

**3D GEOMETRIC HASHING USING TRANSFORM INVARIANT
FEATURES**

**A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES**

**OF
MIDDLE EAST TECHNICAL UNIVERSITY**

BY

ÖMER ESKİZARA

IN PARTIAL FULLFILLMENT OF THE REQUIREMENTS

FOR

THE DEGREE OF MASTER OF SCIENCE

IN

ELECTRICAL AND ELECTRONICS ENGINEERING

APRIL 2009

Approval of the Thesis

**3D GEOMETRIC HASHING USING TRANSFORM INVARIANT
FEATURES**

Submitted by **Ömer Eskizara** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. İsmet Erkmen
Head of Department, **Electrical and Electronics Eng.** _____

Assist. Prof. İlkay Ulusoy
Supervisor, **Electrical and Electronics Eng.** _____

Examining Committee Members

Prof. Dr. Uğur Halıcı
Electrical and Electronics Eng., METU _____

Assist. Prof. İlkay Ulusoy
Electrical and Electronics Eng., METU _____

Prof. Dr. Gözde Bozdağı Akar
Electrical and Electronics Eng., METU _____

Assoc. Prof. H. Şebnem Düzgün
Geodetic and Geographic Inf. Tech., METU _____

M.Sc. Murat Yirci
Image Processing Dep., ASELSAN _____

Date: 10.04.2009

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Ömer Eskizara

Signature :

ABSTRACT

3D GEOMETRIC HASHING USING TRANSFORM INVARIANT FEATURES

Eskizara, Ömer

M.S., Department of Electrical and Electronics Engineering

Supervisor: Assist. Prof. İlkey Ulusoy

April 2009, 93 pages

3D object recognition is performed by using geometric hashing where transformation and scale invariant 3D surface features are utilized. 3D features are extracted from object surfaces after a scale space search where size of each feature is also estimated. Scale space is constructed based on orientation invariant surface curvature values which classify each surface point's shape. Extracted features are grouped into triplets and orientation invariant descriptors are defined for each triplet. Each pose of each object is indexed in a hash table using these triplets. For scale invariance matching, cosine similarity is applied for scale variant triple variables. Tests were performed on Stuttgart database where 66 poses of 42 objects are stored in the hash table during training and 258 poses of 42 objects are used during testing. %90.97 recognition rate is achieved.

Keywords: 3D object recognition, geometric hashing

ÖZ

YER DEĞİŞTİRMEYEN VE DÖNÜŞÜMDEN BAĞIMSIZ ÖZNETELİK KULLANILARAK 3B GEOMETRİK KIYIM

Eskizara, Ömer

Yüksek Lisans, Elektrik Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Yrd. Doç. İlkay Ulusoy

Nisan 2009, 93 sayfa

Dönüşümden ve büyüklükten bağımsız 3B yüzey öznitelikleri kullanılarak 3B cisim tanıma çalışması yapıldı. Ölçek uzayı taraması yapılarak cisimlerin yüzeylerindeki 3B öznitelikler elde edilerek her özniteliğin büyüklüğü elde edilir. Ölçek uzayı her yüzey noktasının şeklini belirten dönüşümden bağımsız yüzey eğrilik değerleriyle oluşturulur. Elde edilen öznitelikler üçlüler halinde gruplanır ve her üçlülerde dönüşümden bağımsız ayıraçlar tanımlanır. Bu üçlüler kullanılarak her cismin her farklı pozunu kıyım tablosuna indekslenir. Büyüklüğe duyarlı değerler için kosinüs benzerlik kullanılarak ölçeğe duyarlı olmayan eşleştirme elde edilir. Yapılan testlerde Stuttgart veri tabanında her 42 cisim için 66 poz kıyım tablosuna eğitim safhasında saklandı ve test aşamasında her 42 cismin 256 pozunu kullanıldı. %90.97 oranında tanıma başarısı sağlandı.

Anahtar Kelimeler: 3B cisim tanıma, kıyım yöntemi

ACKNOWLEDGEMENT

I would like to express my deepest gratitude and appreciation to my supervisor Assist. Prof. İlkey Ulusoy, who inspired, encouraged and supported me at all levels of this study that is prepared in METU Computer Vision & Intelligent Systems Research Lab.

I would like to thank to Erdem Akagündüz who helped me in all technical topics and encouraged me after each small improvement in my study, to Nazlı Özden Akçay, Ahmet Oğuz Öztürk and Mert Erkan Elalmış for their help in English translation, Teoman Ünal and Yasın Kaygusuz for their help in thesis format and appearance and to Başar Dalkılıç for his help in database queries.

The greatest thanks go to my family members and all my friends for their infinite support.

TABLE OF CONTENTS

ABSTRACT	iv
ÖZ	v
ACKNOWLEDGEMENT	vi
TABLE OF CONTENTS	vii
LIST OF FIGURES	ix
LIST OF TABLES.....	xii
CHAPTER.....	1
1 INTRODUCTION	1
1.1 Motivation.....	1
1.2 Literature.....	2
1.3 Scope of the Thesis.....	4
1.4 Organization of the Thesis	6
2 TRANSFORM INVARIANT FEATURES	7
2.1 Curvature.....	7
2.2 Curves on Surfaces	8
2.3 Gaussian and Mean Curvatures.....	9
2.3.1 Gaussian Curvature (K)	9
2.3.2 Mean Curvature (H).....	9
2.4 HK space	9
2.5 Features Obtained	12
3 FEATURE VECTOR CONSTRUCTION FOR GEOMETRIC HASHING	18
3.1 Feature Grouping.....	19
3.2 Optional feature vector values:.....	20
4 GEOMETRIC HASHING USING TRANSFORM INVARIANT FEATURE VECTORS FOR OBJECT RECOGNITION	25
4.1 Hashing	25
4.2 Geometric Hashing	26

4.2.1	Preprocessing	26
4.2.2	Recognition	28
4.3	Hashing Method Proposed	29
4.3.1	Euclidean Similarity	33
4.3.2	City Block Similarity	34
4.3.3	Cosine Similarity	35
4.4	Geometric Meaning of the Similarities.....	37
4.5	Reason for Indexing by Hashing	40
5	EXPERIMENTS AND RESULTS	42
5.1	Base Test:.....	45
5.2	Elimination Test:	46
5.3	Feature Group Test	48
5.4	Length Values Added Test.....	50
5.5	Individual Max Test.....	52
5.6	Angle Tests.....	54
5.7	Length Tests	55
5.8	Normal Angle Tests.....	57
5.9	Type Tests	58
5.10	Feature Number Tests	60
5.11	City Block Similarity Test:.....	61
5.12	Cosine Similarity Test.....	63
5.13	Cosine Similarity Test With Scaled Database	65
5.14	Computation Time Tests	67
5.15	Best Fitting Tests	69
5.16	Analysis of the Overall Results	73
6	CONCLUSION.....	75
6.1	Work Done.....	75
6.2	Future Works.....	76
	REFERENCES	77
	APPENDIX A	80
	APPENDIX B.....	86
	APPENDIX C.....	90

LIST OF FIGURES

Figure 1: Preprocessing phase of object recognition process	5
Figure 2: Recognition phase of object recognition process	6
Figure 3: Saddle surface with normal planes in directions of principal curvatures [25]	9
Figure 4: UVS volume on a face with different aspects of view [1].	11
Figure 5: “knot” object and pose number 0 range image with gray scale depth value	13
Figure 6: Two different pose for object “knot”	14
Figure 7: Features extracted from two pose for object “knot”	15
Figure 8: Two different pose for object “hub”	15
Figure 9: Features extracted from two pose for object “hub”	16
Figure 10: Two different pose for object “kroete”	16
Figure 11: Features extracted from two pose for object “kroete”	16
Figure 12: The feature group and values obtained from feature triplet.	19
Figure 13: Feature elimination according to feature types.....	21
Figure 14: Feature elimination regardless of feature types.....	21
Figure 15: Features with group of 3	22
Figure 16: Features with group of 4	22
Figure 17: Points of frames in coordinate system	27
Figure 18: New coordinate frame constructed (red) with three nonlinear points (blue).	27
Figure 19: Hash table constructed after preprocessing phase	28
Figure 20: Dividing vector space for Euclidian distance.....	30
Figure 21: Dividing vector space for city block distance	30
Figure 22: Dividing vector space for cosine distance.....	31
Figure 23: Geometric location of the matching points for Euclidian similarity	37
Figure 24: Geometric location of the matching points for city block similarity	38

Figure 25: Geometric location of the matching points for cosine similarity	39
Figure 26: Features extracted from two pose for object “kroete”. Left image from training and right image from testing data	40
Figure 27: Stuttgart database with 42 objects.	42
Figure 28: Training set for object “machine”.....	43
Figure 29: Selected 5 objects to test feature descriptors. (agfa, machine, igea, vette and pitbull)	44
Figure 30: Hinton diagram for base test for HK space	46
Figure 31: Rank histogram for base test for HK space.....	46
Figure 32: Hinton diagram for elimination test for HK space	47
Figure 33: Rank histogram for elimination test for HK space	48
Figure 34: Hinton diagram for feature group test for HK space	49
Figure 35: Rank histogram for feature group test for HK space	50
Figure 36: Hinton diagram for length values added test for HK space	51
Figure 37: Rank histogram for length values added test for HK space	51
Figure 38: Hinton diagram for individual max test for HK space.....	53
Figure 39: Rank histogram for individual max test for HK space	53
Figure 40: Angle Tests.....	55
Figure 41: Length Tests	56
Figure 42: Normal Angle Tests	58
Figure 43: Type Tests	59
Figure 44: General Max Tests	61
Figure 45: Hinton diagram for city block similarity test for HK space.....	62
Figure 46: Rank histogram for city block similarity test for HK space.....	63
Figure 47: Hinton diagram for cosine similarity test for HK space	64
Figure 48: Rank histogram for cosine similarity test for HK space	65
Figure 49: Hinton diagram for Euclidian similarity test.....	66
Figure 50: Hinton diagram for cosine similarity test.....	66
Figure 51: Computation Time Tests	68
Figure 52: Best Fitting Tests	70
Figure 53: Hinton Diagram for HK space, 25 objects	71
Figure 54: Rank Histogram for HK space, 25 objects.....	71
Figure 55: Hinton Diagram for HK space, 42 objects	72

Figure 56: Rank histogram for HK space, 42 objects.....	72
Figure 57: Primary screen of the program	80
Figure 58: Functionality of primary screen elements	81
Figure 59: Settings window.....	83
Figure 60: Functionality of settings window.....	83
Figure 61: Output files for tests.....	90
Figure 62: A sample file structure	92
Figure 63: Topology folder content	93

LIST OF TABLES

Table 1: Surface Shape Classification Based on HK Values.....	10
Table 2: Feature type representations.....	12
Table 3: Features descriptors obtained from pose 0 of “knot” object.....	13
Table 4: Some feature vectors obtained from pose 0 for object “Auto”.....	32
Table 5: Feature vectors obtained from different poses with type, angle and length values are used.....	33
Table 6: Angle Test Results.....	54
Table 7: Length Test Results.....	56
Table 8: Normal Angle Test Results.....	57
Table 9: Type Test Results.....	59
Table 10: General Max Test Results.....	60
Table 11: Computation Time Test Results.....	67
Table 12: Best Fitting Test Results.....	69
Table 13: Brief explanation for tests.....	73
Table 14: Definition of Source Code.....	86

LIST OF ABBREVIATIONS

HK	:	Mean and Gaussian Curvature
EGI	:	Extended Gaussian Image
SI	:	Shape Index
GMM	:	Gaussian Mixture Model
UVS	:	Vector space where u and v are surface dimension values and s is scale dimension.
2D	:	Two Dimensional
3D	:	Three Dimensional

CHAPTER 1

INTRODUCTION

1.1 Motivation

Human generally recognizes objects according to their appearance and shapes. It is very hard to design an artificial intelligence system which can recognize objects in a similar way as human predictions. The main problem is in complexity of human recognition. Human recognizes objects by observing various detailed properties of the objects. For example, two same types of objects can be recognized even when they have different color or different shape due to the aspect of view.

The most challenging problem for the object recognition applications is to recognize the objects when they are rotated, translated or scaled. Because, both the appearance and the shape of objects change with the change in the viewing direction.

About two decades ago, lots of studies were done with 3D data. However obtaining 3D data was very difficult. Thus, studies that have been done in the recent years about object recognition are mostly based on 2D techniques. In the light of the improvements in the accuracy of 3D scanning devices, nowadays, 3D recognition techniques are beginning to be used more commonly. With 3D scanners not only appearance but also surface shapes of the objects are acquired. In this thesis, 3D range data is used for object recognition.

The surface shapes of the objects change significantly with the change in the aspect of view. Therefore, different poses of the objects need to be learned so that a new pose can be interpolated by the learned poses. To achieve this aim, in this thesis

Stuttgart 3D database is used which includes 3D data of 42 objects. For each object there are 66 poses to be used for training and 258 poses to be used for testing.

Curvature values obtained from second derivatives of the 3D surfaces are used to obtain object descriptors which are then used for object representation. By down-sampling, pyramid of the data is constructed and multiscale curvature characteristics are obtained ([1]). These surface characteristics are used to represent the objects. So that training poses are stored in a database which is in a hash table format. Testing poses are retrieved from the hash table. Hash table is used because this method is robust to occlusions and also fast during testing.

After a series of experiments, object recognition rate is increased up to %90.97 for 42 objects and %92.85 for 30 objects. With the same database, in [27] they have achieved %93 rate where in [28] recognition rate is %98. The test result is very promising when compared to the works done in the literature for the same database.

1.2 Literature

Over 20 years, many studies have been performed using 3D range data. On the other hand; most effective studies have been done in the recent years, since before then, 3D scanning techniques were inadequate. In the light of the fact that about 20 years ago obtaining 3D data was too difficult, researchers tended to build their works mostly on appearance based approaches [2]. Within these years many of the approaches to this problem rely on the use of local features obtained from small patches of the object, keypoints of objects from 2D data have been used to identify objects [5, 13, 15].

Apart from 2D data, features extracted from 3D data are more descriptive. As in 2D, most of the recognition methods rely on extraction of local features from the objects in 3D. More effective ways have been proposed to retrieve objects before appearance based approaches. More distinctive feature extractions have been added to retrieve objects in 3D. With the local features obtained using some information, objects representations are obtained. Most of researches were done to represent objects by

using curvature estimation. By intersecting planes with surface, curvature values are obtained and with minimal and maximal curvature values surface characteristic is observed. Using minimal and maximal curvature values mean and Gaussian curvature (HK) is proposed in [3]. On the other hand in [11] again by using minimal and maximal curvatures shape index (SI) value is used to define surface characteristics.

Other than curvature estimation normal vectors are used to define surface. In [9], normal vectors around points are used as spin image approach. Although spin image is a 2D feature vector defined for each point on the 3D surface considering the point's neighbors also, it requires high computational load and is highly sensitive to data resolution. In [21], splashes are used as surface definitions for 3D object recognition. Splash method requires normal vector angle differences in a splash which is constructed by a circle around a specified point. Extended Gaussian image (EGI) is proposed in [8] which maps the surface normal vectors in a sphere and object representation is achieved. Although EGI is very successful in defining the global surface shape, with range image data it was hard to obtain full object shape. Also with normal vector approaches, transformation on the object fully changes the representation of an object.

The problem of 3D object recognition deals with matching the features obtained from the test object with the features obtained from the database objects. One of the most well known and common approaches in literature is by using graph-matching method ([7], [10], [14], [16], [18], [19]). In this method the matching is usually inexact and probabilistic ([17], [16], [24]). In [6] with extended Gaussian image, it is preferred to have direct matching.

The most important problem in object detection and recognition is the capability of predictions even if some transformations are applied to the objects. Most common brute-force matching methods have not capability of recognizing with some transformations. Other than brute-force matching, a geometric hashing technique was first proposed in [12] to reduce the amount of computation in such matching algorithms, since complexity is raised exponentially with common approaches. Also with geometric hashing method, transform and scale invariant matching achieved. In

geometric hashing, features obtained from training images are stored in the database with the form of hash table bins and matching of geometric features is done by indexing. In [12], it is proposed to use an orthogonal space representation by choosing base features points for given objects. With the new coordinate frame constructed, feature points are stored in hash table bins with location values as index.

In [23], the overall process is divided into two phases: preprocessing and recognition. At the preprocessing phase, for each model, features are extracted and they are stored to the hash table bins. This phase is computed offline and is not related with the recognition performance. By using new coordinate frames, features are indexed with location values. At recognition phase, testing model features are extracted and by reconstructing the coordinate frames hash table index values are obtained using the similar approach of preprocessing stage. By matching the hash table entry with the index, the “vote” mechanism is applied.

1.3 Scope of the Thesis

Object recognition process need to have following properties:

- Accurate results
- Low computation time
- Scale invariant
- Orientation invariant

Object recognition logic has been a challenge for years. Human being recognizes objects according to their shapes and appearance. For artificial object recognition these methods are used as base applications. As it has been mentioned in Section 1.2 many researchers generally used shape or appearance based methods. With the improvements in 3D modeling for robotics applications, recognition with shapes became more popular. This is because of the recognition process being applied with rich and meaningful information.

In this thesis, 3D surface feature detector proposed in [1] which extracts highly informative feature shapes in a compact way, is used. This method uses HK values and obtains a scale-space of the surface, making the technique resolution invariant.

Orientation and translation invariant matching is achieved by grouping extracted features. Scale invariance matching is obtained by using different similarity definitions. The aim of this thesis is to check the reliability of the feature descriptors and testing the methods to find which method best fits for the given features.

Although the recognition rate of the results has a significant importance, having less computational time with or without changing the rate of correctness is intended. A new geometric hashing method similar to the approach proposed in [12] is used that works with the features obtained from HK values. With the hashing method the online computation duration time decreases since data retrieval from database is faster by using hash table bins.

Whole recognition is separated into two parts: preprocessing and recognition phase. The block diagram for preprocessing phase of object recognition process is explained in Figure 1 and recognition phase of object recognition process is explained in Figure 2.

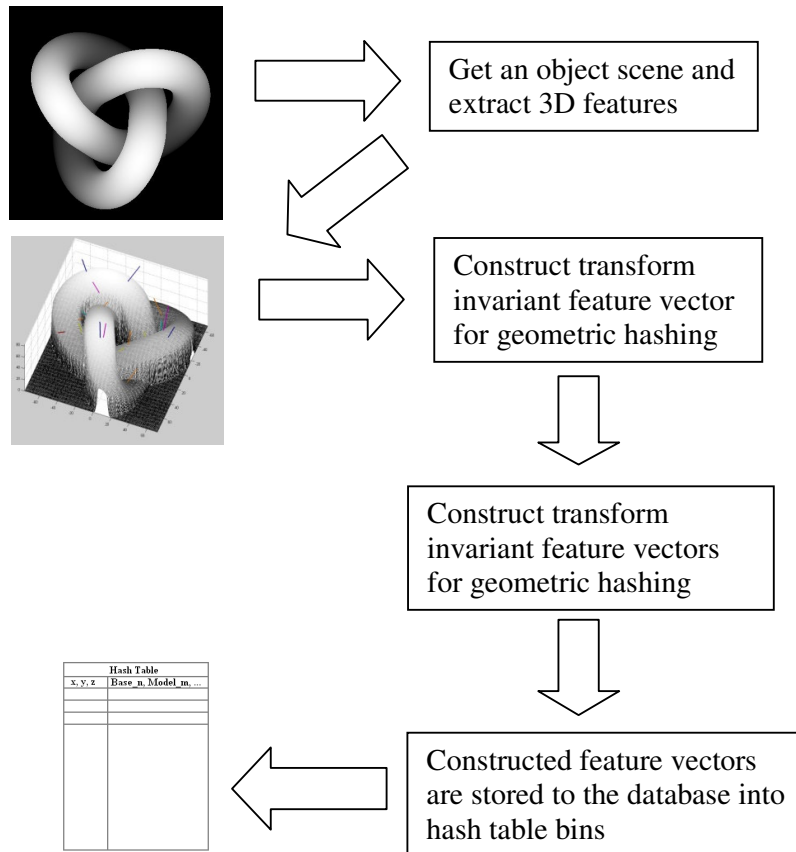


Figure 1: Preprocessing phase of object recognition process

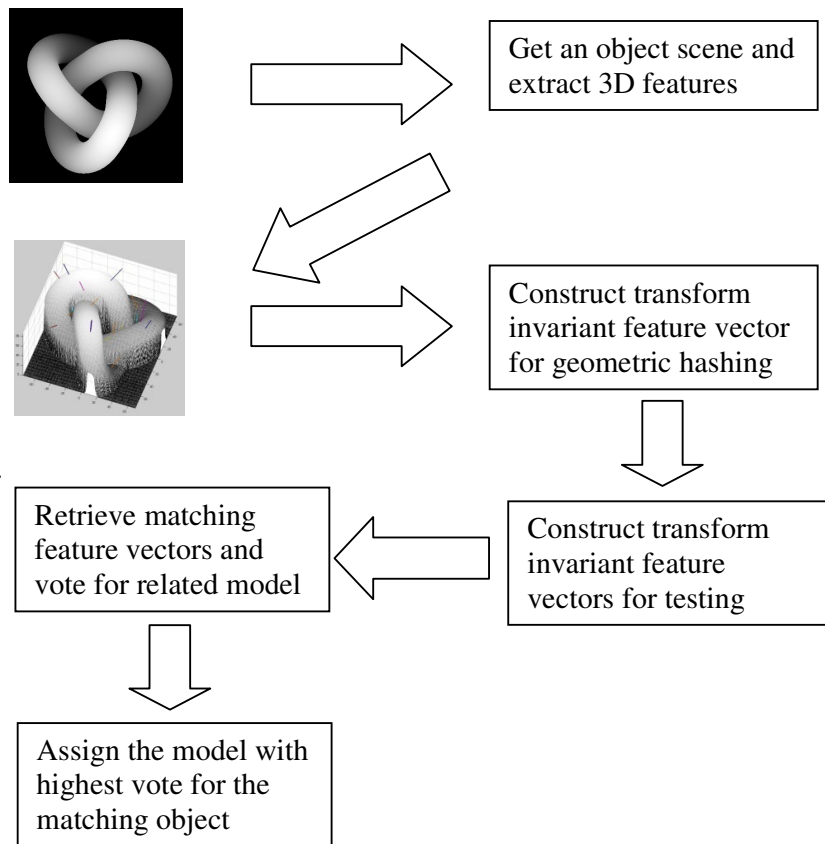


Figure 2: Recognition phase of object recognition process

1.4 Organization of the Thesis

In Chapter 2, transform invariant feature extraction method is explained.

In Chapter 3, transform invariant feature vector for geometric hashing is discussed.

In Chapter 4, indexing, hashing and geometric hashing methods are explained. Proposed matching method is discussed.

In Chapter 5, experiment plan is explained and results of the experiments are shown. Detailed analyses of the experiment results are done. With the result analyses obtained recognition rate is maximized.

Chapter 6 provides a conclusion of the thesis.

CHAPTER 2

TRANSFORM INVARIANT FEATURES

For a good object representation, feature extraction method needs to be transform and resolution invariant. In this thesis, 3D surface feature detector is used in [1] which extracts highly informative feature shapes in a compact way. This Chapter is to give information about extracting features of an object from the range image data. In this Chapter some background information will be given to understand the feature extraction method. As background information curvature, Gaussian and mean curvatures will be explained.

2.1 Curvature

Curvature is the amount of the change in the rotation of a curve. The formal definition of curvature is as equation (1).

$$K = \frac{d\Phi}{ds} \quad (1)$$

Where in this equation K represents curvature, Φ tangential angle and s arc length. Assuming with a 2D parametric equation $x = x(t)$ and $y = y(t)$, equation (1) can be expressed as in equation (2).

$$K = \frac{\frac{d\Phi}{dt}}{\frac{ds}{dt}} = \frac{\frac{d\Phi}{dt}}{\sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2}} = \frac{\frac{d\Phi}{dt}}{\sqrt{x'^2 + y'^2}} \quad (2)$$

To simplify the equation the definition of tangent angle in equation (3) can be used.

$$\tan \Phi = \frac{dy}{dx} = \frac{dy/dt}{dx/dt} = \frac{y'}{x'} \quad (3)$$

To obtain $\frac{d\Phi}{dt}$ term in equation (2), derivative of equation (3) can be taken according to t . This will yield the equation (3) to equations (4), (5), (6), (7) and (8).

$$\frac{d}{dx}(\tan \Phi) = \sec^2 \Phi \frac{d\Phi}{dt} = \frac{x'y'' - x''y'}{x'^2} \quad (4)$$

$$\frac{d\Phi}{dt} = \frac{1}{\sec^2 \Phi} \frac{x'y'' - x''y'}{x'^2} \quad (5)$$

$$\frac{d\Phi}{dt} = \frac{1}{1+\tan^2 \Phi} \frac{x'y'' - x''y'}{x'^2} \quad (6)$$

$$\frac{d\Phi}{dt} = \frac{1}{1+\left(\frac{y'}{x'}\right)^2} \frac{x'y'' - x''y'}{x'^2} \quad (7)$$

$$\frac{d\Phi}{dt} = \frac{x'y'' - x''y'}{x'^2 + y'^2} \quad (8)$$

Combining equations (2) and (8) the curvature value is obtained in equation (9).

$$K = \frac{x'y'' - x''y'}{(x'^2 + y'^2)^{3/2}} \quad (9)$$

In engineering and science applications with reasonable assumptions curvature value can be minimized as in equation (10).

$$K \approx \frac{d^2y}{dx^2} \quad (10)$$

2.2 Curves on Surfaces

In a 3D dimensional surface most common method to characterize surface is to check principal curvatures. To obtain principal curvatures, surfaces with normal vectors that are tangent to the curve are intersected with the curve. With the intersection curvature values are obtained. With all possible curvature values, the minimum and maximum valued curvatures are called principal curvatures. They are shown as k_1 and k_2 . In a saddle surface principal curvature surfaces are shown in Figure 3.

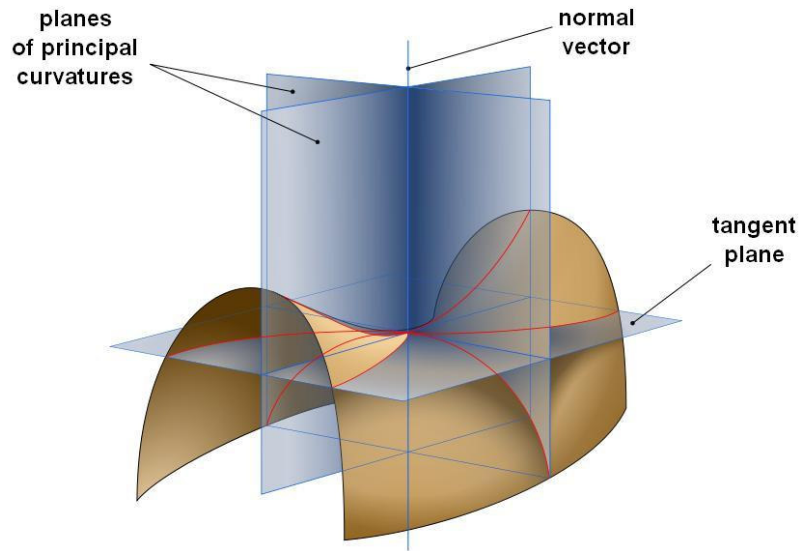


Figure 3: Saddle surface with normal planes in directions of principal curvatures [25]

2.3 Gaussian and Mean Curvatures

2.3.1 Gaussian Curvature (K)

Gaussian curvature (K) is the multiplication of principal curvature values ($k_1 \cdot k_2$). Positive Gaussian curvature value represents sphere where negative Gaussian curvature value represents hyperboloid.

2.3.2 Mean Curvature (H)

Mean curvature (H) is the sum of principal curvature values ($k_1 + k_2$). Positive mean curvature value represents upwards shape where negative mean curvature value represents downwards shape.

2.4 HK space

Scale and orientation invariant 3D features are extracted from the surface of the 3D object by using surface curvature values. This feature extraction method is proposed by Akagündüz and Ulusoy in 2007 [1] and will be summarized here.

In their original work, Besl and Jain [3] calculate mean (H) and Gaussian (K) curvatures and label each sample point on the surface by applying appropriate

thresholds. Then they classify these points according to their Gaussian and mean curvature values according to Table 1.

Table 1: Surface Shape Classification Based on HK Values.

	$K>0$	$K=0$	$K<0$
$H<0$	Peak	Ridge	Saddle Ridge
$H=0$	-	Plane	Minimal
$H>0$	Pit	Valley	Saddle Valley

For a given range image data HK shape classifications define a HK space. This space shows shape characteristics for given u and v surface dimension values.

Using second derivatives of the curves makes features obtained transform invariant since second derivative of a curve do not change with some orientation and translation

Contribution from Akagündüz and Ulusoy to this approach was scale space representation. This scale space representation is obtained by Gaussian pyramiding which is Gaussian filtering and down sampling the range image data.

After applying Gaussian pyramid to the data, surface characteristics are obtained. Since down sampling reduces the pixel numbers, down sampled data pixels are expanded to reflect precise data shape. Each Gaussian pyramid level is called with a scale value. Starting from zero with the original data, scale value “ s ” is incremented with each down sampling. So s^{th} level needs to be widened by 2^s . By using u and v surface dimension values and s scale dimension, UVS scale space is constructed. Note that because of down sampling higher level data will have less detail.

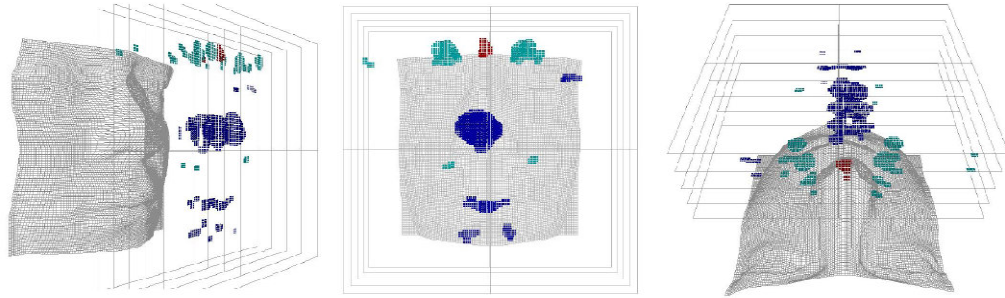


Figure 4: UVS volume on a face with different aspects of view [1].

In Figure 4, human face data is printed in a UVS space with surface characteristics obtained. In the figure peaks (blue), pits (cyan) and saddles (red) on the surface of the face are shown.

With the features obtained with UVS values components of features are connected into each other. After the connected components are estimated feature descriptors: type, location, scale, radius, volume and normal vector are obtained. Here are some explanations about feature descriptors:

1. Type: This value defines the type of the feature (peak, pit, saddle, etc.). In this these types are used as integer values from 1 to 9.
2. Location: These values define the center of mass of features obtained as x, y and z values for raw range data.
3. Scale: This value is the center of weighted scale for the features.
4. Radius: This value is the radius of the biggest area for features in scale levels.
5. Volume: This value is the sum of the areas for features in all scale levels.
6. Normal vector: These values define the normal vector of the object for the given feature location. Normal vector is shown as n_x , n_y and n_z .

2.5 Features Obtained

Feature type definitions used in this thesis are shown in Table 2.

Table 2: Feature type representations

Number	Feature Type
1	Peak
2	Ridge
3	Saddle Ridge
4	-
5	Plane
6	Minimal
7	Pit
8	Valley
9	Saddle Valley

An example of features obtained from “knot” object and pose number 0 (Figure 5) has been given in Table 3. The features shown in Table 3 are described by the values of type, x, y, z nx, ny, nz, scale, vol and radius. Where type shows the feature type (pit(1), peak(7), ridge(2), flat(5), saddle valley(8), saddle ridge(3), minimal(6), saddle valley(9)); x, y and z for feature location; nx, ny and nz for feature normal vector; scale for weighted mean scale value; vol for total area in scales and radius for biggest area radius.

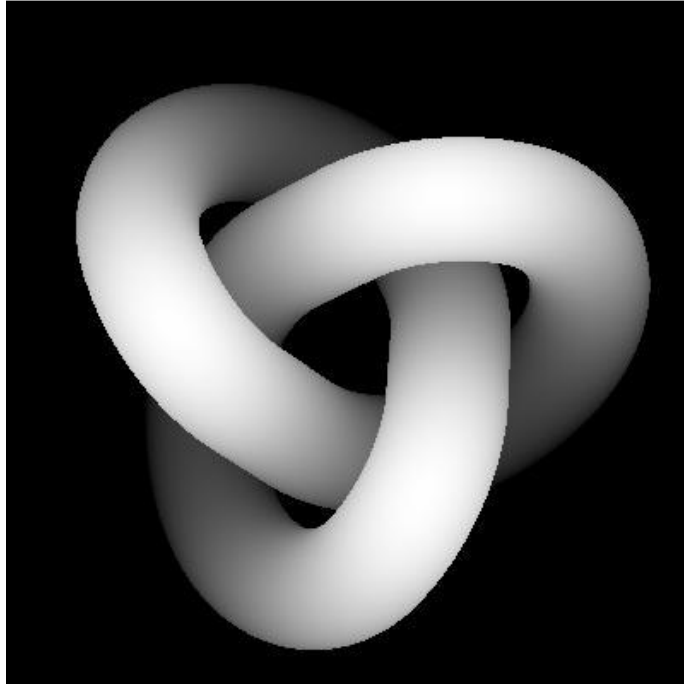


Figure 5: “knot” object and pose number 0 range image with gray scale depth value

Table 3: Features descriptors obtained from pose 0 of “knot” object

type	x	y	z	nx	ny	nz	scale	vol	radius
2	39,983262	15,119089	44,978337	-0,080465	0,218852	0,972435	2,393461	26671	47,536111
2	-6,731152	-41,965047	45,515628	0,255122	-0,0566	0,965251	2,348005	24810	46,287621
2	-34,415174	24,370489	42,352737	-0,196559	-0,365552	0,9098	2,079223	22140	42,822696
1	-20,591732	37,029953	54,742637	0,085754	0,185254	0,978942	2,183757	15194	35,900372
1	-23,087708	-30,666226	54,681214	-0,21478	0,107179	0,970764	1,756283	11341	34,71944
1	40,864808	-2,031336	55,00265	0,057503	-0,193205	0,979472	1,678629	10004	33,339738
6	26,875576	-44,841981	21,827857	0,202803	0,52778	0,824815	1,786746	7892	24,1813
6	-52,185143	-1,023589	21,084086	0,338276	-0,416531	0,843843	1,76779	7743	23,763033
6	26,699114	46,624007	20,953824	-0,560936	-0,175234	0,809101	1,659297	7056	23,466489
6	21,718942	14,443143	30,207035	0,041607	-0,409981	0,911145	2,510312	2085	16,545286
6	-22,802864	12,367844	31,621209	0,294832	0,063365	0,953446	2,762362	2083	18,669476
6	1,925072	-24,868136	30,3028	-0,354601	0,191343	0,91523	2,619	2021	17,260882
7	30,575061	33,628184	19,100493	-0,038344	0,433102	0,900529	3,711853	1527	18,889822
9	-1,976756	-30,088939	26,51655	0,549191	0,202358	0,810827	2,654054	1480	13,980067
9	27,39363	12,376899	25,88368	-0,527044	0,192754	0,82769	2,528622	1415	13,243442
9	-22,844856	17,27302	26,013181	0,135837	-0,598508	0,789517	2,667636	1375	13,877229
7	13,516798	-43,729368	20,073377	0,389756	0,177429	0,903664	3,739274	1212	17,103414
9	-45,493031	-3,013245	14,49752	-0,267205	0,321282	0,908504	2,643952	1174	13,195284
7	-43,202117	11,384242	20,678387	-0,18381	-0,428586	0,884606	3,783985	1074	16,679414
9	30,814901	-34,972106	14,929273	-0,058212	-0,295105	0,95369	2,533898	826	11,382097

As it has been mentioned feature extraction method is transform invariant. With a slightly different aspect of view the features extracted should not change. There may be some small change with some features since line of sight may be lost and new features may be visualized.

Figure 6 shows two close views for the same object. In Figure 7 their features are shown. Features are shown as colored lines in Figures. The length of the lines define the radius of the features obtained and way of the lines represent normal vector way. Color definitions are as following:

- Blue: Peak
- Purple: Ridge
- Red: Saddle Ridge
- Gray: Plane
- Orange: Minimal
- Cyan: Pit
- Green: Valley
- Yellow: Saddle Valley

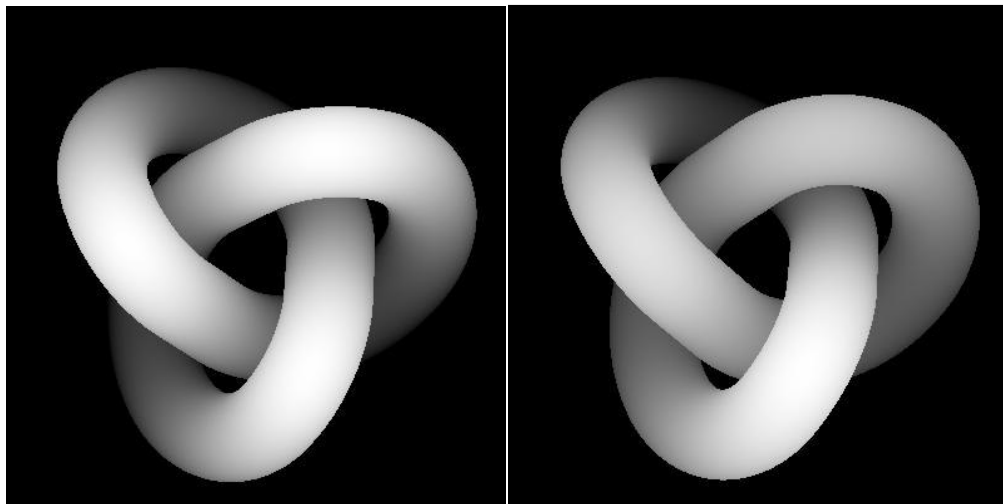


Figure 6: Two different pose for object “knot”

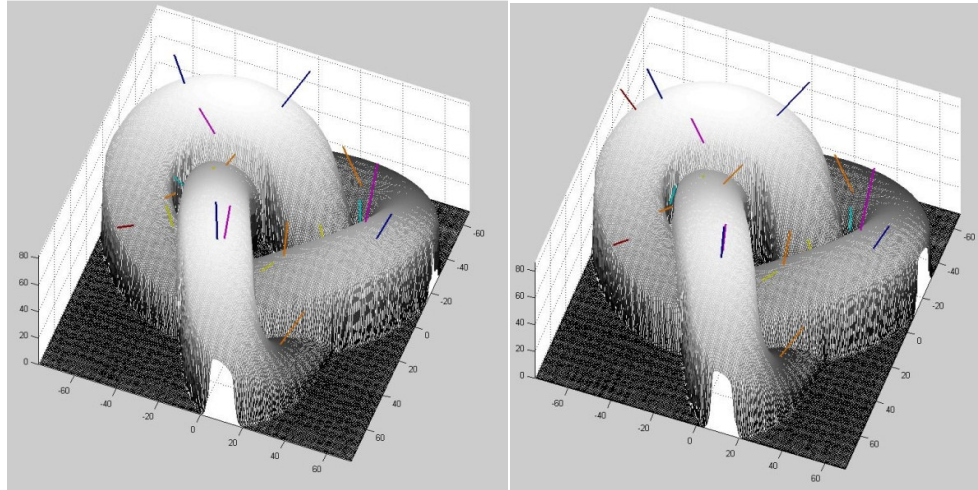


Figure 7: Features extracted from two pose for object “knot”

Figure 8 and Figure 9 show two different views for object “hub” and Figure 10 and Figure 11 show two different views for object “kroete”.

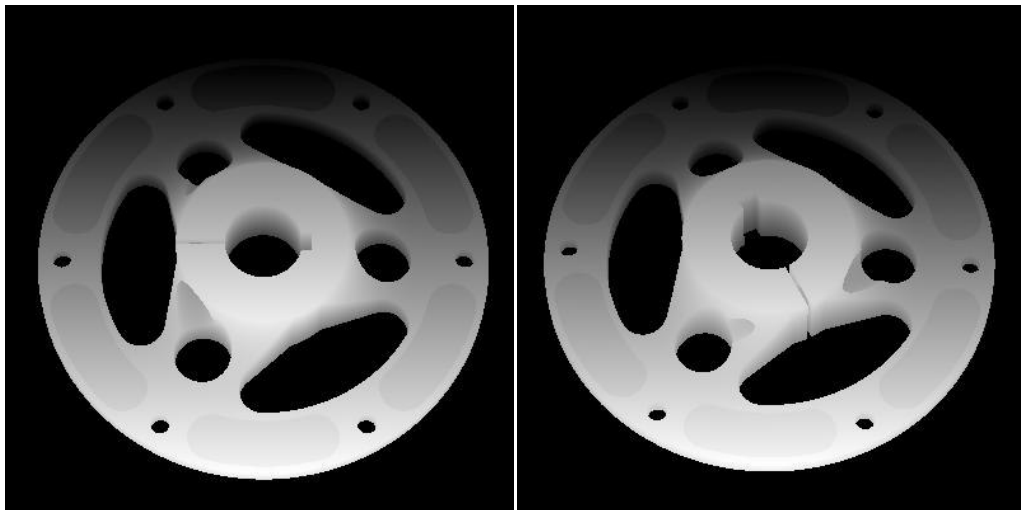


Figure 8: Two different pose for object “hub”

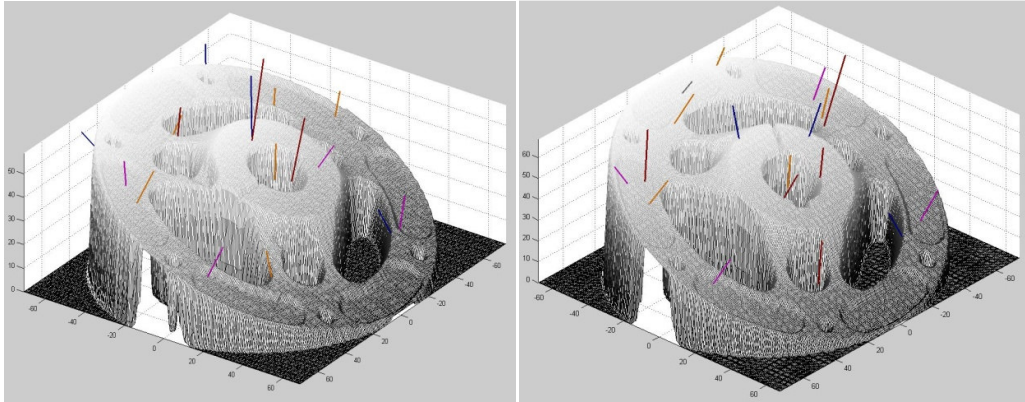


Figure 9: Features extracted from two pose for object “hub”

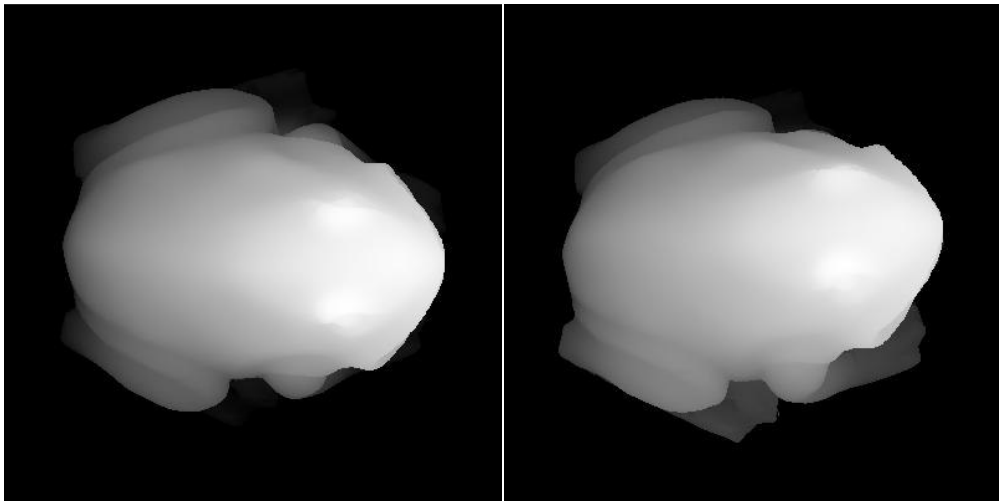


Figure 10: Two different pose for object “kroete”

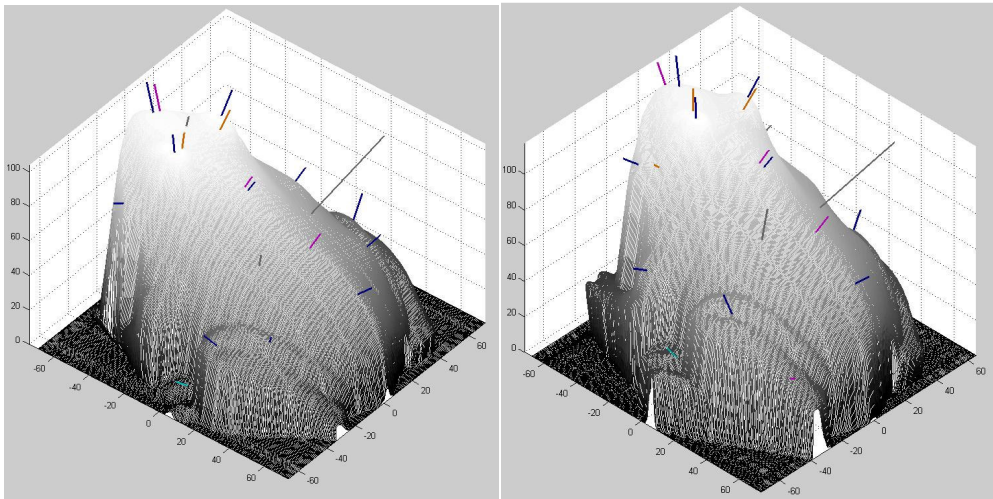


Figure 11: Features extracted from two pose for object “kroete”

With a closer look at the features obtained by two different views, it can be seen that new features can be formed with a small transformation. On the other hand, it can be seen that most of the feature locations and radius do not change with different views.

With feature descriptors in Table 3, feature vectors which will be used for hashing can be obtained. Construction of feature vectors is explained in Chapter 3.

CHAPTER 3

FEATURE VECTOR CONSTRUCTION FOR GEOMETRIC HASHING

In this chapter, feature vector construction for geometric hashing is explained. For an object recognition process, training should be done automated since the objects that will be trained are not predefined. For predefined objects, a method is proposed in [26] that uses faces as training objects.

In [26] Akagündüz and Ulusoy defined descriptive features for 3D object representations. Descriptive features for face are defined to represent left eye, right eye, nose and chin. So by identifying 2 pits, 1 peak and 1 saddle point and using these features' relationships with each other, they have used Gaussian mixture model (GMM) based graphical model to retrieve objects.

For a face left eye, right eye, nose and chin locations and scale represents a good retrieval. Note that these descriptive features are the biggest features for a face retrieval process. However while scanning, if a descriptive feature cannot be found then the retrieval process won't work due to lack of information for graphical model. Also with different aspects of views and due to occlusion these representative features will not be seen. If a descriptive feature is not seen it is impossible to retrieve the object. Also in [26], descriptive features are defined before the process, which is not an automated implementation. With a different training object descriptive features needs to be defined manually.

In this thesis a new method is proposed for representing the objects that are not predefined. In this method biggest features of the object are used. The definition of the biggest features is given later in this chapter. By applying feature grouping of these features, occlusion and transform invariant object representation is achieved.

3.1 Feature Grouping

Each feature element extracted from the surface, has the following attributes, the type (t_i) the positional center of mass (x_i), the scale (s_i), normal vector (n_i) and the volume (v_i). In addition mutual attributes between the features are defined such as distance between two features ($d_{i,j}$), angles between two features normal vectors ($na_{i,j}$) and the angle formed by three features in the 3D world ($\alpha_{i,j,k}$). Subsequently using the attributes defined above, for any three features on a surface an orientation invariant feature vector is constructed. To have transform invariant feature vectors, we need to remove coordinate values in computations. With a similar approach in [26] the easiest way to remove coordinate values from the feature vectors is to have triples of features and obtain angle between those three features. Also in order to improve test results, length between each features and angles between feature normal vectors obtained (Figure 12). Note that when object is transformed location of the features is also transformed, however angle between feature triples, length between features and angle between normal vectors for features do not change.

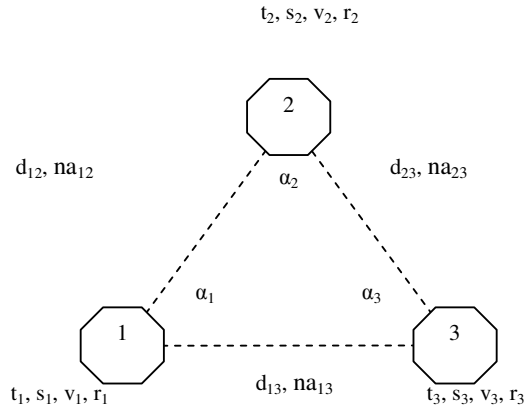


Figure 12: The feature group and values obtained from feature triplet.

In this method a feature vector is defined as the elements of this triplet: ($t_1, s_1, v_1, t_2, s_2, v_2, t_3, s_3, v_3, d_{1,2}, d_{2,3}, d_{1,3}, na_{1,2}, na_{2,3}, na_{1,3}, \alpha_1, \alpha_2, \alpha_3$). Here t_i, s_i, v_i are respectively the type, the scale and the volume of the corresponding node. The metric

distance between nodes i and j is defined as $d_{i,j}$. Observably the angle between i , $i+1$ and $i+2$ is given as α_i .

3.2 Optional feature vector values:

In this thesis, in order to limit the recognition time and in order to remove unnecessary detailed features, features are eliminated according to the volume or radius values. The higher values of volume or radius will have priority against lower ones. Elimination will be done by selecting the biggest “ f ” number of features. The elimination is logical since the bigger features classify the objects better than the smaller ones. By this elimination some small detailed changes will also be eliminated. Since with noise during the scanning process, some small features may appear. And by elimination, features resulted with noise will be removed.

Triple feature grouping method is the simplest one. However grouping with other numbers may give better results. Therefore in this thesis after the feature eliminations, different numbers of groupings are also tested.

In order to try feature descriptor characteristics, the values of feature vector should be optional. In other words, the method should be capable of determining which values will be used for the pose of the objects (such as length, angle, etc.). On the other hand selecting which types of features to use for testing and training the objects is very important since some feature types may not define good results for the objects. So the resulting feature vector approach will be as the following:

- Different number of features will be used.
- Different feature types will be used (pit, peak, ridge, flat, saddle, valley, saddle ridge, minimal, saddle valley).
- Different feature type elimination will be used.
- Elimination priority will be defined according to their volumes or scales.
- Different feature numbers in each group will be used.

According to these options defining steps of the feature vectors will be as following:

a) Choose the feature types to be used.

At this step feature types that will be used is selected. Other types of features will be ignored.

b) Choose the elimination type (Either a maximum number for features will be selected or numbers for each type will be selected)

At this step there are two options:

- Features will be eliminated in their type groups (Figure 13)
- Features will be eliminated regardless of their types (Figure 14)

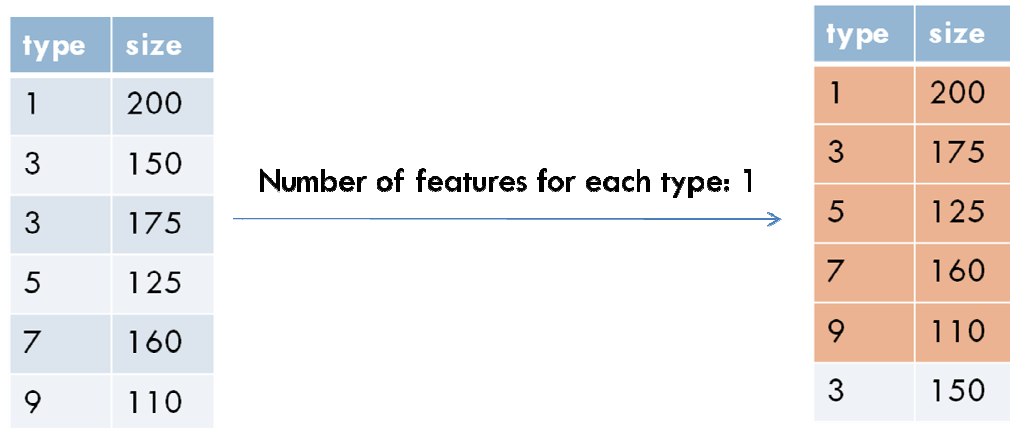


Figure 13: Feature elimination according to feature types

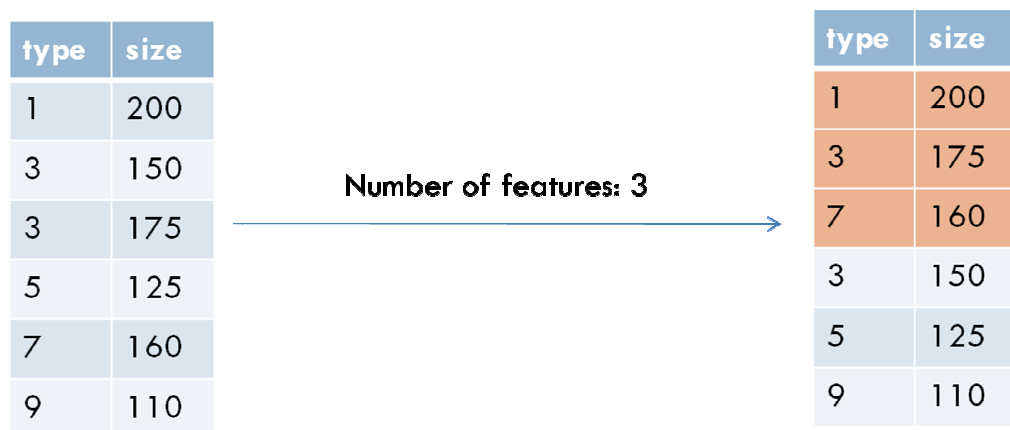


Figure 14: Feature elimination regardless of feature types

c) Choose the feature number in a group (f).

At this step feature number in a group will be defined. As an example with 3 features in a group is shown in Figure 15 while 4 features in a group is shown in Figure 16.

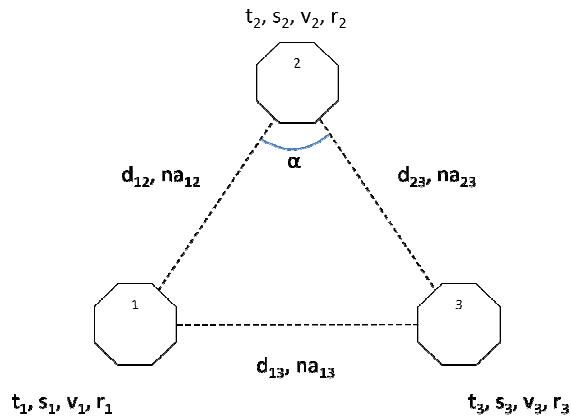


Figure 15: Features with group of 3

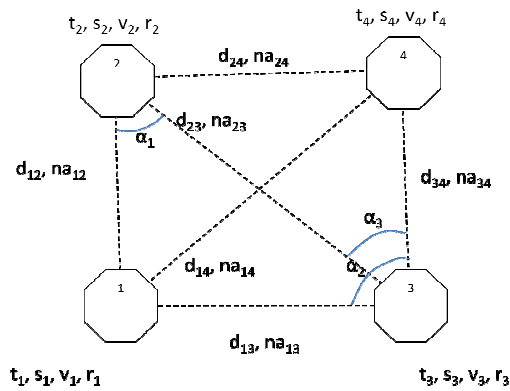


Figure 16: Features with group of 4

d) Choose which feature vector values are to be used.

At this step which feature vector values are to be used will be chosen. For example length, angle and normal angle can be used in test where in another test only length values can be used.

According to the feature groups we obtain the following feature vector:

- Types (f number of integers)
- Volumes (f number of integers) (optional)
- Radius (f number of real numbers) (optional)
- Scales (f number of real numbers) (optional)
- Angles (C(f,3) number of real numbers) (optional)
 - These values are angles between three nodes
 - In order to obtain the angles, first obtain each triple combination of f nodes in this group. Assume the group of nodes are (f=4) 1-2-4-5 with the numbers presenting node numbers. Then obtain all possible triplets: 1-2-4; 1-4-5; 2-4-5
 - For each triplet A-B-C calculate the dot product $V(AB) \cdot V(BC)$. This would give the cosine of the angle between |AB| and |BC|. To obtain the angle use inverse of cosine function.
 - Write each angle in the feature vector.
- Normal Angles (C(f,2) number of real numbers) (optional)
 - These values are angles between two nodes normal vectors
 - In order to obtain the angles between the normal vectors of each node, first obtain each double combination of f nodes in these groups. Assume the group of nodes are (f=4) 1-2-4-5 with the numbers presenting node numbers. Then obtain all possible doubles: 1-2; 1-4; 1-5; 2-4 ...
 - For each double A-B nodes calculate the dot product of the normal vectors. $[nx_A \ ny_A \ nz_A] \cdot [nx_B \ ny_B \ nz_B]$. This would give the cosine of the angle between the two normal vectors of the nodes A and B. To obtain angle use inverse of cosine function.
 - Write each angle in the feature vector.

- Distances ($C(f,2)$ number of real numbers) (optional)
 - These values are distances between two nodes
 - In order to obtain the vector of distances between each node, first obtain each double combination of f nodes in this group. Assume the group of nodes are ($f=4$) 1-2-4-5 with the numbers presenting node numbers. Then obtain all possible doubles: 1-2; 1-4; 1-5; 2-4 ...
 - For each double A-B, calculate the distance between nodes.
 - Get the first distance for this n group, which is $\text{dist}(1-2)$ for example. Divide all distances to this distance $\text{dist}(1-2)$ (in order to assure the scale invariance) (optional)
 - Write each distance in the feature vector.

In order to have similar feature vector presentation for acquired features there should be an order for features. The order of combinations is important. It should be the same each time combinations are calculated. To assure this, feature vectors are put into order considering the points mentioned below:

- Lower feature type number has higher priority and is put in front in the feature vector
- If two feature type numbers are same then the one with greater volume or radius (it is chosen by the user) has higher priority and is put in front in the feature vector

With the help of this order, similarity check between feature vectors can be accomplished more easily. Since, in case two or more types are same in a single group, incorrect similarity check with disoriented features may yield incorrect results.

The feature vector obtained will be transform invariant since none of the values change if the triplet rotates or translates.

CHAPTER 4

GEOMETRIC HASHING USING TRANSFORM INVARIANT FEATURE VECTORS FOR OBJECT RECOGNITION

In object recognition, not only recognition rate, but also recognition time and complexity are very important. Object retrieval can be extremely slow with the increase of objects trained by the system. Especially with standard database implementation during retrieval, redundant information in database is also checked. Key assignment as index for database items makes search algorithms run faster with the key values. Key values need to be unique and in relational database it is called primary key or unique key. Primary key is similar to the Social Security Number. Two people cannot have the same Social Security Number.

Although with primary key values, item retrieval from database gets faster, having simple primary keys do not help with the search algorithm. More explicatively, you cannot find a person without knowing his Social Security Number. In other words, it is essential to have descriptive primary key value.

To have descriptive primary key value hashing method is used.

4.1 Hashing

Hashing is used in many database applications. Hashing satisfies to generate unique key values for a database item. With the help of a hashing function unique keys are generated by using some specific inputs. These inputs consist of database item values. If the outputs from hash function are unique then only the related database item will be retrieved with regenerating primary key value.

Since hash values are unique, speed of insertion and deletion of elements will be constant with complexity $O(1)$.

4.2 Geometric Hashing

Geometric hashing is a transform invariant recognition technique. As understood from the name geometric hashing uses indexing with database implementation for object retrieval. Beside standard hashing method, in geometric hashing method, location values are used as index values.

It is essential to recognize objects even if they have undergone some geometric transformation. The main aim in geometric hashing is to construct an orthogonal coordinate frame with selecting bases from the features obtained and then localize the remaining features according to new coordinate system. Also with indexing according to the location values, retrieval speed of features increases.

Geometric hashing splits the whole recognition process into two phases: preprocessing and recognition.

4.2.1 Preprocessing

In preprocessing phase database is constructed. The operations in preprocessing phase are executed offline and do not affect the recognition time. In object recognition operations, this phase can be called as training, since different poses of objects are stored in the database for object retrieval.

For geometric hashing it is needed to have sparse feature point and feature characteristic inputs. Feature characteristic can be intensity for 2D applications. In this thesis there are many feature characteristics as type, angle, length etc.

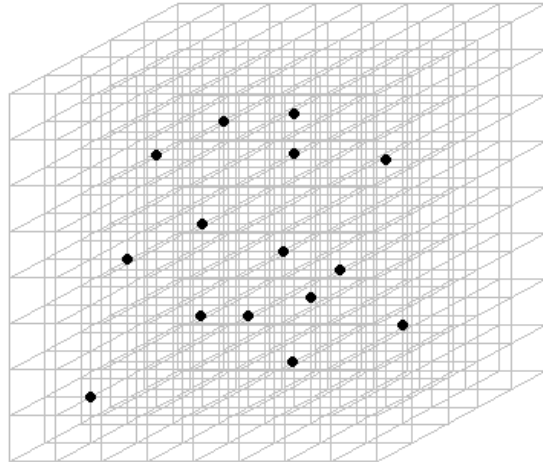


Figure 17: Points of frames in coordinate system

Figure 17 shows feature points obtained from an object in 3D. The aim of the geometric hashing is to construct new orthogonal coordinate system by selecting adequate number of base points. Number of base points in 2D should be 2 while in 3D it must be 3. So for 3D geometric hashing application 3 base points need to be selected. Figure 18 shows new coordinate frame constructed.

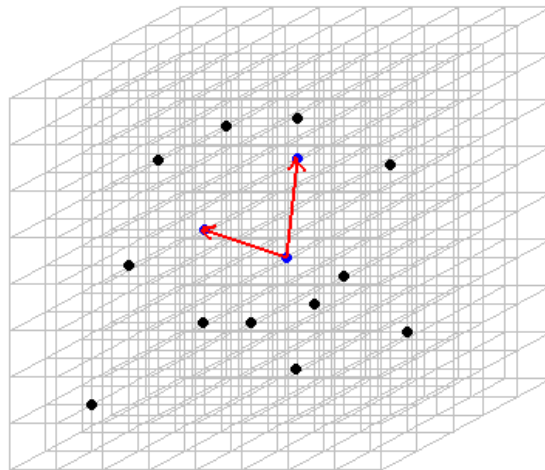


Figure 18: New coordinate frame constructed (red) with three nonlinear points (blue).

Not only one coordinate frame but also all possible coordinate frames are constructed. With the new coordinate frames other feature point representations are stored to the database. The items stored in the database is called hash table. So with “n” number of features obtained, combinations of (n, 3) possible coordinate frames and their representations of points are stored to the database.

With hashing method used by geometric hashing method it is way faster to retrieve feature points. In geometric hashing method location values are used as index values. Note that with many possibilities in location values, index values may not be unique. So in order to store two or more feature points with same index value, they are stored in the same hash table bin.

Hash Table	
x, y, z	Base <u>n</u> , Model <u>m</u> , ...

Figure 19: Hash table constructed after preprocessing phase

Figure 19 shows hash table constructed after preprocessing phase. Left column shows index as location values. Right column shows feature characteristics and points which are chosen as base points for coordinate frame.

4.2.2 Recognition

In recognition phase all operations are done online. So this phase determines the true object recognition process speed. In this phase hash tables are searched for the most similar model for the testing model.

In recognition phase there are some steps to find most similar model. These are:

1. Extract feature points.
2. Choose three base points for coordinate frame.
3. Represent points according to the new coordinate frame.
4. Access the hash table bin for the coordinates evaluated and cast a “vote” for the upcoming models.
5. Obtain a transformation matrix for the models which have “votes” higher than a threshold value and check if the transformation matrix fit with the original model.
6. If the verification at step 5 fails return step 2 with new base points.

Traditional geometric hashing method generally fits with transformed features. However if feature locations differ slightly with the original model, location index values will not result with correct results since the index values do not match. Also with feature grouping, feature vectors are transform invariant already. Therefore, a new hashing method is proposed to qualify fast and correct recognition.

4.3 Hashing Method Proposed

The proposed hashing method is similar to geometric hashing. In this method no coordinate frame is constructed since in this thesis transform invariant feature vector is used. Instead of related base feature points, feature vectors are stored directly in database. And since feature vectors may slightly differ from the original vectors, indexing achieved with dividing the feature vector space into intervals.

With a 2D representation dividing vector space is shown as in Figure 20. Note that dividing is achieved with constant values. These values are assigned according to the threshold values for similarity which will be explained later. Vector space division shown in Figure 20 ensures that for any two feature vectors in the same interval any feature value difference is less than the threshold value assigned. So this ensures that Euclidean distance between two feature vectors is below a threshold value.

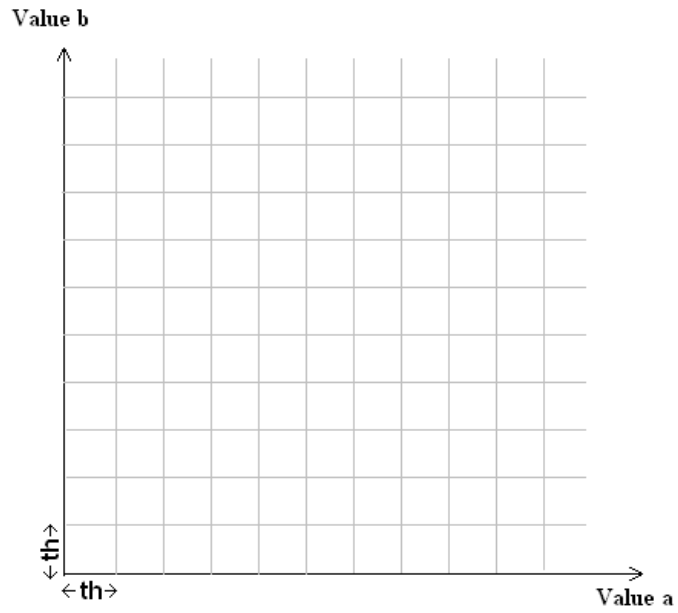


Figure 20: Dividing vector space for Euclidian distance

In Figure 21 a similar vector space division is shown. With this division sum of absolute of the vector value difference will be below a threshold value.

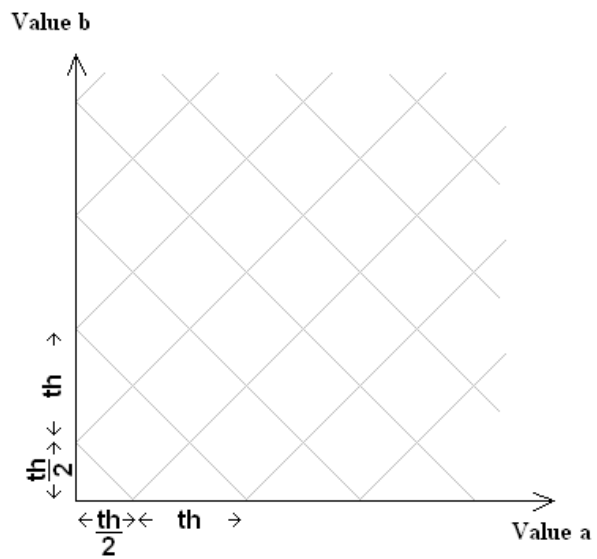


Figure 21: Dividing vector space for city block distance

Different approach for dividing vector space is shown in Figure 22. In this approach the angle between two feature vector and the origin point will be below a threshold value.

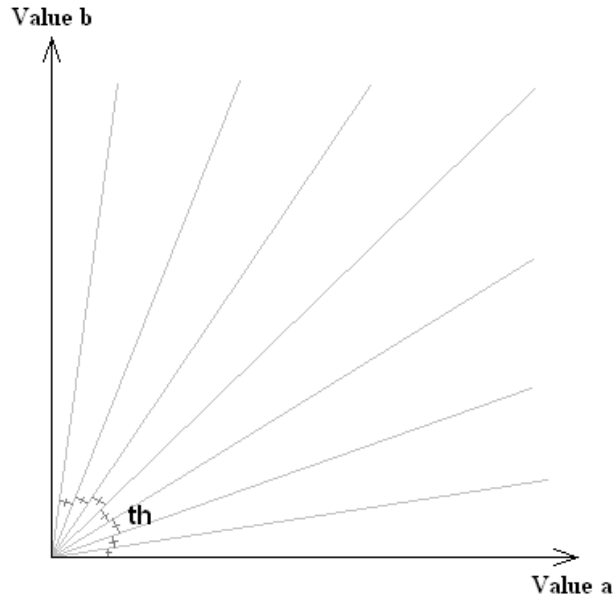


Figure 22: Dividing vector space for cosine distance

For the hashing method, it is essential to use a database to keep records of the objects. Comparing the computation time with the literature is not a scope of this thesis. However, the computation time for the proposed method needs to be compared with the brute-force matching. Therefore database application does not affect the test results. In this thesis, MySQL database is used since it is free and commonly used.

The feature vectors are written to different database tables according to their indexes because index values are used as primary keys and primary keys are unique and also more than one feature vectors need to have same index value for this method. So that, relevant database table can be accessed by using the obtained index. In other words, for each interval, there is a database table with a unique index.

After the features are obtained, feature vectors are created and written to the hash table. Table 4 shows feature vectors obtained from object “Auto” pose number 0.

Table 4: Some feature vectors obtained from pose 0 for object “Auto”

id	pose	type 1	type 2	type 3	scale 1	scale 2	scale 3	angle	length 1	length 2
1	Auto.oogl_scale150000.hk	1	1	1	2,37	2,718	2,697	27	86,657	78,48
2	Auto.oogl_scale150000.hk	1	1	1	2,37	2,718	1,847	132	86,657	64,49
3	Auto.oogl_scale150000.hk	1	1	1	2,37	2,697	1,847	151	78,48	64,49
4	Auto.oogl_scale150000.hk	1	1	1	2,718	2,697	1,847	82	39,264	138,1
5	Auto.oogl_scale150000.hk	1	1	5	2,37	2,718	-1	20	86,657	28,2
6	Auto.oogl_scale150000.hk	1	1	5	2,37	2,718	-1	125	86,657	46,33
7	Auto.oogl_scale150000.hk	1	1	5	2,37	2,718	-1	9,2	86,657	59,8
8	Auto.oogl_scale150000.hk	1	1	5	2,37	2,718	-1	103	86,657	24,52
9	Auto.oogl_scale150000.hk	1	1	5	2,37	2,718	-1	42	86,657	46,3
10	Auto.oogl_scale150000.hk	1	1	5	2,37	2,718	-1	12	86,657	74,73

length 3	volume 1	volume 2	volume 3	radius 1	radius 2	radius 3	normangle 1	normangle 2	normangle 3
39,264	1036	1013	945	11,041	10,645	10,464	73,31936	43,13471	53,598568
138,14	1036	1013	943	11,041	10,645	10,749	73,31936	37,857116	73,770096
138,49	1036	945	943	11,041	10,464	10,749	43,13471	37,857116	74,944918
138,49	1013	945	943	10,645	10,464	10,749	53,598568	73,770096	74,944918
60,867	1036	1013	22575	11,041	10,645	37,91	73,31936	33,140421	42,799499
119,47	1036	1013	14360	11,041	10,645	30,236	73,31936	30,506293	50,889793
29,253	1036	1013	11735	11,041	10,645	27,333	73,31936	45,715739	29,074878
95,073	1036	1013	9950	11,041	10,645	25,168	73,31936	35,016706	68,774173
60,733	1036	1013	4114	11,041	10,645	16,205	73,31936	95,137668	44,262431
20,188	1036	1013	1785	11,041	10,645	10,66	73,31936	37,051289	36,491005

The values in Table 4 are as follows: id, pose, type, scale, angle, length, volume, radius and normal angle. Note that for three features in a group there are 3 type values, 3 scale values, 1 angle value, 3 length values, 3 volume values, 3 radius values and 3 normal angle values in these columns.

As it has been explained in Chapter 3, feature descriptors are optional and may not be used during testing and training. If some feature descriptors are not used (if only angle and length values are used), the feature vectors shown in Table 4 will change as the feature vectors in Table 5. This table will only include type, angle and length value columns inside.

Table 5: Feature vectors obtained from different poses with type, angle and length values are used

id	pose	type 1	type 2	type 3	angle	length 1	length 2	length 3
1	Auto.oogl_scale150000.hk	1	1	1	26,928425	86,656911	78,480247	39,263865
2	Auto.oogl_scale150000.hk	1	1	1	131,552495	86,656911	64,491691	138,139672
3	Auto.oogl_scale150000.hk	1	1	1	151,07512	78,480247	64,491691	138,485454
4	Auto.oogl_scale150000.hk	1	1	1	82,339735	39,263865	138,139672	138,485454
5	Auto.oogl_scale150000.hk	1	1	5	19,7424	86,656911	28,197733	60,866515
6	Auto.oogl_scale150000.hk	1	1	5	125,089254	86,656911	46,333934	119,467537
7	Auto.oogl_scale150000.hk	1	1	5	9,243504	86,656911	59,802606	29,253077
8	Auto.oogl_scale150000.hk	1	1	5	102,61987	86,656911	24,518081	95,073422
9	Auto.oogl_scale150000.hk	1	1	5	41,98499	86,656911	46,296807	60,733071
10	Auto.oogl_scale150000.hk	1	1	5	11,617645	86,656911	74,731135	20,188286

Extraction of feature vectors is applied to both training and testing objects. Training object feature extraction is made offline during the preprocessing stage. In online stage in other words in recognition stage, all feature vectors are extracted for testing object, then test objects are tested with training objects.

Partial matching is used in testing. By using partial matching, all objects are compared one by one. All feature vectors obtained from a test object is compared with all feature vectors of the training objects. If two feature vectors are similar then the related training object receives a “vote”. After all feature vectors of the test object are compared with all feature vectors of the training objects, the highest vote defines the most similar object. Similarity of two feature vectors is defined in three ways. These are:

- Euclidean similarity
- City block similarity
- Cosine similarity

4.3.1 Euclidean Similarity

In Euclidean similarity of two feature vectors is defined as:

- Types exactly match
- The Euclidean distance between volume vectors is below a threshold (thr_vol).
- The Euclidean distance between radius vectors is below a threshold (thr_rad).

- The Euclidean distance between scale vectors is below a threshold (thr_scl).
- The Euclidean distance between angle vectors is below a threshold (thr_ang).
- The Euclidean distance between normal angle vectors is below a threshold (thr_nang).
- The Euclidean distance between length vectors is below a threshold (thr_dist).

For example, consider two feature vectors as f_1 and f_2 are $\{t_{11}, s_{11}, v_{11}, r_{11}, t_{21}, s_{21}, v_{21}, r_{21}, t_{31}, s_{31}, v_{31}, r_{31}, d_{1,21}, d_{2,31}, d_{1,31}, na_{1,21}, na_{2,31}, na_{1,31}, \alpha_1\}$ and $\{t_{12}, s_{12}, v_{12}, r_{12}, t_{22}, s_{22}, v_{22}, r_{22}, t_{32}, s_{32}, v_{32}, r_{32}, d_{1,22}, d_{2,32}, d_{1,32}, na_{1,22}, na_{2,32}, na_{1,32}, \alpha_2\}$ relatively. If following conditions are satisfied than these two feature vectors are similar.

- $t_{11} = t_{12}$ and $t_{21} = t_{22}$ and $t_{31} = t_{32}$
- $\sqrt{(s_{11} - s_{12})^2 + (s_{21} - s_{22})^2 + (s_{31} - s_{32})^2} < thr_scl$
- $\sqrt{(v_{11} - v_{12})^2 + (v_{21} - v_{22})^2 + (v_{31} - v_{32})^2} < thr_vol$
- $\sqrt{(r_{11} - r_{12})^2 + (r_{21} - r_{22})^2 + (r_{31} - r_{32})^2} < thr_rad$
- $\sqrt{(d_{11} - d_{12})^2 + (d_{21} - d_{22})^2 + (d_{31} - d_{32})^2} < thr_dist$
- $\sqrt{(na_{11} - na_{12})^2 + (na_{21} - na_{22})^2 + (na_{31} - na_{32})^2} < thr_nang$
- $|\alpha_1 - \alpha_2| < thr_ang$

4.3.2 City Block Similarity

In city block similarity of two feature vectors is defined as:

- Types exactly match
- The sum of absolute difference between volume values is below a threshold (thr_vol).
- The sum of absolute difference between radius values is below a threshold (thr_rad).
- The sum of absolute difference between scale values is below a threshold (thr_scl).

- The sum of absolute difference between angle values is below a threshold (thr_ang).
- The sum of absolute difference between normal angle values is below a threshold (thr_nang).
- The sum of absolute difference between length values is below a threshold (thr_dist).

For example, consider two feature vectors as f_1 and f_2 are $\{t_{11}, s_{11}, v_{11}, r_{11}, t_{21}, s_{21}, v_{21}, r_{21}, t_{31}, s_{31}, v_{31}, r_{31}, d_{1,21}, d_{2,31}, d_{1,31}, na_{1,21}, na_{2,31}, na_{1,31}, \alpha_1\}$ and $\{t_{12}, s_{12}, v_{12}, r_{12}, t_{22}, s_{22}, v_{22}, r_{22}, t_{32}, s_{32}, v_{32}, r_{32}, d_{1,22}, d_{2,32}, d_{1,32}, na_{1,22}, na_{2,32}, na_{1,32}, \alpha_2\}$ relatively. If following conditions are satisfied than these two feature vectors are similar.

- $t_{11} = t_{12}$ and $t_{21} = t_{22}$ and $t_{31} = t_{32}$
- $|s_{11} - s_{12}| + |s_{21} - s_{22}| + |s_{31} - s_{32}| < thr_scl$
- $|v_{11} - v_{12}| + |v_{21} - v_{22}| + |v_{31} - v_{32}| < thr_vol$
- $|r_{11} - r_{12}| + |r_{21} - r_{22}| + |r_{31} - r_{32}| < thr_rad$
- $|d_{11} - d_{12}| + |d_{21} - d_{22}| + |d_{31} - d_{32}| < thr_dist$
- $|na_{11} - na_{12}| + |na_{21} - na_{22}| + |na_{31} - na_{32}| < thr_nang$
- $|\alpha_1 - \alpha_2| < thr_ang$

4.3.3 Cosine Similarity

In cosine similarity of two feature vectors is defined as:

- Types exactly match
- The angle between two volume vectors is below a threshold (thr_vol).
- The angle between radius vectors is below a threshold (thr_rad).
- The angle between scale vectors is below a threshold (thr_scl).
- The angle between angle vectors is below a threshold (thr_ang).
- The angle between normal angle vectors is below a threshold (thr_nang).
- The angle between length vectors is below a threshold (thr_dist).

For example, consider two feature vectors as f_1 and f_2 are $\{t_{11}, s_{11}, v_{11}, r_{11}, t_{21}, s_{21}, v_{21}, r_{21}, t_{31}, s_{31}, v_{31}, r_{31}, d_{1,21}, d_{2,31}, d_{1,31}, na_{1,21}, na_{2,31}, na_{1,31}, \alpha_1\}$ and $\{t_{12}, s_{12}, v_{12}, r_{12}, t_{22}, s_{22}, v_{22}, r_{22}, t_{32}, s_{32}, v_{32}, r_{32}, d_{1,22}, d_{2,32}, d_{1,32}, na_{1,22}, na_{2,32}, na_{1,32}, \alpha_2\}$ relatively. To achieve angle between two vectors, inverse cosine of their dot product divided by their length must be calculated. If following conditions are satisfied than these two feature vectors are similar.

- $t_{11} = t_{12}$ and $t_{21} = t_{22}$ and $t_{31} = t_{32}$
- $\cos^{-1} \left(\frac{(s_{11} * s_{12}) + (s_{21} * s_{22}) + (s_{31} * s_{32})}{\sqrt{(s_{11})^2 + (s_{21})^2 + (s_{31})^2} * \sqrt{(s_{12})^2 + (s_{22})^2 + (s_{32})^2}} \right) < thr_scl$
- $\cos^{-1} \left(\frac{(v_{11} * v_{12}) + (v_{21} * v_{22}) + (v_{31} * v_{32})}{\sqrt{(v_{11})^2 + (v_{21})^2 + (v_{31})^2} * \sqrt{(v_{12})^2 + (v_{22})^2 + (v_{32})^2}} \right) < thr_vol$
- $\cos^{-1} \left(\frac{(r_{11} * r_{12}) + (r_{21} * r_{22}) + (r_{31} * r_{32})}{\sqrt{(s_{11})^2 + (s_{21})^2 + (s_{31})^2} * \sqrt{(s_{12})^2 + (s_{22})^2 + (s_{32})^2}} \right) < thr_rad$
- $\cos^{-1} \left(\frac{(d_{11} * d_{12}) + (d_{21} * d_{22}) + (d_{31} * d_{32})}{\sqrt{(d_{11})^2 + (d_{21})^2 + (d_{31})^2} * \sqrt{(d_{12})^2 + (d_{22})^2 + (d_{32})^2}} \right) < thr_dist$
- $\cos^{-1} \left(\frac{(na_{11} * na_{12}) + (na_{21} * na_{22}) + (na_{31} * na_{32})}{\sqrt{(na_{11})^2 + (na_{21})^2 + (na_{31})^2} * \sqrt{(na_{12})^2 + (na_{22})^2 + (na_{32})^2}} \right) < thr_nang$
- $\cos^{-1} \left(\frac{\alpha_{11} * \alpha_{12}}{\alpha_{11} * \alpha_{12}} \right) = 0 < thr_ang$

Since it is unnecessary to check angle in one dimensional (angle will result zero), in this thesis similarity check may be adjustable for each value type. For example, while applying cosine similarity for length vectors; Euclidian similarity can be applied for normal angle vectors.

Cosine similarity makes checking similarity scale invariant since angle is not dependant to the scale. So using cosine similarity for scale, volume, radius and length vectors makes the object recognition system scale invariant.

Note that some values may not be used during feature vector extraction. If a value is not being used during feature extraction then no threshold will be applied for that value type.

4.4 Geometric Meaning of the Similarities

As explained earlier, hashing with vector space division is used in this thesis. Hashing is achieved according to the similarity check and threshold values. With this type of hashing only relevant information is retrieved from database.

The logic of dividing the vector space is simple. Assume a testing vector with 2D representation $v = (v_a, v_b)$ (coordinate axis as “a” and “b” (as example “a” and “b” can be length values as “length_1” and “length_2”)) and assume that vector $v_d = (v_{da}, v_{db})$ is a vector in database, then the equation of feature vector matching with Euclidian similarity will be as in equation (11).

$$\sqrt{(v_a - v_{da})^2 + (v_b - v_{db})^2} < th \quad (11)$$

According to the equation (11) geometric location of the matching vectors in the database will be inside a circle having a center (v_a, v_b) and radius th . Geometric location of the matching points for Euclidian similarity is given in Figure 23.

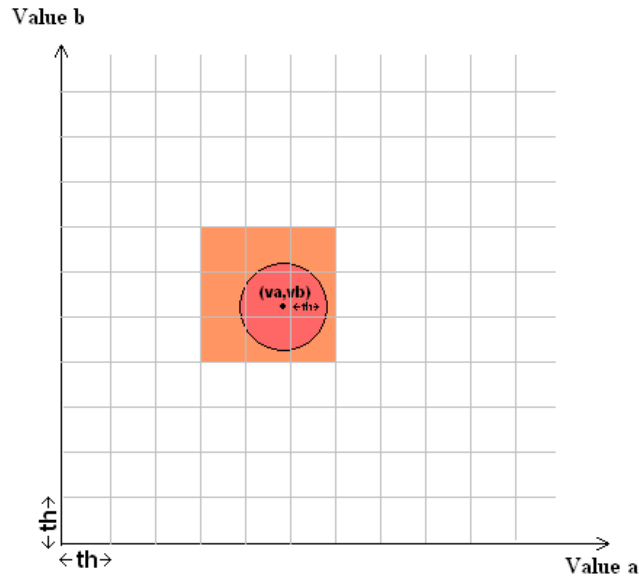


Figure 23: Geometric location of the matching points for Euclidian similarity

In Figure 23 the area shown with red color is the geometric location of matching points. Note that matching point can be in any neighbor interval. Orange colored area shows regions that need to be compared with the testing feature vector.

For city block similarity, matching equation will be as in equation (12).

$$|v_a - v_{da}| + |v_b - v_{db}| < th \quad (12)$$

Equation (12) yields to equations (13), (14), (15) and (16) with all possible signs inside absolute values.

$$v_a - v_{da} + v_b - v_{db} < th \text{ when } v_a > v_{da} \text{ and } v_b > v_{db} \quad (13)$$

$$-v_a + v_{da} + v_b - v_{db} < th \text{ when } v_a < v_{da} \text{ and } v_b > v_{db} \quad (14)$$

$$v_a - v_{da} - v_b + v_{db} < th \text{ when } v_a > v_{da} \text{ and } v_b < v_{db} \quad (15)$$

$$-v_a + v_{da} - v_b + v_{db} < th \text{ when } v_a < v_{da} \text{ and } v_b < v_{db} \quad (16)$$

According to equations (13), (14), (15) and (16), geometric location of matching vector points is nothing but a square rotated with angle of 45° . Geometric location of the matching points for city block similarity is given in Figure 24.

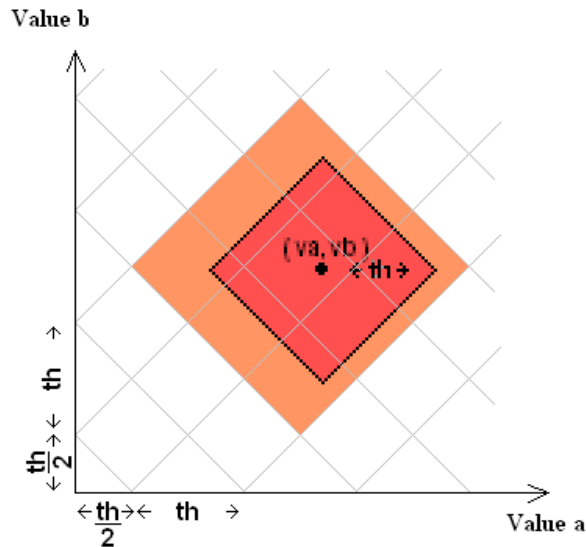


Figure 24: Geometric location of the matching points for city block similarity

In Figure 24 the area shown with red color is the geometric location of matching points. Note that matching point can be in any neighbor interval. Orange colored area shows regions that need to be compared with the testing feature vector.

For cosine similarity, from the definition, matching points, origin point and the vector itself should make an angle smaller than th value. Geometric location of the matching points for cosine similarity is given in Figure 25

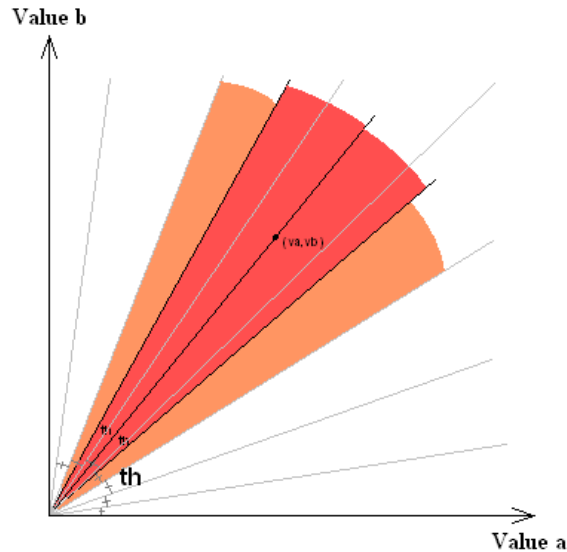


Figure 25: Geometric location of the matching points for cosine similarity

In Figure 25, the area shown with red color is the geometric location of matching points. Note that matching point can be in any neighbor interval. Orange colored area shows regions that need to be compared with the testing feature vector.

Note that for each similarity method, the feature vectors in neighbor regions can match with the testing feature vector. Since computation is required for queries to access database; in order to minimize query count, neighbor tables are not checked. However; feature vectors for training poses are written to the tables of neighbor regions. Since database construction is offline, this process does not affect the computation time.

4.5 Reason for Indexing by Hashing

Indexing makes the computations faster. For the complexity of this algorithm without indexing, when “f” is the feature vector number, searching for matching type complexity will be $O(\log(f))$. Since all testing feature vectors need to be compared, complexity will be $O(f \cdot \log(f))$ to compare two objects. “m” being training objects number and “n” being testing objects number the whole process complexity will be $O(m \cdot n \cdot f \cdot \log(f))$. While with indexing complexity is $O(m \cdot n \cdot f)$ since features of training objects are stored in intervals and type values are also used as index values.

The reason for using partial matching is the recognition will be resistant to occlusion and some outliers. Also with 365 degree recognition, different views for objects need to be trained separately. A probabilistic graphical model cannot be trained with low training data since there are many differences between two training poses with about 25° change.

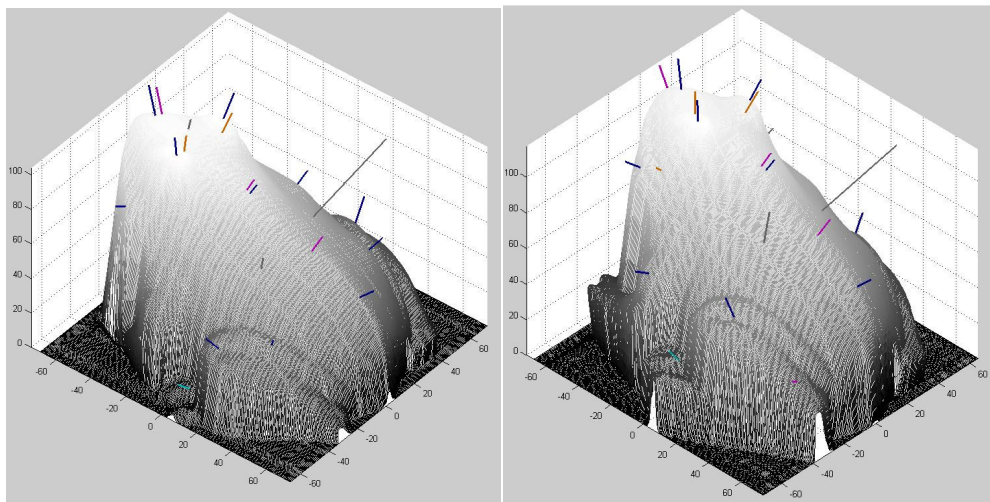


Figure 26: Features extracted from two pose for object “kroete”. Left image from training and right image from testing data

In Figure 26, 20 features for two poses (one of them for testing and one of them for training) are shown. By checking the features, 16 of 20 feature locations related to each other are nearly same. With feature grouping it is expected to have $C(16, 3) = 560$ similar features. If 10 features are used and assuming %70 feature matching, a

combination of $C(7, 3) = 35$ “vote” will be achieved and which will yield a correct recognition. Even if features are extracted incorrect with the result of some occlusion, the resultant similarity “vote” will be sufficient for a correct recognition.

CHAPTER 5

EXPERIMENTS AND RESULTS

In this thesis Stuttgart University Range Image Database [22] is used. In Stuttgart database there are 42 objects. There are 66 training poses and 258 testing poses for each object. This yields to 2772 ($66 \cdot 42$) training poses and 10836 ($258 \cdot 42$) testing poses. Figure 27 shows the 42 objects in Stuttgart database.

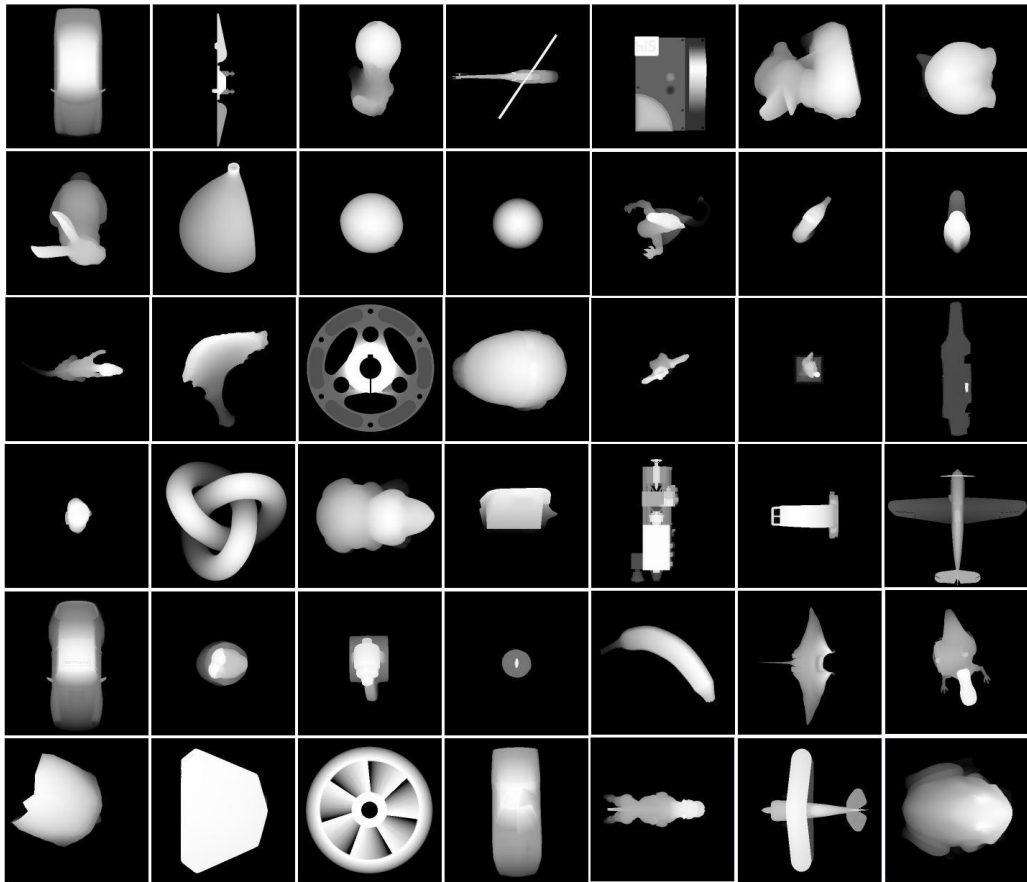


Figure 27: Stuttgart database with 42 objects.

66 training poses from each of the 42 objects are used. These poses are distributed evenly over the whole viewing sphere with angles of $23\text{-}26^\circ$ between viewpoints. The system is tested with 258 poses at $11.5\text{-}13^\circ$ angle shifted viewpoints. A training set for “machine” object is given as example in Figure 28.

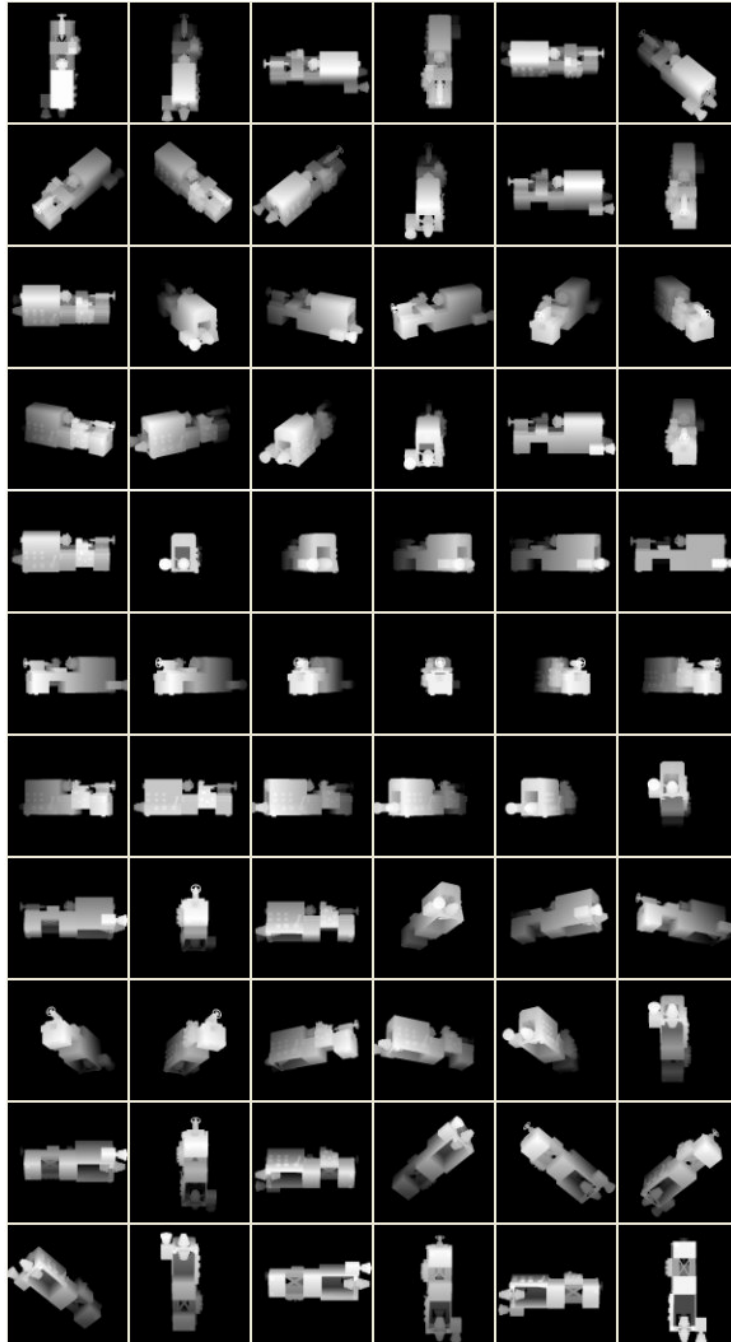


Figure 28: Training set for object “machine”

Since with 42 objects testing process time will be large, only 5 objects are used to define threshold values. These 5 objects are chosen based on some specific properties. Some objects have sharp and smooth edges. Also some objects have less number of features. In order to cover all types of objects agfa, machine, igea, vette and pitbull objects were selected in order to perform tests for threshold estimation and feature definition.

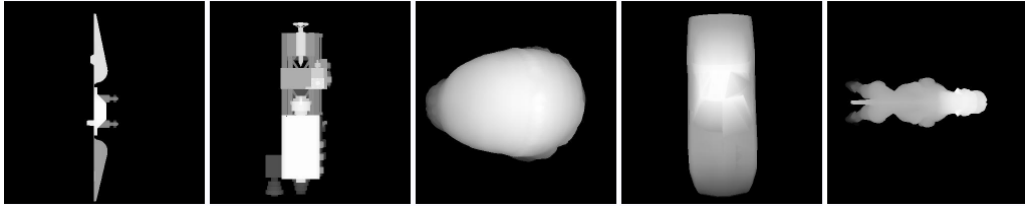


Figure 29: Selected 5 objects to test feature descriptors. (agfa, machine, igea, vette and pitbull)

Figure 29 shows the selected objects to test feature descriptor classifications. Note that agfa and machine have sharp edges and other objects have smooth edges. Also for agfa, igea and vette number of extracted features are less than other objects.

The objective of tests is to define threshold values and to obtain which types and which feature descriptors represent the objects in the database better. For this purpose a series of tests are executed.

According to the test plan a base test is chosen. After the tests were executed by changing only one property, better test characteristics were identified. After all tests were finished best resulting test is executed with the 25 and 42 objects databases to see the results in whole database.

To begin with first test, angle and normal angles feature values are used. Feature types: peak, saddle ridge, plane, pit and saddle valley are used since these feature types are used since they are primary features for human's observations.

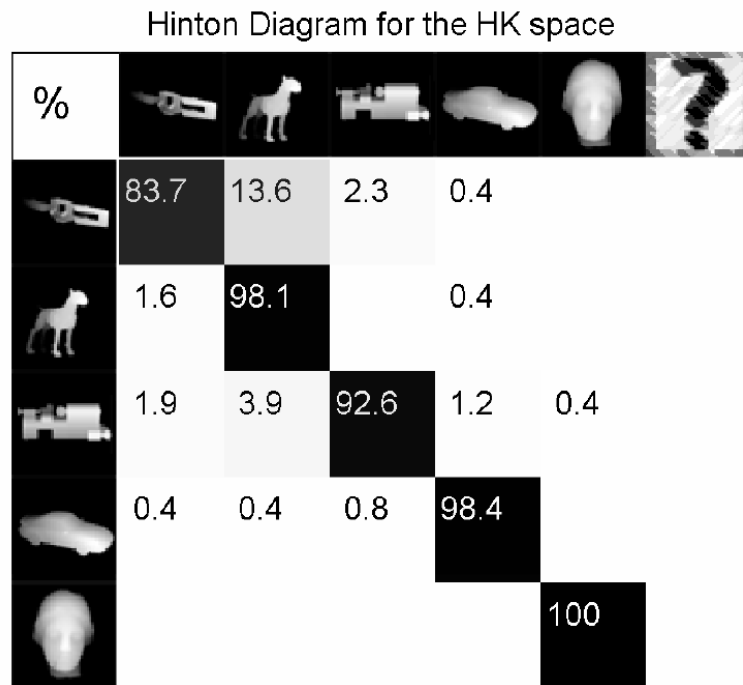
5.1 Base Test:

Test settings:

- 10 biggest features based on their volume values will be used.
- Feature numbers in each group: 3
- Types used: 1(peak), 3 (saddle ridge), 5(plane), 7(pit), 9(saddle valley)
- Feature descriptors: Angle and Normal Angle
- Thresholds: Angle: 10, Normal Angle: 10
- Feature vector comparison: Euclidian Similarity

This is the base test for further analysis. Improvements will be observed by comparing the results with the base test results. In this test and further tests agfa, machine, igea, vette and pitbull objects are used.

Hinton diagram and rank histogram for HK test result are shown in Figures 30-31.



Overall Success in HK space : %94.57

Figure 30: Hinton diagram for base test for HK space

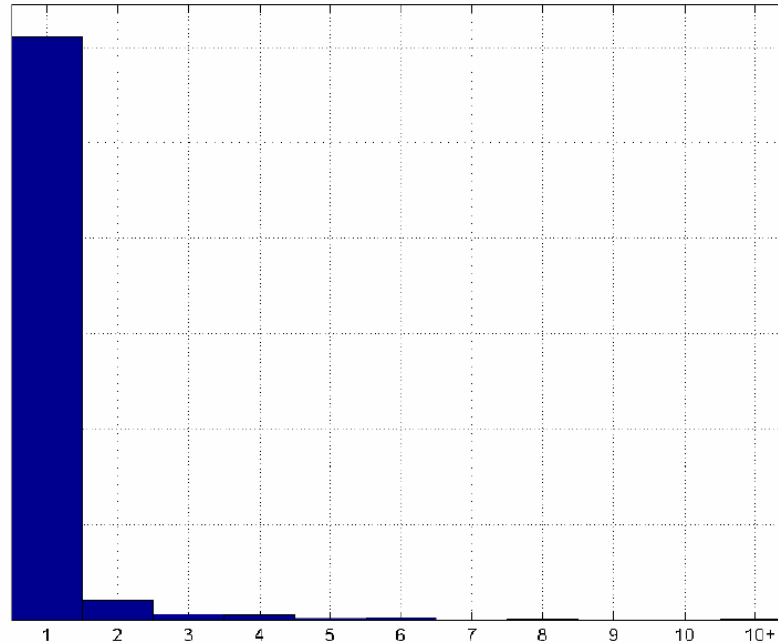


Figure 31: Rank histogram for base test for HK space

According to this base test result it can be seen that results having %94.57 correct detection rate is an encouraging result.

When the Hinton diagrams for the test result is observed, it can be seen that for “agfa” object recognition rate is pretty low. That may be caused by the sharp edges of the object. However; “machine” object has sharp edges and recognition rate for “machine” is higher. The recognition rate may be resulted from the high number of detailed features in “machine” object.

5.2 Elimination Test:

Test settings:

- 10 biggest features based on their radius values will be used.
- Feature numbers in each group: 3
- Types used: 1(peak), 3 (saddle ridge), 5(plane), 7(pit), 9(saddle valley)

- Feature descriptors: Angle and Normal Angle
- Thresholds: Angle: 10, Normal Angle: 10
- Feature vector comparison: Euclidian Similarity

For this test only feature elimination will be done with the radius of the features. The aim of this test is to compare effect of the radius and volume values of features since they both give information about the size of a feature.

Figures 32-33 show Hinton diagram and rank histogram for the elimination test.

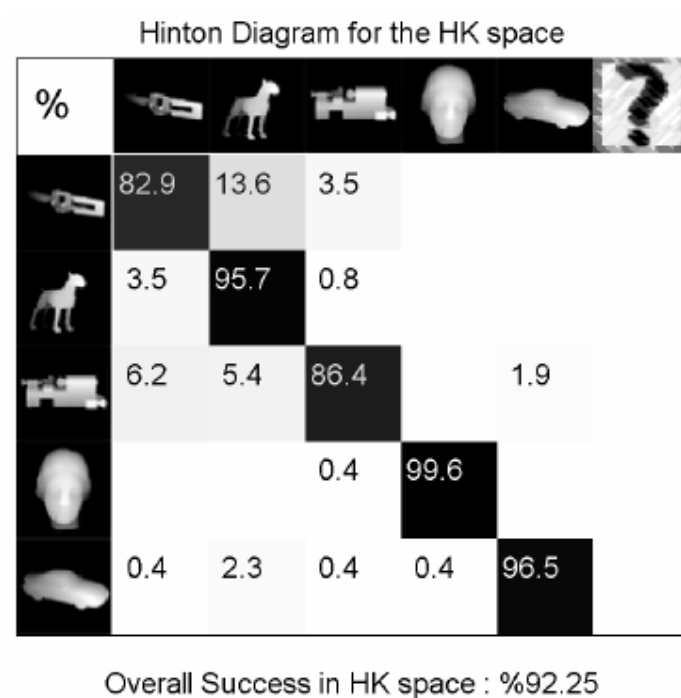


Figure 32: Hinton diagram for elimination test for HK space

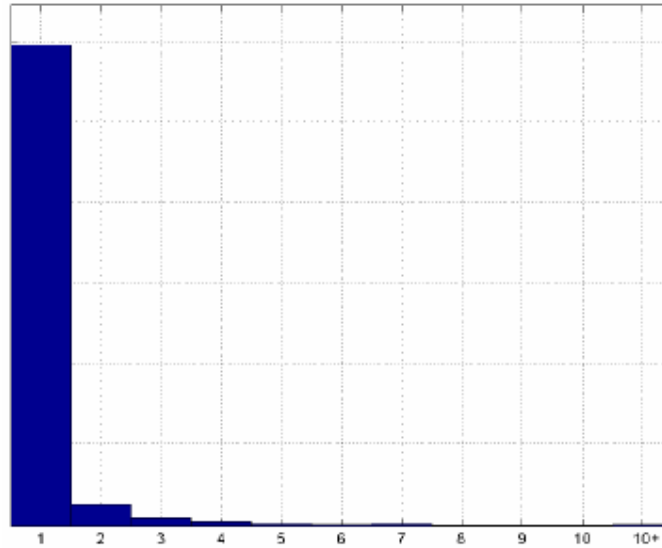


Figure 33: Rank histogram for elimination test for HK space

It can be seen from the Hinton diagram that, for all objects correct recognition rate is reduced. Selection of features based on their volumes leads to better recognition rate than selection of features based on their radius values. According to this test, it is shown that elimination according to the volume values is more descriptive than elimination according to the radius values. This result shows that, not the area of the surface shapes but the total volume of the surface shapes are descriptive for the database that is used.

5.3 Feature Group Test

Test settings:

- 10 biggest features based on their volume values will be used.
- Feature numbers in each group: 4
- Types used: 1(peak), 3 (saddle ridge), 5(plane), 7(pit), 9(saddle valley)
- Feature descriptors: Angle and Normal Angle
- Thresholds: Angle: 10, Normal Angle: 10

- Feature vector comparison: Euclidian Similarity

For this test features are grouped into four instead of three as explained in Chapter 3.

The aim of this test is to see how grouping affects the recognition rate.

Figures 34-35 show Hinton diagram and rank histogram for the feature group tests.

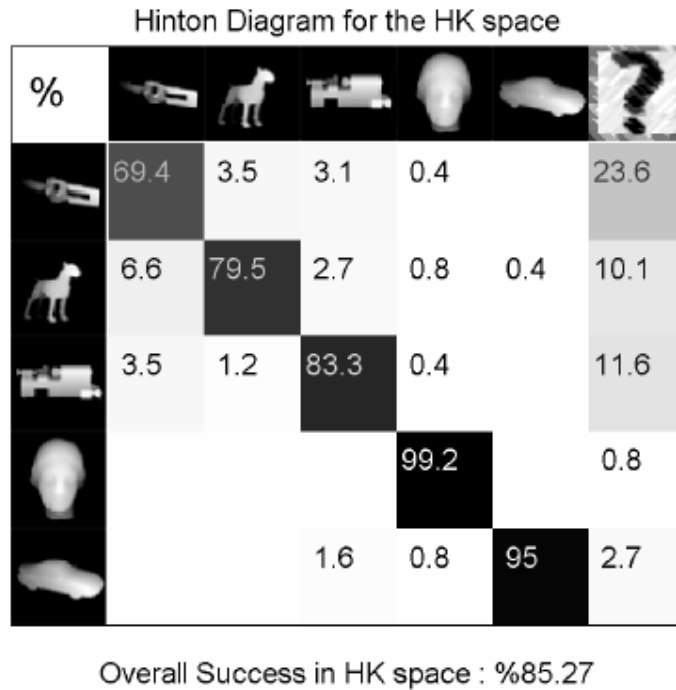


Figure 34: Hinton diagram for feature group test for HK space

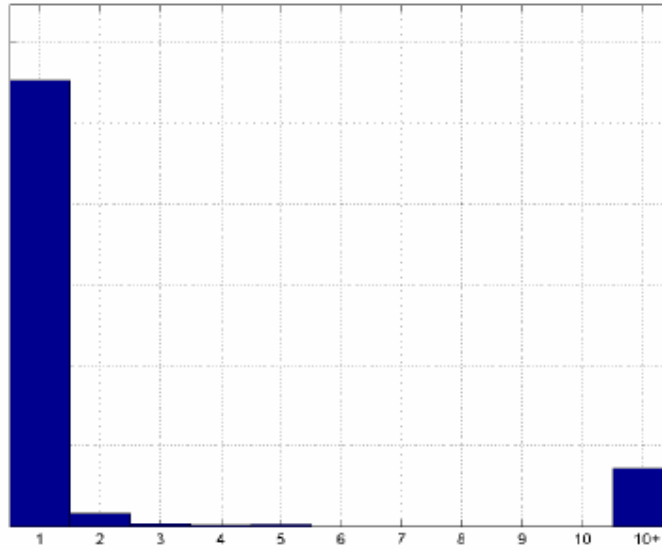


Figure 35: Rank histogram for feature group test for HK space

When the tests are done by feature grouping with four, a huge decrease in recognition rate is observed. And having too many unidentified objects shows that features with groups of four is unreliable. This decrease in the recognition rate may be a result of increase in computations while comparing feature vectors. Since there will be more values in feature vector, possibility of being out of range for the given threshold can be increased.

5.4 Length Values Added Test

Test settings:

- 10 biggest features based on their volume values will be used.
- Feature numbers in each group: 3
- Types used: 1(peak), 3 (saddle ridge), 5(plane), 7(pit), 9(saddle valley)
- Feature descriptors: Angle, Length and Normal Angle
- Thresholds: Angle: 10, Normal Angle: 10, Length 20
- Feature vector comparison: Euclidian Similarity

In this test Length values are added to our base test. The aim of this test is to observe if length values of features are good representation for objects.

Figures 36-37 show Hinton diagram and rank histogram for the length value added test.

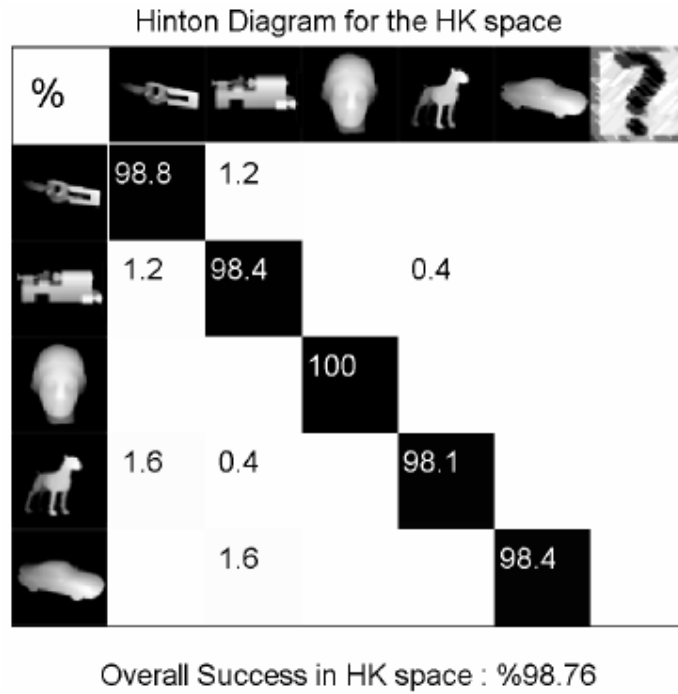


Figure 36: Hinton diagram for length values added test for HK space

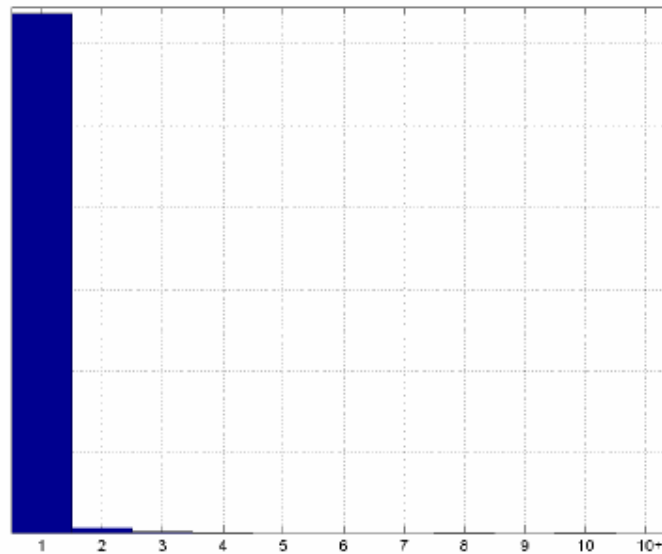


Figure 37: Rank histogram for length values added test for HK space

Adding length values into the feature vector greatly increased the recognition rate. The increase in recognition rate may be the result of the correct localization of features. For a database, where there is no scaling, the length values provide more information about the object and this increases the performance. However scale effect on the database may lead to a decrease in the recognition rate if length values are used with Euclidian similarity.

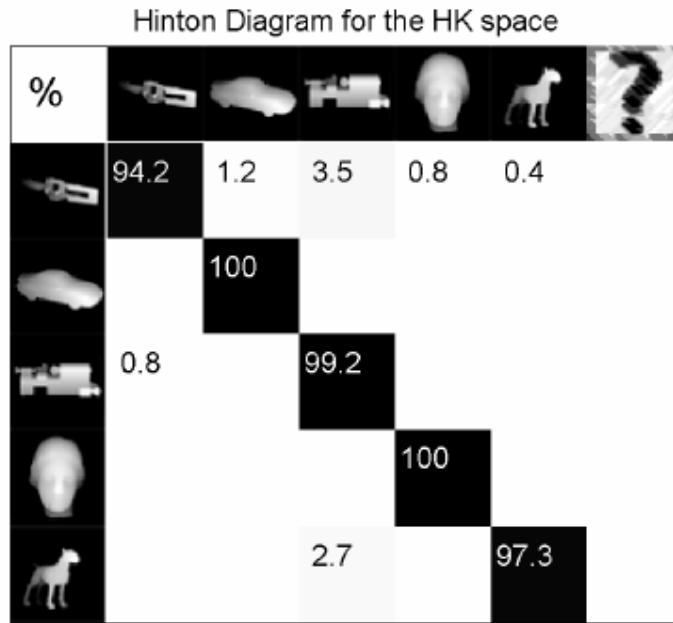
5.5 Individual Max Test

Test settings:

- 4 biggest features for each type, based on their volume values will be used.
- Feature numbers in each group: 3
- Types used: 1(peak), 3 (saddle ridge), 5(plane), 7(pit), 9(saddle valley)
- Feature descriptors: Angle, Length and Normal Angle
- Thresholds: Angle: 10, Normal Angle: 10, Length 20
- Feature vector comparison: Euclidian Similarity

In this test feature elimination will be done according to the size of features for each feature type individually. All feature types may have the same importance while representing the object. For example, for a specific object, a saddle feature can be very important even if it has a small size. The aim of this test is to see if individual feature types represent objects better.

Figures 38-39 show Hinton diagram and rank histogram for individual maximum test.



Overall Success in HK space : %98.14

Figure 38: Hinton diagram for individual max test for HK space

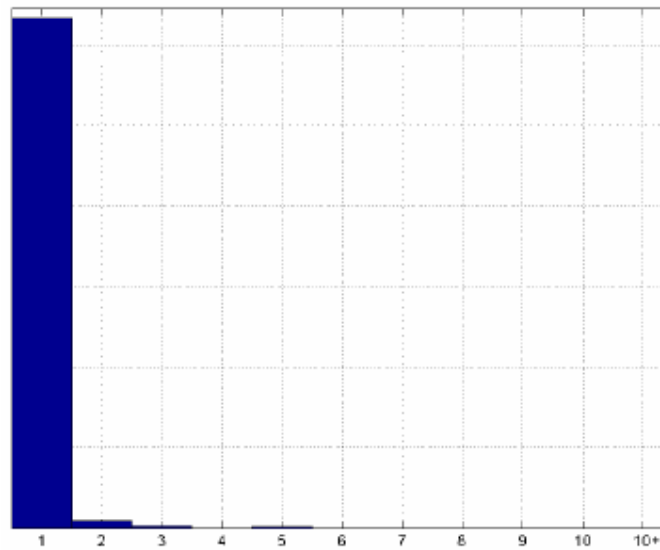


Figure 39: Rank histogram for individual max test for HK space

Having individual maximum values for each feature decreased the recognition rate although more features are used. Having unnecessary detailed features might be the reason of this decrease. When Hinton diagram is observed low recognition rate for

“agfa” object can be noticed. As it has been explained before, “agfa” object has many small detailed features. So having individual maximum decreases the performance since too many small and detailed features are included in object representation.

5.6 Angle Tests

To identify best angle threshold following tests have been executed.

- 10 biggest features based on their volume values will be used.
- Feature numbers in each group: 3
- Types used: 1(peak), 3 (saddle ridge), 5(plane), 7(pit), 9(saddle valley)
- Feature descriptors: Angle, Length and Normal Angle
- Thresholds: Angle: {5, 10, 20}, Normal Angle: 10, Length 20
- Feature vector comparison: Euclidian Similarity

The aim of these tests is to observe recognition rate change with the change in the angle threshold value. With other settings being unchanged, angle threshold value is set to 5, 10, 20 and 30.

According to these tests, results at Table 6 have been obtained.

Table 6: Angle Test Results

Angle Tests	Angle {5}	Angle {10}	Angle {20}	Angle {30}
Recognition Rate (%)	98,44	98,75	98,6	98,2

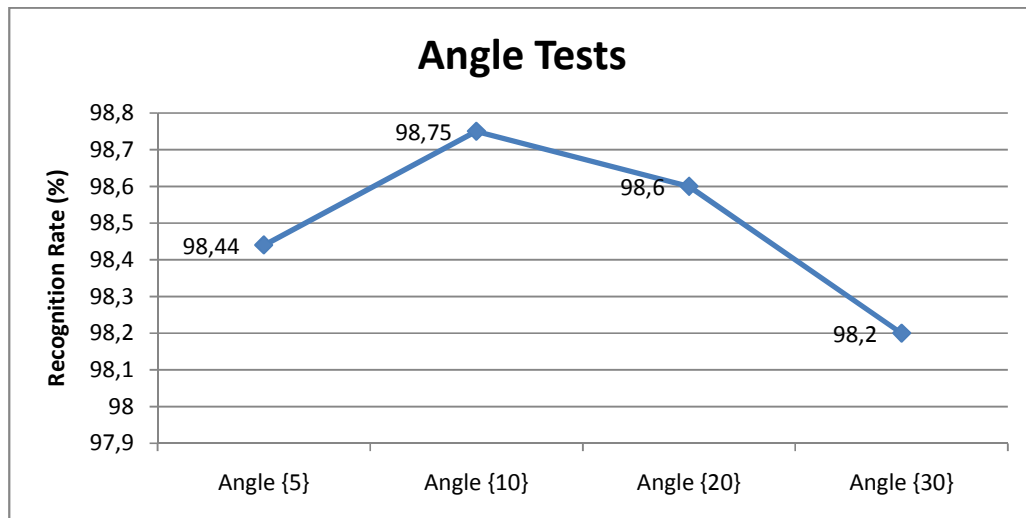


Figure 40: Angle Tests

According to the Figure 40 these tests show with an angle threshold value 10, recognition correctness rate is better for HK space. Optimum angle threshold value is realistic since 10° is not much. For different database objects optimum angle threshold value may vary if database objects are similar to each other such as faces. For further test steps angle threshold value 10 will be used.

5.7 Length Tests

To identify best length threshold following tests have been executed.

- 10 biggest features based on their volume values will be used.
- Feature numbers in each group: 3
- Types used: 1(peak), 3 (saddle ridge), 5(plane), 7(pit), 9(saddle valley)
- Feature descriptors: Angle, Length and Normal Angle
- Thresholds: Angle: 10, Normal Angle: 10, Length { 10, 20, 30}
- Feature vector comparison: Euclidian Similarity

The aim of these tests is to observe recognition rate change with the change in the length threshold value. With other settings being unchanged, length threshold value is set to 10, 20 and 30.

According to these tests, results at Table 7 have been obtained.

Table 7: Length Test Results

Length Tests	Length {10}	Length {20}	Length {30}	Length {40}
Recognition Rate (%)	97,28	98,75	98,52	98,32

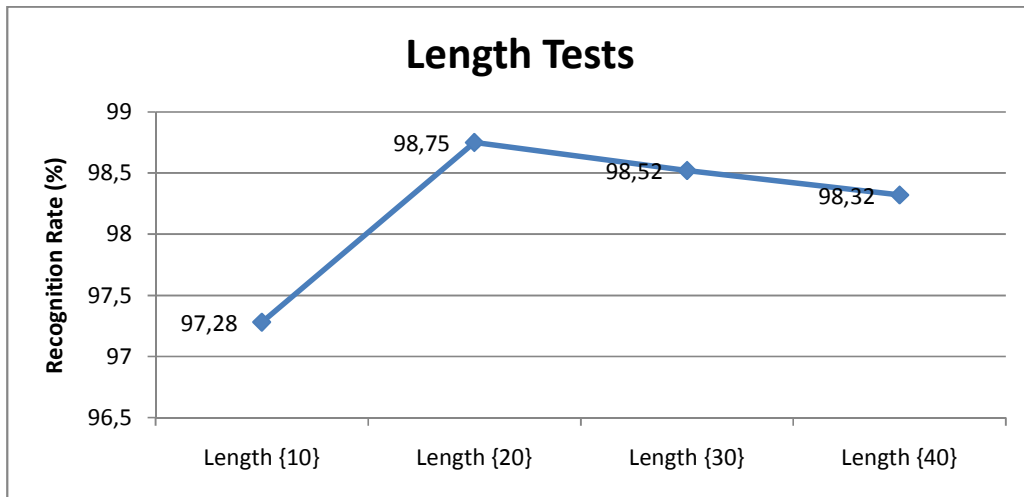


Figure 41: Length Tests

According to Figure 41 these test results show that with length threshold value 20, recognition rate is better for HK space. Decrease in the recognition rate for length threshold value 10 shows that feature localization is not sufficient enough since average total length for objects is about 200. With more accurate feature extraction optimum length threshold should decrease and recognition rate should be increased. For further test steps length threshold value 20 will be used.

5.8 Normal Angle Tests

To identify best normal angle threshold following tests have been executed.

- 10 biggest features based on their volume values will be used.
- Feature numbers in each group: 3
- Types used: 1(peak), 3 (saddle ridge), 5(plane), 7(pit), 9(saddle valley)
- Feature descriptors: Angle, Length and Normal Angle
- Thresholds: Angle: 10, Normal Angle: {10, 20, 30, 40, not used}, Length 20
- Feature vector comparison: Euclidian Similarity

The aim of these tests is to observe recognition rate change with the change in the normal angle threshold value. With other settings being unchanged, normal angle threshold value is set to 10, 20 and 30.

According to these tests, results at Table 8 have been obtained.

Table 8: Normal Angle Test Results

Normal Angle Tests	Normal {5}	Normal {10}	Normal {20}	Normal {30}	Normal {40}	Normal {not used}
Recognition Rate (%)	93,17	98,75	99,53	99,84	99,53	99,45

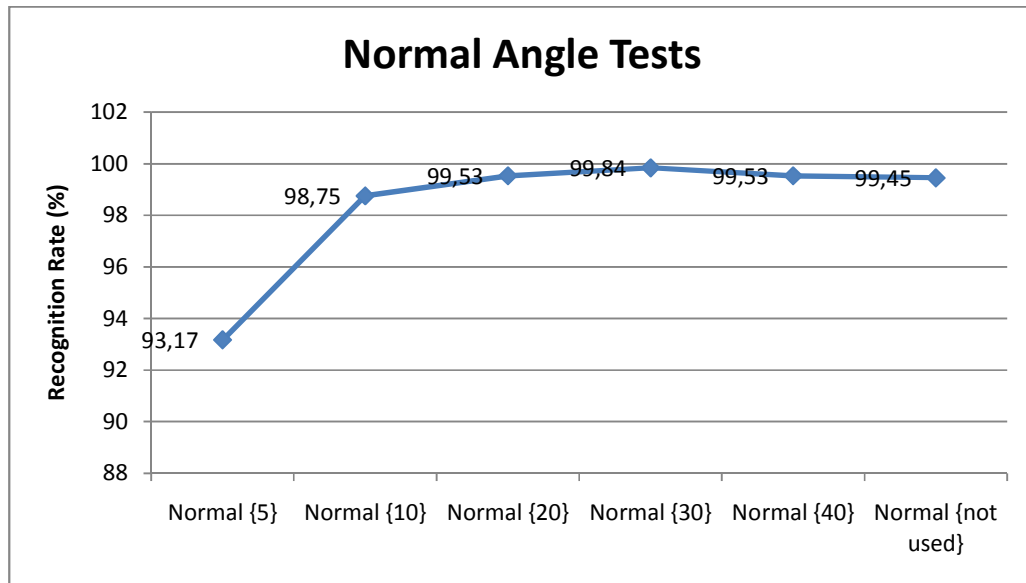


Figure 42: Normal Angle Tests

According to the Figure 42 although with a normal angle threshold value 30 gives best results, it can be seen from the Figure 42 that having no normal angle threshold results do not change much. That is because of normal angle threshold is not reliable for our tests. Inefficient feature localization might have affected the results. So for further test steps normal angle will not be used.

5.9 Type Tests

To identify best discriminative types following tests have been executed.

- 10 biggest features based on their volume values will be used.
- Feature numbers in each group: 3
- Types used: {1, 3, 7, 9}, {1, 3, 5, 7, 9}, {1, 2, 3, 5, 7, 9}, {1, 3, 5, 6, 7, 9}, {1, 3, 5, 7, 8, 9}, {All}
- Feature descriptors: Angle and Length

- Thresholds: Angle: 10, Length 20
- Feature vector comparison: Euclidian Similarity

The aim of these tests is to observe recognition rate change with the change in the feature types used. With other settings being unchanged, different feature types are used.

According to these tests, results at Table 9 have been obtained.

Table 9: Type Test Results

Type Tests	{1 3 7 9}	{1 3 5 7 9}	{1 2 3 5 7 9}	{1 3 5 6 7 9}	{1 3 5 7 8 9}	{All}
Recognition Rate (%)	97,13	99,45	98,91	98,99	99,3	98,91

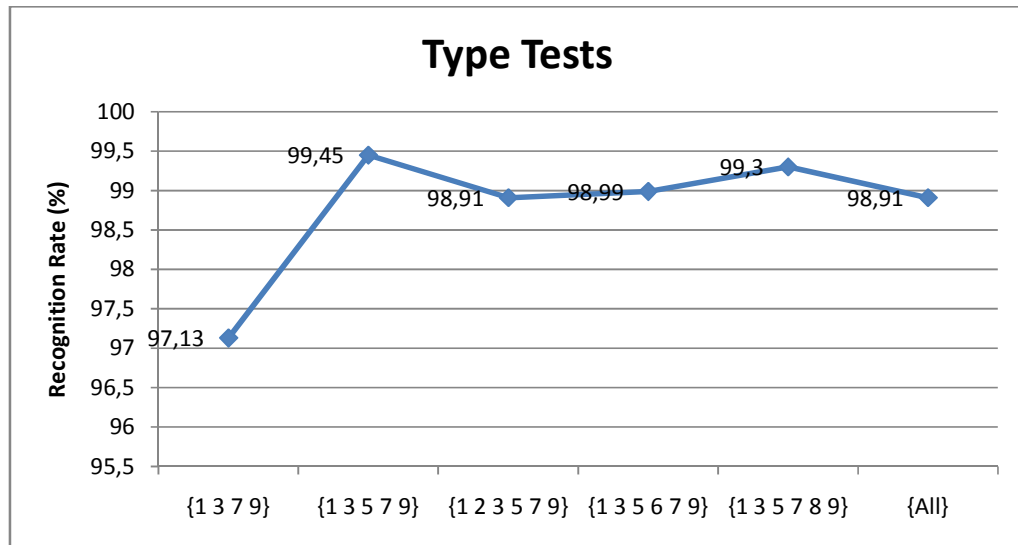


Figure 43: Type Tests

According to the Figure 43 best results for HK space has been obtained with types {1, 3, 5, 7, 9}. As explained before, peak, saddle ridge, plane, pit and saddle valley types are more familiar to human perceptions. Having best results with these feature types was an expected result. So the assumption for base test was true for HK spaces that human being can also recognizes objects with these feature types.

5.10 Feature Number Tests

To identify best general maximum values following tests have been executed.

- 6, 8, 10, 14, 16, 18 biggest features based on their volume values will be used.
- Feature numbers in each group: 3
- Types used: 1(peak), 3 (saddle ridge), 5(plane), 7(pit), 9(saddle valley)
- Feature descriptors: Angle and Length
- Thresholds: Angle: 10, Length 20
- Feature vector comparison: Euclidian Similarity

The aim of these tests is to observe recognition time change with the change in the feature numbers used. With other settings being unchanged, feature number is set to 6, 8, 10, 14, 16 and 18.

According to these tests, results at Table 10 have been obtained.

Table 10: General Max Test Results

	n=6	n=8	n=10	n=14	n=16	n=18
Feature Number Tests	{~40 mins }	{~100 mins }	{~3 hours }	{~13 hours }	{~30 hours }	{~50 hours }
Recognition Rate (%)	96,82	98,74	99,45	99,68	99,61	99,68

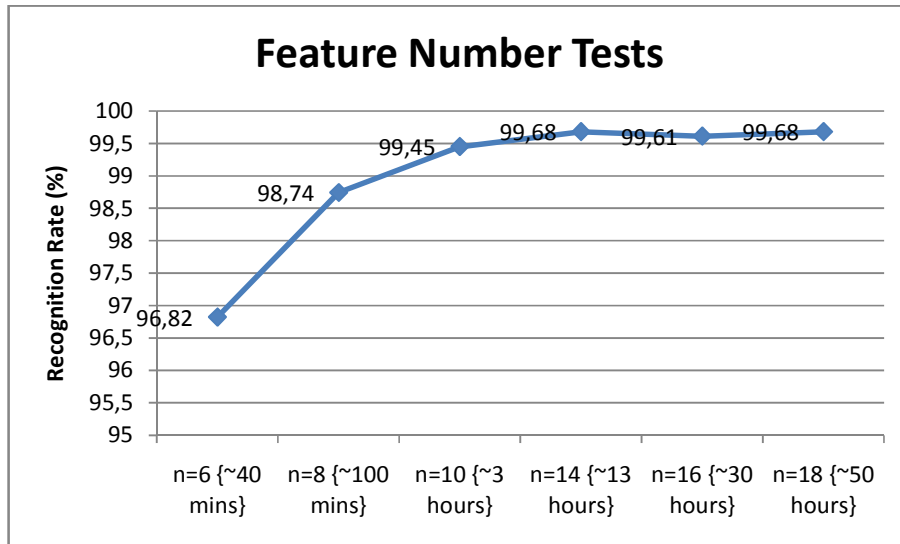


Figure 44: General Max Tests

Figure 44 shows that after 10 features HK space test results do not increase much however computation time increases rapidly. Therefore 10 features is the optimum number for tests. The change in recognition rate and computation time versus feature number is as expected. With more features objects can be represented better. Also with more feature vector construction, comparison count will increase and this will result increase in computation time.

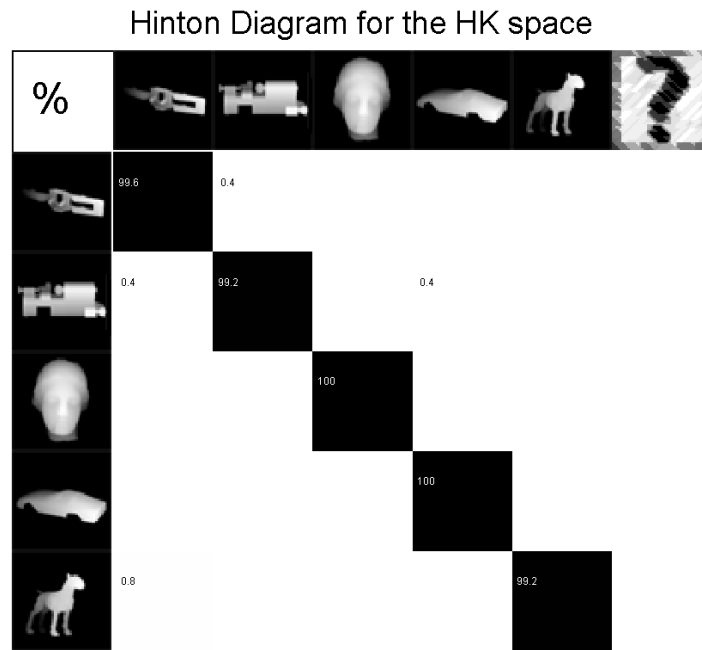
5.11 City Block Similarity Test:

Test settings:

- 10 biggest features based on their volume values will be used.
- Feature numbers in each group: 3
- Types used: 1(peak), 3 (saddle ridge), 5(plane), 7(pit), 9(saddle valley)
- Feature descriptors: Angle, Length and Normal Angle
- Thresholds: Angle: 10, Length: 20, Normal Angle: 30
- Feature vector comparison: City Block Similarity

In the previous tests, best result has been obtained with %99.85 using the settings in this test. Aim of this test to observe the recognition rate difference between city block and Euclidian similarity.

Hinton diagram and rank histogram for HK test result are shown in Figures 45-46.



Overall Success in HK space : %99.61

Figure 45: Hinton diagram for city block similarity test for HK space

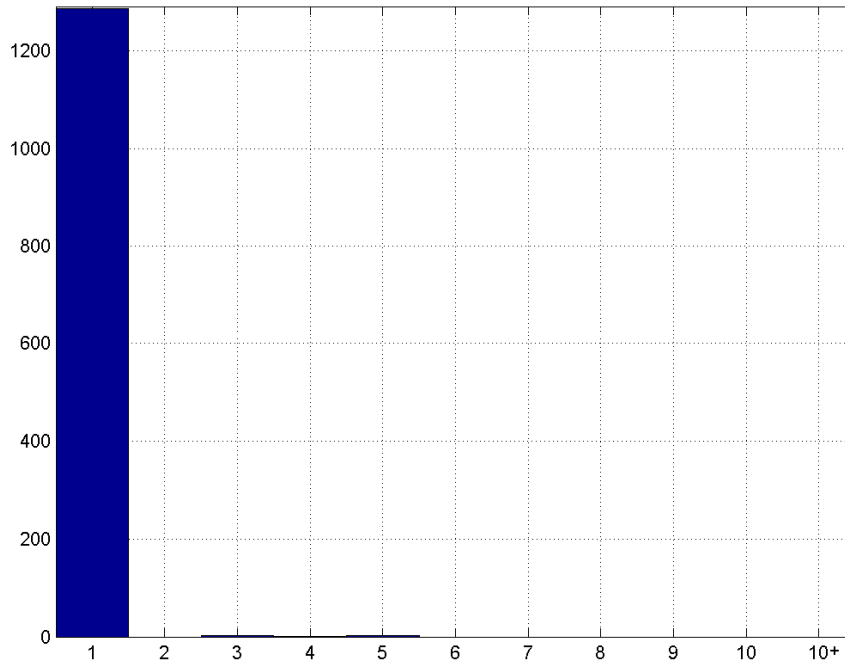


Figure 46: Rank histogram for city block similarity test for HK space

According to results of this test, it is shown that using Euclidian similarity or city block similarity does not change the results much. We have achieved better result with Euclidian similarity since it is more logical to use Euclidian distance between feature vectors.

5.12 Cosine Similarity Test

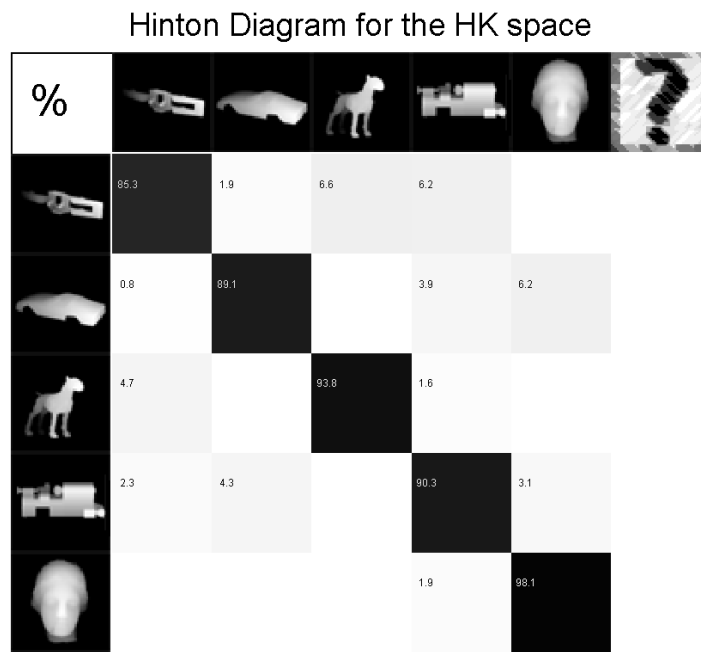
Test settings:

- 10 biggest features based on their volume values will be used.
- Feature numbers in each group: 3
- Types used: 1(peak), 3 (saddle ridge), 5(plane), 7(pit), 9(saddle valley)
- Feature descriptors: Angle, Length and Normal Angle
- Thresholds: Angle: 10, Length: 20, Normal Angle: 30

- Feature vector comparison: Cosine Similarity(Length), Euclidian Similarity (Normal Angle, Angle)

In the previous tests, best result has been obtained with %99.85 using the settings in this test. Aim of this test to observe the recognition rate difference between cosine and Euclidian similarity. Cosine similarity is only applied on length values since only length value for feature vector is dependent on scale.

Hinton diagram and rank histogram for HK test result are shown in Figures 47-48.



Overall Success in HK space : %91.32

Figure 47: Hinton diagram for cosine similarity test for HK space

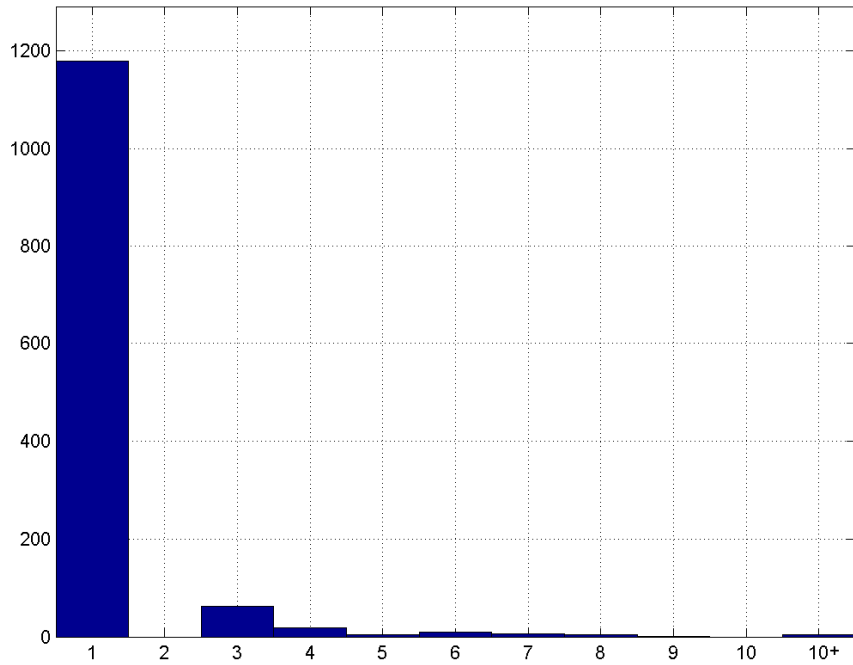


Figure 48: Rank histogram for cosine similarity test for HK space

According to results of this test, it is shown that using Euclidian similarity is better than cosine similarity. This result is obvious since database used in this test is not scaled. The reason for this increase is the increase of votes for irrelevant feature vectors with similar angle between features.

5.13 Cosine Similarity Test With Scaled Database

Test settings:

- 10 biggest features based on their volume values will be used.
- Feature numbers in each group: 3
- Types used: 1(peak), 3 (saddle ridge), 5(plane), 7(pit), 9(saddle valley)
- Feature descriptors: Angle, Length and Normal Angle
- Thresholds: Angle: 10, Length: 20, Normal Angle: 30

- Feature vector comparison: Cosine Similarity(Length), Euclidian Similarity (Normal Angle, Angle)
- 1 object with different scales (0.8, 1, 1.2) are used for testing

With cosine similarity test results decrease. However, scale invariant assumption with this similarity check, should improve the test results. Two scaled testing objects are created and then they are tested with cosine similarity and Euclidian similarity.

Hinton diagram for test results are shown in Figures 49-50.

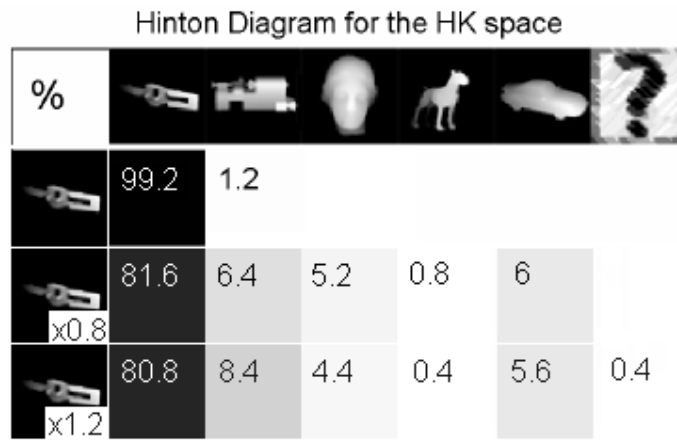


Figure 49: Hinton diagram for Euclidian similarity test

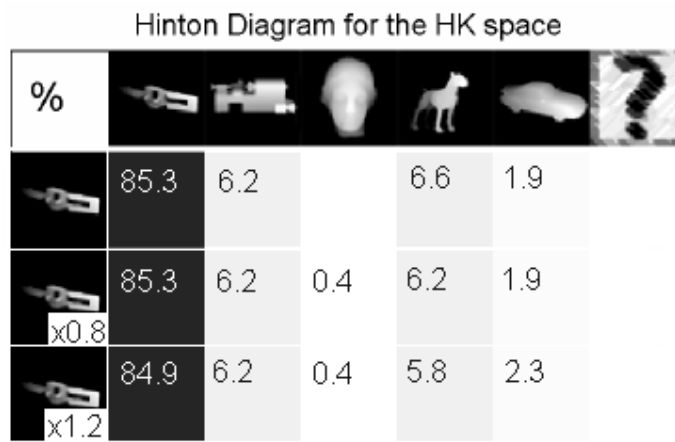


Figure 50: Hinton diagram for cosine similarity test

According to results of this test, it is shown that using Euclidian similarity is dependant to scale. On the other hand, scale independent testing has been achieved by using cosine similarity for length values.

5.14 Computation Time Tests

To identify best computation values following tests have been executed.

- 10 biggest features based on their volume values will be used.
- Feature numbers in each group: 3
- Types used: 1(peak), 3 (saddle ridge), 5(plane), 7(pit), 9(saddle valley)
- Feature descriptors: Angle and Length
- Thresholds: Angle: 10, Length 20
- Number of Objects: 5, 25, 42 objects
- Matching Method: Without Indexing, With Indexing
- Feature vector comparison: Euclidian Similarity

The aim of these tests is to observe recognition time change with the change in the objects number by using two different matching methods. One of the methods is “With Indexing” and it is expected that “With Indexing” recognition time will be reduced. With settings being unchanged, object number is set to 5, 25 and 42.

According to these tests, results at Table 11 have been obtained.

Table 11: Computation Time Test Results

Computation Time Tests	5 Objects	25 Objects	42 Objects
Without Indexing (Hours)	3	74	210
With Indexing (Hours)	1	21	63

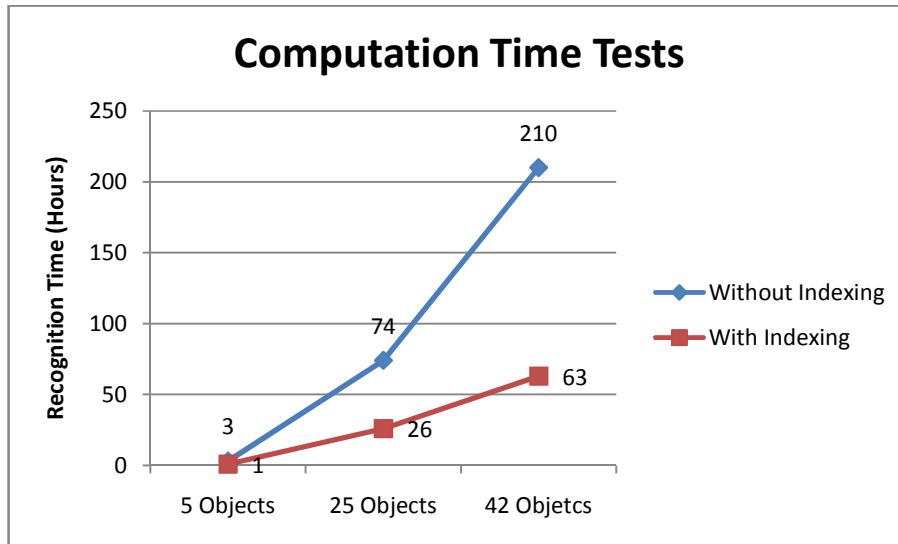


Figure 51: Computation Time Tests

Figure 51 shows that computation time decreases when indexing method is used. This result is not a surprise since less number of feature vectors are compared during matching.

Average feature vector comparison time is about 1.75 micro seconds. This time includes query, implementation, comparison and some small details. The time for comparing a single feature can be decreased with code or system improvements. However it is not the scope of this thesis. On the other hand comparison of the indexing method between brute-force matching is important.

Total computation time increases with the increasing number in the testing and training feature vectors. If a single pose is trained by the system and if a single pose is tested then the online computation time will be $120 \times 120 \times 1.75 \mu\text{s}$ (25.2 ms) (120 features for each pose). If the database is enhanced with 66 poses (1 object training and testing poses) then the computation time will be $66 \times 25.2 \text{ ms}$ (1.66 s). If the whole 42 training objects are included in database then for a single pose testing time will yield to $42 \times 1.66 \text{ s}$ (70 secs). Since there are 258 testing poses for each 42 objects 210 hours testing is an expected result ($258 \times 42 \times 70 \text{ s} = 210 \text{ h}$).

With indexing feature vector comparison is not changed. However not all but only the features in the same region is compared. That's why the computation time reduces by about %70.

5.15 Best Fitting Tests

With the tests done in Sections 5.1-5.10 the best fitting test settings defined as:

- 10 biggest features based on their volume values will be used.
- Feature numbers in each group: 3
- Types used: 1(peak), 3 (saddle ridge), 5(plane), 7(pit), 9(saddle valley)
- Feature descriptors: Angle and Length
- Thresholds: Angle: 10, Length 20
- Number of objects used: 5, 25, 30(for literature comparison), 42(all)
- Feature vector comparison: Euclidian Similarity

The aim of these tests is to observe recognition rate change with the change in the objects number. It is expected that recognition rate will be decreased with the increase in the objects used for testing. With settings being unchanged, object number is set to 5, 25, 30 and 42.

According to these tests, results at Table 12 have been obtained. For 25 and 42 objects Hinton diagrams and rank histograms for these tests are shown in Figures 53-56.

Table 12: Best Fitting Test Results

Best Fitting Tests	5 Objects	25 Objects	30 Objects	42 Objects
Recognition Rate (%)	99,45	93,04	92,85	90,97

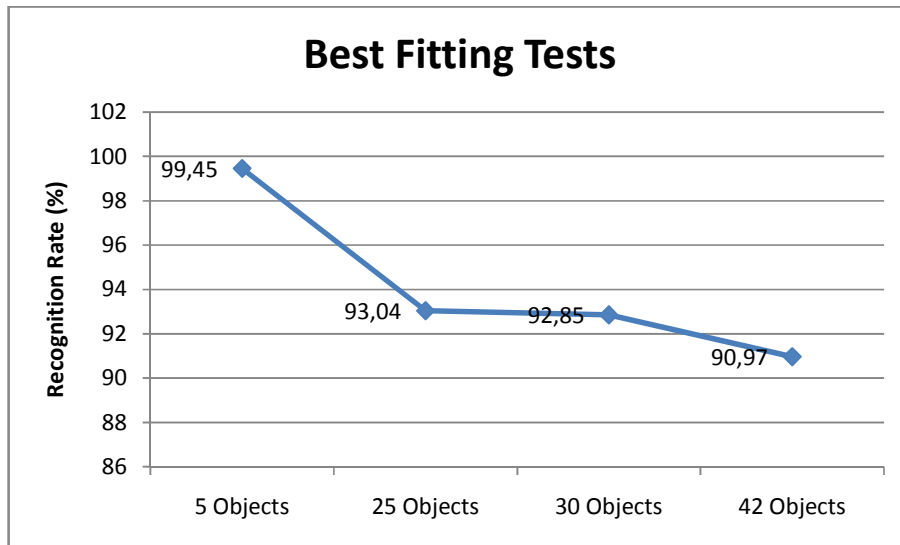


Figure 52: Best Fitting Tests

Figure 52 shows that recognition rate decreases with the increase for number objects in database. This decrease is normal since incorrect recognition will increase with the increase in training test objects.

HK 25 Objects (%93.04)

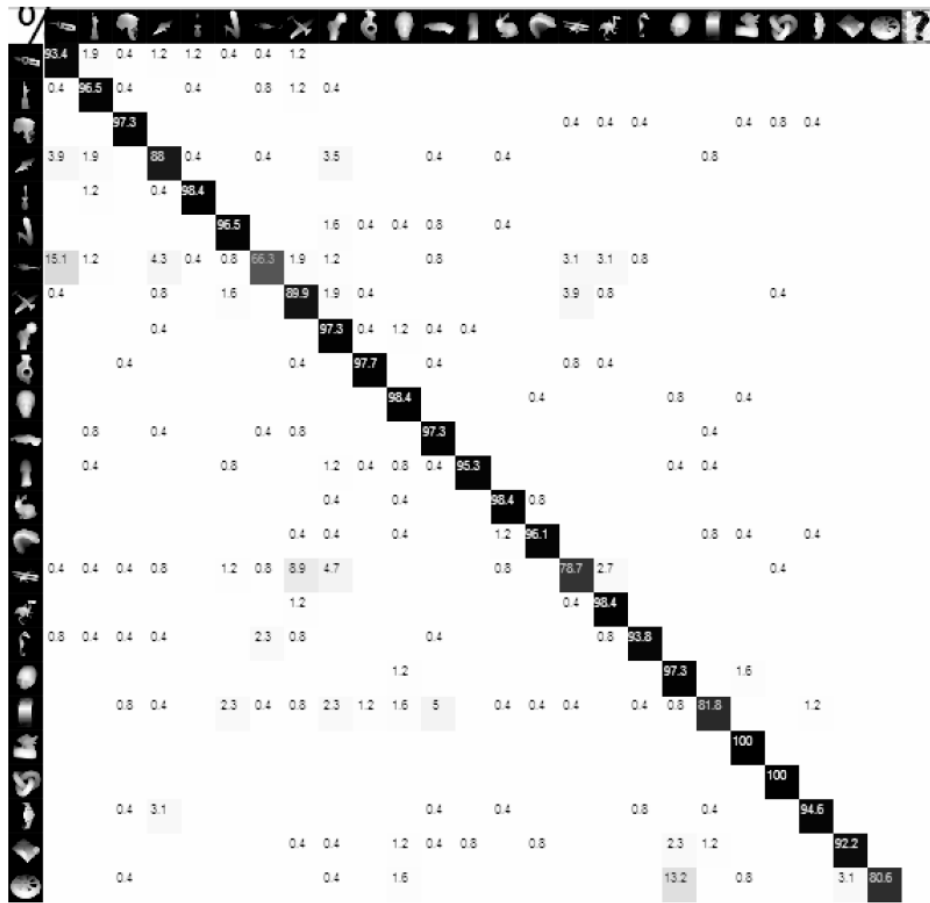


Figure 53: Hinton Diagram for HK space, 25 objects

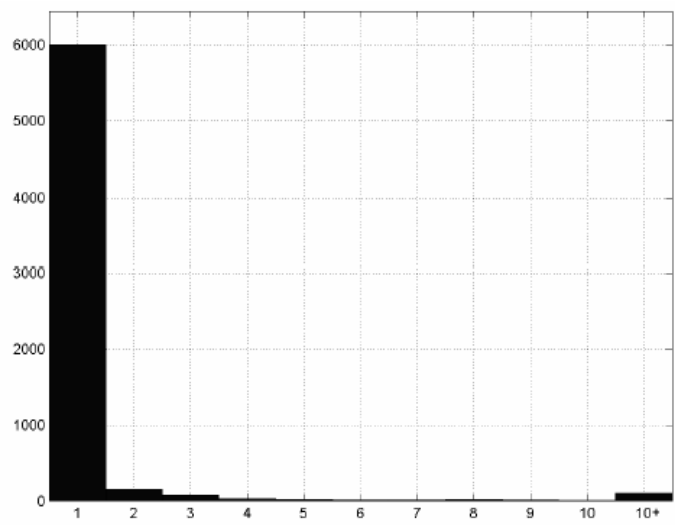


Figure 54: Rank Histogram for HK space, 25 objects

HK 42 Objects (%90.97)

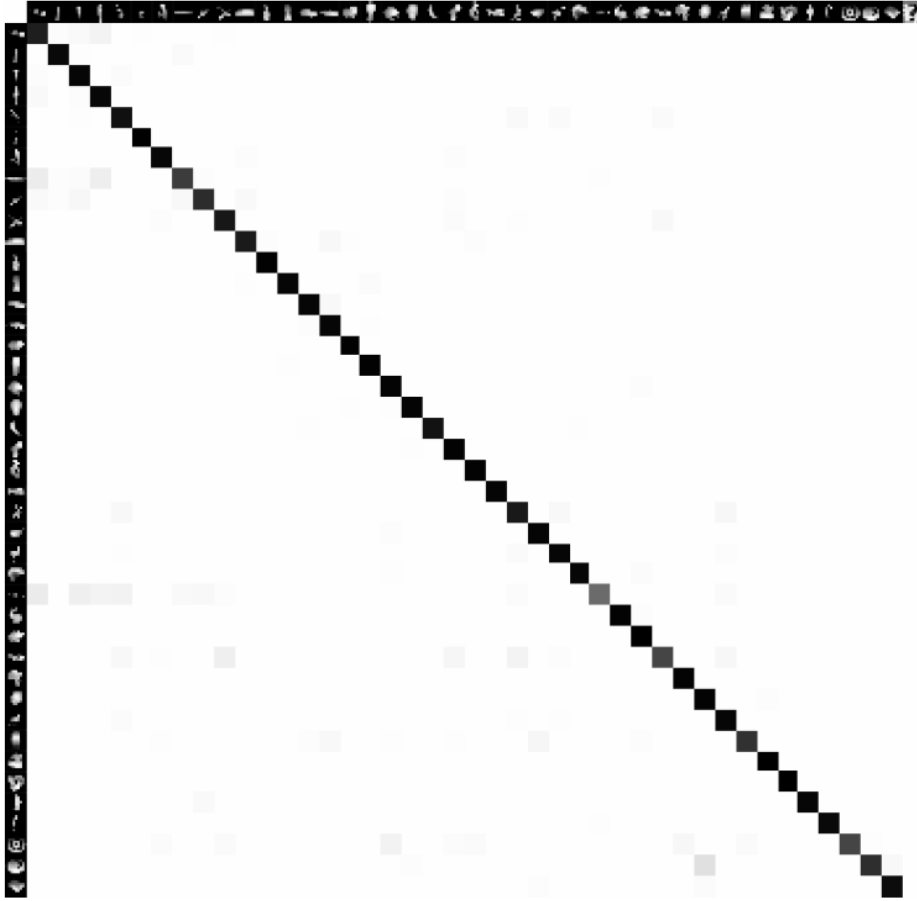


Figure 55: Hinton Diagram for HK space, 42 objects

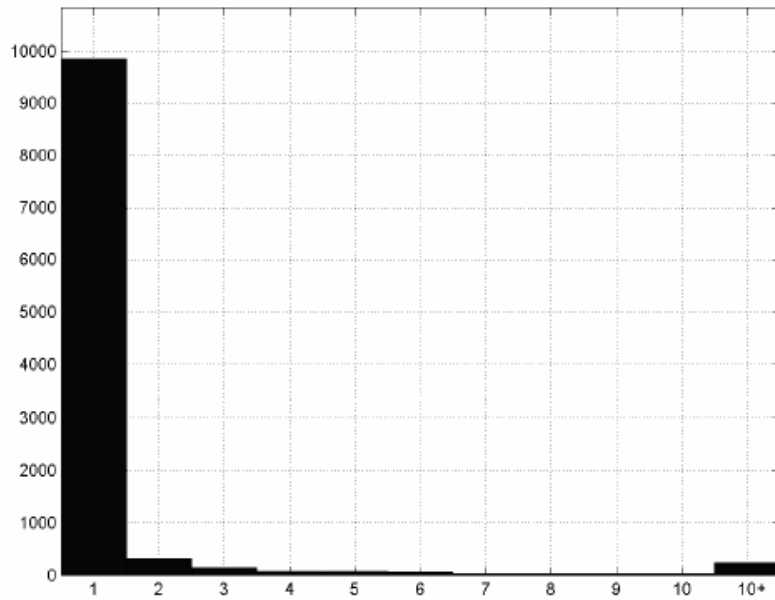


Figure 56: Rank histogram for HK space, 42 objects

5.16 Analysis of the Overall Results

Tests are performed on Stuttgart database where 66 poses of 42 objects are stored in the hash table during training and 258 poses of 42 objects are used during testing. %90.97 recognition rate is achieved for 42 objects. For a comparison with the results in the literature, 30 objects out of 42 have been recognized with a rate of %92.85, where in [27] recognition rate for these 30 objects is %93 and in [28] recognition rate is %98. Comparing with the results obtained before, recognition rate is not better but fairly adequate.

As explained before extracted features have volume, radius and scale values. However test results with these value settings haven't mentioned. Since tests with using volume, radius and scale values were unsuccessful compared to the tests done, these tests aren't included in experiments and results section. Feature extraction method used in this thesis is not an ended study. With the improvements in multiscale surface characteristics, test results will increase.

With the whole database analysis it can be seen that with mechanical objects recognition rate is low compared to other objects. This is because of the feature extraction method being descriptive for natural objects. In other words, surface definitions as peak, pit and saddle features can be detected clearly in natural objects. However for sharp edges, pit and peak features are less like to be confronted where it is nearly impossible to observe a saddle feature.

Brief analyses of tests done are given in Table 13.

Table 13: Brief explanation for tests

Test Name	Maximum Overall Result (5 Objects)	Explanation
Base Test	%94.57	Further tests are analyzed according to this test result
Elimination Test	%92.25	Elimination according to radius values decreased recognition rate
Feature Group Test	%85.27	Test with 4 features in a group decreased recognition rate

Table 13: Brief explanation for tests

Test Name	Maximum Overall Result (5 Objects)	Explanation
Length Values Added Test	%98.75	When length values are added recognition rate is increased
Individual Max Test	%98.14	Having individual maximum for each feature type decreased recognition rate
Angle Tests	%98.75	Maximum result achieved with angle threshold value 10
Length Tests	%98.75	Maximum result achieved with length threshold value 20
Normal Angle Tests	%99.84	Maximum result achieved with normal angle threshold value 30. However without normal angle values recognition rate is not affected much.
Type Tests	%99.45	Maximum result is achieved with feature types 1(peak), 3 (saddle ridge), 5(plane), 7(pit) and 9(saddle valley)
Feature Number Tests	%99.68	With the increase in feature number increased recognition rate. However 10 features is optimum for recognition rate and computation time
City Block Similarity Test	%99.61	City block similarity does not affect recognition rate much
Cosine Similarity Test	%91.32	Cosine similarity decreased recognition rate
Cosine Similarity Test With Scaled Database	N/A	Using cosine similarity satisfy scale invariant recognition
Computation Time Tests	N/A	Indexing method decreases recognition rate about %70
Best Fitting Tests	%99.45	%90.97 recognition rate is achieved with 42 objects

CHAPTER 6

CONCLUSION

6.1 Work Done

In this thesis, by using HK values on 3D range data objects, is a scale space representation, representative surface features are extracted. In order to have transform invariant feature vectors, features are grouped and instead of location values, angles between features and angles between normal vectors of features are used. In addition to angle values, lengths between features are also obtained. Each pose of each object is indexed in a hash table using these groups of features.

Main aim of this thesis is to identify which feature descriptor is better to be used for classification and to define the threshold values that need to be applied for recognition.

Another aim is to decrease the computational time as much as possible without affecting the recognition rate. Comparison of the whole computation time with the literature is not in the scope of this thesis. However computation time reduction is achieved with indexing the feature vectors into intervals. In order to achieve this, a new hashing method is proposed. According to the experiments done for the database it is shown that this new hashing method reduces execution time by %70. The reason for not getting computation times as expected is due to the database system on this thesis. Since database is stored in MySQL database system, by hashing method, accessing database reduced true computation time.

With the series of experiments it is shown that angle and length values of feature vectors obtained by feature grouping are better at object representation. This shows that localization of features do not change much with some rotation. And tests show that localization calculations have minimum error against other feature descriptors.

Another result is that, among 8 feature types, peak, saddle ridge, plane, pit and saddle valley show better characteristics in defining objects.

Proposed method can work with a different database of objects. However threshold values may differ according to the common similarities in database. For example, threshold values should be reduced to work with a face recognition implementation since, faces have small detailed differences.

6.2 Future Works

Current system is using partial matching. It can be observed how new matching methods fit with the features used in this thesis. In addition, new features may be used to identify their feature characteristics.

Feature extraction used in this thesis best fits with the natural object representation. Therefore, recognition rate does not show an improvement when compared to the previous studies implemented on the same database. With some addition to feature value, object representation can be improved. In addition to these, new surface characteristics can be added, holes on the objects can be used as feature type.

As a future study, pose estimation can be studied. With the triples of features extracted from the objects rotations and transformation of objects can be estimated. In the light of the results with these works pose of the matching objects can be obtained.

REFERENCES

- [1] E. Akagündüz and I. Ulusoy (2007). Extraction of 3D Transform and Scale Invariant Patches from Range Scans, Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2nd Beyond Patches Workshop, pp. 1-8.
- [2] Richard J. Campbell and Patrick J. Flynn(2001), “A Survey Of Free-Form Object Representation and Recognition Techniques ”Computer Vision and Image Understanding 81, 166–210
- [3] P. J. Besl and R. C. Jain (1988). Segmentation Through Variable-Order Surface Fitting, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 10, no. 2, pp. 167-192.
- [4] P. J. Burt and E. Adelson (1983). The Laplacian Pyramid as a Compact Image Code” IEEE Trans. Communications, vol. 31, no. 4, pp. 532-540.
- [5] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray (2004). Visual categorization with bags of keypoints, In Workshop on Statistical Learning in Computer Vision, ECCV.
- [6] S. J. Dickinson, D. Metaxas and A. Pentland (1997). The Role of Model-Based Segmentation in the Recovery of Volumetric Parts from Range Data, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, no. 3, pp. 259-267.
- [7] S. Gold and A. Rangarajan (1996). A Graduated Assignment Algorithm for Graph Matching, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 18, no. 4, pp. 377– 388.
- [8] B. K. P Horn (1984) B. K. P. Horn, “Extended Gaussian Images”, Proc. Of the IEEE, pp. 1671-1686.
- [9] A. Johnson and M. Hebert (1998) Efficient Multiple Model Recognition in Cluttered 3D Scenes, In Proc. IEEE Conference on Computer Vision and Pattern Recognition.
- [10] Kim and A. C. Kak, (1991) 3D Object Recognition Using Bipartite Matching Embedded in Discrete Relaxation, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 13, no. 3, pp. 224–251.
- [11] J. J. Koenderink and A. J. Doorn (1992). Surface shape and curvature scale, Image Vis. Comput., vol. 10, no. 8, pp. 557–565.
- [12] Y. Lamdan and H. J. Wolfson (1988) Geometric hashing: A general and efficient model-based recognition scheme, In Proceedings of the Second International Conference on Computer Vision, pages 238-249.

- [13] D. G. Lowe (2004) “Distinctive image features from scale-invariant keypoints”, International Journal of Computer Vision, vol. 60, no. 2, pp. 91-110.
- [14] B. Luo and E. R. Hancock (2001) Structural Matching Using the EM Algorithm and Singular Value Decomposition,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 23, no. 10, pp. 1120–1136.
- [15] K. Mikolajczyk and C. Schmid (2004) Scale and affine invariant interest point detectors”, International Journal of Computer Vision, vol. 60, no. 1.
- [16] M. Pelillo, K. Siddiqi, and S. Zucker (1999) ‘Matching Hierarchical Structures Using Association Graphs,’ IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 21, no. 11, pp. 1105–1120.
- [17] L. G. Shapiro and R. M. Haralick (1981) Structural Descriptions and Inexact Matching, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 3, no. 5, pp. 504–519.
- [18] A. Shokoufandeh, D. Macrini, S. Dickinson, K. Siddiqi, S.W. Zucker (2005) Indexing Hierarchical Structures Using Graph Spectra, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 27, no. 7, pp. 1125-1140.
- [19] A. Shokoufandeh, S. Dickinson, C. Johnsson, L. Bretzner, and T. Lindeberg (2002) On the Representation and Matching of Qualitative Shape at Multiple Scales, Proc., 7th European Conf. on Computer Vision, vol. 3, pp. 759–775, 2002.
- [20] K. Siddiqi, A. Shokoufandeh, S. Dickinson, and S. Zucker (1999) Shock Graphs and Shape Matching, Int. J. Computer Vision, vol. 35, no. 1, pp. 13–32, Nov 1999, doi:10.1023/A:1008102926703.
- [21] F. Stein and G. Medioni (1991) Structural Hashing: Efficient Three Dimensional Object Recognition, IEEE Conf. Computer Vision and Pattern Recognition vol. 3-6 pp. 244 – 250.
- [22] Stuttgart Range Image Database, <http://range.informatik.uni-stuttgart.de/>
- [23] H. J. Wolfson, and I. Rigoutsos (1997). Geometric Hashing: An Overview, IEEE Computational Science and Engineering, 4(4), 10-21.
- [24] J. Worthington and E. R. Hancock (2001) Object Recognition Using Shape-from-shading, IEEE Trans. Patterns Analysis Machine Intelligence, vol. 23, no. 5, pp.535-542.
- [25] Wikipedia “View of the planes establishing the main curvatures on a minimal surface” http://commons.wikimedia.org/wiki/File:Minimal_surface_curvature_planes-fr.svg
- [26] E. Akagündüz and I. Ulusoy (2007). 3D Object Representation Using Transform and Scale Invariant Features, Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, pp. 1-8.

- [27] Günter Hetzel, Bastian Leibe, Paul Levi, Bernt Schiele (2001). 3D object recognition from range images using local feature histograms, Proceedings of CVPR 2001, pp. 394-399.

- [28] Xinju Li, Igor Guskov (2007). 3D object recognition from range images using pyramid matching, ICCV 2007, pp. 1-6.

APPENDIX A

USER MANUAL OF THE SOFTWARE

The software used in this thesis is a dialog based MFC application. Figure 57 shows the primary screen for the program.

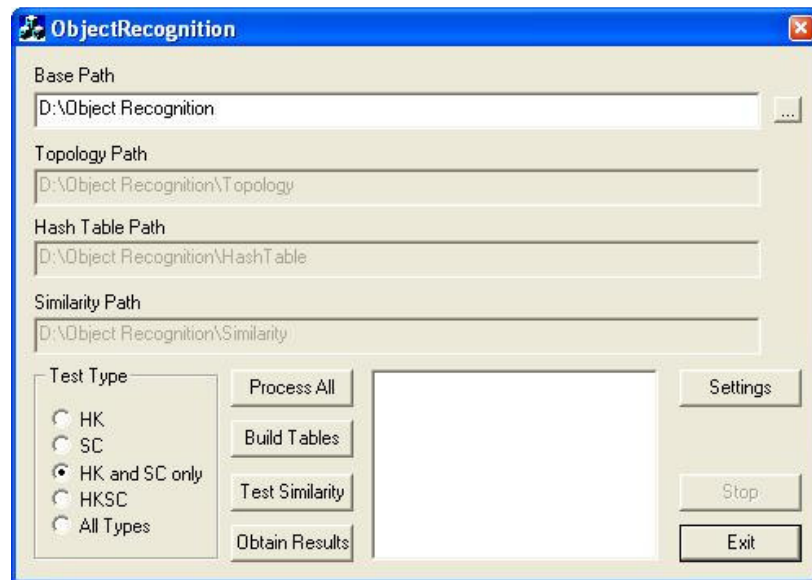


Figure 57: Primary screen of the program

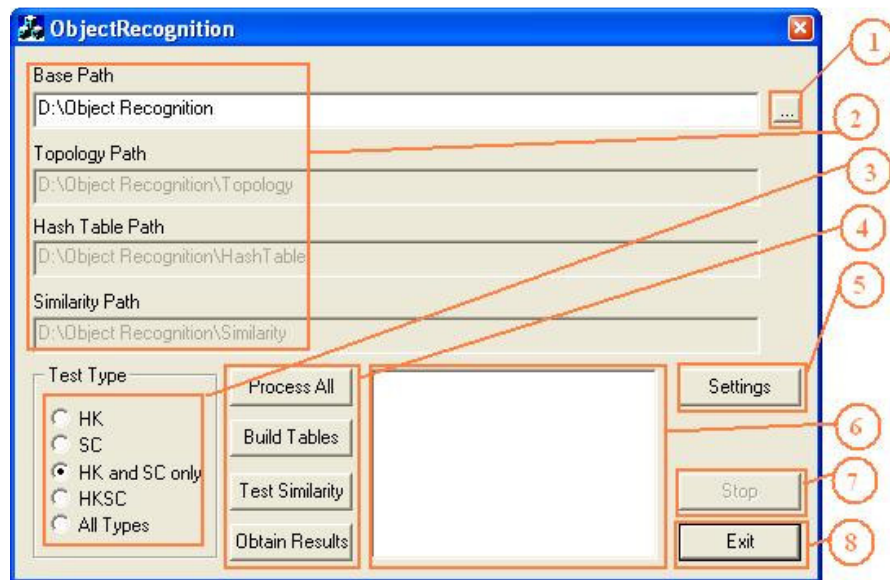


Figure 58: Functionality of primary screen elements

Functionalities of the numbers given in Figure 58 are given below:

- 1- This button is to choose the base path location with a user friendly interface.
- 2- These four edit boxes display the paths that will be used by the program. “Base Path” is editable but other path values cannot be edited and will be changed with the change of “Base Path” value.
- 3- This part is “Test Type” selection part. According to the selection, user defines which feature methods will be tested. There are 3 feature methods. These are “HK”, “SC” and “HKSC”. File extensions define the feature method for “Topology”, “HashTable” and “Similarity” files. (Note that only “HK” version is used on this thesis)
- 4- This part has process related buttons. Each button executes an operation and outputs to related files in folder “%BasePath%\Outputs”.
 - “Build Tables” button builds hash tables for features in “Topology Path”.
 - “Test Similarity” runs the test for files in “Hash Table Path”.

- “Obtain Results” button checks the similarity values in “Similarity Path” and then acquires correctness of the test, Hinton diagram and rank histogram.
- “Process All” button executes “Build Tables”, “Test Similarity” and “Obtain Results” one by one.

5- This button opens the “Settings” window in Figure 59.

6- This panel shows the log of the program.

7- This button stops any processes that are running at that moment.

8- This button exits the program.

The settings of the test can be changed from the “Settings” window of the program. Functionalities of the “Settings” window in Figure 60 are given below:

1- This part is to choose which feature types to be used.

2- If “General Max” is not checked this part is used to input feature number to be used for related feature type number.

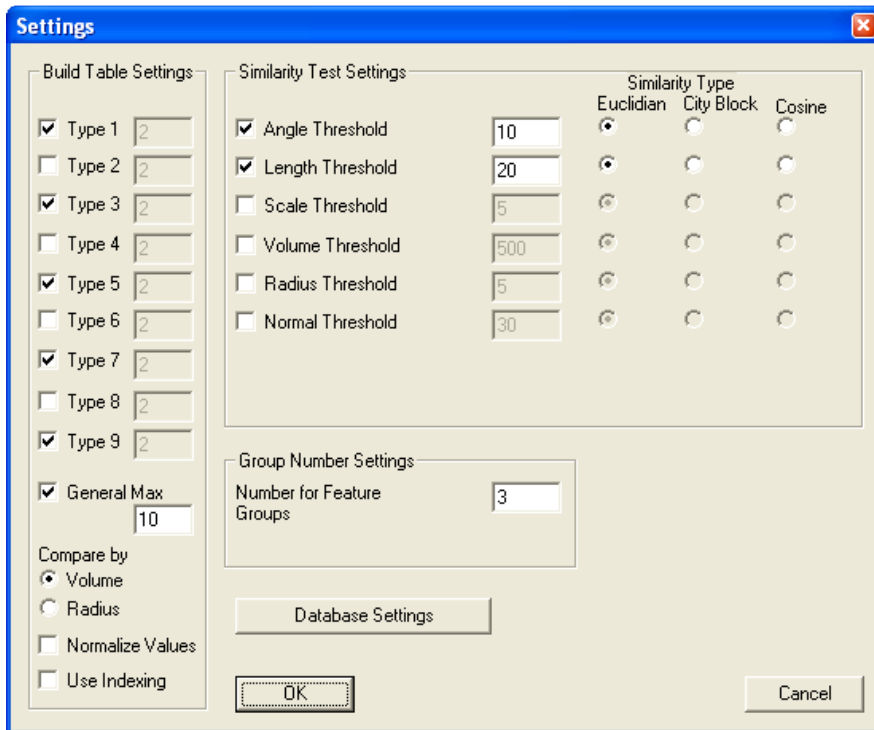


Figure 59: Settings window

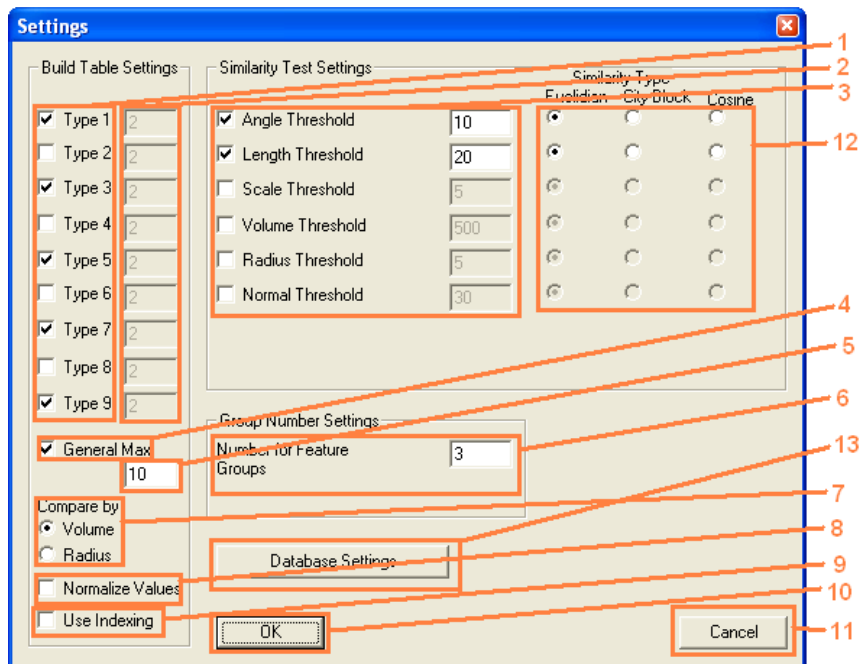


Figure 60: Functionality of settings window

- 3- This part is to choose which feature vector values to be used and define threshold values.
- 4- Enabling this states that feature elimination will be only according to the volume or radius values for the selected types. If this option is disabled each type elimination will be according to the numbers in part 2.
- 5- This number is used when “General Max” is enabled. This number defines the maximum number of features to be used.
- 6- This number shows feature grouping number.
- 7- This part defines if the features will be ordered by their volume or radius.
- 8- Enabling this makes the testing scale invariant by dividing length, volume and radius volumes with the first value in the vector and subtracting scale values with first value in the vector.
- 9- Enabling this makes the system to use indexing method explained in Section 4.3.
- 10- This button saves settings and exits “Settings” window.
- 11- This button discards any changes in settings and exits “Settings” window.
- 12- These radio buttons are used to choose similarity settings.
- 13- This button opens database setting dialog enabling to input database settings.

A proper application for the program should be as follows:

- Open “Object Recognition” program.
- Choose “Base Path”.
- Choose “Test Type”.

- Press “Settings” button.
- Set changes and press “Ok”.
- Press “Process All” button.

Note that “Build Tables”, “Test Similarity” and “Obtain Results” buttons are not necessary but they help the user to separately execute the steps of the process.

APPENDIX B

DEFINITION OF SOURCE CODE

Definition of source code is given in Table 14.

Table 14: Definition of Source Code

Class	Method(Function)	Explanation
BuildTables	BuildTables()	Class for building hash tables. Stores required data for hash table operations.
	Void Build(string type)	Builds the hash table for given type.
	Void SetBasePath(string p)	Sets the base path and output file paths of the hash tables.
	Void SetTypes(bool *t, int *max, bool totMax, int totalMax, bool comp, int nofg, bool norm)	Sets the required hash table settings.
	Void SetThreshold(SimilarityThreshold t)	Sets threshold values to define which of feature descriptors will be used. Also threshold values are required for indexing with feature descriptors.
	Void CreateTable(string inputFileName, string outputFileName)	Creates hash tables for indexing method.
	Oid BuildTablesWithOrder(int *iReturnValue, FeatureTable *tables, Feature *features, int iFeatureNumber, int iFeatureGroups, bool cbv, int *iStart, int iCallNumber, int iPrev);	Groups the features with the given number. Also this method orders the features according to their type, volume or radius
Cmatrix	Cmatrix()	This class is required for matrix

Table 14: Definition of Source Code

Class	Method(Function)	Explanation
		operations for Hinton diagram output.
	Void assign(char* row, char* column)	For input row and column names increases that row and column value by one. If there is no named row or column with input names new row and/or column will be created.
Correctness	Correctness()	Class for computing correctness of the test results.
	Void FindCorrectness(string type)	Computes the test results by comparing testing objects names and most similar objects name for all objects.
	Void SetBasePath(string p)	Sets the base path of the testing folder.
	Int CheckCorrect(string inputFileName)	Computes the test results by comparing testing objects names and most similar objects name for input similarity file.
CreateHistogram	CreateHistogram()	Class for creating rank histogram.
	Void Create(string type)	Creates the rank histogram.
	Void SetBasePath(string p)	Sets the base path of the testing folder.
	Int CheckCorrectValue(string inputFileName, char* table, char* test)	Finds for the similarity file where the matching object name exists. Returns 1 if test was successful
CreateMatrix	CreateMatrix()	Class for creating Hinton diagram.
	Void Create(string type)	Creates the Hinton diagram.
	Void SetBasePath(string p)	Sets the base path of the testing folder.
	Bool CheckCorrect(string inputFileName, char* table, char*	Gets the test and most similar training objects name and

Table 14: Definition of Source Code

Class	Method(Function)	Explanation
	test)	increases the value in matrix for related spot.
Feature	Feature()	Class for features.
	Void initialize()	Initialize its attributes.
	Void AssignFeature(Feature &feature)	Sets the feature with the input feature.
FeatureTable	FeatureTable()	Class for feature vectors.
	Void SetNumberFeatureGroup(int numberfeature)	Sets the feature numbers in each group to be used
	void AssignTable(FeatureTable &table)	Sets the feature vector with the input feature vector.
	Void initialize()	Initialize its attributes.
	Bool CheckSimilarity(FeatureTable &table, SimilarityThreshold threshold)	Checks the similarity of two feature vectors according to the given threshold values.
SimilarityThreshold	SimilarityThreshold()	This class includes threshold values and which feature vector values will be used.
TestSimilarity	TestSimilarity()	Class for testing process.
	Void Recognize(string type)	Executes the testing operation.
	Void SetBasePath(string p)	Sets the base path of the testing folder.
	Void SetThreshold(SimilarityThreshold t)	Sets threshold values and which feature vector values will be used.
	Void SetTypes(int nofg)	Sets the feature numbers in each group.
	Int CheckSimilarity(string testFileName, string tableFileName)	Finds the similarity for two objects without indexing method.
-	void TransverseDirectory(string path, list<FILELIST>& theList)	This function retrieves the folder and files for the selected path.
ObjectRecognition	-	This class is used for MFC application
ObjectRecognitionDlg	-	This class is used for MFC

Table 14: Definition of Source Code

Class	Method(Function)	Explanation
		application
DatabaseConnection	DatabaseConnection()	Class for accessing database
	void Initialize(char* hst, char* usr, char* pss, char* dbn, int prt)	Initializes MySQL database connection
	bool Connect()	Connects to the database
	bool Query(char* sql)	Sends query to the MySQL database
	bool GetRow()	Gets row from the returned query
	void Disconnect()	Disconnects MySQL database

APPENDIX C

GENERAL INFORMATION ABOUT FILE STRUCTURE

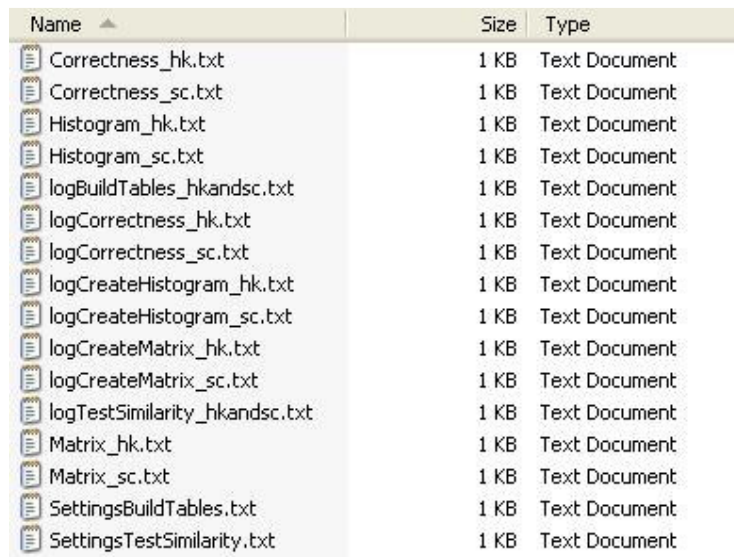
All of the related folders related with the tests need to be found in “Base Path”. The folders that need to be found are as follows:

➤ HashTable:

In this folder Hash Table database is stored. This folder includes two subfolders. Subfolder is “Testing”. “Testing” folder includes files for testing images and their feature vectors.

➤ Outputs:

This folder is where output files are stored. Output files are shown as in Figure 61.



Name	Size	Type
Correctness_hk.txt	1 KB	Text Document
Correctness_sc.txt	1 KB	Text Document
Histogram_hk.txt	1 KB	Text Document
Histogram_sc.txt	1 KB	Text Document
logBuildTables_hkandsc.txt	1 KB	Text Document
logCorrectness_hk.txt	1 KB	Text Document
logCorrectness_sc.txt	1 KB	Text Document
logCreateHistogram_hk.txt	1 KB	Text Document
logCreateHistogram_sc.txt	1 KB	Text Document
logCreateMatrix_hk.txt	1 KB	Text Document
logCreateMatrix_sc.txt	1 KB	Text Document
logTestSimilarity_hkandsc.txt	1 KB	Text Document
Matrix_hk.txt	1 KB	Text Document
Matrix_sc.txt	1 KB	Text Document
SettingsBuildTables.txt	1 KB	Text Document
SettingsTestSimilarity.txt	1 KB	Text Document

Figure 61: Output files for tests

In output files following items are stored:

- Log files: These files include the execution times of the processes.
- Correctness: These files include correctness rate and counts for similarity tests.
- Histogram: These files include rank histograms for similarity tests.
- Matrix: These files include Hinton diagram in matrix format.
- Setting files: These files include settings for hash table creation and threshold values for similarity tests.

➤ Similarity:

In this folder raw results for testing files exist.

➤ Topology:

In this folder features are stored for feature vector extraction. This folder has two subfolders. These are “Training” and “Testing” folders. “Training” and “Testing” folders have separate folders for object names. Inside the object folders features of object poses are written.

For any possible errors, program itself creates missing folders. So these folders are not necessary to exist. However program needs features to implement hashing operations. Also program needs hash tables to test for objects. In addition to these Similarity files need to exist to have clear output files. Because of these for a full process “Topology folder and its contents need to be found.

Figure 62 shows a sample of file structure.

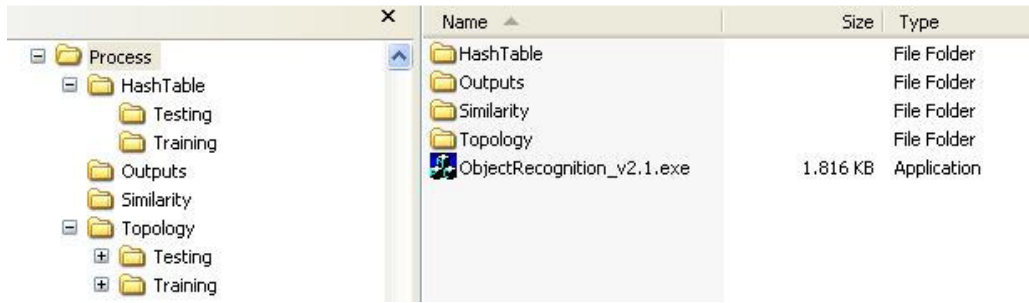


Figure 62: A sample file structure

Figure 63 shows “Topology” folder content and the training objects.

Name	Size	Type
agfa		File Folder
Auto		File Folder
balljoint		File Folder
banana		File Folder
bunny		File Folder
carrier		File Folder
chicken		File Folder
club		File Folder
copter		File Folder
deoflach		File Folder
deorund		File Folder
dino		File Folder
dinopet		File Folder
dragon		File Folder
Ente		File Folder
female		File Folder
fighter		File Folder
hasi		File Folder
hip		File Folder
hub		File Folder
igea		File Folder
isis		File Folder
knot		File Folder
kroete		File Folder
liberty		File Folder
machine		File Folder
manta		File Folder
mole		File Folder
pedaltopf		File Folder
pitbull		File Folder
pitt		File Folder
porsche		File Folder
rocker		File Folder
Schwein		File Folder
screwdriver		File Folder
scull		File Folder
seahorse		File Folder
Sfb		File Folder
teeth		File Folder
Turbine		File Folder
vette		File Folder
watch		File Folder

Figure 63: Topology folder content