angeboten werden, die direkt von der momentan bearbeiteten Aufgabe abhängt. Diese kann zum einen über einen Hilfe-Button bezogen werden. Zum anderen sollen auch Tooltips und Statusbar- oder Infobox-Nachrichten angezeigt werden. Diese Informationen beschreiben die fokusierte Funktion und nützliche Funktionserweiterungen. In VideoQ wurde bereits die Funktion eines Buttons in einem Feld angezeigt, siehe Abbildung 3.11 Bereich 17, jedoch noch nicht dessen weitere Optionen. Der Grad der angezeigten Hilfsinformationen ist abhängig von dem Wissensstand des Nutzers und wird durch die Multi-Layer-Architektur auf jeder Ebene unterschiedlich gehandhabt.

Um den Einsteiger bei der Suche und der Ergebnisbewertung zu begleiten, soll auch ein Assistent bereitstehen. Es muss beim Entwurf außerdem überlegt werden, ob und wie ein Assistent dem Nutzer weit entfernte Präferenzen vorschlagen kann. Dadurch würde sich die Gewichtungsfunktion stärker verändern als bei der Angabe einer Präferenz zwischen Ergebnissen mit kleinen Score-Differenzen. Eine Verringerung der Iterationsdurchläufe, bis der Nutzer ein zufriedenstellendes Ergebnis erhält, ist dadurch wahrscheinlicher. Um den Ablaufprozess deutlich zu machen, wäre eine Anzeige eines Flussdiagramms nützlich. Der aktuelle Zustand, in der sich der Nutzer gerade befindet, wird in diesem Diagramm hervorgehoben.

5 GUI-Prototypen für eine visuelle Mediensuche

In diesem Kapitel werden Prototypen für die Benutzerschnittstelle eines CBIRS entworfen, deren Anforderungen im Kapitel 4 aufgestellt wurden. Wie bereits erwähnt, soll die GUI vorwiegend für das Image-Retrieval entwickelt werden. Eine Erweiterung auf andere Medientypen wie Video soll möglich sein. Zuerst werden die Prototypen vorgestellt und anschließend in Hinsicht auf Benutzerfreundlichkeit bewertet.

5.1 Vorstellung der Prototypen

In diesem Abschnitt werden verschiedene Ideen für eine GUI vorgestellt, die dem Benutzer eine inhaltsbasierte Bildsuche ermöglicht. Zuerst wird der grundlegende Aufbau beschrieben. Es folgen die Entwürfe, wie die Suche und die Ergebnispräsentation gestaltet werden. Am Ende wird ein Überblick über das Zusammenspiel der einzelnen Komponenten gegeben.

5.1.1 Grundlegendes

In einer Studie aus dem Jahr 2002 wird berichtet, dass erfahrene Benutzer von haushaltsüblichen PCs ungefähr 45 % ihrer Zeit verschwenden aufgrund einer unübersichtlichen Menüführung, unlesbaren Dialogen und schwer aufzufindenden Funktionen [CLB+02].

Aus diesem Grund soll bei dem Entwurf eine einfache Bedienung der Benutzeroberfläche im Vordergrund stehen. Die GUI soll sich grundlegend aus den folgenden Teilen zusammensetzen, die in Abbildung 5.1 zu sehen sind:

1. **Titelleiste:** Der Name und das Symbol des Programms sowie Buttons zum Minimieren, Maximieren und Schließen des Fensters befinden sich hier.
2. **Menüleiste:** Die Menüleiste besteht aus den einzelnen Menüobjekten, deren Einträge erweiterte Funktionen zur Verfügung stellen. Der Nutzer kann hier u.a. auf

die Hilfe zugreifen, sein Nutzerprofil einsehen und die Tabs für die verschiedenen Suchmethoden und Ergebnispräsentationen aufrufen.

3. **Tableiste:** Um die Suchanfrage von den Ergebnissen räumlich zu trennen, werden verschiedene Tabs hierfür bereitgestellt. Diese ermöglichen zwischen den Ansichten der Suche und der Ergebnisvisualisierung umzuschalten.
4. **Anzeigebereich eines Tabs:** Hier befinden sich abhängig vom gewählten Tab die Interaktionselemente für die Angabe der Suchanforderungen bzw. für die Darstellung der entsprechenden Ergebnisse.
5. **Informationen über die Multi-Layer-Architektur:** Der Nutzer kann hier erkennen, auf welcher Ebene der mehrschichtigen GUI er sich befindet. Desweiteren kann er zu einer anderen Ebene wechseln.
6. **Hilfe:** Kontextinformationen werden angezeigt, um den Nutzer bei der Interaktion zu unterstützen.

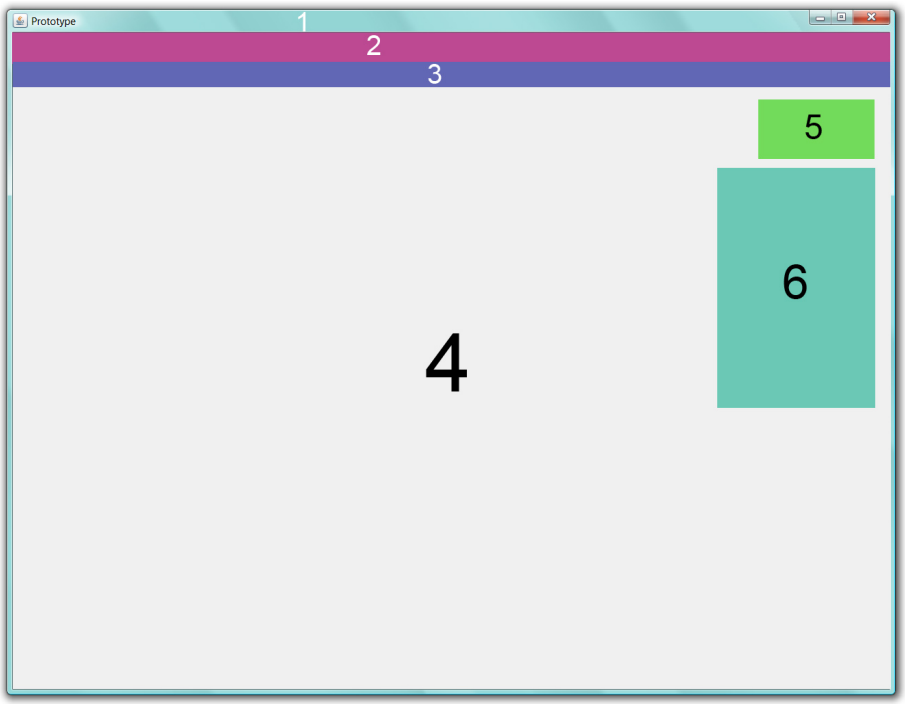


Abbildung 5.1: grundlegender Aufbau der GUI

Benutzerprofil

Die Software soll verschiedene Benutzerprofile verwalten können. Beim Start des Programms fragt das System Nutzername und Passwort ab. Das ist notwendig, damit das

System für einen Nutzer die gelernten Gewichte, die für eine Anfrage zu einem zufriedenstellenden Ergebnis geführt haben, speichern und für spätere Anfragen bei der initialen Gewichtung einbeziehen kann. Die Daten anderer Benutzerprofile können, wie schon unter 4.2.2.3 beschrieben, auch bei der Empfehlung in Betracht gezogen werden. Desweiteren kann der Nutzer durch das Führen von Profilen auf seine älteren Anfragen zurückgreifen. Auch das Zurückspringen auf eine frühere Ergebnisansicht der aktuellen Anfrage ist machbar, um gesetzte Präferenzen vorhergehender Iterationsschritte zu verändern. Diese Aktionen sind unter dem Menüpunkt *Profile/History* möglich. Die Interessen des Nutzers können unter *Profile/Settings* verändert werden. Dort ist auch der Eintrag zu finden, auf welcher Ebene der Multi-Layer-Architektur er sich gerade befindet. Andere Personen wollen wahrscheinlich auch an dem gleichen Rechner ihre Suchanfragen eingeben. Damit das eigene Profil nicht verfälscht wird, wird das Abmelden und Anmelden unter *Profile/Log Out/In* angeboten. Es ist auch möglich sich als Gast anzumelden, jedoch kann dann keine automatische initiale Gewichtung vorgenommen werden, die auf den eigenen Interessen und älteren Anfragen beruht.

Multi-Layer-Umsetzung

Um sowohl Anfängern den Einsteig in das Programm zu erleichtern, als auch die Bedürfnisse der fortgeschrittenen Nutzer zu erfüllen, soll die Benutzerschnittstelle in drei Interaktionsebenen unterteilt sein. Im Bereich 5 der Abbildung 5.1 sollen die Ebenen grafisch abgebildet werden. Eine einfache Umsetzung wäre die Anzeige von drei Stufen einer Treppe, wobei die aktuelle Ebene zur Hervorhebung umrandet wird. Abbildung 5.2 zeigt diese. Der Anfänger ist beim ersten Start des Programms auf der untersten Schicht der Benutzerschnittstelle. Der Wechsel in eine höhere Ebene, um über einen größeren Funktionsumfang der Software zu verfügen, ist durch einen Klick auf die zweite Stufe möglich. Es ist aber auch vorstellbar, dass dem Nutzer ein Wechsel durch das System vorgeschlagen wird. Jeder Nutzer hat ein Benutzerprofil. Dort kann neben den Anfragen und Interessen auch vermerkt werden, welche Funktionen er wie oft aufgerufen hat. Bei Überschreiten eines Schwellwertes einer Variablen, kann dem Nutzer der Dialog zu dem Ebenenwechsel angezeigt werden. Das Benutzerprofil kann über den Menüpunkt *Profile* eingesehen werden. Bei Erreichen der letzten Stufe der GUI stehen dem Nutzer alle Funktionen zur Verfügung. Makros und Shortcuts erleichtern hier die Arbeit.

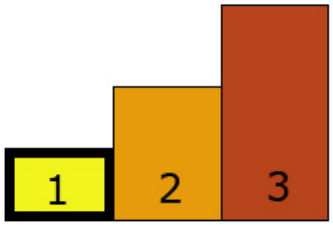


Abbildung 5.2: bildliche Darstellung der Ebenen der Multi-Layer-Architektur

Feedback und Hilfe

Das System gibt dem Nutzer Rückmeldungen u.a. in Form von Dialogen. Diese werden in die Kategorien Information, Warnung und kritische Nachricht unterteilt. Für eine schnelle Unterscheidung der Nachrichtentypen können diese mit Symbolen gekennzeichnet werden. Ein umkreistes i kann für eine Information, ein Dreieck mit einem Ausrufezeichen für eine Warnung und eine Hand auf einer Form, das einem Stoppschild ähnelt, kann für eine kritische Nachricht stehen. Abbildung 5.3 zeigt mögliche Icons. Eine Ampel, dessen aufleuchtende Farbe entsprechend der Wichtigkeit der Nachricht gesetzt wird, ist auch leicht verständlich. Eine farbliche Titelleiste des Dialogfensters, die den Ampelfarben angepasst ist, ist statt der Symbole auch denkbar.

Abbildung 5.3: Icons für die Dialogtypen
links: Information, Warnung und kritische Nachricht [Axi09]
rechts: Ampeläquivalent für Information [Mar09]

Für Aktionen, die länger als 2 Sekunden in ihrer Abarbeitung benötigen, kann eine animierte Sanduhr oder eine Uhr mit einem rotierenden Zeiger verwendet werden. Bei Prozessen, die länger als 10 Sekunden dauern, soll ein Fortschrittsbalken mit einer Prozentanzeige den Nutzer darauf hinweisen. Beispielhaft für den Suchprozess ist das Dialogfenster in Abbildung 5.4.

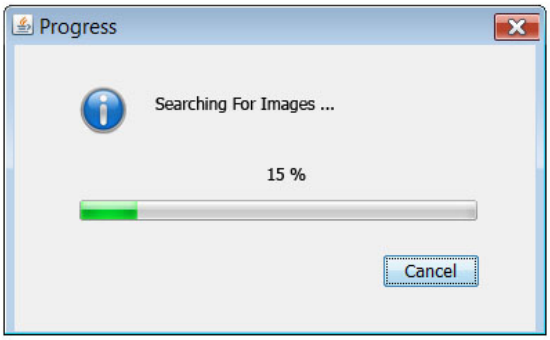


Abbildung 5.4: Dialog über Suchfortschritt

Der Nutzer erhält Kontextinformationen in der Infobox auf der rechten Seite, siehe Abbildung 5.1 Bereich 6. Dieses Fenster kann er beliebig an eine andere Bildschirmposition verschieben. Falls er das Fenster, nachdem er es geschlossen hat, wieder benötigt, kann er es über das Menü *Help/Infobox* erneut aufrufen. Der Inhalt der angezeigten Informationen ist abhängig von der Ebene, in der sich der Anwender momentan befindet.

Bei jedem Wechsel auf eine höhere Stufe der GUI wird automatisch ein Tab geöffnet, der übersichtlich die neu verfügbaren Funktionen vorstellt und kleine Beispiele liefert. Auf der ersten Ebene wird zusätzlich ein Video angezeigt, dass das grundsätzliche Arbeiten mit der Software vorstellt und den Anwendungszweck verdeutlicht. Soll der Tab nicht mehr angezeigt werden, so kann dieser schnell über einen Close-Button rechts im Tab geschlossen werden. Das ist hier einfacher gelöst als bei dem Image-Retrieval-System Emir aus 3.4. Dort werden die einzelnen Tabs nur umständlich über das Menü geschlossen.

5.1.2 Suchformulierung

Die verschiedenen Suchverfahren werden in Tabs organisiert. Die wissenschaftliche Einteilung der Suchmethoden in Browsing, Navigation und anfragebasierte Suche ist für den Nutzer nicht selbsterklärend. Den meisten Anwendern ist nicht der Unterschied zwischen Browsing und Navigation bekannt. Desweiteren sind diese Verfahren hauptsächlich in der Interaktion des Nutzers mit dem System verschieden. Bei der Navigation sucht der Nutzer zielorientiert, beim Browsen stöbert er ziellos in der Datenbank. Aus diesen Gründen werden die verschiedenen Möglichkeiten anhand der Suchausführung in manuelle und automatische Suche aufgeteilt. Browsing und Navigation sind unter dem Tab *Manual Search* und die anfragebasierte Suche unter dem Tab *Automatic Search* zu finden. Die Tabs lassen sich, wenn sie durch den Nutzer geschlossen worden sind, über den Menüpunkt *Search* erneut aufrufen. Im oberen Anzeigebereich eines jeden Tabs für die Suche befinden sich zwei

Radio-Buttons. Der Nutzer kann hier zwischen Bildern und Videos als Ergebnistyp wählen. Der Platz für weitere Radio-Buttons ist vorhanden, um eine Erweiterung um andere Medientypen zu ermöglichen. Der untere Bereich des Tabs verändert sich entsprechend der Auswahl.

5.1.2.1 Browsing und Navigation

Für das Browsen und Navigieren werden im Folgenden diese Ansätze vorgestellt:

- Self-Organizing-Map
- Tree-Map

Self-Organizing-Map

Das sequentielle Browsen in der Bilddatenbank ist zu aufwendig und kann dem Nutzer nicht zugemutet werden. Stattdessen soll eine Menge von Bildern nach ihrer visuellen Ähnlichkeit sortiert präsentiert werden. Bilder werden durch Feature-Vektoren beschrieben. Ein Foto ist einem anderen Foto ähnlicher, je kleiner die Distanz der beiden zugehörigen Vektoren ist. Abhängig von den verwendeten Features, welche Farben, Texturen oder Formen charakterisieren können, kann der Vektor einige hundert Dimensionen besitzen. Der Mensch kann sich jedoch nicht in hochdimensionalen Vektorräumen orientieren. Self-Organizing-Maps (SOMs) sind hierfür ein geeignetes Mittel. Diese von Teuvo Kohonen 1982 eingeführte Visualisierungs- und Abstraktionstechnik bildet einen hochdimensionalen Eingaberaum auf einen niedrigdimensionalen Ausgaberaum ab, indem sich der Nutzer bewegen kann [Koh82, Koh01].

Neuronen werden hier auf einem quadratischen Gitter angeordnet, wobei jedes von ihnen durch einen n-dimensionalen Gewichtsvektor $m_i = (\mu_{i1}, \mu_{i2}, \ldots, \mu_{in})^T$ beschrieben wird, der aus dem Eingaberaum stammt. Der Begriff Neuron stammt aus der Biologie und wird hier im übertragenen Sinn verwendet. Es ist eine Nervenzelle, welche auf Erregung spezialisiert ist. Zu Beginn werden die Gewichtsvektoren der Neuronen zufällig initialisiert. Diese lernen, indem ein Trainingsvektor $x = (\xi_1, \xi_2, \ldots, \xi_n)^T$ an das Gitter angelegt wird. Das Neuron mit dem Gewichtsvektor, der dem Trainingsvektor am ähnlichsten ist, also die geringste Distanz

$$\min_i \{d(x, m_i)\} = d(x, m_c)$$

aufweist, ist das Gewinnerneuron und hat den Index c. Der Gewichtsvektor m_c sowie die Vektoren der Nachbarneuronen von c werden nach der Regel

$$m_i(t+1) = m_i(t) + h_{ci}(t)[x(t) - m_i(t)]$$

angepasst, wobei t eine diskrete Zeitkonstante ist. Die Funktion h_{ci} ist die Nachbarschaftsfunktion, die mit größerwerdenden Werten von t sowie für Neuronen i mit ansteigender Entfernung von c abnimmt. Diese Funktion sorgt dafür, dass zu Beginn der Lernphase eine Grobsortierung der Neuronenkarte stattfindet und anschließend nur noch kleinere Anpassungen in direkter Nähe des Gewinnerneurons vorgenommen werden. Es werden viele Trainingsvektoren zufällig aus dem Eingaberaum ausgewählt und damit gewisse Gewichtsvektoren der Neuronen an diese angepasst. Ähnliche Neuronen sind nach der Lernphase auf der Karte nah beieinander, während unähnliche weiter entfernt voneinander liegen. Wird ein Trainingsvektor häufiger angelegt als andere, so ist der Bereich auf der Karte stärker ausgeprägt. Das Prinzip ist dem Nervensystem des menschlichen Gehirns nachempfunden, weshalb SOMs eine Art von neuronalen Netzen sind.

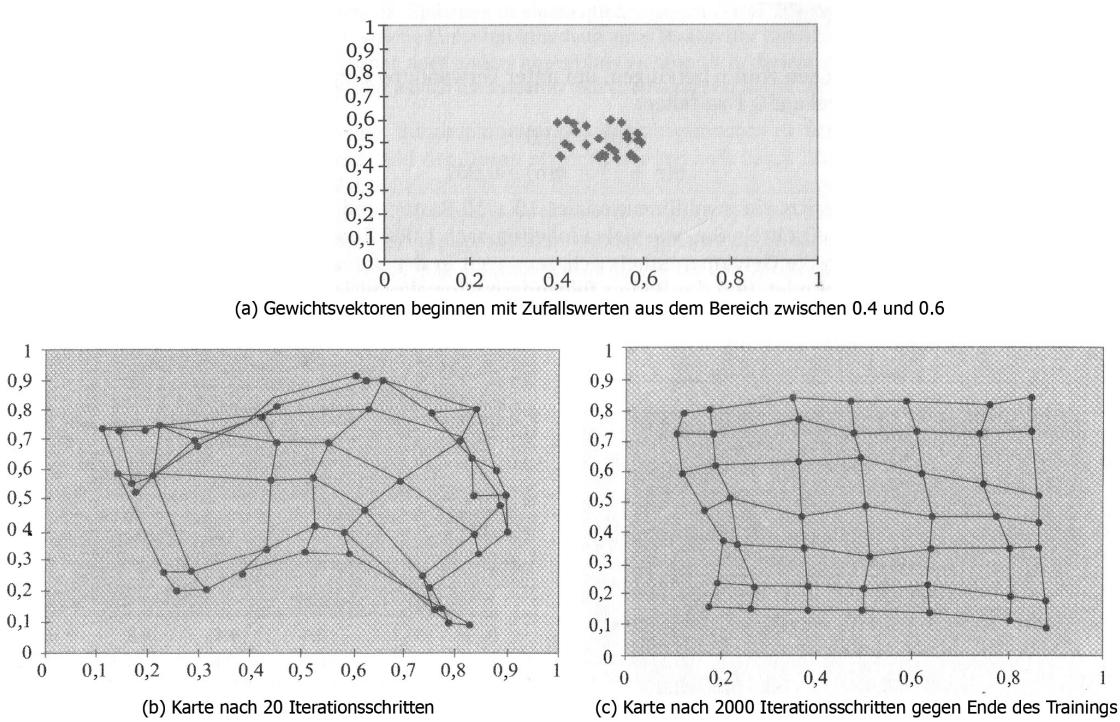


Abbildung 5.5: Lernprozess von Neuronen mit zweidimensionalen Gewichtsvektoren [Cal03]

Die Abbildung 5.5 zeigt den Lernprozess für Neuronen mit zweidimensionalem Gewichtsvektor $(\mu_{i1}, \mu_{i2})^T$. Die Trainingsvektoren sind in diesem Beispiel aus einer einheitlichen Verteilung gezogen, sodass nach dem Training die Karte fast gleichmäßig den Eingangsraum abdeckt. Nachbarschaftsbeziehungen der Nervenzellen werden mithilfe von Linien dargestellt.

SOMs können beim Browsen verwendet werden, um dem Nutzer ähnliche Bilder in Clustern anzuzeigen, damit dieser sich besser in einer großen Menge von Bildern zurechtfindet. Das Prinzip wird beispielsweise bei der Software Image Sorter[1] verwendet, die an der Hochschule für Technik und Wirtschaft in Berlin entwickelt wurde. Die Position eines Neurons kann maximal von einem Bild eingenommen werden, damit sich Bilder bei der Präsentation nicht überschneiden. Desweiteren hat die Karte für mehr Bilder Platz als tatsächlich einzuordnen sind. In Image Sorter bestehen die hochdimensionalen Gewichtsvektoren aus den Werten des abgewandelten MPEG-7 Color-Layout-Deskriptors, der als stark komprimiertes 8x8 Pixel Thumbnail anzusehen ist [BRGF08]. Typischerweise werden Bildermengen in 20 Iterationen sortiert, wobei für 200 Bilder weniger als 50 ms bei einem Prentium 4,3 GHz Prozessor benötigt werden.

Wie Image Sorter beispielsweise 200 Bilder aus der Privatsammlung nach der Position der Farben sortiert, ist in der Abbildung 5.6 zu sehen. Neben der klassischen Kartenansicht ist auch eine Darstellung auf einer rotierbaren Kugel verfügbar, die jedoch aufgrund des eingeschränkten Sichtbereichs keinen Überblick mehr bietet.

[1] `http://imagesorter.softonic.de/`

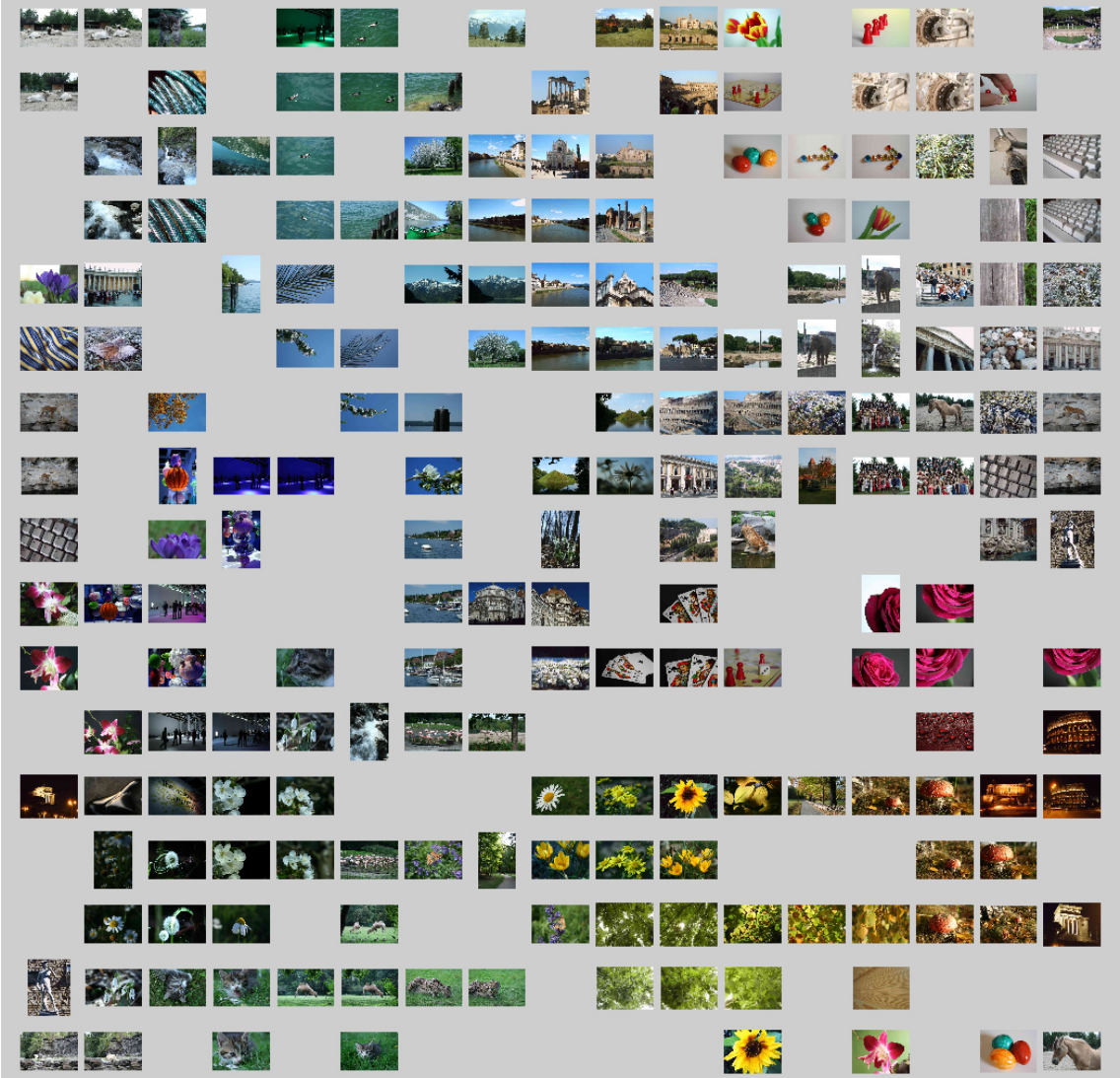


Abbildung 5.6: Sortierung von 200 Bildern nach der Position der Farben in einem Bild mit Image Sorter 3.0

Ein Vergleich verschiedener Farb-, Textur- und Formfeatures sowie lokaler Deskriptoren über mehrere repräsentative Bilddatenbanken hat ergeben, dass das Farbhistogramm zur Ähnlichkeitsbestimmung am besten geeignet ist [DKN08]. Die Klassifizierungsfehlerrate (ER) ist im Durchschnitt mit 22,1 % unter allen Features am niedrigsten. Ein Anfragebild wird hierbei richtig klassifiziert, wenn das mit einer Distanzfunktion gefundene ähnlichste

Bild für das Anfragebild q relevant ist.

$$ER = \frac{1}{|Q|} \sum_{q \in Q} \begin{cases} 0 & \text{wenn ähnlichstes Bild relevant ist} \\ 1 & \text{sonst} \end{cases}$$

Desweiteren ist die Mean Average Precision (MAP) im Durchschnitt über alle Datenbanken mit 58,1 % für das Farbhistogramm am größten. Precision ist das Verhältnis der Anzahl relevanter Bilder in der Ergebnismenge zu der Anzahl der Ergebnisbilder. Für die Berechnung von MAP sei auf [DKN08] verwiesen. Aus diesem Grund soll bei der Verwendung einer SOM für die GUI das Farbhistogramm die Werte des Eingaberaums, der Gewichts- und Trainingsvektoren bestimmen, wobei jeder Farbbereich des Histogramms jeweils eine Dimension darstellt. Für die Sortierung von Grauwertbildern, welche z.B. im medizinischen Bereich in Form von Röntgenaufnahmen auftreten, muss auf ein anderes Feature zurückgegriffen werden.

Probleme bei großen Bilddatenbanken Sollen Tausende von Bildern sortiert werden, wovon bei üblichen Bilddatenbanken auszugehen ist, so werden die Thumbnails der Fotos aufgrund der begrenzten Bildschirmgröße zu klein auf der SOM dargestellt. Eine Lösung können hierarchische SOMs (HSOMs) sein. Auf der linken Seite der Abbildung 5.7 sind diese schematisch zu sehen. Der Nutzer soll beim Starten des Programms zuerst einen Überblick erhalten, indem nur die oberste SOM-Ebene angezeigt wird. Diese stellt eine grobe Einteilung weniger Bilder in Cluster anhand der Farbe dar. Zu jedem Cluster existiert jeweils eine weitere SOM. Sie zeigt zu einem gewählten Foto weitere aus dem ähnlichen Farbbereich. Nach diesem Prinzip setzt sich eine weitere Unterteilung der SOM auf unteren Ebenen fort. Wissenschaftler am Department of Information Science an der Universität von Otago haben HSOMs für ihren Prototypen COVIC eingesetzt.Abbildung 5.8 stellt links die SOM auf höchster Ebene dar. Nach einem Klick auf das Foto mit dem Auto im linken Bereich der Karte wird die zugehörige SOM tieferer Schicht geöffnet, welche eine feinere Granularität aufweist. Im Gegensatz zu Image Sorter ist bei COVIC eine Überlagerung der Bilder möglich, was jedoch beim Browsen in der hier zu entwickelnden GUI ausgeschlossen wird. Eine gitterartige Struktur ist außerdem für den Lesefluss besser geeignet.

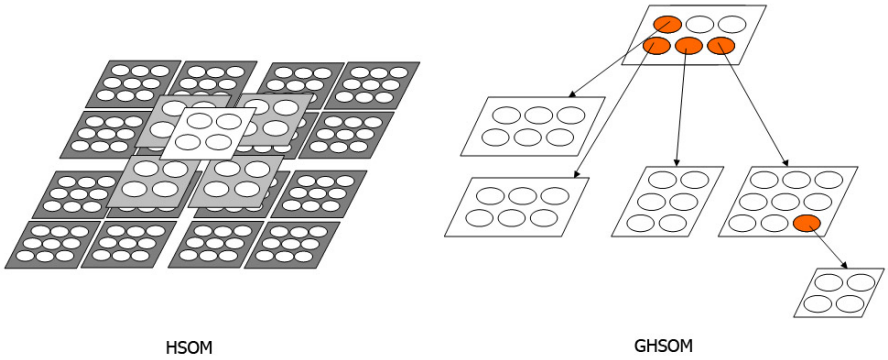


Abbildung 5.7: Abwandlung der klassischen SOMs - HSOM und GHSOM [Rau09]

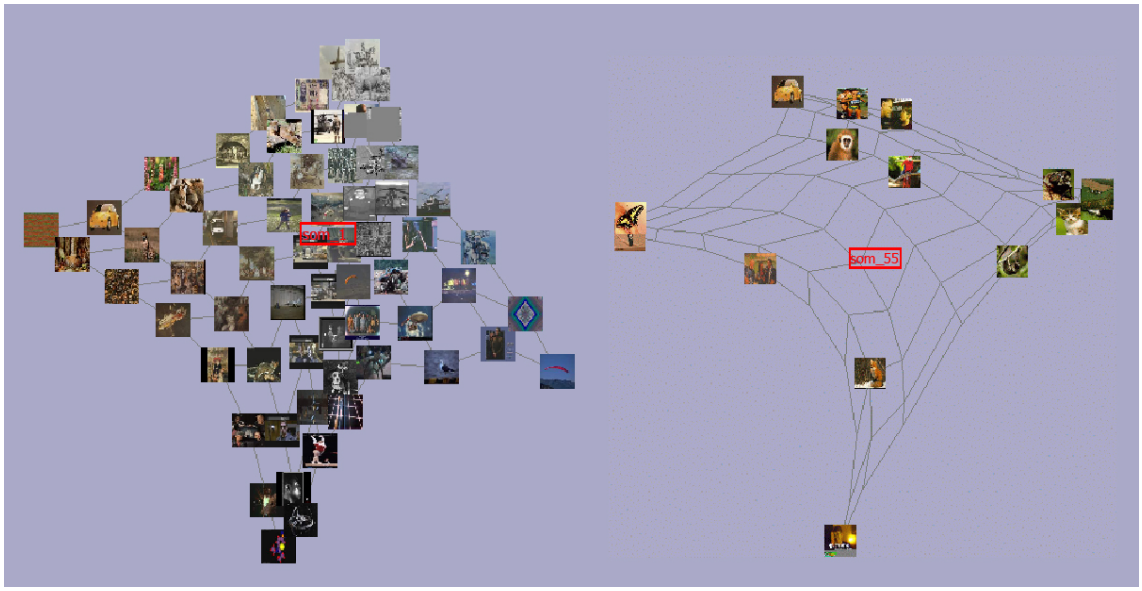


Abbildung 5.8: Verwendung von HSOM beim Prototyp COVIC; links: SOM 1 der Ebene 1; rechts: SOM 55 der Ebene 2 bei Auswahl des Fotos mit abgebildetem Auto [DZP04]

Bei einer großen zu sortierenden Bildmenge bietet die HSOM neben der klassischen SOM auch den Vorteil, dass der Lernprozess schneller ist. Die einzelnen Karten sind kleiner, weshalb weniger Vergleiche durchzuführen sind, um das Gewinnerneuron c zu finden, und weniger Gewichtsvektoren anzupassen sind. Bei beiden Varianten müssen jedoch vor dem Training bereits die Größenparameter der Karte bzw. Karten festgelegt werden. Bei unbekannten Datenbeständen ist es besser, die Größe einer Karte sowie die Hierarchietiefe dynamisch durch das System anpassen zu lassen. Das Problem kann durch den Einsatz

von wachsenden hierarchischen SOMs (GHSOMs) gelöst werden. Eine neue SOM entsteht dort, wo die Abbildung der Trainingsvektoren noch nicht genau genug ist. Die genauen Kriterien, um Neuronen zu einer Karte hinzuzufügen bzw. eine neue SOM anzulegen, können in [DMR01] nachgelesen werden. Mit den Parametern τ_1 und τ_2 kann die Granularität der Datenrepräsentation pro Kartenebene und die höchste Granularität der Datenrepräsentation festgelegt werden. Abhängig von diesen kann die Hierarchie sehr tief mit kleinen Karten sein oder flach mit mehr Bildern pro SOM. In Tests sollen die optimalen Werte für die Parameter ermittelt werden, um eine hohe Benutzerfeundlichkeit zu gewährleisten.

Hierarchiedarstellungen für eine thematische Klassifikation Die farbliche Sortierung der Bilder ist für ein komfortables Browsen in der gesamten Datenbank gut geeignet. Durch die Verwendung des Low-Level-Features Farbe kann jedoch keine eindeutige Aussage über das Abgebildete getroffen werden. Dem Nutzer soll jedoch zusätzlich eine Navigation anhand der dargestellten Themen angeboten werden, weil dieses Klassifikationsschema schnell zu dem Finden eines gesuchten Bildes verhelfen kann. Die Zuordnung der Fotos zu den thematischen Klassen wird aufgrund der semantischen Lücke hauptsächlich manuell vorgenommen, wobei Clusteransätze und das Extrahieren von High-Level-Features wie z.B. die Fotobeschreibung eine Vorarbeit hierfür leisten können [Hen08]. Die Hierarchie der Kategorien kann durch eine Liste dargestellt werden. Diese ist bereits aus vielen Anwendungen bekannt, Einsatz findet sie beispielsweise im Windows Explorer. In Abbildung 5.9 ist auf der linken Seite eine Klassifikation in Listendarstellung zu finden. Eine Hierarchie hängt von der zugrundeliegenden Bilddatenbank ab, weshalb hier nur eine erdachte Klassifikation zur Verdeutlichung dient. Neben dieser zweidimensionalen Darstellung können auch Cone-Trees bzw. Cam-Trees eingesetzt werden, die eine zusätzliche Dimension zur Präsentation benötigen, wie es die Abbildung 5.9 auf der rechten Seite zeigt.

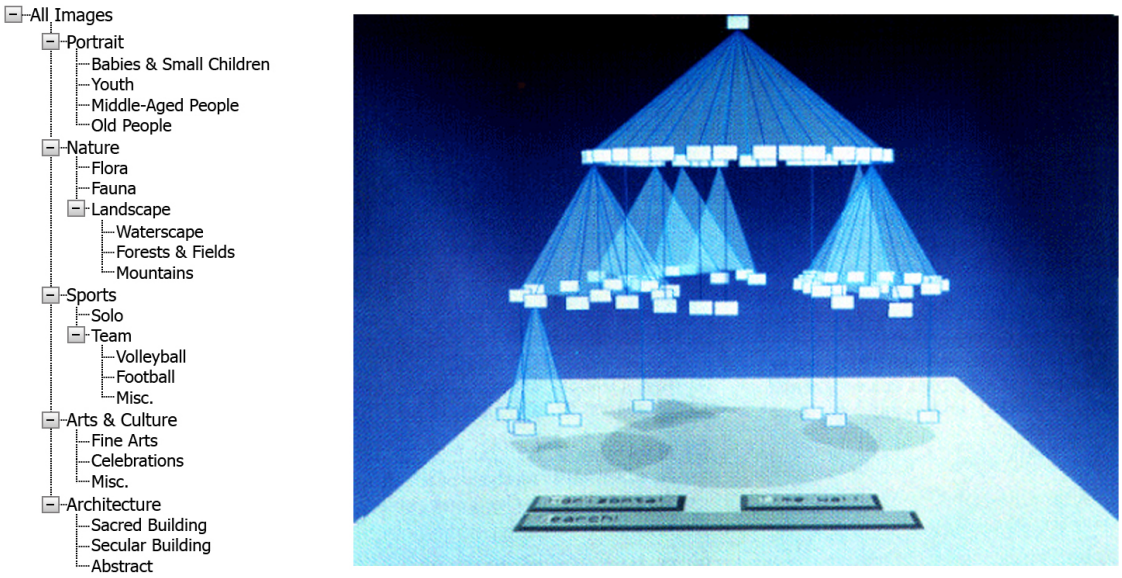


Abbildung 5.9: links: Beispielhierarchie in Listendarstellung; rechts: Cone-Trees [RMC91]

Bei Cone-Trees werden die Kinder eines Knotens in einem horizontal ausgerichteten Kreis angeordnet, der zusammen mit den Verbindungslinien zum übergeordneten Knoten einen Kegel bildet [Spe07]. Zu Beginn wird nur ein Kegel angezeigt. Wenn ein Knoten ausgewählt ist, wird der entsprechende Kegel tieferer Ebene eingeblendet [Arn06], wobei mit zunehmender Tiefe der Durchmesser eines solchen abnimmt. Die Kegelhülle wird transparent gehalten, um die im Hintergrund befindlichen Knoten sehen und auswählen zu können [Naj03]. Ein selektierter Knoten wird durch eine animierte Rotation des Kegels um die eigene Achse in den Vordergrund gedreht. Der Pfad von der Wurzel bis zum anvisierten Knoten wird hervorgehoben. In Abbildung 5.9 sind die Knoten nicht beschriftet dargestellt. Cam-Trees sind das horizontale Pendant zu den Cone-Trees. Dessen Knotennamen sind aufgrund der Ausrichtung besser lesbar.

Bewertung der Hierarchiedarstellungen Cone-Trees nutzen den zur Verfügung stehenden Platz aufgrund der rotierbaren Kegel gut aus, jedoch beansprucht diese Variante im Vergleich zu der Listenform mehr Platz. Zwar werden die Kegel bei Cone-Trees transparent gehalten, jedoch verdecken einige Knotennamen andere Knoten. Die kleiner werdenden Radien in tieferen Ebenen führen zu einer schlechteren Übersicht. Desweiteren zeigt eine Studie, dass Probanden in der listenartigen Baumpräsentation schneller Knoten auffinden als innerhalb der Cone-Trees [Arn06]. Es muss jedoch angemerkt werden, dass die Testkandidaten im Vergleich zu der Liste von den Cone-Trees begeistert waren und von

einem besseren Gefühl für die Struktur gesprochen haben. Viele Argumente sprechen für die Verwendung der Liste für die Hierarchiedarstellung bei der Navigation, weshalb diese in der GUI eingesetzt werden soll.

Kombination von GHSOMs und thematischen Listen Die Liste der Kategorien soll im linken Bereich des Tabs *Manual Search* angeordnet werden. Rechts daneben sollen sich entsprechend des ausgewählten Zweiges die zugehörigen Bilder befinden. In vielen Bildbetrachtungsprogrammen wie z.B. in ACDSee Foto-Manager[2] werden die Fotos als Thumbnails in Spalten und Reihen aufgelistet, in denen sich mithilfe von Scrollleisten bewegt wird. Zur besseren Orientierung innerhalb der angezeigten Bilder sollen stattdessen in diesem Ansatz die Fotos nach Farbe in GHSOMs sortiert präsentiert werden. Der Nutzer kann in der Wurzelebene der thematischen Klassifikation die repräsentativen Bilder der Farbcluster überblicken. Bei Bedarf kann dieser durch einen Klick auf ein Foto in die detailliertere Ansicht gelangen, welche nur die zu dem gewählten Foto farbähnlichen anzeigt. Die Anordnung dieser ist durch die entsprechende SOM zweiter Ebene festgelegt. Das ist bereits ausführlich beschrieben worden. Möchte der Nutzer Bilder mit einem bestimmten abgebildeten Inhalt sehen, so wählt er den passenden Listeneintrag auf der linken Seite durch Klick aus. Die Ansicht auf der rechten Seite wird modifiziert. Die in diese Kategorie eingeordneten Fotos werden auch hier farblich in GHSOMs sortiert angezeigt. Das setzt eine erneute Berechnung von GHSOMs voraus, weil die Bilder aufgrund des abgebildeten Inhalts gefiltert werden müssen. Die Abbildung 5.10 stellt den Tab der manuellen Suche in der Kombination von GHSOMs und Liste dar. Die Kategorie *Nature* ist ausgewählt und der Nutzer befindet sich in der höchsten GHSOM-Ebene.

Fotos können somit sowohl nach abgebildetem Thema als auch nach der Farbe gesucht werden. Das Browsen und das Navigieren wird durch die kombinierte Verwendung von GHSOMs und der Listendarstellung ermöglicht.

[2] `http://store.acdsee.com/store/acd/de_DE/DisplayProductDetailsPage/productID.106893200`

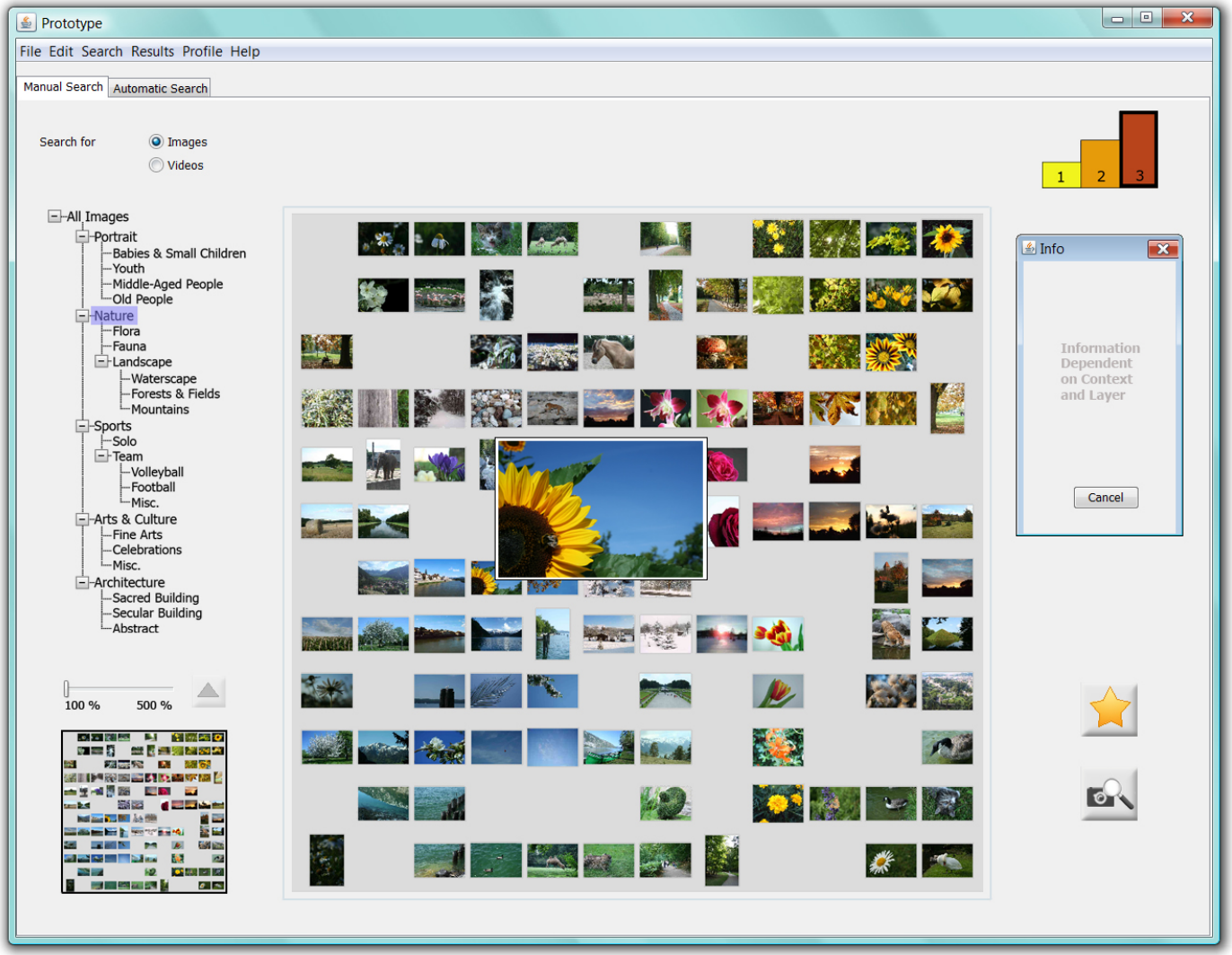


Abbildung 5.10: Tab *Manual Search* in der Variante mit GHSOMs und Liste

Initial befindet sich der Nutzer in der Übersichtsebene, wie unter Abschnitt 4.3.2 eingeführt. Mit dem Scrollrad kann die gesamte Ansicht der gleichen GHSOM-Ebene vergrößert werden. Um den Überblick zu bewahren, wird links unten in der Ecke eine Übersichtskarte angezeigt, die die SOM in kleinerer Ansicht zeigt und den Ausschnitt markiert, welcher gerade zu sehen ist. Der Ausschnitt kann mit Drag & Drop verschoben werden. Das funktioniert sowohl in der SOM als auch in der Übersichtskarte. Die SOM hat keine Kanten, weil die linke und rechte Seite sowie die obere und untere verbunden sind. Die Karte befindet sich sozusagen auf einer kugelförmigen Oberfläche, sie wird jedoch ohne Verzerrung dargestellt. Dieses Prinzip ist Image Sorter entlehnt. Das Herauszoomen ist über die entgegengesetzte Laufrichtung des Scrollrads möglich. Desweiteren steht ein Schieberegler bereit, mit dem einfach der Vergrößerungsgrad angegeben werden kann. Verweilt die Maus auf einem Bild für eine gewisse kurze Zeit, so wird das Bild an dieser Position etwas vergrößert angezeigt, falls es nicht bereits durch Zoomen diese Größe besitzt. Das ist die sogenannte Übergangsebene. In der Abbildung 5.10 ist deshalb das Bild mit der Sonnenblume vergrößert. Um weitere Bilder in den ähnlichen Farben ansehen zu können, muss der Nutzer auf das Foto klicken. Es erscheint die GHSOM der zweiten Ebene für dieses Foto, welche eine feinere Abstufung der Farben zeigt. Die Zoomfunktion vergrößert nur in der gleichen Hierarchieebene der GHSOM. Der Nutzer würde sich wahrscheinlich wundern, wenn durch Scrollen ab einem gewissen Punkt andere Fotos in einer neuen Struktur vorliegen würden. Aus diesem Grund können tiefergelegene GHSOMs nur durch Klicken auf ein Foto eingesehen werden. Soll auf eine höhere Ebene zurückgesprungen werden, so muss mit der rechten Maustaste geklickt werden. Das Gleiche kann auch über einen Klick auf den Button rechts neben dem Schieberegler ausgeführt werden. Dieser Button ist deaktiviert, wenn man sich in der obersten Ebene befindet, was in der Abbildung 5.10 der Fall ist. Will der Anwender das Foto fast bildschirmfüllend betrachten, so genügt ein einfacher Klick auf das Foto. Zu dem Foto werden für den Nutzer interessante Metadaten angezeigt. Falls dem Nutzer ein Bild gefällt, so kann er es per Drag & Drop auf das Stern-Symbol in der rechten unteren Ecke speichern. Durch diese Funktion kann der Nutzer schnell und unkompliziert favorisierte Fotos sammeln. Standardmäßig ist ein Speicherort voreingestellt, der jedoch verändert werden kann.

Desweiteren kann ein gefundenes Foto auch als Beispielbild für eine Anfrage dienen. Auch hier wird dieses durch Verschieben des Bildes auf das darunterliegende Kamera-Lupen-Symbol gelöst. Durch einen Wechsel auf den Tab *Automatic Search* ist das Bild im entsprechenden Bereich sichtbar.

Tree-Map

Tree-Maps stellen eine weitere Variante der Hierarchievisualisierung eines Baumes dar, die in den frühen 1990er Jahren von Ben Shneiderman entwickelt worden sind [Shn92b]. Die ursprüngliche Form, genannt Slice-and-Dice-Tree-Maps, ist für die Darstellung der Ordnerstrukturen auf einer Festplatte konzipiert worden. Seitdem sind viele Abwandlungen für neue Einsatzgebiete entstanden.

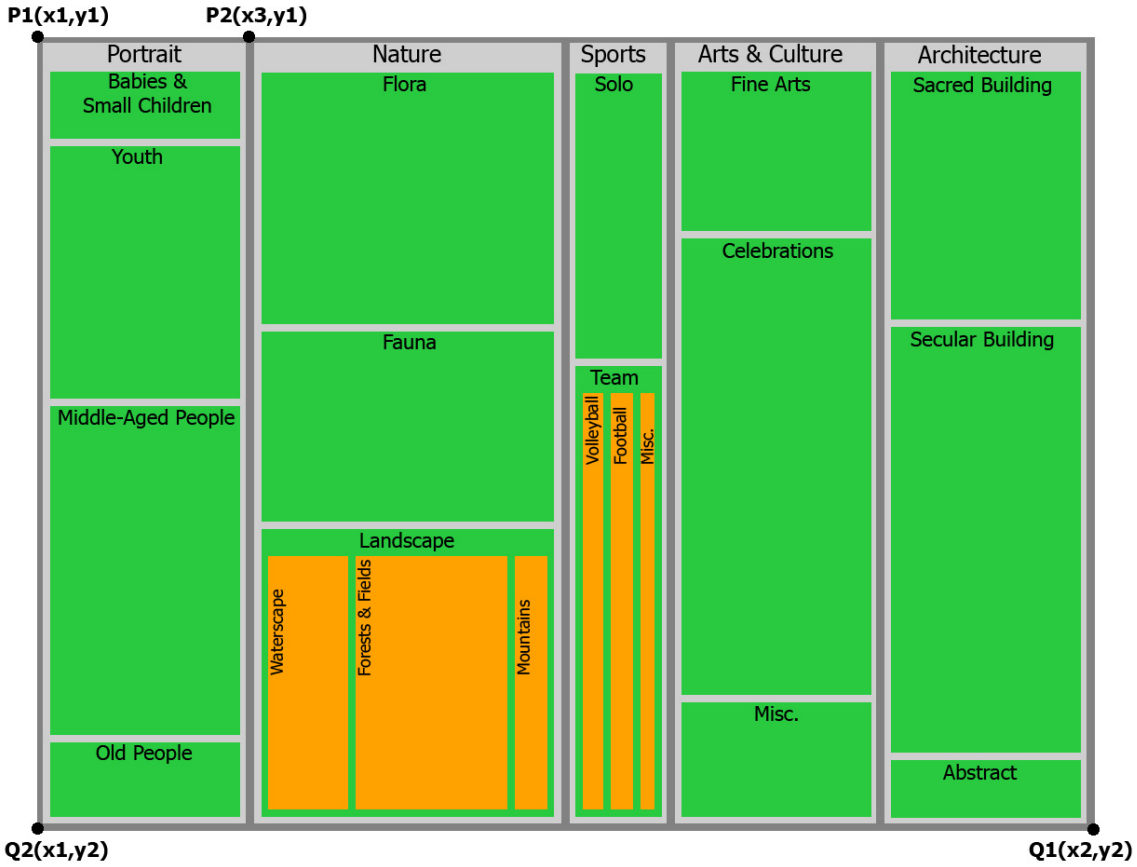


Abbildung 5.11: Slice-and-Dice-Treemap einer Beispielklassifikation

Die Abbildung 5.11 zeigt die Beispielklassifikation aus Abbildung 5.9 im Slice-and-Dice-Layout. Rechtecke stellen die Knoten eines Baumes dar, wobei die Rechtecke der Kindsknoten im Rechteck des Vaters eingeschlossen sind. Das äußere Viereck ist die Baumwurzel und ist durch die Punkte P1(x1,y1) und Q1(x2,y2) festgelegt. Dieses wird horizontal in 5 Rechtecke für die Kinder aufgeteilt. Die relative Größe der Rechtecke ist abhängig von der Gewichtung der Nachfahren. Früher entsprach das der Speicherplatzgröße eines Ordnerinhalts, im vorliegenden Fall gibt die Fläche Auskunft über die Anzahl der Fotos in einer

Kategorie. Für die Position der Trennungslinie zwischen den Themen *Portrait* und *Nature* wird somit folgende Formel verwendet :

$$x3 = x1 + \frac{Fotoanzahl_{Portrait}}{Fotoanzahl_{Wurzel}} * (x2 - x1)$$

Es ist aus der Tree-Map somit ablesbar, dass die *Portrait*-Kategorie genauso viele Bilder enthält wie *Arts & Culture*, jedoch nur halb so viel wie *Nature*. Dieses Verfahren wird rekursiv fortgesetzt, wobei die Abarbeitungsrichtung für jede Ebene der Hierarchie um 90° gekippt wird. Die Unterkategorien von *Portrait* sind somit vertikal angeordnet.

Bewertung Slice-and-Dice-Tree-Map Im Gegensatz zu den bisher vorgestellten Hierarchiedarstellungen kann bei Tree-Maps zusätzlich die relative Größe der einzelnen Knoten abgelesen werden. Desweiteren kann den Rechtecken der Blätter eine Farbe zugewiesen werden, welche eine zusätzliche Information angibt. Ursprünglich hat diese die Dateiart gekennzeichnet. Im Beispiel werden die Farben jedoch nur für die Unterscheidung der einzelnen Ebenen eingesetzt. Anstatt die Rechtecke nur mit Farbe auszufüllen, könnten die enthaltenen Bilder dargestellt werden. Je größer jedoch die Hierarchiestruktur wird, desto kleiner und schmaler werden die Rechtecke. Hinzukommt die Größe der Datenbank, wodurch bei zunehmender Bildanzahl die Thumbnails immer kleiner dargestellt werden. Das verschlechtert die Lesbarkeit. Eine Variante wäre eine Zoom-Funktion anzubieten. Diese würde zwar den zweiten Aspekt abschwächen, jedoch die länglichen dünnen Rechtecke bleiben erhalten.

Quantum-Tree-Maps Um Bilder in den Rechtecken des Tree-Maps optimal zu platzieren, soll im Folgenden der Algorithmus zum Aufstellen von Quantum-Tree-Maps vorgestellt werden. Mit diesem werden Rechtecke als ein Vielfaches einer elementaren Größe generiert, und zwar ein Vielfaches der Fotogröße. Das ist notwendig, weil Bilder ein festes Seitenverhältnis aufweisen. Alle Bilder werden gleich groß dargestellt und bildlich gesprochen auf einem Gitter über alle Rechtecke hinweg angeordnet. Dadurch kann der Nutzer schnell die Gruppen von Bildern überfliegen.

Der folgende Algorithmus ist aus [BSW02] zusammengefasst. Erwähnt werden muss, dass Rechtecke der Kategorien mit einem Seitenverhältnis nah bei 1, also Quadrate, wünschenswert sind, weil sie visuell attraktiver sind und der Mensch die Fläche eines Quadrates besser abschätzen kann [Bed01]. Rechtecke stellen für den einzusetzenden Fall die thematischen Kategorien dar und die Objekte innerhalb der Rechtecke sind die Fotos.

- **Eingabe von $L_1, \ldots, L_n$:** Das ist eine geordnete Sequenz von Nummern. L_i gibt die Anzahl der Objekte an, die im resultierenden Rechteck platziert werden soll.
- **Eingabe von W und H:** Das ist die elementare Größe, die Objektgröße.
- **Eingabe einer Box R:** R ist das Rechteck der Wurzel, in der sämtliche Rechtecke eingeschlossen werden sollen.
- **Ausgabe von $R_1, \ldots, R_n$:** Das ist eine geordnete Sequenz von Rechtecken, die in der Box R liegen. R_i ist groß genug (bzw. möglicherweise größer), um ein Gitter von Objekten unterzubringen, dessen Anzahl durch L_i festgelegt ist.

Der Algorithmus ähnelt QuickSort. Es wird ein Pivotelement L_p gewählt und in der Box platziert, wobei $L_1 \ldots L_{p-1}$ auf der einen Seite des Pivotelements, und $L_{p+1} \ldots L_n$ auf der anderen Seite positioniert werden.

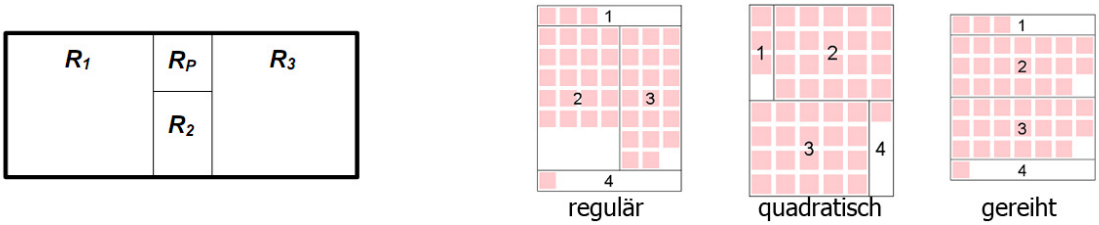


Abbildung 5.12: Quantum-Tree-Map - grundlegende Aufteilung der Box in Rechtecke für $n > 4$ (links) und die drei Varianten für $n = 4$ (rechts) [Bed01]

1. Wenn $n = 1$, dann soll ein Rechteck R_1 berechnet werden, welches exakt L_1 Objekte auf einem Gitter anordnet, welches so gut wie möglich dem Seitenverhältnis der Box nahekommt. Anschließend stoppt der Algorithmus.
2. Wenn $n \leq 4$, dann sollen die Objekte entweder im regulären, quadratischen oder gereihten Layout angeordnet werden. Es soll die Variante genommen werden, dessen durchschnittliches Seitenverhältnis am nahesten zu 1 ist. Die rechte Seite der Abbildung 5.12 zeigt die Unterschiede für den Fall $n = 4$.
3. Wähle das Pivotelement L_P, wobei dieses das mittlere ($P = \frac{n}{2}$) oder das größte Element sein kann oder eines, dass das Ergebnis in Elemente fast gleicher Größe aufteilt.
4. Wenn die Breite von R größer als dessen Höhe ist, so teile R in 4 Rechtecke R_1, R_P, R_2 und R_3 auf, wie es die Abbildung 5.12 auf der linken Seite zeigt. Wenn die Höhe größer als die Breite ist, dann verwende die gleiche Aufteilung, jedoch um 90° gedreht.
5. Stecke L_P in das Rechteck R_P, wobei dessen Breite und Höhe im nächsten Schritt festgelegt wird.

6. Lege die Elemente $(L_1, \ldots, L_{P-1}) = L_A$ in das Rechteck R_1. Die restlichen Elemente außer L_P sollen in zwei Listen $L_B = (L_{P+1}, \ldots, L_r)$ und $L_C = (L_{r+1}, \ldots, L_n)$ aufgeteilt werden, wobei L_B in das Rechteck R_2 kommt und L_C in R_3. r soll so gewählt werden, dass R_P ein Seitenverhältnis so gut wie möglich von 1 bekommt.
7. Gehe für die Listen L_A in R_1, L_B in R_2 und L_C in R_3 rekursiv vor. Beginne hierfür bei Schritt 1.
   a) Verschiebe die Rechtecke in R_P, R_2 und R_3, um ein Überlappen mit R_1 zu vermeiden.
   b) Vergewissere dich, dass R_P und R_2 die gleiche Breite und diese zusammen die gleiche Höhe wie R_1 haben. R_1 muss wiederum auch die gleiche Höhe wie R_3 haben. Ist die Breite oder die Höhe zu klein, so soll diese entsprechend angepasst werden.

Aufgrund der Anpassung der Rechtecksgröße in Schritt 1, muss in den Schritten 7a und 7b nach der Auflösung der Rekursion die Größe angepasst werden. Das ist nötig, damit keine Überlappungen auftreten und die Reihen und Spalten eines jeden Rechtecks aneinander angepasst werden, um ein großes Gitter über allen Rechtecken zu bilden. Das ist in Abbildung 5.12 auf der rechten Seite gut zu erkennen. Einige Stellen im Gitter bleiben jedoch unbesetzt. Der Algorithmus verspricht für eine größere Anzahl an Objekten ein besseres Seitenverhältnis nahe 1 als für kleine Fotomengen. Das liegt daran, dass größere Zahlen mehr Möglichkeiten der Gitterbildung bieten. Beispielweise soll ein Rechteck 1000 Bilder in einer Gitterstruktur darstellen. Hierfür wäre die Aufteilung des Gitters in 31x33, 32x32, 30x34 etc. möglich. Bei insgesamt 5 Bildern ergeben sich hingegen die Varianten 5x1, 2x3 oder 3x2. Es wird mehr Platz verschwendet und das Rechteck weicht stärker vom gewünschten Quadrat ab.

Quantum-Tree-Maps werden beispielsweise in der Software PhotoMesa[3] zur Anzeige von Bildern eingesetzt. Abbildung 5.13 zeigt die aktuelle Version 3.1.2 in der Praxis. 186 Fotos werden in 8 grau umrahmten Rechtecken dargestellt. Die Rechtecke sind die angewählten Ordner aus der Verzeichnisstruktur. Der rote Rahmen zeigt den Bereich, der bei einem Klick auf die linke Maustaste vergrößert werden würde. Verweilt die Maus auf einem Bild, so wird es etwas größer angezeigt und bei einem Doppelklick in hoher Auflösung eingeblendet. Die aktuelle Version der Software stellt jedoch nur ausgewählte Ordner ohne Hierarchie dar.

[3] `http://photomesa.en.softonic.com/`

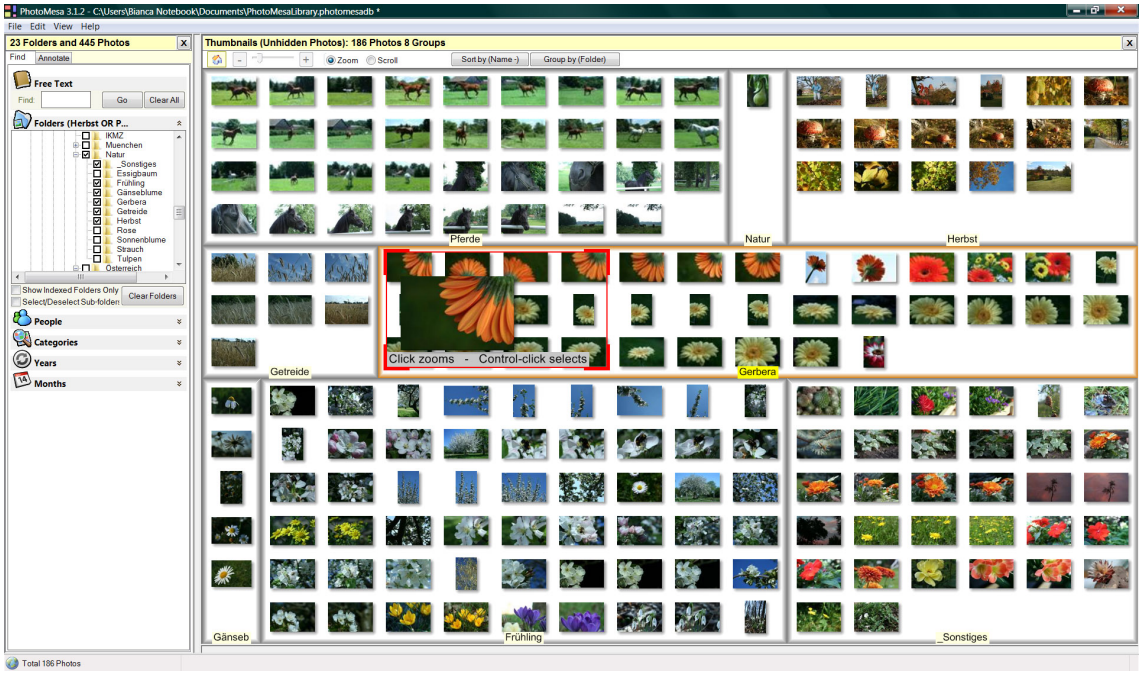


Abbildung 5.13: PhotoMesa 3.1.2 in der privaten Anwendung

Für die GUI in dieser Arbeit sollen die Hierarchieen ersichtlich sein. Die Rechtecke der direkten Nachfolger der Wurzel sollen jeweils verschiedene Rahmenfarben erhalten. Im verwendeten Beispiel kann die Kategorie *Portrait* rot sein, *Nature* grün, *Sports* blau usw. Deren Kinder erhalten Farbabstufungen der Vaterfarbe. Abbildung 5.14 zeigt dieses beispielhaft für die Kategorie *Nature*. Das Rechteck eines Blattes wie *Flora* wird mit der Farbe ausgefüllt. Der Nutzer kann aufgrund der Farben die Kategorien schneller voneinander unterscheiden und die einzelnen Ebenen erkennen. Der Algorithmus behält auch die Ordnung der Kategorien bei, welche von links nach rechts bzw. von oben nach unten verläuft. Im Gegensatz zu der Verwendung von GHSOMs sollen hier alle Bilder angezeigt werden. Es tritt dann das Problem auf, dass bei vielen Bildern die Thumbnails aufgrund begrenzter Bildschirmfläche zu klein sind. Zwar haben die Rechtecke ein besseres Seitenverhältnis als bei Slice-and-Dice-Tree-Maps und sind somit besser lesbar, jedoch sind auch diese ab einer bestimmten Hierarchiestufe nicht mehr erkennbar. Aus diesen Gründen soll Zoomen möglich sein. Dieses wird über das Scrollrad gelöst. Desweiteren kann eine Kategorie vergrößert werden, sodass dessen Rechtecksbreite oder -höhe die maximalen Ausmaße der Ansicht annimmt. Ist die Kategorie ein Blatt, so genügt ein Klick auf die freie Fläche neben einem Foto. Ist die Kategorie eine höhere Ebene, so muss auf den Rahmen geklickt werden. Um zu erkennen, welches Viereck vergrößert werden würde, erscheint beim Dar-

überfahren der Maus, sei es eine freie Fläche eines Blattes oder der Rahmen einer höheren Kategorie, ein auffälliger andersfarbiger Rahmen. Die Idiome bei Quantum-Tree-Maps sind denen des vorgestellten Entwurfs bei GHSOMs sehr ähnlich. Beim kurzen Verweilen mit der Maus auf einem Bild wird auch in diesem Entwurf das Bild etwas vergrößert angezeigt. Erst ab einer bestimmten Fotogröße ist diese Funktionalität aktiviert. Bei sehr vielen kleinen Thumbnails würde die Ansicht sonst zu unruhig wirken, wenn durch kleinste Bewegungen der Maus ein neues Fotos angezeigt wird. Per Drag & Drop kann der Ausschnit verschoben werden, wobei auch hier eine Übersichtskarte zur schnellen Einordnung in die gesamte Tree-Map verfügbar ist. Ein Bild wird durch ein Doppelklick in der Detailansicht präsentiert.

Abbildung 5.14: Quantum-Tree-Map für den Bereich *Nature*

Entscheidung für eine Darstellungsvariante

Beide vorgestellte Varianten, GHSOMs in Kombination mit Listen und Quantum-Tree-Maps, haben ihre Vor- und Nachteile. Während bei Quantum-Tree-Maps direkt erkennbar ist, wie viele Bilder in einer Kategorie enthalten sind, ist das bei GHSOMs nicht der Fall, da die Bilder auf mehreren Ebenen verteilt sind. Auf der anderen Seite können die Rechtecke bei Tree-Maps bei einer großen Anzahl an Bildern so klein werden, dass die Kategorien ohne Zoomen nicht erkennbar nicht. Bei der Listendarstellung hingegen können alle Kategorien eingesehen werden, mit der Voraussetzung alle sind geöffnet. Beide Varianten haben die gleichen Idiome für den Wechsel zwischen der Übersichts-, Übergangs-

und Detailebene. Bleibt die Maus kurz über einem Foto stehen, so wird dieses etwas vergrößert angezeigt. Ein Doppelklick auf ein Bild mit der linken Maustaste öffnet dessen Detailansicht auf einem neuen Tab. Desweiteren sind bei beiden die Bilder in einem Gitter angeordnet. Das gewährleistet einen schnellen Lesefluss und keines der Bilder wird von einem anderen verdeckt. Sind die Bilder einer Kategorie eingeblendet, beispielweise von *Nature*, dann ist bei Quantum-Tree-Maps ersichtlich, welche Fotos in welcher Unterkategorie von *Nature* eingeordnet sind. Bei GSHOMs kann nicht auf Anhieb unterschieden werden, ob ein Bild zu Waterscape oder Forests & Fields gehört. Erst durch Klick auf den Eintrag in der Liste auf der linken Bildschirmseite wird eine Filterung vorgenommen. Das ist jedoch nur ein sehr kleiner Nachteil, da hier nur ein Klick mehr nötig wird. Ein Problem kann bei der ersten Verwendung von GSHOMs auftreten. Ein Nutzer befindet sich auf der höchsten Ebene einer höheren Kategorie. Er entdeckt ein Bild, dessen Motiv er interessant findet. Mit dem Gedanken weitere Fotos mit diesem Motiv zu finden, klickt er auf das Bild. Er gelangt in eine tiefere GHSOM, wo farbähnliche Bilder sortiert aufbereitet sind, jedoch nicht zwingend motivähnliche. Das Problem ist in tieferen Themenkategorien eher nicht von Bedeutung, weil die Wahrscheinlichkeit für ein ähnliches Motiv bei einem tieferen Listeneintrag zunimmt. Im Fall einer großen Bilddatenmenge gehen beide Verfahren unterschiedlich vor. Während beim Quantum-Map alle Bilder gleichzeitig angezeigt werden und eine Zoomfunktion bereitsteht, werden bei den GHSOMs abhängig von den Parametern τ_1 und τ_2 die Bilder auf mehreren Karten verteilt. Im optimalen Fall werden die Parameter so gewählt, dass genügend Bilder auf einer Ebene in gut erkennbarer Größe angezeigt werden, um einen guten Überblick über alle Farbbereiche zu bekommen. Zu viele Fotos verschlechtern die Übersicht und führen zu kleineren Fotos. Zu wenige sind jedoch auch nicht gut, weil dadurch die Hierarchie tiefer wird, welche die Orientierung des Nutzers mehr fordert und mehr Interaktion verlangt. Bei Quantum-Tree-Maps kann es sehr schnell vorkommen, dass die Bilder zu klein angezeigt werden. Das wird verstärkt durch die Tatsache, dass aufgrund der Einordnung der Fotos in Kategorien nicht zwingend alle Gitterpositionen belegt werden. Vergrößern ist deshalb nötig. Der Nutzer steht jedoch dann vor dem Problem, welchen Bereich er zoomen soll. Wenn er keine Vorstellung von dem Thema hat, sucht er im schlechtesten Fall sequentiell die Menge durch. Bei GHSOMs steht er nicht vor dieser Aufgabe. Hier werden neben den Themen die Fotos auch nach Farbe sortiert. Das führt zu einer schnelleren Orientierung. Farben spielen für den Menschen eine große Rolle bei der Unterscheidung von Bildern bzw. beim Feststellen von Ähnlichkeiten. Das Ergebnis der bereits vorgestellten Studie [DKN08], welche die verschiedenen aktuellen Features vergleicht, verstärkt diese Aussage. Aufgrund der genannten Vorteile von GHSOMs gegenüber Quantum-Tree-Maps, vor allem aufgrund des resultierenden geringeren

Zeitaufwands bei der Suche wegen farblicher Sortierung der Fotos, soll diese Variante in Kombination mit Listen bei der GUI eingesetzt werden. In Abbildung 5.10 ist die endgültige Version des Tabs *Manual Search* auf der dritten Ebene der Multi-Layer-Architektur zu sehen. Es ist denkbar, dass in Zukunft weitere Feature-Deskriptoren entwickelt werden, die neben dem Farbhistogramm noch bessere Resultate bei der Ähnlichkeitsanalyse von Fotos liefern können. Diese können dann als Eingabewerte für die GHSOMs verwendet werden.

Multi-Layer-Umsetzung Auf der ersten Ebene der Multi-Layer-Architektur hat der Nutzer nur sehr wenige Einschränkungen. Die Sortierung der Bilder nach Farben ist leicht erkennbar. Um mit dem System zu interagieren kann der Nutzer die gängigen Mausaktionen testen und sieht deren Ergebnis. Zooming und Panning sowie das Auswählen einer Kategorie aus der Listendarstellung sind dem Nutzer bereits aus vielen anderen Anwendungen bekannt, wodurch der Einstieg leicht fällt. Die Funktionsbelegungen der Maustasten stehen zusätzlich in der Infobox. Der Wechsel zwischen der Detailansicht, welche ein Foto sehr groß und mit den zugehörigen Metadaten zeigt, und der Übersichtsebene soll auch für den Anfänger angeboten werden. Die Oberfläche ist übersichtlich gehalten. Um den Nutzer nicht zu Beginn mit der Vielschichtigkeit der SOM zu konfrontieren, kann der Nutzer nicht in tiefere Ebenen wechseln, das ist erst ab der zweiten Multi-Layer-Schicht möglich. Zuerst soll das grundlegende Verständnis aufgebaut werden. Die unterschiedlichen Ansichten der einzelnen Hierarchiestufen der GHSOMs können den Anfänger verunsichern. Das schränkt zwar die Menge an Bildern ein, jedoch stehen auf der obersten SOM für den Anfang genügend Bilder zur Auswahl. Auf der ersten Stufe der Multi-Layer-Architektur verbleibt der Nutzer außerdem nur für eine kurze Zeit. Als Einstieg soll die oberste SOM-Ebene ausreichen. Außerdem kann der Nutzer bei einer konkreten Vorstellung von dem Motiv direkt in der Liste das Thema anklicken und es erscheinen neue Fotos. Die Buttons für das Speichern von favorisierten Bildern und das Angeben eines Beispielbildes für eine Anfrage stehen erst ab der zweiten Stufe bereit. Weil die GUI in diesem Tab relativ einfach gehalten ist, kommen für den Experten auf der dritten Stufe keine weiteren Funktionen hinzu.

5.1.2.2 Anfragebasierte Suche

Bei einer konkreten Vorstellung von dem zu suchenden Bild kann der Nutzer eine Anfrage formulieren. Hierfür steht der Tab *Automatic Search* bereit. Dieser ist in drei Bereiche eingeteilt, welche durch einen dünnen grauen Rahmen räumlich voneinander abgegrenzt

werden. Oben links befindet sich der Bereich *Query by Metadata*, wo klassische Datenbankbedingungen anzugeben sind. Rechts daneben kann der Nutzer unter *Query by Example & Sketch* Fotos als Beispielbilder laden und eigene Skizzen entwerfen, die dem Ergebnis ähnlich sehen sollen. Unterhalb dieser beiden Komponenten befindet sich *Composition & Weigthing*, wo die Bedingungen, die unter *Query by Metadata* und *Query by Example & Sketch* angegeben bzw. gezeichnet wurden, zu einer Anfrage zusammengesetzt und gewichtet werden. Die Abbildung 5.15 zeigt den initialen Zustand des Tabs Query, den der Nutzer auf der Ebene 3 der Multi-Layer-Architektur vorfindet.

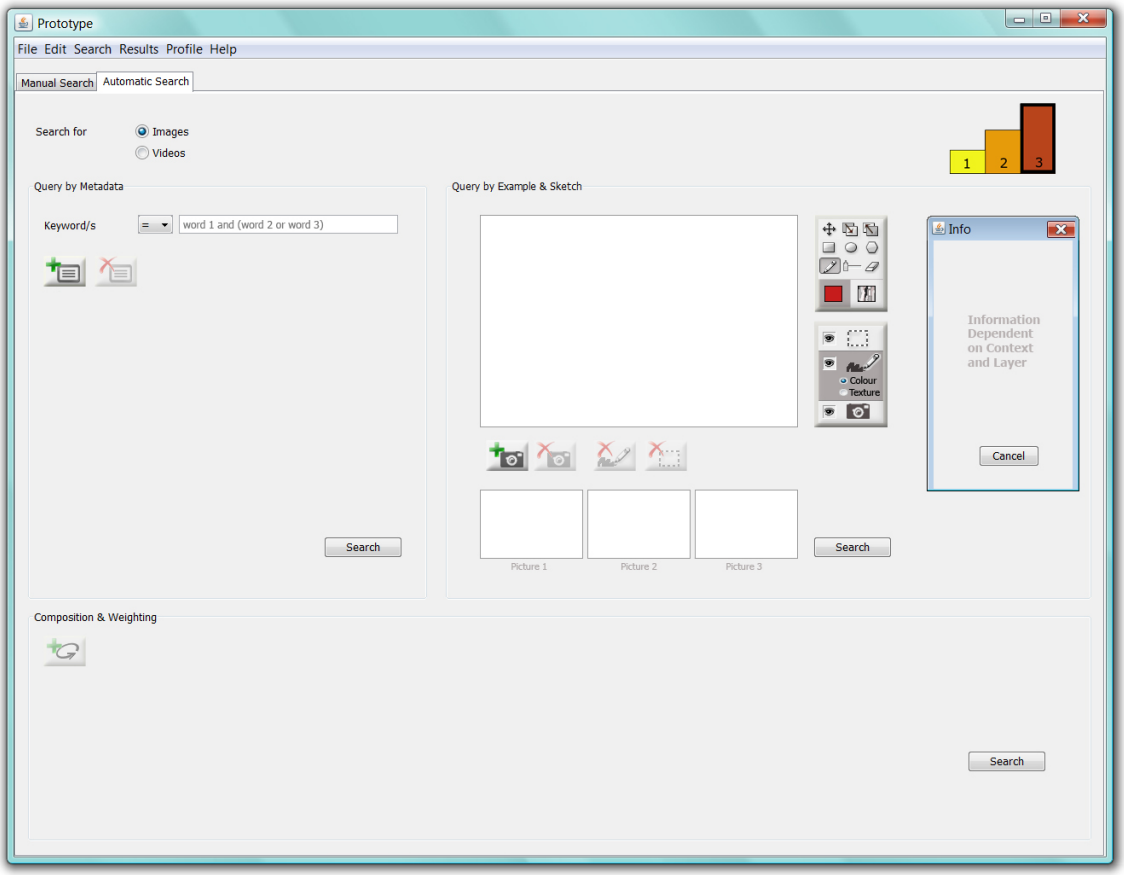


Abbildung 5.15: Tab der anfragebasierten Suche im initialen Zustand auf Ebene 3 der Multi-Layer-Architektur

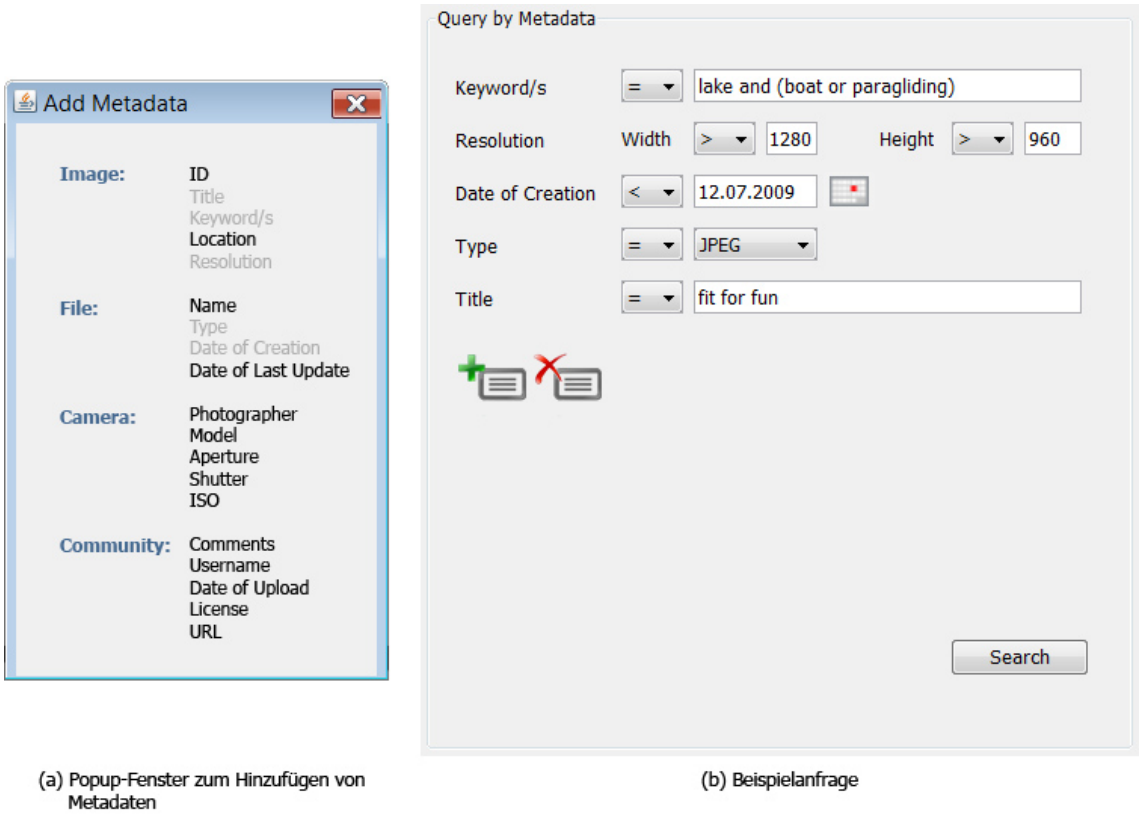
Suche per Metadaten

Im Bereich *Query by Metadata* kann der Nutzer klassische Datenbankbedingungen aufstellen. Im Anfangszustand ist eine Eingabe von Schlüsselwörtern möglich, die im Beschreibungstext von Bildern auftauchen sollen. Diese können mit den Booleschen Operatoren

NOT, AND, OR verknüpft werden. Falls mehrere Begriffe ohne die genannten Operatoren aneinandergereiht werden, so wird intern mit einer gewichteten OR-Verknüpfung gearbeitet, wobei die initialen Gewichte z.B. alle auf 1 gesetzt sind. Um die für weitere exakte Bedingungen benötigten Metadaten auszuwählen, muss der Nutzer auf den mit einem Plus gekennzeichneten Button klicken. Im erscheinenden Popup-Fenster werden alle zur Verfügung stehenden Metadaten in Kategorien aufgelistet. Die Abbildung 5.16 zeigt ein solches Fenster. Ein Klick auf einen Listeneintrag genügt, das Fenster schließt sich automatisch und der Text wird links im *Query by Metadata*-Bereich angezeigt. Bereits ausgewählte Metadaten sind im Popup-Fenster grau unterlegt und können kein zweites Mal angeklickt werden. Der Vorteil der Verwendung eines Popup-Fensters zum Auswählen von Daten gegenüber der Verwendung von Comboboxen besteht in der übersichtlichen Aufteilung der Metadaten nach Kategorien. Desweiteren wäre aufgrund der langen Auswahlliste bei einer Combobox nur ein Ausschnitt sichtbar und somit Scrollen nötig.

Beispiel 3 *Es wird nach JPEG-Bildern mit einer höheren Auflösung als 1280 x 960 Pixeln gesucht, deren Abgebildetes durch den Begriff See und mindestens eines der Worte Boot und Paragliding beschrieben wird. Der Titel soll „fit for fun“ lauten. Das Aufnahmedatum muss zeitlich vor dem 12. Juli 2009 liegen. Die Anfrage ist in Abbildung 5.16 auf der rechten Seite zu sehen.*

Rechts neben den hinzugefügten Metadaten befindet sich jeweils eine Combobox, welche die Vergleichsoperatoren $=, \neq, <, >, \leq$ und $\geq$ enthält. Nicht bei jeder Bedingung sind alle Operatoren zulässig, so stehen z.B. bei *Title* nur $=$ und $\neq$ zur Verfügung, wobei intern mit den Konzepten des Information-Retrieval gearbeitet wird. Ähnliche Begriffe werden somit auch beachtet. Im Abschnitt zu High-Level-Features unter 4.2.2.2 wurde diese Thematik bereits angesprochen. Der Wert für die Bedingung wird im rechten Feld eingegeben bzw. aus einer Combobox ausgewählt, wenn der Wertebereich abzählbar klein ist. Die zweite Variante tritt z.B. bei der Auswahl des Dateityps eines Bildes auf. Ein Datum wird beispielsweise in einem Kalender angeklickt, somit wird die Angabe von nicht existierenden Tagen und falschen Datumsformaten vermieden. Das viereckige Symbol neben der Datumsangabe von *Date of Creation* ist der Button, um den Kalender zu öffnen. Allgemein werden Syntaxfehler bereits bei der Eingabe unterbunden. Im Feld *Resolution* werden so nur Zahlenwerte aus einem festgelegten Intervall akzeptiert. Soll eine Bedingung entfernt werden, muss auf den Button mit dem roten Kreuz geklickt werden. In einem Popup-Fenster stehen alle Einträge, die auch beim Hinzufügen vorzufinden sind. Hier sind jedoch nur die Einträge auswählbar, die bereits als Bedingung vorhanden sind.



(a) Popup-Fenster zum Hinzufügen von Metadaten

(b) Beispielanfrage

Abbildung 5.16: Bereich *Query by Metadata* - Beispiel 3

Um eine automatische Suche nur anhand der Metadaten zu starten, muss auf den Search-Button geklickt werden. Intern werden alle Bedingungen in der auftretenden Reihenfolge mit OR verknüpft. Wie in 4.2.2.3 beschrieben können alle Bedingungen entweder mit 1 gleich gewichtet oder die Gewichte über eine Zufallsfunktion bestimmt oder Nutzerprofile zu Rate gezogen werden.

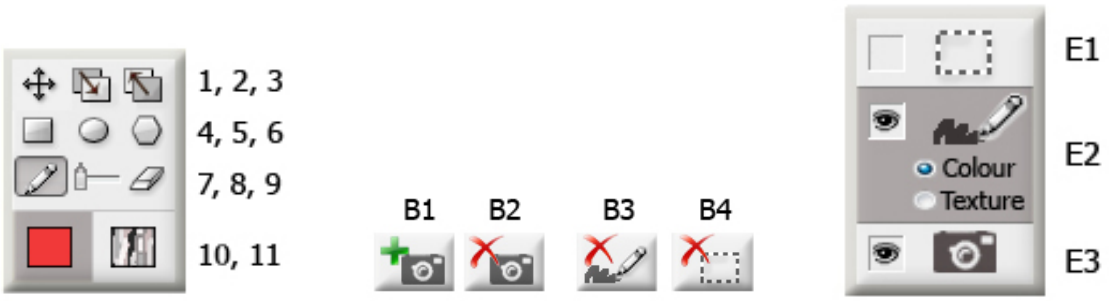
Suche per Anfragebild und Skizze

Der Nutzer kann seine konkreten Vorstellungen bezüglich Farbe, Form und Textur des zu suchenden Fotos in die Anfrage einbeziehen. Diese Low-Level-Features werden im rechten oberen Bereich des Query-Tabs spezifiziert. Das besondere hierbei ist die mögliche Kombination von Query by Example und Query by Sketch. Ein Anfragebild kann somit ein Foto, eine Skizze oder deren kombinierte Variante sein. Es können maximal drei Bilder für die Suche angegeben werden. Diese werden in verkleinerter Ansicht in Reihe dargestellt, unterhalb der großen Zeichenfläche des aktuell zu bearbeitenden Bildes. Soll ein anderes Bild bearbeitet werden, so muss auf die entsprechende Vorschau geklickt werden.

Ein orangefarbener Rahmen kennzeichnet, welches Bild momentan in der Zeichenfläche angezeigt wird. Unterhalb der sogenannten Leinwand befinden sich Buttons, welche die Abbildung 5.18(b) zeigt. Mit Button B1 ist das Laden eines Fotos aus einem Verzeichnis in die Zeichenfläche möglich. Ist beispielweise *Picture 2* ausgewählt, so wird das Foto dort hinzugefügt. Button B2 entfernt das Foto von der Leinwand. Bei Unzufriedenheit mit dem Gemalten kann die gesamte Skizze mit dem Button B3 gelöscht werden. Mit Button B4 werden alle gesetzten Segmente entfernt.

Verwaltung von Ebenen Um eine getrennte Behandlung von Foto und Skizze in einer kombinierten Anfrage zu ermöglichen, werden diese als Ebenen ähnlich wie bei Adobe Photoshop verwaltet. Abbildung 5.18(c) zeigt die drei vorhandenen Ebenen. Das Foto, erkennbar an dem Kamerasymbol, ist die Ebene E3, es ist standardmäßig die unterste. Eine Ebene höher befindet sich die Skizze. Diese hat zwei mögliche Ansichten: Farbe und Textur. Nur eine von beiden ist sichtbar, wobei mithilfe der Radiobuttons *Colour* und *Texture* die Ansicht gewechselt werden kann. Höhere Ebenen verdecken tiefere Ebenen, Zeichnungen sind somit immer sichtbar und verändern das Foto nicht. Ein Wechsel der Ebenenreihenfolge von Skizze und Foto ist auch möglich, indem per Drag & Drop die zweite Ebene nach unten oder die dritte Ebene um eins höher verschoben wird. Das ist jedoch nur sinnvoll, wenn das Foto nicht die gesamte Fläche in Anspruch nehmen soll. Auf der höchsten Ebene E1 sind die Segmente anzufinden, weil sich diese sowohl auf die Skizze als auch auf das Foto beziehen. Die Angabe von Segmenten ist sinnvoll, damit beispielsweise ein dominanter Hintergund eines Bildes nicht die Farb- und Textureigenschaften maßgeblich bestimmt. Eine automatische Segmentierung wird zwar vom System durchgeführt, eine manuelle Kennzeichnung der Segmente verspricht jedoch eine höhere Qualität [Hen08]. Ein Beispiel für die manuelle Unterteilung eines Bildes in Segmente ist in Abbildung 5.17 zu finden.

Abbildung 5.17: manuelle Unterteilung eines Fotos in 3 Segmente



(a) Toolbar (b) Buttons für die Ebenen (c) Ebenenverwaltung

Abbildung 5.18: Interaktionselemente im Bereich *Query by Example & Sketch*

Welche Ebene aktuell ausgewählt ist, ist an der grauen Untermalung zu erkennen. Veränderungen betreffen nur diese Ebene, andere Ebenen bleiben unberührt. Standardmäßig ist die Skizzen-Ebene aktiviert, wie es die Abbildung 5.18(c) zeigt. Ein Klick in den Bereich einer anderen Ebene aktiviert diese. Ebenen können auch ausgeblendet werden, wie es in Abbildung 5.18(c) für die Ebene E1 der Fall ist. Hierzu muss auf das Augen-Symbol geklickt werden, welches anschließend ausgeblendet wird. Ein erneuter Klick blendet die Ebene wieder ein. Das ist sinnvoll um beispielsweise etwas zu zeichnen, ohne von den angegebenen Segmenten oder den möglicherweise unruhigen Strukturen eines Fotos abgelenkt zu werden.

Werkzeugleiste Das Foto, die Skizze oder die Segmente werden in der Zeichenfläche bearbeitet. Hierfür stehen Werkzeuge in der Toolbar bereit, welche in Abbildung 5.18(a) zu sehen sind. Im Folgenden werden die Funktionen der einzelnen Werkzeuge vorgestellt:

- **Werkzeuge 1, 2 und 3 zum Verändern der Position, Größe und Reihenfolge:** Mit dem Werkzeug 1 kann etwas Markiertes per Drag & Drop verschoben werden. Die Maus verändert sich währendessen zu einem Kreuz, das dem Symbol 1 der Toolbar entspricht. Für eine Markierung muss nur in den entsprechenden Bereich geklickt werden. Im Gegensatz zu einem Foto, das vollständig markiert wird, wird in einer Skizze nur der zusammenhängende Bereich mit gleicher Eigenschaft wie Farbe oder Textur markiert. Es können also innerhalb der Skizze mehrere separat zu betrachtende Figuren existieren. Bei einer Markierung erscheinen dann die vier Eckpunkte eines Rechtecks, welches die maximale horizontale und vertikale Ausdeh-

nung der markierten Fläche anzeigt. An den Eckpunkten kann eine Veränderung der Größe bei Beibehaltung des Seitenverhältnisses vorgenommen werden. Der Mauszeiger verwandelt sich zu einem Pfeil mit zwei entgegengesetzten Spitzen. Soll nur die Höhe oder nur die Breite angepasst werden, so muss der Mauszeiger nicht an den Eckpunkten, sondern an den Kanten angesetzt werden. Die Aufhebung der Markierung erfolgt durch erneutes Klicken in den markierten Bereich. Das Werkzeug 2 bringt etwas Markiertes innerhalb der gleichen Ebene in den Vordergrund. Wird z.B. ein rotes Rechteck gezeichnet und anschließend ein blauer Kreis, der das Rechteck teilweise bedeckt, so kann das Viereck im Nachhinein vor dem Kreis platziert werden. Die umgekehrte Funktion hat Werkzeug 3. Dort wird etwas in den Hintergrund innerhalb der gleichen Ebene verschoben.

- **Werkzeuge 4, 5 und 6 zum Zeichnen von geometrischen Figuren:** Werkzeug 4 ermöglicht das Zeichnen eines Vierecks durch Aufziehen der Fläche von der linken oberen zur rechten unteren Ecke. Bei einer Skizze wird dieses automatisch mit der eingestellten Farbe oder Textur gefüllt. Ist die Segmentebene aktiviert, so werden die Kanten des aufgezogenen viereckigen Segments schwarz-weiß gestrichelt dargestellt. Mit dem Werkzeug 5 können Ellipsen gezeichnet werden auf die gleiche Art wie mit dem Werkzeug 4. Polygone lassen sich mit Werkzeug 6 formen. Es wird mit jedem Klick eine Ecke erzeugt. Der Kantenzug wird entweder durch Doppelklick oder durch Klick auf den Anfangspunkt geschlossen.

- **Werkzeuge 7, 8 und 9 zum freien Zeichnen und Löschen:** Mit dem Stift-Werkzeug 7 kann der Nutzer beliebige Linien und Formen zeichnen. Die Stiftbreite ist durch Verschieben des Reglers 8 anpassbar. In einer Skizze wird mit der eingestellten Farbe oder Textur gemalt. Ist stattdessen die Segment-Ebene aktiviert, so wird die Kante des Segments gezeichnet. Falls der Kantenzug nicht vom Nutzer geschlossen wird, so wird automatisch eine gerade Linie vom Anfangs- bis zum Endpunkt gezogen. Werkzeug 9 stellt einen Radierer dar. Mit diesem können aus einer Ebene Bereiche gelöscht werden. Die Breite wird auch hier mit dem Regler 8 vorgegeben. In einer Skizze werden so Teile des Gemalten entfernt. Aus einem Foto können irrelevante Regionen beseitigt werden. In der Segment-Ebene wird beim teilweisen Löschen der Kante das gesamte Segment gelöscht, weil die Information eines nicht geschlossenen Kantenzugs nicht aussagekräftig ist. Eine andere Variante, um in einer Skizze eine Figur zu entfernen, ist auch das Markieren und das anschließende Drücken der DEL- oder ENTF-Taste.

- **Werkzeuge 10 und 11 zum Festlegen von Farbe und Textur:** Das Feld des Buttons 10 zeigt die momentan eingestellte Farbe. Mit einem Klick auf den Button öffnet sich ein Popup-Fenster. Dieses ermöglicht es eine Farbe auszuwählen, wie es auch in Adobe Photoshop gehandhabt wird. Es kann direkt die Farbe im HSV-Farbraum angeklickt oder die konkreten HSV- oder RGB-Werte angegeben werden. Bei einem Klick auf den Button 11 erscheint auch ein Popup-Fenster, welches eine Liste von verfügbaren Texturen enthält. Mit einem Klick auf eine Textur wird diese die aktuelle.

  Nicht nur die Radiobuttons in der Ebene E2 aus Abbildung 5.18(c) zeigen an, ob man sich in der Farb- oder Texturansicht befindet, sondern auch die graue Unterlegung eines der Werkzeuge 10 oder 11. Bei Veränderung der Farbe oder der Textur wird somit automatisch in die entsprechende Ansicht gewechselt. Die Radiobuttons passen sich der Auswahl an. Wird von der Farbansicht in die der Textur gewechselt, so werden bei Figuren ohne zugewiesene Textur nur die Kanten angezeigt. Das Gleiche gilt für die andere Richtung. Abbildung 5.19 zeigt ein Viereck und einen Kreis, wobei beide eine Farbe, jedoch nur das Rechteck eine Textur besitzt.

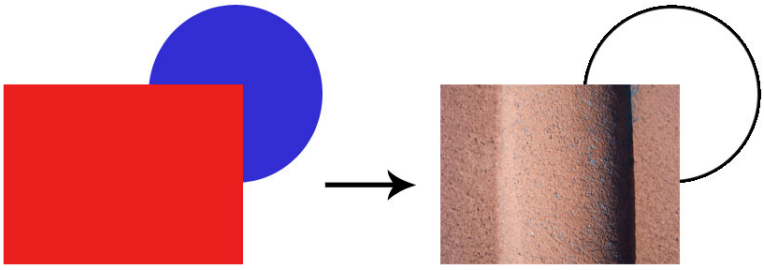


Abbildung 5.19: Wechsel von der Farbansicht in die Texturansicht

Beim Skizzieren stehen alle Werkzeuge bereit. Für die Bearbeitung der Foto-Ebene können nur die Werkzeuge 1, 8 und 9 verwendet werden, weil mit allen anderen nicht das Foto selbst verändert werden kann. Um dieses dem Nutzer kenntlich zu machen, werden nicht verfügbare Werkzeuge hellgrau dargestellt. Die für die Suche irrelevanten Bereiche in einem Foto können mit dem Radierer aus dem Bild entfernt werden. Desweiteren kann eine Skalierung und Positionierung vorgenommen werden. Hat der Nutzer die Ebene der Segmente aktiviert, so funktionieren alle Werkzeuge bis auf 10 und 11, da die Farbe und die Textur für die Skizze, nicht aber für Segmente, wichtig sind.

Beispiel 4 *Es wird nach einem Bild gesucht, dass in einem See ein Boot oder über einem See einen Paraglider zeigt. In Abbildung 5.20 ist diese zum Beispiel 3 passende ähnlichkeitsbasierte Anfrage zu sehen. Hier besteht das erste Anfragebild aus einem Foto und einer Skizze von einem Paraglider. Das zweite Bild ist ein Foto von einem Segelboot auf einem See und das dritte zeigt ein gemaltes Motorboot mit einem Sandstrand als Hintergrund. Bei allen drei Bildern sind die Segmente ausgeblendet.*

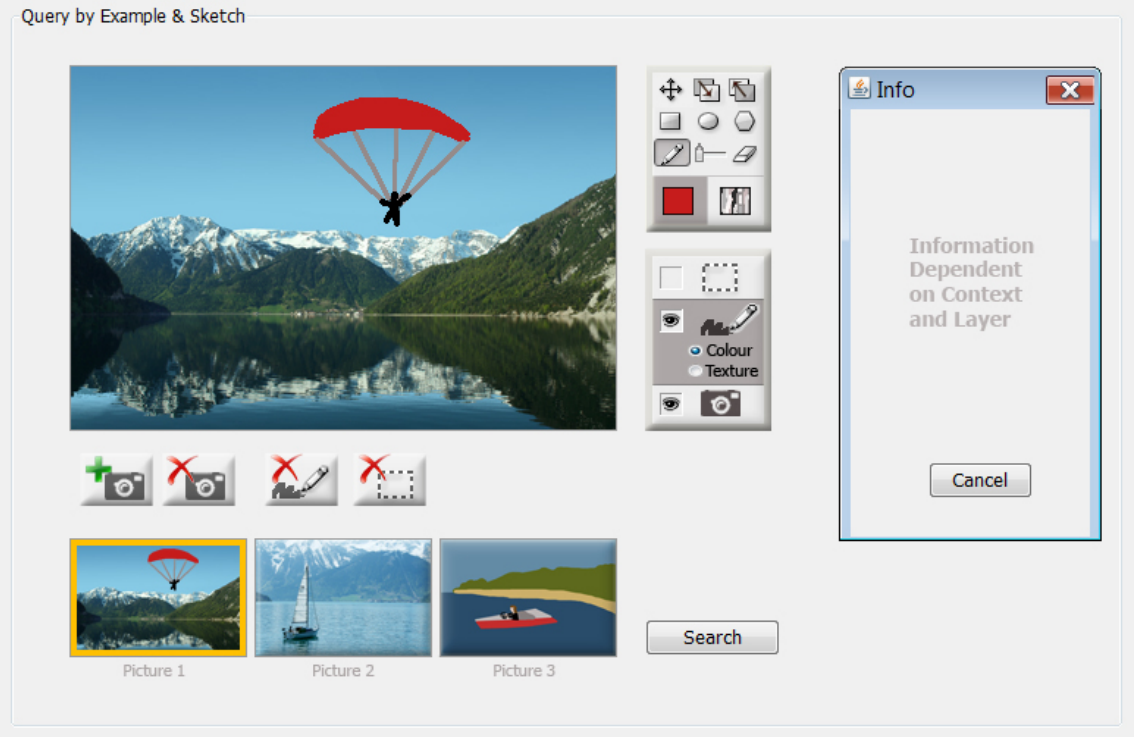


Abbildung 5.20: *Query by Example & Sketch* - Beispiel 4

Möchte der Nutzer nur anhand der Fotos und Skizzen suchen, so genügt ein Klick auf den Search-Button im Bereich *Query by Example & Sketch*. Intern werden alle Bilder mit OR verknüpft, da diese jeweils eine mögliche Variante des gesuchten Fotos darstellen. Ausgeblendete Ebenen werden nicht in die Suche einbezogen, da der Nutzer einen Grund gehabt hat diese aus der Ansicht zu entfernen.

Komposition und Gewichtung der Anfragebedingungen

Fortgeschrittenen Nutzern soll die Möglichkeit gegeben werden selbst die einzelnen Bedingungen miteinander durch Boolesche Operatoren zu verknüpfen und zu gewichten. Hierfür steht der im unteren Fenster platzierte Bereich *Composition & Weighting* bereit. Dieser erstreckt sich horizontal über die gesamte Fensterbreite. Dadurch soll ersichtlich werden,

dass sich die Angaben auf die beiden darüberliegenden Bereiche beziehen. Jede Zeile in *Query by Metadata* und jedes der Bilder in *Query by Example & Sketch* stellt eine Bedingung dar. Für die Komposition der Bedingungen werden im Folgenden mehrere Varianten vorgestellt:

- Venn-Diagramm
- Graph
- Textfelder

Venn-Diagramm

Venn-Diagramme sind aus der Mathematik bekannt, wo sie in der Mengenlehre Mengenoperationen visualisieren. In dieser Arbeit sollen sie der grafischen Visualisierung von Booleschen Ausdrücken dienen. Für jede Bedingung B_i wird ein Kreis K_i benötigt. Die Kreise werden innerhalb einer Grundmenge G so zueinander angeordnet, dass sich alle schneiden. Die Schnittmenge von zwei Kreisen K_i und K_j entspricht der Konjunktion der Bedingungen B_i und B_j. Die Disjunktion von B_i und B_j ist die Vereinigungsmenge von K_i und K_j. Die Negation von B_i entspricht der Fläche von G, von der die Kreisfläche von K_i abgezogen wird. Die Venn-Diagramme können automatisch vom System anhand aller angegebenen Bedingungen erstellt werden. Durch einen Klick in einen Bereich der Kreise wird die Fläche, die durch die schwarzen Linien eingegrenzt wird, farblich hervorgehoben. Abbildung 5.21 zeigt Anfragen mit 3, 4 bzw. 6 Bedingungen. Die linke Anfrage entspricht $(B_1 \wedge B_2) \vee (B_1 \wedge B_3)$, wobei in diesem Fall die Klammern weggelassen werden dürfen, weil AND stärker bindet als OR. Die mittlere Anfrage heißt $(B_1 \wedge B_2) \vee (B_3 \wedge \neg B_4)$. Die Anfrage mit 6 Bedingungen bedeutet $B_1 \vee (B_2 \wedge B_3 \wedge \neg B_4) \vee (B_5 \wedge B_6)$.

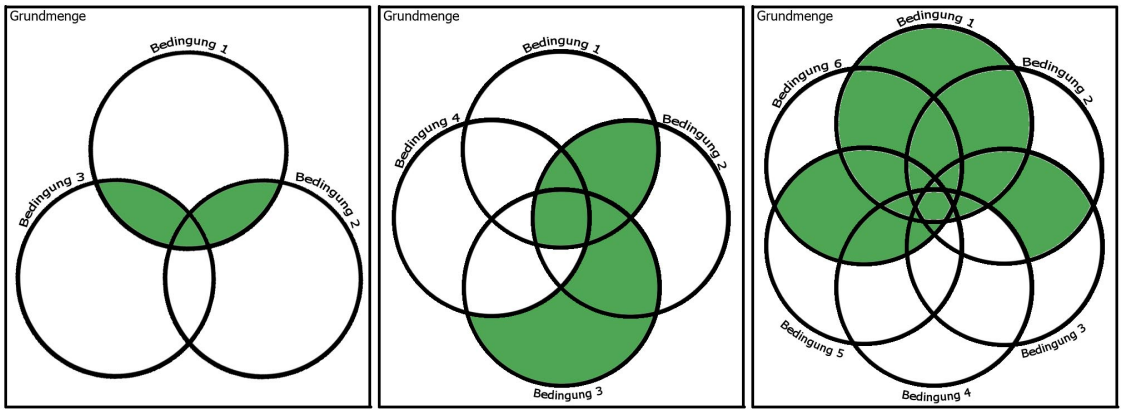


Abbildung 5.21: Venn-Diagramme für 3, 4 bzw. 6 Bedingungen

Bewertung der Venn-Diagramme Venn-Diagramme sind eine einfache Variante um Bedingungen zu verknüpfen. Im Alltag werden die Begriffe UND und ODER teilweise anders verwendet als in der Booleschen Algebra, wodurch die Formulierung einer Anfrage zu Problemen führen kann. Mit Venn-Diagrammen wird dieses Problem behoben, weil der Nutzer nicht mit den Begriffen, sondern mit der Semantik dieser arbeitet. Er markiert nur die Bereiche, die für seine Anfrage wichtig sind. Eine Verwechslung kann somit nicht auftreten. Jedoch haben Venn-Diagramme auch Nachteile. Für Anfragen mit mehr als 4 Bedingungen werden die Diagramme unübersichtlich und das Aufstellen einer Anfrage somit schwierig. Das Beispiel mit 6 Bedingungen verdeutlicht diesen Aspekt. Außerdem entstehen durch viele Bedingungen viele kleine Flächen, wodurch der Nutzer bei einfachen Anfragen viele Bereiche markieren muss. Zum einen ist das dem Anwender zu aufwendig und zum anderen erfordern die kleinen Abmessungen präzise Maussteuerung und verlangsamen somit die Interaktion. Die GUI soll auch für mehr als 4 Bedingungen einfach zu bedienen sein. Aus den genannten Nachteilen werden deshalb Venn-Diagramme nicht für die Komposition der Bedingungen verwendet.

Graph

Eine Anfrage kann auch als ungerichteter Graph dargestellt werden. Ein Knoten ist eine Bedingung. Eine Kante kann die Disjunktion oder Konjunktion repräsentieren, weil diese binäre Operatoren zwischen zwei Bedingungen sind. Eine unäre Negation bezieht sich nur auf eine Bedingung, ist also eine Eigenschaft eines Knotens. Um eine Klammerung von Ausdrücken zu ermöglichen, sollen höherpriore Kanten auffälliger gekennzeichnet werden als niederpriore. Eine Möglichkeit ist der Einsatz von gesättigteren Farben und einer größeren Strichbreite für stärker bindende Kanten.

Die Knoten werden vom Nutzer erzeugt, indem dieser auf eine freie Fläche klickt. Es öffnet sich anschließend ein Popup-Fenster, welches alle Bedingungen auflistet. Die Negierung einer Bedingung ist durch Setzen eines Häkchens in der Combobox hinter dem entsprechenden Eintrag möglich. Abbildung 5.22 zeigt für die Anfrage, die sich aus den Beispielen 3 und 4 zusammensetzt, das Fenster zum Hinzufügen eines Knotens. Ein existierender Knoten kann im Nachhinein verändert werden, indem auf diesen doppelt geklickt wird, sodass sich das Fenster erneut öffnet. Ein Knoten kann mit rechter Maustaste per Drag & Drop verschoben werden. Mit einem rechten Mausklick auf den Knoten kann dieser gelöscht werden, indem auf die erscheinende Zeile *Delete Node* geklickt wird. Das Hinzufügen aller Knoten kann bei vielen Bedingungen einfacher realisiert werden, indem automatisch alle Knoten hinzugefügt werden. Die Knoten werden hierbei ellipsenförmig angeordnet. Das Hinzufügen eines Knotens in den Bereich *Composition & Weighting* nach jedem Erstellen

einer neuen Bedingung in *Query by Metadata* bzw. *Query by Example & Sketch* würde den Nutzer ablenken. Die Anordnung der Knoten würde sich ständig verändern. Aufgrund der Bewegungen würde die Aufmerksamkeit des Anwenders auf den unteren Anwendungsbereich gelenkt werden. Das ist jedoch arbeitsunproduktiv. Aus diesem Grund wird ein Button bereitgestellt, der alle Knoten in einem einzigen Schritt hinzufügt, wenn der Nutzer dieses wünscht. Der Button hierfür ist in der Abbildung 5.22 unten rechts zu sehen. Sowohl die manuelle als auch die automatische Variante der Erzeugung von Knoten soll angeboten werden.

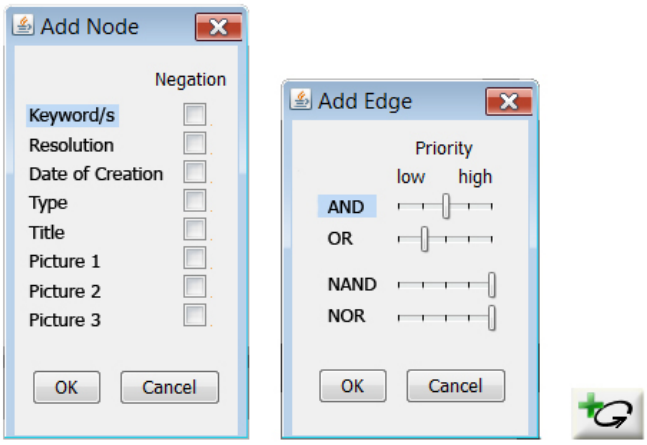
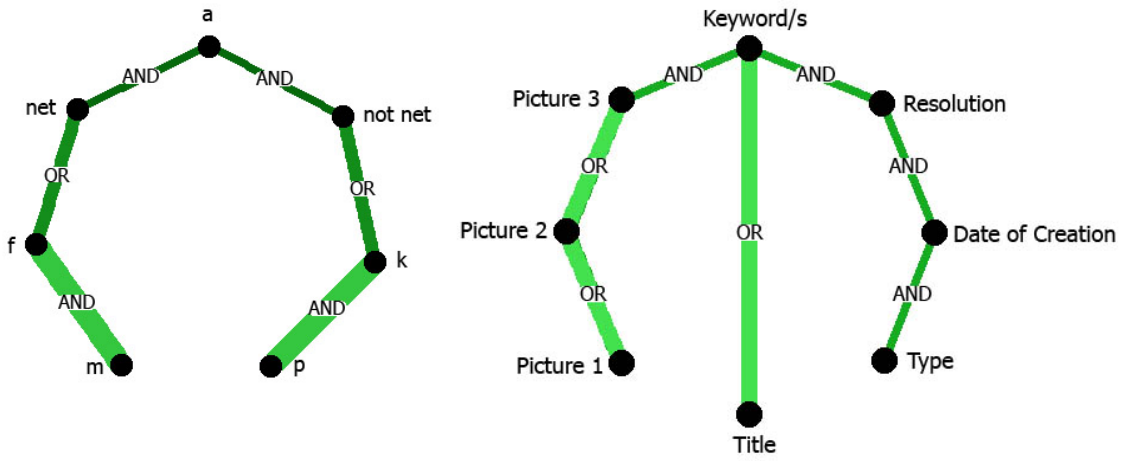


Abbildung 5.22: Popup-Fenster zum Hinzufügen eines Knotens bzw. einer Kante zu einem Graph

Eine Kante wird gezeichnet, indem auf einen Knoten die linke Maustaste geklickt und gehalten und über einen anderen Knoten losgelassen wird. Es öffnet sich auch hier ein Popup-Fenster, welches in Abbildung 5.22 zu sehen ist. Es kann eine Auswahl zwischen AND, OR, NAND und NOR getroffen werden. NAND und NOR sind zusammengesetzte Operatoren: a NAND b $= \neg(a \wedge b)$, a NOR b $= \neg(a \vee b)$. NAND und NOR stehen zusätzlich in der Liste, weil die Idiome für das nachträgliche Negieren einer Konjunktion bzw. Disjunktion mehrere Klicks benötigen würden. Um eine Priorisierung der Operatoren anzubieten, die überlicherweise in einem Booleschen Textausdruck mit einer Klammerung vorgenommen wird, stehen Schieberegler bereit. Diese Regler zeigen initial den Rang der Prioritäten ohne Klammerung an. NAND und NOR binden am stärksten. An zweithöchster Stelle befindet sich AND und an letzter OR. Soll eine Disjunktion vor einer Konjunktion ausgewertet werden, so muss für OR der Prioritätswert höher eingestellt werden als der Standardwert von AND. Die Kante erscheint dann dicker und farblich gesättigter. Die



(a) Beispiel 2 (b) Beispiel 5

Abbildung 5.23: Graphendarstellung von Anfragen

Abbildung 5.23(a) zeigt die Netbook-Anfrage des Beispiels 2 als Graph: a $\wedge$ ($\neg$ net $\vee$ (k $\wedge$ p)) $\wedge$ (net $\vee$ (f $\wedge$ m)).

Beispiel 5 *Eine mögliche Komposition der Beispielanfragen 3 und 4 ist (Picture 1* $\vee$ *Picture 2* $\vee$ *Picture 3)* $\wedge$ *(Keyword/s* $\vee$ *Title)* $\wedge$ *Resolution* $\wedge$ *Date of Creation* $\wedge$ *Type. Der zugehörige Graph wird in Abbildung 5.23(b) gezeigt.*

Gewichtung in einem Graphen Um in einem Graphen das Gewicht eines Terms relativ zu einem anderen anzugeben, werden jeweils 2 Regler auf einer Kante positioniert. Beiden steht jeweils eine Kantenhälfte zur Verfügung. Es wird mit dem Wertebereich [0,1] gearbeitet. Je näher ein Regler per Drag & Drop zum jeweiligen Knoten verschoben wird, desto größer wird das Gewicht. Befindet er sich also direkt am Knoten, wird ihm ein Gewicht von 1 zugewiesen. Befindet er sich in der Kantenmitte, so hat er das Gewicht 0. Neben der Position soll zusätzlich auch die Farbe des Reglers Aufschluss über das Gewicht geben, um eine schnellere Einordnung in den Wertebereich zu ermöglichen. Eine Skala neben dem Graphen zeigt die Einteilung. Initial sind die Gewichte ausgeblendet, um den Graphen bei der Erstellung einfach zu halten. Erst durch das Setzen des Häkchens vor *Display Weights* werden alle Regler gesetzt und die umrahmte Anzeige mit der Skala eingeblendet. Zur Verdeutlichung dient die Abbildung 5.24, welche das Beispiel 5 aufgreift. Für eine präzi-

se Gewichtssetzung sind zusätzlich auch die numerischen Werte anzeigbar, wenn *Display Exact Values* ausgewählt ist. Dieses ist in Abbildung 5.25 zu sehen.

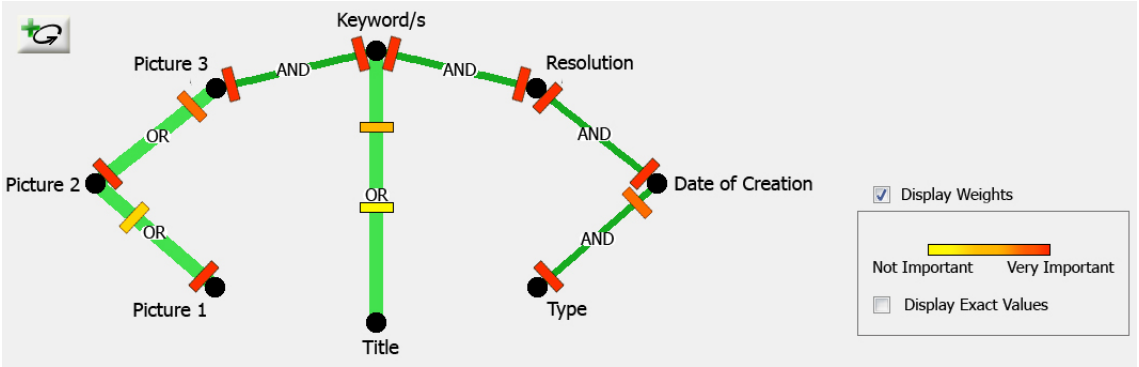


Abbildung 5.24: Beispiel 5 als Graph mit Gewichten

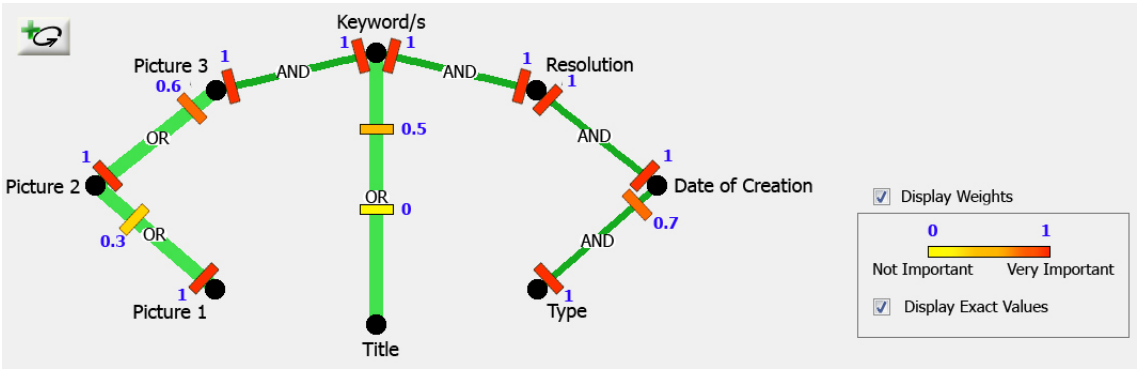


Abbildung 5.25: Beispiel 5 als Graph mit Gewichten und zusätzlicher Anzeige der numerischen Gewichtswerte

Probleme mit der Graphendarstellung Probleme beim internen Auswerten eines Graphen können entstehen, wenn gleiche Prioritäten von Kanten vergeben werden, die einen gemeinsamen Knoten besitzen. Abbildung 5.26 zeigt zwei solcher Fälle. Im rechten Graphen befinden sich zwei Kanten gleicher niederer Priorität, die den Knoten b gemein haben. Es gilt hier das Assoziativgesetz $((b \vee c) \vee a) \vee d = ((b \vee c) \vee d) \vee a$, welches mit der Definition 1 für die algebraische Summe bzw. das algebraische Produkt gezeigt werden kann. Auf der linken Seite der Abbildung existieren jedoch zwei Kanten gleicher Priorität, jedoch mit unterschiedlichen Operatoren. Hier muss für die Auswertung trotz gleicher manuell angegebener Prioritäten eine Kante höher eingestuft werden. Der Konjunktion wird in solchen Fällen der Vorrang vor der Disjunktion gewährt. Damit wird zwar das Problem der

Auswertung gelöst, jedoch entspricht dann die visuelle Darstellung der Kantenprioritäten nicht der internen Verarbeitung des Graphen.

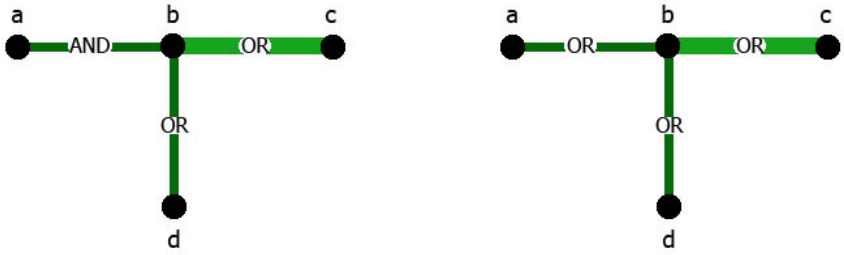


Abbildung 5.26: gleiche Prioritäten von Kanten bei Graphen

Laut Definition 2 gilt bei einer gewichteten Anfrage das Assoziativgesetz nicht mehr: $((b \vee_{\theta_b,\theta_c} c) \vee_{\theta_{bc},\theta_a} a) \vee_{\theta_{bca},\theta_d} d \neq ((b \vee_{\theta_b,\theta_c} c) \vee_{\theta_{bc},\theta_d} d) \vee_{\theta_{bcd},\theta_a} a$. Deshalb tritt in beiden Fällen das Problem auf, dass das System wegen nicht eindeutiger Prioritätenvergabe ein Gewicht einem anderen Term zuordnen könnte, als der Nutzer eigentlich vorgesehen hatte.

Ein zweiter problematischer Punkt bei Verwendung eines Graphen sind Zyklen. Diese dürfen nicht auftreten, weil wiederum die Auswertung uneindeutig wird. Aus diesem Grund muss der Anwender darauf hingewiesen werden eine Kante zu entfernen. Damit dieser den Sachverhalt versteht, soll eine Warnung direkt nach dem Auftreten des Zyklus wie in Abbildung 5.27 erscheinen. Initial ist eine Kante bereits markiert.

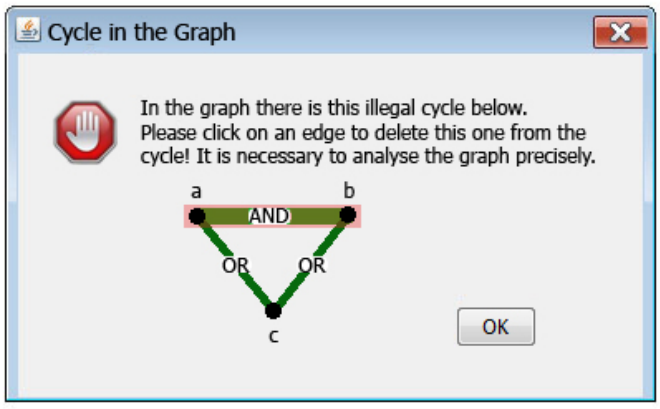


Abbildung 5.27: Warnung bei Auftreten eines Zyklus im Graph

Desweiteren kann es vorkommen, dass ein Nutzer zu wenige Kanten angibt. Der Graph ist dann nicht zusammenhängend. Eine Lösung wäre eine implizite Disjunktion der zusammenhängenden Teilgraphen. Außerdem kann der Anwender auch einige Bedingungen nicht angegeben haben. Diese werden dann nicht in die Suche einbezogen.

Ein weiterer Aspekt ist die übersichtliche Anordnung des Graphen. Kantenschnitte sollen minimiert werden, damit die Kantenbeschriftung erkennbar ist. Desweiteren sollen Kanten Geraden sein, da diese in ihrer Richtung einfacher und schneller zu verfolgen sind als gebogene Linien. Zum einen kann der Nutzer die Kanten wie bereits beschrieben selbst umordnen. Zum anderen muss abgewägt werden, ob dieses auch automatisch angeboten werden soll. Eine mögliche Variante für den letzten Fall wäre die Verwendung von *Force Directed Placement* (FDP). Der von Fruchterman und Reingold entwickelte Algorithmus ist eine verbesserte Adaption des Spring-embedder Models von Eades [Che04]. Um minimale Kantenschnitte, eine gleichmäßige Verteilung der Kanten im zweidimensionalen Raum, einheitliche Kantenlängen und Symmetrie zu erreichen, wird sich der Mechanik und der Teilchenphysik bedient. Knoten entsprechen atomaren Partikeln und Kanten werden als Federn angesehen. Anziehungskräfte f_a bestehen zwischen verbundenen Knoten und abstoßende Kräfte f_d zwischen allen Knotenpaaren. Verbundene Knoten sollen nah beieinander liegen, jedoch nicht zu nah. Knoten mit einer Distanz d ziehen sich mit der Kraft $f_a(d) = \frac{d^2}{k}$ an und stoßen sich mit der Kraft $f_r(d) = -\frac{k^2}{d}$ ab. K entspricht der optimalen Distanz zwischen zwei Knoten, die von der vorhandenen Fläche und der Gesamtanzahl der Knoten abhängig ist. Die Kräfte werden iterativ für jeden Knoten berechnet, woraus sich nach jedem Iterationsschritt eine Veränderung der Knotenposition ergibt. Ziel ist es ein statisches Gleichgewicht zu erreichen, bei dem sich für jeden Knoten beide Kräfte gegenseitig aufheben. In der Praxis wird der Algorithmus, der in [FR91] nachgelesen werden kann, nach einer geschätzten Anzahl an Durchläufen gestoppt. Nutzertests müssen durchgeführt werden, die bei einer durchschnittlichen Anzahl an aufgestellten Bedingungen die Lesbarkeit von erstellten Graphen untersuchen. Mit diesen kann eine Entscheidung getroffen werden, ob es sinnvoll ist eine automatische Umordnung der Knoten mittels FDP anzubieten.

Textfelder

Neben der Darstellung als Graph kann eine Anfrage auch als Kette von verschiebbaren Textfeldern dargestellt werden. Jede Bedingung, jeder Operator und jede Klammer ist ein einzelner Baustein, der eine spezifische Farbe besitzt. Abbildung 5.28 zeigt die Beispielanfrage 5 in dieser Variante mit ausgeblendeten Gewichten. Da die Anfrage eine Aneinanderreihung von Termen ist, wäre es denkbar den Nutzer die Anfrage selbst eintippen zu lassen. Um Rechtschreibfehler sowie nicht auszuwertende Anfragen zu verhindern, werden funktional festgelegte Bausteine verwendet. Diese können unter Einhaltung von Syntaxregeln innerhalb der Anfrage per Drag & Drop verschoben werden. Gültige Positionen werden beim Darüberfahren mit der Maus durch einen kleinen Pfeil gekennzeichnet.

Es steht ein Button bereit, mit dem alle Bedingungen in der Reihenfolge ihrer Erscheinung automatisch auf der Anzeigefläche erzeugt werden. Dieser ist der erste Button von links. Initial steht dann zwischen allen Bedingungen eine Disjunktion. Die Felder können auch manuell hinzugefügt werden. Beim Klick auf den zweiten Button von links erscheint ein Popup-Fenster, wo alle verfügbaren Bedingungen aufgelistet werden. Nach der Auswahl verändert sich der Mauszeiger zum Textbaustein mit der entsprechenden Beschriftung. Der Nutzer reiht diesen per Drag & Drop an eine Stelle innerhalb der Anfrage ein. Mit einem Klick auf das Feld lässt sich die Bedingung im Nachhinein auch verändern. Ein Button für das Hinzufügen einer Disjunktion oder Konjunktion steht nicht bereit. Es wird stattdessen mit dem Einordnen der Bedingung automatisch ein OR-Baustein an der richtigen Stelle hinzugefügt. Das gewährleistet stets eine zusammenhängende Anfrage. In der Combobox ist neben dem OR auch ein AND aufgelistet. Mit Ausnahme dieser beiden Operatoren können alle Felder verschoben werden. Um die Anfrage um eine Negation zu ergänzen, klickt der Nutzer auf den NOT-Button. Das Feld kann nur vor einer öffnenden Klammer oder vor einer Bedingung eingefügt werden. Prioritäten werden über Klammern angegeben. Diese können mit dem ersten Button von rechts erzeugt werden. Um stets ein Paar von Klammern einzufügen, verändert sich die Maus zuerst in eine öffnende Klammer und nach dem Eingliedern dieser in die Anfrage automatisch in eine schließende. Das Setzen von nur einer Klammer wird somit ausgeschlossen sowie das Einreihen einer schließenden Klammer vor der öffnenden. Das Löschen eines Bausteins wird mit einem Klick auf die rechte Maustaste vorgenommen. Im Zuge dessen wird auch der Operator auf gleicher Prioritätsstufe entfernt. Das alleinige Löschen eines Operators ist nicht möglich.

Gewichtung in der Textfelderdarstellung Wie bereits beim Graph werden hier verschiebbare Regler eingesetzt. Diese befinden sich jeweils links und rechts neben dem Feld der Konjunktion bzw. der Disjunktion. Auf Wunsch des Nutzers können diese eingeblendet werden. Die Farbe und die Position des Reglers geben Auskunft über die Wichtigkeit eines Terms, wobei auch hier wieder zusätzlich die numerischen Gewichtswerte angezeigt werden können. Die Abbildung 5.29 stellt die gewichtete Variante des Beispiels 5 mit dem Boot und dem Paraglider dar. Befindet sich der Regler weit oben, so ist die Bedingung sehr wichtig. Das ist intuitiv verständlich, weil weiter oben befindliche Objekte als besser und bedeutender eingestuft werden als tiefergelegene. Ein realitätsnaher Bezug wäre z.B. die Siegerehrung bei sportlichen Wettkämpfen. Der Goldmedaillengewinner steht auf dem Podest erhöht über dem Silber- und Bronzemedaillengewinner.

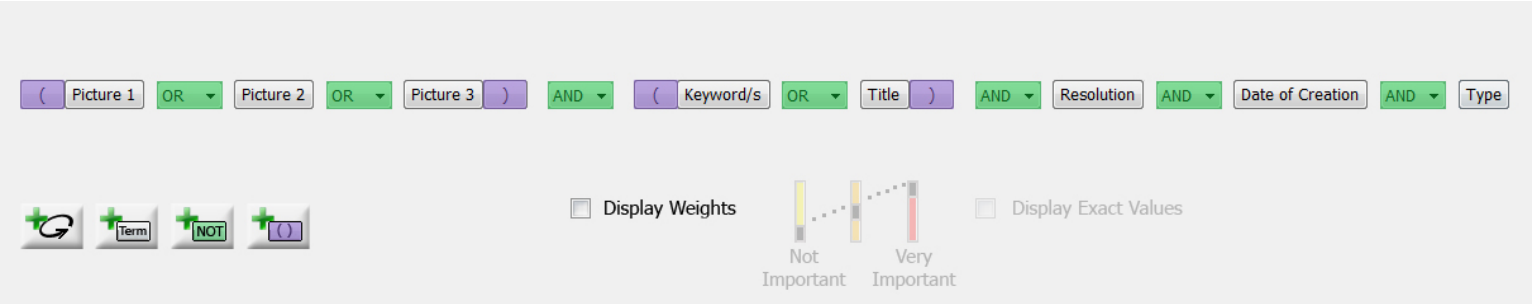


Abbildung 5.28: Beispiel 5 in der Darstellung mit Textfeldern

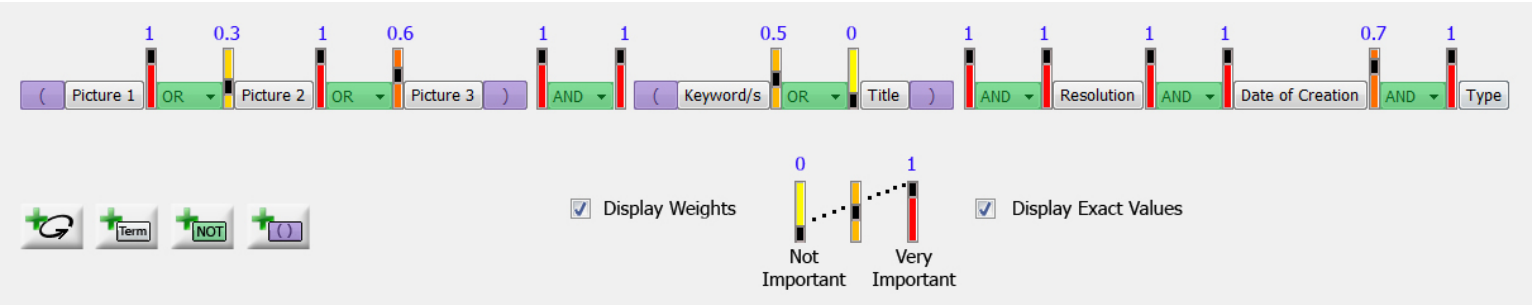


Abbildung 5.29: Beispiel 5 in der Darstellung mit Textfeldern und Gewichten sowie zusätzlicher Anzeige der numerischen Gewichtswerte

Probleme mit der Textfelderdarstellung Wie bei der Graphendarstellung kann auch hier das Problem der nicht eindeutigen Vergabe von Prioritäten auftreten. Zwar wird beim Einfügen einer öffnenden Klammer stets auch die schließende gesetzt, jedoch kann der Nutzer auch ein Paar von Klammern nicht gesetzt haben. Dann kann nicht festgestellt werden, auf welchen Term sich das Operatorgewicht bezieht. Dieser Aspekt wurde bereits ausführlich beschrieben. Intern kann aufgrund der europäischen Leserichtung die Regel festgelegt werden, dass linksstehende Ausdrücke stärker binden als rechtsstehende. Für die Beispielanfrage 5 würde sich somit bei einer Gewichtung Folgendes ergeben: (Picture 1 $\vee_{a,b}$ Picture 2 $\vee_{c,d}$ Picture 3) $\wedge e, f$ (Keyword/s $\vee_{g,h}$ Title) $\wedge_{i,j}$ Resolution $\wedge_{k,l}$ Date of Creation $\wedge_{m,n}$ Type = ((((((Picture 1 $\vee_{a,b}$ Picture 2) $\vee_{c,d}$ Picture 3) $\wedge_{e,f}$ (Keyword/s $\vee_{g,h}$ Title)) $\wedge_{i,j}$ Resolution) $\wedge_{k,l}$ Date of Creation)$\wedge_{m,n}$ Type). Somit bezieht sich beispielsweise das Gewicht c auf den Term (Picture 1 $\vee_{a,b}$ Picture 2) anstatt nur auf Picture 2.

Bei Anfragen die mehr als 8 Bedingungen enthalten, ist das Anzeigen auf einer einzigen Zeile nicht mehr möglich. Die restlichen Bedingungen müssen auf die nächste Zeile verschoben werden. Allgemein kann festgestellt werden, dass bei großen Anfragen, besonders mit zusätzlichem Einblenden von Gewichten, die Übersichtlichkeit stark abnimmt.

Entscheidung für eine Darstellungsvariante

Die Venn-Diagramme sind als Variante für das Aufstellen einer Anfrage bereits zu Beginn ausgeschieden. Treten mehr als 4 Bedingungen auf, so ist das Markieren der vielen kleinen Flächen eine mühsame Arbeit und kostet Zeit. Der entscheidende Faktor ist, dass es bei großen Anfragen sehr viel Überlegung benötigt, um die richtigen Flächen auszuwählen. Die einfache Bedienung soll im Vordergrund stehen. Zur Auswahl für die Komposition und das Setzen von Gewichten bleibt die Graphen- und Textfelderdarstellung. Die Textfelderaufstellung ähnelt den bisherigen Darstellungen Boolescher Anfragen. Fehleingaben vom Benutzer werden hier durch den Einsatz von vordefinierten Bausteinen und Restriktionen der Positionierung vermieden. Auch beim Graphen sind nur Knoten aus der Menge aufgestellter Bedingungen auswählbar. Nicht zusämmenhängende Graphen können jedoch auftreten, weil das automatische Setzen von Booleschen Operatoren, wie es bei den Textfeldern der Fall ist, nicht möglich ist. Es könnte erst zum Zeitpunkt des Suchstarts eine Kante hinzugefügt werden, weil erst dann der Graph vom Benutzer vollständig konstruiert wurde. Es würde den Nutzer jedoch verunsichern, wenn eine Kante die beiden Teilgraphen ohne sein Zutun verbinden würde. Das Problem wird gelöst, indem intern die einzelnen Bäume durch eine Disjunktion verknüpft werden, diese aber nicht als Kante gezeichnet wird. Bei Graphen kann es auch zu Zyklen kommen. Beim Hinzufügen der Kante, die zu einem Zyklus führen würde, wird der Benutzer auf das Löschen einer Kante hingewiesen.

Falls der Nutzer die Warnung durch Schließen ignoriert, wird die zuletzt gezeichnete Kante automatisch entfernt. Das Problem wird so gut behoben. Die Prioritäten von Operatoren werden beim Graphen gut durch die Dicke und die Sättigung der Farbe der Kante deutlich. Bei den Textfeldern hingegen ist das nicht auf den ersten Blick erkennbar. Zwar werden die Klammern farblich hervorgehoben, jedoch müssen dessen Positionen zueinander verglichen werden, um Rückschlüsse ziehen zu können. Bei beiden Darstellungsvarianten können gleiche Prioriäten der Operatoren auftreten. Beim Graphen wird in diesem Fall eine Konjunktion höher eingestuft als eine Disjunktion, und bei gleichen Operatoren eine zufällige Kante ausgewählt. In der Textfelderdarstellung wird aufgrund der Leserichtung der linksstehende Term höher priorisiert. Der entscheidende Punkt ist das einfache Aufstellen einer konkreten Anfrage. In der Annahme der Nutzer lässt die Bedingungen automatisch generieren, um somit eine Zeitersparnis zu haben, ist bei den Textfeldern ein häufiges Verschieben der Bausteine sehr wahrscheinlich. Die Knoten des Graphen sind zu Beginn optimal ausgerichtet, und zwar so, dass jede Kante gezogen werden kann. Desweiteren kann eine automatische Umsortierung mittels FDP bereitgestellt werden, wodurch die Kantenschnitte beseitigt werden und der Graph ästhetisch ausgerichtet wird. Dadurch spart der Nutzer Zeit. Desweiteren ist der Graph bei mehreren Bedingungen übersichtlicher als die Textfelder. Die Textfelder sind linear aneinandergereiht, notfalls in unterschiedlichen Zeilen, wodurch der Überblick vernachlässigt wird. Werden zusätzlich Gewichte eingeblendet, so ist das Setzen dieser im Graph einfacher und schneller als bei den Textfeldern. Bei der letzteren Variante ist das visuelle Erfassen der zugehörigen Schieberegler nicht so schnell möglich, weil die Bausteine eng aneinander liegen. Die Übersicht bei der zusätzlichen Anzeige von Gewichten nimmt in dieser Variante weiter ab. Aufgrund der genannten Faktoren Übersichtlichkeit und Erkennbarkeit, einfache und schnelle Handhabung soll die Graphendarstellung in der GUI verwendet werden um eine Anfrage mit Gewichten aufzustellen. Die Abbildung 5.30 zeigt den Tab *Automatic Search* auf der letzten Stufe der Multi-Layer-Architektur, wo alle bisher erläuterten Bestandteile der automatischen Suche in einer Gesamtansicht vereinigt sind.

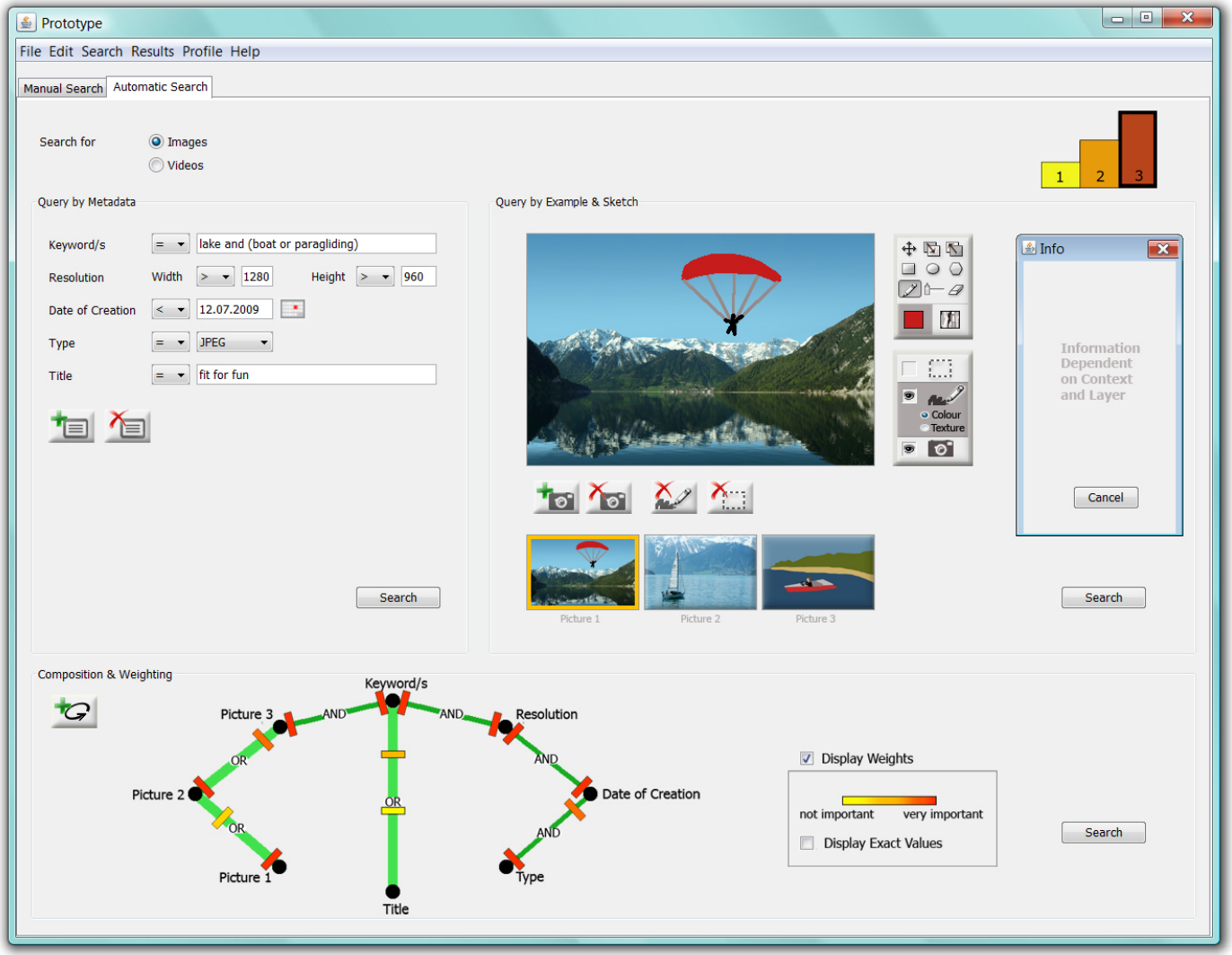


Abbildung 5.30: Tab der anfragebasierten Suche mit der aus den Beispielen 3, 4 und 5 zusammengesetzten Anfrage

Multi-Layer-Umsetzung

Buttons, Felder und Funktionen, die auf einer Stufe deaktiviert sind, werden vollständig ausgeblendet, sodass der Anfänger nicht irrittiert wird. Der Einsteiger kann im Feld *Query by Metadata* Stichwörter unter *Keyword/s* eingeben. Das Hinzufügen weiterer Bedingungen ist für ihn meist irrelevant und soll deshalb hier noch nicht möglich sein. Im Feld *Query by Example & Sketch* kann nur ein Beispielbild aus einem Verzeichnis hinzugefügt werden. Das Zeichnen einer Skizze bedarf den Umgang mit den Werkzeugen und ist deshalb hier deaktiviert. Die Suche kann dann gestartet werden über einen der Search-Buttons, wobei diese nur nach den Metadaten oder nur nach dem Foto oder in Kombination von beiden erfolgen kann. Für die letzte Variante muss auf den Search-Button im Bereich *Composition & Weighting* geklickt werden. Abbildung 5.31 zeigt den initialen Zustand auf Ebene 1.

Auf der zweiten Stufe der Multi-Layer-Architektur ist keine Einschränkung im Metadaten-Bereich vorhanden. Alle verfügbaren Metadaten können ausgewählt werden. Desweiteren kann zusätzlich gezeichnet werden. Jetzt können insgesamt 3 Bilder in eventueller Kombination mit Skizzen angegeben werden. Die Werkzeugleiste und die Ebenenanzeige sind somit aktiviert. Ausgenommen davon ist das Angeben von Segmenten, wodurch die Segmentebene nicht eingeblendet ist. Der Nutzer wird im Normalfall keine Segmente angeben wollen, da das mit etwas Aufwand verbunden ist. Außerdem hat er so einen besseren Einstieg in die Ebenenverwaltung, wenn zu Beginn nur zwei anstatt drei Ebenen angezeigt werden. Abbildung 5.32 zeigt den initialen Zustand auf Ebene 2.

In der Expertenstufe kommt dann die Funktionalität der Segmente hinzu. Erst für den versierten Anwender steht im unteren Bereich der Anzeige die Möglichkeit bereit, die Bedingungen selbst mit Booleschen Operatoren mittels eines Graphen zu verknüpfen. Der Grund hierfür ist das notwendige Verständnis der Booleschen Algebra. Bei einer genauen Vorstellung von der Wichtigkeit der Bedingungen können die Operatoren anstatt zufällig auch manuell gewichtet werden. Das ist in den unteren Stufen nicht möglich, weil das hauptsächlich von Experten verlangt wird. Der durchschnittliche Anwender möchte nicht soviel Zeit in die Anfrageformulierung investieren.Er verlässt sich auf Automatismen, die für ihn die Arbeit übernehmen. Im Zuge des Relevance-Feedbacks werden die Werte der Gewichte modifiziert entsprechend den gesetzten Benutzerpräferenzen. Bereits zu Beginn der Ausführungen zur anfragebasierten Suche wurde der initiale Zustand auf Ebene 3 in Abbildung 5.15 gezeigt.

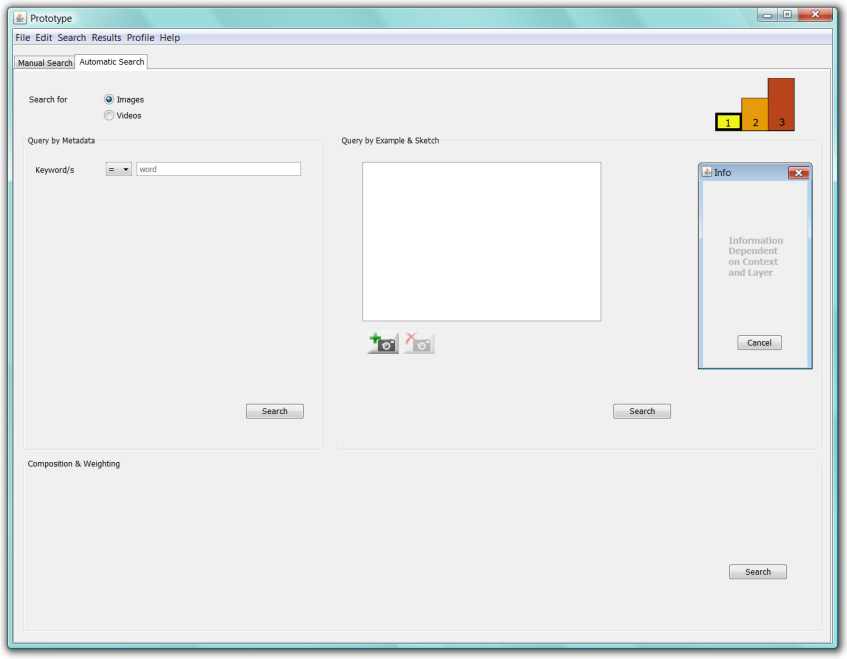


Abbildung 5.31: Tab der anfragebasierten Suche im initialen Zustand auf Ebene 1

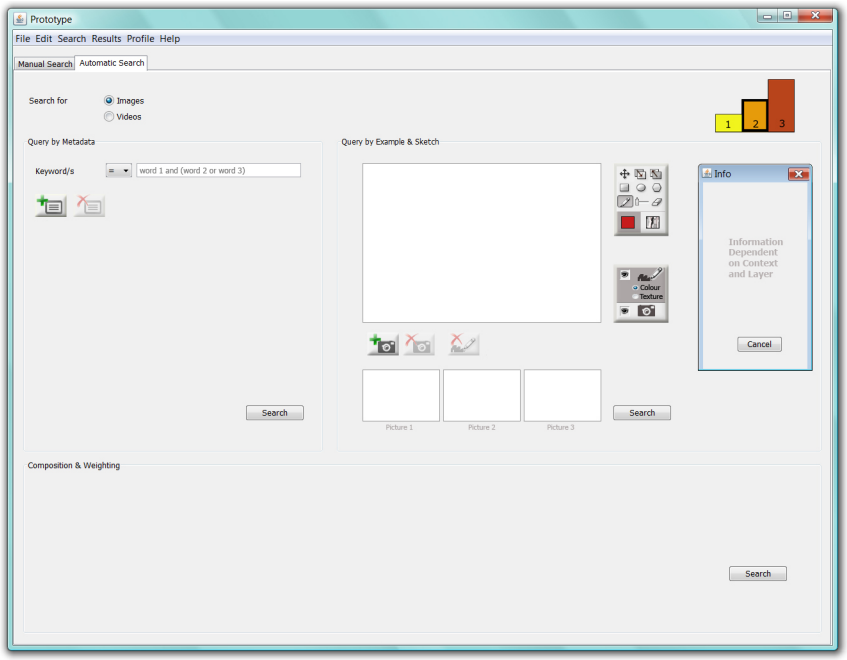


Abbildung 5.32: Tab der anfragebasierten Suche im initialen Zustand auf Ebene 2

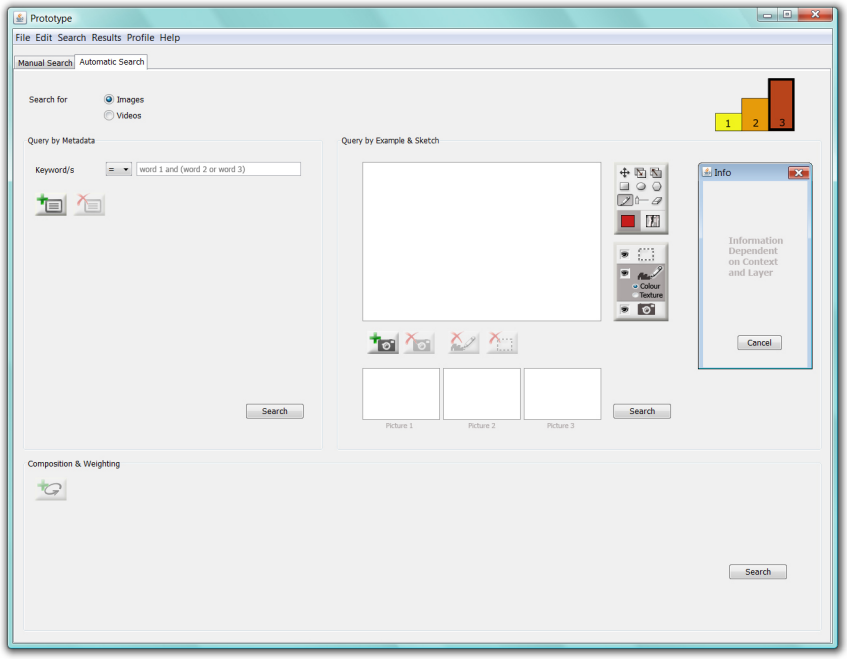


Abbildung 5.33: Tab der anfragebasierten Suche im initialen Zustand auf Ebene 3

5.1.3 Ergebnispräsentation und Relevance-Feedback

Nach einem Klick auf einen Search-Button unter dem Tab *Automatic Search* öffnet sich der Tab *Results*. Dort befinden sich die Ergebnisse der anfragebasierten Suche. Auf den angezeigten Systempräferenzen gibt der Nutzer im Zuge des Relevance-Feedbacks seine eigenen Präferenzen an.

5.1.3.1 Ergebnispräsentation

Im Folgenden werden die verschiedenen Varianten für die Ergebnispräsentation vorgestellt:

- modifiziertes Hasse-Diagramm
- Spirale
- Polygon

Modifiziertes Hasse-Diagramm

Für die Darstellung der charakteristischen Präferenzmenge eignet sich ein Hasse-Diagramm. Mit diesem Diagramm lässt sich eine Halbordnung übersichtlich darstellen. In Abbildung 5.34 wird ein Beispiel gezeigt. Ist o_1 Teiler von o_2, wobei o_1 und o_2 nur aus der

Menge der Teiler von 12 stammen dürfen, so existiert eine Kante zwischen diesen beiden Elementen, wenn $o_1 \neq o_2$ und kein o_3 existiert mit o_1 Teiler von o_3 und o_3 Teiler von o_2. o_1 wird dann unterhalb von o_2 gezeichnet, um die Richtung der Teilbarkeit anzuzeigen.

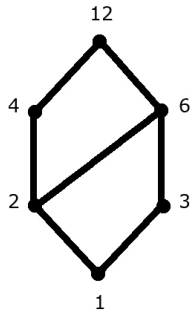


Abbildung 5.34: Hasse-Diagramm für die Relation $R = \{(o_1, o_2) \,|o_1$ Teiler von $o_2\}$ mit $o_1, o_2 \in \{1, 2, 3, 4, 6, 12\}$

Wie auch im Hasse-Diagramm existieren in der charakteristischen Präferenzmenge P keine reflexiven und transitiven Paare. Desweiteren ist die Menge um nutzlose sowie für die k-Äquivalenz unnötigen Präferenzen reduziert. Aus dem Algorithmus zur Bestimmung von P aus [SZ09] kann abgelesen werden, dass diese Menge maximal k-1 Präferenzen enthält. Für die Präsentation dieser minimalen Präferenzmenge werden im Gegensatz zum Hasse-Diagramm die Präferenzen $o_1 \geq o_2$ nicht vertikal gerichtet angezeigt, sondern von links nach rechts. Der Nutzer sieht aufgrund seiner Leserichtung zuerst das bessere Foto o_1. Um die Präferenz noch deutlich sichtbar zu machen, nimmt in Abbildung 5.35 die Sättigung der Hintergrundfarbe zu den als schlechter eingestuften Fotos ab.

Sortiert man alle Präferenzen $o_i \geq o_{i+1}$ aus P anhand der Position i eines Fotos im Rank aufsteigend, so wird die erste Präferenz dieser sortierten Menge in der ersten Zeile abgebildet. Das Prinzip setzt sich für folgende Präferenzen aus P in der gleichen Zeile fort. Bilder, die als Objekt in einem Paar von P existieren, werden immer dann in einer neuen Zeile platziert, wenn keine Präferenz in P zum letzten Foto der vorhergehenden Zeile existiert. Sei hier vereinfacht $\{o_1 \geq o_2, o_2 \geq o_3, o_5 \geq o_6, o_6 \geq o_7, o_7 \geq o_8\}$ die charakteristische Präferenzmenge P, so würden die Fotos 1, 2 und 3 hintereinander in der ersten Zeile stehen und die Bilder 5, 6, 7 und 8 in der zweiten. Je mehr Fotos in einer Zeile eingeordnet werden müssen, desto kleiner wird deren Anzeigegröße. Für eine große Zeilenanzahl wird eine Scrollleiste eingeblendet. Für Experten können neben den Präferenzen auch die konkreten Score-Werte von Interesse sein. Diese sind über das Setzen eines Häkchens in der Check-Box, die sich neben der Ansicht befindet, unter jedem Foto einblendbar. Die Abbildung 5.35 zeigt 15 Ergebnisse in der modifizierten Hasse-Diagramm-Darstellung für das stets

verwendete Beispiel in Abbildung 5.30 mit dem Boot bzw. dem Paraglider in Verbindung mit einem See.

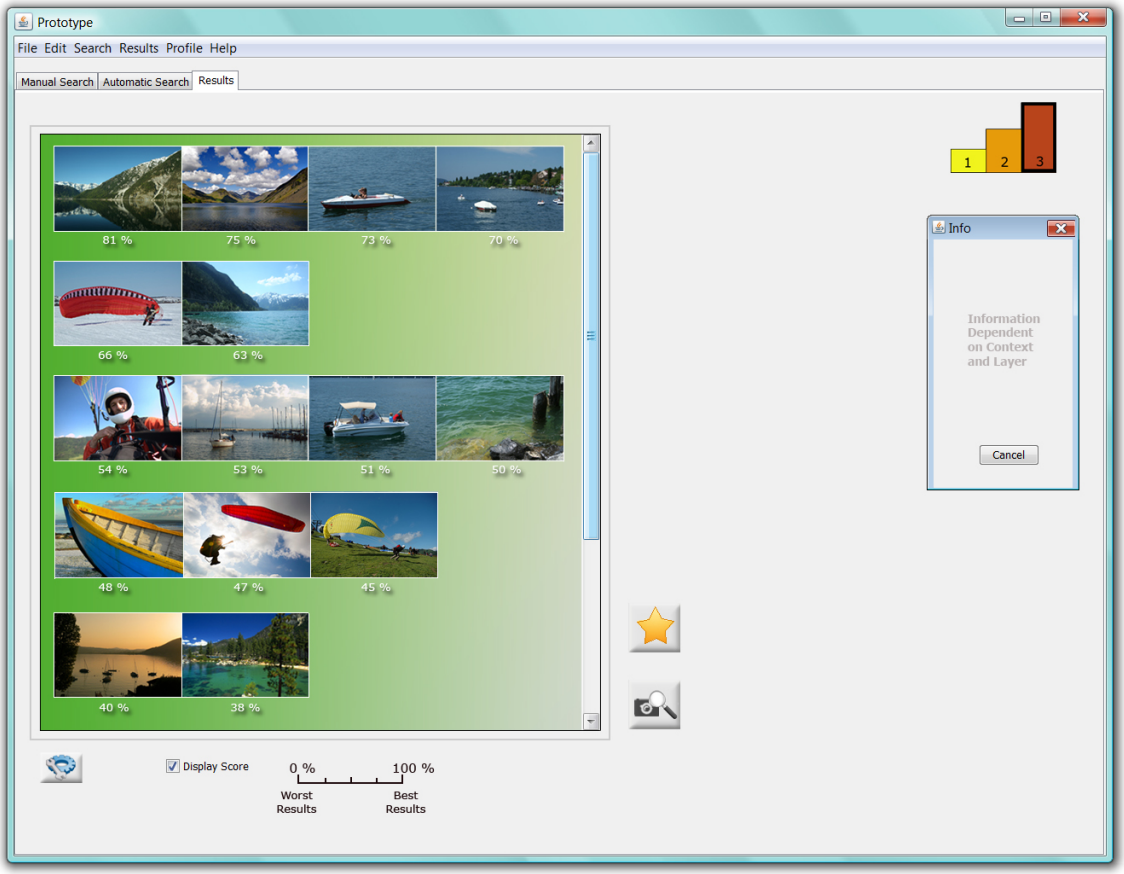


Abbildung 5.35: 15 Fotos in der modifizierten Hasse-Diagramm-Darstellung

Informationsebenen Für den Wechsel zwischen Übersicht-, Übergangs- und Detailebene werden die gleichen Idiome wie bei der manuellen Suche bei SOMs verwendet. Das ist nötig, um Konsistenz zu bewahren, wie es die 1. Regel aus Abschnitt 4.1.3 besagt. Beim Darüberfahren der Maus über ein Bild wird dieses etwas vergrößert angezeigt. Es können auch zusätzliche Metadaten unter der etwas vergrößerten Darstellung des Bildes eingeblendet werden, wie beispielsweise Aufnahmeort. Auch nur die Metadaten anzuzeigen ist eine Variante. Abbildung 5.36 zeigt zwei Beispiele der Übergangsebene. Initial wird nur das Foto eingeblendet, für andere Einstellungen steht der Konfigurationsbutton bereit. Aus einer Liste können nach einem Klick auf diesen Button die Einträge gewählt werden, wie sie bereits in Abbildung 5.16 zu sehen waren. Um in die Detailansicht eines Fotos zu gelangen, muss auf die linke Maustaste doppelt geklickt werden. Es öffnet sich

ein neuer Tab namens *Details*. Um mit möglichst wenig Aufwand wieder in die ursprüngliche Tabansicht zurückzuwechseln, wird das Pendant zum Öffnen verwendet. Es genügt ein Doppelklick mit der rechten Maustaste auf einen beliebigen Bereich, ausgenommen Buttons und Ähnliches. Der Tab kann aber auch einfach geschlossen werden. Ein Kreuz direkt neben dem Tabnamen steht hierfür bereit. Für die folgenden Ansätze zur Ergebnispräsentation gilt das Gleiche für die Informationsebenen wie hier beschrieben, es sei denn es wird auf Ausnahmen hingewiesen.

Abbildung 5.36: mögliche Darstellungen in der Übergangsebene - Foto und Metadaten (links), nur Metadaten (rechts)

Spirale

Eine weitere Möglichkeit die Präferenzen anzuzeigen, ist eine Spirale zu verwenden. Die nach dem antiken Mathematiker Archimedes benannte Spirale, die Archimedische Spirale, kann mit der Formel $r(\varphi) = c \cdot \varphi$ beschrieben werden, welche in Polarkoordinaten angegeben ist [DGGO08]. r ist der Radius, $c > 0$ eine positive Konstante und φ der Winkel. Die Besonderheit der Archimedischen Spirale besteht im konstanten Windungsabstand im gesamten Definitionsbereich.

Die Abbildung 5.37 zeigt links die Archimedische Spirale für den Definitionsbereich $[0, 8\Pi]$.

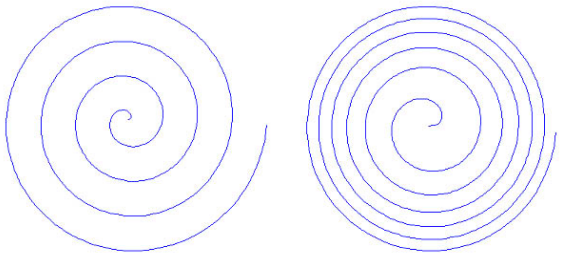


Abbildung 5.37: links: Archimedische Spirale [JA09a], rechts: Fermatsche Spirale [JA09b]

Bilder sollen entlang der Spirale platziert werden, wobei der Mittelpunkt eines jeden Fotos auf der Spirale liegt. Der Nachfolger eines Fotos wird direkt an der Kante des Vorgängers platziert. Die Abbildung 5.38 verdeutlicht dieses, indem die Mittelpunkte rot gekennzeichnet sind. Die eingezeichnete schwarze Linie und die Mittelpunkte sollen dem Nutzer nicht angezeigt werden. Der Parameter c muss in Abhängigkeit von der Größe der anzuzeigenden Bilder so gewählt werden, dass sich die Fotos von benachbarten Spiralwindungen nicht schneiden.

Abbildung 5.38: Platzierung von 30 Fotos auf einer Archimedischen Spirale

Im Pol befindet sich das Foto aus der charakteristischen Präferenzmenge P mit dem höchsten Score-Wert. Die Präferenz $o_i \geq o_{i+1} \in P$ ist aus der Position der Bilder abzulesen. Das Foto i ist dann als Vorgänger des Fotos $i + 1$ dargestellt. Unter Vorgänger ist dasjenige der beiden angrenzenden Bilder zu verstehen, welches beim Ablaufen der Spirale vom Mittelpunkt an zuerst angefunden wird. Ein Foto ist somit besser als ein anderes, wenn es einen kleineren Radius besitzt.

Neben der Archimedischen Spirale könnte auch eine Fermatsche Spirale verwendet werden. Diese ist in Abbildung 5.37 für den Definitionsbereich $[0, 12\pi]$ zu sehen. Die besseren Ergebnisse befinden sich wie bisher nahe des Pols und werden hier größer dargestellt, die schlechteren kleiner, weil mit ansteigendem Radius der Windungsabstand bei dieser Spiralenart kleiner wird. Die Größe eines Fotos ist hierbei funktional abhängig von dem Score-Wert. Dem Nutzer können dadurch die Systempräferenzen noch deutlicher werden. Der Blick kann durch den Verlauf der Spirale und durch die Größenabnahme der Fotos gezielter von den relevanteren zu den irrelevanteren gelenkt werden, siehe Abbildung 5.39.

Abbildung 5.39: 30 Ergebnisse in der Darstellung einer Fermatschen Spirale

Neben der Vergrößerung von nur einem Foto, indem die Maus auf diesem verweilt, kann der Nutzer auch in die Ansicht hineinzoomen, um bei Bedarf auch mehrere irrelevantere Bilder anzuschauen. Das ist genauso wie bei den SOMs über das Scrollrad geregelt. Panning sowie das Einblenden der konkreten Score-Werte ist auch wieder möglich.

Um die gesamte Anordnung noch mehr der Form einer Spirale anzugleichen, müssten Tests unternommen werden, inwiefern die Positionierungen mit unterschiedlichen Berechnungsmethoden der Ästhetik des Nutzers entsprechen. Man könnte beispielsweise statt die Forderung zu erfüllen, dass alle Mittelpunkte der Fotos auf der Spirale zu liegen haben, auch die Fläche als Kriterium heranziehen. Ein Bild wird immer noch an der Kante des Vorgängers ausgerichtet. Die eine variable Koordinate wird so bestimmt, dass die Linie der Spirale die Fläche des Fotos halbiert.

Polygon

Bei den bisherigen Darstellungen konnte man nur die Systempräferenzen ablesen bzw. zusätzlich auch den genauen Ähnlichkeitswert anzeigen lassen. Den Nutzer kann es auch interessieren, inwieweit die einzelnen Bedingungen, die er in seiner Anfrage angegeben hat, bei einem Foto erfüllt werden. In einem gleichseitigen n-eckigen Polygon stellen die Geraden vom Mittelpunkt bis zu den Ecken einzelne Bedingungen dar. Die Position (x_o, y_o) eines Fotos o innerhalb dieses Polygons ist abhängig von der Erfüllung der einzelnen Bedingungen.

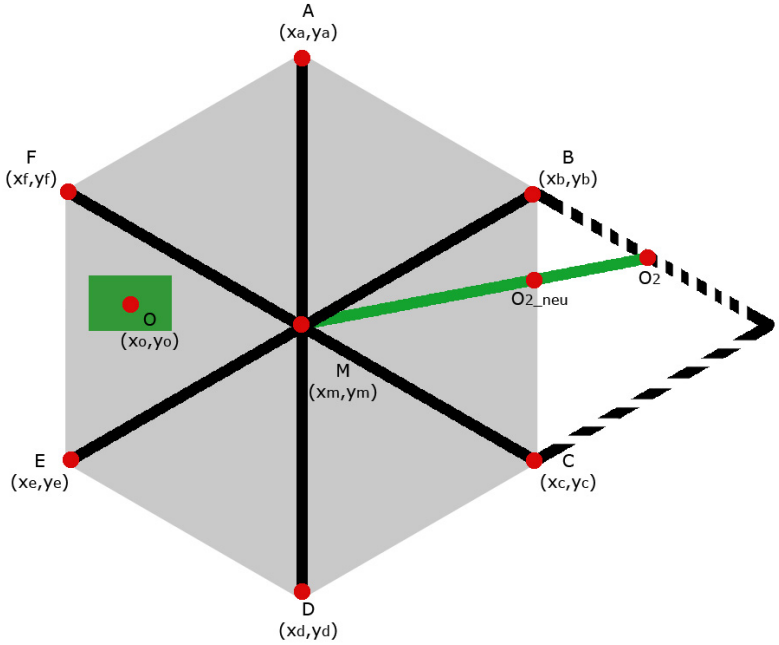


Abbildung 5.40: sechseckiges Polygon

Abbildung 5.40 zeigt ein gleichseitiges Sechseck. Für die Berechnung der Position eines Fotos kommt folgende Formel zum Einsatz:

$$(x_o, y_o)^T = s_a \cdot (x_a - x_m, y_a - y_m)^T + s_b \cdot (x_b - x_m, y_b - y_m)^T + \ldots + s_f \cdot (x_f - x_m, y_f - y_m)^T$$

s_i stellt den Grad der Erfüllung einer Bedingung für ein Foto dar. Bei einer Booleschen Bedingung können nur die Werte 0 oder 1 angenommen werden, bei Score-basierten Bedingungen liegt s_i im Intervall [0,1]. Wird ein Beispielbild in die Anfrage integriert, so entspricht s_i dem Ähnlichkeitswert eines Fotos zum QBE. Betrachtet man nur die Gerade $\overline{MA}$, so wird ein Foto auf dem Punkt A platziert, wenn es vollständig diese Bedingung erfüllt, bei minimaler Erfüllung liegt es auf dem Mittelpunkt M, sonst dazwischen. Es

kann vorkommen, dass die berechnete Position eines Bildes auch außerhalb des Polygons liegt, z.B. für $s_b + s_c > 1$ und $s_a = s_d = s_e = s_f = 0$. Dieser Fall ist in der Abbildung 5.40 zu sehen. Die Koordinaten können anhand der obigen Formel für dieses Beispiel Werte aus dem Bereich bis zur gestrichelten Linie annehmen. Der Nutzer würde es nicht verstehen, warum ein Foto außerhalb des Vielecks liegt. Aus diesem Grund soll der Punkt O_2 auf den auf der Kante befindlichen Punkt O_{2neu} projiziert werden, so wie es in der Skizze aufgezeigt ist. Mit dem Resultat dieser mathematischen Ungenauigkeit kann der Nutzer eher umgehen.

Idiome zum Wechseln der Ansicht und der angezeigten Informationen Für das bisher eingesetzte Beispiel aus Abbildung 5.30 ist eine mögliche Anordnung von 15 Fotos in einem Polygon in Abbildung 5.41 zu sehen. Initial werden nur die Fotos im Polygon dargestellt, dieses ist auf der linken Seite erkennbar. Bewegt der Nutzer die Maus über ein Bild, so wird dieses, wie bisher auch, etwas vergößert angezeigt. Bei einem Klick auf ein Bild zeigen rot gefärbte Geraden auf den einzelnen Achsen den Grad der Erfüllung der entsprechenden Bedingung an. Das zeigt die rechte Seite der Abbildung 5.41. Wie abzulesen ist, ist z.B. die Bedingung „Title=fit for fun“ vollständig erfüllt, während die Ähnlichkeit zu den Beispielbildern vom System als gering angegeben wird. Um den Blick auf das Wichtige zu lenken, werden die anderen Bilder leicht ausgeblendet.

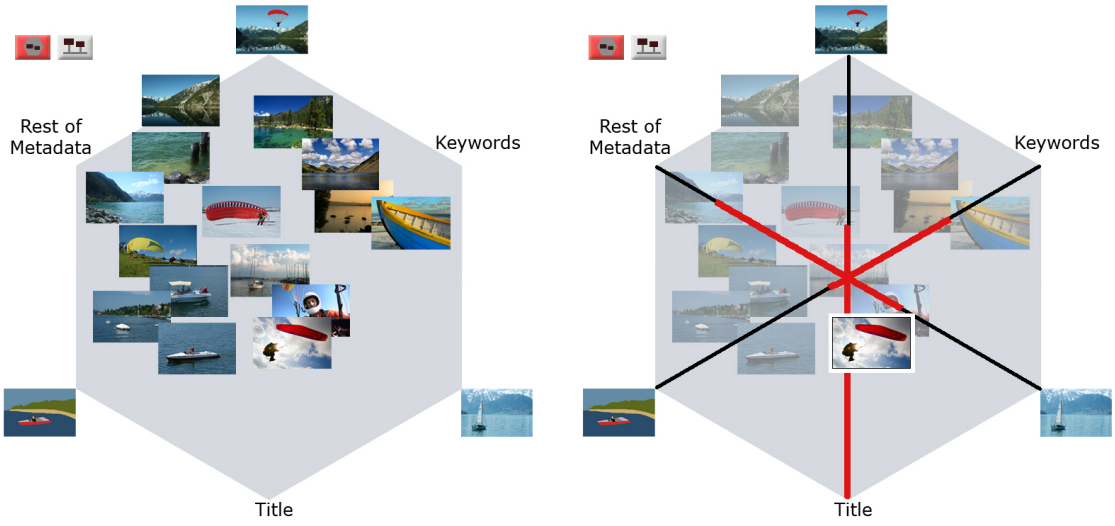


Abbildung 5.41: Polygon in der Draufsicht - links: initialer Zustand; rechts: Informationen über den Grad der Erfüllung der einzelnen Bedingungen für das hervorgehobene Bild

Der Nutzer möchte jedoch auch wissen, welche Bilder in Abhängigkeit der gesetzten Gewichte vom System besser eingestuft werden. Das Polygon ist bisher nur in der Sicht von oben abgebildet, wo die Erfüllung der Bedingungen einzeln betrachtet wurde, ohne deren Gewicht aus der Anfrage einzubeziehen. Um den Grad der Erfüllung eines Fotos einer gewichteten Anfrage anzuzeigen, also den Gesamt-Score-Wert, kann das Polygon über Bewegungen der Maus mit gedrückter rechter Taste dreidimensional bis zur Seitenansicht gekippt werden. Die Fotos drehen sich hierbei mit, bleiben perspektivisch unverzerrt, sodass diese weiterhin identifizierbar sind. Die Bilder sind jeweils auf einem orthogonal zur Polygon-Ebene befindlichen Stab befestigt, wodurch die Präferenzen leicht durch die Höhenunterschiede der Stäbe ablesbar sind. Ein Foto, welches sich höher als ein anderes befindet, wird vom System besser eingeschätzt. Die exakte Länge des Stabes ist durch den Score-Wert festgelegt. In der Abbildung 5.42 wird links das Polygon teilweise gekippt und rechts wird es in der Seitenansicht gezeigt. Auch in dieser Variante sind die Gesamt-Score-Werte über eine Checkbox einblendbar. Der Nutzer kann sich interaktiv in der gesamten Ansicht bewegen. Rechts- und Linksbewegungen der Maus bei gedrückter rechter Taste lassen die Ansicht horizontal rotieren. Dadurch können in der Seitenansicht verdeckte Bilder sichtbar werden. Wird die Maus mit gedrückter rechter Taste nach oben oder unten bewegt, wird die Ansicht vertikal rotiert, sodass stufenlos von der Sicht von oben in die Sicht der Seite und andersherum gewechselt wird. Es stehen zusätzlich zwei Buttons in der linken oberen Ecke bereit, um mit einem Klick in eine der Ansichten wechseln zu können. Über das Scrollrad lassen sich die Objekte näher heranholen. Mit der linken Maustaste wird die Ansicht nicht verändert. Der Nutzer kann, wie weiter oben beschrieben, dadurch weitere Informationen über die einzelnen Objekte erhalten und abrufen.

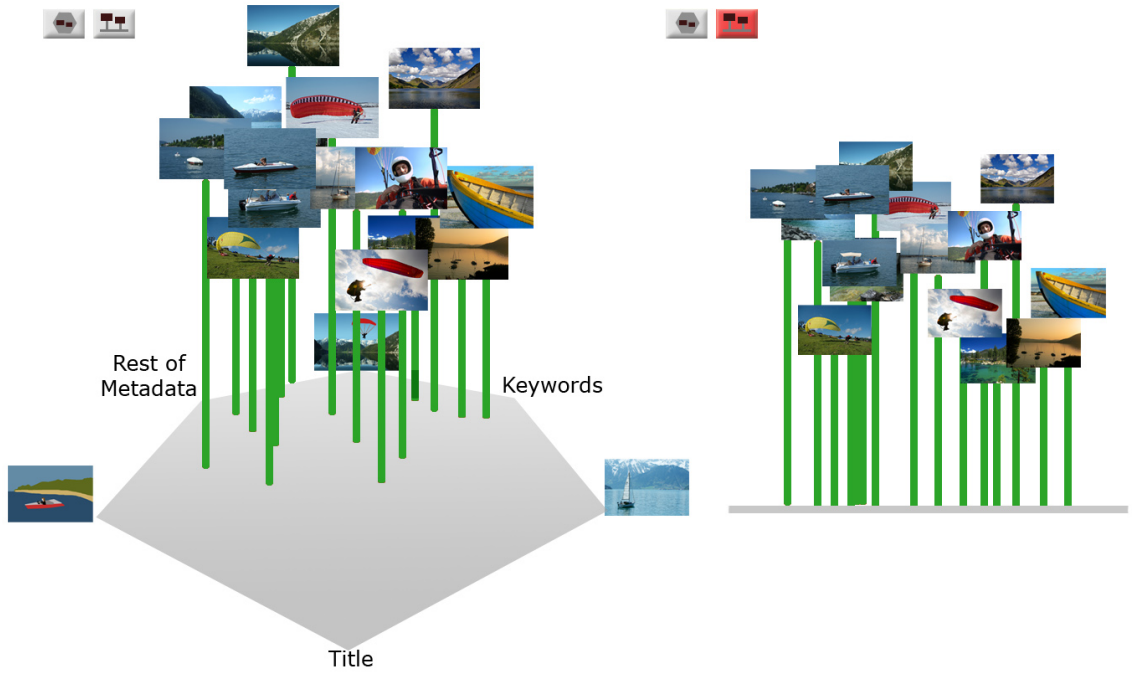


Abbildung 5.42: links: Polygon in der Vogelperspektive; rechts: Polygon in der Seitenansicht

Probleme mit der Polygondarstellung Ein Foto kann als ein Vektor mit n Dimensionen aufgefasst werden, wobei jede Dimension den Erfüllungsgrad einer bestimmten Bedingung für dieses Bild angibt. Eine Anfrage kann aus vielen Bedingungen bestehen, wodurch die Vektoren hochdimensional werden. Da sich der Nutzer die Lage der Fotos in diesen Vektorräumen nicht vorstellen kann, sind die Bilder auf einer zweidimensionalen Fläche dargestellt. Das bedeutet jedoch, dass die Achsen im Polygon für mehr als zwei Bedingungen $n > 2$ nicht orthogonal sind. Das führt zu dem Problem, dass die Abbildung eines Fotos auf eine Position im Polygon nicht bijektiv ist. Zur Verdeutlichung können im obigen Beispiel mit 6 Bedingungen zwei Fotos $o1$ und $o2$ mit unterschiedlichen Vektoren $(1, 0, 1, 0, \ldots, 0)^T$ und $(0, 1, 0, 0, \ldots, 0)^T$ auf die gleiche Position projiziert werden:

$$\begin{aligned} (x_{o1}, y_{o1})^T &= \quad 1 \cdot (x_a - x_m, y_a - y_m)^T + \quad 1 \cdot (x_c - x_m, y_c - y_m)^T \\ = (x_{o2}, y_{o2})^T &= \quad 1 \cdot (x_b - x_m, y_b - y_m)^T \end{aligned}$$

Aufgrund der linearen Abhängigkeiten der Achsen sind verschiedene Interpretationen für die Lage eines Fotos im Polygon möglich. Hat ein Foto die Position $(x_b - x_m, y_b - y_m)^T$,

so kann das zum einen bedeuten, dass nur die Bedingung B vollständig erfüllt ist, zum anderen können nur die beiden Bedingungen A und C erfüllt sein. Ein weiteres Beispiel wäre die Platzierung eines Fotos im Mittelpunkt des Polgyons. Der Nutzer würde höchstwahrscheinlich daraus schließen, dass keine der Bedingungen erfüllt ist. Das ist nicht zwingend der Fall, aufgrund der Linearkombination können sich einzelne Werte gegenseitig auslöschen. Um das Problem etwas abzuschwächen, kann der Nutzer die genauen Werte für den Erfüllungsgrad der einzelnen Bedingungen durch Klick auf ein Foto erfahren. Desweiteren können Boolesche Bedingungen, weil sie nur die Werte 0 oder 1 einnehmen, die Lage der Fotos zu stark beeinflussen. Aus diesem Grund sind in der Abbildung 5.41 die Achsen für die Booleschen Metadaten unter *Rest of Metadata* zusammengefasst. *Title* und *Keyword/s* können aufgrund der Beachtung der Vagheit der Sprache Werte aus dem Intervall [0,1] annehmen. Neben dem Problem der verschiedenen Interpretationen der Lage eines Fotos nimmt auch mit zunehmender Anzahl an Bedingungen die Übersichtlichkeit ab. Um diesen beiden Problemen entgegenzutreten, kann die Karhunen-Loeve-Transformation (KLT) angewendet werden [Fuk90]. KLT ist in der Literatur auch unter den Begriffen Hauptachsentransformation (HAT) oder Principal-Component-Analysis (PCA) zu finden. Mithilfe von KLT können lineare Abhängigkeiten von Dimensionen erkannt und entfernt werden, sodass die Zahl der Ecken eines Polygons auf das Minimum reduziert wird. Können einzelne Dimensionen als Linearkombination anderer dargestellt werden, so können diese auch weggelassen werden, weil sie nur redundante Daten darstellen. Die Vektoren befinden sich dann bildlich gesprochen entlang einer Achse. Die am stärksten ausgeprägte Achse wird als Hauptachse bezeichnet, wodurch HAT seinen Namen trägt. Die Achsen liegen jedoch meist nicht parallel zu den kartesischen Koordinatenachsen, wodurch diese nicht einfach entfernt werden können. Ziel ist es deshalb durch Rotation und Translation der Vektoren die Achsen auf die Dimensionen abzubilden. Dimensionen mit einer geringen Streuung können dann entfernt werden. In [Sch06] wird die Vorgehensweise zur Bestimmung der Hauptachse und der Rücktransformation der Vektoren in den um einige Dimensionen reduzierten Vektorraum genauer beschrieben. Die einzelnen Achsen des entstehenden Vektorraumes sind jedoch für den Menschen meist schwer zu benennen. Für den Nutzer müssen die Achsen verständlich beschriftet werden, sonst kann er das Polygon nicht deuten. Aus diesem Grund muss getestet werden, ob für die Bilder einer konkreten Datenbank die Hauptachsentransformation etwas nützt.

Es kann auch zur Informationsanhäufung kommen, wenn mehrere Bilder die gleiche oder eine ähnliche Position haben. Desweiteren ist der Platz im Polygon begrenzt. Deswegen werden mehrere Polygonebenen eingesetzt. Ab einer gewissen Anzahl an Bildern werden Fotos mit einem Score-Wert unter einem Schwellwert in der nächsttieferen Ebene ange-

zeigt. Nicht nur von der Anzahl der Bilder, sondern auch von dem mittleren Grad der Überlappung der Bilder einer Ebene ist der Schwellwert abhängig. Ist dieser Grad noch sehr klein, verdeckt beispielsweise nur eines von vielen Fotos ein anderes mehr als 50 %, so können diese abwechselnd nach einer gewissen Zeit angezeigt werden. Die Anzahl der Ebenen wird entsprechend dynamisch ermittelt. Der Nutzer kann über zwei Buttons in eine höhere oder tiefere Ebene wechseln.

Entscheidung für eine Darstellungsvariante

In der modifizierten Hasse-Diagramm-Darstellung werden die Präferenzen aufgrund der Leserichtung und der Aufteilung in Zeilen deutlich, bei den Polygonen sind diese über Höhenunterschiede in der Seitenansicht erkennbar. Bei der Fermatschen Spirale wird der Blick des Nutzers durch die Größe eines Bildes und den Verlauf der Kurve von den besser bewerteten Fotos zu den schlechter bewerteten gelenkt. Bei der Polygondarstellung können in Abhängigkeit von den gestellten Bedingungen und der vorliegenden Bilddatenbank in einigen Regionen des Polygons keine Bilder platziert werden. Desweiteren kann es auch zur Informationsanhäufung kommen. Dieses Problem wird durch den Einsatz von mehreren Polygonen minimiert. Das Hauptproblem bei der Polygondarstellung ist die zweidimensionale Darstellung eines n-dimensionalen Vektorraums in der Draufsicht, wodurch falsche Interpretationen möglich sind. Zwar können eventuell auftretende lineare Abhängigkeiten der einzelnen Bedingungen durch die KLT beseitigt werden, jedoch bleibt höchstwahrscheinlich die Dimension der Vektoren größer 2. Außerdem könnte bei der KLT in den meisten Fällen keine leicht verständliche Bezeichnung der Ecken des Polygons gefunden werden. Das Problem der verschiedenen Interpretationen für eine Position wird zwar abgeschwächt durch die Anzeige der konkreten Werte aller Dimensionen beim Darüberfahren der Maus über ein Foto, jedoch ist dieses auch nur für ein Foto gleichzeitig möglich. Die Polygondarstellung wird aufgrund der genannten Nachteile nicht für die GUI eingesetzt. Es bleiben die Fermatsche Spirale und das Hasse-Diagramm zur Auswahl. Beim Hasse-Diagramm werden die Präferenzen von links nach rechts angezeigt. Bilder werden in einer neuen Zeile angezeigt, wenn keine weitere Präferenz zum letzten Bild der aktuellen Zeile existiert. Um das zu verdeutlichen ist der Abstand zwischen den Fotos einer Zeile minimal gehalten und zwischen verschiedenen Zeilen größer. Bei der Spirale kann im Gegensatz zum Hasse-Diagramm nicht abgelesen werden, wo keine Präferenzen zwischen aufeinanderfolgenden Bildern existieren. Der Vorteil der Spirale ist jedoch, dass Bilder mit besseren Score-Werten größer angezeigt werden. Sie werden dadurch schneller wahrgenommen als irrelevante. Die Größe eines Fotos ist funktional abhängig von dem erreichten Score-Wert eines Fotos. Die relative Größe der Fotos zueinander gibt somit Aufschluss über die Score-

Wert-Differenzen. Aufgrund der letzten beiden genannten Vorteile soll die Spirale in der GUI für die Ergebnispräsentation verwendet werden.

5.1.3.2 Relevance-Feedback

Nachdem sich der Nutzer die Systempräferenzen P angeschaut hat, gibt er seine eigenen Präferenzen P' an. Er kann explizit zwei Fotos miteinander vergleichen $o_1 \geq o_2$ *für* $o_1, o_2 \in O_{Rank}$. Es könnten zwei Comboboxen bereitgestellt werden, aus denen ein Paar von Bildern ausgesucht werden kann, um sie in Relation zueinander zu stellen. Das wäre jedoch zu aufwendig. Es sollen stattdessen Möglichkeiten vorgestellt werden, die wenige Interaktionen des Nutzers benötigen und leicht verständlich sind. Im Speziellen sind das diese:

- Klassen
- konzentrische Kreise

Klassen

Die Ergebnisse kann der Nutzer in relevante und irrelevante unterteilen. Auf der rechten Seite des *Results*-Tabs befinden sich hierfür zwei Bereiche, die durch Symbole gekennzeichnet sind. Als Emblem für die relevanten Fotos kommen ein grünes Häkchen und eine Hand mit Daumen nach oben auf grünem Hintergund in Frage. Ein Häkchen tritt in der Bedeutung von *in Ordnung*, *Aufgabe erledigt* oder *gut* auf. Die grüne Farbe signalisiert Bestätigung. Eine Hand mit Daumen nach oben kann für eine *Eins*, *gut*, aber auch für *per Anhalter fahren* stehen. Das sind die weitverbreiteten Interpretationen. In einigen wenigen Ländern wie Griechenland und Australien wird die Geste jedoch als Beleidigung aufgefasst [PP04]. Irrelevante Bilder können in die Klasse eingeordnet werden, welche durch einen Mülleimer oder durch einen Daumen nach unten gerichtet dargestellt ist. Ein Mülleimer tritt in vielen der heutigen Betriebssysteme auf. Dateien, die nicht mehr benötigt werden, gelangen in den Papierkorb. Sie sind nicht mehr relevant für die Arbeit. Das Gegenstück für die Geste der Bestätigung ist eine Hand mit dem Daumen nach unten zeigend. In der vorliegenden GUI sollen die Gesten mit dem Daumen verwendet werden, um den Nutzer die Klassen anzuzeigen, in welche er passende und untreffende Fotos einordnen kann. Der Mülleimer kann für einige Anwender zu der fehlerhaften Interpretation führen, dass sie die Bilder damit aus der Datenbank löschen. Der Daumen nach oben wird für die relevanten Bilder genommen, weil die Darstellung damit konsistent ist. Die GUI wird hauptsächlich für Westeuropäer entworfen, die den Daumen nach oben zeigend als positive Bestätigung auffassen. Der Anteil der möglichen Anwender, die diese Geste falsch deuten könnten, ist sehr gering. Die Farben der Symbole sind gezielt eingesetzt.

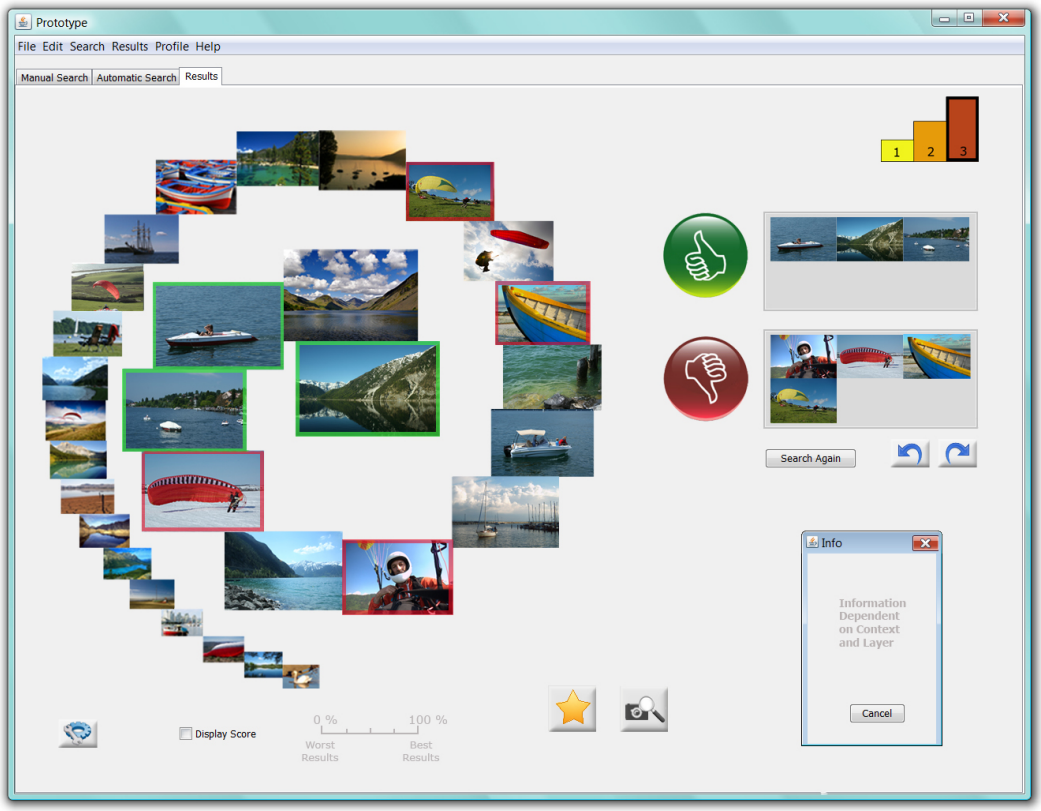


Abbildung 5.43: Ansicht des Tabs *Results* bei Verwendung einer relevanten und irrelevanten Klasse für Relevance-Feedback

Wie bereits im Abschnitt zu Fitts' Law beschrieben ist, ist die Zeit für das Erreichen der Felder von dem zurückzulegenden Weg, der Größe der Buttons etc., der Maus und von dem Nutzer selbst abhängig. Das bedeutet in diesem Fall, dass die beiden Symbole groß genug angezeigt werden und sich in der Nähe der Ergebnisse befinden müssen. Desweiteren soll auch eine Mehrfachauswahl von Elementen möglich sein, um Zeit zu sparen. Zum einen kann ein Rechteck über der Bildermenge aufgezogen werden, sodass alle darin befindlichen bzw. angeschnittenen Fotos markiert werden. Desweiteren kann ein Element durch einen Klick mit links markiert werden, mehrere durch gleichzeitiges Halten der <Shift> bzw. <Strg>-Taste, wie es aus anderen Anwendungen bereits bekannt ist. Die markierten Elemente werden farblich umrahmt und mit Drag & Drop in die gleiche Klasse einsortiert. Eine verkleinerte Ansicht der Bilder befindet sich rechts neben dem jeweiligen Symbol. Die Abbildung 5.43 zeigt die sich ergebende Ansicht des Tabs *Results* für die Klassen-Variante. In der Regel wird der Nutzer nicht mehr als 6 Fotos pro Kategorie in einem Iterationsschritt angeben. Falls trotzdem der Fall eintreffen sollte, werden die Bilder kleiner

angezeigt. Bewegt sich die Maus über den grauen Bereich, so werden alle darin enthaltenen Bilder in einer größeren Ansicht präsentiert. Der Nutzer kann bereits zugeordnete Bilder auch in die andere Klasse verschieben. Soll im Nachhinein die Bewertung für ein Element aufgehoben werden, so muss das Bild auf eine beliebige Stelle, jedoch außerhalb der beiden großen Symbole und außerhalb der zugehörigen grauen Bereiche verschoben werden.

Auswertung Ist der Nutzer mit seinen Angaben zufrieden, so klickt er auf den Button *Search Again*. Intern wird die Klasseneinteilung in Präferenzpaare umgewandelt, weil die Funktion prefsToWeight Präferenzen als Eingaben verlangt. Jedes Foto o_{rel} aus der relevanten Gruppe wird mit jedem Element o_{irrel} aus dem irrelevanten Bereich in Relation zueinander gesetzt: $o_{rel} \geq o_{irrel}$. Es ergeben sich bei n_{rel} gefallenden und n_{irrel} nicht zusagenden Fotos $n_{rel} \cdot n_{irrel}$ Präferenzpaare. Eine Differenzierung innerhalb einer Gruppe ist nicht möglich. Dem Anwender wird mit einem Dialogfenster mitgeteilt, dass aufgrund seiner Bewertungen eine neue Ergebnismenge berechnet wird, was etwas Zeit in Anspruch nehmen kann. In dieser Informationsnachricht wird zusätzlich ein Fortschrittsbalken angezeigt.

Reversible Präferenzen Wie bereits im Abschnitt 4.4 erläutert, sollen in früheren Iterationsschritten gesetzte Präferenzangaben rückgängig zu machen sein. Es stehen ein Undo- und ein Redo-Button für das Zurücksetzen der Ansicht auf frühere bzw. das Zurückspringen in die aktuelle Ansicht bereit. Diese befinden sich unter den beiden Klassen und sind durch Pfeile gekennzeichnet. Der Nutzer kennt die Symbole und Bedeutung dieser aus anderen Anwendungen. Außerdem können die Funktionen auch über den Menüpunkt *Edit* aufgerufen werden. Für den Experten stehen zusätzlich die Tastenkombinationen <Strg>+<Z> und <Strg>+<Y> bereit.

Zyklen Zyklen von Nutzerpräferenzen $o_1 \geq o_2$ und $o_2 \geq o_1$ mit $o_1 \neq o_2$ können nur dann auftreten, wenn ein Bild beiden Klassen zugeordnet werden würde. In der Ergebnismenge sind bereits bewertete Fotos entweder rot oder grün umrandet. Der Nutzer kann, während er die Ergebnisse anschaut, direkt ablesen, ob er sie bereits als relevant oder irrelevant eingestuft hat, ohne den Blick nach rechts richten zu müssen. Will der Nutzer jedoch ein Foto, welches bereits als relevant eingestuft wurde, aus der Ergebnismenge in den irrelevanten Bereich ziehen, so wird er auf den Widerspruch hingewiesen. Im Dialogfenster muss er das Foto einen der beiden Klassen zuordnen. Somit existieren zwei Möglichkeiten um ein bereits beurteiltes Foto in eine andere Klasse einzuordnen, die gerade erwähnte und die, wo von einer in die andere Kategorie das Bild verschoben wird.

Konzentrische Kreise

Eine andere Variante, um den Nutzer die Ergebnisse bewerten zu lassen, sind konzentrische Kreise. Diese sind in Abbildung 5.44 auf der rechten Seite zu erkennen. Wie bei einer Zielscheibe sollen die dem Nutzer am besten zusagenden Fotos per Drag & Drop in der Mitte platziert werden. Irrelevantere Fotos befinden sich auf den äußeren Kreisen. Mit abnehmendem Radius nimmt somit die Relevanz eines Fotos für den Nutzer ab. Um die Bedeutung der Kreise neben der Beschriftung noch deutlicher hervorzuheben, erhalten die grünen Kreisringe zum Rand hin eine geringere Sättigung. Eine Mehrfachauswahl kann kann genauso wie bei den Klassen vorgenommen werden. Der Abstand der markierten Bilder zueinander in der Spirale wird beim Verschieben in die Kreise so verkleinert, dass alle Bilder in der Zielscheibe Platz finden.

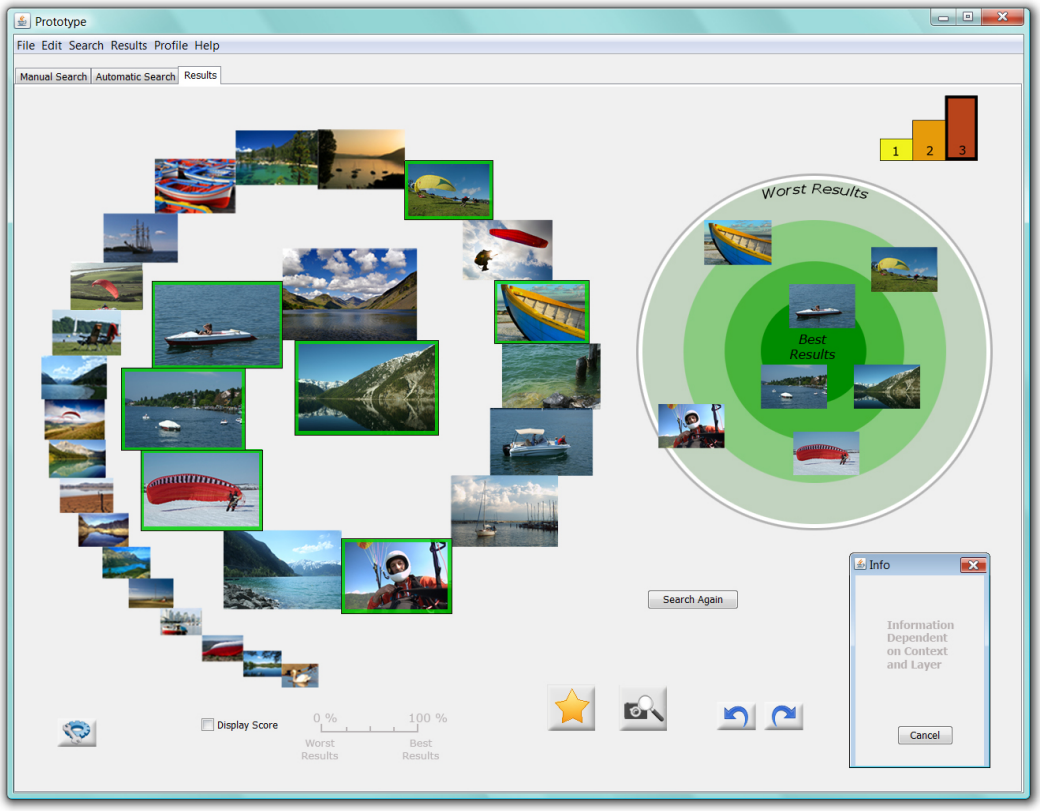


Abbildung 5.44: Ansicht des Tabs *Results* bei Verwendung konzentrischer Kreise für Relevance-Feedback

Auswertung Im Gegensatz zu den Klassen tritt hier eine größere Differenzierung auf. Es gibt zwei Varianten der Auswertung. Zum einen kann der Abstand des Mittelpunkts

(x_i, y_i) eines jeden Fotos o_i zum Mittelpunkt der konzentrischen Kreise (x_k, y_k) mithilfe der Euklidischen Distanz berechnet werden. $r_i = \sqrt{(x_i - x_k)^2 + (y_i - y_k)^2}$ entspricht somit dem Radius. Anschließend wird die Differenz der Radien eines jeden möglichen Fotopaares bestimmt. Weil der Nutzer nicht pixelgenau die Fotos platziert, ist zusätzlich ein Toleranzwert t nötig. Für $r_2 - r_1 \geq t$ gilt $o_1 \geq o_2$, für $r_2 - r_1 < -t$ gilt $o_1 < o_2$. Für eine Differenz aus dem Bereich $[-t, t]$ kann keines der beiden Fotos präferiert werden. Es ist nicht nötig, die Radien exakt zu bestimmen, es ist nur das Verhältnis der Radien zueinander wichtig. Das rechenintensive Wurzelziehen, um den Radius zu erhalten, kann entfallen, weil diese Funktion streng monoton ist.

Zum anderen können Präferenzen ermittelt werden über die Zuordnung eines Bildes zu einem der konzentrischen Kreise. Es exisitieren hier 4 Gruppen. Ein Foto gehört der Gruppe an, wo es den größten Flächenanteil einnimmt. Die Präferenzen existieren dann zwischen den Fotos in den unterschiedlichen Gruppen. Alle Fotos, die dem kleinsten Kreis angehören, werden als besser angesehen als die Fotos, die den äußeren Kreisen zugeordnet sind. Das Prinzip setzt sich dann für die anderen Gruppen analog fort.

Zwar können in der ersten Auswertungsmethode noch mehr Präferenzen abgeleitet werden als aus der letzten, jedoch entspricht die letzte Variante mehr dem Empfinden des Nutzers. Die meisten Anwender orientieren sich nicht an der Position der Mittelpunkte, sondern an dem Anteil, wie viel ein Bild eine Fläche bedeckt. Aus diesem Grund wird die Methode der Gruppeneinteilung durch Flächenbedeckung für das Ableiten der Präferenzen in den konzentrischen Kreisen verwendet.

Zyklen Wie auch in der Darstellung mit den Daumen werden bereits bewertete Bilder in der Ergebnismenge farblich hervorgehoben. Der Einsatz der gleichen Rahmenfarben wie die Farben der konzentrischen Kreise ist nicht ratsam, weil die unterschiedlichen Grüntöne auf Anhieb nur in direkter Nebeneinanderstellung unterschieden werden können. Möchte der Nutzer ein Bild mehr als einmal in der Zielscheibe platzieren, so wird wie auch im vorhergehenden Ansatz direkt nach dem Verschieben eine Warnung angezeigt. Der Anwender muss sich dann entscheiden, ob die neue oder die alte Position des Fotos übernommen werden soll.

Entscheidung für eine Darstellungsvariante

Bei beiden Methoden der Bewertung von Ergebnissen müssen die Fotos in den entsprechenden Bereich nach rechts verschoben werden. In der Darstellung mit den zwei Klassen, symbolisiert durch die Gesten Daumen hoch und Daumen runter, können die Ergebnisse nicht so differenziert durch den Nutzer bewertet werden wie bei den konzentrischen

Kreisen. Dort existieren 4 Gruppen. Es können somit bei der Zielscheibe deutlich mehr Präferenzen abgeleitet werden. Das bedeutet wiederum, dass die Anzahl der Iterationen, bis der Nutzer ein zufriedenstellendes Ergebnis hat, reduziert wird. Folglich soll die Methode mit den konzentrischen Kreisen in der GUI für das Relevance-Feedback umgesetzt werden.

Multi-Layer-Umsetzung im Results-Tab

Auf der untersten Ebene der Multi-Layer-Architektur sind die Ergebnisse in einer Spirale angeordnet. Das Bewerten der Ergebnisse durch Verschieben der Fotos in die Zielscheibe ist intuitiv und soll deshalb dem Anfänger bereits angeboten werden. Auf der zweiten Stufe stehen die Buttons für das Speichern von favorisierten Bildern und das Angeben eines Beispielbildes für eine Anfrage bereit. Das Zurücksetzen der Ansicht soll durch die Undo- und Redo-Buttons möglich sein. Um beim Darüberfahren mit der Maus über ein Foto nicht nur dessen vergrößerte Ansicht zu sehen, sondern zusätzlich Metadaten eingeblendet zu bekommen, steht der Button für die Konfigurationen in der zweiten Ebene bereit. Der Experte auf der dritten Ebene kann die exakten Score-Werte einsehen und Tastenkombinationen zur Mehrfachauswahl verwenden, um diese gleichzeitig in die Zielscheibe zu verschieben.

5.2 Diskussion der Prototypen

Bereits bei der Vorstellung der Entwürfe sind an entsprechender Stelle Vor- und Nachteile der verschiedenen Ansätze diskutiert worden. In diesem Abschnitt soll deshalb eine zusammenfassende Beurteilung der GUI gegeben werden. Es wird Bezug genommen auf die Konzepte, für welche sich im vorhergehenden Abschnitt zur Umsetzung entschieden wurde.

In der GUI sind die einzelnen Funktionalitäten durch Tabs voneinander getrennt. Browsing und Navigation befinden sich auf einem gemeinsamen Reiter, die anfragebasierte Suche und die Präsentation der berechneten Ergebnisse bezüglich der Anfrage auf jeweils einem eigenen. Durch diese Aufteilung ist die GUI überschaubar und die Ansicht auf jedem Tab einfach gehalten.

Anfragebasierte Suche

Durch die Umsetzung einer vielschichtigen Architektur erhalten Anfänger einen guten Einstieg in die Software. Das ist besonders bei der Suche mithilfe einer Anfrage von Vorteil. Nur leicht verständliche und für den Anfang benötigte Funktionen sind auf der untersten Ebene verfügbar wie das Eingeben eines Begriffs und der Angabe eines Beispielbildes. Das

Innovative der GUI ist die Kombination von *Query by Example* und *Query by Sketch* in einem Bild. Dieses wird dem fortgeschrittenen Anwender ermöglicht, weil Kenntnisse im Umgang mit den Werkzeugen nötig sind. Für Experten ist zusätzlich eine Segmentierung der Bilder möglich. Getestet werden muss, ob diese die Funktion auch verwenden.

Bedingungen werden für den Anfänger und Fortgeschrittenen automatisch mit einer Disjunktion verbunden. Um die Ansprüche des versierten Anwenders zu befriedigen, kann dieser eine Anfrage mittels eines Graphen aufstellen. Diese Variante ermöglicht ein schnelles Kombinieren der Bedingungen. Desweiteren können die Anfragegewichte zum einen automatisch, z.B. mithilfe von personalisierten Nutzerprofilen, und zum anderen vom Experten auch manuell über Regler gesetzt werden. Die Gewichte sind hierbei exakt angebbar. Der Graph bleibt zwar im Gegensatz zu den vorgestellten Textfeldern auch bei beispielsweise 8 Bedingungen übersichtlich, jedoch nimmt die Übersichtlichkeit mit zunehmender Bedingungsanzahl auch ab. Abhängig ist das unter anderem auch von dem Nutzer, welcher die Knoten setzt und die Kanten zwischen diesen zieht. Bei schlechter Positionierung verbundener Knoten können sich Kanten schneiden, sodass deren Beschriftung schlecht lesbar ist. Abhilfe kann die automatische Umsortierung mittels FDP schaffen. Ob FDP bei den durchschnittlichen Anfragen benötigt wird und inwieweit der Nutzer versteht, was nach dem Betätigen des dafür vorgesehenen Buttons geschieht, muss in Nutzertests untersucht werden. Desweiteren ist vorstellbar, dass es dem Nutzer schwerfallen könnte, die Prioritäten einer Kante anzugeben. Um eine Kante höher als eine andere zu priorisieren, muss der Prioritätswert der anderen Kante abgelesen werden. Außerdem kann der Fall eintreten, dass angrenzende Kanten gleich stark binden. Für die interne Auswertung wird dann zwischen diesen Kanten eine beliebige ausgewählt. Es muss überlegt werden, wie die letzten beiden Punkte eventuell noch verbessert werden könnten.

Aufgrund der Unterteilung des Tabs der anfragebasierten Suche in drei Bereiche mit jeweils einem eigenen Search-Button hat der Nutzer die Möglichkeit anhand von Metadaten oder anhand einer Skizze und eines Bildes oder anhand der Kombination beider Bereiche zu suchen. Negativ ist jedoch zu bewerten, dass die kombinierte Suche nur über den unteren Search-Button im Feld *Composition & Weighting* gestartet werden kann, auch wenn der Nutzer selbst keinen Anfragegraphen erstellt hat.

Browsen und Navigieren

Vorteilhaft für den Nutzer ist die Kombination von Browsen anhand von Farben und Navigieren innerhalb Klassifikation, welche das Abgebildete der Fotos spezifiziert. Dieses ist mit einer Liste und hierarchisch wachsenden SOMs realisiert.

Die Listendarstellung ist aus sehr vielen Anwendungen bekannt und stellt somit keine

Hürde bei der Interaktion dar. Zu erwähnen ist jedoch der Aufwand für die Erstellung der Klassifikation und die Einordnung der Fotos in diese. Zwar können die bereits bestehenden Annotationen der Bilder genutzt werden, jedoch liegen diese meist nicht genormt vor und erschweren somit den Prozess der Kategorisierung. Clusterverfahren im Information-Retrieval können hierbei die manuelle Arbeit reduzieren. Die Fotos mit den Begriffen, die in denselbem Cluster liegen wie der Kategoriename, können dieser Kategorie zugeordnet werden.

Die Sortierung der Bilder anhand ihrer Farbähnlichkeit ermöglicht dem Nutzer eine bessere Orientierung innerhalb der Bilddatenbank als das sequentielle Suchen. Farbhistogramme, wie Vergleiche verschiedener Features zeigen [DKN08], sind derzeit das geeigneteste Feature für die Sortierung der SOMs. Die Wahl von τ_1 und τ_1 zur Festlegung der Granularität ist entscheidend wie flach oder wie tief die Karten werden können. Verschiedene Wertekombinationen sollten in Tests untersucht werden, um einen guten Kompromiss zu finden zwischen der Anzahl der abgebildeten Fotos auf einer Karte und der Anzahl der SOM-Ebenen. Es gilt außerdem zu klären, ob der Nutzer den Wechsel zwischen den verschiedenen SOM-Ebenen versteht und die Orientierung innerhalb des Gesamten nicht verliert. Eine Ansicht, in welcher SOM-Ebene man sich momentan befindet, kann hier eine Verbesserung darstellen.

Ergebnispräsentation

Hat der Nutzer die Suche nach Fotos gestartet, die seiner Anfrage entsprechen, so werden die Ergebnisse in einem neuen Tab angezeigt. Bei der Fermatschen Spirale werden Präferenzen über die Lage in der Spirale deutlich. Bei zwei angrenzenden Bildern wird dasjenige vom System präferiert, welches den geringen Radius hat. Desweiteren wird ein Foto größer dargestellt, je höher sein Score-Wert ist. Zwar werden die relevanteren Fotos durch diese beiden Festlegungen zuerst vom Nutzer angeschaut, jedoch kann der Nutzer aufgrund der angrenzenden Bilder auch annehmen, dass alle Top-k-Fotos dargestellt sind und nicht nur die charakteristischen Präferenzen. Es ist nicht ersichtlich, dass einige Fotos der besten k fehlen.

Beim modifizierten Hasse-Diagramm ist die charakteristische Präferenzmenge P besser erkennbar. Das Foto o_i der Präferenz $o_i \geq o_{i+1}$ wird links von o_{i+1} angezeigt. Bilder in einer neuen Zeile bedeuten, dass ein oder mehrere Fotos nicht angezeigt werden, weil diese nicht in P existieren. Durch die Aufteilung der Fotos nach diesem Schema in Zeilen kann der Nutzer besser erkennen, dass nicht alle Fotos der besten k angezeigt werden. Problematisch wird es, wenn viele Zeilen benötigt werden und Scrollen nötig wird. Dadurch hat der Nutzer nicht mehr alle Fotos im Blick.

Die Polygondarstellung weist andere Vorteile auf. Dort sind die Präferenzen sofort anhand der Stabhöhe der Bilder, die in P liegen, in der Seitenansicht erkennbar. Positiv zu bewerten ist die Draufsicht, wo die Lage eines Fotos durch die Erfüllbarkeit der einzelnen Bedingungen der Anfrage bestimmt wird. Die Interpretation ist jedoch nicht eindeutig möglich. Um das Problem zu beseitigen, wird nach Auswahl eines Fotos für dieses die Erfüllung einer jeden Bedingung durch rote Linien aufgezeigt. Je mehr Bedingungen der Nutzer angibt, desto mehr Ecken besitzt das Polygon. Anwendung könnte aus diesem Grund die KLT finden. Es besteht jedoch dann das Problem der Benennung der Achsen. Desweiteren tritt das Problem der Anhäufung von Bildern in bestimmten Gebieten auf, wodurch einige Fotos von anderen überdeckt werden. Eine Lösung wäre die Bereitstellung von weiteren Polygonen, welche Ergebnisobjekte aus einem bestimmten Score-Wert-Bereich aufnehmen.

Jedes dieser drei Varianten zur Ergebnispräsentation hat sowohl positive als auch negative Aspekte. Es ist deshalb auch vorstellbar, anstatt der Spirale eine der beiden anderen Entwürfe für die Darstellung der Fotos zu verwenden. Der Nutzer kann die angezeigten Fotos bewerten. Das Prinzip der Zielscheibe ist leicht verständlich. Der Bereich hierfür ist groß genug, um mehr als 10 Bewertungen vornehmen zu können. Aufgrund der 4 entstehenden Gruppen ist es wahrscheinlicher, dass die Anzahl der Präferenzen bis zum Erreichen des Zielranks geringer ist als bei der Verwendung von nur zwei Klassen. Wie viele Bewertungen der Nutzer pro Iterationsschritt letztendlich angibt, muss getestet werden.

Die goldenen Regeln des Dialog-Designs

Die 1. Regel aus 4.1.3 besagt, dass die Benutzeroberfläche konsistent sein soll. Bei der anfragebasierten Suche bestehen alle Buttons für das Hinzufügen aus einem Plus-Symbol. Zusätzlich ist auf diesen erkennbar, was hinzugefügt werden soll. Das Gleiche gilt für den Entfernen-Button. Die Idiome für den Wechsel zwischen den Informationsebenen, beispielweise von der Übersicht in die Darstellung der Details eines Fotos, sind sowohl beim Browsen und Navigieren als auch bei der Ergebnispräsentation die gleichen.

Die 5. Regel fordert eine einfache Fehlerbehandlung. Das ist beispielsweise umgesetzt beim Relevance-Feedback, wenn der Nutzer ein Foto zweimal an unterschiedlichen Positionen in der Zielscheibe einordnen will. Auch beim Auftreten von Zyklen in der Graphendarstellung wird der Nutzer auf den Widerspruch verständlich hingewiesen. Die Behebung der Fehler ist aufgrund von Systemvorschlägen einfach durch den Nutzer vorzunehmen. Bei der Angabe von bestimmten Metadaten in einer Anfrage werden Fehleingaben verhindert, indem Felder nur Daten aus einem bestimmten Wertebereich zulassen. Rechtschreibfehler, z.B. bei der Angabe eines Titels, werden gemindert, indem Konzepte des Information-

Retrievals verwendet werden, um auch Bilder zu ähnlichen Begriffen zu suchen. Das sind nur einige Beispiele hierfür.

Die Forderung der 6. Regel nach reversiblen Eingaben wird erfüllt, indem der Nutzer zum einen einzelne Aktionen wie das Zeichnen mit dem Bleistift zurücknehmen kann, zum anderen die Ansicht auf frühere Präferenzen zurücksetzen kann. Neben den Menüeinträgen und Tastenkombinationen für das Rückgängigmachen von einzelnen Operationen existieren Buttons unterhalb der Zielscheibe für die Undo- und die Redo-Funktion bezüglich des Relevance-Feedbacks.

Der Benutzer soll laut der 7. Regel selbst die Kontrolle über das Programm besitzen. Wenn der Anwender eine Anfrage manuell aufstellen will, so kann er alle Knoten durch Klick auf einen Button erzeugen lassen. Auch für die Umsortierung dieser mittels FDP wird erst nach Betätigung des zugehörigen Buttons die Aktion ausgelöst. Der Nutzer versteht somit die Reaktion des Systems. An dieser Stelle sei auch auf die Erfüllung der anderen Regeln durch die GUI hingewiesen.

Nutzertests

Die GUI enthält wenige Metaphern wie die Werkzeuge beim Skizzieren eines Anfragebildes. Sie ist, wie gefordert, hauptsächlich eine idiomatische Schnittstelle. Einfache Idiome stehen bereit. Die Bewertung der Nutzerschnittstelle ist jedoch sehr subjektiv. Aus diesem Grund ist es notwendig die ausgewählten Entwürfe zu implementieren, um die Prototypen von Nutzern testen zu lassen. Während verschiedener Suchszenarien von Probanden werden die Maus- und Tastaturaktionen aufgezeichnet. Neben diesen kann auch die Blickverfolgung, sogenanntes Eye-Tracking, Aufschluss über Probleme geben. Außerdem kann die subjektive Meinung der Testkandidaten zur Benutzerfreundlichkeit durch Fragebögen zum Ausdruck kommen. Die Ergebnisse der Tests sollten beispielsweise klären, ob der Nutzer schnell und unkompliziert eine gewünschte Skizze erstellen und diese mit einem Foto kombinieren kann und wie leicht für eine bestimmte Anfrage ein Graph erstellt werden kann. Die Idiome für die Navigation innerhalb der SOMs und der Ergebnispräsentation sowie für den Wechsel zwischen den Informationsebenen müssen in ihrer Benutzerfreundlichkeit untersucht werden. Wichtig ist auch, dass der Nutzer das Prinzip der Multi-Layer-Architektur versteht und weiß, dass er zum einen auomatisch auf Vorschlag des Systems und zum anderen auch manuell auf eine höhere Stufe wechseln kann. Desweiteren soll festgestellt werden, wie der Nutzer die Möglichkeit der Angabe von eigenen Bewertungen der Ergebnisse mithilfe der Zielscheibe annimmt.

Im Großen und Ganzen kann das Resümee gezogen werden, dass die vorgestellten Entwürfe für eine grafische Benutzeroberfläche für eine inhaltsbasierte Bildsuche die Anforde-

rungen aus dem vorhergehenden Kapitel größtenteils erfüllen.

6 Integrationsmöglichkeit in vorliegende Arbeiten am Lehrstuhl

Am Lehrstuhl Datenbank- und Informationssysteme wird für die Suche von ähnlichen Multimedia-Objekten das Similarity-Query-System (SQ-System) entwickelt, welches sich den Konzepten von CQQL bedient. Im Folgenden soll die Funktionalität dieses Systems anhand einer Skizze erklärt werden, die bereits in etwas anderer Form im Kapitel 2 eingesetzt wurde.

Sebastian Zech hat in seiner Bachelorarbeit das LIRE-Framework zur MPEG-7-basierten Feature-Extraktion untersucht. LIRE ist ein von Mathias Lux entwickeltes CBIRS, dass auf den Programmen Caliph und Emir basiert, welche in dieser Arbeit unter 3.4 vorgestellt wurden. Für die Feature-Extraktion, welche im Schema 6.1 den zweiten Schritt darstellt, soll für das SQ-System die frei verfügbare LIRE-Bibliothek verwendet werden. Die verschiedenen Features, welche sowohl einzeln als auch aggregiert vorliegen, sollen dann in einer XML-Datei abgespeichert werden.

In der vorliegenden Bachelorarbeit sind im Kapitel 5 Prototypen für die grafische Benutzeroberfläche zur inhaltsbasierten Suche von Bildern vorgestellt worden. Die Entwürfe kommen den Anforderungen der Anfragesprache CQQL an eine GUI nach. Die Einordnung der Konzepte in das allgemeine Schema eines MMRS ist in der Abbildung 6.1 als dicker Rahmen zu erkennen. Sowohl für die Formulierung von Anfragen als auch für das Browsing und Navigieren sind verschiedene Ansätze diskutiert worden. Der Nutzer kann die herkömmlichen Datenbankanfragen mit den Ähnlichkeitsanfragen kombinieren. Hierfür wird auf die XML-Datei zugegriffen, um beispielweise die zur Verfügung stehenden Metadaten als Bedingung auswählen zu können oder die Bilder anhand ihrer Farbe in einer SOM dem Nutzer zu präsentieren.

Unterschiedliche Varianten für die Darstellung der Ergebnisse, genauer der charakteristischen Präferenzen, wurden in dieser Arbeit aufgezeigt. Der Nutzer kann auch diese im Zuge des Relevance-Feedbacks bewerten, um eine seinem Ähnlichkeitsempfinden besser angepasste Ergebnismenge zurückgeliefert zu bekommen. Am Lehrstuhl existiert bereits eine Implementierung des Lernalgorithmus, der die Gewichte aus den Nutzerpräferenzen

ermittelt. Diese wurde für das Kamerabeispiel aus [SZ09] eingesetzt. Für das SQ-System muss in Zukunft eine CQQL-Funktion dynamisch aufgestellt werden, die abhängig ist von den Benutzereingaben. Bisher sind die Algorithmen auf das Beispiel mit der Kamera zugeschnitten. Andere als die bisher genannten Bestandteile bzw. Funktionalitäten des SQ-Systems sind derzeit oder zukünftig in Entwicklung. Ziel ist es, in Bezug auf diese Arbeit, die besten hier vorgestellten Prototypen für die GUI des SQ-Systems umzusetzen.

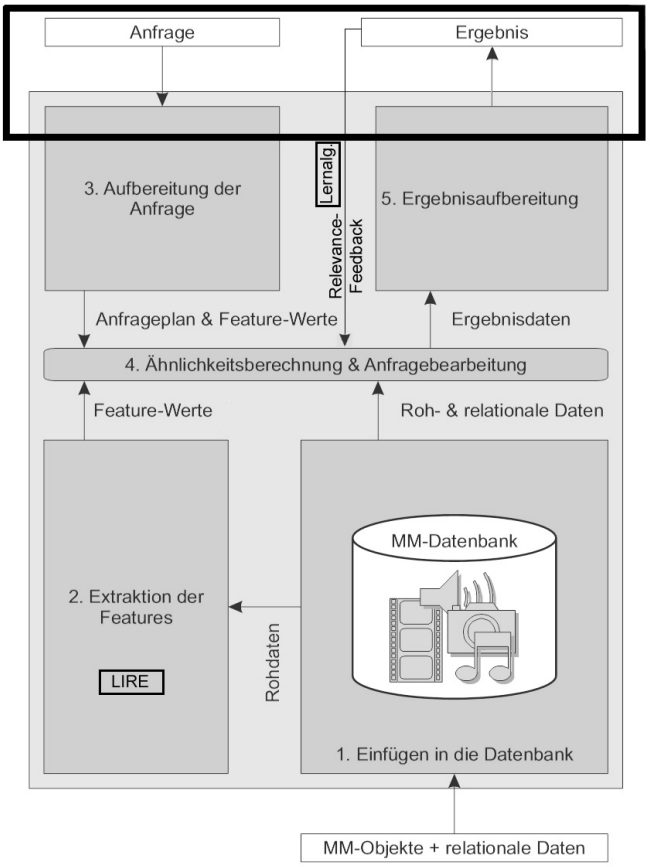


Abbildung 6.1: Ablauf des Multimedia-Retrievals im SQ-System, nach [Sch06]

7 Zusammenfassung und Ausblick

In der vorliegenden Arbeit wurden Prototypen für die grafische Benutzeroberfläche zur inhaltsbasierten Suche von Bildern auf Grundlage von CQQL entworfen und evaluiert.

Zuerst wurde im Kapitel 2 der Begriff Multimedia geklärt. Die Unterschiede zwischen Daten- und Information-Retrieval wurden aufgezeigt und der generelle Suchablauf in einem MMRS beschrieben. Hervorzuheben ist, dass aus den Rohdaten der Multimedia-Objekte die Low-Level-Features extrahiert werden, um eine Suche anhand des Abgebildeten zu ermöglichen. Unter High-Level-Features sind die relationalen Daten zu verstehen. Zwischen beiden Arten von Features besteht die semantische Lücke, die es gilt in Zukunft zu überwinden. Im Anschluss wurde die Anfragesprache CQQL vorgestellt. Diese erlaubt die Formulierung von logikbasierten Anfragen, wobei Boolesche und Score-basierte Bedingungen kombiniert werden können. Desweiteren wird eine Gewichtung der Operanden vorgenommen, um das subjektive Ähnlichkeitsempfinden in die Anfrage einfließen zu lassen. Der Nutzer gibt im Rahmen des Relevance-Feedbacks seine Präferenzen auf den Systempräferenzen an, wodurch die Gewichte entsprechend modifiziert werden. Dieser Prozess wird iterativ solange fortgesetzt bis der Nutzer zufrieden ist mit dem Ergebnis oder die Suche abbricht.

Im Kapitel 3 wurden Benutzeroberflächen von einigen CBIRS vorgestellt und bewertet. Zusätzlich wurde ein System zur Suche von Videos untersucht. Die Suchmöglichkeiten, die Präsentation der Ergebnisse und das Relevance-Feedback waren die Kriterien bei der Evaluierung. QBIC in der Verwendung vom State Hermitage Museum bietet bereits eine intuitive Bedienung für das Zeichnen einer Skizze. Bei anderen Systemen, außer VideoQ, fehlt diese Funktionalität. Keines der getesteten Systeme hat die Möglichkeit angeboten eine Anfrage als Kombination von exakten Bedingungen, Query by Sketch und Query by Example aufzustellen. Ergebnisse sind immer als Liste präsentiert worden. Desweiteren wird nur in 2 der 5 Systeme Relevance-Feedback eingesetzt. Die Bewertung der Ergebnisse durch den Nutzer ist bei diesen jedoch aufwendig.

Die Anforderungen an die GUI für die inhaltsbasierte Suche von Fotos wurden im Kapitel 4 zusammengetragen. Die Zielgruppe sind Anwender, die sich nicht wissenschaftlich mit der Thematik des Multimedia-Retrievals auseinandersetzen. Die Benutzeroberfläche

soll idiomatisch sein und eine einfache Bedienbarkeit aufweisen. Die 8 goldenen Regeln des Dialog-Designs sollen erfüllt werden. Zu diesen zählt beispielsweise die Forderung nach Konsistenz und reversiblen Eingaben. Um die GUI für Anfänger einfach zu halten, um einen leichten Einstieg in die Benutzung der Software zu bekommen, und gleichzeitig den fortgeschrittenen und versierten Anwendern mehr Möglichkeiten zum Interagieren und zum Spezifieren ihrer Anfrage zu bieten, soll eine Multi-Layer-Architektur eingesetzt werden. Die GUI soll im Gegensatz zu den im vorhergehenden Kapitel bewerteten Benutzeroberflächen die Möglichkeit anbieten, eine Anfrage zu erstellen, die Metadaten, eine Skizze und ein Beispielbild für die Suche von Bildern miteinander kombiniert. Desweiteren soll auch Browsen und Navigieren möglich sein. Es sollen Entwürfe für die Ergebnispräsentation vorgestellt werden, welche das Ablesen der charakteristischen Präferenzen erlauben und zusätzliche Informationen zum Foto abrufbar machen. Nutzer sollen in wenigen Schritten ihre Bewertungen abgeben können, wobei Methoden zum Ableiten von Präferenzpaaren für die Berechnung der Gewichte präsentiert werden sollen.

Im Kapitel 5 wurden auf den Anforderungen basierend verschiedene Entwürfe für die GUI präsentiert und bewertet. Hierzu zählt eine Treppe, welche die Ebene anzeigt, in der sich der Nutzer momentan in der vielschichtigen Benutzeroberfläche befindet. Navigation und Browsing ist in nach Farben sortierten GHSOMs möglich, wobei eine Filterung nach abgebildeten Themen über eine Liste erfolgt. Tree-Maps haben sich hierfür nicht durchgesetzt. Bei einer konkreten Vorstellung von dem zu suchenden Bild können die klassischen Datenbankbedingungen, welche in Textfeldern eingegeben werden, mit bis zu drei Beispielbildern und integrierter Skizze in einer Anfrage verknüpft werden. Für die Erstellung einer Anfrage mit manuell gesetzten Gewichten hat sich die Graphendarstellung gegenüber den verschiebbaren Textfeldern und den Venn-Diagrammen als die bessere Variante herausgestellt. Für die Ergebnispräsentation wurden ein modifiziertes Hasse-Diagramm, eine Spirale und die Polygondarstellung verglichen, wobei alle ihre Stärken und Schwächen in unterschiedlichen Aspekten aufweisen. Beim Relevance-Feedback fiel die Entscheidung nicht auf die zwei Klassen mit den Handgestiken, sondern auf die konzentrischen Kreise, wo eine Bewertung noch differenzierter vorgenommen werden kann. Die Diskussion der Prototypen hat ergeben, dass sehr viele der Anforderungen erfüllt wurden, jedoch noch einige wenige Verbesserungen möglich wären.

Da die bisherigen Bewertungen der Entwürfe subjektiv ausfallen, ist es wichtig die besten Protoytpen umzusetzen, um Probleme und Verbesserungsvorschläge durch Nutzertests herauszufinden. Die Erkenntnisse sollen in die Weiterentwicklung der Benutzeroberfläche fließen. Die Beurteilung der GUI durch Probanden ist jedoch nur dann sinnvoll, wenn die zugrundeliegende Architektur fertiggestellt ist. Die GUI kann, wie im Kapitel 6 beschrie-

ben, in das SQ-System integriert werden. Dieses befindet sich derzeit jedoch in Entwicklung.

Neben der Suche von Bildern ist für zukünftige Projekte die Weiterentwicklung der GUI um die inhaltsbasierte Suche von Videos erstrebenswert. Wie auch bei Fotos können bei Videos Query by Example und Query by Sketch zur Anfrageformulierung Einsatz finden. Aufgrund der Zeitabhängkeit der Videos müssten weitere Werkzeuge bereitgestellt werden, mit denen sowohl die Objekt- als auch die Kamerabewegungen beschrieben werden können. Vorstellbar ist die Angabe eines Vektors, der die Richtung und die Geschwindigkeit vorgibt. Eine andere Variante wäre das Zeichnen des Anfangs- und Endbildes eines gesuchten Videoausschnitts. Der Nutzer kann die Bilder zwischen diesen vom System interpolieren lassen, wobei neben Translation und Rotation auch Skalierung und Deformation Beachtung finden müssen. Für das Selektieren einer bestimmten Szene eines existierenden Beispielvideos für die Anfrage müssen Abspielfunktionen bereitgestellt werden und Markierungen gesetzt werden. Neben den bisherigen Angaben sollte ein Video auch anhand von Audioinformationen gesucht werden können. Das Eingeben einer beispielhaften Audiodatei oder die Aufnahme von gesummten Melodien und Instrumentalstücken über ein Mikrofon sind einige Varianten. Neben der Aufstellung einer Suchanfrage muss auch die Ergebnispräsentation der GUI angepasst werden. Viele Videos in einer gemeinsamen Ansicht könnten beispielsweise durch charakteristische Keyframes angezeigt werden. Ein möglicher Ansatz ist in [BUSB08] zu finden, wo die Keyframes eines Videosegments geclustert werden. Das Bild, welches die geringste Distanz zum Mittelpunkt eines Clusters hat, wird als Repräsentant angezeigt. Eine Kameraeinstellung kann somit aus mehreren Keyframes existieren, wobei diese beispielsweise räumlich nebeneinander oder zeitlich gemultiplext an der gleichen Position angezeigt werden. Die Ansätze gilt es in Zukunft zu untersuchen und Entwürfe zu entwickeln.

Abkürzungsverzeichnis

CALIPH ... Common And Light-Weight Photo Annotation

CBIRS ..... Content-Based-Image-Retrieval-System

CBIVRS ... Content-Based-Image and Video-Retrieval-System

EMIR ..... Experimental Metadata Based Image Retrieval

EXIF ..... Exchangeable Image File Format

FDP ..... Force Directed Placement

FIRE ..... Flexible Image Retrieval Engine

GHSOM ... Growing-Hierarchical-Self-Organizing-Map

GIFT ..... The GNU Image Finding Tool

GUI ..... Graphical User Interface

HSOM ..... Hierarchical-Self-Organizing-Map

IPTC IIM .. International Press Telecommunications Council - Information Interchange Model

IR ..... Information-Retrieval

IRS ..... Information-Retrieval-System

KLT ..... Karhunen-Loeve-Transformation

MMR ..... Multimedia-Retrieval

MMRS ..... Multimedia-Retrieval-System

MPEG ..... Moving Picture Experts Group

QBE ..... Query by Example

QBIC ..... Query by Image Content

SOM ..... Self-Organizing-Map

SQL ..... Structured Query Language

VIPER .... Visual Information Processing for Enhanced Retrieval

VOCR ..... Video Optical Character Recognition

XML ..... Extensible Markup Language

XMP ..... Extensible Metadata Platform

Abbildungsverzeichnis

Literaturverzeichnis

[ADV97a] ADVENT Project: *About VideoQ*. Version: 1997. `http://persia.ee.columbia.edu:8080/about.html`, Abruf: 27. 06. 2009

[ADV97b] ADVENT Project: *Getting Started*. Version: 1997. `http://persia.ee.columbia.edu:8080/start.html`, Abruf: 27. 06. 2009. – Videosuche mit VideoQ

[ADV97c] ADVENT Project: *VideoQ - An object oriented video search engine*. Version: 1997. `http://persia.ee.columbia.edu:8080/.index.html`, Abruf: 27. 06. 2009

[ADV97d] ADVENT Project: *Visual Search*. Version: 1997. `http://persia.ee.columbia.edu:8080/visual.html`, Abruf: 27. 06. 2009. – visuelle Videosuche mit VideoQ

[Arn06] Arndt, Henrik: *Integrierte Informationsarchitektur: Die erfolgreiche Konzeption professioneller Websites (X.media.press)*. Berlin, Heidelberg : Springer, 2006. – ISBN 3540240748

[Axi09] Axialis Software: *Realistic Buttons*. Version: 2009. `http://www.axialis.com/objects/ip_icon_02.shtml`, Abruf: 13. 07. 2009

[Bed01] Bederson, Benjamin B.: PhotoMesa: a zoomable image browser using quantum treemaps and bubblemaps. In: *UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology*. New York, NY, USA : ACM, 2001. – ISBN 1–58113–438–X, S. 71–80

[BRGF08] Barthel, Kai U. ; Richter, Sebastian ; Goyal, Anuj ; Follmann, Andreas: Improved Image Retrieval Using Visual Sorting and Semi-Automatic Semantic Categorization of Images, 2008, S. 100

[BSW02] Bederson, Benjamin B. ; Shneiderman, Ben ; Wattenberg, Martin: Ordered and quantum treemaps: Making effective use of 2D space to display hierarchies. In: *ACM Trans. Graph.* 21 (2002), Nr. 4, S. 833–854

[BUSB08] Borth, Damian ; Ulges, Adrian ; Schulze, Christian ; Breuel, Thomas M.: Keyframe Extraction for Video Tagging & Summarization. In: *Informatiktage 2008*, Gesellschaft für Informatik (GI), 2008, S. 45–48

[BVBF07] Blanken, Henk ; Vries, Arjen P. ; Blok, Henk E. ; Feng, Ling: *Multimedia Retrieval (Data-Centric Systems and Applications)*. Berlin, Heidelberg : Springer-Verlag, 2007

[Cal03] Callan, Robert: *Neuronale Netze im Klartext*. München : Pearson Studium, 2003. – ISBN 3–8273–7071–X

[CC84] Carroll, John M. ; Carrithers, Caroline: Training Wheels in a User Interface. In: *Communications of the ACM* 27 (1984), Nr. 8, S. 800–806

[CCM+97] Chang, Shih-Fu ; Chen, William ; Meng, Horace J. ; Sundaram, Hari ; Zhong, Di: VideoQ: an automated content based video search system using visual cues. In: *MULTIMEDIA '97: Proceedings of the fifth ACM international conference on Multimedia*. New York, NY, USA : ACM, 1997. – ISBN 0–89791–991–2, S. 313–324

[Cha09] Chabloz, Nicolas: *Viper - Image Retrieval Test - Demo*. Version: 2009. `http://dolphin.unige.ch/gift/demo.php`, Abruf: 20. 06. 2009

[Che04] Chen, Chaomei: *Information Visualization: Beyond the Horizon (2nd Edition)*. London, England : Springer, 2004. – ISBN 1–85233–789–3

[CLB+02] Ceaparu, Irina ; Lazar, Jonathan ; Bessiere, Katie ; Robinson, John ; Shneiderman, Ben: Determining Causes and Severity of End-User Frustration. In: *International Journal of Human-Computer Interaction* 17 (2002), S. 333–356

[CRC07] Cooper, Alan ; Reimann, Robert ; Cronin, Dave: *About Face 3: The Essentials of Interaction Design*. Indianapolis, Indiana, USA : Wiley Publishing, Inc., 2007. – ISBN 978–0–470–08411–3

[Des05] Deselaers, Thomas: *FIRE - Flexible Image Retrieval Engine*. Version: 2005. `http://www-i6.informatik.rwth-aachen.de/~deselaers/cgi_bin/fire.cgi`, Abruf: 20. 06. 2009

[Des09] Deselaers, Thomas: *Homepage of Thomas Deselaers*. Version: 2009. `http://thomas.deselaers.de/FIRE`, Abruf: 20. 06. 2009

[DFAB03] Dix, Alan ; Finlay, Janet E. ; Abowd, Gregory D. ; Beale, Russell: *Human-Computer Interaction (3rd Edition)*. Upper Saddle River, NJ, USA : Prentice-Hall, Inc., 2003. – ISBN 0–13–046109–1

[DGGO08] Dresky, Caroline von ; Gasser, Ingenuin ; Günzel, Silke ; Ortlieb, Claus P.: *Mathematische Modellierung: Eine Einführung in zwölf Fallstudien*. Wiesbaden : Vieweg+Teubner, 2008. – ISBN 3–83–510252–4

[DKN08] Deselaers, Thomas ; Keysers, Daniel ; Ney, Hermann: Features for Image Retrieval: An Experimental Comparison. In: *Information Retrieval* 11 (2008), Nr. 2, 77-107. http://www-i6.informatik.rwth-aachen.de/publications/downloader.php?id=516&row=pdf

[DMR01] Dittenbach, Michael ; Merkl, Dieter ; Rauber, Andreas: Recent advances with the growing hierarchical self-organizing map. In: *Proceedings of the 3rd Workshop on Self-Organizing Maps*. Lincoln, England, 2001, S. 140–145

[DZP04] Deng, Da ; Zhang, Jianhua ; Purvis, Martin: Visualisation and comparison of image collections based on self-organised maps. In: *ACSW Frontiers '04: Proceedings of the second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalisation*. Darlinghurst, Australia, Australia : Australian Computer Society, Inc., 2004, S. 97–102

[Eur95] Europäisches Komitee für Normung: *Deutsche Fassung, Ergonomische Anforderungen für Bürotätigkeiten mit Bildschirmgeräten, Teil 10: Grundsätze der Dialoggestaltung (ISO 9241-10: 1995)*. Version: 1995. http://www.interactive-quality.de/site/DE/int/pdf/ISO_9241-10.pdf, Abruf: 12. 07. 2009

[FL95] Faloutsos, Christos ; Lin, King-Ip: FastMap: a fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In: *SIGMOD '95: Proceedings of the 1995 ACM SIGMOD international conference on Management of data*. New York, NY, USA : ACM, 1995. – ISBN 0–89791–731–6, S. 163–174

[Fli09] Flickr: *Erweiterte Suche*. Version: 2009. http://www.flickr.com/search/advanced/, Abruf: 06. 05. 2009

[FR91] Fruchterman, Thomas M. J. ; Reingold, Edward M.: Graph drawing by force-directed placement. In: *Software: Practice and Experience* 21 (1991), Nr. 11, S. 1129–1164

[Fre05] Free Software Foundation: *The GNU Image-Finding Tool*. Version: 2005. http://www.gnu.org/software/gift/, Abruf: 20. 06. 2009

[FSN+95] Flickner, Myron ; Sawhney, Harpreet ; Niblack, Wayne ; Ashley, Jonathan ; Huang, Qian ; Dom, Byron ; Gorkani, Monika ; Hafner, Jim ; Lee, Denis ; Petkovic, Dragutin ; Steele, David ; Yanker, Peter: Query by Image and Video Content: The QBIC System. In: *Computer* 28 (1995), Nr. 9, S. 23–32. – ISSN 0018–9162

[Fuk90] Fukunaga, K.: *Introduction to Statistical Pattern Recognition (2nd Edition)*. New York : Academic Press, 1990

[FW00] Fagin, Ronald ; Wimmers, Edward L.: A formula for incorporating weights into scoring rules. In: *Theor. Comput. Sci.* 239 (2000), Nr. 2, S. 309–338

[Gal07] Galitz, Wilbert O.: *The Essential Guide to User Interface Design: An introduction to GUI Design Principles and Techniques.* 3rd. Wiley Publishing, Inc., 2007

[Hag04] Haggarty, Rod: *Diskrete Mathematik für Informatiker.* München : Pearson Studium, 2004

[Hen08] Henrich, Andreas: *Information Retrieval 1. Grundlagen, Modelle und Anwendungen.* Otto-Friedrich-Universität Bamberg : Creative Commons License, 2008 `http://www.uni-bamberg.de/fileadmin/uni/fakultaeten/wiai_lehrstuehle/medieninformatik/Dateien/Publikationen/2008/henrich-ir1-1.2.pdf`

[Her06] Herczeg, Michael: *Interaktionsdesign - Gestaltung interaktiver und multimedialer Systeme.* München : Oldenbourg Verlag, 2006

[HJN03] Hauptmann, Alexander G. ; Jin, Rong ; Ng, Tobun D.: Video Retrieval Using Speech and Image Information. In: *In Storage and Retrieval for Multimedia Databases 2003, EI'03 Electronic Imaging.* Santa Clara, CA : Publications, 2003, S. 148–159

[IBM09] IBM Research: *QBIC – Query by Image Content.* Version: 2009. `http://domino.research.ibm.com/comm/pr.nsf/pages/rsc.qbic.html`, Abruf: 14. 06. 2009

[Int09] International Press Telecommunications Council: *IIM Information Interchange Model.* Version: 2009. `http://www.iptc.org/cms/site/index.html;jsessionid=aaMPpsDlrXff?channe%l=CH0108`, Abruf: 22. 06. 2009

[JA09a] Jaeckel, Stephan ; Amboni, Sergej: *Archimedische Spirale.* Version: 2009. `http://www.mathe.tu-freiberg.de/~hebisch/spiralen/s_archimed.html`, Abruf: 27. 08. 2009

[JA09b] Jaeckel, Stephan ; Amboni, Sergej: *Fermatsche Spirale.* Version: 2009. `http://www.mathe.tu-freiberg.de/~hebisch/spiralen/s_fermat.html`, Abruf: 27. 08. 2009

[Koh82] Kohonen, Teuvo: Self-organized formation of topologically correct feature maps. In: *Biological Cybernetics* 43 (1982), January, Nr. 1, S. 59–69

[Koh01] Kohonen, Teuvo: *Self-organizing maps (3rd Edition).* Berlin, Heidelberg, New York : Springer, 2001. – ISBN 3–540–67921–9

[Kos09] Kosch, Harald: Content-Based Image Retrieval Systems - Reviewing and Benchmarking. In: *Proceedings of the 9th Workshop on Multimedia Metadata (WMM'09) held in conjunction with the 13th French Multimedia Conference on Compression and Representation of Audiovisual Signals (CORESA 2009)*, 2009

[Käs05] Käster, Thomas: *Intelligente Bildersuche durch den Einsatz inhaltsbasierter Techniken*, Universität Bielefeld, Technische Fakultät, Dissertation, 2005. http://bieson.ub.uni-bielefeld.de/volltexte/2005/713/

[Kud07] Kudrass, Thomas: *Taschenbuch Datenbanken.* München : Fachbuchverlag Leipzig im Carl-Hanser-Verlag, 2007

[LG05] Lux, Mathias ; Granitzer, Michael: Retrieval of MPEG-7 based Semantic Descriptions. In: *BTW-Workshop WebDB Meets IR in context of the 11. GI-Fachtagung für Datenbanksysteme in Business, Technologie und Web.* Karlsruhe, März 2005

[Lux05a] Lux, Mathias: *Flashdemo of Emir.* Version: 2005. http://www.semanticmetadata.net/demo/EmirDemo.htm, Abruf: 21. 06. 2009

[Lux05b] Lux, Mathias: *Retrieval und Annotation von digitalen Photos mit MPEG-7.* Version: 2005. http://www.semanticmetadata.net/publications/mlux-mp7ws-uni-klu-2005-s%lides.pdf, Abruf: 22. 06. 2009

[Lux07] Lux, Mathias: *SemanticMetadata.net.* Version: 2004-2007. http://www.semanticmetadata.net/features/, Abruf: 22. 06. 2009

[Man02] Mann, Thomas M.: *Visualization of Search Results from the World Wide Web*, Universität Konstanz, Fachbereich Informatik und Informationswissenschaft, Dissertation, 2002

[Mar09] Maryland Limited Liability Corporation: *Clker.com - Traffic Light.* Version: 2009. http://www.clker.com/clipart-2840.html, Abruf: 13. 07. 2009

[Mes94] Messinger, Heinz: *Langenscheidts Handwörterbuch Englisch.* 6. Berlin, München : Langenscheidt KG, 1994

[Mil56] Miller, George A.: The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information. In: *The Psychological Review* 63 (1956), 81–97. http://users.ecs.soton.ac.uk/~harnad/Papers/Py104/Miller/miller.html

[MSS02] Manjunath, B. S. ; Salembier, Phillipe ; Sikora, Thomas: *Introduction to MPEG 7: Multimedia Content Description Interface.* West Sussex, England : John Wiley and Sons, Ltd, 2002

[Naj03] Najarro, Martha Argentina B.: *Visualisierung von Informationsräumen*, Technische Universität Ilmenau, Fakultät für Informatik und Automatisierung, Institut für Praktische Informatik und Medieninformatik, Dissertation, 2003

[NM65] Nelder, J. A. ; Mead, R.: A Simplex Method for Function Minimization. In: *Computer Journal* 7 (1965), S. 308–313

[PP04] Pease, Allan ; Pease, Barbara: *Die kalte Schulter und der warme Händedruck.* Berlin : Ullstein Buchverlage GmbH, 2004. – ISBN 3–550–076004–5

[Rau09] Rauber, Andreas: *SOMs und verwandte Verfahren.* Version: 2009. `http://cocoon.ifs.tuwien.ac.at/lehre/veranstaltungen/04w_neural-comp/s%om.pdf`, Abruf: 20. 08. 2009

[RHC99] Rui, Yong ; Huang, Thomas S. ; Chang, Shih-Fu: Image Retrieval: Current Techniques, Promising Directions And Open Issues. In: *Journal of Visual Communication and Image Representation* 10 (1999), S. 39–62

[RMC91] Robertson, George G. ; Mackinlay, Jock D. ; Card, Stuart K.: Cone Trees: animated 3D visualizations of hierarchical information. In: *CHI '91: Proceedings of the SIGCHI conference on Human factors in computing systems.* New York, NY, USA : ACM Press, 1991. – ISBN 0897913833, S. 189–194

[Roc71] Rocchio, J. J.: Relevance Feedback in Information Retrieval. In: *The SMART Retrieval System-Experiments in Automatic Document Processing* (1971), S. 313–323

[Sch04] Schulz, Nadine: *Formulierung von Nutzerpräferenzen in Multimedia-Retrieval-Systemen*, Otto-von-Guericke-Universität Magdeburg, Dissertation, 2004. – Fakultät für Informatik

[Sch06] Schmitt, Ingo: *Ähnlichkeitssuche in Multimedia-Datenbanken. Retrieval, Suchalgorithmen und Anfragebehandlung.* München : Oldenbourg Wissenschaftsverlag GmbH, 2006

[Sch07] Schmitt, Ingo: Weighting in CQQL / Brandenburg University of Technology at Cottbus, Institute of Computer Science. 2007 (04). – Computer Science Reports

[Sch08] Schmitt, Ingo: QQL: A DB&IR Query Language. In: *VLDB J.* 17 (2008), Nr. 1, S. 39–56

[Shn92a] Shneiderman, Ben: *Designing the User Interface : strategies for effective human-computer interaction.* 2. ed. Reading, Mass. [u.a.] : Addison-Wesley, 1992. – ISBN 0–201–57286–9

[Shn92b] Shneiderman, Ben: Tree visualization with tree-maps: A 2-D space-filling approach. In: *ACM Transactions on Graphics* 11 (1992), S. 92–99

[Shn03] Shneiderman, Ben: Promoting Universal Usability with Multi-Layer Interface Design. In: *CUU '03: Proceedings of the 2003 conference on Universal usability.* New York, NY, USA : ACM, 2003. – ISBN 1–58113–701–X, S. 1–8

[SM00] Schumann, Heidrun ; Müller, Wolfgang: *Visualisierung: Grundlagen und allgemeine Methoden.* Berlin, Heidelberg : Springer, 2000. – ISBN 3–540–64944–1

[Sou09] SourceForge: *SOURCEFORGE.NET - Caliph and Emir*. Version: 1999-2009. `http://sourceforge.net/project/showfiles.php?group_id=105915`, Abruf: 22. 06. 2009

[Spe07] Spence, Robert: *Information Visualization: Design for Interaction (2nd Edition)*. Upper Saddle River, NJ, USA : Prentice-Hall, Inc., 2007. – ISBN 0–132–0655–09

[SRP07] Sharp, Helen ; Rogers, Yvonne ; Preece, Jenny: *Interaction design: beyond human - computer interaction*. 2. ed. Chichester, England : Wiley, 2007. – ISBN 978–0–470–01866–8

[Sta03] State Hermitage Museum: *QBIC Colour and Layout Searches*. Version: 2003. `http://www.hermitagemuseum.org/fcgi-bin/db2www/qbicSearch.mac/qbic?sel%Lang=English`, Abruf: 08. 06. 2009. – Verwendung von IBM's QBIC zur Suche in der Kunstsammlung des State Hermitage Museum

[SZ09] Schmitt, Ingo ; Zellhöfer, David: Lernen nutzerspezifischer Gewichte innerhalb einer logikbasierten Anfragesprache. In: *BTW*, 2009, S. 137–156

[VT02] Veltkamp, R.C. ; Tanase, M.: Content-Based Image Retrieval Systems: A Survey / Department of Computing Science, Utrecht University. Version: Oktober 2002. `http://give-lab.cs.uu.nl/cbirsurvey/cbirsurvey.pdf`, Abruf: 29. 05. 2009. 2002 (UU-CS-2000-34). – Forschungsbericht. – überarbeitete und erweiterte Version vom Forschungsbericht UU-CS-2000-34, Oktober 2000

[Zel09] Zellhöfer, David: Ein Hybridansatz zur Interaktion mit Retrievalsystemen. In: *Proceedings of the 21th GI-Workshop on Foundations of Databases*. Warnemünde, June 2009

[ZL08] Zellhöfer, David ; Lehrack, Sebastian: Nutzerzentriertes maschinenbasiertes Lernen von Gewichten beim Multimedia-Retrieval. In: *Grundlagen von Datenbanken*, 2008, S. 51–55

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Bachelorarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

(Ort, Datum) (Bianca Böckelmann)