ALBERT-LUDWIGS-UNIVERSITÄT
FREIBURG
INSTITUT FÜR INFORMATIK

Lehrstuhl für Mustererkennung und Bildverarbeitung
Prof. Dr. Hans Burkhardt

3D Shape Retrieval mit lokalen Merkmalen

Diplomarbeit

Alexander Streicher

Mai 2008 – November 2008

ALBERT-LUDWIGS-UNIVERSITÄT FREIBURG
FAKULTÄT FÜR ANGEWANDTE WISSENSCHAFTEN

Institut für Informatik
Lehrstuhl für Mustererkennung und Bildverarbeitung
Prof. Dr.-Ing. Hans Burkhardt

3D Shape Retrieval mit lokalen Merkmalen

Diplomarbeit

Verfasser: Alexander Streicher

Abgabedatum: 19. November 2008

Gutachter: Prof. Dr.-Ing. Hans Burkhardt
Prof. Dr. Matthias Teschner

Betreuer: Dipl.-Inf. Janis Fehr

ALBERT-LUDWIGS-UNIVERSITÄT FREIBURG
INSTITUT FÜR INFORMATIK
Lehrstuhl für Mustererkennung und Bildverarbeitung
Prof. Dr.-Ing. Hans Burkhardt

Georges Köhler Allee
Geb. 052, Zi 01-029
D-79085 Freiburg
Tel. (0761) 203-8260

8. Mai 2008

Aufgabenstellung für die Diplomarbeit
von Herrn Alexander Streicher.

3D Shape Retrieval mit lokalen Merkmalen

Im Bereich der 2D Bildsuche (Image Retrieval) haben sich Verfahren durchgesetzt, welche im Wesentlichen auf der Verwendung von lokalen Merkmalen (Patches) beruhen.
In der vorliegenden Arbeit sollen nun die etablierten Konzepte von 2D Image auf 3D-Shape Retrieval angewand werden. Dabei sollen unter Anderem 3D Interest-Point Detektoren, lokale Merkmale für 3D Patches sowie Clustering Verfahren zur Erstellung eines Codebooks untersucht werden.
Diese Verfahren sollen anhand der "Princeton Shape Benchmark" (PSB) [1] und an einer eigenen Objektdatenbank evaluiert werden. Die Aufgabenstellung beinhaltet dabei u.A.:

- Einarbeitung in die Literatur sowie die bestehenden Softwarebibliotheken.
- Implementation und Evaluation der globalen Referenzverfahren aus [1] und [2].
- Entwurf und Implementation von geeigneten 3D Interest-Point Detektoren, [3].
- Entwurf und Implementation von geeigneten Cluster-Verfahren zur Erstellung des Codebooks.
- Implementation des oben beschriebenen Retrieval Algorithmus.
- Evaluation der Methoden anhand der PSB und der eigenen Objektdatenbank.

Literatur:

[1] Philip Shilane, Patrick Min, Michael Kazhdan, and Thomas Funkhouser, "The Princeton Shape Benchmark", *Shape Modeling International*, Genova, Italy, June 2004.

[2] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz, "Rotation invariant spherical harmonic representation of 3D shape descriptors." *in Symposium on Geometry Processing*, June 2003.

[3] Fehr, J., Burkhardt, H.,Phase based 3D Texture Features, in Proceedings DAGM 2006, Springer LNCS 4174, pp 263-272

ERKLÄRUNG

Hiermit erkläre ich, dass ich diese Abschlussarbeit selbständig verfasst habe, keine anderen als die angegebenen Quellen/Hilfsmittel verwendet habe und alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten Schriften entnommen wurden, als solche kenntlich gemacht habe. Darüber hinaus erkläre ich, dass diese Abschlussarbeit nicht, auch nicht auszugsweise, bereits für eine andere Prüfung angefertigt wurde.

Ort, Datum

Unterschrift

Danksagung

Mein Dank gilt Prof. Dr.-Ing. Hans Burkhardt, dass er es mir ermöglicht hat diese Arbeit am Lehrstuhl für Mustererkennung und Bildverarbeitung anzufertigen.

Auch gilt mein Dank den Mitarbeitern des Lehrstuhls für die sehr angenehme Arbeitsatmosphäre und die allgemeine Hilfsbereitschaft.

Ein besonderer Dank gilt meinem Betreuer Dipl.-Inf. Janis Fehr für seine engagierte Betreuung sowie für die geduldige Beantwortung meiner Fragen.

Kurzfassung

Im Bereich der 2D Bildsuche (Image Retrieval) haben sich Verfahren durchgesetzt, welche im Wesentlichen auf der Verwendung von lokalen Merkmalen (Patches) beruhen. In der vorliegenden Arbeit werden die etablierten Konzepte von 2D Image auf 3D-Shape Retrieval übertragen. Dazu werden 3D Interest-Point Detektoren, lokale Merkmale für 3D Patches sowie Clustering Verfahren zur Erstellung eines universellen Codebuches untersucht. Es werden 3D Interest-Point Detektoren vorgestellt, die basierend auf Kugelflächenfunktionen stabile Punktpositionen an charakteristischen Stellen auf 3D Modellen finden. Die Repräsentation lokaler Merkmale als Patches wird durch eine Abbildung als Kugelflächenfunktionen erreicht. Bei der Erstellung des Codebuches werden Methoden zur Erzeugung universeller Codebucheinträge dargelegt, und Histogramm-basierte Features vorgeschlagen. Diese Verfahren werden anhand der "Princeton Shape Benchmark" (PSB) evaluiert.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Die wachsende Zahl digital verfügbarer 3D Modelle ist in den letzten Jahren rapide gestiegen. Diese Entwicklung wird höchstwahrscheinlich andauern und, wie im Fall von textuellen Daten und Bildern, die Entwicklung von 3D Suchmaschinen erforderlich machen, die ein Retrieval auf den Datenbanken mit 3D Modellen ähnlich effizient ausführen können, wie dies bei der Suche nach Web-Seiten oder Bildern schon heute der Fall ist. Während aber die Entwicklung von Suchmaschinen heute als Ziel die semantische Erfassung von Texten und Bildern im Hinblick auf das *Semantic Web* verfolgt, ist man bei der Verarbeitung von dreidimensionalen Daten noch am Anfang der Entwicklung effizienter Retrieval-Systeme. Das Erkennen und Wiederfinden von räumlichen Modellen ist Gegenstand aktueller Forschungsaktivitäten und bis heute größtenteils Grundlagenforschung.

Im Bereich der dreidimensionalen Datenverarbeitung stellen 3D Modelle digitale Repräsentationen von real existierenden Objekten oder von künstlich erzeugten Daten dar, die nicht zwangsweise nach real existierenden Abbildern modelliert wurden.

Reduziert man diese digitale Repräsentation auf nur die geometrische Form, so spricht man von einem 3D *Shape*. Nach Kendall *et* al. [27] ist die Form (*Shape*) eines 3D Modells die geometrische Information, die zurückbleibt, wenn die Verschiebung, Skalierung und Rotation eliminiert werden. In dieser Arbeit wurden 3D Modelle benutzt, die dieser Definition genügen und nur durch geometrische Informationen parametrisiert werden.

Um von einem Objekt eine digitale 3D Repräsentation zu erhalten, gibt es mehrere Methoden. Die Digitalisierung real existierender Objekte kann etwa mittels Laser-Scanning vorgenommen werden, bei denen ein Objekt von allen Seiten mit einem Laser abgetastet wird. Zu den künstlich erzeugten Objekten lassen sich Daten aus dem Bereich des Computer Aided Design (CAD) zählen, wie sie beispielsweise im Maschinenbau erstellt werden. Aber auch die Visualisierung biologischer oder chemischer Daten, wie etwa von Organen beziehungsweise Molekülen, basiert auf der Darstellung als dreidimensionale Modelle. Und

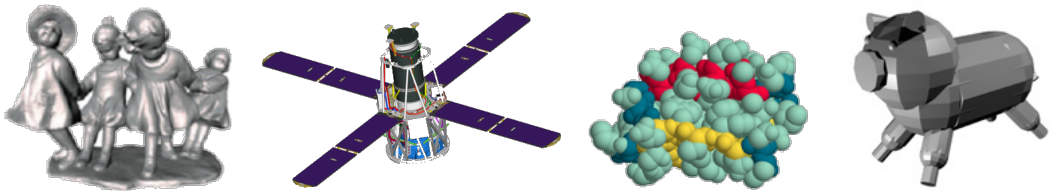


Abbildung 1.1: Typbeispiele von 3D Modellen (von links nach rechts): 3D Laser-Scan von Figuren, CAD-Modell eines Satelliten, 3D Modell eines Proteins, künstliches VRML-Modell aus der *Princeton Shape Benchmark*.

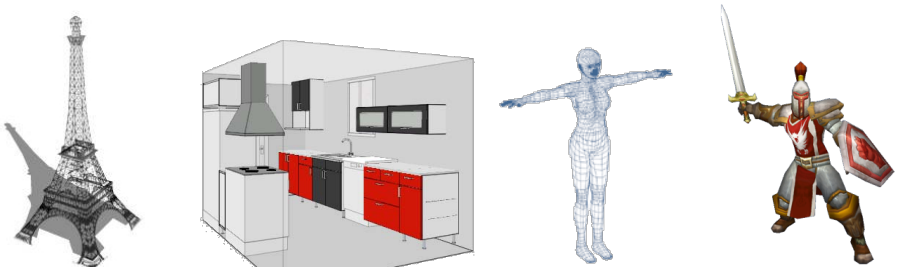


Abbildung 1.2: 3D Modelle im Alltag (von links nach rechts): Eiffelturm aus Google Earth, Küchenstudio Ikea, Basis-Modell aus Second Life („Avatar“), Computerspielfigur.

auch in der Medizin wird die dreidimensionale Datenverarbeitung erfolgreich bei der Darstellung von Daten aus der Magnetresonanztomographie eingesetzt. In Abbildung 1.1 sind Beispiele zu den aufgezählten Bereichen vorgestellt.

In den letzten Jahren wurde die Digitalisierung von realen Objekten immer weiter vereinfacht. Dank leistungsfähiger Computer und einem großen Angebot an Software zur dreidimensionalen Datenverarbeitung, ist mit einem Anstieg digitaler 3D Modelle zu rechnen. Auch im Privatanwenderbereich sind heute schon für die dreidimensionale Datenverarbeitung leistungsstarke Personal Computer und für Laien benutzbare Software verfügbar, so dass die Erstellung von 3D Modellen nicht nur auf den Bereich der professionellen Datenverarbeitung beschränkt ist.

Im Zuge der Durchdringung handelsüblicher Personal Computer mit leistungsfähiger 3D Hardware und dem Bandbreitenausbau des Internets, wird schon heute an vielen Stellen Gebrauch von 3D Modellen gemacht, zum Beispiel bei der digitalen Erkundung unseres Planeten mit Google Earth oder in der virtuellen Welt von Second Life (siehe Abbildung 1.2).

Mit der steigenden Zahl von 3D Modellen im Internet und anderen Applikationen wird auch die Bedeutung der 3D Datensuche, speziell also des *3D Model Retrievals*, steigen. Es kann davon ausgegangen werden, dass 3D Daten eine ähnliche Verbreitung finden werden, wie es zuvor mit den 2D Bilddaten der Fall war. Mit gängigen Internet-Suchmaschinen ist es heute möglich neben textuellen Daten auch Bilder oder andere Inhalte zu suchen. Dies wird die Erstellung von Datenbanken mit 3D Inhalten begünstigen. Neben rein kommerziellen Datenbanken mit 3D Modellen sind frei zugängliche Datenbanken publik gemacht worden. Beispiele hierfür sind die Viewpoint-Datenbank mit 3D Modellen unterschiedlicher Art, das National Design Repository für CAD Modelle, die 3D Modelle auf *www.3dcafe.com*, oder die auch in dieser Arbeit genutzte Datenbank der Princeton Shape Benchmark [41].

Da die Erstellung von neuen 3D Modellen meist nicht mit einer Annotierung des dargestellten Modells verbunden ist, oder aber die Annotierungen verloren gehen, weil sie etwa als externe Daten außerhalb der eigentlichen Objektdatei hinterlegt wurden, sind Verfahren zur Extraktion der intrinsischen Informationen zu einem 3D Modell notwendig, um eine effiziente Suche auf den Datenbanken mit 3D Modellen auszuführen. Es ist vorstellbar, dass in Kombination mit geeignet gelernten Wissensbasen (*Knowledge Base*) sich

semantische Aussagen über 3D Modelle treffen lassen, was von enormer Bedeutung für ein effizientes Retrievalsystem sein kann.

In dieser Diplomarbeit wird ein Verfahren zur Suche in einer Datenbank mit 3D Modellen vorgestellt, das vollständig ohne Annotierungen oder andere externe Informationen auskommt.

1.1 Zielsetzung

Die vorliegende Diplomarbeit hat als Ziel ein Verfahren zu entwickeln, um lokale 3D Merkmale (*Patches*) zu finden, mit diesen ein Codebuch mit universellen Objektteilen zu erstellen, und anhand von einer Datenbank mit 3D Modellen (*3D Shapes*) in einem standardisierten Retrieval-Prozess zu evaluieren. Grundlage für die lokalen Merkmale sind Kugelflächenfunktionen (*Spherical Harmonics*), die an charakteristischen Stellen (*Interest Points*) entwickelt und im folgenden Prozess einem Clustering unterzogen werden, um sowohl Gruppierungen von Objektteilen als auch Gruppierungen von 3D Modellen zu finden. Die Herausforderung bei dieser Art der Objektklassifizierung liegt darin universelle Patches für das Codebuch zu finden, die einerseits invariant bezüglich den Intra-Klassenvariationen, andererseits aber auch diskriminativ bezüglich anderen Klassen sind.

1.2 Aufbau dieser Arbeit

Diese Arbeit ist wie folgt gegliedert:

Nach dieser Einleitung werden andere Arbeiten vorgestellt, die allgemein das 3D Retrieval zum Thema haben, und spezieller die Arbeiten, die das Retrieval mittels lokaler Merkmale oder mit Hilfe eines Codebuches untersuchen.

In **Kapitel 2** werden theoretischen Grundlagen und grundlegende Werkzeuge vorgestellt, auf denen das entwickelte Verfahren beruht. Es wird eine Einführung in den Retrievalprozess gegeben, auch werden die Werkzeuge beschrieben, die zur Erstellung der lokalen Merkmale und des Codebuches benutzt werden. Dazu gehören neben den Spherical Harmonics, die für die Erstellung lokaler Features benutzt werden, auch Cluster-Verfahren zum Erlernen des Codebuches.

Kapitel 3 behandelt die Extraktion von lokalen Merkmalen, gegliedert in die Suche nach Interest Points und der Abbildung der lokalen Punkte in Form von Spherical Harmonics Koeffizienten. Diese an lokalen Stellen extrahierten Merkmale stellen die Grundlage für die lokalen Features dar, wie sie für das Codebuch verwendet werden.

Die Erzeugung des Codebuches wird im darauf folgenden **Kapitel 4** beschrieben. Auch wird darauf eingegangen, wie aus den lokalen Merkmalen Repräsentanten gebildet werden, um ein Codebuch mit möglichst universellem Charakter zu erhalten.

Auf die durchgeführten Experimente und Ergebnisse wird in **Kapitel 5** eingegangen. Neben der Präsentation der Ergebnisse des entwickelten Verfahrens wird die Princeton Shape Benchmark vorgestellt und die Ergebnisse bezüglich anderen Arbeiten eingeordnet.

Abschließend werden in **Kapitel 6** die erzielten Ergebnisse zusammengefasst und mögliche Erweiterungen diskutiert.

1.3 Überblick über das Verfahren

Das in dieser Diplomarbeit entwickelte Verfahren besteht hauptsächlich aus der Extraktion der Merkmale (Patches) und der Erzeugung des universellen Codebuches. Der Transfer dieser Konzepte von 2D auf 3D, und die damit verbundene Entwicklung von Methoden in 3D zur Merkmalsextraktion und Codebucherzeugung, ist der Beitrag dieser Diplomarbeit.

In Abbildung 1.3 ist der Arbeitsablauf des Verfahrens schematisch dargestellt.

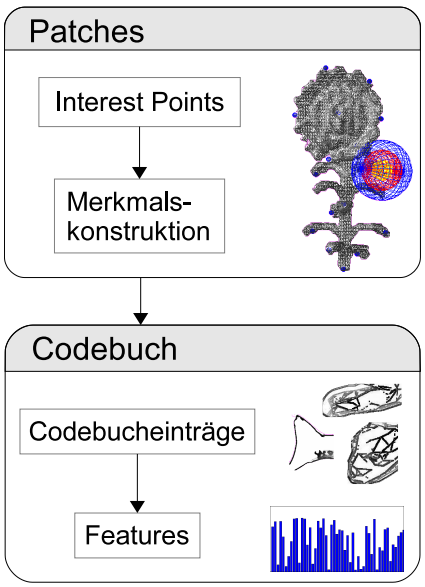


Abbildung 1.3: Schematischer Arbeitsablauf des entwickelten Verfahrens zum Codebuch-basierten 3D Shape Retrieval mit lokalen Merkmalen. Dem Schritt der Merkmalsextraktion - hier Extraktion lokaler Patches - folgt die Erstellung des Codebuches mit den lokalen Features.

1.4 Verwandte Arbeiten

Das *3D Model Retrieval* ist in den letzten Jahren Ziel intensiver Forschungsaktivitäten geworden, wobei vieles als Grundlagenforschung betrachtet werden kann. Ziel all dieser Arbeiten ist es das 3D Shape Retrieval Problem effizient zu lösen. Dabei wird einerseits versucht die etablierten Verfahren aus 2D auf 3D zu übertragen, andererseits werden aber auch neue Verfahren bezüglich 3D erforscht.

Das Retrieval-Problem ist grundsätzlich nicht auf die bekannten Verfahren der Merkmalsextraktion und des Matching beschränkt. Methoden wie Iterative Closest Point (ICP), generalisierte Hough-Transformation oder das Geometric Hashing, könnten das Retrieval-Problem zwar lösen, sind aber wegen der hohen Dimensionalität der 3D Daten für einen Online-Schritt zu ineffizient. Aus diesem Grund werden Anstrengungen unternommen, die klassischen Konzepte aus der 2D Bildsuche mit der Erstellung von Merkmalen auf den Bereich des 3D Model Retrieval zu übertragen und somit effizientere Methoden zur Lösung des Retrieval Problems in 3D zu finden.

Im Folgenden werden Arbeiten vorgestellt, die das 3D Shape Retrieval behandeln und sich den Konzepten der Merkmalsextraktion und der Erstellung von Codebüchern bedienen.

Mit der *Princeton Shape Benchmark* (PSB) wurde von Shilane *et* al. [41] eine frei verfügbare Datenbank mit 3D Modellen und standardisierten Werkzeugen zur Evaluation von Merkmalsdeskriptoren für das 3D Retrieval vorgestellt. Es werden zwölf Merkmalsdeskriptoren anhand dieser Datenbank mit unterschiedlichen statistischen Werkzeugen evaluiert. Die Autoren schlagen vor, dass die Datenbank als Referenzmedium für 3D Retrieval Algorithmen genutzt werden soll. Einige Arbeiten greifen dies auf und evaluieren ihre Algorithmen anhand der PSB [17, 38, 3, 14]. Die 3D Daten der PSB sind nicht nur auf die Evaluation bezüglich 3D Retrieval beschränkt. Fehr *et* al. [15] nutzen die Daten, um ihr Verfahren zur schnellen Rotationsschätzung auf Spherical Harmonics zu evaluieren.

Neben dem Princeton Shape Benchmark gibt es eine Reihe anderer Programme zur Evaluation verschiedener 3D Retrieval Algorithmen. Im MPEG-7 Standard [32] werden Anforderungen an 3D Deskriptoren definiert, um die Deskriptorerstellung zu standardisieren. Ein anderer Benchmark auf spezialisierten Datenbeständen ist der *SHape REtrieval Contest* (*SHREC*), zum Beispiel werden bei Temerinac *et* al. [46] Methoden zum Retrieval auf 3D Proteinstrukturen untersucht.

Für eine weitergehende Übersicht und Vergleiche unterschiedlicher Arten von Feature Deskriptoren und deren Performance beim 3D Retrieval sei auf [45, 7, 6] verwiesen.

Ähnlich wie bei der 2D Bildsuche gibt es in 3D viele Arbeiten, die sich mit der Extraktion geeigneter 3D Interest Points beziehungsweise von 3D Merkmalen beschäftigen. Die in der vorliegenden Arbeit verwendeten Spherical Harmonics werden in einer Reihe anderer Arbeiten zur Extraktion von Feature Deskriptoren benutzt [40, 50, 26, 49, 34, 14, 33].

Kazhdan *et* al. [26] beschreiben einen Ansatz, der auf der Verwendung der Spherical Harmonics beruht und einen rotationsinvarianten Deskriptor basierend auf der Energie in den Frequenzen liefert. Bei diesem *Spherical Harmonic Descriptor* (SHD) werden als Volumen gerenderte 3D Daten auf eine Kollektion von Kugeloberflächen reduziert, und der rotationsinvariante Deskriptor wird dadurch erzeugt, dass diese sphärischen Funktionen nach Spherical Harmonics entwickelt und die Harmonics pro Band aufsummiert werden.

Für jeden Radius aus der Kollektion der Kugeloberflächen wird die L_2-Norm berechnet und in einem 2D-Histogram aufgetragen. Dieses Histogram ist das Feature, bestehend aus den Radien und der Frequenzen.

Ein anderer Ansatz basierend auf Bildern unterschiedlicher Perspektiven des 3D Modells wird von Chen *et* al. [8] als *Light Field Descriptor* vorgestellt. Ein Modell wird durch eine Kollektion von Aufnahmen repräsentiert, die von gleichmäßig ausgewählten Punkten auf einer das Modell umgebenden Kugel aufgenommen wurden. Die Idee ist, dass ähnliche Modelle aus unterschiedlichen Blickwinkeln weiterhin ähnlich aussehen. Dazu werden 100 orthogonale Projektionen aufgenommen und mittels Zernike Momenten und Fourier Deskriptoren repräsentiert.

Statt ein Objekt als multidimensionalen Feature Deskriptor abzubilden, können auch die geometrischen und topologischen Strukturen eines 3D Modells untersucht werden. Hilaga *et* al. [20] führen topologische Vergleiche mittels *Multiresolutional Reeb Graphs* durch, bei denen die topologische Struktur eines 3D Modells zu unterschiedlichen Auflösungen kodiert wird. Ein ähnlicher Ansatz wird auch bei Sundar *et* al. [44] gewählt, indem 3D Objekte als Skelettgraphen repräsentiert und die Vergleiche mittels graphenbasierten Techniken durchgeführt werden.

Das Konzept ein Codebuch als effiziente Indexstruktur für das Retrieval zu nutzen kommt aus dem Bereich der Texturanalyse und der textuellen Dokumentenverarbeitung, wo es als *Bag-of-Features* beziehungsweise *Bag-of-Words* bekannt ist.

In 2D beschäftigen sich einige Arbeiten mit diesem Konzept. Ein zu dieser Arbeit ähnlicher Ansatz in 2D wird von Dance *et* al. [9] vorgestellt. Es werden Bildbestandteile von Fotos anhand eines Codebuches klassifiziert, das zuvor anhand extrahierter Patches auf den Daten trainiert wurde. Zur Klassifizierung werden Histogramme als Features benutzt, die eine Häufung der Patches beim Retrieval widerspiegeln. Die Klassifizierung der Histogramme wird mit einer Support-Vector-Machine und einem Naive Bayes Ansatz durchgeführt.

Jurie *et* al. [25] greifen das Codebuch-Konzept auf und untersuchen, wie sich effiziente Codebücher zur visuellen Klassifikation erstellen lassen. Hauptaugenmerk wird auf die Anzahl der Patches und das k-Means basierte Clustering gelegt. Es wird gezeigt, dass für ein dichtes Sampling von Features das k-Means basierte Codebuch zur Überanpassung neigt, da die Cluster-Zentren sich an den Regionen höchster Dichte sammeln. Als Lösung wird eine Erweiterung des k-Means Algorithmus vorgeschlagen, die eine Restriktion der Cluster-Zuordnungen anhand eines festen Radius einführt.

Nowak *et* al. [35] untersuchen wie stark sich die Codebuchgröße auf die Klassifikation bei Bag-of-Features basierten Verfahren auswirkt. Auf den verwendeten 2D Daten wird festgestellt, dass für eine große Anzahl von zufällig und gleichmäßig ausgewählten Features die Ergebnisse besser sind als bei Verfahren mit komplexerer Merkmalsextraktion (Harris-Laplace, Laplacian of Gaussians, etc.). Es wird aber auch beobachtet, dass für eine große Anzahl von Codebucheinträgen eine Sättigung eintritt und die Ergebnisse nicht wesentlich verbessert werden können.

Die auch in der vorliegenden Arbeit verwendete Idee das Codebuch mit universellen Features zu füllen wird von Winn *et* al. [52] in 2D vorgestellt. Es werden diejenigen Features miteinander zu neuen, universellen Features verschmolzen, die nur wenig zur Unterscheidung von Klassen beitragen. Im Vergleich zu den oben vorgestellten Arbeiten, werden hier

aber alle Pixel eines Bildes berücksichtigt, die dann zu den Codebucheinträgen verschmolzen werden.

Ein mit dieser Diplomarbeit vergleichbares Verfahren in 3D wurde von Huber *et* al. [23] für die teilbasierten 3D Objektklassifikation vorgestellt, bei dem festgelegte Teile von Autos klassifiziert werden. Aber sowohl die Anzahl der Teile als auch die Klassen werden fest vorgegeben, nämlich die drei Klassen für die Konzepte Front, Mitte und Rückseite der Autos. Mit einem hierarchischen Cluster-Verfahren werden die Teile aus den drei Abschnitten jeweils einer Klasse zugewiesen.

Kapitel 2

Grundlagen

Die theoretischen Grundlagen für die folgenden Kapitel werden in diesem Kapitel aufgeführt. Neben einer kurzen Einführung in das Retrieval und die dort verwendeten Werkzeuge zur Evaluierung werden die für die Feature Extraktion benötigten Spherical Harmonics, die Kugelflächenfunktionen, vorgestellt. Daran anschließend wird auf Werkzeuge zur Korrelation und Rotationsschätzung von Spherical Harmonics eingegangen. Anhand der Korrelationswerte der Spherical Harmonics wird ein Clustering durchgeführt, was einerseits für die Merkmalsextraktion verwendet wird, andererseits auch für die Erstellung des Codebuches. Die hierzu untersuchten Cluster-Verfahren werden vorgestellt, und es wird auf die Evaluation der Cluster-Verfahren eingegangen.

2.1 Retrieval

Unter *Information Retrieval* versteht man das Suchen und Auffinden von Informationen in Datenbanken. Vergleicht man das Retrieval mit der Informationsaufnahme beim Menschen, so stellt das Retrieval den kognitiven Prozess dar, bei dem neue Informationen (dazu zählen alle sensorischen Empfindungen) mit anderen, bereits bekannten beziehungsweise gelernten Informationen verknüpft werden, die im Gedächtnis, der zentralen „Datenbank“, gespeichert sind.

Das menschliche Gehirn ist in der Lage neue Informationen zu charakterisieren und zu klassifizieren. Dabei speichert es nicht alle Informationen vollständig, sondern nur eine hinlängliche Abstraktion, die es ihm danach ermöglicht sich an die Ursprungsinformation zu erinnern und diese eindeutig wiederzufinden. Abhängig vom Grad der semantischen Information und dem bereits vorhandenen Hintergrundwissen können neue, noch unbekannte Informationen oftmals ausreichend genau klassifiziert werden.

Maschinelles Retrieval folgt den Grundsätzen der Natur und bedient sich eines ähnlichen Prozesses, der sich in die drei Stufen der Datenerfassung, Suche und Suchergebnisausgabe gliedern lässt.

Bei der Datenerfassung werden in der Regel Merkmale auf den Daten erzeugt, um den hochdimensionalen Merkmalsraum zu abstrahieren. Diese objektbezogenen, charakteristischen und eindeutigen Merkmale beschreiben die Daten entweder ganzheitlich (globale Features) oder im Verbund teilweiser Merkmale (lokale Features). Die Entropie eines gegebenen Datensatzes ist ausschlaggebend, wie komplex die Merkmale zu konstruieren sind, und wie hoch die Dimensionalität des Features sein wird. Solch ein Feature wird beim Retrieval im Vektorraummodell durch einen reellwertigen *Feature Vector* $\vec{x} \in \mathbb{R}^n$ abgebildet; man spricht hier auch vom sogenannten Feature-Deskriptor.

Die Suche bei einem Retrievalprozess nutzt die in einer Datenbank hinterlegten Feature-Vektoren, um für eine Suchanfrage (Query) eine Teilmenge der ähnlichsten Kandidaten zu finden. Beim Retrieval im Vektorraummodell wird eine Ähnlichkeitssuche nach einem Suchobjekt q in der Regel als eine k-Nächste-Nachbar Klassifikation ausgeführt. Das Ergebnis ist eine sortierte Liste mit k Objekten $o_i,\ i = 1..k$, deren Distanzen zu q bezüglich einer Metrik am geringsten sind und danach geordnet werden können.

Für die Distanzmetrik wird in den meisten Retrievalsystemen auf die Familie der *Minkowski*-Distanzfunktionen zurückgegriffen. Die Minkowski-Distanz l_p ist definiert als

$$l_p(\vec{x}, \vec{y}) = \sqrt[p]{\sum_{i=1}^{n} |x_i - y_i|^p} \quad , \vec{x}, \vec{y} \in \mathbb{R}^n,\ p \geq 1. \tag{2.1}$$

Die bekanntesten Vertreter dieser Distanzfamilie sind die *Manhatten Distanz* für l_1, die *euklidische Distanz* für l_2 oder die Maximumsdistanz für $l_\infty = max|\vec{x} - \vec{y}|$. Erweiterungen zur Minkowksi-Distanz sind etwa die gewichtete Minkowski-Distanz oder die Mahalanobis-Distanz. Es werden aber auch andere Distanzmetriken verwendet, wie zum Beispiel die *Haussdorff Distanz*, die *Elastic Matching Distance* oder die *Earth Mover's Distance*. Diese Distanzmetriken sind aber aufwändig zu berechnen und liefern meist nur für spezielle Probleme bessere Ergebnisse. Im Rahmen dieser Diplomarbeit hat sich an Testdaten gezeigt, dass sich die Familie der Minkowski-Distanzen, speziell die euklidische Distanz, besser für das vorgestellte Verfahren eignet.

Neben der maschinellen Klassifizierung anhand der Distanzmetriken gibt es bei manchen Retrievalsystemen die Option die Suchergebnisse von den Benutzern bewerten zu lassen und dem Retrievalsystem als Feedback zurückzugeben. Anhand dieser Rückmeldung kann das Retrievalsystem die Klassifizierung für zukünftige Anfragen optimieren. Diese Form des überwachten Lernens findet sich heute in Form von sogenannten Web 2.0 Anwendungen wieder, wo die Benutzer die Ergebnisse zum Beispiel durch Benotung qualitativ bewerten können.

Anwendungsbeispiele für das Retrieval in der elektronischen Datenverarbeitung sind etwa Suchmaschinen im Internet, Bilddatenbanken, Digitale Bibliotheken, Literaturdatenbanken oder computergestützte Antwortsysteme.

Das Retrieval ist von der konventionellen Suche in einer klassischen Datenbank abzugrenzen. Die Suche beim Retrieval wird durch die Probleme der Vagheit und der Unsicherheit beeinflusst.

Die Vagheit besagt, dass eine Anfrage an ein Retrievalsystem vage Bedingungen enthält, da der Benutzer seine Anfrage nicht hinreichend präzise formulieren kann. Anders als bei syntaktisch klar definierten Abfragesprachen, wie zum Beispiel der Structured Query Language (SQL) bei Datenbanken, sind die Anfragen an Retrievalsuchmaschinen meist vage und nicht eindeutig formuliert. Bei textuellen Suchanfragen bedienen sich entsprechende Retrievalsysteme eines Katalogs von Satz- und Wortkonzepten[1], um die Suchanfrage zu formalisieren.

[1] Ein solcher Katalog kann zum Beispiel ein Wörterbuch oder ein Thesaurus sein; fortgeschrittenere Techniken nutzten logikbasierte Wissensbasen (Knowledge Base), die zur semantischen Analyse einer Suchanfrage dienen können.

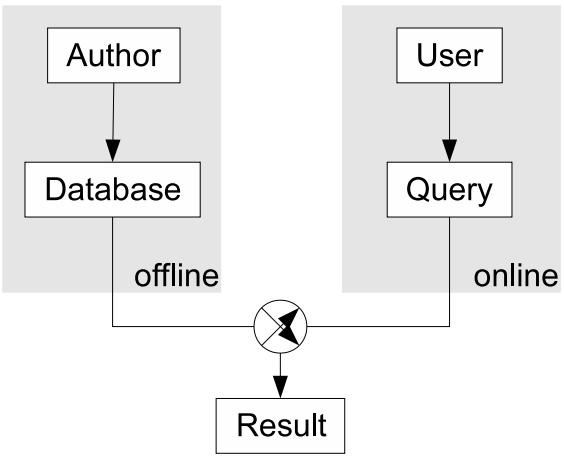


Abbildung 2.1: Schematische Darstellung des Retrieval Prozesses, geteilt in einen Offline- und einen Online-Schritt. Im Offline-Schritt wird eine Retrievaldatenbank von einem Experten erstellt, auf die dann von einem Benutzer zugegriffen werden kann.

Ein anderes Problem ergibt sich durch Unsicherheiten, die in einem Retrievalsystem vorherrschen. Das System kann keine definitive Garantie geben, dass die Ergebnisse fehlerfrei und vollständig sind. Sei dies entweder wegen den erwähnten vagen Suchanfragen, oder weil das korrekte Ergebnis nicht bekannt oder nicht in der Datenbank verfügbar ist. Die Vollständigkeit des Suchraums ist nicht garantiert. Hier ist ein weiterer Unterschied zur herkömmlichen Suche in Datenbanken: Liefert eine Suchanfrage kein exaktes Ergebnis, so wird beim Information Retrieval eine Liste mit zur Suchanfrage ähnlichen Ergebnissen zurückgeliefert, statt einer negativen Rückmeldung bei einer herkömmlichen Datenbank.

Der Aufbau eines Retrievalsystems besteht im Allgemeinen aus einem Offline- und einem Online-Schritt (siehe Abbildung 2.1). Ersteres ist das Lernen der Daten in einer Datenbank, so dass beim Online-Schritt die gesuchten Informationen schnell gefunden werden können. Der Lernprozess ist in der Regel zeitintensiv und muss nur einmalig ausgeführt werden. Man spricht vom Offline-Schritt, weil der Lernprozess nicht zeitnah an eine Suchanfrage gekoppelt ist. Im Online-Schritt dagegen ist eine Suchanfrage (Query) an die Datenbank zeitkritisch, da Benutzer des Retrievalsystem nur kurze Wartezeiten akzeptieren werden.

2.2 Evaluation

Das Primärziel für 3D Retrieval Algorithmen ist, die Suche auf einer 3D Datenbank effektiv und effizient auszuführen. Unter der Effektivität eines Information Retrieval Systems versteht man die Leistungsfähigkeit des Systems dem Benutzer die angeforderte Information in der erwarteten Qualität, unter Berücksichtigung der Zeit und der Aufwandskosten, zu liefern; ein perfektes Information Retrieval System liefert immer die gewünschte Information. Im Gegensatz zur schwieriger zu beurteilenden Effektivität eines Retrievalsystems, kann die Effizienz durch eine Analyse der verwendeten Algorithmen oder durch

Benchmark-Tests bestimmt werden. Eine Analyse der Effizienz ist in dieser Arbeit nicht Gegenstand der Evaluation, vielmehr liegt der Fokus auf der Analyse der Ergebnisse, die durch das vorgestelle Verfahren erreicht werden können. Dazu muss die Effektivität des Systems anhand empirischer Methoden eruiert werden.

Um ein Retrievalsystem bewerten zu können, müssen die korrekten Antworten des Systems auf jede Suchanfrage bekannt sein. Dazu wird das Konzept der Relevanz herangezogen. McGill *et* al. [39] definieren Relevanz für ein Information Retrieval System als

> *„Relevance is the correspondence in context between an information requirement statement (a query) and an article (a document), that is, the extend to which the article covers the material that is appropriate to the requirement statement.“*

Relevante Ergebnisse decken also die Suchanfrage derart ab, dass sie in einer gewissen Ähnlichkeit zur Anfrage stehen. Die Ähnlichkeitsrelation von Informationen wird in der Regel durch das Einwirken von Experten bestimmt. Eine Ähnlichkeitsrelation kann formal definiert werden als

$$r : Q \times D \to \mathbb{R},$$

wobei Q die Menge der Anfragen und D die Menge der in der Datenbank des Retrievalsystems hinterlegten Daten sind. Die Relevanz kann durch r entweder als unscharfe Ähnlichkeit mithilfe einer Distanz gekennzeichnet sein (siehe die Minkowski-Norm 2.1), oder als binäre Wahrheitsfunktion, die nur die Werte $\{0, 1\}$ zulässt - also für relevant und nicht relevant.

In der vorliegenden Arbeit sind in der Menge Q der Suchanfragen 3D Modelle, die in der 3D Datenbank D der Princeton Shape Benchmark (PSB) gefunden werden sollen, und die anhand einer gegebenen Ähnlichkeitsrelation r als relevant zur Suchanfrage eingestuft werden. r wurde von den Autoren der PSB, die in ihrer Eigenschaft als Autoren der Datenbank als Experten auftreten, vorgegeben und liegt in Form einer Klassifizierungsliste vor. Die Evaluation eines Retrievalsystems für die Modelle der Princeton Shape Benchmark orientiert sich an dieser vorgegebenen Klassifizierung.

Für eine Evaluation der Ergebnisse des vorgestellten Retrievalsystems stehen Methoden aus der Statistik zur Verfügung, die diese vorgegebene Klassifikation berücksichtigen. Sie sind Teil des Frameworks der PSB.

Auf der einen Seite gibt es Methoden, die einen numerischen Vergleichswert zurückliefern, auf der anderen Seite werden die Aussagen der Tests als Graphen visualisiert.

Zu den visuellen Methoden zählen die *Precision-Recall Plots*, Distanz- beziehungsweise Rangordnungsdiagramme, oder die *Receiver Operating Charakteristic* (ROC) Kurve.

Methoden, die ein Retrievalergebnis durch einen Funktionswert charakterisieren, sind die Nächste-Nachbar-Klassifikation, der *E-Measure* oder der *Discounted Cumulative Gain* (DCG). Im Folgenden werden einige dieser Methoden kurz vorgestellt.

Precision-Recall-Plot Das Precision-Recall-Diagramm stellt den Zusammenhang zwischen der Genauigkeit (Precision) und dem Rückruf (Recall) beim Information Retrieval dar. Recall und Precision sind Maße zur Beurteilung der Güte von Retrievalergebnissen und sind für eine Ergebnismenge P und eine Menge R von relevanten Objekten definiert als

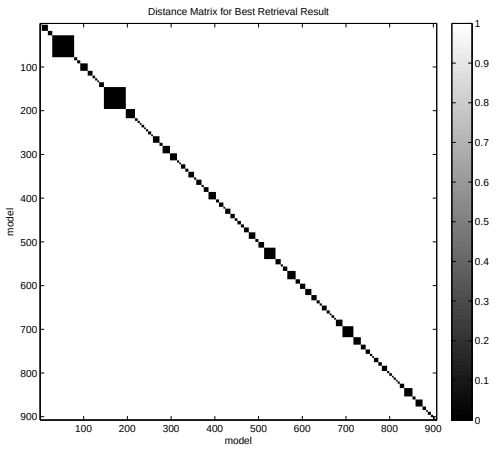


Abbildung 2.2: Perfekte Distanzmatrix für ein Retrievalergebnis auf der Princeton Shape Benchmark. Die Klassen stellen sich als Blöcke im Bild dar. Für größte Ähnlichkeit nehmen die Pixel den Wert 0 an, für größte Unähnlichkeit den Wert 1.

$$\text{Recall:} \quad \frac{|R \cap P|}{|R|} \qquad \text{Precision:} \quad \frac{|R \cap P|}{|P|} \; .$$

Der Recall stellt das Maß für die Vollständigkeit des Retrievalergebnisses dar, die Precision dagegen liefert ein Maß zur Genauigkeit einer Suche und kann als Indikator für die Fähigkeit des Retrievalsystems angesehen werden, nicht relevante Objekte unberücksichtigt zu lassen.

Precision und Recall werden für die Auswertung von Retrievalsystemen in der Regel gemeinsam in einem Precision-Recall-Diagramm betrachtet, wo auf der Abszisse die Precision und auf der Ordinate der Recall aufgetragen werden. Für ein perfektes Retrievalergebnis, bei dem genau alle relevanten Objekte zurückgeliefert werden, nehmen Precision und Recall den Wert 1 an und das zugehörige Diagramm ist eine horizontale Linie am oberen Ende des Diagramms (Precision = 1.0). Kurvenartige Verläufe dagegen zeigen ein schlechteres Ergebnis an.

Distanzmatrix Beim Bild einer Distanzmatrix gibt die Intensität eines Pixel $p(i, j)$ die Distanz zwischen den Objekten i und j an; auf der Diagonale haben alle Pixel den Wert 0, $p(i, i) = 0$. Liegen beim Retrieval die Objekte in geordneter Form nach Klassen vor, so stellt sich ein perfektes Retrievalergebnis in der Distanzmatrix als schwarze Blöcke (Distanz gleich 0) in der Größe der jeweiligen Klassen entlang der Diagonalen dar (siehe Abbildung 2.2); das schlechteste Ergebnis dagegen wäre das invertierte Bild.

First Tier & Second Tier Ein Rangordnungsdiagramm (*Tier-Image*) visualisiert neben den Ergebnisse eines Nächste-Nachbar-Vergleiches zusätzlich die ersten und zweiten Ränge. Den Query-Anfragen in den Zeilen j werden die Suchergebnisse in den Spalten i gegenübergestellt. Für ein Objekt in einer Klasse mit $|C|$ anderen Elementen gibt der Pixelwert $p(i, j)$ die Zugehörigkeit wie folgt an:

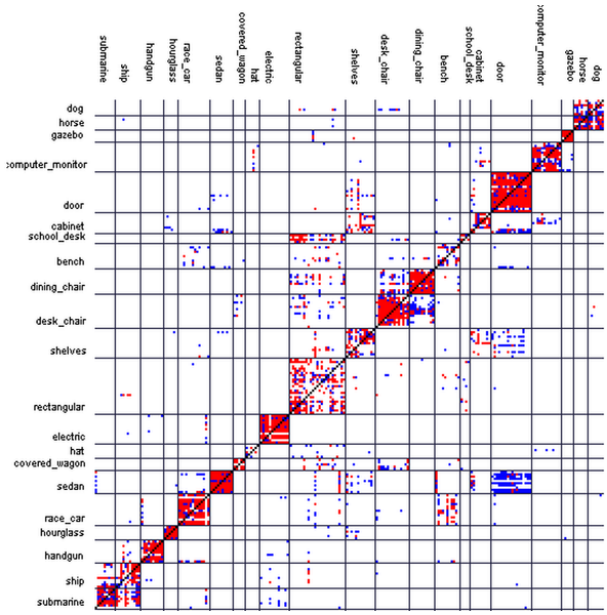


Abbildung 2.3: Rangordnungsdiagramm aus der PSB (für Light Field Deskriptor, LFD). Schwarze Pixel stehen für NN-Ergebnisse, rote für *first tier* und blaue für *second tier* Ergebnisse. Dargestellt ist ein Ausschnitt aus einem Bild mit 907×907 Pixel. (Quelle: [41])

- Schwarzer Pixel: Objekt i ist gleich Objekt j (Identität) oder sein nächster Nachbar.
- Roter Pixel: Objekt i ist im Bereich der ersten $|C| - 1$ Ergebnisse (erster Rang, *first tier*).
- Blauer Pixel: Objekt i ist im Bereich der ersten $2 \cdot (|C| - 1)$ Ergebnisse (zweiter Rang, *second tier*).

Für nach Klassen geordnete Listen stellt sich das Rangordnungsdiagramm für ein perfektes Retrievalergebnis als rote Blöcke entlang einer schwarzen Diagonalen dar. Bei weniger guten Ergebnissen mischen sich blaue Pixel in die Blöcke beziehungsweise die roten und blauen Pixel verteilen sich außerhalb der Blöcke (siehe Abbildung 2.3). An einem Rangordnungsdiagramm lässt sich ablesen, ob Objekte derselben Klasse besser zueinander passen, als Objekte anderer Klassen.

Für die Berechnung der statistischen Maße *first tier* und *second tier* werden dieselben Zuordnungen wie beim Diagramm benutzt.

E-Measure Der *E-Measure* ist ähnlich zu Precision und Recall, aber für eine eingeschränkte Anzahl von Retrievalergebnissen [41], und ist für eine Ergebnismenge P und eine Menge R von relevanten Objekten definiert als

$$E = \frac{2}{\frac{1}{P} + \frac{1}{R}} \cdot$$

Die Idee dahinter ist, dass sich der Benutzer eines Retrievalsystems nur für die besten - also die obersten - Ergebnisse interessieren wird. Im benutzten Framework werden deswegen nur die ersten 32 Modelle aus der Suchergebnisliste für die Berechnung des E-Measure Wertes benutzt.

Discounted Cumulative Gain Ähnlich wie beim E-Measure werden beim Discounted Cumulative Gain (DCG) die korrekten Ergebnisse an der Spitze der Ergebnisliste stärker gewichtet, als korrekte Ergebnisse im weiteren Verlauf zum Listenende hin.

Nach [41] wird für eine sortierte Ergebnisliste R eine neue Liste G mit Gewinnen G_i (Gains) erzeugt, indem Listenelemente G_i nach ihrer korrekten Klassifizierung entweder mit 0 (kein Gewinn) oder mit 1 (Gewinn) gekennzeichnet werden, so dass sich der DCG rekursiv wie folgt defniniert:

$$DCG_i = \left\{ \begin{array}{cc} G_1 & , i = 1 \\ DCG_{i-1} + \frac{G}{log_2(i)} & , \text{sonst} \end{array} \right\}$$

Das Ergebnis ergibt sich dann aus der Division durch den maximal möglichen DCG:

$$DCG = \frac{DCG_k}{1 + \sum_{j=2}^{|C|} \frac{1}{log_2(j)}} ,$$

wobei C die Klasse der Elemente zu Beginn der Liste ist, die korrekt klassifiziert wurden. k ist die Anzahl der Elemente in der Datenbank.

2.3 Spherical Harmonics

In den folgenden Kapiteln dieser Diplomarbeit wird beschrieben, wie auf einem 3D Objekt an ausgewählten Punkten Repräsentationen dieser Stellen extrahiert werden. Sowohl die Suche nach diesen Stellen als auch die Repräsentationen basieren auf den Kugelflächenfunktionen, den sogenannten Spherical Harmonics [21, 19]. Anwendungsbereiche der Spherical Harmonics erstrecken sich auf die theoretische Physik bei der Berechnung von Atomorbitalen, oder auf die Geophysik bei der Berechnung des Geoids. Auch in der Informatik werden die Spherical Harmonics eingesetzt, so zum Beispiel bei der Repräsentation von 3D Shapes [26, 14] oder in der Computergraphik zur Berechnung indirekten Lichtes [18].

Um ein 3D Objekt mit Spherical Harmonics ($\mathcal{SH}$) zu repräsentieren, wird das als Volumen vorliegende Objekt als sphärische Funktionen auf Kugeloberflächen definiert und mit den Spherical Harmonics Basisfunktionen in den Spherical Harmonics Raum überführt. Dies stellt sich analog zur Fourier Transformation dar, bei der ein Ausgangssignal durch eine Kombination mehrerer Basisfunktionen repräsentiert wird. Während aber bei der Fourier Transformation die Basisfunktionen nur auf Sinus und Kosinus basieren, sind bei den Spherical Harmonics zusätzlich die Legendrepolynome Teil der Basisfunktionen.

Sei eine Kugeloberfläche S^2 gegeben, die durch die Winkel θ und ϕ als Breiten- und Längengrad auf den kartesischen Koordinaten (x, y, z) parametrisiert wird mit

$$(sin\theta cos\phi, sin\theta sin\phi, cos\theta) \rightarrow (x, y, z) ,$$

und eine auf S^2 definierte Funktion $f(\theta, \phi)$. Dann kann f durch eine Transformation $\mathcal{SH}$ und den Basisfunktionen Y_l^m nach Spherical Harmonics entwickelt werden mit

$$\mathcal{SH} : S^2 \rightarrow \mathbb{C}^{\mathcal{N}}$$

$$f(\theta, \phi) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \hat{f}_{lm} Y_l^m(\theta, \phi) \ , \tag{2.2}$$

wobei $\mathcal{N}$ die Dimension im Spherical Harmonics Raum ist. Die Entwicklungskoeffizienten $\hat{f}_{lm}$ sind definiert als

$$\hat{f}_{lm} = \int_{\theta=0}^{2\pi} \int_{\phi=0}^{\pi} Y_l^{m*}(\theta, \phi) \cdot f(\theta, \phi) \cdot sin\theta \, d\theta d\phi \ , \tag{2.3}$$

und die Kugelflächenfunktionen $Y_l^m : S^2 \rightarrow \mathbb{C}$ als

$$Y_l^m(\theta, \phi) = \sqrt{\frac{2l+1}{4\pi} \frac{(l-m)!}{(l+m!)}} \cdot P_l^m(cos\theta) e^{im\phi}. \tag{2.4}$$

Der Parameter l ist das Band beziehungsweise die Frequenz der Entwicklung, wohingegen der Parameter m die Ordnung der Entwicklung zu einem spezifischem Band l ist. Die Funktionen P_l^m sind die assoziierten Legendrepolynome.

Analog zur Fourier Transformation können die nach Spherical Harmonics entwickelten Ausgangsfunktionen $f(\theta, \phi)$ auf der Kugeloberfläche S^2 über die Rücktransformation $\mathcal{SH}^{-1}$ rekonstruiert werden mit

$$\mathcal{SH}^{-1} : \mathbb{C}^{\mathcal{N}} \rightarrow S^2$$

$$f(\theta, \phi) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \hat{f}_{lm} Y_l^m(\theta, \phi) \tag{2.5}$$

Für eine weitergehende Beschreibung der Spherical Harmonics Entwicklung siehe zum Beispiel [21] oder [18].

Die entwickelten Koeffizienten können als Projektion auf Einheitskugeln mit gewichteten Grau- beziehungsweise Farbwerten dargestellt werden. Abbildung 2.4 zeigt die ersten vier Bänder.

Da die Summe über l in 2.2 aus praktischen Gründen nicht bis nach Unendlich entwickelt werden kann, wird die Entwicklung auf ein maximales Band l_{max} beschränkt. Die Entwicklung 2.2 nach Spherical Harmonics bis zu l_{max} ergibt sich dann als

$$f(\theta, \phi) = \sum_{l=0}^{l_{max}} \sum_{m=-l}^{l} \hat{f}_{lm} Y_l^m(\theta, \phi) \tag{2.6}$$

Die Rücktransformation $\mathcal{SH}^{-1}$ für ein maximales Band ergibt sich analog.

Je höher l_{max} gewählt wird, desto genauer ist die Approximation der sphärischen Funktion durch die Spherical Harmonics Transformation (siehe Abbildung 2.5).

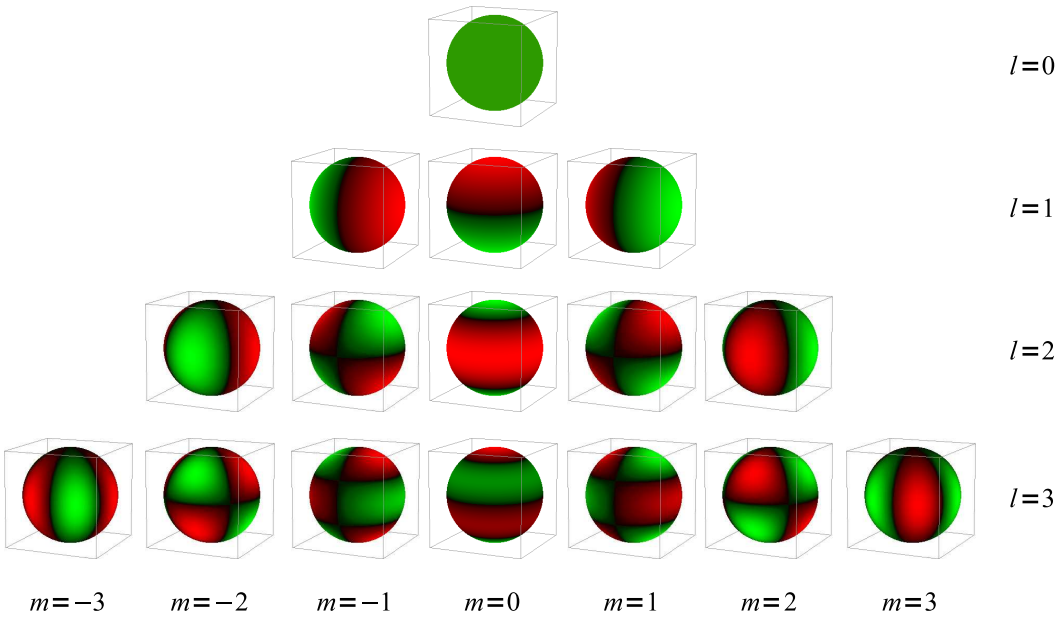


Abbildung 2.4: Visualisierung von Spherical Harmonics Koeffizienten für die ersten vier Bänder ($l = 3$) durch Projektion der Koeffizienten auf Einheitskugeln. Grün (hellgrau) sind positive Werte, rot (dunkelgrau) sind negative.

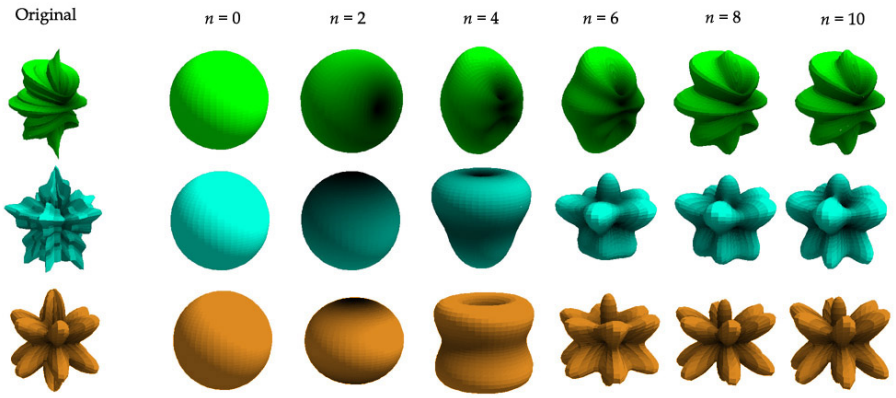


Abbildung 2.5: Entwicklung von drei Beispielfunktionen nach Spherical Harmonics. Für höhere Grade der Entwicklung ergeben sich bessere Approximationen an das Original. (Quelle: [18])

Für ein maximales Band l_{max} ist die Dimensionalität $\mathcal{N}$ der Koeffizienten

$$\mathcal{N} = \sum_{l=0}^{l_{max}} (2l+1) = (n+1)^2. \tag{2.7}$$

2.4 Korrelation und Rotationsschätzung von Kugelflächenfunktionen

Für die Konstruktion neuer Merkmale im Codebook aus mehreren Spherical Harmonics Features ist es notwendig neben einem Ähnlichkeitsmaß die Rotationsparameter zu kennen.

Da die naive Methode der Parameterschätzung mittels Korrespondenzsuche zu rechenintensiv ist, wird auf ein Verfahren von Fehr *et* al. zurückgegriffen, das die Rotationsparameter und das Ähnlichkeitsmaß zweier Spherical Harmonics Signale schnell und akkurat schätzt [15]. Andere Verfahren zur Rotationsschätzung wurden etwa von [5, 31] vorgestellt, doch sind bei diesen die Approximationen unpräziser und auf kleine Rotationen beschränkt.

Das Verfahren von Fehr *et* al. basiert auf einer schnellen Korrelation in der Harmonic Domain [29] und schätzt die Rotationsparameter am Ort der höchsten Korrelation im Frequenzbereich.

Für zwei reellwertige Signale $f, g \in \mathbb{R}$ auf der Einheitskugel S^2 ist das Ziel die Rotationsparameter der Signale zueinander so zu schätzen, dass die Signale möglichst deckungsgleich liegen. Im Folgenden wird die Rotation einer Funktion auf der Einheitskugel S^2 durch die Rotationsparameter $R(\phi, \theta, \psi)$ aus der Rotationsgruppe $SO(3)$ ausgedrückt, wobei ϕ, θ, ψ die Eulerwinkel in x-Konvention mit $ZY'Z''$ sind.

Um die Rotationsparameter zu finden wird zuerst eine Korrelation der Signalwerte durchgeführt. Die Korrelationsfunktion $c : SO(3) \to \mathbb{R}$ ergibt sich als

$$c(R) := \int_{SO(3)} f \cdot (R \cdot g)\, d\phi d\theta d\psi, \tag{2.8}$$

wobei $\varphi, \psi \in [0, 2\pi[$ und $\theta \in [0, \pi[$ die Rotationsparameter in Euler Notation sind. Sie beschreibt die Ähnlichkeit zweier Signale über den Korrelationswert.

Da diese Funktion im Ortsbereich rechenintensiv auszuwerten ist, wird die Korrelation im Frequenzraum (Harmonic Domain) durchgeführt. Das Ergebnis ist nach einer Rücktransformation mittels inverser *Fast Fourier Transform* (FFT^{-1}) eine komplexe Korrelationsmatrix $S \in \mathbb{C}^3$. Jeder Punkt in S gibt die Korrelation zwischen den Voxeln in f und g an jeweils gleichen Stellen an. Im Anschluss werden die Rotationsparameter φ, θ, ψ geschätzt, indem an der Stelle mit der höchsten Korrelation die Winkel abgeleitet werden. Die beiden Signale haben bezüglich dieser ersten Schätzung der Drehwinkel die höchste Ähnlichkeit.

Die mit der Korrelationsfunktion 2.8 ermittelten Winkel stellen nur eine Approximation dar, denn die Qualität der Winkelauflösung hängt direkt von der Auflösung der entwickelten Spherical Harmonics ab, das heißt, dass das maximal entwickelte Band l_{max} Ausschlag gebend ist. Abhängig vom gewählten Band l_{max} wird die Winkgelauflösung im schlechtesten Fall angegeben mit

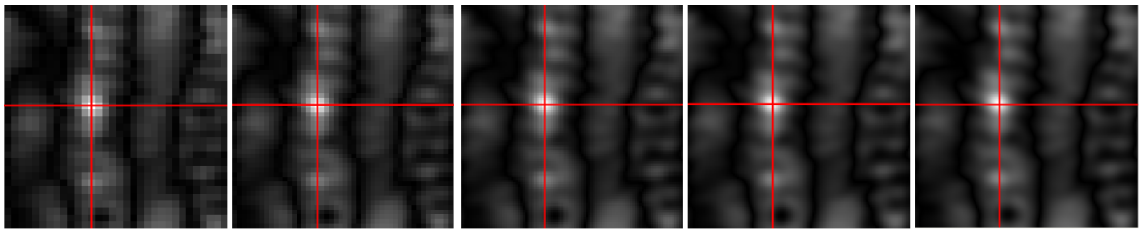


Abbildung 2.6: Erhöhung der Auflösung einer Korrelationsmatrix bei der *sinc*-Interpolation mit *zero-padding* p. Von links nach rechts steigt die Auflösung für $p = 0$, $p = 64$, $p = 128$, $p = 245$. An markierten Stelle befindet sich die Stelle der größten Korrelation und der höchsten Winkelauflösung. (Quelle: Fehr *et* al. [15])

$$\epsilon_{corr} = 2 \cdot \frac{180}{2l_{max}} + \frac{90}{2l_{max}}.$$

Dies bedeutet, dass für eine Spherical Harmonics Entwicklung mit $l_{max} = 10$ die zu erwartende Winkelauflösung im schlechtesten Fall $\epsilon_{corr} = 22,5$ Grad beträgt - eine Genauigkeit von einem Grad kann erst bei einer Entwicklung ab dem 225. Band erreicht werden[2].

Um dennoch eine gute Genauigkeit zu erreichen wird mittels einer *sinc*-Interpolation die Genauigkeit im Frequenzbereich erhöht. Dazu wird die Auflösung der Korrelationsmatrix im Frequenzbereich erhöht, indem eine 3D Erweiterung vorgenommen wird und die entstehenden Zwischenräume der Matrix mit Nullen aufgefüllt werden (*zero padding*). Durch diese einfache Maßnahme kann die Auflösung der Korrelation stark erhöht werden (siehe Abbildung 2.6). Dies reduziert die zu erwartende Winkelauflösung im schlechtesten Fall zu

$$\epsilon_{corr}^{padd} = 2 \cdot \frac{180}{2l_{max} + p} + \frac{90}{2l_{max} + p}.$$

Demnach kann mit der *sinc*-Interpolation mit $p = 430$ eine Winkelauflösung von einem Grad schon bei einer Entwicklung im zehnten Band ($l_{max} = 10$) erreicht werden. Da aber auch die *sinc*-Interpolation Rechenzeit kostet und eine derart genaue Auflösung meist nicht benötigt wird, wurde in dieser Diplomarbeit $p = 70$ und $l_{max} = 10$ gewählt, was einer Winkelauflösung im schlechtesten Fall von $e_{corr} = 5°$ entspricht.

2.5 Morphologie in 3D

Die aus der zweidimensionalen Bildverarbeitung bekannten morphologischen Operatoren wurden auf die Verwendung mit dreidimensionalen Datensätzen erweitert.

Das Konzept der *boundary extraction*, also das Extrahieren der Kontur, ist bei den binären Volumendaten der 3D Modelle äquivalent mit der Extraktion der Oberfläche.

[2] Eine Spherical Harmonics Entwicklung mit einem derart hohen Band ist äußerst rechenintensiv und kann praktisch zu suboptimalen Ergebnissen führen.

(a) Strukturelement 2D (b) Strukturelement 3D (c) Konturschnitt

Abbildung 2.7: Strukturelemente für die Konturfindung bei Bildern (a) und Volumendaten (b). Beispiel am Modell „Hase“ (c).

Die Konturfindung bei Binärbildern mit morphologischen Operatoren ist die Subtraktion des erodierten Bildes vom Originalbild, wobei als Strukturelement eine mit Einsen gefüllte 3×3 Matrix genommen wird.

Nach [16] ist die Kontur β einer Menge A mit dem Strukturelement B definiert als

$$\beta(A) = A - (A \ominus B),$$

wobei der Operator $\ominus$ die Erosion meint mit

$$A \ominus B \quad = \quad \{z|(B)_u \cap A^c = \emptyset\}.$$

$(B)_z$ ist die Translation einer Menge B mit einem Punkt $z = (z_1, z_2, z_3)$ und wird definiert als $(B)_z = \{c|c = b + z,\, b \in B\}$, A^c ist das Komplement der Menge A und $\emptyset$ die leere Menge.

Für Mengen von Daten in 3D wird für die Konturfindung die mit Einsen gefüllte $3 \times 3 \times 3$ Matrix B benutzt (siehe Abbildung 2.7).

2.6 Clustering

Für die Erstellung des Codebuches werden Cluster-Verfahren genutzt. Ziel ist es, ein universell gültiges Codebuch zu finden, das einen reduzierten Satz an „universellen“ Features enthält.

Beim Clustering wird eine Menge von zunächst unstrukturierten Eingabedaten durch Konstruktion homogener Klassen (*Cluster*) optimal strukuriert, das heißt in Gruppen mit möglichst ähnlichen Elementen gruppiert.

Bei den Cluster-Verfahren werden die zwei Arten partitionierendes und hierarchisches Clustering unterschieden. Auf beide Arten wurde in dieser Arbeit zurückgegriffen, und auf beide wird im Folgenden kurz eingegangen.

Zu den benutzten partitionierenden Cluster-Verfahren zählt der *Expectation Maximization* (*EM*) und der k-Means Algorithmus. Beide teilen die Eingabedaten in eine a priori

bekannte Anzahl Cluster auf. Dieser Typ des Clusterings wird sowohl bei der Merkmalsextraktion als auch beim Clustering des Codebuches angwendet.

Bei den hierarchischen Cluster-Verfahren wurde ein anhäufendes Verfahren (*agglomerative clustering*) zur Erstellung des Codebuches verwendet. Im Gegensatz zu den teilenden Cluster-Verfahren (*divisive clustering*), die vom gesamten Datensatz ausgehend diesen in kleinere Cluster aufteilen, sind beim agglomerativen Clustering alle Elemente in einem Cluster und es werden sukzessive diese Cluster zu größeren Gruppierungen verbunden.

Notwendige Bedingung für alle Cluster-Verfahren ist, dass die Ähnlichkeit beziehungsweise die Unähnlichkeit zwischen zwei Elementen bestimmt werden kann. Hier kommen in der Regel Distanzfunktionen zum Einsatz, die auf der Minkowski-Distanz beruhen (siehe Formel 2.1 in Abschnitt 2.1).

2.6.1 k-Means basiertes Clustering

Der wohl bekannteste Vertreter der partitionierenden Cluster-Verfahren ist der k-Means Algorithmus. Dem Namen entsprechend basiert dieser Algorithmus auf der Bestimmung der Mittelwerte.

Der Algorithmus wird auf dem Datensatz mit k zufällig gewählten Zentren initialisiert. Die einzelnen Merkmalswerte im Merkmalsraum werden gemäß ihrer Distanz dem nächstgelegenen Cluster-Zentrum (*Centroid*) zugewiesen. Aufgrund dieser neuen Zuordnungen werden die Positionen der Zentren neu berechnet.

Diese beiden Schritte, die Zuordnung anhand der Cluster-Zentren und die Neupositionierung der Zentren, werden so lange iteriert, bis das Verfahren konvergiert. Ein mögliches Konvergenzkriterium ist, wenn sich die Clusterzuordnungen nicht mehr verändern, das heißt wenn die k Cluster stabil bleiben.

Wegen seiner Einfachheit und nachvollziehbaren Ergebnissen ist der k-Means sehr populär. Zu den Nachteilen allerdings gehört, dass die Anzahl k der zu erwartenden Cluster a priori bekannt sein muss. Auch ist das Ergebnis stark von der Initialisierung der Startwerte abhängig, da diese beeinflussen, ob k-Means in einem lokalen Minimum bezüglich den Punkt-zu-Centroid Distanzen stecken bleibt.

Ein Ansatz diese Probleme zu umgehen besteht darin ein wiederholtes k-Means anzuwenden, was unter *repeated* k-Means bekannt ist. Auf denselben Daten wird der k-Means Algorithmus mehrmals mit zufälligen Anfangswerten initialisiert. Am Ende wird das Ergebnis desjenigen Durchlaufs genommen, dessen Punkt-zu-Centroid Distanzen minimal sind. Dies orientiert sich an der Überlegung dasjenige Clustering Ergebnis auszuwählen, das das kleinste empirische Risiko trägt (siehe Vapnik *et* al. [48]). Dennoch ist nicht garantiert, dass ein globales Optimum gefunden wird [53].

In dieser Diplomarbeit wurden auch k-Means basierte Cluster-Verfahren untersucht, die eine automatische Bestimmung der Cluster-Anzahl k liefern, und die auf reduzierten Eingangsdaten ablaufen. Die Reduktion der Eingangsdaten kann etwa über eine Singulärwertzerlegung oder eine Hauptachsenanalyse [11] geschehen. Für einen kleinen Datensatz sind die Clustering-Ergebnisse exemplarisch in Abbildung 2.8 abgebildet.

Ein k-Means basierter Algorithmus, der die Anzahl der Cluster automatisch schätzt ist der von D. Pelleg [37] entwickelte X-Means Algorithmus. Die Idee bei X-Means ist, dass anhand eines quantitativen Maßes, nämlich dem *Bayesian Information Criterion* (BIC), entschieden wird, ob eine Menge von Punkten weiter aufgeteilt werden soll. Dadurch wird

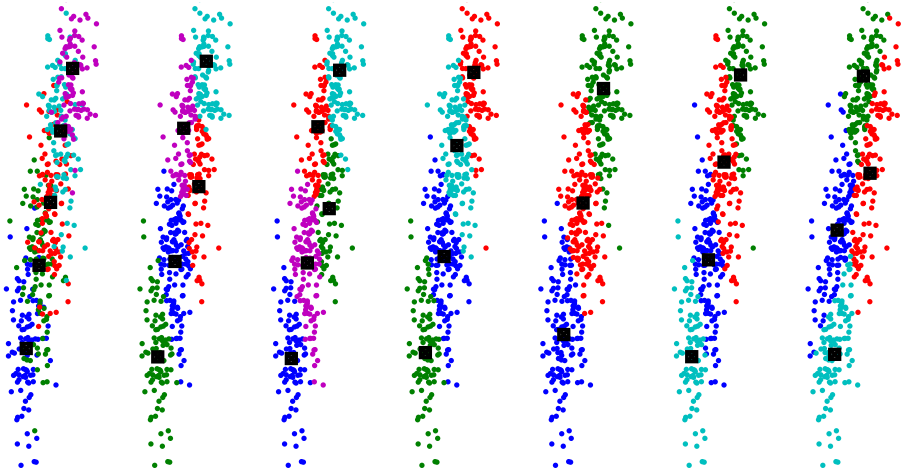


Abbildung 2.8: Vergleich verschiedener k-Means basierter Cluster-Verfahren in Kombination mit Wiederholung und Dimensionsreduktionstechniken. Die Centroids sind als schwarze Boxen markiert. Die Farben sind willkürlich und nur für die Trennung der Klassen eingezeichnet.
Von links nach rechts: (1) Ground-Truth mit 5 Cluster; (2) k-Means mit vorgegebenem k; (3) wie (2) aber zusätzlich mit PCA; (4) *repeated k-Means* (findet $k = 4$ Cluster); (5) *repeated k-Means* mit PCA (findet $k = 4$ Cluster); (6) und (7) sind X-Means mit und ohne PCA (beide finden $k = 4$ Cluster).

erreicht, dass nur die vielversprechendsten Teilmengen aufgeteilt werden. Der Algorithmus ist im Vergleich mit *repeated* k-Means schneller und liefert gute Ergebnisse, was die Anzahl der Cluster anbelangt (siehe das sechste Bild von rechts in Abbildung 2.8).

Sind die Ausgangsdaten zu groß, so kann mittels Methoden zur Dimensionsreduktion der Datensatz reduziert werden. Ding *et* al. [11] schlagen einen k-Means Algorithmus vor, bei dem die Daten mit einer Principal Component Analysis reduziert wurden. Sie zeigen, dass die Ergebnisse unabhängig von der Reduktion der Daten sind (siehe das dritte Bild in Abbildung 2.8).

2.6.2 Expectation Maximization (EM)

Der Expectation Maximization (EM) Algorithmus wurde erstmals 1977 von Dempster *et* al. beschrieben [10]. Er basiert auf der Annahme, dass die Verteilung der zu clusternden Datenpunkte (Instanzen) zufällig ist. Ziel ist es die Parameter der unbekannten Wahrscheinlichkeitsverteilungen und damit die Cluster zu bestimmen. Ist die Art der Wahrscheinlichkeitsverteilung bekannt, kann der EM-Algorithmus auf Variablen mit nur teilweise bekannten beziehungsweise noch nicht beobachteten Instanzen angewendet werden; das Clustering-Problem kann online gelöst werden.

Ähnlich wie bei k-Means werden die Cluster-Zuordnungen iterativ aktualisiert. Statt aber die Objekte genau einem Cluster zuzuordnen, wird jedem Datenpunkt eine Wahrscheinlichkeit bezüglich seiner Mitgliedschaft in allen Cluster zugeordnet. Dadurch beeinflusst jeder Datenpunkt die Parameter jedes Clusters entsprechend seiner Wahrscheinlichkeit.

Der EM-Algorithmus sucht nach der maximum likelihood Hypothese h', die den Erwartungswert $E[lnP(Y|h')]$ auf der Gesamtmenge Y der Daten maximiert. Dadurch ist es möglich auch bei unbekannten Daten die Verteilung der Gesamtdaten zu schätzen. Der EM-Algorithmus nutzt dazu die aktuell bekannte Hypothese h anstelle der unbekannten Parameter für die Schätzung der tatsächliche Verteilung auf Y.

Das Schätzen und Verfeinern der gesuchten Parameter ist unterteilt die zwei Schritte *Expectation* und *Maximization* wird iterativ so lange ausgeführt bis die Wahrscheinlichkeitsfunktion $P(Y|h')$ konvergiert:

1. Beim *Expectation* (E) Schritt werden die aktuelle Hypothese h und die beobachteten Daten X benutzt, um anhand einer Funktion $Q(h'|h)$ und den beobachteten Daten X eine Wahrscheinlichkeitsverteilung bezüglich Y zu schätzen. Dabei wird angenommen, dass die unbekannten Parameter θ für die tatsächliche Verteilung gleich sind wie die aktuelle Hypothese:

$$Q(h'|h) \leftarrow E[lnP(Y|h')|h, X]$$

2. Beim anschließenden *Maximization* (M) Schritt werden diejenigen Hypothesen h durch die neue Hypothese h' ersetzt, die Q maximieren:

$$h \leftarrow \arg\max_{h'} Q(h'|h)$$

Durch das Modell über Wahrscheinlichkeiten wirken sich fließende Übergänge oder Rauschen zwischen den Datenpunkten weniger störend auf die Cluster-Zuordnungen aus. Dies ist von Vorteil wenn die Datenpunkte relativ gleichmäßig verteilt sind und die Cluster als Regionen mit erhöhter Datenpunktdichte auftreten.

Der EM-Algorithmus muss wie k-Means mit einer vorgegebenen Anzahl n von Clustern initialisiert werden, wobei diese n-Wahrscheinlichkeitsverteilungen darstellen und jede für einen Cluster steht.

Die in dieser Diplomarbeit benutzte Version des EM-Algorithmus schätzt die Cluster-Anzahl n automatisch anhand der logarithmischen Wahrscheinlichkeit (*log-likelihood*). Als Implementierung wurde auf den Algorithmus im frei verfügbaren *Weka* Machine-Learning Paket zurückgegriffen [53].

2.6.3 Agglomeratives Clustering

Das agglomerative Clustering [24] zählt zu den hierarchischen Cluster-Verfahren. Anstatt die Datenpunkte wie bei k-Means schrittweise zu unterteilen, werden bei den anhäufenden Cluster-Verfahren die Datenpunkte schrittweise anhand eines Schwellenwertes bezüglich ihrer Distanzen zu neuen Clustern verschmolzen. Bei diesem *top-to-bottom* Verfahren stellt sich die gelernte Struktur als Baum in Form eines Dendrogramms dar. Die hierarchische Zerlegung zeigt sich in immer kleineren Teilmengen an den Knoten und als Cluster an den Blättern.

Das agglomerative Clustering startet, indem jeder Datenpunkt einem eigenen Cluster zugeordnet wird. Während jeder Iteration werden die zwei ähnlichsten Instanzen verbunden.

Die Ähnlichkeit definiert sich hierbei über das verwendete Distanzmaß. Der Algorithmus stoppt, wenn entweder ein Distanzschwellenwert erreicht wurde, oder wenn keine weiteren Datenpunkte mehr vorliegen. Die Anzahl der Cluster kann somit über die Anpassung des Schwellenwertes kontrolliert werden.

Für das agglomerative Clustering gibt es eine Reihe von Methoden zur Verbindung von Instanzen. In der Literatur sind diese unter der Bezeichnung *Linkage* (Verbindung) bekannt. Die Methoden unterscheiden sich maßgeblich durch die zugrunde liegende Berechnung der Distanz. Während die einfachste Methode, das *Single Linkage*, die kürzeste Distanz als Maß nimmt, werden bei aufwändigeren Verbundmethoden, zum Beispiel beim *Ward's Linkage*, komplexere Distanzmaße genommen.

Im Folgenden werden einige Methoden vorgestellt, wobei die folgende Notation verwendet wird: Cluster c wird aus den Clustern p und q durch Verbund geformt, n_c ist die Anzahl der Elemente in einem Cluster c, x_{ci} ist das i-te Element im Cluster c, und $d(p,q)$ ist die Distanz zwischen zwei Clustern p und q. Mit $dist(x,y)$ ist die Distanz zwischen zwei Elementen gemeint. Diese Distanz wird durch eine Distanzmatrix vorgegeben und muss beim Linkage nicht nochmal berechnet werden.

Single Linkage Der minimale Abstand zweier Elemente aus zwei Clustern:

$$d(p,q) = min\{dist(x_{pi} - x_{qj})\} \qquad i \in \{1,...,n_p\}, j \in \{1,...,n_q\}$$

Single Linkage ist äquivalent mit der Nächster-Nachbar-Klassifikation und demnach sehr schnell, da billig in der Berechnung.

Complete Linkage Im Gegensatz zum Single Linkage wird hier der maximale Abstand zwischen zwei Elementen aus zwei Clustern genommen.

$$d(p,q) = max\{dist(x_{pi} - x_{qj})\} \qquad i \in \{1,...,n_p\}, j \in \{1,...,n_q\}$$

Average Linkage Der Mittelwert der paarweisen Distanzen aller Elemente in beiden Clustern ergibt die Distanz:

$$d(p,q) = \frac{1}{n_p n_q} \sum_{i=1}^{n_p} \sum_{j=1}^{n_q} dist(x_{pi}, x_{qj})$$

Centroid Linkage Der Abstand der Mittelwerte $\bar{x}_p$ und $\bar{x}_q$ (Centroids) der beiden Cluster:

$$d(p,q) = |\bar{x}_p - \bar{x}_q| \ ,$$

für $\bar{x}_p = \frac{1}{n_p} \sum_{i=1}^{n_p} x_{pi}$.

Ward's Linkage Die Zunahme der Varianz beim Vereinigen der beiden Cluster:

$$d^2(p,q) = n_p n_q \frac{|\bar{x}_p - \bar{x}_q|^2}{n_p + n_q}$$

Bei allen Methoden steigt die Distanz zwischen den Vereinigungsschritten streng monoton. Als Abbruchkriterium kann entweder die Anzahl der bislang verbundenen Cluster oder die Distanz dienen, bei letzterem wenn die Distanz zwischen zwei zu vereinigenden Clustern einen Schwellenwert übersteigt.

2.7 Clustering Evaluation

Die Evaluation von Clustering Ergebnissen ist nicht ohne weiteres möglich, da die tatsächliche Klassifizierung per se meist nicht bekannt ist, sondern schließlich durch das Clustering gefunden werden soll. Ist eine tatsächliche Klassifizierung (*Ground-Truth*) gegeben, so ließe sich anhand dieser das Cluster-Verfahren für den gegebenen Datensatz bewerten. Eine andere Möglichkeit besteht darin, das Clustering-Ergebnis erst im Folgenden, anhand eines darauf aufbauenden, anderen Ergebnisses im Verarbeitungsprozess zu bewerten. Zum Beispiel sind die Clustering-Ergebnisse die Grundlage für das Codebuch, die Evaluation des Verfahrens kann aber erst am Ende des Verarbeitungsprozesses bei der Evaluation des Retrievals geschehen, das wiederum direkt vom Codebuch abhängt.

In der Regel ist keine Ground-Truth gegeben und eine Evaluation muss bereits nach dem Clustering erfolgen. Eine Möglichkeit ist, dass ein Experte sich die Ergebnisse in Form eines Dendrograms visualisieren lässt und die Cluster-Zuordnungen daran überprüft. Eine andere Möglichkeit besteht in einer qualitativen Beurteilung des Clusterings anhand der gebildeten Cluster im Bezug zur Distanzmatrix. Dies ist beim kophenetischen Korrelationskoeffizient der Fall. Das Dendrogramm und der kophenetische Korrelationskoeffizient, werden im Folgenden kurz vorgestellt.

2.7.1 Dendrogramm

Ein hierarchisches Cluster-Verfahren kann wegen seiner hierarchischen Struktur als Baum visualisiert werden. Für das agglomerative Clustering ergibt sich ein sogenanntes Dendrogramm, das eine gewichtete Taxonomie der Datenpunkte zeigt (siehe Abbildung 2.9). Die Gewichtung zeigt sich anhand des Abstandes der Knoten eines Pfades bezüglich jeder Ebene. Ein Dendrogramm spiegelt die hierarchischen Verschmelzungen der Distanzmatrix wider, wobei die Verschmelzungknoten in Entfernung ihrer Distanz zum Vorgängerknoten versetzt aufgetragen werden. Die Höhe der Verknüpfung im Cluster-Baum gibt so die Distanz der verknüpften Elemente an.

2.7.2 Kophenetischer Korrelationskoeffizient

Eine Möglichkeit ein Dendrogramm zu beurteilen ergibt sich über die Bestimmung des kophenetischen Korrelationskoeffizienten [43]. Die kophenetische Korrelation ist die Korrelation der tatsächlichen Unähnlichkeiten, wie sie durch die Distanzmatrix gegeben sind,

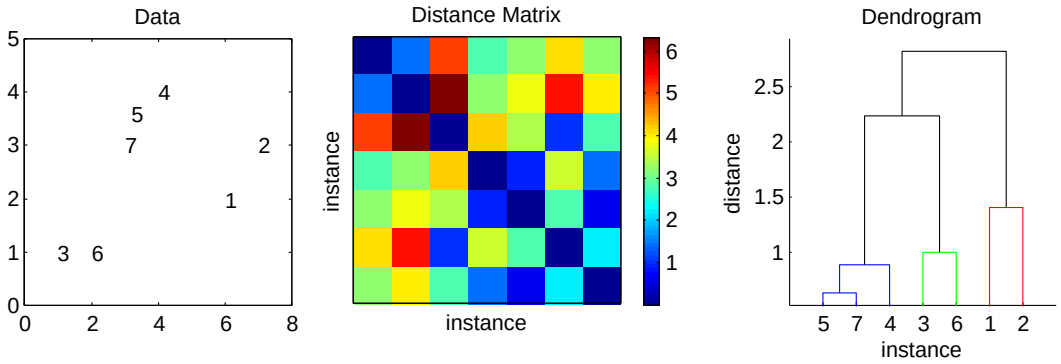


Abbildung 2.9: Beispiel für eine Distanzmatrix und Dendrogramm nach Single-Linkage Clustering.

und der Unähnlichkeiten, wie sie im Dendrogramm dargestellt werden. Der kophenetische Korrelationskoeffizient gibt also an, wie gut ein Dendrogram die tatsächlichen Daten modelliert.

Die kophenetische Distanz gibt an, wie groß der Abstand zweier Knoten im Dendrogramm ist. Um nun den Korrelationskoeffizienten zu bestimmen, werden diese Distanzen mit den Distanzen der Originaldaten korreliert. Für ein gutes Clustering-Ergebnis gilt, dass die Verknüpfungen der Elemente im Dendrogramm stark mit den Distanzen zwischen den Objekten in der Distanzmatrix korrelieren.

Für eine Distanzmatrix D und dem ermittelten Dendrogramm T berechnet sich der kophenetische Korrelationskoeffizient als

$$c = \frac{\sum_{i<j}(D_{ij} - \bar{d})(Z_{ij} - \bar{z})}{\sqrt{\sum_{i<j}(D_{ij} - \bar{d})^2 \cdot \sum_{i<j}(Z_{ij} - \bar{z})^2}} \quad ,$$

wobei D_{ij} der Eintrag in der Distanzmatrix für die Objekte i und j ist, Z_{ij} die kophenetische Distanz zwischen i und j ist, und $\bar{d}$ und $\bar{z}$ die Mittelwerte der Distanzmatrix respektive der kophenetischen Distanzen sind. Für ein gutes Clustering-Ergebnis nähert sich dieses Maß dem Wert 1 an.

Beispiel Für das Beispiel in Abbildung 2.9 ergibt sich für unterschiedliche Linkage-Methoden, bei mittlerer Distanz $\bar{d} = 3.13$, folgende Werte:

Linkage-Methode	$\bar{z}$	c
simple	1.5	0.824
complete	2.46	0.777
average	1.92	0.827
centroid	1.87	0.826
ward	2.63	0.826

Kapitel 3

Merkmale

Merkmale (*engl.* Features) parametrisieren ein Eingangssignal, indem sie das Signal in einer abstrahierten und meist konzentrierten Form beschreiben. Ein hochdimensionales Signal, wie es in der Bildverarbeitung oder auch hier im Bereich der dreidimensionalen Mustererkennung vorliegt, wird auf eine weniger dimensionierte Beschreibung reduziert. Im Vergleich mit anderen Techniken zur Dimensionsreduktion, wie der Singulärwertzerlegung oder der Hauptkomponentenanalyse (*Principal Component Analysis*), ist hier aber die wichtige Unterscheidung zu treffen, dass die Reduktion nicht bijektiv ist und durch den Abstraktionsprozess Informationen verloren gehen.

In diesem Kapitel werden zuerst die notwendigen Vorüberlegungen zur Erzeugung von Merkmalen formuliert. Darauf aufbauend folgt der Teil zur Detektion von Interest Points und der Konstruktion lokaler Merkmale.

Die Anforderungen an die zu extrahierenden Merkmale sind, dass sie

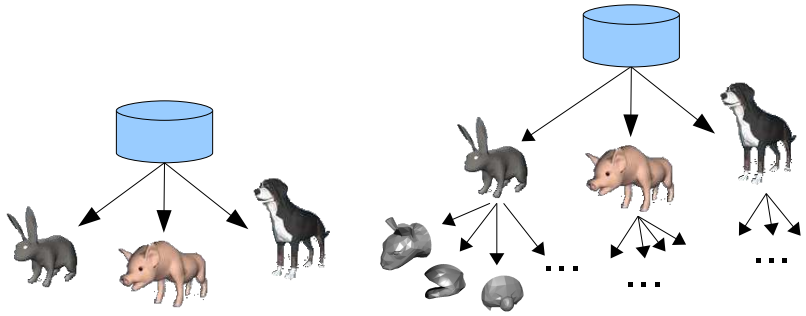
1. wohldefiniert sind, das heißt, dass jedes erzeugte Merkmal genau einen Punkt im Musterraum beschreibt, und dass

2. mit ihnen eine hohe Wiederholungsrate (Repeatability) erreicht wird. Gleiche Merkmale müssen an gleichen Punkten wiedergefunden werden. Auch müssen sie robust gegenüber gestörten Daten sein, wie es zum Beispiel bei verrauschten Daten der Fall ist.

3. Auch wird gefordert, dass sie invariant gegenüber den meisten Transformationen (Rotation, Skalierung, Translation, etc.) sind.

Ein weiterer Anspruch an die Merkmale ist, dass sich mit ihnen das Korrespondenzproblem optimal lösen lässt. Als Korrespondenzproblem wird das Problem bezeichnet, in zwei getrennten Eingangssignalen diejenigen Signalpaare zu finden, die auf denselben Wert zeigen. Übertragen auf die dreidimensionalen Modelle bedeutet dies, dass auf zwei identischen Volumina identische Features erzeugt werden, die, bezogen auf das Korrespondenzproblem, eindeutig zugeordnet werden können. Im Gegenzug bedeuet dies aber auch, dass verschiedene Signale anhand der Features separiert werden können.

Die meisten Feature-basierten Verfahren zur 3D Ähnlichkeitssuche konzentrieren sich auf Vergleichsverfahren, die Eigenschaften von globalen Features nutzen. Globale Features werden auf dem gesamten Objekt erzeugt und repräsentieren auch das Objekt im Gesamten. Im Gegensatz dazu stehen lokalen Merkmale, wie sie in dieser Diplomarbeit auf 3D Shapes untersucht wurden.



(a) Objektbasierte Merkmale (b) Teilbasierte Merkmale

Abbildung 3.1: Erweiterte Klassifikationstiefe der Klassifikation bei lokalen Features.

3.1 Lokale Merkmale

Mit lokalen Merkmale (lokale Features) können partielle Daten beschrieben werden. Statt wie bei globalen Merkmalen das Objekt nur mit einem einzigen Feature zu repräsentieren, werden auf dem Objekt an unterschiedlichen Stellen Features erzeugt. Die Verteilung der Features auf dem Objekt kann zufällig und gleichmäßig (Normalverteilung) erzeugt werden, oder an Stellen, die für das Objekt charakteristisch sind. Zum Beispiel könnten in einem Foto mit einer Sonnenblumenwiese die lokalen Merkmale an den dunklen Stellen der Blütenkörbe gefunden werden. Übertragen auf 3D sind zum Beispiel beim Modell einer Ameise die charakteristischen Stellen an den Beinen und Gelenken, den Fühlern, und an Kopf, Thorax oder Abdomen zu finden. Die Verteilung der Features an mehreren Punkten im Raum ist eine notwendige Bedingung für das partielle Objektretrieval.

Lokale Features erweitern die Anforderung der Robustheit gegenüber Signalstörungen auf eine Robustheit gegenüber lokal verfälschten Signalen, wie sie bei fehlerhafter Erfassung oder Teilüberlagerung mit anderen Signalen (Überdeckung) entstehen können. Während bei lokalen Features bei derartigen Signaländerungen nur wenige oder sogar nur ein einziges Feature verfälscht wird, sind bei globalen Features die Auswirkungen im gesamten Merkmal zu beobachten, was das Feature im Gesamten verfälscht. Dies wirkt sich auf die zweite Anforderung aus, weil Merkmale nicht wiedergefunden werden können.

Ein weiterer Aspekt von lokalen Features ist, dass sie eine flexiblere Klassenrepräsentation erlauben. Dadurch, dass nur Teile eines Objekts extrahiert werden, können gleichartige Strukturen über Objektgrenzen hinweg gruppiert werden, was die Granularität der Klassifizierungsvorschrift erhöht, wie in Abbildung 3.1 schematisch dargestellt.

Der Konstruktion der Merkmale geht die Merkmalsextraktion (engl. *Feature Extraction*) voraus. Die in dieser Arbeit angewendete Merkmalsextraktion folgt dem bekannten, modularen Aufbau aus der Mustererkennung (siehe Abbildung 3.2). Sie ist der erste Schritt nach der Digitalisierung und Vorverarbeitung der Originaldaten aus dem Objektraum, wobei in der vorliegenden Arbeit die vorverarbeitenden Schritte übersprungen werden konnten, da die Eingangsdaten als rauschfreie Daten vorlagen.

In Anlehnung an die bekannten Verfahren zur Erzeugung von Features im zweidimensionalen Fall werden hier Verfahren zur Feature Extraktion in 3D vorgestellt. Ähnlich wie in 2D ist die Merkmalsextraktion zweistufig aufgebaut:

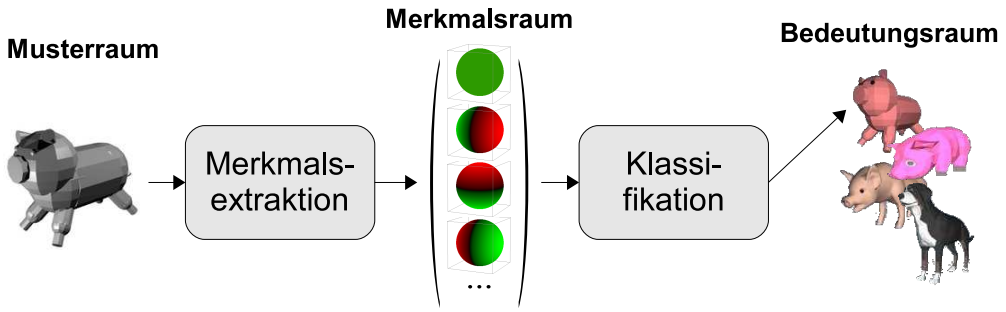


Abbildung 3.2: Allgemeines Schema zur Mustererkennung mit Merkmalen.

1. Suche von charakteristischen Punkten (engl. *Interest Points*) auf den Daten.

2. Verknüpfung der Interest Points mit Merkmalsdeskriptoren (engl. *Feature Deskriptor / Feature Vector*)

Der Feature-Vektor beschreibt die nähere Umgebung eines Interest Points. Diese Kombination aus der gefundenen Stelle und dem beschreibenden Feature Deskriptor wird im Folgenden als Merkmal beziehungsweise Feature bezeichnet.

Der in Abbildung 3.2 dargestellte Schritt Merkmalsextraktion wird in diesem Kapitel formuliert.

Feature-Deskriptor

Die Repräsentation der Features als sogenannten Feature-Vektor ist der Standardansatz für das multimediale Retrieval [12]. Es wird in der Literatur oft als das *Feature-Vektor Paradigma* bezeichnet, da damit die Feature-Repräsentation auf effiziente und einfache Datenstrukturen (Vektoren) abgebildet werden können, die gleichsam einen effizienten Vergleich zweier Features erlauben. Ein Feature-Vektor wird auch als Feature-Deskriptor bezeichnet, da er die durch ihn assoziierte Struktur deskriptiv widerspigelt.

Ein Feature-Vektor hat im Allgemeinen die folgende Form:

$$\vec{x} \in \mho,$$

wobei als $\mho$ meist der n-dimensionale Vektorraum $\mathbb{R}^n$ benutzt wird, so dass $\vec{x} \in \mathbb{R}^n$ ist.

In 2D ist die Bildung von Feature-Vektoren ein populäres Konzept. Bei der *Scale Invariant Feature Transform* (SIFT) [30] zum Beispiel wird die lokale Umgebung an Interest Points durch einen 128-dimensionalen Feature-Vektor beschrieben, indem die Eigenschaften lokaler Umgebungen in Histogrammen aufgetragen und daraus die Feature-Vektoren gebildet werden. In dieser Diplomarbeit stellen die reellwertigen Codebuchhistogramme die Feature-Vektoren dar.

Für den paarweisen Vergleich verschiedener Feature-Vektoren ist es notwendig eine Metrik zu definieren. Im Allgemeinen wird hierzu eine Distanzfunktion δ definiert mit

$$\delta : \; V_1 \times V_2 \rightarrow \mathbb{R}_0^+,$$

wobei $V_1 = \{\vec{x}_1, ..., \vec{x}_n\}$ die Features des ersten Objektes und $V_2 = \{\vec{y}_1, ..., \vec{y}_n\}$ die Features des zweiten Objektes sind. Kleine Werte von δ bedeuten eine starke Ähnlichkeit (kleine Distanz) und große Werte einen starken Unterschied; sind die Features hinreichend eindeutig kann für $\delta = 0$ die Identität festgestellt werden.

Für die bei den Features verwendete Dimensionalität n ist anzumerken, dass zwar einerseits die Beschreibung durch die Features mit höherer Dimensionalität steigt, weil mehr und dadurch genauere Informationen im Feature-Vektor abgelegt werden. Andererseits sinkt für hochdimensionale Datenstrukturen die Performance stark ab. Dies ist unter dem *Fluch der Dimensionalität* bekannt [4].

Um ein Objekt mittels Features zu beschreiben, die im Retrieval-Prozess effizient verarbeitet werden können, ist es daher von Bedeutung, dass sie das Objekt mit möglichst wenig Information möglichst eindeutig beschreiben, aber auch, dass ein Objekt durch so wenige Features wie nötig beschrieben werden kann. In dieser Arbeit ist dies für die Codebuchgröße von Bedeutung, die sich durch die Anzahl der eingesetzten lokalen Merkmale definiert.

Die Anzahl der Codebucheinträge hängt direkt davon ab, wie viele lokale Merkmale auf den Objekten extrahiert werden, beziehungsweise wie viele Stellen pro Objekt jeweils betrachtet werden. Dies steuert die Anzahl der Interest Points, auf die im Folgenden eingegangen wird.

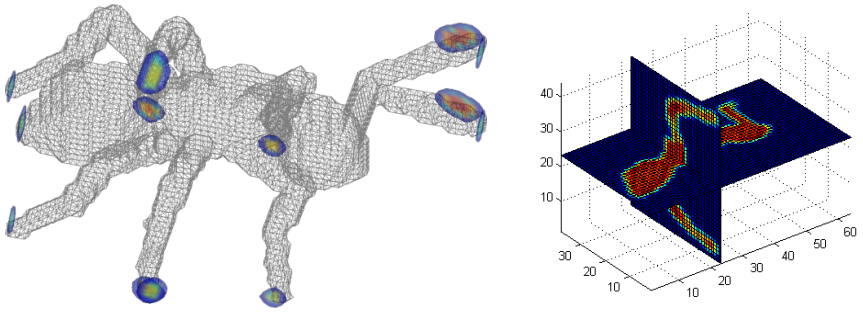
3.2 Interest Points

Um in den Volumendaten Interest Points zu finden, wurden verschiedene Verfahren untersucht. Neben dem naiven Ansatz auf der Objektoberfläche gleichmäßig verteilte Punkte zu suchen, wurden Algorithmen zur Bestimmung von Punkten an charakteristischen Stellen entwickelt, die auf der Entwicklung nach Spherical Harmonics (siehe 2.3) basieren. Als Vorbild dienen Verfahren aus der 2D Bilderkennung, wo lokale Features an speziell ausgewählten Stellen extrahiert werden, indem zuvor der Musterraum in einen neuen Raum überführt wird und dort die Interest Points extrahiert werden. Beispielsweise wird in 2D auf eine Überführung in den Skalierungsraum (*Scale Space*) zurückgegriffen, während hier die Überführung in den Raum der Kugelflächenfunktionen vollzogen wird. Ein prominentes Beispiel in 2D für die Überführung in den Skalierungsraum ist die *Scale Invariant Feature Transform* (SIFT) [30] bei der die Interest Points an Extremstellen im Skalierungsraum extrahiert werden.

Im Folgenden werden zwei entwickelte Verfahren detailliert ausgeführt, die auf der Verwendung von Spherical Harmonics basieren und sich Verfahren aus der Morphologie in 3D zu Nutze machen, um auf der Oberfläche der Objekte gleichmäßig verteilte Punkte zu finden.

3.2.1 Phasenbasierte Features

Die später in dieser Arbeit vorgestellten Verfahren zur Suche nach Interest Points nutzen phasenbasierte Spherical Harmonics Features, die aus der Phase von Spherical Harmonics extrahiert werden, und die an charakteristischen Stellen auf einem Modell Werte um den Mittelwert annehmen.



(a) Selektierte Stellen (b) Schnitt

Abbildung 3.3: Spherical Harmonics Phasen-Features

Die phasenbasierten 3D Textur Features ($\mathcal{SH}_{phase}$-Features) wurden von Fehr *et* al. [13] entwickelt. Diese Methode extrahiert rotations- und grauwertinvariante Features auf 3D Volumendaten im Spherical Harmonics Raum.

Bezogen auf die Sichtweise der Mustererkennung, gilt für das Signal eines Modells die Tatsache, dass wie bei der Fourier Entwicklung auch, die charakteristische Information dominant in der Phase des Signalspektrums ist, statt im Betrag der Koeffizienten [13]. Ähnlich wie bei Verfahren zur Kantendetektion in 2D (Harris-Laplace, Difference of Gaussian, Laplacian of Gaussian, etc.) wird mit den Phasen-Features die Eigenschaft ausgenutzt, dass bei Frequenzsprüngen die Features charakteristische Werte annehmen. Da die Phasen-Features auf der Entwicklung nach Spherical Harmonics beruhen, und diese wiederum ein Analogon zur Fourier Transformation darstellen, ist die höchste Energie im Bereich um den Mittelwert zu finden.

Als Beispiel sind in Abbildung 3.3 die hohen Energien der Phasen-basierten Features eingezeichnet. Die im Folgenden aufgeführten Methoden zur Extraktion von Features an charakteristischen Punkten nutzten diese Eigenschaft der Phasen-Features aus.

3.2.2 Clustering-basierte Interest Points

Die Idee hinter diesem Verfahren ist, dass die binären Volumendaten in reelle Werte transformiert werden, so dass anschließend mit einem Clustering-Ansatz Regionen gleicher Art gefunden werden können.

Das Verfahren ist mehrstufig aufgebaut:

1. Entwicklung nach Spherical Harmonics.

2. Punktreduktion anhand der Phase der Spherical Harmonics und äquidistantes Sampling von Punktkandidaten.

3. Clustering der Korrelationsmatrix.

4. Clustering der Punktregionen gleicher Art.

Im Folgenden werden die einzelnen Schritte erläutert.

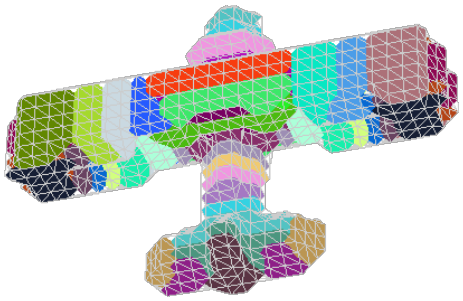


Abbildung 3.4: Punktkorrespondenzen nach Korrelation der Spherical Harmonics entwickelt an jedem Oberflächenpunkt.

Entwicklung nach Spherical Harmonics Im ersten Schritt werden die binären Volumendaten mittels einer Entwicklung nach Spherical Harmonics in eine Darstellung mit komplexen Koeffizienten überführt.

$$V \in \{0,1\} \rightarrow \hat{V} \in \mathbb{C}$$

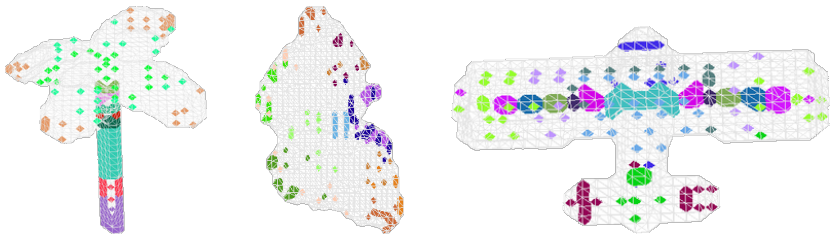
Die Berechnung der Spherical Harmonics kann mit einer schnellen Faltung im Frequenzbereich realisiert werden.

Punktreduktion Für die Eintragung der Features im Codebuch muss eine Punktreduktion vorgenommen werden. Diese Reduktion muss derart sein, dass die Features nur noch an interessanten Stellen im Objekt zu finden sind und Redundanzen somit eliminiert werden. Dazu wird die Eigenschaft ausgenutzt, dass die phasenbasierten Spherical Harmonics an Stellen mit interessanten Eigenschaften, um Beispiel Kanten, Werte um den Mittelwert annehmen. Da am Mittelwert an sich noch nicht die interessanten Stellen zu finden sind, beziehungsweise zu wenige Punkte sich sammeln, wurde eine Heuristik entwickelt, die sich schrittweise an den Mittelwert annähert. Dadurch kann eine vorgegebene Anzahl von Interest Points gesucht werden.

Punktkorrelation An den zurückbleibenden Stellen werden Spherical Harmonics auf dem Objekt entwickelt und für alle Stellen paarweise die Korrelationswerte (siehe Abschnitt 2.4) bestimmt. Diese Werte können in einer Korrelationsmatrix aufgetragen werden, in der die Punktkorrespondenzen - also die Ähnlichkeiten der Spherical Harmonics Entwicklungen an jeder Stelle im Objekt - eingetragen sind. Im Weiteren werden anhand dieser Korrelationsmatrix Objektregionen gesucht.

Clustering der Korrelationsmatrix Das Clustering der Korrelationsmatrix mit dem EM-Clustering liefert Objektregionen gleicher Art, wie in Abbildung 3.5 gezeigt. Zum Vergleich ist in Abbildung 3.4 das Ergebnis eines Clusterings von einer Korrelationsmatrix auf allen Objektpunkten - und nicht nur auf dem reduzierten Satz - dargestellt. Anhand der unterschiedlichen Farben, die jeweils einen Cluster repräsentieren, ist zu sehen, dass sich mit dieser Methode die Punkte auf der Oberfläche grundsätzlich unterscheiden lassen.

Im Vergleich mit dem Clustering-Ergebnis desselben Flugzeugs auf dem reduzierten Ausgangsdatensatz in Abbildung 3.5 (c) zeigt sich, dass der Ansatz auf der reduzierten



(a) Palme (b) Schmetterling (c) Flugzeug

Abbildung 3.5: Regionen gleicher Art nach Clustering mit dem EM-Algorithmus auf den phasenbasierten Merkmalen. Jede Farbe steht für einen Cluster.

Menge eine Symmetrie berücksichtigt, da die Granularität der möglichen Cluster verringert wurde. Berücksichtigt man die Tatsache, dass die Korrelation auf allen Punkten auf der Oberfläche mittels Spherical Harmonics Korrelation sehr teuer ist, so ist dies ein erfreuliches Ergebnis.

Clustering von Regionen Um eine weitere Reduktion der Punktmengen zu erreichen, wird ein lokal beschränktes Clustering auf den Regionen durchgeführt, indem innerhalb eines vorgegebenen Radius die Punktkandidaten eines Clusters verschmolzen werden. Der Massenmittelpunkt eines jeden lokalen Clusters ist der Centroid und gleichzeitig ein Interest Point Kandidat.

Um eine zusätzliche Stabilität bei der Zuordnung der Regionen zu erreichen, und um Regionen gleicher Art nur einem Cluster zuzuordnen, wurde ein Ansatz gewählt, der die Symmetrie von Objekten ausnutzt. Sind etwa bei einem Objekt des Typs „Ameise“ die beiden gleich aussehenden Fühler in zwei unterschiedlichen Clustern, und ist bekannt, dass das Objekt symmetrisch ist, so werden diese beiden Cluster-Regionen getrennt behandelt.

Die dreidimensionalen Positionen der ermittelten n Punktkandidaten $p_i = \bar{x}$, für $\bar{x} \in \mathbb{R}^3$ und $i = 1...n$, werden mit k-Means nochmals gruppiert, wobei k abhängig von der Anzahl der Symmetrieachsen gewählt wird. Als Distanz für k-Means wird die euklidische Distanz zwischen paarweise verschiedenen Punkten p_1, p_2 benutzt:

$$dist(p_1, p_2) = \|\bar{x}_1 - \bar{x}_2\|_2 .$$

Ist die Distanz zweier Punkte gering (in Relation zu anderen Punkten), dann liegen sie mit hoher Wahrscheinlichkeit im selben Cluster.

Oberflächensampling Zusätzlich zu den Kandidaten des phasenbasierten Samplings werden gleichmäßig verteilte Punkte mit äquidistanten Abständen auf der Objektoberfläche selektiert und der Kandidatenliste hinzugefügt. Die Extraktion der Oberfläche kann mit morphologischen Operatoren (siehe 2.5) schnell durchgeführt werden. Äquidistante Punkte auf der Oberfläche können mittels Kugelschnitten mit festen Radien gefunden werden.

Ausgehend von einem zufälligen Aufpunkt auf der Objektoberfläche wird eine Kugel mit den Ursprungsdaten logisch mittels UND verknüpft, wobei der Inhalt der Kugel den Wert Null und die Kugeloberfläche den Wert Eins hat. Die logischen Schnitte werden sukzessive

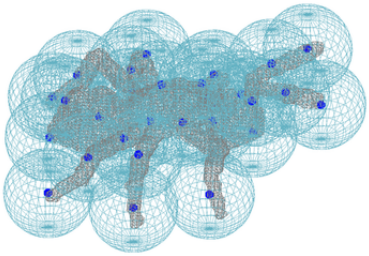
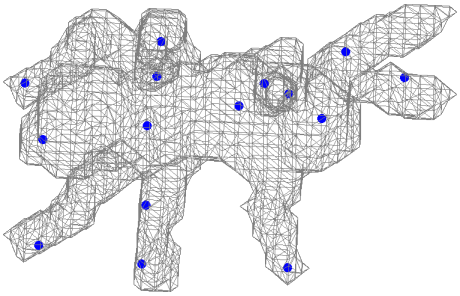
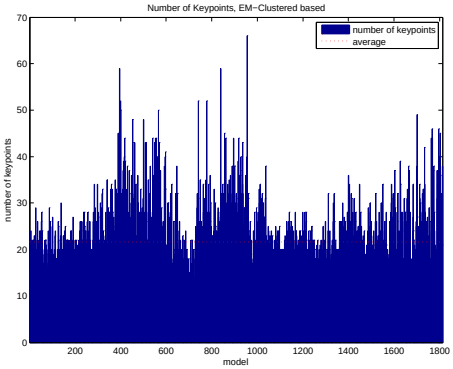


Abbildung 3.6: Äquidistantes Sampling von Punkten auf der Objektoberfläche mittels sphärischen Schnitten.



(a) Interest Points auf dem Modell „Ameise“.



(b) Anzahl der Interest Points pro Modell auf dem PSB Datensatz.

Abbildung 3.7: Das Clustering-basierte Verfahren zur Bestimmung von Interest Points auf der Princeton Shape Benchmark (PSB).

mit gleichen Kugelschnitten so lange weitergeführt, bis das gesamte Objekt durchlaufen wurde (siehe Abbildung 3.6). Die derart erzeugten Punkte haben alle äquidistante Distanzen zueinander.

Durch das äquidistante Sampling wird einerseits eine vollständige Abdeckung des Objekts erreicht, andererseits wird durch die verschiedenen Strukturen der Objekte eine gleichmäßige Verteilung der Punkte garantiert.

Ergebnisse auf der PSB Nach dem Clustering der Regionen und dem Oberflächensampling liegen als Ergebnis die gefundenen Interest Points auf dem Objekt vor. Für die Modelle der Princeton Shape Benchmark liefert dieses Verfahren durchschnittlich 21,6 Interest Points (siehe Abbildung 3.7).

Schwächen des Verfahrens Die teuerste Rechenoperation bei der Erzeugung der Clustering-basierten Features ist das Clustering mittels dem k-fach gefalteten EM-Algorithmus. Auch sind die erzeugten Interest Points nicht numerisch stabil, da das Cluster-Verfahren eine zufällige Initialisierung hat.

Im Folgenden wird ein anderes Verfahren beschrieben, das auf den phasenbasierten Features aufsetzt, aber einen anderen Weg zur Reduktion der Punkte wählt, um einerseits das teure Clustering zu umgehen und andererseits numerisch stabile Punkte zu finden.

3.2.3 Phasenbasierte Interest Points

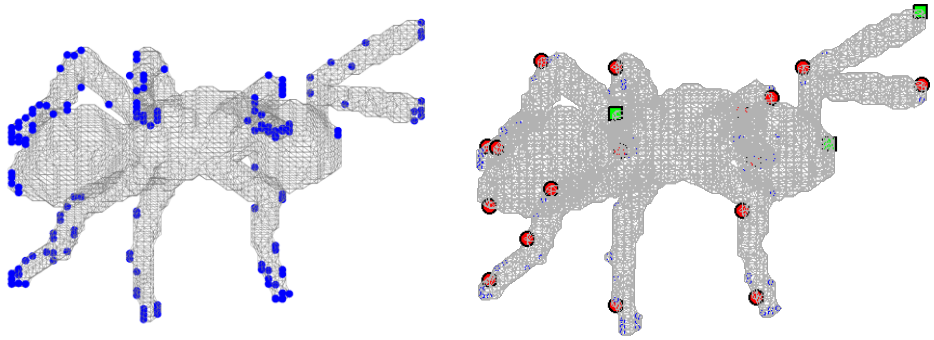
Wie in Abbildung 3.3 dargestellt, können die interessanten Stellen auf einem Objekt mit den $\mathcal{SH}_{phase}$-Features markiert werden. Diese Werte befinden sich in der Nähe des Mittelwerts. Die Idee für den Algorithmus zur Suche der Interest Points auf Basis der $\mathcal{SH}_{phase}$-Features ist, dass ein Schwellenwert gesucht wird, so dass nur noch die Punkte an den Kanten berücksichtigt werden. Dazu wird eine Heuristik angewandt, die relevante Punkte extrahiert, deren Werte sich in der Nähe des Mittelwerts befinden.

Algorithmus 3.1 Adaptive Schwellenwertanpassung (auf phasenbasierten Spherical Harmonics Features)

$n \leftarrow 0$
$i \leftarrow 0$
$\xi \leftarrow \mu_{Data}$
$\Delta_{step} \leftarrow \frac{\mu_{Data}}{\sigma_{Data}}$
$i_{max} \leftarrow \left\lfloor \frac{\sigma^2_{Data} - \mu_{Data}}{\Delta_{step}} + 1 \right\rfloor$
while $n \leq N \wedge \xi \leq \sigma^2_{Data} \wedge i \leq i_{max}$ **do**
  $\Delta \leftarrow \mu_{Data} + i \cdot \Delta_{step}$
  $\xi \leftarrow \xi + \Delta$
  $n \leftarrow |\langle DataX | \forall x \in Data : x > 0 \wedge x < \xi \rangle|$
  $i \leftarrow i + 1$
end while

Adaptive Schwellenwertanpassung Der Algorithmus 3.1 sucht nach einer vorgegebenen Zahl N von Interest Points in den Phasenwerten $Data$ der Spherical Harmonics, indem eine adaptive Schwellenwertsuche vorgenommen wird. Es wird beim Mittelwert μ_{Data} begonnen in Schritten Δ_{step} den Schwellenwert ξ zu erhöhen, so lange bis die geforderte Anzahl N gefunden wird, der Schwellenwert die Varianz σ^2_{Data} übersteigt, oder die maximale Zahl Schleifendurchläufe i_{max} überschritten wird. Während jedes Schleifendurchlaufs werden die Daten bezüglich des aktuellen Schwellenwertes begrenzt, und es werden die Anzahl der Voxel gezählt, die auf den aktuell begrenzten Daten noch vorhanden sind. Ein Beispiel für derart gefundene Interest Points ist in Abbildung 3.8(a) zu sehen. Der vorgestellte Algorithmus liefert numerisch stabile Ergebnisse und ist einfach zu implementieren.

Verschmelzung Wie in Abbildung 3.8(a) erkennbar, ist die Anzahl der gefundenen Interest Points redundant. Um die Anzahl der Punkte zu reduzieren, werden benachbarte Punkte zu einem neuen Punkt verschmolzen. Der neue Punkt ist derjenige Kandidat, der am nächsten zum Massenmittelpunkt einer lokalen, benachbarten Punktgruppe liegt. Damit wird sichergestellt, dass der neue Interest Point innerhalb des Objektes liegt, denn der Massenmittelpunkt ist eine künstliche Position, die außerhalb des Objektes liegen kann.



(a) Vor der Verschmelzung (b) Nach der Verschmelzung

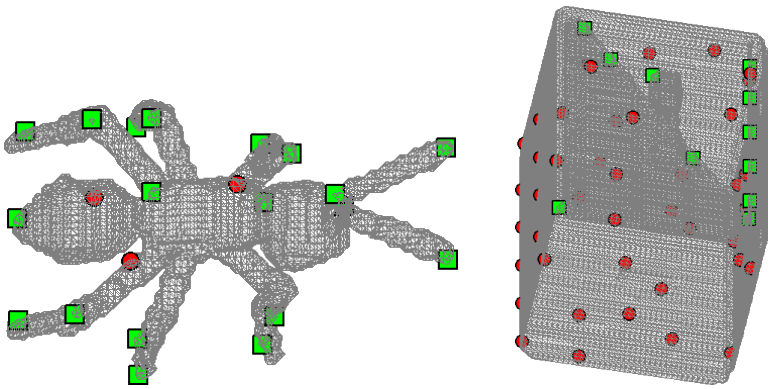
Abbildung 3.8: Beispiel für SHphase basierte Interest Points. In (a) sind alle gefundenen Interest Points als blaue Punkte aufgetragen. In (b) stellen die roten Punkte die neuen Punkte nach der Verschmelzung dar, während die grünen Kästchen beibehaltene Punkte sind. Die kleinen blauen Punkte sind die ursprünglichen Punkte aus (a) als Referenz.

In Abbildung 3.8(b) sind die neuen Interest Points nach Verschmelzung dargestellt. Die neuen Punkte sind Repräsentanten einer lokalen Gruppe. Können keine weiteren Punktkandidaten in der lokalen Nachbarschaft gefunden werden, so werden die ursprünglichen Interest Points beibehalten.

Oberflächensampling Um eine Verteilung der Interest Points nicht nur an den durch die phasenbasierten Stellen zu erreichen, wird eine gleichmäßige Abtastung (*Sampling*) der Objektoberfläche vorgenommen (siehe 3.2.2 und Abbildung 3.6). Die so gefundenen Punkten werden mit der Menge der $\mathcal{SH}_{phase}$-Punkten vereinigt, indem nahe benachbarte Punkte wie bei der Verschmelzung im vorherigen Abschnitt kombiniert werden.

Das Oberflächensampling wird notwendig, wenn zu wenige Punkte durch die rein phasenbasierte Suche gefunden werden. Dies kann bei Objekten der Fall sein deren Struktur wenige charakteristische Eigenschaften aufweist, wie etwa bei homogenen und glatten Flächen. Ein Beispiel hierfür ist eine Truhe, wo der Griff zwar eine starke Veränderung im Musterraum darstellt, die glatten Seitenwände aber nicht viele Kanten aufweisen. In Abbildung 3.9 (b) ist solch eine Truhe dargestellt. Die phasenbasierte Suche findet Interest Points auf dem Deckel und dem Griff der Truhe. Da die phasenbasierten Features an den Seitenwänden aber gleiche beziehungsweise zu ähnliche Werte aufweisen, bleiben diese unberücksichtigt. Durch das Oberflächensampling werden dagegen auch andere Objektstellen erreicht. Bei dem Beispiel „Ameise“ wiederum sind die Auswirkungen weniger gravierend, da die meisten interessanten Stellen bereits durch die phasenbasierte Suche abgedeckt wird (siehe Abbildung 3.9 (a)).

Ergebnisse auf der PSB Das Verfahren ist numerisch stabil; die $\mathcal{SH}_{phase}$-basierten Interest Points werden immer an den gleichen Stellen gefunden. Die zufällige Komponente ist die Menge der zusätzlichen Punkte, die durch das Oberflächensampling erzeugt wird. Die



(a) Ameise (b) Truhe

Abbildung 3.9: Zusätzlich zu den $\mathcal{SH}_{phase}$-basierten Interest Points (grüne Kästen) durch Oberflächensampling (rote Punkte) erweiterte Anzahl Interest Points. Während bei (a) die Anzahl nicht wesentlich steigt (nur um 3 Punkte), ist bei (b) eine deutliche Steigerung zu erkennen.

Punkte des Oberflächensamplings sind notwendig, weil dadurch eine relativ gleichmäßige Abdeckung des Objekts gewährleistet wird.

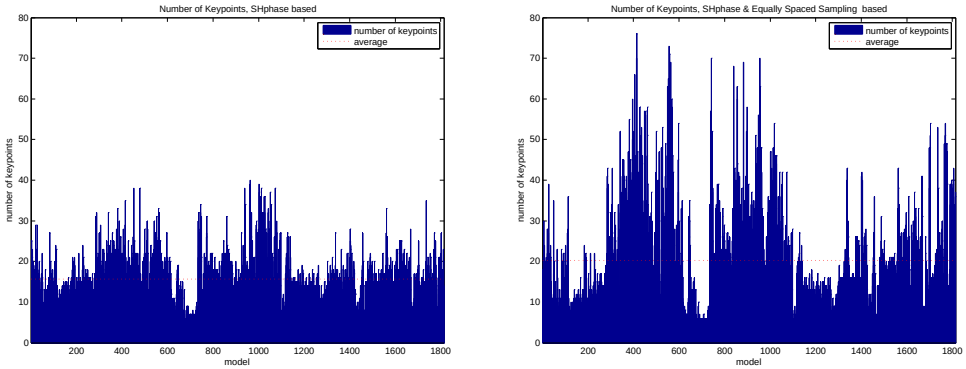
Auf den Daten der Princeton Shape Benchmark werden für die rein phasenbasierte Suche im Mittel 15,6 Punkte pro Objekt gefunden, während es mit dem zusätzlichen Oberflächensampling im Mittel 20,2 Punkte sind (siehe Abbildung 3.10).

3.3 Patch

An den gefundenen Interest Points werden Spherical Harmonics über mehrere Radien entwickelt, um eine Repräsentation der lokalen Umgebungen an diesen Stellen zu erhalten. Die Kombination aus Interest Point und Spherical Harmonics Koeffizienten wird im weiteren Verlauf als *Patch* bezeichnet. Ein solches Patch beschreibt die lokale Umgebung eines Objektes, wie in Abbildung 3.11 beispielhaft dargestellt.

Für die Konstruktion lokaler Merkmale gibt es verschiedene Ansätze. Am gebräuchlichsten ist es einen Feature-Vektor in Kombination mit einer Metrik zu benutzen. Andere Ansätze erzeugen statistische Werte auf den Objekten, wie zum Beispiel Oberflächensummen oder Distanzen zwischen zufällig ausgewählten Punkten auf dem Objekt, die dann als Histogramm-Features dargestellt werden . Es gibt aber auch Ansätze, die von der Struktur dreidimensionaler Modelle Gebrauch machen, und das 3D Modell als Graphen in Form eines Skeletts abbilden [20, 44]. Bei Fehr *et* al. werden Spherical Harmonics zur Erstellung von Histogrammen genutzt, mit denen mittels einer Support-Vector-Machine eine Objektklassifikation durchgeführt wird. Ein anderer Ansatz mit Spherical Harmonics und Repräsentation der Features als zweidimensionale Histogramme wird von Kazhdan *et* al. [26] verfolgt.

In der vorliegenden Arbeit werden die lokalen Features anhand von Patches erzeugt, die auf einer Entwicklung nach Spherical Harmonics basieren. Wie sich diese Spherical Harmonics Patches darstellen, wird im Folgenden erläutert.



(a) Anzahl der rein phasenbasierten Interest Points.

(b) Anzahl phasenbasierte Interest Points mit Oberflächensampling.

Abbildung 3.10: Vergleich der rein phasenbasierter Interest Point Suche und mit zusätzlichem Oberflächensampling.

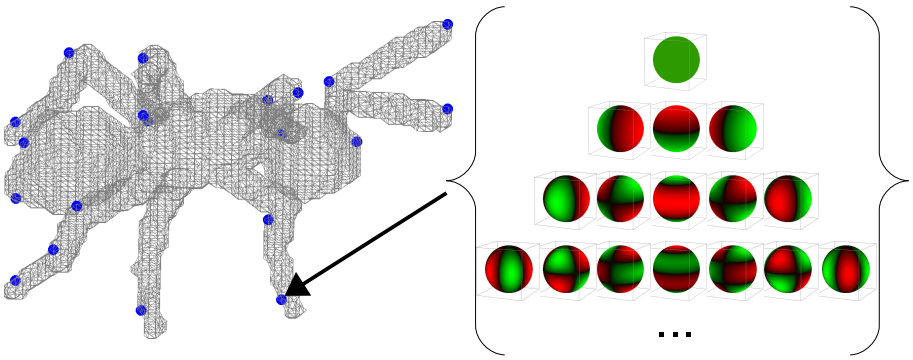


Abbildung 3.11: Repräsentation eines lokalen Merkmals mit Spherical Harmonics .

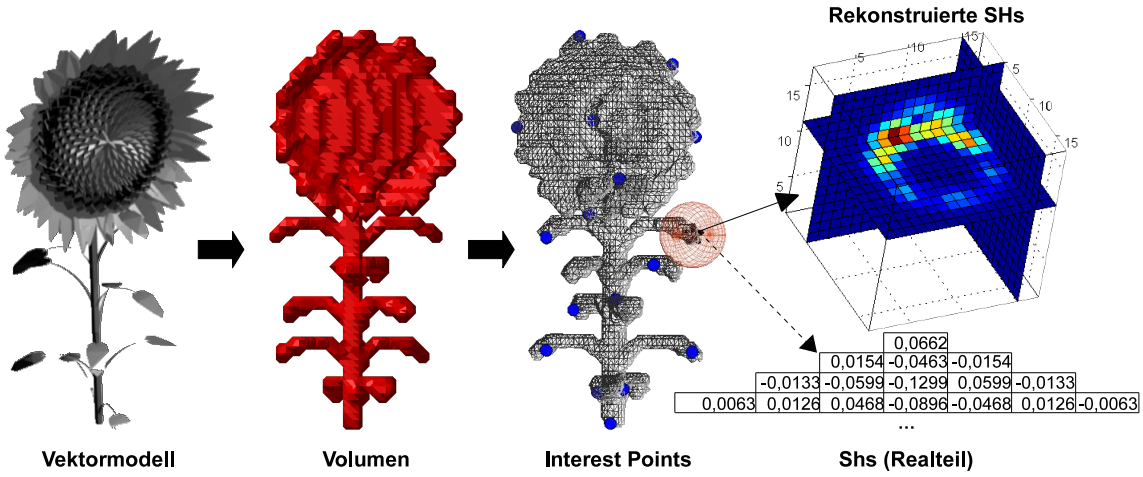


Abbildung 3.12: Schema Prozessablauf für Spherical-Harmonics-Patch. Von links nach rechts: Vektordatenmodell, Voxel-Rasterung als Volumen, Detektion von Interest Points, Spherical Harmonics Koeffizienten an ausgewählter Stelle mit Radius 5 (rekonstruiert und für die ersten vier Bänder im Realteil).

Der Datentyp der Spherical Harmonics ist eine Matrix mit komplexen Werten. Ein Ausgabebeispiel für ein Spherical Harmonics Feature als Matrix für die ersten drei Bänder ist (nur für den Realteil):

	$m=-2$	$m=-1$	$m=0$	$m=1$	$m=2$
$l=0$			0,1894		
$l=1$		0,0986	-0,1168	-0,0986	
$l=2$	-0,1051	-0,0739	-0,1028	0,0739	-0,1051

Mit steigendem Band l wird zwar die Repräsentationsgenauigkeit der Spherical Harmonics erhöht, doch steigt auch die Rechenzeit und der Speicherverbrauch (und damit auch die Dimensionalität der Features) quadratisch (siehe Formel 2.7).

Für eine annähernd genaue Approximation der zugrunde liegenden Oberflächenstrukturen wurde die Entwicklung auf das zehnte Band ($l=10$) begrenzt. Dadurch ergibt sich nach Formel 2.7 für die komplexen Feature Deskriptoren eine Dimension von $\sum_{l=0}^{10}(2l+1)=121$.

In Abbildung 3.12 ist der gesamte Prozess der Merkmalsextraktion und Konstruktion des Patch an einem Beispiel dargestellt. An der durch die roten Kugel markierten Stelle im dritten Bild von rechts wurde im Radius fünf eine Spherical Harmonics Entwicklung durchgeführt. Die Koeffizienten für den Realteil sind für die ersten vier Bänder als reelle Werte aufgelistet (siehe auch Abbildung 2.4 zur Visualisierung der Spherical Harmonics).

Um eine skalierungsinvariante Repräsentation an einem Interest Point zu erhalten, werden die Spherical Harmonics Entwicklungen auf sphärischen Funktionen mehrerer Radien ausgeführt und bezüglich der relevanten Voxel auf den sphärischen Funktionen normiert. Dieses multi-resolution Patch wird dann als lokale Merkmalsbeschreibung im Codebuch eingetragen.

3.4 Diskussion

Die vorgestellten Verfahren zur Extraktion von Merkmalen auf den Modellen der Princeton Shape Benchmark (PSB) funktionieren sehr gut und auch schnell.

Da in einigen Veröffentlichungen (zum Beispiel bei Nowak *et* al. [35]) davon ausgegangen wird, dass zufällig und gleichmäßig verteilte Interest Points für die Erstellung des Codebuches ausreichen, wurde versucht das Retrievalverfahren auch mit Features an derartigen Stellen auszuführen. Für den weiteren Prozess - der Erstellung eines Codebuches anhand der Patches an diesen Interest Points - zeigte sich dieser Ansatz aber als nicht praktikabel, da er zu rechenintensiv ist. Werden zum Beispiel 100 Features pro Objekt angelegt, und nur für eine Auswahl von zwei Modellen pro Klasse im PSB-Datensatz die Patches extrahiert, so müssen dennoch $180 \cdot 100 \cdot \frac{(180 \cdot 100+1)}{2} \approx 1,6 \cdot 10^{10}$ aufwändige Spherical Harmonics Korrelationen ausgeführt werden. Aus diesem Grund wurde darauf Wert gelegt nur an den charakteristischen Stellen die Interest Points zu finden, um möglichst viel Information abzugreifen, bei gleichzeitiger Begrenzung der Interest Points Anzahl.

Kapitel 4

Codebuch

Um in einer Retrievaldatenbank die gesuchten Objekte schnell und effizient wiederzufinden, wird ein Codebuch für die Anfragen an die Datenbank benutzt. Dieses Codebuch enthält neben den im vorherigen Kapitel beschriebenen lokalen Patches eine identifizierende Beschreibung zu jedem Objekt. Dabei ist von Bedeutung, dass nicht alle Patches in das Codebuch eingetragen werden, sondern nur diejenige Auswahl, die möglichst viele Patches eindeutig beschreiben vermag. Somit ist das Codebuch die komprimierte Repräsentation gelernter Patches: Es enthält möglichst universelle Beschreibungen von 3D Patches.

In diesem Kapitel werden die Überlegungen formuliert, wie ein solches Codebuch erstellt werden kann. Danach werden die einzelnen Schritte zur Codebucherstellung erklärt.

Ein Codebuch wird spezifisch für einen vorliegenden Datensatz in vier Schritten gelernt:

1. Extraktion lokaler Merkmale und Patches,
2. Gruppierung der Patches nach Ähnlichkeit,
3. erzeugen „universeller“ Patches und Eintragung in einem Vokabular (Codebuch), und
4. Erstellung von Antwort-Histogrammen anhand einer Trainingsmenge auf den Patches im Codebuch.

Die Histogramme geben wieder, wie oft ein bestimmtes Patch auf einem Objekt angetroffen wurde. Bezogen auf das Konzept des Feature-Vektors sind sie die lokalen Features. Ziel ist es ein minimales Codebuch zu finden, das nur diejenigen signifikanten Features enthält, die bei einer Retrievalanfrage (Query) die Intra-Klassendistanz minimieren und die Inter-Klassendistanz maximieren.

4.1 Bag-of-Features

Die Erstellung des Codebuches für das Retrieval bedient sich eines Prinzips, das in der Literatur als *Bag-of-Features* bekannt ist. Dieses stammt aus der Texturanalyse [35], wo atomare Strukturen (Textons) in einem Vokabular für die spätere Objekterkennung abgelegt werden. Das Verfahren läuft analog zum *Bag-of-Words* Konzept aus dem Bereich der Textanalyse, wo Wort-Wolken oder Vokabularien selbstständig gelernt werden. In der inhaltsbasierten Bildklassifikation ist es ein populäres Verfahren, einerseits weil es auf einem einfachen Konzept basiert, andererseits wegen seiner guten Performance [35]. Das Konzept wurde auf die Verwendung zur inhaltsbasierten Objektklassifikation in 3D erweitert, wobei das Codebuch den „Beutel“ mit Features darstellt.

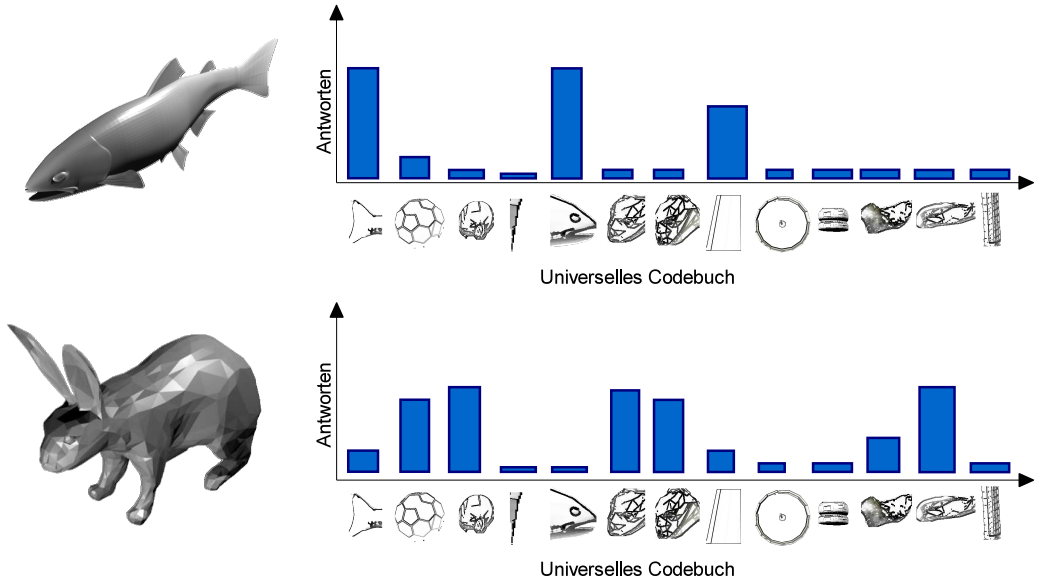


Abbildung 4.1: Beispiel für den klassischen Bag-of-Features Ansatz für 3D Shapes. Verschiedene Shapes generieren unterschiedliche Antworthistogramme auf einem universellen Codebuch. Im Codebuch stehen die gelernten Konzepte unterschiedlicher Patches.

Der Hauptaspekt des Bag-of-Features Konzepts ist, dass die räumlichen Relationen zwischen den Features verworfen werden. Die Features werden als lose Kollektionen unabhängiger Merkmale betrachtet, die gleichzeitig möglichst atomar und universell sind.

Die Grundidee ist folgendermaßen:

1. Repräsentation der 3D Shapes als Kollektion unabhängiger Patches.
2. Evaluation der Patches eines Objektes auf dem Codebuch.
3. Die für jedes Objekt gefundene Verteilung der Patches im Deskriptorraum wird als Feature für das Objekt benutzt.

Die Verteilungen lassen sich als Histogramme darstellen, die bei einem Query schnell durchsucht werden können (etwa mit einer Support-Vector-Machine, die mit einem Histogram-Intersection-Kernel ausgestattet ist). In Abbildung 4.1 ist ein Beispiel für einen Bag-of-Features Ansatz auf den 3D Modellen abgebildet. Die auf den 3D Modellen extrahierten Patches liefern bei der Anfrage an das Codebuch unterschiedliche Antworten.

Nach Nowak *et* al. [35] gibt es bei der Erstellung eines derartigen Codebuches vier grundsätzliche Implementierungsentscheidungen:

1. Auf welche Weise werden die Codebucheinträge ausgesucht?
2. Auf welche Art werden die Codebucheinträge dargestellt?
3. Wie werden die Verteilungen charakterisiert?

4. Wie werden Objekte anhand des Codebuches klassifiziert?

In der vorliegenden Arbeit wurden für die Selektion der Codebucheinträge die in Kapitel 3 vorgestellten Patches benutzt. Diese Patches werden jedoch nicht nativ in das Codebuch eingetragen, sondern einem Clustering unterworfen, um ähnliche Patches zu gruppieren und aus diesen Gruppierungen universelle Codebucheinträge zu finden.

Die univsersellen Einträge können als Repräsentanten für allgemein gültige Konzepte für Patches angesehen werden. Werden zum Beispiel beim Clustering alle Füße mehrerer Objekte in einer Klasse gruppiert, so stellt der zugehörige Codebucheintrag das Konzept „Fuß“ dar. Neben dem Vorteil der Generalisierung ist ein weiterer Aspekt, dass die Anzahl der Codebucheinträge durch diese Art der Kompression stark reduziert wird, was einen Geschwindigkeitsvorteil für die Suche beim Retrieval ergibt. Die Erzeugung dieser Codebucheinträge wird im übernächsten Abschnitt 4.3 erklärt.

Die dritte Frage zur Implementierung, also wie sich die Verteilungen der Codebucheinträge auf Anfrageobjekten an das Codebuch darstellen, wird über Histogramme gelöst. Die Anzahl der Bins eines Codebuchhistogramms ist die Anzahl der Codebucheinträge, sprich die Anzahl der lokalen Patches. Anhand der verfügbaren Patches werden Histogramme auf einem Trainingsdatensatz gelernt. Diese Histogramme stellen die Feature Vektoren dar, die beim Retrieval zum Vergleich mehrerer Objekte dienen.

Das Ziel ist, dass die Histogramme eine kleine Intra- und eine große Interklassendistanz aufweisen, damit anhand der Histogramme eine Klassifizierung der Modelle in Klassen vorgenommen werden kann. In Abbildung 4.2 sind zu einem Query-Objekt (PSB Modellnummer *m1397*) andere ähnliche Modelle in der ersten Zeile aufgelistet. In der zweiten Zeile stehen die Modelle, deren Distanz zum Query-Objekt am größten sind.

4.2 Codebuch Clustering

Das Clustering des Codebuches ist im Prinzip eine Kategorisierung beziehungsweise eine Klassifikation der im Codebuch enthaltenen Patches (siehe 3.3) in Gruppen gleicher Art. Zum Beispiel sollen auf den Objekten der Princeton Shape Benchmark in der Klasse der Arthropoden die Beine und die Fühler jeweils in eine Kategorie eingeordnet werden, oder in der Klasse der Flugzeuge die Seitenruder (siehe Abbildung 4.3).

Um die Patches zu clustern muss ein Maß für die Ähnlichkeit beziehungsweise Unähnlichkeit definiert werden. Da jedes Patch ein lokales Feature darstellt, das durch eine Spherical Harmonics Entwicklung repräsentiert wird, kann die Ähnlichkeit der Patches über eine Spherical Harmonics Korrelation (siehe 2.4) bestimmt werden. Aus der Korrelation über alle N Patches ergibt sich eine $N \times N$ Korrelationsmatrix, die für das Clustering benutzt wird. In Abbildung 4.4 sind für eine Zeile aus der Korrelationsmatrix die 10 besten den 10 schlechtesten Patches gegenübergestellt. Während die obere Zeile eher „Bananenähnliche“ Patches enthält, sind es in der unteren Zeile eher halbkugelförmige Patches.

Für das Clustering der Codebuch-Patches werden in der Literatur sowohl partitionierende Cluster-Verfahren wie k-Means und Derivate genutzt [9, 51, 52], als auch hierarchische Verfahren [1, 23]. Beide wurden in dieser Diplomarbeit für das Clustering des Codebuches untersucht.

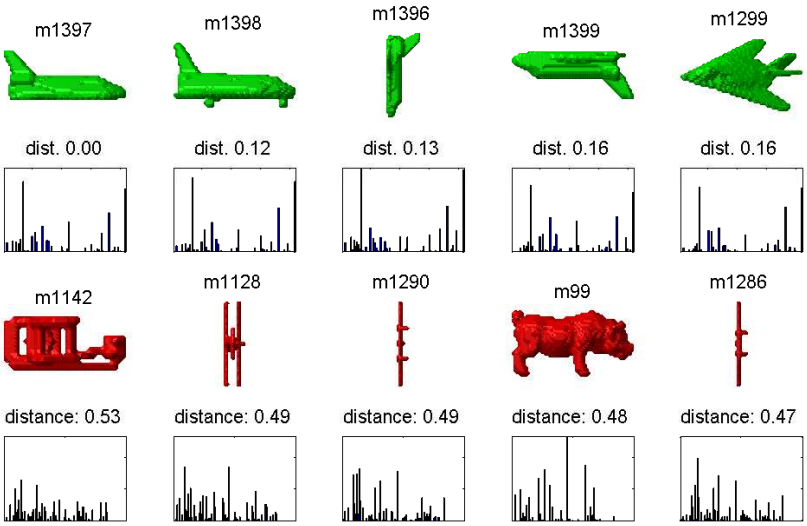


Abbildung 4.2: Histogramme für gleiche und verschiedene Modelle. Das erste Modell links oben ist das Query-Objekt, dementsprechend die Distanz 0. Die Modelle der ersten Zeile von links nach rechts sind die zu diesem Modell ähnlichsten. In der zweiten Zeile stehen die mit der größten Distanz zum Query-Objekt, also die unähnlichsten.

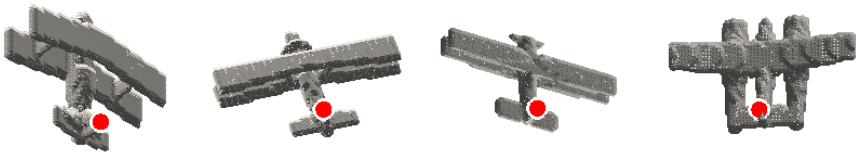


Abbildung 4.3: Gruppierung gleichartiger Patches anhand der Korrelation der Spherical Harmonics basierten Features an den Patches. Die markierten Punkte sind die Positionen der lokalen Features.

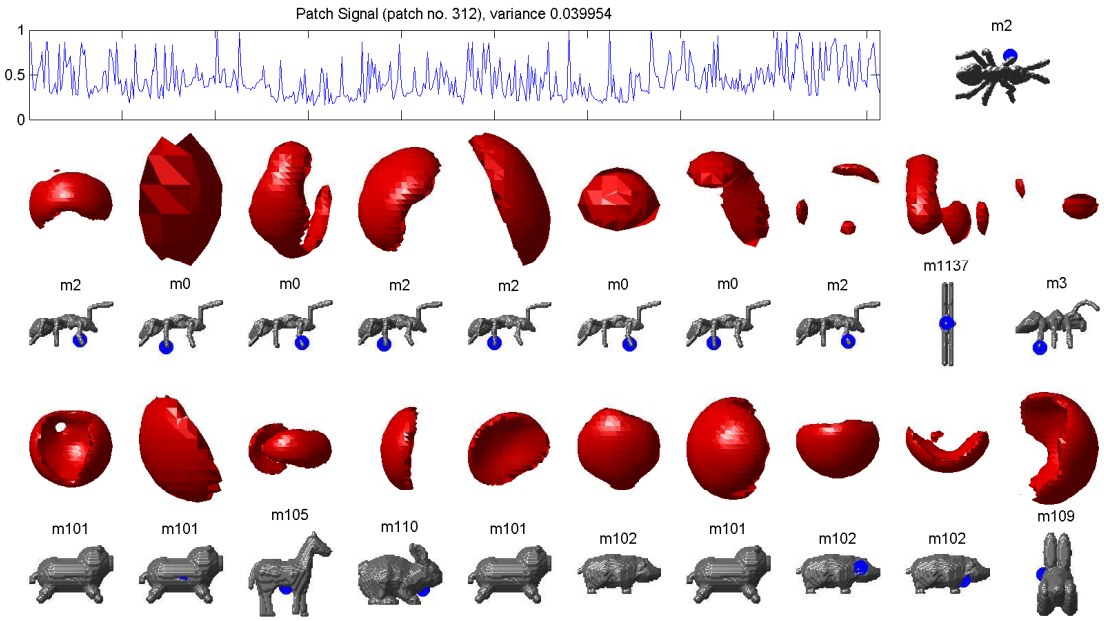


Abbildung 4.4: Die 10 ähnlichsten und die 10 verschiedensten Patches in der Korrelationsmatrix für eine ausgewählte Zeile. In der ersten Bildzeile ist das Signal der Matrixzeile für das Objekt *m2* aufgetragen. Zeile zwei und drei enthalten die 10 bestkorreliertesten Patches, wobei Zeile drei anzeigt, von welchem Objekt dieses Patch stammt; Zeile vier und fünf entsprechend für die Schlechtesten.

k-Means Clustering

Die Werte in der Korrelationsmatrix geben die Ähnlichkeiten der Patches zueinander wieder. Diese Ähnlichkeit wird als Distanzmaß für das k-Means Clustering verwendet.

Um für jedes Patch eine Gruppe ähnlicher Patches zufinden, wird in der Korrelationsmatrix zeilenweise ein *repeated* k-Means ausgeführt. Pro Zeile ergeben sich Cluster von Korrelationswerten durch die bei k-Means gefundenen Cluster-Centroids. Anhand dieser Cluster lassen sich die ähnlichen Patches über die Positionen in der Matrix gruppieren. Da die Anzahl der Cluster nicht a priori bekannt ist, muss k empirisch ermittelt werden.

Aus den Patches eines Clusters werden bei der Konstruktion der Codebucheinträge (siehe übernächsten Abschnitt 4.3) Cluster-Repräsentanten ermittelt.

Agglomeratives Clustering

Ausgehend von der Korrelationsmatrix wird eine Distanzmatrix gebildet und diese gruppiert. Da bei diesem Verfahren die Anzahl der Cluster nicht a priori bekannt ist, wurde eine Methode zur dynamischen Anpassung des Schwellenwertes der Verschmelzungsdistanz entwickelt, die sich auf die Anzahl der Cluster auswirkt.

Für das Clustering der Distanzmatrix wurde das *average-linkage* Clustering gewählt (siehe Abschnitt 2.6.3). Bei den verwendeten Daten der Princeton Shape Benchmark ist die Verteilung gleichartiger Patches nicht gleichmäßig, weil für unterschiedliche Kategorien die Strukturen der Modelle verschieden sind. Die average-linkage eignet sich für solche Daten, da sie die verschiedenen Strukturen mittelt. Als Bestätigung dieser Tatsache kann auch angesehen werden, dass der kophenetische Korrelationskoeffizient für das average-linkage Clustering den höchsten Wert liefert.

Die Anzahl der Cluster und damit die Anzahl der Codebucheinträge ergibt sich über den Cutoff-Wert. Dieser Schwellenwert gibt an, wie lange die Patches noch weiter in unterschiedliche Cluster sortiert werden.

Die Bestimmung des Cutoff-Schwellenwertes wird an das Wissen über ähnliche Patches geknüpft. Der Ähnlichkeitswert (Korrelationswert) zwischen zwei Patches, die durch Spherical Harmonics repräsentiert werden, bewegt sich im Bereich $[0; 1]$. Für identische Spherical Harmonics ist der Korrelationswert 1, für total verschiedene 0. Für in der Struktur ähnliche Patches konnte empirisch ein Korrelationswert von größer als 0,5 ermittelt werden. Dieses Wissens dient als Richtwert für den Cutoff-Wert bei der Erstellung des Cluster-Baums.

4.3 Codebucheinträge

Anhand des Clusterings der Patches in Gruppen gleicher Art werden repräsentative Codebucheinträge ausgewählt. Ziel ist die Reduktion des Codebuches durch Verschmelzen von Codebucheinträgen, die nur wenig zur Klassenunterscheidung beitragen [52]. Dies kann als eine Anwendung der *„Information Bottleneck Method“ [47]* angesehen werden, bei der ein Kompromiss zwischen der Genauigkeit und der Komplexität der neuen Codebucheinträge eingegangen wird. Es werden diejenigen Codebucheinträge gesucht, die eine maximale Unterscheidbarkeit zwischen Objektklassen gewährleisten, aber gleichzeitig derart sind, dass mit ihnen Objekte derselben Klasse identifiziert werden können - die Informationen

werden „durch einen Flaschenhals gequetscht“, so dass die Essenz der Information übrig bleibt.

Für eine Menge $\hat{F}_{lm} = \{(\hat{f}_{lm})_1, ..., (\hat{f}_{lm})_n\}$ von n Spherical Harmonics Koeffizienten wird durch eine Auswahlfunktion ξ ein Element $(\hat{f}_{lm})_\lambda \in \hat{F}_{lm},\ \lambda \in \{1, ..., n\}$ repräsentativ ermittelt mit

$$\xi : \hat{F}_{lm} \rightarrow (\hat{f}_{lm})_\lambda.$$

Die Anforderung an den neuen Repräsentanten $(\hat{f}_{lm})_\lambda$ ist, dass dieser die Eigenschaften seiner Menge bestmöglich widerspiegelt. Die repräsentierte Gruppe ist in Bezug auf das Codebuch ein Cluster mit n Elementen, also mit n Spherical Harmonics Koeffizienten. Das durch ξ ausgewählte Element wird dann der Repräsentant der Cluster-Gruppe und der neue universelle Codebucheintrag.

Für die Auswahlfunktion ξ wurden zwei Methoden entwickelt. Die erste Methode basiert auf einer Auswahl des Cluster-Repräsentanten anhand der Korrelationswerte bezüglich der lokalen Korrelationsmatrix. Die zweite Methode verschmilzt die Teilmenge einer Gruppe und synthetisiert daraus den Repräsentanten.

Da die Patches die Spherical Harmonics Entwicklungen über mehrere Radien enthalten, wird die Auswahl des Repräsentanten getrennt für jeden Radius einzeln ausgeführt.

4.3.1 Best-Of Methode

Für jeden Cluster wird eine lokale $n \times n$ Korrelationsmatrix $\dot{S} \in \mathbb{R}^2$ bestehend aus den Korrelationswerten der jeweiligen n Cluster-Elementen gebildet. Die Auswahlvorschrift ξ ist derart, dass diejenige Instanz r - also derjenige Codebucheintrag - mit der höchsten Zeilensumme ausgewählt wird (siehe auch Abbildung 4.5):

$$\begin{aligned} r &= \arg\max_{1 \le i \le n} \{\sum_{j=1}^{n} \dot{S}(i,j)\} \qquad i, j, r \in \{1, ..., n\} \\ (\hat{f}_{lm})_\lambda &\leftarrow (\hat{f}_{lm})_r \qquad (\hat{f}_{lm})_\lambda \in \hat{F}_{lm} \end{aligned}$$

Für das selektierte Cluster-Element $(\hat{f}_{lm})_\lambda$ gilt, dass es bezüglich den anderen Mitgliedern seiner Cluster-Gruppe $\hat{F}_{lm}$ am allgemeinsten ist, da alle anderen Elemente zu diesem ähnlich sind. Dieses Element wird als Repräsentant des Clusters ausgewählt und im Codebuch eintragen.

4.3.2 Varianz-basierte Verschmelzung

Die Idee hinter diesem Verfahren ist die Varianz der durch die Spherical Harmonics Koeffizienten repräsentierten Funktionen $f(\theta, \phi)$ zu minimieren und damit diejenigen Oberflächenpunkte zu eliminieren, die für ein eventuelles Rauschen verantwortlich sind; die Streuung über mehrere Oberflächenpunkte wird minimiert. Das Ergebnis ist ein schärferes Profil der Kombination mehrerer Ausgangsfunktionen. Zur Veranschaulichung ist die Idee in Abbildung 4.6 schematisch dargestellt.

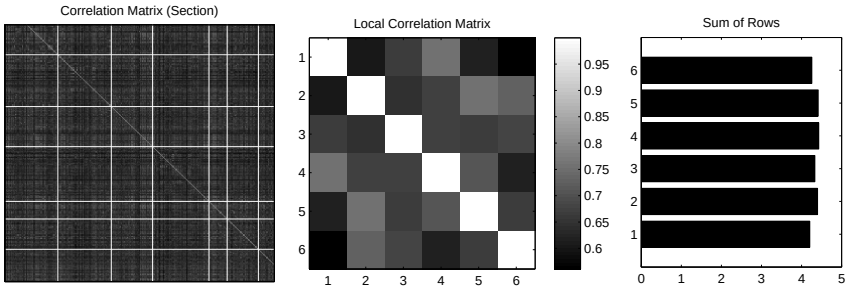


Abbildung 4.5: Lokale Korrelationsmatrix und Zeilensummen bei der Best-of Methode. Von links nach rechts: Korrelationsmatrix, Lokale Korrelationsmatrix für Cluster, Zeilensummen.

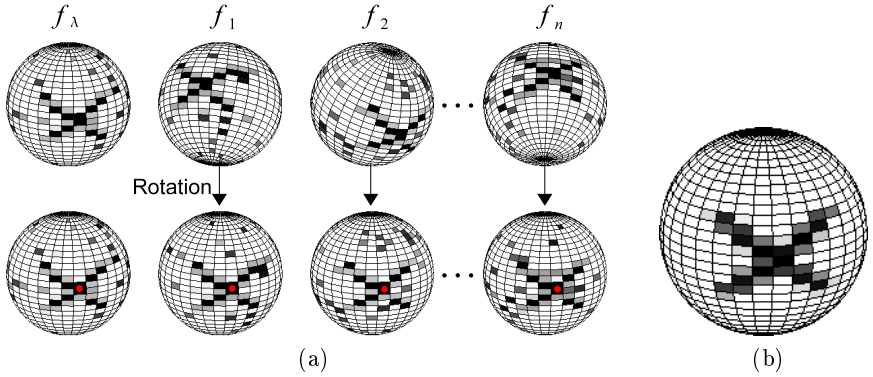


Abbildung 4.6: Schema zur Varianz-basierten Verschmelzung. Die Patches eines Clusters werden bezüglich des Referenzelements ausgerichetet (a). Der Repräsenant (b) ist eine neue Kugeloberfläche mit niedriger Varianz.

Für die Berechnung der Varianz müssen die Spherical Harmonics Koeffizienten rücktransformiert werden. Da diese Transformation rechenintensiv ist und auch die rücktransformierten Volumendaten viel Speicherplatz benötigen, wurde ein online-Algorithmus zur Berechnung der Varianz implementiert, der sich an dem numerisch stabilen Algorithmus von Knuth [28] orientiert. In 4.1 ist der Algorithmus aufgelistet.

Algorithmus 4.1 Online-Berechnung der Varianz σ^2 von mehreren sphärischen Funktionen repräsentiert durch Spherical Harmonics Koeffizienten.

$n \leftarrow 0$
$\mu \leftarrow 0$
$\Theta \leftarrow 0$
$(\hat{f}_{lm})_\lambda \leftarrow \text{bestof}(\hat{F}_{lm})$ {Best-Of Methode}
for all $(\hat{f}_{lm})_i \in \hat{F}_{lm}$ **do**
  $n \leftarrow n + 1$
  $(\hat{f}_{lm})_i^R \leftarrow R_{(\hat{f}_{lm})_\lambda}((\hat{f}_{lm})_i)$ {$\mathcal{SH}$-Rotation}
  $f_i \leftarrow \mathcal{SH}^{-1}((\hat{f}_{lm})_i^R)$ {$\mathcal{SH}$-Rekonstruktion}
  $\Delta \leftarrow f_i - \mu$
  $\mu \leftarrow \mu + \frac{\Delta}{n}$
  $\Theta \leftarrow \Theta + \Delta \cdot (f_i - \mu)$
end for
$\sigma^2 \leftarrow \Theta/(n-1)$
σ^2

Da die Spherical Harmonics Repräsenatationen ($\mathcal{SH}$-Patches) an unterschiedlichen Punkten auf einem 3D Modell entwickelt wurden, und die Modelle nicht rotationsinvariant vorliegen, müssen die $\mathcal{SH}$-Patches erst ausgerichtet werden, so dass sie deckungsgleich liegen. Dazu werden alle Elemente $(\hat{f}_{lm})_i$, $i \in \{1, ..., n\}$ eines n-elementigen Clusters $\hat{F}_{lm}$ bezüglich eines Refenzelementes $(\hat{f}_{lm})_\lambda$, $\lambda \in \{1, ..., n\}$ rotiert, wobei $i \neq \lambda$ gilt. Das Referenzelement wird nach der Best-Of Methode (siehe vorherigen Abschnitt 4.3.1) ermittelt. Zur Bestimmung der Rotationswinkel wird eine schnelle Korrelation mit Padding $p = 70$ ausgeführt (siehe Abschnitt 2.4). Darauf basierend werden die Cluster-Elemente mit den sogenannten *Wigner D-Matrizen* [15] im Spherical Harmonics Raum bezüglich der Referenzkoeffizienten ausgerichtet. Diese neuen Spherical Harmonics werden mit $(f)_i \leftarrow \mathcal{SH}^{-1}((\hat{f}_{lm})_i^R)$ in den Ortsbereich rücktransformiert (siehe Formel 2.5), wo sie dann in Form von Grauwerten vorliegen, die auf der Kugeloberfläche definiert sind. Die Varianz berechnet sich aus diesen Grauwerten.

Für die Erstellung des Cluster-Repräsentanten werden nun punktweise diejenigen Punkte auf der aufsummierten Varianz-Kugeloberflächenfunktion Θ ausgewählt, die eine niedrige Varianz und damit eine niedrige Streuung aufweisen. Für den Schwellenwert wurde 2σ gewählt, sodass die Streuung um 95% reduziert wird. Diese Filterung findet im Ortsbereich statt, und da die Ausgangssignale binär sind werden auch wieder Binärwerte auf die Kugeloberfläche aufgetragen:

$$\tilde{f} = \begin{cases} 1 & \text{, für } \Theta(\theta, \phi) < 2\sigma \\ 0 & \text{, sonst} \end{cases}$$

$\Theta(\theta, \phi)$ gibt den entsprechenden Varianzwert an der Stelle θ, ϕ auf der Kugeloberfläche an. Die neue Kugeloberfläche wird mit $\mathcal{SH}(\widetilde{f}) \rightarrow \hat{f}_{lm}$ wieder nach Spherical Harmonics entwickelt und als Repräsentant der Gruppe im Codebuch eingetragen.

Da sich die $\mathcal{SH}$-Patches in einem Cluster untereinander ähnlich sind, müssen nicht alle Elemente $(\hat{f}_{lm})_i$ in die Erzeugung des Repräsentanten einfließen, sondern nur eine Stichprobe. Dies hat eine erhebliche Auswirkung auf die benötigte Rechenzeit, weil sowohl die Spherical Harmonics Korrelation als auch Rücktransformation rechenintensive Operationen sind.

4.4 Codebuch Histogramme

Die letztlich im Retrieval benutzten Feature-Vektoren sind die Histogramme, die sich aus dem Vergleich der Spherical Harmonics Patches im Codebuch mit den Patches eines Query Objektes ergeben.

Dazu wird das Codebuch anhand eines gegebenen Datensatzes trainiert, indem für jedes Objekt im Datensatz ein Histogramm als Feature angelegt wird. Das Codebuch besteht also neben den (universellen) Patches zusätzlich aus den Histogrammen. Ein solches Histogramm ist der Feature-Vektor zu einem Objekt und gibt an, wie oft ein bestimmtes Patch im Codebuch angetroffen wurde. Histogramme können im Gegensatz zu den Spherical Harmonics Patches wesentlich schneller verglichen werden, mit einfachen Methoden beginnend bei einem Histogram-Intersection-Kernel bis hin zu Support-Vector-Maschines mit einem solchen Kernel.

Abhängig vom gewählten Clustering stellen die Codebucheinträge mehr oder weniger ähnliche Patches dar. Um bei den Antwort-Histogrammen auch die Relation zu anderen, ähnlichen Patches zu erfassen, wurden Fuzzy-Histogramme als lokale Features gewählt. Fuzzy-Histogramme zählen nicht nur bezüglich eines einzigen Elements, wie oft dieses vorkommt, sondern gewichten die Zählung nach dem Auftreten der Elemente. Auf diese Weise werden auch diejenigen Codebuch-Patches im Histogramm berücksichtigt, die eine weniger starke Ähnlichkeit zum Anfrage-Feature aufweisen.

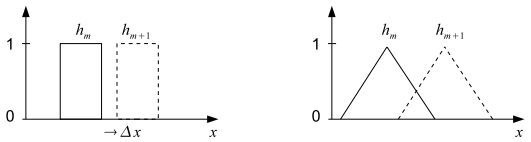
Um die Histogramme zu erstellen, werden die Spherical Harmonics Patches eines Query-Objektes paarweise mit den Patches im Codebuch korreliert und die Fuzzy-Histogramme anhand des Korrelationswertes ermittelt.

Nach Siggelkow *et* al. [42] wird ein Fuzzy-Histogramm für N Feature-Vektoren $x^{(n)}, n = 0, 1, ..., N-1$ mit $m = 0, 1, ..., M-1$ Bins definiert als

$$\text{hist}_m = \frac{1}{N} \sum_{n=0}^{N-1} h_m(x^{(n)}) \ .$$

Für die Zuordnungsfunktion $h_m(x)$ von einem Feature-Vektor zu einem Histogramm wurde eine lineare Gewichtungsfunktion gewählt. Anhand der Korrelationswerte der Spherical Harmonics Koeffizienten wird eine Randordnung festgelegt, so dass das bestkorrelierendste Patch mit 1 und das unähnlichste mit 0 gewichtet wird.

Beim Retrieval werden die Histogramme als Features zum Vergleich der Objekte benutzt. Dazu wird anhand der Histogramme eine Distanz ermittelt, die die Ähnlichkeit



(a) (b)

Abbildung 4.7: Fuzzy-Histogramme. Die Bin-weisen Zuordnungen überlagern sich über mehrere Bins (b) statt wie im klassischen Fall nur auf einem (a). (Nach [42])

zwischen einem Query-Objekt und bezüglich eines Objektes in der Retrievaldatenbank widerspiegelt. Als Distanzmetrik wurde ein Histogram-Intersection-Kernel gewählt:

$$\text{distance}(\text{hist}_1, \text{hist}_2) = \frac{\sum \min(\text{hist}_1, \text{hist}_2)}{\min(|\text{hist}|_1|, \text{hist}_2|)} \ .$$

Mit den paarweisen Distanzen wird beim Retrieval die Distanzmatrix erstellt.

4.5 Diskussion

Als größte Herausforderung bei der Erzeugung des Codebuches hat sich das Clustering herausgestellt. Schwierig ist einerseits die korrekte Anzahl von Codebucheinträgen - also die korrekte Anzahl repräsentativer Patches - zu finden, andererseits auch wie die Cluster erzeugt werden sollen. Agglomeratives Clustering hat sich hier als vorteilhaft erwiesen, da sich die Distanzen beim Verschmelzungsschritt aus der Distanzmatrix ergeben, die sich schnell durch Invertierung der (auf $[0;1]$ normierten) Korrelationsmatrix berechnen lässt. Da durch empirische Methoden der Ähnlichkeitswert (Korrelationswert) zwischen zwei Shapes beziehungsweise zwischen zwei Patches ermittelt werden kann, konnte als Cutoff-Schwellenwert beim Erzeugen des Cluster-Baums dieser Wert als Richtwert benutzt werden.

Die teuerste Operation bei der Erstellung des Codebuches ist die Korrelation der Spherical Harmonics Koeffizienten. Diese Operation wird sowohl beim Anlegen der Korrelationsmatrix für das Clustering benutzt, als auch beim Lernen der Codebuchhistogramme, da auch hier Korrelationswerte zwischen den Patches im Codebuch und auf den zu lernenden Objekten ermittelt werden müssen. Abhängig von der Anzahl der Objekte, die für die Codebucherzeugung ausgewählt wurden, steigt die Größe der Korrelationsmatrix und damit die Anzahl der paarweisen Korrelationen quadratisch in $O(n^2)$. Nutzt man die Tatsache aus, dass die Korrelation zweier Spherical Harmonics Koeffizienten symmetrisch ist, so reduziert sich die Anzahl der paarweisen Korrelationen auf $n(n+1)/2$ Vergleiche, was aber immer noch quadratisch ist.

Von Vorteil für das Retrieval ist, dass die für das Lernen der Codebuchgröße relevante Korrelationsmatrix im Offline-Schritt nur einmalig gelernt werden muss.

Kapitel 5

Experimente

Die in den vorherigen Kapiteln beschriebenen lokalen Features und Techniken zur Erstellung des Codebuches wurden auf dem Datensatz der *Princeton Shape Benchmark* getestet. In diesem Kapitel werden die Daten der Princeton Shape Benchmark genauer vorgestellt und die Ergebnisse des 3D Retrievals auf diesen gezeigt.

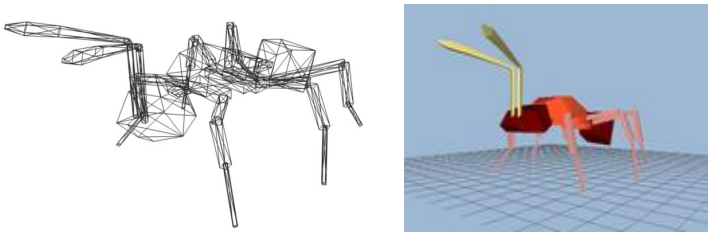
5.1 Princeton Shape Benchmark

Die Princeton Shape Benchmark (PSB) [41] ist ein frei verfügbares Framework zur Evaluation von Shape Retrieval Algorithmen. Es besteht aus einer Sammlung von 1814 dreidimensionalen Modellen, einer Klassifikationsvorschrift und Werkzeugen zur Evaluierung von Retrieval-Methoden. Die Intention hinter der Princeton Shape Benchmark ist eine allgemein gültige Datenbank zur Verfügung zu stellen, die als Referenzmedium für verschiedene Retrieval-Ansätze dient und als Basis zum standardisierten Vergleich angesehen werden kann, was von einigen Arbeiten aufgegriffen wurde [38, 3, 17, 14, 15].

Die 3D Modelle liegen als vektorielle Punkt- und Meshdaten im, von Princeton entwickelten, Objekt File Format (OFF) vor. In Abbildung 5.1 sind sowohl die 3D Punkt- und Meshdaten abgebildet, die das Objekt als Gittermodell beschreiben, als auch ein Bild der Originaldatei im VRML-Format[1].

Die PSB Datenbank liegt in vier semantisch abgestuften Klassifikationsebenen vor, die von Hand angelegt wurden. Diese sind hierarchisch angeordnet und orientieren sich an der

[1] Die Virtual Reality Modeling Language (VRML) ist eine Beschreibungssprache für 3D-Szenen und ein Standardformat für die Repräsentation von interaktiven 3D Vektorgrafiken.



(a) Vektordaten (Gittermodell) (b) Original mit Textur

Abbildung 5.1: 3D Punkt- und Meshdaten des Modells „Ameise“ aus der Princeton Shape Benchmark (PSB) [41].

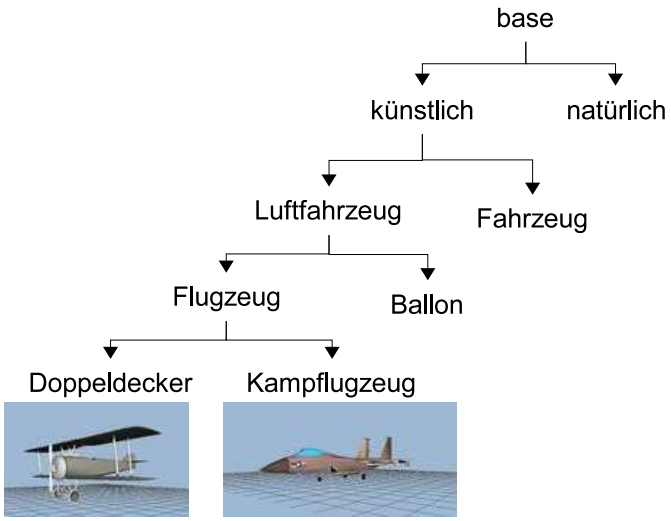


Abbildung 5.2: *base*-Klassifikation der Princeton Shape Benchmark

Funktion und Form eines Objektes (siehe Abbildung 5.2). Die vier Klassen sind *base*, *coarse*, *coarse1* und *coarse2*, aufsteigend von der feinsten Klassifizierung (base) zur gröbsten (coarse2). Zusätzlich wurde die Datenbank in eine Trainings- und Testmenge zu jeweils 907 Objekten aufgeteilt.

In der vorliegenden Arbeit wurden die Experimente auf der feinsten Klassifizierungsstufe, der *base*-Klassifizierung, ausgeführt. Diese Stufe enthält 92 Klassen mit durchschnittlich 10,1 Objekten für den Trainingsdatensatz und 9,9 Objekten für den Testdatensatz. Größte Klassen sind die Klassen „fighter_jet“ und „human“ mit jeweils 50 Modellen.

Volumendaten Rendering

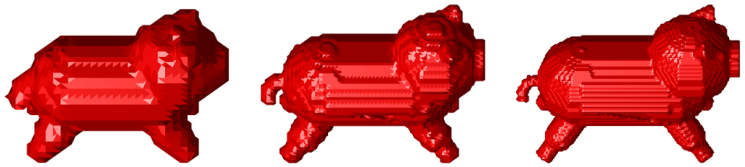
Die PSB-Shapes wurden als Volumendaten in unterschiedlichen Auflösungen gerendert. In Abbildung 5.3 sind diese Renderings für die Auflösungen 32, 64 und 128 Voxel exemplarisch abgebildet. Mit einer Rendering-Auflösung von N^3 ist gemeint, dass die längste Seite der umgebenden Box die Länge N hat. Zum Beispiel ist die Dimensionalität der in 5.3 (b) abgebildeten Ameise $38 \times 64 \times 44$. Während mit höheren Auflösungen offensichtlich der Detaillierungsgrad der Modelle steigt, wächst aber auch der benötigte Rechenaufwand in $O(n^3)$.

Für diese Arbeit wurde eine Auflösung von $N = 64^3$ für alle Modelle benutzt. Die Ameise in 5.3 (b) zum Beispiel besteht für diese Auflösung aus 6010 Voxel, das Schwein in 5.3 (e) aus 24941 Voxel. Die Anzahl der Voxel auf den Oberflächen dieser Objekte sind 4210 Voxel für die Ameise und 9242 Voxel für das Schwein.

5.2 Lernen des Codebuches

Die Erstellung des Codebuches folgt dem in Kapitel 4 vorgestellten Ablauf und ist im Schema 5.4 zum Überblick über die einzelnen Schritte dargestellt. Im Folgenden werden die einzelnen Stufen und verwendeten Parameter näher ausgeführt.

(a) 32^3 (b) 64^3 (c) 128^3



(d) 32^3 (e) 64^3 (f) 128^3

Abbildung 5.3: Als Volumendaten in verschiedenen Auflösungen gerenderte Modelle der Princeton Shape Benchmark.

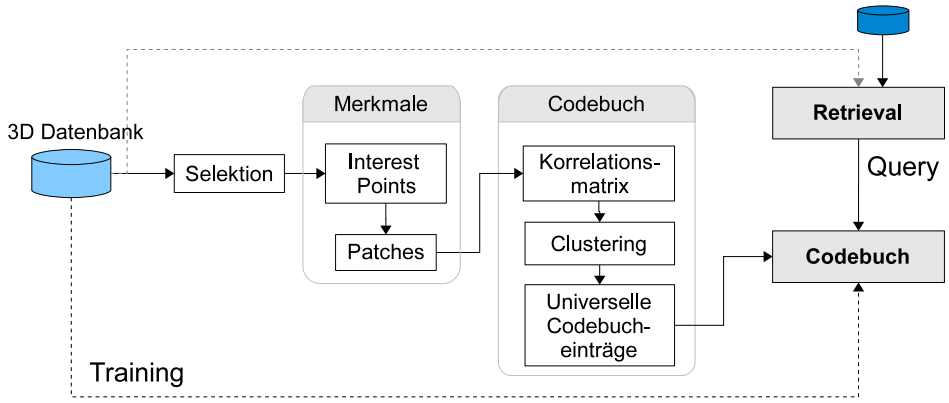


Abbildung 5.4: Arbeitsablauf zum Lernen des Codebuches.

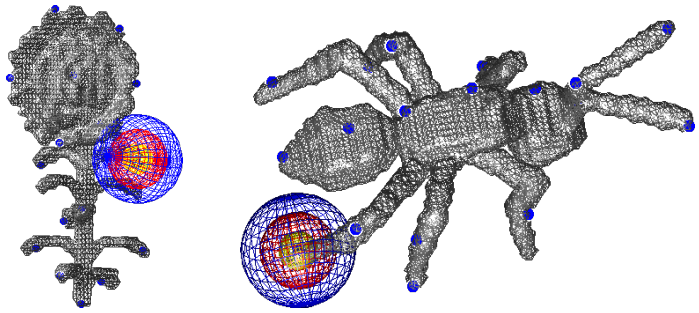


Abbildung 5.5: Beispiele für multi-radien Patches an selektierten Interest Points.

Patches Nach dem Rendering der 3D Modelle aus der Princeton Shape Benchmark als Volumen wurden die phasenbasierten Spherical Harmonics Features an allen Voxeln im Radius 1 erzeugt. Anhand dieser Features wurden dann die Interest Points gesucht, wie in Kapitel 3 gezeigt wird. Als Interest Point Detektor wurde der vorgestellte phasenbasierte Ansatz mit nachträglichem Oberflächensampling angewendet. Durchschnittlich ergeben sich pro Objekt 20 Interest Points (siehe Abbildung 3.10b). Die Patches für die weitere Verarbeitung im Codebuch ergeben sich aus der Entwicklung nach Spherical Harmonics an den Interest Points. Um die lokale Umgebung an einem Punkt zu unterschiedlichen Skalierungen zu erhalten, wurde die Spherical Harmonics Entwicklung für mehrere Radien ausgeführt. Für die 64^3 Volumenmodelle wurden dazu die Radien 3, 6 und 9 benutzt, wie in Abbildung 5.5 an einem ausgewählten Interest Point exemplarisch dargestellt.

Codebuch Für die Erstellung des Codebuches werden eine Stichprobe aus dem Gesamtdatensatz der PSB benutzt. Aus jeder Klasse werden zufällig eine beschränkte Anzahl Modelle ausgewählt, auf denen Patches extrahiert werden. Werden alle Modelle der PSB Datenbank miteinbezogen, so hat sich gezeigt, dass das Codebuch zur Überanpassung neigt, da in diesem Fall die Codebucheinträge zu spezifisch werden. Für die Größe der betrachteten Auswahlmengen wird $N = 90$, $N = 180$ und $N = 270$ gewählt, für jeweils nur ein, zwei oder drei Elemente pro Klasse. Für die Menge $N = 90$, in der nur jeweils ein einziges Modell pro Klasse berücksichtigt wird, ergeben sich - abhängig von der zufälligen Auswahl der Modelle - circa 1800 Patches, für $N = 180$ etwa 3600 und für $N = 270$ etwa 5800 Patches.

Im nächsten Schritt werden diese Patches miteinander verglichen und eine Korrelationsmatrix erstellt. Der normierte Korrelationswert im Bereich zwischen 0 und 1 zweier Patches wird hierzu bezüglich aller betrachteten Radien gemeinsam ermittelt.

Anhand der Korrelationsmatrix werden Klassen von Patches gleicher Art gesucht, indem ein Clustering auf der Matrix ausgeführt wird. Um für die Erstellung der universellen Patches genügend Daten zur Verfügung zu haben, wird die Anzahl der Cluster begrenzt, indem der Cutoff-Wert für das agglomerative Clustering den Wert 0,8 nicht unterschreiten darf. Dies ergibt zum Beispiel für $N = 90$ eine Cluster-Anzahl von 57. In Abbildung 5.6 ist zum Beispiel ein Cluster mit Patches zu sehen, die von spinnenartigen Modellen stammen. Für ein solches Clustering-Ergebnis liefert der kophenetische Korrelationskoeffizient einen Wert von 0,838072. In Abbildung 5.7 sind für die initiale Selektion von $N = 90$ Modellen auf 50 Cluster die Verteilungen der Patches pro Cluster sowie die Anzahl der Cluster pro

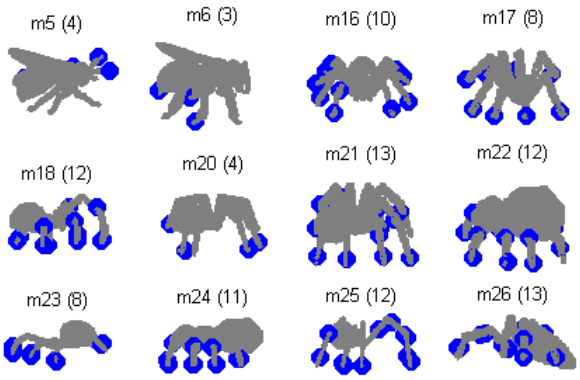


Abbildung 5.6: In einen Cluster gruppierte Patches spinnenartiger Modelle nach agglomerativen Clustering. Die Zahl in der Klammer gibt die Anzahl beteiligter Patches pro Modell an.

Modell dargestellt. Es zeigt sich eine relativ gleichmäßige Verteilung der Patches auf den Clustern.

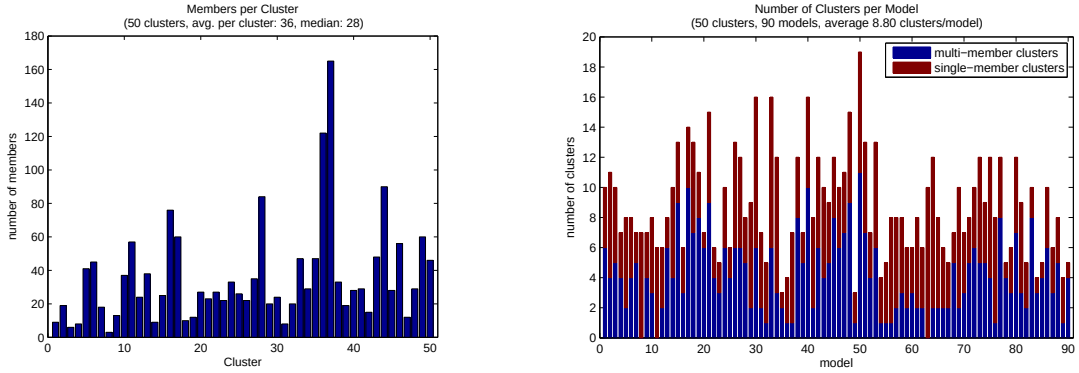
Bezüglich der k-Means basierten Verfahren lässt sich sagen, dass das Clustering der Patches über die Korrelationsmatrix nicht zufriedenstellend funktioniert. Die resultierenden Cluster enthalten entweder extrem viele Elemente, oder nur ein Einziges. Eine eher gleichmäßige Unterteilung, wie sie auf den Daten der PSB zu erwarten ist, konnte nicht erreicht werden.

Für die gruppierten Patches werden nun nach der varianzbasierten Verschmelzungsmethode (siehe Abschnitt 4.3.2) die Cluster-Repräsentanten bestimmt und als neue Patches im Codebuch eingetragen. Da die Spherical Harmonics an einem Patch für mehrere Radien entwickelt wurden, wird die Erstellung der neuen Patches auch pro Radius getrennt vorgenommen. Das neue Patch ist dann wiederum eine multi-radien Repräsentation.

Anhand dieser neuen Patches wird dann das Codebuch trainiert. Dazu werden für alle Modelle der *base*-Klassifikationsstufe in der Princeton Shape Benchmark die Patches extrahiert, und zwar in einer statistisch höheren Anzahl, um relevante Antwort-Histogramme zu erhalten. Dazu wurden für alle Modelle jeweils 100 Patches extrahiert, die von 100 gleichmäßig verteilten Punkten auf der Oberfläche stammen. Dies garantiert, dass nahezu alle Stellen eines Modells repräsentiert werden und die Histogramme das gesamte Objekt abdecken.

Eigene Experimente haben ergeben, dass eine Trennung der Antwort-Histogramme nach Radius bessere Resultate liefert. Daher wird statt einer kombinierten Spherical Harmonics Korrelation eine Korrelation pro Radius ausgeführt. Die Einzelhistogramme pro Radius hintereinander gehängt ergeben die Feature Vektoren für das Retrieval. Die Auftrennung der Radien ermöglicht eine höhere Sensitivität der Histogramme bezüglich unterschiedlich skalierter Patches.

Auf dem gelernten Codebuch werden beim Retrieval der Modelle aus dem Testdatensatz der PSB nach derselben Methode wie beim Lernprozess Histogramme pro Query-Modell erstellt. Eine Anfrage an das Codebuch liefert über den Vergleich der Histogramme eine Distanz, mit der die Ähnlichkeiten zwischen dem Query-Modell und allen anderen Mo-



(a) Patches pro Cluster

(b) Cluster pro Modell

Abbildung 5.7: Anzahl der Patches pro Cluster (a) und Anzahl der Cluster pro Modell (b) mit Differenzierung der Cluster-übergreifenden Verteilungen. Bei Multi-Member Clustern verteilen sich die Cluster-Zuordnungen über Objektgrenzen hinweg.

dellen in der Datenbank bestimmt werden kann. Im Idealfall würde das Query-Modell bezüglich allen anderen Modellen in der Datenbank die niedrigste Distanz aufweisen.

5.3 Ergebnisse

Die Ergebnisse des vorgestellten Verfahrens liegen für den gesamten Datensatz der *base*-Klassifikationsstufe im Bereich von 26% - 30% Genauigkeit für die Nächste-Nachbar-Klassifikation (NN) und um 39% Genauigkeit beim Discounted Cumulative Gain (DCG). Siehe dazu die Auflistung in Tabelle 5.1.

Im Vergleich mit anderen 3d Shape Deskriptoren aus der PSB schneidet das eigene Verfahren vergleichsweise schlecht ab. Eine Auflistung der Ergebnisse dieser Verfahren im Vergleich mit dem eigenen ist in Tabelle 5.2 zu sehen. Diese Tabelle ist ein Ausschnitt einer umfassenderen Evaluation aus der PSB [41].

Zum Vergleich wurde der *Spherical Harmonic Descriptor* (SHD) von Kazhdan *et* al. [26] implementiert. Dieser 3D Shape Deskriptor erzeugt eine rotationsinvariante Repräsentation der Modelle, indem für mehrere Radien die Norm der Spherical Harmonic Frequenzen in einem 2D Histogramm gespeichert werden.

Die anderen Verfahren im Vergleich sind:

- Extended Gaussian Images (EGI) [22]: Die Verteilung der Oberflächennormalen dient als 3D Deskriptor. Auch hier werden sphärische Funktionen für den Deskriptor benutzt.
- D2 Shape Distribution (D2) [36]: Histogramm über die Verteilung paarweise verschiedener Punkte auf der Oberfläche.
- Shape Histogramme (SHELLS) [2]: Ähnlich wie D2, aber als Histogramm über die Distanzen zwischen dem Massenmittelpunkt und Punkten auf der Oberfläche.

Codebuchgröße	NN[1]	1st-Tier	2nd-Tier	E-Measure	DCG
40	30%	12%	16%	10%	39%
50	29%	10%	15%	9%	38%
57	27%	10%	14%	9%	37%
92	26%	10%	15%	9%	37%

Tabelle 5.1: Ergebnisse des vorgestellten Verfahrens für unterschiedliche Codebuchgrößen. [1]NN: Nächste-Nachbar-Klassifikation

Verfahren	NN	1st-Tier	2nd-Tier	E-Measure	DCG
LFD	65,7%	38,0%	48,7%	28,0%	64,3%
SHD	55,6%	30,9%	41,1%	24,1%	58,4%
EGI	37,4%	19,7%	27,7%	16,5%	47,2%
D2	31,1%	15,8%	23,5%	13,9%	43,4%
Eigenes Verfahren	30,3%	11,6%	16,3%	10,1%	38,9%
SHELLS	22,7%	11,1%	17,3%	10,2%	38,6%

Tabelle 5.2: Vergleich mit fünf anderen Algorithmen aus der PSB (Ausschnitt einer Vergleichstabelle mit 12 Deskriptoren nach [41]). [1]NN: Nächste-Nachbar-Klassifikation.

- Light Field Deskriptor (LFD) [8]: Für ein Modell werden aus mehreren Blickwinkeln Bilder aufgenommen. Die Ausgangspositionen für die Aufnahmen ergeben sich aus zufällig ausgewählten Punkten auf einer das Modell umgebenden Kugel.

Zum grafischen Vergleich sind im Precision-Recall-Plot (Abbildung 5.8) die Ergebnisse bezüglich mehrere Codebuchgrößen aufgetragen. Die niedrigeren Genauigkeiten spiegeln sich in einer stärker gekrümmten Kurve wider.

Im Rangordnungsdiagramm 5.9 lässt sich erkennen, dass das eigene Verfahren zwar für große Gruppen eine ausreichende Genauigkeit erreicht, bei vielen anderen Klassen sind die Treffer aber weit entfernt von der Diagonale zu finden, was auf eine niedrige Diskriminierbarkeit der eigenen Features schließen lässt.

In der Tabelle 5.3 ist die Retrieval-Performance für die besten acht Klassen aufgetragen. Für Modelle der Klassen „Ablagen“, „Kampfflugzeuge“ und „Menschen“ ergibt sich eine NN-Genauigkeit von über 80%, und für die Klassen „Ballone“, „Regenschirme“, „Rennautos“, „Köpfe“ und „Raumfähren“ sind es noch über 60%. Zum Gegenvergleich sind auch die acht Klassen mit der schlechtesten Performance aufgelistet.

5.4 Diskussion

Vergleicht man das vorgestellte Verfahren mit den Ergebnisse anderer Arbeiten, so ist das erreichte Ergebnis nicht zufriedenstellend. Die Gründe dafür sind beim Clustering und der Erstellung der Codebucheinträge zu vermuten.

	Möbel-Ablagen	Kampf-flugzeuge	Menschen	Ballone	Regen-schirme	Renn-autos	Köpfe	Raum-fähren
NN	85%	82%	80%	67%	67%	64%	63%	60%
DCG	51%	75%	71%	50%	48%	50%	52%	39%

	Äxte	Schlangen	Schultische	Räder	Zelte	Satelliten-schüssel	Hämmer	Stealth-Bomber
NN	0%	0%	0%	0%	0%	0%	0%	0%
DCG	13%	14%	14%	14%	15%	15%	15%	16%

Tabelle 5.3: Retrieval Performance bezüglich den besten (oben) und den schlechtesten (unten) acht Klassen, sortiert nach NN. Menschen (2): Modelle mit ausgestreckten Armen.

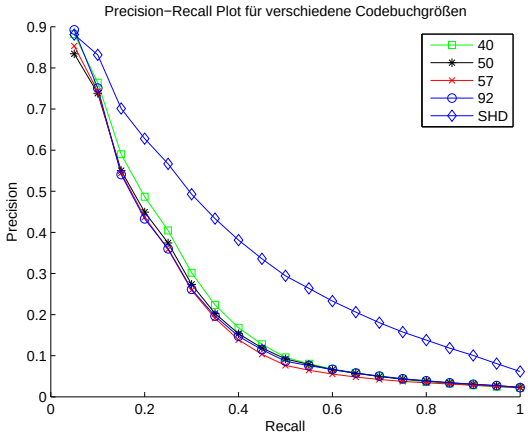
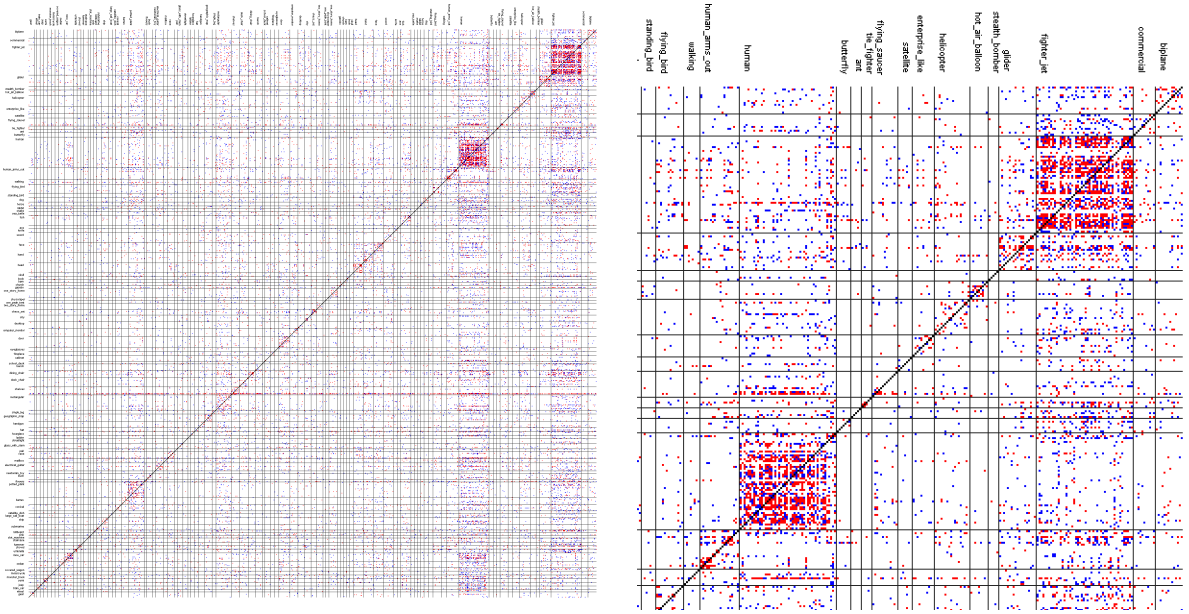


Abbildung 5.8: Precision-Recall Plot für vier ausgewählte Retrieval-Ergebnisse und deren Codebuchgrößen.



(a) Komplett (b) Ausschnitt

Abbildung 5.9: Rangordnungsdiagramm. Schwarze Pixel auf der Diagonalen sind exakte Treffer nach NN, rote Pixel: *1st-Tier*, blue Pixel: *2nd-Tier*.

Die Anzahl der Modelle bei der initialen Selektion zur Auswahl der Patches als auch beim Training des Codebuches spielen auch eine große Rolle. In den Experimenten konnte beobachtet werden, dass die Ergebnisse für eine kleine Anzahl Modelle ($N = 90$) bessere Ergebnisse liefert als für größere, zum Beispiel für $N = 270$. Eine derart kleine Selektion stützt sich aber auf die Qualität der vorgegebenen Klassifikationsvorschrift. Im Falle der Princeton Shape Benchmark wurden die Objekte sorgfältig in Klassen einsortiert. Für weniger gute Vorklassifikationen ist der vorgestellte Ansatz zur Selektion initialer Modelle nicht ohne Weiteres übertragbar. Werden alle Modelle für die initiale Selektion verwendet, so lässt sich eine Überanpassung beobachten, und die Ergebnisse fallen wesentlich schlechter aus.

Experimente mit der Korrelationsmatrix nach der Erzeugung der Patches ergaben, dass hier eine hohe Wiedererkennungsrate erreicht werden kann, wenn nur die Spherical Harmonics Patches an sich verglichen werden. Der Abstieg der Genauigkeit ist demnach in den nachgelagerten Schritten beim Clustering und der Erzeugung der universellen Codebucheinträge zu suchen.

Die Evaluation des Clusterings der Patches stellte sich als aufwändig und fehleranfällig dar. Trotz der recht guten kophenetischen Relationswerte von 0,8 bis 0,9 waren die Ergebnisse nicht signifikant besser. Das agglomerative Clustering lieferte im Vergleich zu k-Means noch die besseren Ergebnisse. Die Schwierigkeit in der Evaluation liegt, wie im Grundlagenkapitel erwähnt, in der fehlenden Ground-Truth. Da die Schritte bei der Erstellung des Codebuches auf den Clustering-Ergebnissen aufbauen, konnte eine Evaluation erst nach der Fertigstellung des Codebuches durchgeführt werden. Die visuelle Inspektion

der Clustering-Ergebnisse ergab immer recht gute Resultate (siehe zum Beispiel Abbildung 5.6).

Die Parametersuche für die einzelnen Schritte zur Erstellung des Codebuches erwiesen sich als zeitaufwändig und fehleranfällig.

5.5 Laufzeiten

Eine Anfrage an das Codebuch ist abhängig von der gewählten Codebuchgröße und den Patches auf dem Query-Objekt. Für eine Codebuchgröße von 50 und 100 Patches auf dem Query-Modell liegt die Anfragezeit im Bereich von weniger als 10 Sekunden.

Die Berechnungen der phasenbasierten Features, die Detektion der Interest Points und die Extraktion der Patches läuft innerhalb von circa 2 Sekunden ab. Ausgenutzt wurde hierbei, dass die Routinen zur Berechnung und Korrelation der Spherical Harmonics parallelisiert werden konnten. Dafür standen Rechner mit bis zu 16 Prozessorkernen und jeweils 3 GHz Taktung zur Verfügung. Für die Berechnung mehrerer Spherical Harmonics Entwicklungen auf vielen Modellen und der anschließenden Bildung der Korrelationsmatrix war es von Vorteil, dass die Rechner mehrere Gigabyte an Arbeitsspeicher zur Verfügung stellten.

Trotz Parallelisierung benötigte die Erstellung des Codebuches und die darauf folgenden Query-Anfragen auf dem gesamten Datensatz der Princeton Shape Benchmark in der Regel mehrere Stunden. Da die Erstellung des Codebuches zum Offline-Schritt des Retrieval gehört wird eine Beschleunigung als nicht primär notwendig erachtet. Eine Optimierung könnte eher an den Schritten ansetzten, die auf das Codebuch aufbauen. Hier könnte durch die Implementierung geeigneter Lookup-Table Techniken die Zeit wesentlich verkürzt werden, indem die Spherical Harmonics Koeffizienten und die Korrelationswerte zwischen verschiedenen Patches vorgehalten werden.

Kapitel 6

Zusammenfassung & Ausblick

Diese Arbeit stellt ein neues Verfahren für das 3D Shape Retrieval vor, das auf lokalen Merkmalen und einer Indexstruktur in Form eines Codebuches basiert. Dazu wurden Interest Point Detektoren entwickelt und ein Verfahren zur Erstellung eines universellen Codebuches vorgeschlagen.

Zur Detektion von Interest Points wurden zwei Methoden entwickelt, die für 3D Modelle die charakteristischen Stellen finden. Neben einer Clustering-basierten Methode, die Regionen gleicher Art innerhalb eines Modells markiert und in diesen Regionen die Interest Points findet, wurde eine Methode vorgeschlagen, die nur auf den phasenbasierten Spherical Harmonics Features beruht, und die numerisch stabile Positionen ermittelt. Neben Methoden zum gleichmäßigen Sampling von Interest Points auf den Modellen wurden auch Methoden vorgestellt um redundante Positionen zu eliminieren.

Für die Vorstufe der lokalen Features werden in Form von Spherical Harmonics Repräsentationen lokale Patches an den Interest Points extrahiert, die für die Erstellung des Codebuches dienen.

Für die Erstellung eines universellen Codebuches wurde ein Verfahren vorgeschlagen, das auf dem Bag-of-Features Konzept beruht. Das Codebuch enthält in Form von Patch-Histogrammen die lokalen Features. Zur Bildung von Universal-Patches im Codebuch werden Verfahren zur Bestimmung bester Cluster-Repräsentanten präsentiert. Für die Repräsentation der Codebuch-Histogramme wurde eine Technik zur weichen Abbildung der Verteilungen über mehrere Patches vorgestellt, die auf den Fuzzy-Histogrammen beruht.

Das entwickelte Verfahren wurde auf den Daten der Princeton Shape Benchmark [41] evaluiert. Auf einigen Klassen funktioniert das Verfahren sehr gut, ist aber bezüglich den Gesamtdaten ausbaufähig. Durch die beschriebenen Methoden wird aber eine Grundlage für ähnliche, Codebuch-basierte 3D Shape Retrieval Verfahren gelegt.

Einige Fragen bezüglich der Codebuchgröße und der Auswahl geeigneter Patches für das universelle Codebuch sind noch offen. Hier müssen Verfahren modelliert werden, die eine relevante Auswahl von initialen Selektionen für den Codebuch-Lernprozess liefern können. Für ein flexibles Retrieval sind auch Verfahren vorstellbar, die eine dynamische Codebuchgröße berücksichtigen und ein inkrementelles Clustering anbieten.

Positionshistogramme Ein Vorschlag zur Verbesserung der Retrievalergebnisse zielt auf den histogrammbasierten Vergleich:

Statt die Histogramme nur als Antworten auf den Codebucheinträgen zu kodieren, besteht die Möglichkeit die Positionen der Features auf dem Query-Objekt mitzuberücksichtigen. Um Rotationsinvarianz zu gewährleisten werden die Antworthistogramme an

Punkten auf der Oberfläche einer das Objekt umgebenden Kugel gebildet. Diese Punkte auf der Oberfläche ergeben sich aus der Projektion der Interest Points auf die nächstgelegenen Oberflächenpunkte der Kugel. Diese können durch den geometrischen Schnitt einer Geraden durch den Objektmittelpunkt und einem Interest Point bestimmt werden. In Abbildung 6.1 ist dies exemplarisch dargestellt. Diese Methode wird wahrscheinlich positive Auswirkungen auf die Retrievalergebnisse haben, da hier auch die geometrische Information über die Position der Patches in die Histogramme miteinfließt.

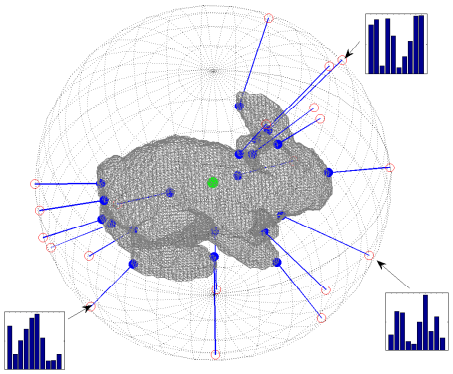


Abbildung 6.1: Projektion von Merkmalspositionen auf eine Kugeloberfläche. Blau markierte Punkte sind die Features, rot markierte die projizierten Punkte. Der grüne Punkt ist der Mittelpunkt. An drei Stellen sind beispielhaft Histogramme für die Antworten auf dem Codebuch eingezeichnet.

Anhang A

Beispielmodelle aus der Princeton Shape Benchmark

Anhang B

Verwendete Software und Bibliotheken

Diese Diplomarbeit wurde mit der Programmiersprache *C++* unter Linux entwickelt und nutzt für die Verarbeitung und Speicherung der Daten etablierte Softwaretools und Bibliotheken. Während die rechenintensiven Algorithmen in C++ implementiert wurden, wurden für die Evaluation der Ergebnisse und die weitere Verarbeitung und Visualisierung der Daten *Matlab* benutzt. Für automatisierte Routinearbeiten wurden Skripte in der Programmiersprache *Python* geschrieben.

Matlab Das von der Firma MathWorks, Inc. entwickelte Werzeug eignet sich unter anderem zur Lösung mathematischer Probleme, zur schnellen Entwicklung von Algorithmen, zur Analyse von Datensätzen und zur Visualisierung. Die in dieser Arbeit enthaltenen Graphiken wurden größtenteils mit Matlab erstellt.

Blitz++ Die in C++ entwickelten Algorithmen nutzten als Datentyp mehrdimensionale *Arrays*, die von der C++-Klassenbibliothek Blitz++ zur Verfügung gestellt werden. Wegen der templatebasierten und hoch optimierten Implementierung wurden die Datentypen dieser Bibliothek für alle Array-basierten Daten verwendet. Blitz++ ist frei verfügbar.

NetCDF, HDF5 NetCDF (Network Common Data Form) und HDF5 (Hierarchical Data Format, Version 5) sind plattformunabhängige Datenformate zur optimierten Speicherung großer Daten. Da speziell n-dimensionale oder Array-strukturierte Daten unterstützt werden, bieten sich diese Formate für die Speicherung der 3D Daten und des Codebuches an.

Weka Weka (Waikato Environment for Knowledge Analysis) von der University of Wikato [53] ist eine freie Sammlung von Machine Learning Software, implementiert in Java. Für die Entwicklung und das Testen von Cluster-Verfahren wurde auf Weka zurückgegriffen. Das EM-basierte Clustering wurde mittels den von Weka bereitgestellten Bibliotheken in Java implementiert.

LMB Tools Die vom Lehrstuhl für Mustererkennung entwickelten Bibliotheken konnten in dieser Arbeit genutzt werden. Speziell die umfangreiche C++ Bibliothek *libFeature3D* [13, 15] und die NetCDF- und HDF5-Pakete wurden genutzt.

Abbildungsverzeichnis

Literaturverzeichnis

[1] Agarwal, S. und A. Awan: *Learning to Detect Objects in Images via a Sparse, Part-Based Representation*. IEEE Trans. Pattern Anal. Mach. Intell., 26(11):1475–1490, 2004. Member-Dan Roth. 4.2

[2] Ankerst, M., G. Kastenmüller, H.-P. Kriegel und T. Seidl: *Nearest Neighbor Classification in 3D Protein Databases*. In: *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, S. 34–43. AAAI Press, 1999. 5.3

[3] Ansary, T. F., M. Daoudi und J.-P. Vandeborre: *A Bayesian 3-D Search Engine Using Adaptive Views Clustering*. IEEE Transactions on Multimedia, 9(1):78–88, 2007. 1.4, 5.1

[4] Böhm, C., S. Berchtold und D. A. Keim: *Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases*. ACM Comput. Surv., 33(3):322–373, 2001. 3.1

[5] Burel, G. und H. Henoco: *Determination of the orientation of 3D objects using spherical harmonics*. Graph. Models Image Process., 57(5):400–408, 1995. 2.4

[6] Bustos, B., D. Keim, D. Saupe und T. Schreck: *Content-Based 3D Object Retrieval*. Computer Graphics and Applications, IEEE, 27(4):22–27, 2007. 1.4

[7] Bustos, B., D. A. Keim, D. Saupe, T. Schreck und D. V. Vrani: *Feature-based similarity search in 3D object databases*. ACM Comput. Surv., 37(4):345–387, December 2005. 1.4

[8] Chen, D.-Y., X.-P. Tian, Y.-T. Shen und M. Ouhyoung: *On Visual Similarity Based 3D Model Retrieval*. Computer Graphics Forum, 2003. 1.4, 5.3

[9] Dance, C., J. Willamowski, L. Fan, C. Bray und G. Csurka: *Visual categorization with bags of keypoints*. In: *ECCV International Workshop on Statistical Learning in Computer Vision*, 2004. 1.4, 4.2

[10] Dempster, A. P., N. M. Laird und D. B. Rubin: *Maximum Likelihood from Incomplete Data via the EM Algorithm*. Journal of the Royal Statistical Society. Series B (Methodological), 39(1):1–38, 1977. 2.6.2

[11] Ding, C. und X. He: *K-means clustering via principal component analysis*. In: *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, S. 29, New York, NY, USA, 2004. ACM. 2.6.1, 2.6.1

[12] Faloutsos, C.: *Searching Multimedia Databases by Content*. Kluwer Academic Publishers, Norwell, MA, USA, 1996. 3.1

[13] Fehr, J. und H. Burkhardt: *Phase based 3D Texture Features*. In: *Proceedings of the 28th Pattern Recognition Symposium of the German Association for Pattern Recognition (DAGM 2006), Berlin, Germany*, S. 263–272. LNCS, Springer, 2006. 3.2.1, B

[14] Fehr, J. und H. Burkhardt: *Harmonic Shape Histograms for 3D Shape Classification and Retrieval*. In: *IAPR Workshop on Machine Vision Applications (MVA2007)*, Tokyo, Japan, 2007. 1.4, 2.3, 5.1

[15] Fehr, J., M. Reisert und H. Burkhardt: *Fast and Accurate Rotation Estimation on the 2-Sphere without Correspondences*. In: Forsyth, D., P. Torr und A. Zisserman (Hrsg.): *Computer Vision – ECCV 2008*, Bd. 5303 d. Reihe *LNCS - Lecture Notes in Computer Science*, S. 239–251. Springer, October 2008. 1.4, 2.4, 2.6, 4.3.2, 5.1, B

[16] Gonzalez, R. C. und R. E. Woods: *Digital Image Processing (3rd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2006. 2.5

[17] Goodall, S., P. Lewis und K. Martinez: *3-D Shape Descriptors and Distance Metrics for Content-Based Artefact Retrieval*. In: Lienhart, R. W., N. Babaguchi und E. Y. Chang (Hrsg.): *Storage and Retrieval Methods and Applications for Multimedia 2005*, Bd. 5682, S. 87–97. SPIE-IS&T, 2005. 1.4, 5.1

[18] Green, R.: *Spherical Harmonic Lighting: The Gritty Details*. Archives of the Game Developers Conference, March 2003. 2.3, 2.3, 2.5

[19] Groemer, H.: *Geometric Applications of Fourier Series and Spherical Harmonics*. Cambridge University Press, Cambridge, 1996. 2.3

[20] Hilaga, M., Y. Shinagawa, T. Kohmura und T. L. Kunii: *Topology matching for fully automatic similarity estimation of 3D shapes*. In: *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, S. 203–212, New York, NY, USA, 2001. ACM Press. 1.4, 3.3

[21] Hobson, E.: *The Theory of Spherical and Ellipsoidal Harmonics*. Chelsea Pub Co, 1955. 2.3, 2.3

[22] Horn, B. K. P.: *Extended gaussian images*. Proceedings of the IEEE, 72(2):1671–1686, 1984. 5.3

[23] Huber, D., A. Kapuria, R. Donamukkala und M. Hebert: *Parts-based 3D object classification*. In: *Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, Bd. 2, S. II–82–II–89 Vol.2, 2004. 1.4, 4.2

[24] Jain, A. K. und R. C. Dubes: *Algorithms for clustering data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 3 1988. 2.6.3

[25] Jurie, F. und B. Triggs: *Creating efficient codebooks for visual recognition*. Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on, 1:604–610 Vol. 1, Oct. 2005. 1.4

[26] Kazhdan, M., T. Funkhouser und S. Rusinkiewicz: *Rotation invariant spherical harmonic representation of 3D shape descriptors*. In: *SGP '03: Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, Bd. 1, S. 156–164, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association. 1.4, 2.3, 3.3, 5.3

[27] Kendall, D. G., D. B. (Author), C. (Author) und H. Le: *Shape and Shape Theory*. Wiley & Sons, 1999. 1

[28] Knuth, D. E.: *Art of Computer Programming, Volume 2: Seminumerical Algorithms (3rd Edition)*. Addison-Wesley Professional, November 1997. 4.3.2

[29] Kovacs, J. A. und W. Wriggers: *Fast rotational matching*. Acta Crystallographica Section D, 58(8):1282–1286, Aug 2002. 2.4

[30] Lowe, D. G.: *Object Recognition from Local Scale-Invariant Features*. In: *Proc. of the International Conference on Computer Vision ICCV, Corfu*, S. 1150–1157, 1999. 3.1, 3.2

[31] Makadia, A., L. Sorgi und K. Daniilidis: *Rotation Estimation from Spherical Images*. In: *ICPR '04: Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 3*, S. 590–593, Washington, DC, USA, 2004. IEEE Computer Society. 2.4

[32] Martínez, J. M.: *MPEG-7: Overview of MPEG-7 Description Tools, Part 2*. IEEE MultiMedia, 9(3):83–93, 2002. 1.4

[33] Mousa, M., R. Chaine und S. Akkouche: *Frequency-Based Representation of 3D Models using Spherical Harmonics*, 1 2006. The 14-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision. 1.4

[34] Mousa, M., R. Chaine und S. Akkouche: *Frequency-based representation of 3D point-based surfaces using spherical harmonics*. MG&V, 15(3):537–546, 2006. 1.4

[35] Nowak, E., F. Jurie und B. Triggs: *Sampling Strategies for Bag-of-Features Image Classification*. In: *Computer Vision ECCV 2006*, S. 490–503. Springer, 2006. 1.4, 3.4, 4.1, 4.1

[36] Osada, R., T. Funkhouser, B. Chazelle und D. Dobkin: *Matching 3D Models with Shape Distributions*. In: *SMI '01: Proceedings of the International Conference on Shape Modeling & Applications*, S. 154, Washington, DC, USA, 2001. IEEE Computer Society. 5.3

[37] Pelleg, D.: *X-means: Extending K-means with efficient estimation of the number of clusters*. In: *In Proceedings of the 17th International Conf. on Machine Learning*, S. 727–734. Morgan Kaufmann, 2000. 2.6.1

[38] Reisert, M. und H. Burkhardt: *Feature Selection for Retrieval Purposes.*. In: Campilho, A. C. und M. S. Kamel (Hrsg.): *ICIAR (1)*, Bd. 4141 d. Reihe *Lecture Notes in Computer Science*, S. 661–672. Springer, 2006. 1.4, 5.1

[39] Salton, G. und M. J. McGill: *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1986. 2.2

[40] Saupe, D. und D. V. Vranic: *3D Model Retrieval with Spherical Harmonics and Moments*. In: *Symposium on Pattern Recognition*, S. 392–397, London, UK, 2001. Springer-Verlag. 1.4

[41] Shilane, P., P. Min, M. Kazhdan und T. Funkhouser: *The Princeton Shape Benchmark*. In: *Shape Modeling and Applications*, S. 167–178, 2004. 1, 1.4, 2.3, 2.2, 2.2, 5.1, 5.1, 5.3, 5.2, 6

[42] Siggelkow, S. und H. Burkhardt: *Improvement of Histogram-Based Image Retrieval and Classification*. In: *ICPR '02: Proceedings of the 16 th International Conference on Pattern Recognition (ICPR'02) Volume 3*, S. 30367, Washington, DC, USA, 2002. IEEE Computer Society. 4.4, 4.7

[43] Sokal, R. und F. Rohlf: *The comparison of dendrograms by objective methods*. Taxon, 1962. 2.7.2

[44] Sundar, H., D. Silver, N. Gagvani und S. Dickinson: *Skeleton based shape matching and retrieval*. In: *Shape Modeling International, 2003*, S. 130–139, 2003. 1.4, 3.3

[45] Tangelder, J. W. und R. C. Veltkamp: *A survey of content based 3D shape retrieval methods*. In: *SMI '04: Proceedings of the International Conference on Shape Modeling Applications*, S. 145–156, Washington, DC, USA, 2004. IEEE Computer Society. 1.4

[46] Temerinac, M., M. Reisert und H. Burkhardt: *SHape REtrieval Contest 2007: Protein Retrieval Track*. In: *Proceedings of SHREC2007 3D Shape Retrieval Contest, SMI'07*, Lyon, France, 2007. 1.4

[47] Tishby, N., F. Pereira und W. Bialek: *The information bottleneck method*. In: *Proceedings of the 37-th Annual Allerton Conference on Communication, Control and Computing*, S. 368–377, 1999. 4.3

[48] Vapnik, V. N.: *Statistical Learning Theory*. Wiley-Interscience, September 1998. 2.6.1

[49] Vranic, D. V.: *An improvement of rotation invariant 3D-shape based on functions on concentric spheres*. In: *International Conference on Image Processing*, Bd. 3, S. 757–760, 2003. 1.4

[50] Vranic, D. V. und D. Saupe: *Description of 3D-Shape using a Complex Function on the Sphere*. In: *Proc. IEEE International Conference on Multimedia and Expo*, 2002. 1.4

[51] Weber, M.: *Unsupervised learning of models for object recognition*. Doktorarbeit, California Institute of Technology, Pasadena, CA, USA, 2000. Supervisor-Pietro Perona. 4.2

[52] Winn, J., A. Criminisi und T. Minka: *Object Categorization by Learned Universal Visual Dictionary*. In: *ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision*, Bd. 2, S. 1800–1807, Washington, DC, USA, 2005. IEEE Computer Society. 1.4, 4.2, 4.3

[53] Witten, I. H. und E. Frank: *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, June 2005. WEKA. 2.6.1, 2.6.2, B