STEGANOGRAPHY THROUGH PERSPECTIVE INVARIANCE

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

YAĞIZ YAŞAROĞLU

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2012

Approval of the thesis:

STEGANOGRAPHY THROUGH PERSPECTIVE INVARIANCE

submitted by **YAĞIZ YAŞAROĞLU** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy 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 Engineering**

Prof. Dr. A. Aydın Alatan _____
Supervisor, **Electrical and Electronics Engineering Dept., METU**

Examining Committee Members:

Assist. Prof. Dr. Afşar Saranlı
Electrical and Electronics Engineering Dept., METU _____

Prof. Dr. A. Aydın Alatan
Electrical and Electronics Engineering Dept., METU _____

Assoc. Prof. Dr. Çağatay Candan
Electrical and Electronics Engineering Dept., METU _____

Assist. Prof. Dr. Pınar Duygulu Şahin
Computer Engineering Dept., Bilkent University _____

Assoc. Prof. Dr. İsmail Avcıbaş
Electrical and Electronics Engineering Dept., Turgut Özal University _____

Date: _____

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: YAĞIZ YAŞAROĞLU

Signature :

ABSTRACT

STEGANOGRAPHY THROUGH PERSPECTIVE INVARIANCE

Yaşaroğlu, Yağız
Ph.D., Department of Electrical and Electronics Engineering
Supervisor : Prof. Dr. A. Aydın Alatan

September 2012, 104 pages

A novel approach for watermarking of 3D models is introduced, for which data is embedded into 3D models, whereas extracted from their projected 2D visual or 2D-plus-depth representations. Such a watermarking system is valuable, since most of the 3D content is being consumed as 2D visual data. Apart from the efficiency of embedding data into 3D models before generation of arbitrary 2D projections, in some use cases, such as free viewpoint video or computer games, 2D content has to be rendered at the client, where watermarking is less secure. In order to achieve this aim, 3D-2D perspective projection invariants, as well as 3D projective invariants are used and utilization of such invariants enables the method to be independent of the viewpoint from which 2D representations are generated. The first method proposed employs a perspective projection invariant to extract hidden data from an arbitrary 2D view of a watermarked 3D model. Data is encoded in the relative positions of six interest points, selection of which requires minimal criteria. Two main problems for such a watermarking system are identified as noise sensitivity of the invariant and repeatability of the interest point detection. By optimizing an objective function considering this sensitivity, the optimal 3D interest point displacements are obtained. Performance of the proposed system is evaluated through simulations on polygonal 3D mesh models and the results strongly indicate that perspective invariant-based watermarking is feasible. As an extenstion for 2D plus depth

representation of 3D models, data embedded in 3D models is also detected by combining information in 2D views and range data by utilizing another projective invariant. Finally, the problem of repeatable interest point detection that remain detectable after data embedding, is also examined and a novel method to identify such repeatable interest points is presented. The proposed methods indicate a new direction in watermarking research.

Keywords: 3D watermarking, steganography, projective invariants

ÖZ

İZDÜŞÜMSEL DEĞİŞMEZLİK İLE BİLGİ SAKLAMA

Yaşaroğlu, Yağız
Doktora, Elektrik Elektronik Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. A. Aydın Alatan

Eylül 2012, 104 sayfa

Üç boyutlu (3B) damgalamaya, 3B modellere gömülen verinin iki boyutlu (2B) ve iki boyut artı derinlik gösterimlerinden çıkarıldığı yeni bir yaklaşım sunulmaktadır. Bu 3B damgalama sistemi, 3B içeriğin büyük kısmı 2B gösterimlerinden tüketildiği için değerlidir. Verinin 3B modellere 2B görünümler üretilmeden önce gömülmesinin, üretilen 2B görünümleri damgalamaktan daha verimli olmasının yanında, serbest bakış açılı video ve bilgisayar oyunları gibi kimi uygulamalarda 2B içerik, damgalamanın daha az güvenli olduğu kullanıcı cihazlarında üretilmektedir. Önerilen yöntemde 3B-2B perspektif izdüşüm değişmezleri ile 3B izdüşüm değişmezleri kullanılarak yöntemin 2B içeriğin üretildiği bakış açısından bağımsız olması sağlanmıştır. İlk metod, bir perspektif izdüşüm değişmezi kullanarak, damgayı bir 3B modelin herhangi bir 2B görünümünden sezmektedir. Veri, neredeyse tamamen serbest olarak belirlenen altı ilgi noktasının göreli konumlarına saklanır. İzdüşüm değişmezleri kullanan 3B-2B damgalama sistemlerinin performansını etkileyen iki ana unsur, değişmezin gürültü hassasiyeti ve tekrar edebilir ilgi noktası tespiti olarak belirlenmiştir. Bu hassasiyet göz önüne alınarak oluşturulan bir kriterin optimize edilmesi ile veri saklanması için optimal 3B ilgi noktası konumları tespit edilmiştir. Önerilen sistemin performansı 3B poligon modeller üzerinde yapılan deneyler ile sınanmış, elde edilen sonuçlar izdüşümsel değişmezlerle damgalamanın uygulanabilir olduğunu göstermiştir. Ayrıca, 2B artı derinlik gösterimlerine

uygulanabilir şekilde, 3B modellere gömülen veri bir 3B izdüşümsel değişmez kullanılarak, 2B görünümlerden ve derinlik verisinden elde edilen bilginin birleştirilmesi ile de sezilmektedir. Son olarak, veri gömüldükten sonra tespit edilebilir kalacak şekilde, tekrar edilebilir ilgi noktası tespiti problemi ele alınmış ve bu ilgi noktalarının bulunması için yeni bir yöntem önerilmiştir. Sunulan tüm yöntemler bilgi saklama araştırmalarında yeni bir yöne işaret etmektedir.

Anahtar Kelimeler: 3B damgalama, bilgi saklama, izdüşümsel değişmezler

To my wife.

ACKNOWLEDGMENTS

First, I would like to thank my supervisor Prof. Dr. A. Aydın Alatan for accepting me back after the break in my studies, and supporting me through my “impossible mission”. It is difficult to find words that can express my gratitude for his guidance, advice, criticism, encouragements and insight throughout the research.

I am also deeply indebted to the members of my thesis committee, Assist. Prof. Dr. Afşar Saranlı, Assoc. Prof. Dr. Çağatay Candan, Assist. Prof. Dr. Pınar Duygulu Şahin, Assoc. Prof. Dr. İsmail Avcıbaş, and Assist. Prof. Dr. Hüsrev Taha Sencar for their support and suggestions which improved the quality of my work.

I would also like to thank my partners at Mobilus, Ekin Dino who did my job while I studied, and Özgür Deniz Önür who encouraged with a positive example, and proofread this dissertation.

Last but not the least; I would like to thank my family and my wife Ayşegül for loving and supporting me in all my endeavors.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

The volume of digital 3D content is increasing dramatically. Once confined to research laboratories and computer aided design stations, 3D content is starting to enter our living rooms, through computers, video game consoles and 3D televisions. Even mobile devices have 3D capabilities that enable ubiquitous access to 3D content.

Similar to what had happened with digital audio, image and video content, this proliferation of 3D content inevitably resulted in a problem for content owners: Copyright protection. Watermarking can be used to solve this problem for 3D digital content similar to other digital content [4]. In addition to copyright protection, 3D content can also be used as an additional communication channel transmitting invisible information embedded on itself.

Many schemes for 3D watermarking has appeared in the literature [5]. Vast majority of these works deal with embedding secret data in, and extracting hidden data from 3D models. On the other hand, there is an increase in the amount of 3D content that is distributed and consumed in 2D. This creates a use case for 3D-2D watermarking, that is, extracting embedded data in 3D models from 2D views.

1.1 Digital Watermarking

Digital watermarking is described as embedding secret information in digital content so that it can be detected in a copy of the content as long as the fidelity of the copy to the original is above a certain level. Embedded information can be used to demonstrate ownership, track a particular copy of the content, ensure fidelity, and even as a secondary or covert communication channel [6, 4].

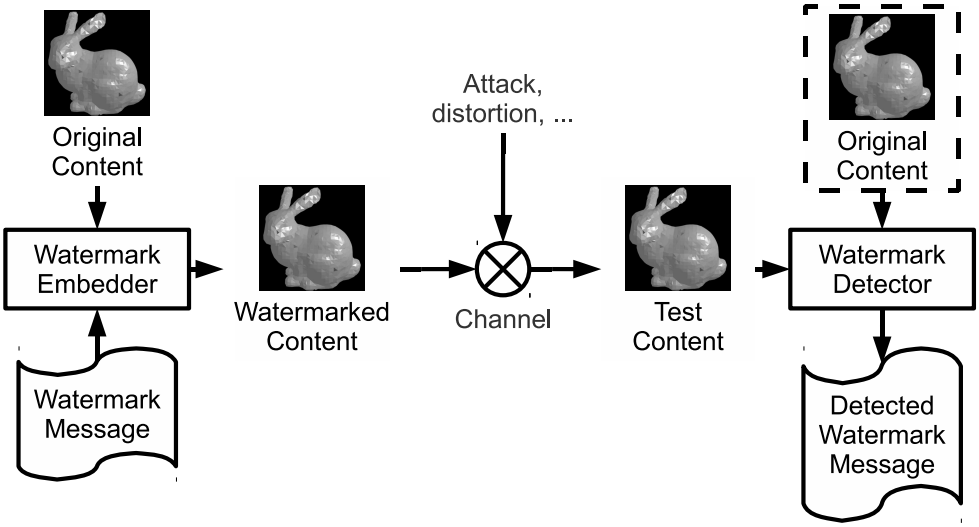


Figure 1.1: A simple watermarking system.

A simple watermarking scheme is given in Figure 1.1. Secret information is embedded in the original content in the embedder to obtain watermarked content. Watermarked content is then delivered to consumers through a channel. Various attacks, degradations, distortions might be applied to watermarked content in the channel. Detector tries to detect embedded secret information from received and possibly distorted watermarked content. If the original content is used during detection, this is a *non-blind* or *informed* watermarking scheme. Otherwise, this watermarking scheme is said to be *blind* or *oblivious* [4].

1.1.1 Applications and Requirements

Embedding data in content has a broad range of applications. *Owner identification*, in which user embeds secret or public information in content is the obvious application. Embedded information can be used to prove ownership in case of a dispute, or to publish copyright information [4].

Broadcast monitoring is another area where watermarking can be employed. Identifying information is embedded in the content, so that devices monitoring broadcasting channels can determine whether the watermarked content has been broadcasted. Advertisers and content creators, like musicians and movie producers, are typical users of these applications [4, 2].

Content distributors may want to track an illegal copy of their content to its source. This can be achieved by a *fingerprinting* system. In fingerprinting, each distributed copy of content is embedded with a unique identifier [4, 7].

Watermarking can also be used to determine *authenticity* of digital content. Digital content is very easy to manipulate. When content is manipulated, a fragile watermark embedded in the content will change along with the content and will provide information about which part of the content has been altered [8].

Finally, watermarking can be used as a secondary communication channel. Secret information can be embedded into digital content imperceptibly, providing a *covert communication* channel with low probability of detection [4].

Digital watermarking systems are evaluated according to several criteria. The prominent ones are visibility of the watermark (or imperceptibility), data payload or bit-rate of the watermarking system, if the watermarking system is blind or informed, robustness against attacks and false positive rate of the watermark detection process [4, 5]. Computational complexity and speed of the system, and ability to integrate with existing encoding and decoding processes are other practical properties that can be used to evaluate a watermarking system [2].

In evaluating a digital watermarking system, relative importance of these properties, and occasionally even definitions of these properties vary according to application [4]. An authentication system will call for a less robust watermark than an owner identification. A covert communication application will have to be more imperceptible than a copyright notice watermark, and so on.

Among these properties, capacity, visibility, and robustness turn out to be more important when common watermarking applications are considered [2, 4, 5, 6]:

- **Visibility/Imperceptibility**: The amount of distortion caused by embedding the watermark. Most watermarking systems aim to be as imperceptible as possible to keep watermarked content as close to original as possible. Objective distortion measures, such as signal to watermark ratio, or subjective experiments can be used to measure imperceptibility of a watermark [4].

- **Robustness**: Ability of the watermarking system to resist attacks in the transmission

channel. These attacks are signal processing operations that are characteristic of the channel [2]. They include distortion or noise caused during normal transmission of content, such as image and video coding or 3D to 2D perspective projection. On the other hand, malicious behavior by a user, such as resizing or cropping an image, and modifying intensity values, are also considered attacks. [4].

- **Capacity**: The amount of information that can be embedded in the content. Also called the bit-rate of the system, or data payload; this property is usually controlled by the expected robustness and imperceptibility requirements [4].

Depending on the application, a watermarking system has to make a trade of between these three requirements. Capacity increase through embedding more data, or robustness increase by increasing watermark strength will result in lower imperceptibility. In addition, capacity is inversely related with robustness. That is, for a given amount of distortion in the content, if more bits are embedded in the watermark, it will be harder to detect the watermark without causing errors. This can be taught of as each visibility level having a certain amount of available "space" to hide data. If more of that space is used to hide the same data strongly to increase robustness, capacity decreases. On the other hand, if more of available space is used to embed new data, data cannot be embedded as strongly, decreasing robustness.

The specific application scenario determines the trade off between conflicting properties. A watermarking system should balance these requirements carefully in order to be useful [4].

1.2 Motivation

Many watermarking methods that operate on audio, image and video signals have been developed [4]. On the other hand, 3D watermarking research, which aims to embed secret information in representations of 3D content, is not as mature. Research in 3D watermarking is mostly concerned with 3D-3D watermarking, that is, embedding data in 3D content and extracting data from 3D content [5]. Today there is an increasing amount of data created in 3D that is being consumed in 2D. Movies with CGI content, or computer games being two prominent examples. 3D-3D watermarking fails to address this concern. Therefore, there is a need for watermarking applications that are able to extract data from 2D views of 3D content.

2D-2D watermarking can be used to embed data in 3D content after 2D views are generated. Consider the block diagram below (Figure 1.2), which depicts a simple 2D-2D watermarking system for computer generated video.

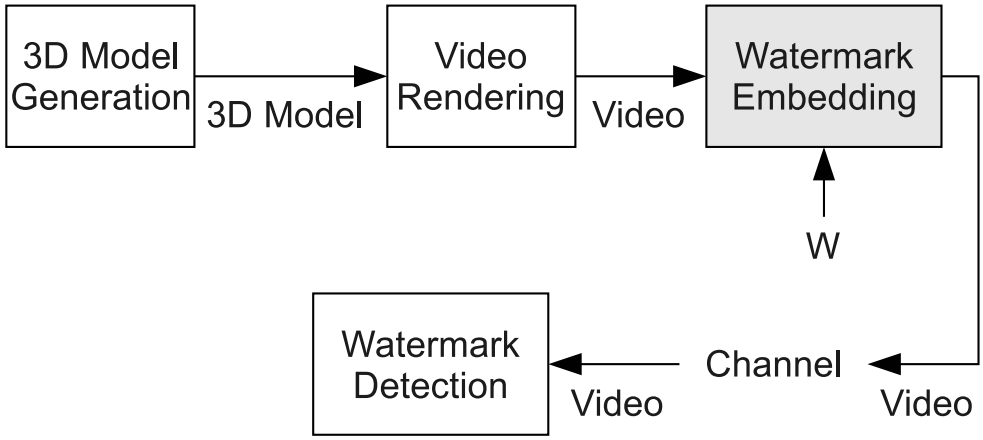


Figure 1.2: 2D-2D watermarking applied to computer generated video.

Notice that data is embedded after video is rendered. In this case, any of the standard video watermarking methods can be used.

Another alternative is to use 3D-2D watermarking methods. These methods embed data in 3D, but extract it from 2D views. Block diagram given below shows the same scenario as Figure 1.2, but this time 3D-2D watermarking is used.

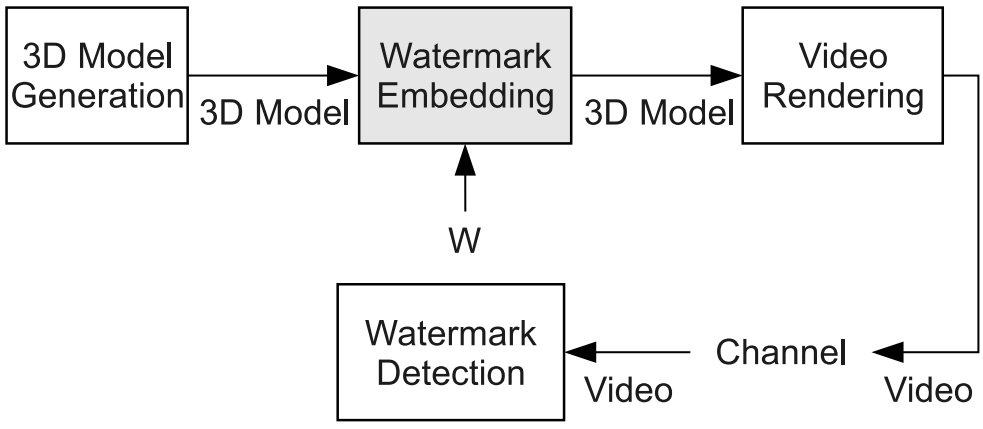


Figure 1.3: 3D-2D watermarking applied to computer generated video.

This time watermark is embedded in the 3D model, video of watermarked model is rendered

from numerous viewpoints, and subsequently watermark is detected from the arbitrarily rendered views. A 3D-2D watermarking scheme would be computationally less costly to embed than 2D-2D watermarking schemes, since data is embedded only once; on the model rather than generated views.

Noise characteristics of 3D-2D watermarking schemes are different than 2D-2D watermarking. In 3D-2D watermarking data is embedded in geometric primitives such as vertices and edges of a 3D mesh. Rendered videos of the 3D mesh will contain a different kind of distortion than, for example, transform domain watermarked videos. This difference can be beneficial in certain applications, especially if perceptible distortion can be kept at low levels.

There are also applications in which data that is created in 3D reaches the consumer in 3D, whereas the end product is in 2D. 3D model libraries used in computer aided design is one such application. An architect uses a 3D-2D watermarked 3D model in his plans. The rendered plans are in 2D, but the watermark should be still readable.

Similarly, in some applications such as computer games or free viewpoint TV, the last rendering step to 2D is done in the recipient, either from full 3D models, or other representations of 3D scenes like video plus depth images [9]. In such applications, using 3D-2D watermarking is safer as well as less costly than watermarking in the client.

A watermarking system in which data embedded in 3D models is extracted from generated views would address the use cases mentioned above. The thesis of this dissertation is that such a system can be built using 3D and 2D projective invariants, and 3D to 2D perspective invariants.

1.3 Scope of the Thesis

This dissertation deals with 3D-2D watermarking, a sub-class of 3D watermarking. The aim of this dissertation is to demonstrate a novel approach to embed data in 3D polygonal mesh models using invariant relations, in such a way that it is possible to extract hidden data from arbitrary 2D and 2D plus depth (2.5D) representations of the model. Presented method for 3D-2D watermarking utilizes a perspective projection invariant relation, derivation of which is provided. 3D-2.5D watermarking method utilizes a 3D invariant. Both methods' perfor-

mances are evaluated utilizing simulations on models.

A review of the background subjects relevant to dissertation is also given. Among them are 2D image and depth image forming from 3D models, 2D and 3D projective geometry and projective invariants.

The problem of repeatable interest point detection in the context of viewpoint independent watermarking systems is investigated. Requirements for such interest points are presented. A method that aims to satisfy the requirements is given, and tested in simulations.

1.4 Main Contributions

The primary contribution of this dissertation is devising a new approach to 3D-2D watermarking. Utilizing geometric invariants in 3D-2D watermarking is a promising area that is yet insufficiently explored. The following are the contributions of the dissertation in this area.

A 3D-2D watermarking system utilizing perspective invariants is proposed. Although the utilized perspective invariant is shown to be noise intolerant, simulations demonstrate performance of the system to be very promising, even with a heuristics based data embedding method. When an optimal data embedding method is utilized performance increases even more indicating that 3D-2D watermarking methods based on perspective invariants is feasible.

A 3D-2.5D watermarking system utilizing basic 3D projective invariants is proposed. This system combines information from 2D images and depth maps of a 3D model to extract embedded data. Experiments performed on models show that this approach to watermarking is promising.

Finally, the problem of finding repeatable interest points for viewpoint independent watermarking is investigated. A procedure aiming to address requirements for such interest points is presented. A number of interest point detectors are evaluated for fitness to the problem. Feasibility of the system is demonstrated on a realistic model.

1.5 Outline of the Thesis

Chapter 2: Background topics related to the dissertation, such as 2D and 3D projective geometry and invariants, hierarchy of transformations, 2D and depth image formation from 3D models, are reviewed. Literature about 3D-3D watermarking, 3D-2D watermarking, and projective invariants is briefly discussed. An introduction to applying perspective invariants to the problem of 3D-2D watermarking with invariants is presented.

Chapter 3: A 3D-2D watermarking method utilizing a perspective invariant relation is introduced. Primary difficulties in such a method are identified. Behavior of the invariant relation is briefly examined. Optimal and heuristics based data embedding methods are developed. Performance of the method is evaluated by performing simulations on synthetic data and 3D models.

Chapter 4: A 3D-2.5D watermarking method utilizing a projective invariant is introduced. Optimal and heuristics based data embedding methods are developed. Performance of the method is evaluated by performing simulations on synthetic data and 3D models.

Chapter 5: The problem of identifying repeatable interest points in 3D-2D and 3D-2.5D watermarking systems with projective invariants is investigated. A procedure to identify repeatable interest points is proposed. Various interest point detection algorithms are evaluated for fitness to the problem. Feasibility of the suggested method is evaluated by performing simulations on a realistic, textured model.

Chapter 6: Summarizes the dissertation and its main contributions, concludes the dissertation with a list of areas for future research.

CHAPTER 2

BACKGROUND TOPICS

This chapter provides a review of background topics related to the dissertation. 2D and 3D projective geometry and transformations are related to projective invariants, and are discussed in the following section. Hierarchy of transformations and examples to invariants are also provided.

The next section introduces 2D image formation and depth image formation from 3D models. First the pinhole camera model is given, and then, through step by step generalization, a model for a practical camera is obtained. Depth image formation process is described in detail and is shown to be a projective transformation.

A brief review of literature related to 3D-3D watermarking, 3D-2D watermarking and perspective invariants is presented, proceeded by an introduction to applying perspective invariants to the problem of 3D-2D watermarking.

2.1 Review of Projective Geometry, Transformations and Invariants

Projective invariants are a central concept of this dissertation. A review of projective geometry, homogeneous coordinates, transformations and invariants are presented in this section. Throughout this chapter, italic uppercase letters such as "A" represent matrices, bold lowercase letters and symbols such as "$\mathbf{a}, \mathbf{0}$" represent vectors, and italic small letters represent scalars.

In the rest of this section, first the two dimensional projective geometry and transformations are explained. The differences from three dimensional case is then laid out.

2.1.1 Planar Projective Geometry and Transformations

A point (x, y) in $\mathbb{R}^2$ is represented in *homogeneous coordinates* as a 3x1 vector by adding a third coordinate 1: $(x, y, 1)^T$. In homogeneous coordinates, the set of vectors $(kx,\ ky,\ k)^T$ for non-zero values of k all represent the point (x, y) in $\mathbb{R}^2$. An arbitrary homogeneous vector of the form $\mathbf{x} = (x_1, x_2, x_3)^T$ where $x_3 \neq 0$ represents the point $(x_1/x_3,\ x_2/x_3)$ on the plane.

If the third coordinate x_3 is non-zero, the homogeneous vector represents a finite point in $\mathbb{R}^2$. Whereas points with $x_3 = 0$ are points at infinity. The set of all points represented by homogeneous vectors, representing both finite and infinite points, is called the projective space and denoted as $\mathbb{P}^2$ in two dimensions.

A *planar projective transformation* is an invertible mapping $h(x)$ from $\mathbb{P}^2$ to itself such that three points x_1, x_2 and x_3 lie on the same line if and only if $h(x_1), h(x_2)$ and $h(x_3)$ are also linear [10]. In other words, under a projective transformation, lines are mapped to lines. Other names for a projective transform are *collinearity, projectivity* and *homography*.

A planar projective transform is a linear transformation on 3x1 homogeneous vectors represented by a non-singular 3x3 matrix:

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}. \tag{2.1}$$

Same relation is concisely written as $\mathbf{x}' = H\mathbf{x}$ where H is non-singular, and thus invertible. Note that multiplying H with a scalar does not affect the projective transform. Thus a planar projective transform has eight degrees of freedom [10].

2.1.2 Hierarchy of 2D Transformations

Special cases of the projective transform are investigated in this section, starting form the most specific case. Examples for invariants are also provided along with special case transforms. An *invariant* is an element or a quantity of a geometric configuration that remains unchanged under a transform [11]. A simple invariant example is given below.

Length as an invariant under rotation Rotation in the plane by an angle θ is defined in in homogeneous coordinates as

$$R = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}. \tag{2.2}$$

The distance between two 2D points $\mathbf{x_1}$ and $\mathbf{x_2}$ in homogeneous coordinates , that is $(\mathbf{x_1} - \mathbf{x_2})^2$, is invariant under rotation. If $\mathbf{x'_i} = R \cdot \mathbf{x_i},\ i \in \{1, 2\}$, the distance between $\mathbf{x'_1}$ and $\mathbf{x'_2}$ is given as

$$\begin{aligned} L(\mathbf{x'_1}, \mathbf{x'_2}) &= (\mathbf{x'_1} - \mathbf{x'_2})^2 \\ &= (R \cdot \mathbf{x_1} - R \cdot \mathbf{x_2})^2 \\ &= ((x_{1x} - x_{2x})\cos\theta - (x_{1y} - x_{2y})\sin\theta)^2 + ((x_{1x} - x_{2x})\sin\theta + (x_{1y} - x_{2y})\cos\theta)^2 \\ &= (\mathbf{x_1} - \mathbf{x_2})^2 \\ &= L(\mathbf{x_1}, \mathbf{x_2}). \end{aligned} \tag{2.3}$$

That is, distance between two points do not change under rotation [11].

2.1.2.1 Isometric Transformation

An isometric transformation preserves the Euclidean distance of points. It is defined as

$$\begin{pmatrix} x'_1 \\ x'_2 \\ 1 \end{pmatrix} = \begin{pmatrix} \epsilon\cos\theta & -\sin\theta & t_1 \\ \epsilon\sin\theta & \cos\theta & t_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ 1 \end{pmatrix}, \tag{2.4}$$

where $\epsilon = \pm 1$ [10]. If $\epsilon = 1$, the transform is said to be orientation-preserving and is a *Euclidean transformation*. Euclidean transformations represent motion of a rigid body. They can be divided into a 2x2 rotation matrix R and a 2x1 translation vector $\mathbf{t}$ as given below. $\mathbf{0}$ is 2x1 zero vector.

$$\mathbf{x}' = \begin{pmatrix} R & \mathbf{t} \\ \mathbf{0}^T & 1 \end{pmatrix} \mathbf{x} \tag{2.5}$$

There are three degrees of freedom in a planar Euclidean transform; two for translation, one for rotation. Distance between points, angle between lines and area are examples for invariants of isometric transformations [11].

2.1.2.2 Similarity Transformation

A similarity transformation is an isotropic scaling (scaling of all coordinates with the same factor) in addition to an isometric transformation. Algebraic definition of a similarity transform is given as:

$$\begin{pmatrix} x'_1 \\ x'_2 \\ 1 \end{pmatrix} = \begin{pmatrix} s\epsilon\cos\theta & -s\sin\theta & t_1 \\ s\epsilon\sin\theta & s\cos\theta & t_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ 1 \end{pmatrix}. \tag{2.6}$$

In the case where there is no reflection, that is $\epsilon = 1$, this relation can be concisely written as

$$\mathbf{x}' = \begin{pmatrix} sR & \mathbf{t} \\ \mathbf{0}^T & 1 \end{pmatrix} \mathbf{x}, \tag{2.7}$$

where s represents the isotropic scaling [10].

A similarity transform preserves the shape. It has four degrees of freedom. Angles between lines are invariant under similarity transformation, thus parallel lines are mapped to parallel lines. Although lengths are not invariant, ratio of two lengths is invariant. Similarly, ratio of areas are also invariant [11].

2.1.2.3 Affine Transformation

An *affine transformation*, or *affinity*, is a non-singular linear transformation combined with a translation. Its matrix representation is given below.

$$\begin{pmatrix} x'_1 \\ x'_2 \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & t_1 \\ a_{21} & a_{22} & t_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ 1 \end{pmatrix} \tag{2.8}$$

Or briefly,

$$\mathbf{x}' = \begin{pmatrix} A & \mathbf{t} \\ \mathbf{0}^T & 1 \end{pmatrix} \mathbf{x}, \tag{2.9}$$

where A is a non-singular 2x2 matrix [10]. A can be decomposed as

$$A = R(\theta)R(-\phi)DR(\phi), \tag{2.10}$$

$$D = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}. \tag{2.11}$$

That is, A is composed of a rotation by ϕ, followed by scaling with λ_1 and λ_2 in rotated x and y directions, rotation by $-\phi$ and finally another rotation by θ. The only difference from similarity transform is the presence of non-isotropic scaling by λ_1 and λ_2. This scaling oriented at angle ϕ is the essence of affine transformation, and it provides two additional degrees of freedom: ϕ and λ_1/λ_2 [10].

Some important invariants of affine transformation are parallel lines, ratio of lengths of parallel lines and ratio of areas [11].

2.1.2.4 Projective Transformation

Planar projective transformation, defined in Section 2.1.1, is a non-singular linear transformation of *homogeneous* coordinates. It generalizes affine transform, which is a non-singular linear transformation of *inhomogeneous* coordinates. In block matrix form, planar projective transform is defined as

$$\mathbf{x}' = \begin{pmatrix} A & \mathbf{t} \\ \mathbf{v}^T & v \end{pmatrix} \mathbf{x} \tag{2.12}$$

where $\mathbf{v} = (v_1, v_2)^T$. As mentioned in 2.1.1, a projective transform in the plane has eight degrees of freedom [10].

An important invariant of projective transformation is the *cross ratio* of four collinear points described below. Other invariant examples are collinearity and concurrency [11].

Cross Ratio The cross ratio is the classic example of projective invariants [10]. Although the ratio of lengths on a line is not projective invariant, ratio of ratio of lengths is. Cross ratio is defined on four collinear points x_1, x_2, x_3, x_4 as [12]:

$$cr(x_1, x_2, x_3, x_4) = \frac{|x_1 x_3||x_2 x_4|}{|x_1 x_4||x_2 x_3|} \tag{2.13}$$

Figure 2.1: Four collinear points.

2.1.3 Projective Geometry and Transformations in 3D

Analogous to the case in 2D, a point (x, y, z) in $\mathbb{R}^2$ is represented as the 4x1 vector $(kx, ky, kz, k)^T$ where k is non-zero. Any 4x1 vector $\mathbf{x} = (x_1, x_2, x_3, x_4)^T$ is a point in $\mathbb{P}^3$. If x_4 is non zero, $\mathbf{x}$ represents a finite point in $\mathbb{R}^3$. Otherwise, $\mathbf{x}$ represents a point at infinity.

A *projective transformation in 3D* is a mapping from $\mathbb{P}^3$ to itself, and is defined analogous to the planar case as an invertible linear transformation on 4x1 homogeneous vectors represented by a non-singular 4x4 matrix, as given below:

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \\ x'_4 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} & h_{14} \\ h_{21} & h_{22} & h_{23} & h_{24} \\ h_{31} & h_{32} & h_{33} & h_{34} \\ h_{41} & h_{42} & h_{43} & h_{44} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}, \tag{2.14}$$

briefly written as $\mathbf{x}' = H\mathbf{x}$ where H is non-singular. 3D projective transformation has 15 degrees of freedom, since multiplying H with a scalar does not affect the transformation [10].

2.1.4 Hierarchy of 3D Transformations

Hierarchy of 3D transformations is similar to 2D transformation hierarchy. The transforms, their matrix forms and examples for invariants are given in the Table 2.1 below. Invariants mentioned in Section 2.1.2 for 2D transformations are also invariants for 3D transformations [10].

Table 2.1: Hierarchy of 3D transformations.

Transformation	Matrix	Invariants
Euclidean 6 dof	$\begin{pmatrix} R & \mathbf{t} \\ \mathbf{0}^T & 1 \end{pmatrix}$	Volume.
Similarity 7 dof	$\begin{pmatrix} sR & \mathbf{t} \\ \mathbf{0}^T & 1 \end{pmatrix}$	The absolute conic.
Affine 12 dof	$\begin{pmatrix} A & \mathbf{t} \\ \mathbf{0}^T & 1 \end{pmatrix}$	Parallelism of planes, ratio of volumes, centroids.
Projective 15 dof	$\begin{pmatrix} A & \mathbf{t} \\ \mathbf{v}^T & v \end{pmatrix}$	Intersection and tangency of surfaces in contact.

In table 2.1, R is a 3x3 3D rotation matrix, $\mathbf{t}$ is a 3x1 translation vector, $\mathbf{0}$ is the 3x1 zero vector, s is the isotropic scaling factor, $\mathbf{A}$ is a non-singular 3x3 matrix, $\mathbf{v}$ is a general 3x1 vector, and v is a scalar.

2.2 3D Steganography and Invariants

This dissertation investigates the feasibility of a view independent 3D-2D watermarking method. It is proposed that a satisfactory 3D-2D watermarking method can be built using projective invariants.

Image formation process, that is producing 2D views of 3D models, usually involves perspective projection, similar to the projection in a pin hole camera. The next section provides an overview of the 3D to 2D image formation process.

In some cases depth information obtained from 3D models are available, in addition to 2D images. Obtaining the depth information from a model can be thought of as a mapping from 3D to 3D. This mapping is a projective transformation. This process is outlined in Section 2.2.2.

In the final section, role of invariants in 3D-2D and 3D-2.5D watermarking contexts is examined.

2.2.1 2D Image Formation

A camera is a mapping from world (3D) coordinates to image (2D) coordinates. The simplest model of a camera is the pin hole model, which is depicted in Image 2.2.

The basic pin hole camera has a *camera center* or *optical center*, and an *image plane* or *focal plane*. The line from camera center that is perpendicular to the image plane is called the *principal axis*, and the plane that passes through camera center parallel to image plane is called the *principal plane*. The point that the principal axis intersects the image plane is called the *principal point*.

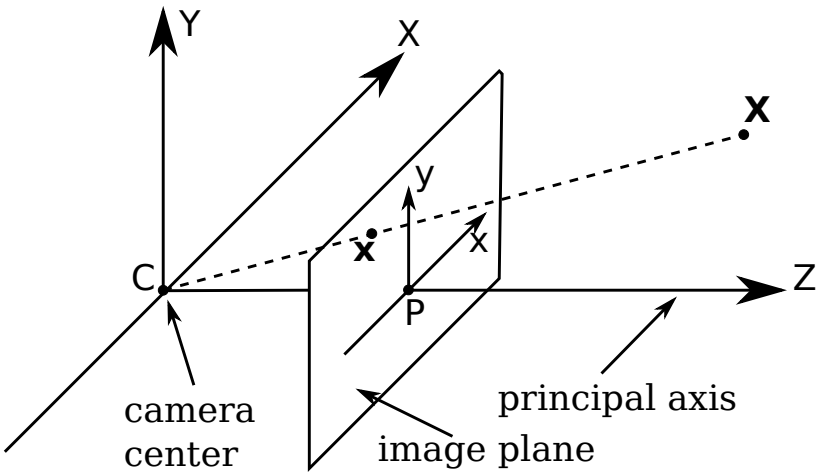


Figure 2.2: Pin hole camera model.

In Figure 2.2, the camera center is at the origin of the coordinate system, and image plane is the $z = f$ plane. Points in 3D space are projected onto the two dimensional image plane.

Any point $\mathbf{x} = (x, y, z)^T$ is mapped to the point at which the line from $\mathbf{x}$ to camera center intersects the image plane. The coordinates of this point is $(fx/z, fy/z, f)^T$, as can be calculated using similar triangles. If we drop the third coordinate;

$$(x, y, z)^T \rightarrow (fx/z, fy/z)^T. \tag{2.15}$$

This operation is a mapping from $\mathbb{R}^3$ to $\mathbb{R}^2$. If **x** is expressed in homogeneous coordinates, this mapping can be depicted in matrix algebra as

$$\begin{pmatrix} fx/z \\ fy/z \\ 1 \end{pmatrix} \cong \begin{pmatrix} fx \\ fy \\ z \end{pmatrix} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}. \tag{2.16}$$

This multiplication can be written more briefly as $\mathbf{x} = P\mathbf{X}$, where the bold uppercase **X** represents the 3D point in homogeneous coordinates, and bold lowercase **x** represents 2D point in homogeneous coordinates. The matrix P is said to be the *camera projection matrix* for the pin hole model. Note that P can be written as

$$P = K\left(I \;\middle|\; 0\right), \tag{2.17}$$

where $K = diag(f, f, 1)$ is called the calibration matrix of the camera. Pin hole camera model is too simple to account for image formation in general. Following sections generalize this model to arrive at a more practical projective camera model.

2.2.1.1 Principal Point Offset

In most cases, points on the image plane are not defined according to the center of the image but to one corner of the image. This offset can be represented by a change in the matrix K, where p_x and p_y are the offset amounts in horizontal and vertical axes, as given below.

$$\begin{pmatrix} fx + zp_x \\ fy + zp_y \\ z \end{pmatrix} = \begin{pmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{2.18}$$

$$\mathbf{x} = \begin{pmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{pmatrix} \Big(I \;\Big|\; 0 \Big) \mathbf{X} \tag{2.19}$$

$$\mathbf{x} = K \Big(I \;\Big|\; 0 \Big) \mathbf{X} \tag{2.20}$$

2.2.1.2 Camera Rotation and Translation

In the above equations **X** is assumed to be in the camera coordinate frame. However, generally, **X** will be expressed in the world coordinate frame and will need to be converted to camera frame coordinates. If $\tilde{\mathbf{X}}$ is defined as **X** in inhomogeneous coordinates, and $\tilde{\mathbf{X}}_{\mathbf{cam}}$ is defined as $\tilde{\mathbf{X}}$ expressed in camera coordinate frame, $\tilde{\mathbf{X}}_{\mathbf{cam}} = R \cdot (\tilde{\mathbf{X}} - \tilde{\mathbf{C}})$, where R is the 3x3 matrix representing camera coordinate axis orientation, and $\tilde{\mathbf{C}}$ is the camera center in inhomogeneous world coordinates.

When combined with conversion of **X** to camera coordinates, equation 2.20 becomes

$$\mathbf{x} = KR \Big(I \;\Big|\; -\tilde{\mathbf{C}} \Big) \mathbf{X}, \text{ or equivalently} \tag{2.21}$$

$$\mathbf{x} = K \Big(R \;\Big|\; \mathbf{t} \Big) \mathbf{X}, \tag{2.22}$$

where $\mathbf{t} = -R\tilde{\mathbf{C}}$.

2.2.1.3 Non-Square Pixels and Skew

In the models derived above it was assumed that pixels in the imaging device, for example a CCD sensor, are unit squares. This is rarely the case. In practice size of the pixels decrease with image resolution, and they may not be square.

Skew is used to describe the case when x and y axes of the image plane are not perpendicular. Skew may be caused by the imaging device, or it might be observed when an image is captured from another image of an object.

Calibration matrix modified to account for both of these generalizations is given below:

$$K = \begin{pmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{2.23}$$

In equation 2.23, $\alpha_x = fm_x$ and $\alpha_y = fm_y$ where m_x and m_y are the number of pixels per unit distance in x and y directions in image plane. Similarly $x_0 = p_x m_x$ and $y_0 = p_y m_y$ are coordinates of the principal point in terms of pixel dimensions. Scalar s is the skew parameter, which is usually zero.

The camera that is represented by the camera matrix

$$P = KR\left(I \;\middle|\; -\tilde{\mathbf{C}}\right) \tag{2.24}$$

where K is of the form 2.23 is called a *finite projective camera*. A finite projective camera has eleven degrees of freedom. Note that the 3x3 left hand side matrix of P, KR, is non-singular. Conversely, any 3x4 matrix having a non-singular 3x3 left side matrix is the camera matrix for a finite projective camera [10]. Indeed, P can be written as $P = (M \,|\, \mathbf{p_4}) = M(I \,|\, M^{-1} - \tilde{\mathbf{C}})$ where M is non-singular 3x3 and $\mathbf{p_4}$ is the fourth column of P. M can be decomposed into the upper triangular matrix K and orthogonal rotation matrix R using RQ matrix decomposition [10].

2.2.1.4 General Projective Camera

A camera with the camera matrix written as

$$P = \left(M \;\middle|\; \mathbf{p_4}\right), \tag{2.25}$$

where P is of rank 3, is a general projective camera. If M is non-singular, the camera is a finite camera. If M is singular, it is an infinite camera, that is its center is a infinity.

2.2.2 Depth Map Formation

A *depth map*, *depth image*, or *disparity map* is an image that has depth information as its pixel values. Typically, a depth map is obtained by calculating the distance of every 3D point $\mathbf{X}$ to the image plane along the principal axis, and scaling and quantizing the depth values so that the depth information can be represented as pixel intensity values. For example in Figure 2.3 the depth map is a monochromatic, 8-bit image [3]. Depth range is restricted between z_{near} and z_{far}. This range is linearly quantized between 0 and 255 to fit in 8-bit image depth, 0 depicting z_{far}.

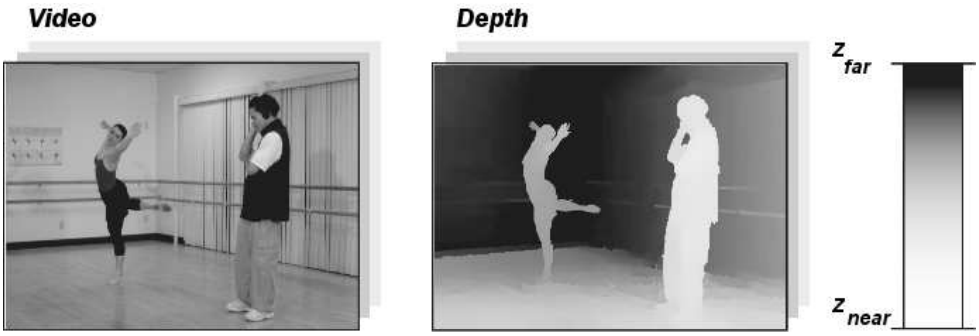


Figure 2.3: 3D data consisting of 2D video and depth map [3].

Depth structure of a scene obtained from depth map can be combined with the 2D image or video data to build a 3D-like representation, which is often called a 2.5D representation [3]. Depth and 2D information can be used to build a stereo image pair as in [13, 9]. Depth maps and videos from multiple cameras positioned to cover a range of viewpoints are used to render the scene from novel viewpoints in [14]. Multiple depth layers are used in Layered Depth Images [15] to provide free viewpoint video.

Depth map is aligned with the accompanying 2D view, that is, their samples at a given image coordinate correspond to the same 3D point. In a 2D view - depth map pair, depth map is obtained using the same projective camera transformation as the image[1]. For the purposes of

[1] This is true for rendered synthetic scenes. When imaging natural scenes depth map is obtained through different cameras. Even when this is the case 2D image and depth map have to be aligned before consumption.

this dissertation, it will be demonstrated that mapping of 3D point $\mathbf{X} = (x, y, z, 1)^T$ in world coordinates to depth map point $\mathbf{X_d} = (x', y', z', 1)^T$ is a projective transformation. That is,

$$\mathbf{X_d} = H_{depth} \cdot \mathbf{X} \tag{2.26}$$

where H_{depth} is a 4x4 non-singular matrix. In $\mathbf{X_d}$, x' and y' are the 2D image coordinates of $\mathbf{X}$ projected through the camera matrix, and z' is the depth value scaled between z_{near} and z_{far}. That is, if depth is represented with d,

$$z' = 255 \cdot (\frac{z_{far} - d}{z_{far} - z_{near}}) \tag{2.27}$$

In [10] depth of a 3D point $\mathbf{X}$ related to a camera with camera matrix $P = \left(M \quad \middle| \mathbf{p_4}\right)$ is given by

$$d = depth(\mathbf{X}, P) = \frac{sign(\det M)\omega}{\|\mathbf{m^3}\|}. \tag{2.28}$$

In this relation, $\|\mathbf{m^3}\|$ is the magnitude of third row of M, and $\omega = \mathbf{p}^3 \cdot \mathbf{X}$, where $\mathbf{p}^3$ denotes the third row of P.

If P is the 3x4 camera matrix of a finite camera, the following relation between 3D point $\mathbf{X}$ and 2D point $\mathbf{x}$ exists:

$$\mathbf{x} = P\mathbf{X}, \tag{2.29}$$

$$\begin{pmatrix} \omega x' \\ \omega y' \\ \omega \end{pmatrix} = P \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}. \tag{2.30}$$

The matrix T obtained by vertically concatenating P and transpose of the vector $\mathbf{j} = (0, 0, 0, 1)^T$ is non-singular, because left 3x3 sub-matrix of P, that is M, is non-singular.

$$T = \begin{pmatrix} P \\ \mathbf{j}^T \end{pmatrix} \tag{2.31}$$

When the world point $\mathbf{X} = (x, y, z, 1)^T$ is transformed using T,

$$\begin{pmatrix} \omega x' \\ \omega y' \\ \omega \\ 1 \end{pmatrix} = T \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}. \tag{2.32}$$

Multiplying both sides with

$$H = \begin{pmatrix} \frac{1}{\omega} & 0 & 0 & 0 \\ 0 & \frac{1}{\omega} & 0 & 0 \\ 0 & 0 & -\frac{255 \cdot k}{z_{far} - z_{near}} & -\frac{255 \cdot z_{far}}{z_{far} - z_{near}} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.33}$$

where $k = {}^{sign(\det M)}/_{\|\mathbf{m}^3\|}$ so that $k \cdot \omega = d$, we obtain

$$\begin{pmatrix} x' \\ y' \\ z' = 255 \cdot (\frac{z_{far} - d}{z_{far} - z_{near}}) \\ 1 \end{pmatrix} = H \cdot T \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{2.34}$$

which is the desired result. Since H is non-singular, $H_{depth} = H \cdot T$ is a projective transformation. Mapping 3D points to depth map points is a projective transformation.

2.3 Literature

2.3.1 3D Watermarking

3D watermarking is watermarking of a particular representation of a 3D scene. This representation can be a 3D representation such as a polygonal mesh, a point cloud, a parametric

surface, an implicit surface or a volumetric representation [16]. Watermark can also be embedded in 2D views obtained from 3D models. Table 2.2 lists 3D watermarking categories, and their protected components as given in [2].

Table 2.2: 3D Watermarking categories, and their protected components [1, 2].

	Protected Component
3D-3D Watermarking	Geometry representation
3D-2D Watermarking	Geometry or Texture of 3D Object
2D-2D Watermarking	Images of 3D Object

3D watermarking research mostly deals with 3D-3D watermarking. In a survey of existing 3D watermarking methods given in [5], dated 2008, only two of the 45 investigated algorithms are 3D-2D methods.

In the following sections, brief information about existing 3D-3D watermarking methods, and more detailed summaries of three 3D-2D watermarking methods found in the literature will be presented.

2.3.1.1 3D-3D Watermarking Literature

Research in this area deals with embedding data on the geometry of a 3D object, by making slight modifications. Research can be categorized into two: Spatial domain methods and transform domain methods.

Most spatial domain methods operate on 3D primitives like vertices, edges and surface normals.

For example in [17], the watermark is embedded in the radius component of vertices in a spherical coordinate system, center of which is the center of mass of the 3D object. Radius component of vertices are sampled into K uniform bins, where K is the desired bit capacity. Then, radius components of each bin are modified according to the binary watermark to be embedded, as given in the relation below:

$$r'_i = (1 + \alpha R_k) \times r_i \tag{2.35}$$

where R_k is -1 if watermark bit is 0, 1 otherwise. α determines the strength of the watermark. In order to detect the watermark, sample means prior to embedding are required.

The method in [18] uses the direction of normals of mesh triangles to embed data. The unit sphere is divided into regions. Every normal on the 3D object belongs to one of these regions. To embed data, normals of the model are altered, changing the dispersion of normals in a given region on the unit sphere. Reducing dispersion encodes a 0, increasing dispersion encodes a 1. The regions to embed data are determined by a secret key. This method is said to be resilient against 3D attacks like polygon reduction.

Another method, [19], embeds data in the distance between a vertex, and the barycenter of the vertex's neighbors. This value is called a 'normal'. To embed data, these normals are modified according to bits of the binary watermark. The 'normal' metric defined in this paper is invariant to affine transforms, thus the watermarking method is resilient to affine transformations of the 3D object.

Other spatial domain methods extract 2D information from 3D models and use image watermarking based algorithms. An example is [20], which computes a cylindrical depth map of a given 3D model. A cylindrical coordinate system is placed on the mesh, so that its origin coincides with the barycenter. The coordinate system is aligned with the principal axis of inertia of the object. A depth map is obtained by converting the cylindrical coordinates (r, θ, x) to a 2D image where x-axis is θ, y-axis is z, and image values are r. Resulting image is then watermarked using a DCT based algorithm, and changes are reflected back to the 3D model. The method is robust against mesh simplification and noise.

Transform domain methods embed the watermark into transform domain coefficients of 3D models. In [21], wavelet decomposition is applied to the 3D mesh. Watermark is then embedded in the wavelet coefficients of significant magnitude. Watermarked mesh is obtained by reconstruction from modified wavelet coefficients. A similar method based on wavelet decomposition of semi-regular meshes [22], embeds data in wavelet coefficients. Watermark is detected by computing the correlation coefficient between watermark and the mesh under inspection.

A comparison of spatial and transform domain methods is given in Table 2.3 below [2]:

Table 2.3: Comparison between spatial and transform domain 3D-3D watermarking methods [2].

	Pros	**Cons**
Spatial Domain Methods	Lower complexity, robustness against cropping	Difficulty in finding perceptually significant regions, weakness against local deformations
Transform Domain Methods	Robustness against compression and additive noise, good integration with visual perception	Higher computational cost, weakness against cropping

2.3.1.2 3D-2D Watermarking Literature

3D-2D watermarking is relatively unexplored in the literature. In one approach, watermark is embedded in the apparent contour of of 3D objects [23]. Data is embedded on textures of 3D objects in another method [24]. [2] uses a projective invariant to do 3D-2D watermarking. This dissertation builds on the idea presented in [2].

In [23], the watermark is embedded on the Fourier transform coefficients of the contour of silhouettes of a 3D model. A number of views are used to obtain contours, and the silhouettes are stored for use in detection. During detection from a 2D view, first the best silhouette that fits the view is detected. Then, the view's contour is extracted. Correlation of this contour with the watermark is thresholded to determine the watermark's existence. The method is robust against rotation/translation/scaling attacks.

In [24], watermark is embedded in the texture of 3D models, and is detected on 2D models after texture reconstruction. A robust watermarking algorithm designed for still images is utilized to embed data. The knowledge of the original 3D object is needed for watermark extraction, and the geometry and texture mapping function are required for the texture recovery step. Rendering parameters should also be known a priori. Most of the effort goes into reconstructing the texture, after which it is sufficient to do image watermark detection.

The final 3D-2D watermarking technique [2] uses the projective invariant five point cross ratio [25]. This projective invariant is related to cross ratio. Given five coplanar points (x_1, x_2, x_3, x_4, x_5 in Figure 2.4), one of the points is selected and connected to other four with lines. Intersection points of any arbitrary line with the four lines are marked (y_1, y_2, y_3, y_4 in

Figure 2.4). Cross ratio of these four points, which is given below, is the projective invariant:

$$cr(y_1, y_2, y_3, y_4) = \frac{|y_1 y_3||y_2 y_4|}{|y_1 y_4||y_2 y_3|} \tag{2.36}$$

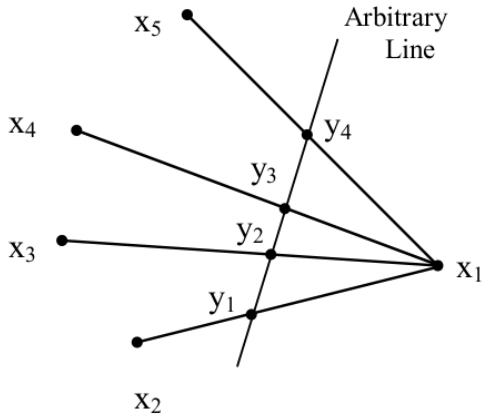


Figure 2.4: Five point cross ratio.

Advantage of five point cross ratio over four point cross ratio is that it is less constrained. Finding five coplanar points on a 3D model is much easier than finding four collinear points.

The method applied in [2] to embed watermark on a 3D model is outlined below:

1. Identify five coplanar feature points on the 3D object,
2. Change the location of the feature points on the model in order to change the cross-ratio to a value that will represent the embedded information.

In order to detect the watermark, the following steps must be carried out:

1. Identify the same five co-planar points in the 2D image or video.
2. Calculate the associated cross-ratio to extract the information due to the value of this ratio.

Identification of five coplanar feature points is an important aspect of the algorithm. The feature points need to be identified on 2D views in the detector. Therefore, they are identified on 2D views instead of 3D in watermark embedder. Identification process is given below:

1. Generate 2D views in a range of angles.
2. Use SIFT to identify candidate feature points on every view.
3. Candidate feature points that are available on all views are assumed to be stable points for the selected angle range.
4. Five coplanar points are found among the stable points.

An informed algorithm in which descriptors of the points on which watermark is embedded are known to decoder is suggested in [26].

This method [26] is the first 3D-2D watermarking method using projective invariants.

2.3.2 Perspective Invariants

A perspective invariant is an element or a quantity of a geometric configuration that remains constant under perspective projection from three dimensional space to two dimensional space. A brief discussion about the relationship between projective invariants and perspective invariants is given in Section 2.4.

There has been considerable amount of interest in perspective invariants. Interest has been mainly related to object identification, but there has been at least one study [2] that used projective invariants in 3D watermarking.

Perspective invariants operate on various 3D structures. Point sets [27, 25], line sets [25], curves [28], conics [11] are among the structures investigated for projective invariance.

It is known that there is no general case perspective invariant on arbitrary 3D point sets [29]. However, by constraining the sets of points we can obtain usable perspective invariants. Two examples were given above, the cross ratio [12] required four collinear points, and five point cross ratio [25] required five coplanar points. Zhu et al. derived a perspective invariant for six points on two adjacent planes [30]. There are more constrained, and more specific perspective invariants as well (for example points lying at the vertices of a polyhedron [31]). However, constraining the point set limits the usefulness of the perspective invariant.

Notably, YuanBin et al. provided a general perspective invariant which has minimal constraints [27]. This perspective invariant operates on 6 3D points of which 4 do not lie on the

same plane. Also, of the 6 corresponding 2D points, 3 should not lie on the same line. Yuan-Bin et al. then derive two examples of more constrained perspective invariants, one of which is identical to the one Zhu et al. has provided [30].

2.4 Perspective Invariants in 3D-2D Watermarking

In this section, an introduction to applying perspective invariants to the problem of 3D-2D watermarking is presented.

Given a 3D model M, and its 2D counterpart m obtained by a projective camera, a non-trivial invariant relation between M and m can be used for watermarking. This 3D to 2D *perspective invariant relation* can be defined as $f(M,m) = 0$, given that $m = P \cdot M$, where P is any arbitrary projective camera matrix. The invariant relation f establishes a relationship between 3D models and 2D views independent of the camera matrix. Given many possible source models $\{M_j\}$ and a 2D model m produced from one of the source models, one can decide the source model by finding M_j that satisfies $f(M_j, m) = 0$.

Such a relationship provides a valuable tool for 3D-2D watermarking. Data is hidden in a model by modifying M and producing n different versions $\{M_i\}$. Assuming k 2D views are generated from each M_i using $P_1, P_2, ..., P_k$, producing $m_{11} = P_1 \cdot M_1, m_{12} = P_2 \cdot M_1, ... for M_1; m_{21} = P_1 \cdot M_2, m_{22} = P_2 \cdot M_2, ... for M_2$, and so on.

If the invariant relation $f(x,y)$ is available, then

$$f(M_1, m_{1j}) = 0 \ for \ j \ in \ [1,k], \tag{2.37}$$

$$f(M_1, m_{2j}) \neq 0 \ for \ j \ in \ [1,k], \tag{2.38}$$

$$f(M_2, m_{1j}) \neq 0 \ for \ j \ in \ [1,k], \tag{2.39}$$

$$f(M_2, m_{2j}) = 0 \ for \ j \ in \ [1,k]. \tag{2.40}$$

Using this information; given the set of possible models ($\{M_1, M_2, ..., M_n\}$), and any perspective projected view m_{ij}, the model m_{ij} is related to can be determined, which is the embedded information.

Note that, a projective invariant is constant under transformations between spaces of equal

dimensions. However, the *perspective invariant* $f(M,m) = 0$ is constant under perspective projection from a three dimensional space to the plane. Perspective invariant relation $f(M,m) = 0$ can be constructed from projective invariants. As an example, consider cross ratio given in Section 2.1.2. If the same four collinear points are identified in M and m, their cross ratios should be equal, since cross ratio is invariant under projective transformation, and perspective projection. Thus, the invariant relation can be described as:

$$f(M,m) = cr(M) - cr(m) \tag{2.41}$$

Another approach for developing this invariant relation is to derive a relationship between 3D and 2D projective invariants. If the model M has a set of 3D projective invariants given by α and the model's projected image m has a set of 2D projective invariants given by β, then if M is sufficiently complicated, the projective invariants will satisfy one or more non-trivial polynomial constraints $f(M,m)$, arising from the fact that m is a projection of M [32]. In [32], a 3D-2D invariant relation is derived between r points and $6-r$ lines, $r \notin \{0,4\}$. Another example to this approach is given in Chapter 3 [27].

Most such perspective invariant relations defined in the literature have constraints on their inputs which limit their usefulness in watermarking applications. For example, cross ratio referred to above requires input points to be collinear, which limits its practical applicability. There may not be sufficient four collinear points on a 3D model suitable for data hiding.

On the other hand, it is known that there is no general case 3D-2D perspective invariant relation on arbitrary 3D point sets from single images [26]. However, in [27] a general relationship on 6 3D points to their 2D counterparts is defined with minimal constraints; four of the points should be non-coplanar and three of the projected 2D points should be non-collinear. Chapter 3 of this dissertation investigates the feasibility of using the basic invariant relation relation described in [27] to build a simple 3D-2D steganography system.

CHAPTER 3

UTILIZATION OF PERSPECTIVE INVARIANTS IN 3D-2D WATERMARKING

It was demonstrated in Section 2.2 that it is possible to use projective invariants in 3D-2D watermarking. Feasibility of a projective invariant based watermarking system is demonstrated first on random point sets not attached to a 3D model. Then, experiments are conducted on simple 3D meshes. Performance of the system is evaluated by methods using heuristics and also optimization of an appropriate cost function.

In this chapter, watermark is embedded in 3D models represented as polygonal meshes, since polygonal meshes have widespread use and they can be more efficiently processed than all other 3D representations [16]. The following properties are desired in a perspective invariant relation to be used for 3D-2D watermarking on 3D polygonal meshes:

1. Perspective invariant should work on points, edges, faces or normals which are the features of 3D meshes. Invariants working on other geometric features, such as curves, conic sections and so on, are not suitable.

2. Perspective invariant should impose as minimum constraint on 3D mesh features to embed data in as possible. This constraint means that a larger number of features is usable for hiding data.

The following section introduces a perspective projection invariant that satisfies these two conditions.

3.1 Perspective Invariant Relation

YuanBin et al. [27] have derived a point based 3D to 2D perspective projection invariant relation that involves 3D and 2D projective invariants. This perspective invariant relationship has two constraints: Given 6 3D interest points $P_i,\ i = 1, ..., 6$ and their perspective projected 2D counterparts $p_i,\ i = 1, ..., 6$;

1. 3D points P_1, P_2, P_3, P_4 should be non-coplanar,

2. 2D points p_1, p_2, p_3 should be non-collinear.

These two constraints are weaker than the constraints on other perspective invariants [12, 25, 30, 31]. In fact, this perspective invariant relation has also been used to derive more constrained ones [27].

Perspective projection by a projective camera relation can be given as:

$$k_i \cdot \begin{pmatrix} u_i \\ v_i \\ 1 \end{pmatrix} = M \cdot \begin{pmatrix} x_i \\ y_i \\ z_i \\ 1 \end{pmatrix}, \tag{3.1}$$

where $P_i = (x_i, y_i, z_i, 1)^T$ and $p_i = (u_i, v_i, 1)^T$ in homogeneous coordinates, k_i are non zero scalars, M is the 3x4 projective camera matrix of rank 3 (identical to P in equation (2.25)) and $i = 1, 2, 3, 4, 5, 6$.

Since P_1, P_2, P_3 and P_4 represent four non-coplanar points in $\mathbb{R}^3$ they are linearly independent. On the other hand, P_i are members of the four dimensional homogeneous vector space, and no five points in a four dimensional space can be linearly independent. These two fact allow the following relations to be written [27]:

$$P_5 = \alpha_1 P_1 + \alpha_2 P_2 + \alpha_3 P_3 + \alpha_4 P_4, \tag{3.2}$$

$$P_6 = \beta_1 P_1 + \beta_2 P_2 + \beta_3 P_3 + \beta_4 P_4. \tag{3.3}$$

Since P_1, P_2, P_3 *and* P_4 are linearly independent, the coefficients α_i and β_i are unique. A

similar relation can be written for 2D points: defining p_4, p_5, p_6 in terms of p_1, p_2, p_3 with unique coefficients $\lambda_i, \rho_i, \tau_i$.

$$p_4 = \lambda_1 p_1 + \lambda_2 p_2 + \lambda_3 p_3, \tag{3.4}$$

$$p_5 = \rho_1 p_1 + \rho_2 p_2 + \rho_3 p_3, \tag{3.5}$$

$$p_6 = \tau_1 p_1 + \tau_2 p_2 + \tau_3 p_3. \tag{3.6}$$

Note that, since last (homogeneous) coordinates of P_i and p_i are all 1, $\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = 1$ and $\beta_1 + \beta_2 + \beta_3 + \beta_4 = 1$, also $\lambda_1 + \lambda_2 + \lambda_3 = 1$, $\rho_1 + \rho_2 + \rho_3 = 1$ and $\tau_1 + \tau_2 + \tau_3 = 1$.

If (3.2) and (3.3) are expanded by multiplying each side of the equations by perspective transformation matrix M and combining with the perspective transformation relation (3.1), the following equations are obtained [27]:

$$k_5 p_5 = k_1\alpha_1 p_1 + k_2\alpha_2 p_2 + k_3\alpha_3 p_3 + k_4\alpha_4 p_4, \tag{3.7}$$

$$k_6 p_6 = k_1\beta_1 p_1 + k_2\beta_2 p_2 + k_3\beta_3 p_3 + k_4\beta_4 p_4. \tag{3.8}$$

Substituting (3.4), (3.5), and (3.6) in the above equations (3.7) and (3.8),

$$k_5(\rho_1 p_1 + \rho_2 p_2 + \rho_3 p_3) = k_1\alpha_1 p_1 + k_2\alpha_2 p_2 + k_3\alpha_3 p_3 + k_4\alpha_4(\lambda_1 p_1 + \lambda_2 p_2 + \lambda_3 p_3), \tag{3.9}$$

$$k_6(\tau_1 p_1 + \tau_2 p_2 + \tau_3 p_3) = k_1\beta_1 p_1 + k_2\beta_2 p_2 + k_3\beta_3 p_3 + k_4\beta_4(\lambda_1 p_1 + \lambda_2 p_2 + \lambda_3 p_3). \tag{3.10}$$

Rearranging these equations and collecting terms in front of p_i,

$$(k_1\alpha_1 + k_4\alpha_4\lambda_1 - k_5\rho_1)p_1 + (k_2\alpha_2 + k_4\alpha_4\lambda_2 - k_5\rho_2)p_2 + (k_3\alpha_3 + k_4\alpha_4\lambda_3 - k_5\rho_3)p_3 = 0, \tag{3.11}$$

$$(k_1\beta_1 + k_4\beta_4\lambda_1 - k_6\tau_1)p_1 + (k_2\beta_2 + k_4\beta_4\lambda_2 - k_6\tau_2)p_2 + (k_3\beta_3 + k_4\beta_4\lambda_3 - k_6\tau_3)p_3 = 0. \quad (3.12)$$

Since $p_1, p_2,$ and p_3 are non-zero and not collinear, they are linearly independent. Thus, coefficients in the above equations are zero. This provides us with a set of linear equations [27]:

$$k_1\alpha_1 + k_4\alpha_4\lambda_1 - k_5\rho_1 = 0 \quad (3.13)$$

$$k_2\alpha_2 + k_4\alpha_4\lambda_2 - k_5\rho_2 = 0 \quad (3.14)$$

$$k_3\alpha_3 + k_4\alpha_4\lambda_3 - k_5\rho_3 = 0 \quad (3.15)$$

$$k_1\beta_1 + k_4\beta_4\lambda_1 - k_6\tau_1 = 0 \quad (3.16)$$

$$k_2\beta_2 + k_4\beta_4\lambda_2 - k_6\tau_2 = 0 \quad (3.17)$$

$$k_3\beta_3 + k_4\beta_4\lambda_3 - k_6\tau_3 = 0. \quad (3.18)$$

This set of linear equations can be written in the matrix form

$$\begin{pmatrix} \alpha_1 & 0 & 0 & \lambda_1\alpha_4 & -\rho_1 & 0 \\ 0 & \alpha_2 & 0 & \lambda_2\alpha_4 & -\rho_2 & 0 \\ 0 & 0 & \alpha_3 & \lambda_3\alpha_4 & -\rho_3 & 0 \\ \beta_1 & 0 & 0 & \lambda_1\beta_4 & -\tau_1 & 0 \\ 0 & \beta_2 & 0 & \lambda_2\beta_4 & -\tau_2 & 0 \\ 0 & 0 & \beta_3 & \lambda_3\beta_4 & -\tau_3 & 0 \end{pmatrix} \begin{pmatrix} k_1 \\ k_2 \\ k_3 \\ k_4 \\ k_5 \\ k_6 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \quad (3.19)$$

Since k_i are known to be non-zero, the set of linear equations (3.19) must have a non-trivial solution. This non-triviality implies that determinant of the coefficient matrix must be zero. The following relation is obtained, if this determinant is calculated and simplified through division by $\frac{\alpha_2\alpha_3\alpha_4\beta_1\beta_1}{\alpha_1}$:

$$\rho_3(\lambda_1\tau_2 - \lambda_2\tau_1)\frac{\alpha_1\beta_3\alpha_1\beta_4}{\alpha_3\beta_1\alpha_4\beta_1} - \rho_3(\lambda_1\tau_3 - \lambda_3\tau_1)\frac{\alpha_1\beta_2\alpha_1\beta_4}{\alpha_2\beta_1\alpha_4\beta_1} + \tau_1(\lambda_2\rho_3 - \lambda_3\rho_2)\frac{\alpha_1\beta_2\alpha_1\beta_3}{\alpha_2\beta_1\alpha_3\beta_1} + \rho_1(\lambda_2\tau_3 - \lambda_3\tau_2)\frac{\alpha_1\beta_4}{\alpha_4\beta_1} - \tau_2(\lambda_1\rho_3 - \lambda_3\rho_1)\frac{\alpha_1\beta_3}{\alpha_3\beta_1} + \tau_3(\lambda_1\rho_2 - \lambda_2\rho_1)\frac{\alpha_1\beta_2}{\alpha_2\beta_1} = 0. \quad (3.20)$$

Note that (3.20) provides a relationship between relative positions of 3D points P_i and 2D points p_i, which are expressed in the coefficients α_i, β_i and $\lambda_i, \rho_i, \tau_i$, respectively. Relation (3.20) is a 3D to 2D perspective invariant relation, since it does not contain any term related to the perspective projection matrix M [27]. This relation can be directly used in a 3D-2D watermarking system.

In [27], this relation is further simplified by grouping α_i, β_i and $\lambda_i, \rho_i, \tau_i$ coefficients as:

$$I_{12} = \frac{\alpha_1\beta_2}{\alpha_2\beta_1},\ I_{13} = \frac{\alpha_1\beta_3}{\alpha_3\beta_1},\ I_{14} = \frac{\alpha_1\beta_4}{\alpha_4\beta_1}, \tag{3.21a}$$

$$K_{12}^{\lambda\tau} = \frac{\lambda_1\tau_2}{\lambda_2\tau_1},\ K_{13}^{\lambda\tau} = \frac{\lambda_1\tau_3}{\lambda_3\tau_1},\ K_{12}^{\rho\tau} = \frac{\rho_1\tau_2}{\rho_2\tau_1},\ K_{13}^{\rho\tau} = \frac{\rho_1\tau_3}{\rho_3\tau_1},\ K_{32}^{\lambda\rho} = \frac{\lambda_3\rho_2}{\lambda_2\rho_3}. \tag{3.21b}$$

By this grouping, the relative positions of six 3D interest points produce three coefficients $I = \{I_{12}, I_{13}, I_{14}\}$. The relative positions of six 2D interest points produce five coefficients $K = \{K_{12}^{\lambda\tau}, K_{13}^{\lambda\tau}, K_{12}^{\rho\tau}, K_{13}^{\rho\tau}, K_{32}^{\lambda\rho}\}$ [27]. These coefficients are 3D projective invariants and 2D projective invariants, respectively [27, 33, 34].

The perspective invariant that is to be used in this chapter is the function f given below, which is simply (3.20) rewritten in terms of projective invariants in (3.21a) and (3.21b). Note that due to the relation $K_{32}^{\lambda\rho} = \frac{K_{12}^{\lambda\tau}K_{13}^{\rho\tau}}{K_{13}^{\lambda\tau}K_{12}^{\rho\tau}}$, $K_{32}^{\lambda\rho}$ does not appear in the below function.

$$\begin{aligned} f(I,K) = (K_{12}^{\lambda\tau} - 1)I_{13}I_{14} - \frac{K_{12}^{\lambda\tau}K_{13}^{\rho\tau}(K_{13}^{\lambda\tau} - 1)}{K_{13}^{\lambda\tau}K_{12}^{\rho\tau}}I_{12}I_{14} \\ + (1 - \frac{K_{12}^{\lambda\tau}K_{13}^{\rho\tau}}{K_{13}^{\lambda\tau}K_{12}^{\rho\tau}})I_{12}I_{13} + (K_{13}^{\rho\tau} - \frac{K_{12}^{\lambda\tau}K_{13}^{\rho\tau}}{K_{13}^{\lambda\tau}})I_{14} \\ - (K_{12}^{\lambda\tau}\frac{K_{12}^{\lambda\tau}K_{13}^{\rho\tau}}{K_{13}^{\lambda\tau}})I_{13} + (\frac{K_{12}^{\lambda\tau}K_{13}^{\rho\tau}}{K_{12}^{\rho\tau}} - K_{13}^{\rho\tau})I_{12} = 0 \end{aligned} \tag{3.22}$$

This perspective invariant relation is referred to as $f(I, K)$ in terms of projective invariants as in (3.22) rather than the relation in (3.20), although they are equivalent.

It should be noted that a very similar relationship is derived by Weiss and Ray using a different method [33, 34].

3.1.1 Preliminary Tests on the Perspective Invariant

Perspective invariant given in (3.22) is a function of seven variables (I_{ij} *and* K_{mn}^{pq}). Relative positions of 3D points P_i and 2D points p_i determine these variables. Positions of 2D points are in turn determined by positions of 3D points P_i and the perspective projection matrix M, although M does not appear in the final perspective invariant. Visualization of $f(I,K)$, and how it responds to changes in P_i and p_i, is challenging due to the interrelations between variables.

Figure 3.1 is presented as a preliminary analysis. Figure 3.1 is an image of $\log|f(I,K)|$ for fixed $P_1, P_2, P_3, P_4, P_5, P_6$ 3D vertex values and p_2, p_3, p_4, p_5, p_6 2D pixel values. The image is a partial evaluation of the function; only p_1 is kept variable. Its coordinates (x,y) are varied across the image surface so that at the top left corner p_1 is $(0,0)$ and at the bottom right corner it is $(800,600)$. In red regions the function is undefined. In other regions darker pixels denote smaller $\log|f(I,K)|$ values. Blue dots are actual positions of p_1, p_2, p_3, p_4, p_5, and p_6.

The thin dark line is composed of a series of local minima, about a pixel thick, as can be seen in the zoomed-in inset. Gray regions occupying much of the remaining space show that the function is mostly slow changing. Different configurations of P_i produce similar figures, with large areas of slow changing values and thin valleys of local minima.

These preliminary tests show that correctly detecting locations of watermarked 2D points in images is important. A one pixel error while detecting 2D point locations may make a large change in $|f(K,I)|$ value, depending on the direction of error.

This sensitivity to error is undesirable in 3D-2D watermarking. Due to this sensitivity to the localization of points, $|f(I,K)|$ might not approach to zero, causing a watermark to be missed.

Feasibility of this perspective invariant in 3D-2D watermarking is more thoroughly investigated in the rest of this chapter, by means of simulations on random point sets and 3D meshes.

3.2 3D-2D Watermarking with Perspective Invariant

Using the perspective invariant described in Section 3.1, data is embedded in a 3D mesh model by modifying positions of 6 interest points. Each modified model is used to convey

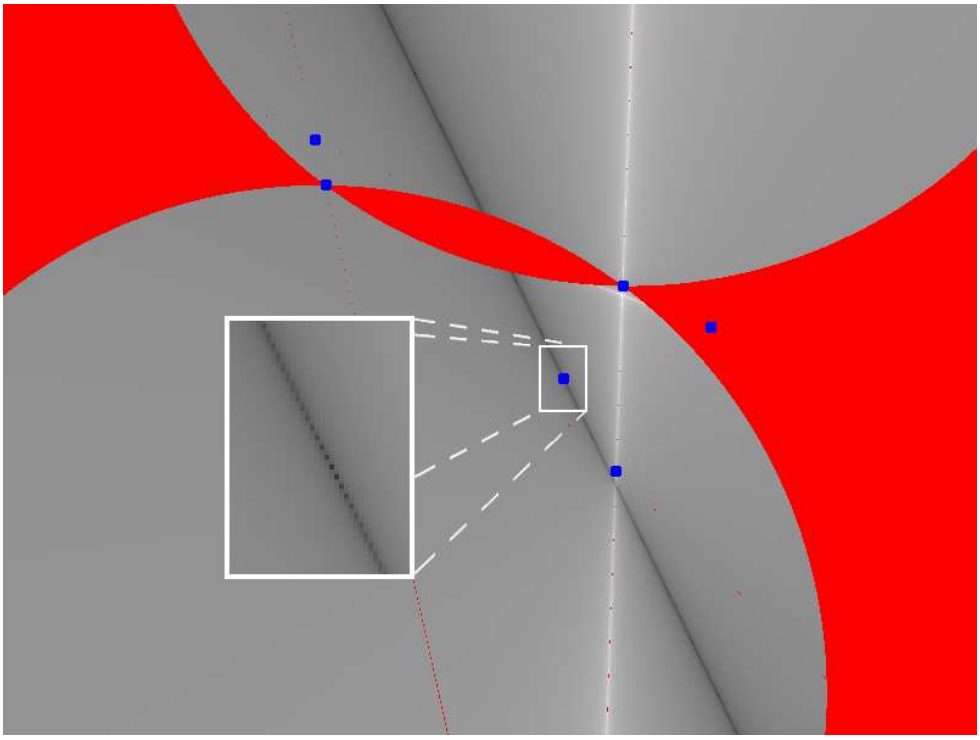


Figure 3.1: Perspective invariant log output $\log|f(I,K)|$. Small rectangle is zoomed-in.

different information, as introduced in section 2.4.

Brief outline of the data embedding process on a 3D model M is given below:

1. Select six interest points in 3D $P = \{P_i\}, i \in [1,6]$.
2. Embed data on 3D interest points by modifying their relative positions; creating different versions of the 3D model.
3. Calculate I coefficients for each version of the 3D model ($S_I = \{I_i\}, i \in [1,n]$).

If there are n versions of the model, $\log_2(n)$ bits of information are embedded. 2D views of different model versions are then produced. Camera position and perspective projection do not affect the algorithm, as they do not appear in the perspective invariant. However, it is important that interest points should be visible in 2D views.

Data is extracted from 2D views by applying the following steps:

1. Given a 2D view, detect interest points, $p = \{p_i\}, i \in [1,6]$.

2. Calculate K coefficients using interest points.

3. Using the perspective invariant, find the I_{min} value that gives the minimum value of $|f(I_i, K)|$ for $I_i \in S_I$.

I_{min} value shows which version of the 3D model was used. To extract data, watermark detector has to have prior knowledge of S_I.

Note that the set of coefficients S_I can be thought as a viewpoint independent representation of all versions of the model. A hypothetical 3D-2D watermarking detector can be devised that stores all model versions and compares transmitted 2D views to these models to detect embedded data. However, in this case, transmitted 2D images would need to be registered to 3D models, and camera parameters would need to be determined before detection. Since I coefficients are viewpoint independent, no registration is needed prior to watermark detection. Therefore, storing S_I is more efficient than storing all model versions in the detector.

3.2.1 Selection and Detection of Interest Points

The proposed watermarking system depends on interest points being detected on arbitrary 2D views of models. Ideally, interest points should remain detectable, when they are viewed by using different camera angles, under changing lighting conditions and even after data is embedded within the model. In this chapter, to simplify interest point detection problem and accurately judge performance of the perspective invariant, a basic interest point detection scheme is implemented. The problem of repeatable interest point detection is examined separately in Chapter 5.

3.2.1.1 Detection of Interest Points

Interest points are selected to be centroids of triangular faces of a 3D mesh model. Selected faces are painted with distinct hues (red, green, blue, yellow, cyan, and magenta), while unselected faces are left white. Since triangle centroids are invariant under perspective projection, detection of interest points in 2D views is achieved by isolating regions of constant hue, and calculating their centers of gravity.

3.2.1.2 Selection of Interest Points

Interest point detection is simplified by marking them with distinct hues; however, interest points still need to be selected carefully. This selection is inevitably related to camera position. Naturally, if an interest point is not visible in a 2D view then the watermark cannot be detected. Any 3D-2D watermarking system embedding data on 3D mesh features has this dependence on camera direction.

Therefore, a *view direction* is selected to represent a set of possible camera positions around that direction. Watermarked 2D views are to be produced within a range around the selected view direction. Interest points are also selected according to this view direction. Therefore, calculated I coefficients are also related to this view direction. Watermark can be embedded multiple times using different view angles, if detection is required from 2D views that are obtained from camera angles far from the chosen view direction.

This approach has some precedent in the literature. In [2], embedded watermark is only detectable from the range of 2D views it is visible. In [23], if multiple camera directions are needed, watermark is embedded by using different camera directions, on the contours of a model. Silhouettes of the model from different camera directions are stored, and best suitable silhouette is selected by comparison with given 2D view before watermark detection. Segmentation of such silhouettes from arbitrary views is still an open problem. Moreover, those silhouettes are expected to be time-varying in a typical application.

Among all the faces of a given model, the following constraints are applied to find a set of candidate interest points that are suitable for watermarking for a given view direction:

- **Occlusion constraint:** Face centroids should not be occluded.
- **Direction constraint:** Angle between face normal and view angle, that is the vector connecting face centroid to center of a camera positioned at the view direction, should be less than 60 degrees.
- **Adjacency constraint:** Selected faces should not be adjacent, to prevent unwanted motion of adjacent faces during watermark embedding.
- **Invariant constraint:** First four selected centroids of the faces should not be coplanar, due to the perspective invariant constraints.

After applying these constraints, remaining centroids of the faces are candidates for being used as 3D interest points in watermarking.

3.2.2 Embedding Data

Data is embedded in a 3D model by changing positions of 3D interest points in a way that is not possible by a projective transformation. This operation can be shown as $P'_i = P_i + \vec{v_i}$ where $\vec{v_i}$ is the translation vector for i-th interest point and not all $\vec{v_i}$ are equal.

Watermark energy can be defined as the average of translation vector magnitudes, as given in (3.23). Watermark energy will determine the amount of 3D mesh distortion and contribute to how robustly watermark is embedded.

$$s_w = \frac{\sum|\vec{v_i}|}{6} \tag{3.23}$$

It can be deduced from Figure 3.1 that watermark energy alone will not determine the robustness of the embedded watermark. Direction of $\vec{v_i}$ is important, since the perspective invariant functions value is direction dependent around interest points. This fact can also be inferred from the observation that if 3D interest points are moved towards the camera during data embedding, their 2D counterparts will not move significantly, and embedded watermark will not be extracted.

The translation vectors, $vecv_i$, can be determined by methods using heuristics and through optimization of an appropriate objective function. Both approaches are investigated in this chapter. In both cases, the following hypotheses are used to enable data embedding process: If interest points are displaced parallel to image plane of the camera, largest interest point displacement will be generated on 2D views. If interest points are displaced parallel to the face surface, 3D mesh distortion will be lower.

In the following analysis of data embedding methods, $\vec{c_i}$ is the vector connecting camera center and interest point P_i, and $\vec{n_i}$ is the normal vector of the face of the 3D mesh on which P_i resides. Both $\vec{c_i}$ and $\vec{n_i}$ are defined as unit vectors.

3.2.2.1 Data Embedding Using Heuristics

In data embedding methods using heuristics, each interest point is displaced by a fixed amount along a direction. Displacement direction is heuristically selected based on view vector ($\vec{c_i}$) and face normal ($\vec{n_i}$).

The following displacement directions are tested:

- Move towards or away from the camera: $\vec{v_i} = s_w \cdot |\vec{c_i}|$.
- Move perpendicular to view vector, parallel to image plane: $\vec{v_i} = s_w \cdot |\vec{c_{i\perp}}|$.
- Move perpendicular to face normal (parallel to face surface) and perpendicular to view vector (parallel to image plane): $\vec{v_i} = s_w \cdot |\vec{n_i} \times \vec{c_i}|$.

Keeping translation parallel to image plane results in largest interest point displacement in generated 2D views, which is expected to create a large difference in the value of the watermark function. This expectation is based on the behavior of the perspective invariant. For the best performance, direction and magnitude of $\vec{v}$ should take into account the behavior of the perspective invariant. Embedding data optimally aims to address this problem.

3.2.2.2 Optimal Data Embedding

When data is embedded in a 3D model to produce different model versions M_i, and a 2D image m_j obtained from one of the models, it is expected that $|f(I_i, K_j)| = 0$ only for $i = j$. In other cases, $|f(I_i, K_j)|$ will be larger than zero. To maximize watermark detection performance, difference between these two values should be maximized.

On the other hand, a constraint on the amount of displacement of interest points should be imposed to limit the amount of 3D mesh distortion due to watermarking. Thus, the objective function O, that is to be maximized, is proposed below:

$$O = \big|\, |f(I_j, K_j)| - |f(I_i, K_j)| \,\big|$$
$$\text{where } i \neq j,\ |\vec{v_k}| < \epsilon, \text{ and } P'_k = P_k + \vec{v_k},\ k \in [1, 6]. \quad (3.24)$$

An optimization on this objective function should be carried out in 3D. In this case an 18-dimensional space (3 dimensions for 6 interest points) must be searched. The heuristics stated in the previous section might be used to reduce the dimensions; and hence, operation time of the search. Different data embedding approaches based on optimization and application of heuristics to reduce search space dimension is given below.

- **3D optimization:** Optimize in a 3D region centered around P_i. No heuristics applied.
- **2D optimization:** Optimize on the plane perpendicular to view vector, parallel to image plane, centered at P_i.
- **2D optimization:** Optimize on the plane perpendicular to face normal, parallel to face surface, centered at P_i.
- **1D optimization:** Optimize on face normal centered at P_i.
- **1D optimization:** Optimize on a line perpendicular to both face normal and view vector, centered at P_i.

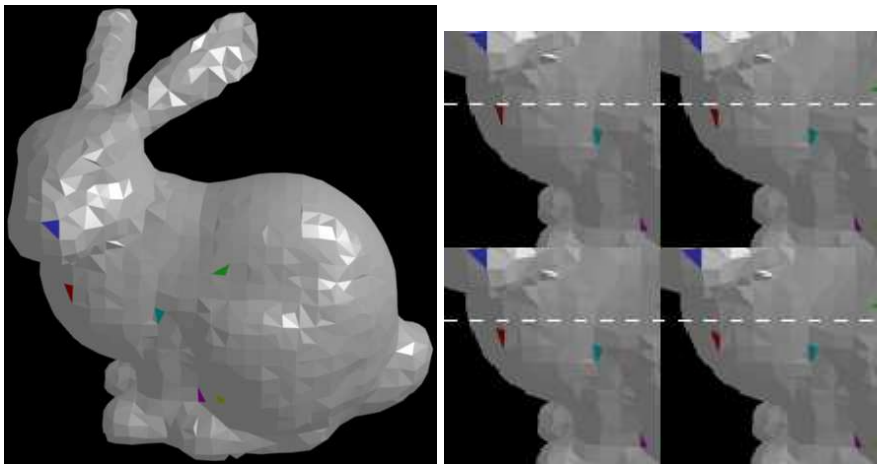
Note that using camera location based heuristics is just an ad-hoc solution to reach an acceptable data embedding method. It is not mandatory to use camera location while embedding data. In fact, some of the data embedding methods, for example 3D optimization, do not utilize camera position. In general, the proposed method of watermarking using perspective invariants is camera position independent.

Data embedded on a single interest point utilizing the $s_w \cdot |\vec{n_i} \times \vec{c_i}|$ heuristic method for different watermark energies (s_w) on the *Bunny* model is shown in Figure 3.2. As the watermark energy is increased, the red triangle on *Bunny*'s chest moves down further.

3.2.3 Watermark Detection

$S_I = \{I_i\}, i \in [1, n]$, the set of I coefficients associated with a model, or equivalently, the set of watermark functions $S_F = \{F_i\}, i \in [1, n]$ where F_i is defined as $F_i(x) = |f(I_i, x)|$, is available at the watermark detector.

When a 2D image of the model is received, interest points are detected and K coefficients are



(a) Interest points (b) Embedding data

Figure 3.2: Interest points on original *Bunny* model and embedded data on modified models, for watermark energies 0.025, 0.05, 0.075, 0.1 (dashed lines added for clarity).

calculated. I_{min} that minimizes the following relation determines which version of model was used to generate the 2D view, as follows:

$$I_{min} = \arg\min_{I_i \in S_I} |f(I_i, K)| \tag{3.25}$$

3.3 Experimental Results

Simulations are conducted in two phases. In the first phase randomly sampled 3D points independent of a model are used as interest points. Interest point detection is simulated by adding Gaussian noise onto 2D interest point coordinates. In the second phase, interest points are marked on 3D mesh models as explained in Section 3.2.1. Interest points are detected utilizing face colors.

In both phases, constrained Newton Conjugate-Gradient is used as the optimization method [35, 36]. Optimization regions are defined as cubes, squares and line segments depending on the number of dimensions. Longest axis of the optimization region is set to be watermark energy upper bound value, that is, ϵ in (3.24).

3.3.1 Interest Point Detection on Synthetic Data

For each iteration of the experiment, six 3D interest points are randomly selected from a uniform distribution around the 3D origin. A perspective camera, located on the negative z-axis looking towards the origin is used to obtain 2D interest point coordinates. 2D coordinates are then scaled to correspond to pixel coordinates of a square image of 800 pixels on one side. Distribution of random 3D points and camera parameters are selected to be compatible with the simulations for 3D mesh models. Gaussian noise is added to pixel coordinates to simulate interest point detection.

In this phase, since there are no models and no faces, only the following data embedding methods are simulated:

- 3D Optimization
- 2D Optimization on the plane perpendicular to view vector ($2D - \vec{c_\perp}$)
- 1D Optimization along the view vector ($1D - \vec{c}$)
- Heuristics: Along the view vector ($\vec{c}$ heuristics)
- Heuristics: Perpendicular to view vector ($\vec{c_\perp}$ heuristics)

One bit of information is embedded by creating one additional version of the original model using selected data embedding method. Performance of the proposed system is examined for a range of watermark energy upper bounds (0.05, 0.1, 0.15, 0.2), viewing angles (0°, 15°, 30°, 45°) and 2D interest point Gaussian noise levels (0.0, 0.5, 1.0, 1.5).

Watermark detection is declared as successful, if

$$|f(I_2, K_2)| < |f(I_1, K_2)| \tag{3.26}$$

where I_1 is obtained from the original model and I_2 from the modified (data embedded) model. K_2 is obtained from a 2D view of the modified model. For optimization based methods, the following objective function is used:

$$O = | \, |f(I_2, K_2)| - |f(I_1, K_2)| \, | \text{ where } |\vec{v_k}| < \epsilon, k \in [1, 6]. \tag{3.27}$$

If at any point in the iteration, selected interest points or their 2D counterparts fail to conform to perspective projection constraints given in Section 3.1, iteration is restarted using another set of random interest points.

In Figure 3.3, performance of five data embedding methods is plotted against standard deviation of Gaussian noise added to 2D interest point coordinates, at 0° viewing angle, across all watermark energy levels. As can be seen, all methods exhibit perfect performance when there is no noise added, except $\vec{c}$ heuristics. This result is expected, since by moving 3D interest points towards the camera, no change is observed in 2D interest point locations. Detection performance degrades as noise strength is increased. Optimization based methods have better performance than heuristics based methods for any noise level, while 2D optimization shows the best performance, and other optimization methods closely following. At noise levels of 0.5 and 1.5 pixels standard deviation, detection performance drops to 95.3% and 88.6% for 2D optimization, and 92.8% and 82.7% for $\vec{c_\perp}$ heuristics.

In Figure 3.4, detection rate is plotted against viewing angle (from 0° to 45°) for a noise level of 0.5 pixel standard deviation. Changing viewing angle in this range reduces performance of optimization and $\vec{c_\perp}$ heuristics based methods by about 3%. On the other hand, detection rate for $\vec{c}$ heuristics increases greatly with viewing angle, since when viewed from an angle, 2D interest points' translation towards the original camera position becomes more visible.

Finally, Figure 3.5 displays detection rate against watermark energy at 0° viewing angle and 1.5 pixel noise standard deviation. This noise value is selected to distinctively evaluate different methods' performance. $\vec{c_\perp}$ heuristics based method shows better performance by increasing watermark energy. Optimization based methods are ahead of heuristics based methods, although they do not exhibit a clear relationship in terms of detection rate. 3D, 2D and 1D optimization methods reach peak detection rates of 92.2%, 91.6% and 91.6%, respectively. $\vec{c_\perp}$ heuristics follows with a peak detection rate of 85.8%.

Results obtained in this phase demonstrate that 3D-2D watermarking by perspective invariants is feasible, even when 2D interest point detection is noisy. 2D optimization among the optimization based methods, and $\vec{c_\perp}$ among heuristics based methods have better performance,

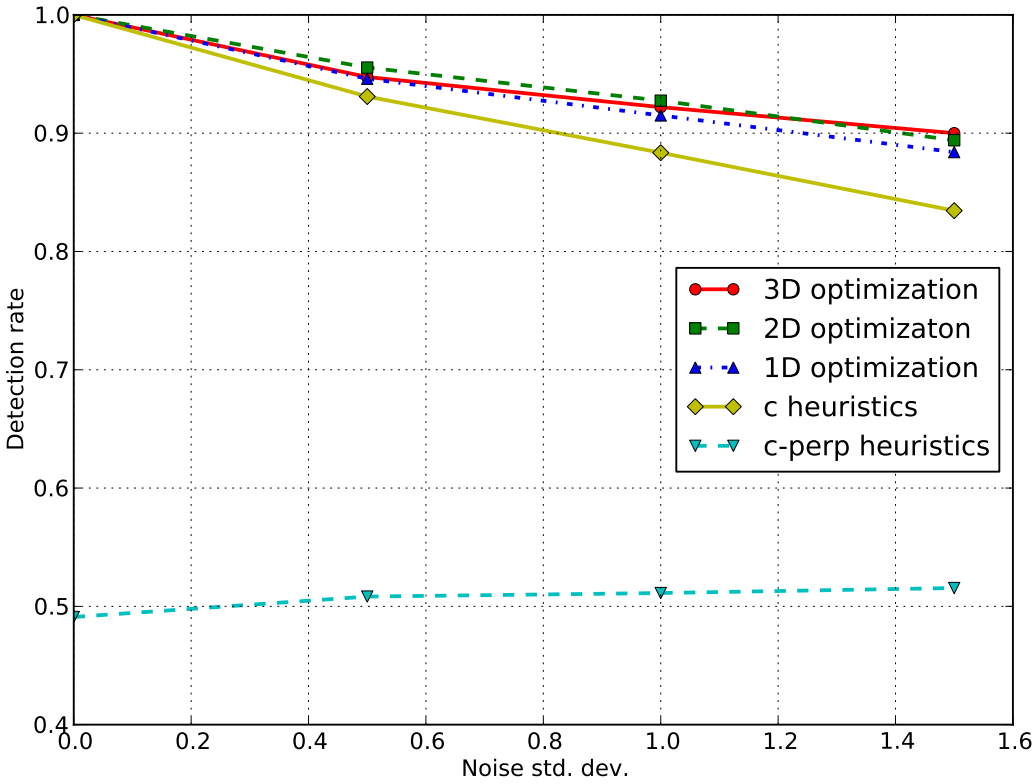


Figure 3.3: Detection rate versus Gaussian noise standard deviation, plotted for different data embedding methods.

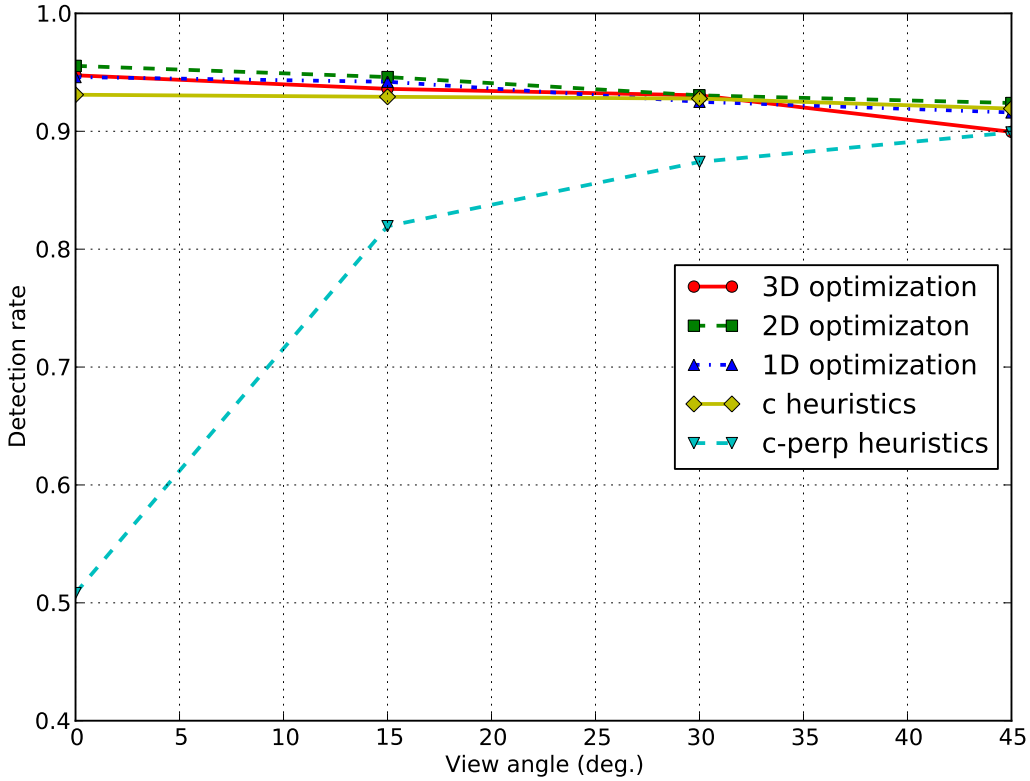


Figure 3.4: Detection rate versus viewing angle, plotted for different data embedding methods.

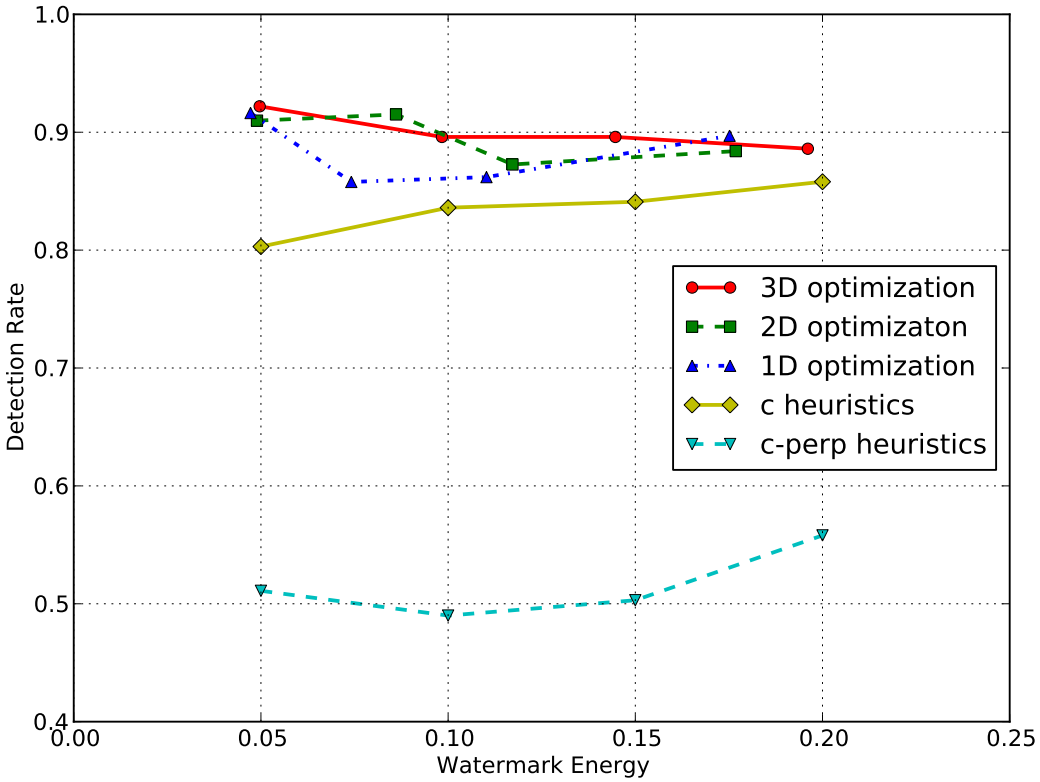


Figure 3.5: Detection rate versus watermark energy (s_w), plotted for different data embedding methods.

when compared to others. Optimization based methods exhibit better performance, especially under higher noise levels and low watermark energies.

3.3.2 Interest Point Detection on 3D Models

In this phase simulations are done on triangular faced mesh models with no textures. Mesh models are rendered in Blender 3D software. Rendered 3D scene contains the model to be rendered, a camera and a lamp (see Figure 3.6). 2D views are rendered with a camera of 35 mm focal length. Camera is rotated around the model to the desired view angle.

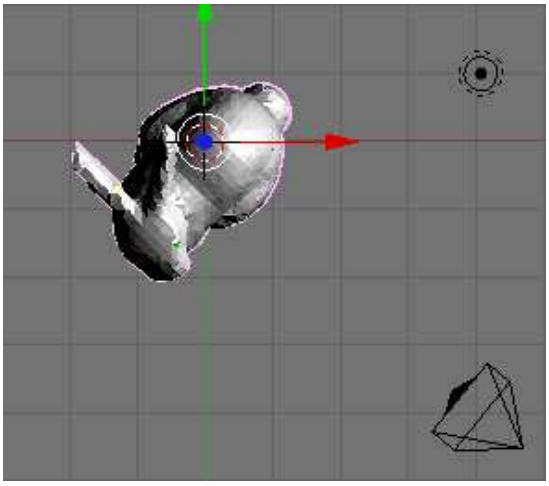


Figure 3.6: Top view of 3D modeling environment. 3D model is near top left corner. Camera is near bottom right.

Six models from Stanford 3D Scanning repository [37] and Aim@Shape repository [38] are used (Figure 3.7). Number of faces and bounding box sizes of the models are given in Table 3.1.

An iteration of the simulation involves the following steps:

1. Interest points are randomly selected and marked on the model as described in Section 3.2.1. Model-0 is created.

2. Watermark is embedded by moving interest points according to the selected watermark energy, as described in Section 3.2.2. Model-1 is created.

Table 3.1: Properties of models used in experiments.

Model name	Number of faces	Bounding box size
Bunny	3851	(3.35, 3.27, 2.59)
Dragon	11102	(4.06, 2.87, 1.82)
Armadillo	6918	(5.08, 6.06, 4.61)
Buste	10214	(4.51, 2.43, 1.80)
Fertility	12080	(5.99, 4.34, 2.20)
Happy Buddha	15536	(2.11, 5.14, 2.11)

3. 2D views are obtained for the original model (Model-0) and the modified model (Model-1), after rotating the camera by the specified view angles.

4. I coefficients for Model-0 and Model-1 are calculated, F_0 and F_1 obtained.

5. Mesh distance values are calculated between Model-1 and Model-0, using the Metro tool [39].

6. Signal to watermark energy ratios of 2D watermarked views to original model views are calculated.

If not exactly six interest points are detected, or detected interest points violate the perspective invariant constraint given in Section 3.1, the iteration is discarded. The iteration is considered a success, if watermark in a 2D image generated from Model-1 is successfully detected ($F_1(K_1) < F_0(K_1)$).

Data is embedded using the following methods:

- 3D Optimization
- 2D Optimization on the plane perpendicular to view vector ($2D - \vec{c_\perp}$)
- 2D Optimization on the plane perpendicular to face normal ($2D - \vec{n_\perp}$)
- 1D Optimization along the face normal ($1D - \vec{n}$)
- 1D Optimization the line perpendicular to face normal and view vector ($1D - \vec{n} \times \vec{c}$)
- Heuristics: Along the line perpendicular to face normal and view vector ($\vec{n} \times \vec{c}$)

Figure 3.7: Mesh models used in simulations, with interest points marked.

Performance of the proposed system is investigated for watermark energy constraint values (for optimization based methods) and watermark energy values (for heuristics based method) of 0.002, 0.01, 0.025, 0.05, 0.075, 0.1. View angles of 0°, 7.5°, 15° are simulated. Higher watermark energy values produce quite pronounced distortion on models and thus were not simulated. Larger view angles result in occlusion of interest points, causing too many discarded iterations.

Mesh distortion is measured by Hausdorff distance. Hausdorff distance defined on two input meshes X and Y is given below. $d(x, Y)$ is the Euclidean distance from a point x on X to the closest point on Y:

$$d_H(X, Y) = \max\{\max_{x \in X} d(x, Y), \max_{y \in Y} d(y, X)\}$$

Detection rate of all data embedding methods for a viewing angle of 0° are given in Figure 3.8. The x-axis denotes watermark energy (s_w). Optimization based methods are plotted by accumulating results from all iterations in 6 buckets according to watermark energy. Bucket sizes are selected so that there are equal number of samples in each bucket. Detection rate of each bucket is calculated by dividing the number of successful detections by the number of samples. Detection rates are plotted at average watermark energy for each bucket. This approach prevents erroneous results caused by small number of results in a bucket.

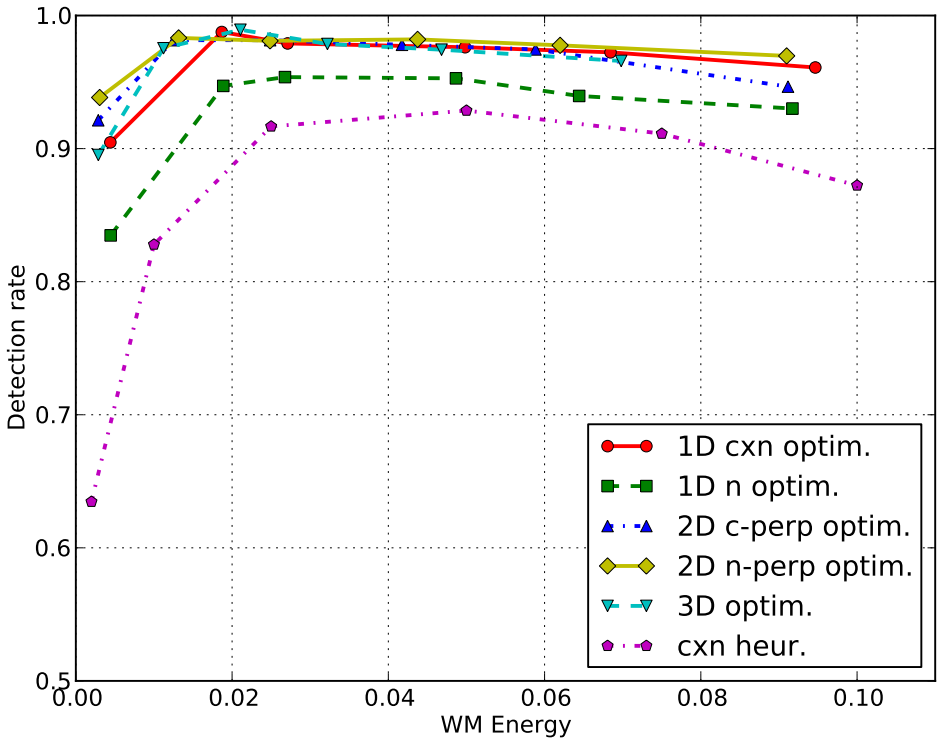


Figure 3.8: Detection rate versus watermark energy bound or watermark energy (s_w or ϵ). Color represents data embedding method (red: $1D - \vec{n} \times \vec{c}$, green: $1D - \vec{n}$, blue: $2D - \vec{c_\perp}$, yellow: $2D - \vec{n_\perp}$, cyan: 3D optimization, magenta: $\vec{n} \times \vec{c}$ heuristics).

Overall, detection rates for optimization based methods are higher than ($\vec{n} \times \vec{c}$) heuristics method, reaching 99.0% for 3D optimization at 0.021 watermark energy (cyan line) and 98.8% for 1D optimization on the line $\vec{n} \times \vec{c}$ at 0.019 watermark energy (red line). The $\vec{n} \times \vec{c}$ heuristics method exhibits at best 92.9% detection rate at 0.050 watermark energy. It can also be observed that optimization based methods show significantly better performance at lower watermark energies: When watermark energy is equal to 0.021, 3D optimization performs 10.1% better than heuristics based method.

For all methods, detection rate is observed to decrease, when watermark energy is increased beyond a level. The main reason for this unexpected situation is the increased probability of self occlusion of faces after data embedding, and in consequence, increased localization errors in interest point detection at high watermark energy levels. In Table 3.2, performance of all data embedding methods is given at 0.1 watermark energy upper bound for two cases: All iterations included, and those iterations having interest point detection errors of more than 0.5

Table 3.2: Detection rate increase when interest point detection errors are removed at 0.1 watermark energy.

Method	All	Errors discarded
3D Optimization	96.5%	98.2%
$2D - \vec{c_\perp}$	94.7%	97.1%
$2D - \vec{n_\perp}$	97.0%	98.2%
$1D - \vec{n}$	93.1%	94.5%
$1D - \vec{n} \times \vec{c}$	96.1%	98.9%
$\vec{n} \times \vec{c}$ heuristics	87.2%	95.5%

pixels discarded. The detection rate increase once again emphasizes importance of interest point detection and localization.

As can be seen in Figure 3.9, data embedding methods that displace interest points perpendicular to face normal (1D optimization along $\vec{n} \times \vec{c}$, 2D optimization on $\vec{n_\perp}$, and $\vec{n} \times \vec{c}$ heuristics) produce the lowest mesh distortion, as expected. Optimization along the face normal results in largest mesh distortion. In general, mesh distortion increases by increasing watermark energy almost linearly.

Figure 3.10 shows detection rate versus view angle characteristics for all data embedding methods. 3D optimization is observed to perform slightly better than other methods at 15° viewing angle, although detection performance degradation of only 1D optimization on $\vec{n}$ method is significant (2.5%). Signal to watermark ratio between generated 2D views p and p' generated from Model-0 and Model-1, respectively, is given in (3.28). In (Figure 3.11), signal to watermark ratio is observed to be inversely related to watermark energy, as expected.

$$SWR(p, p') = 10 \log \frac{\sum p^2(x,y)}{\sum |p(x,y) - p'(x,y)|^2} \tag{3.28}$$

The results obtained in this phase indicate that optimization based methods continue to perform better than heuristics based methods, even when 3D models are used. On the other hand, reduced detection rate in higher watermark energy values demonstrate the difficulty and importance of interest point detection and localization after data is embedded in the mesh.

Data embedding methods that displace interest points perpendicular to face normal result in smaller mesh distortion in 3D, and higher signal to watermark ratio in 2D. On the other hand 3D optimization, despite being computationally costly, is both camera position independent

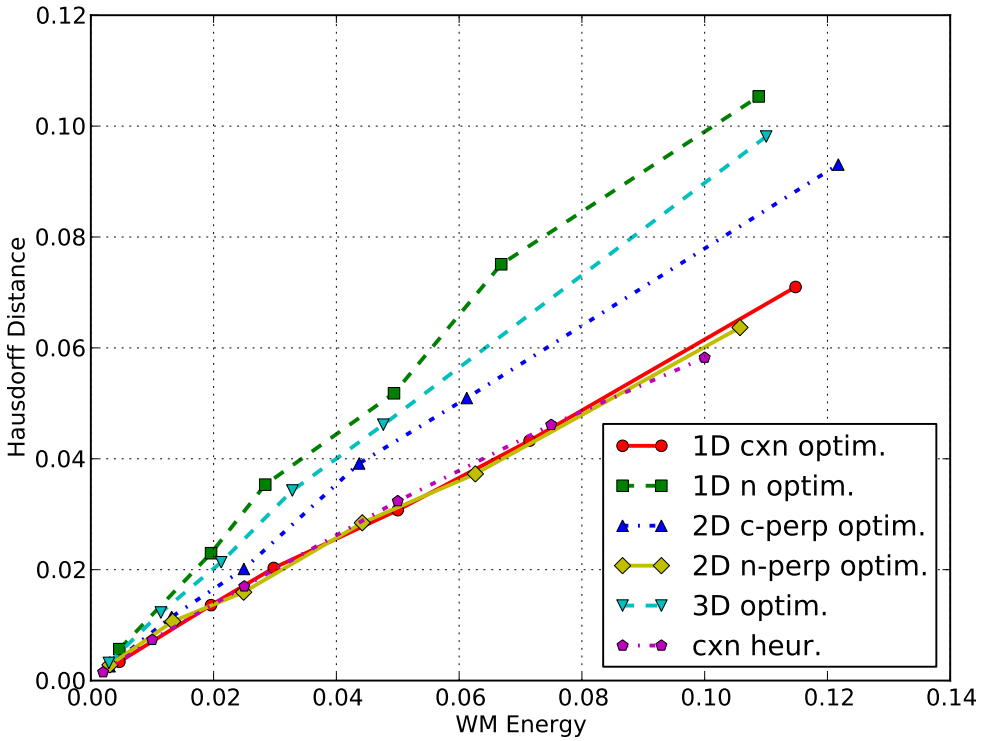


Figure 3.9: Mesh distortion versus watermark energy bound or watermark energy (s_w or ϵ), plotted for different data embedding methods.

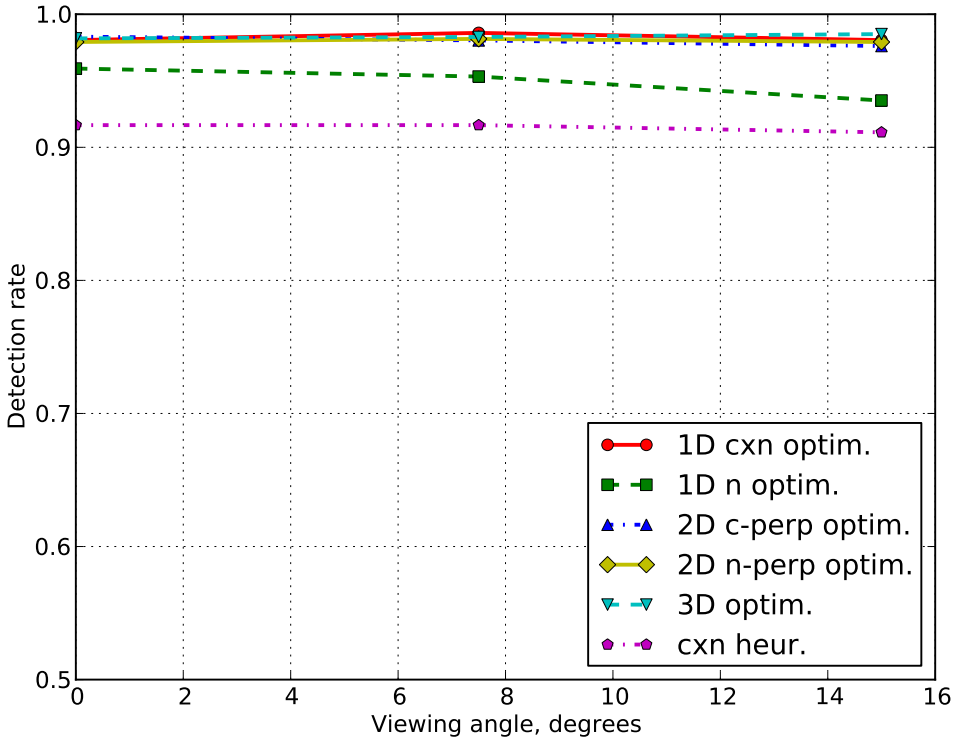


Figure 3.10: Detection rate versus viewing angle, plotted for different data embedding methods.

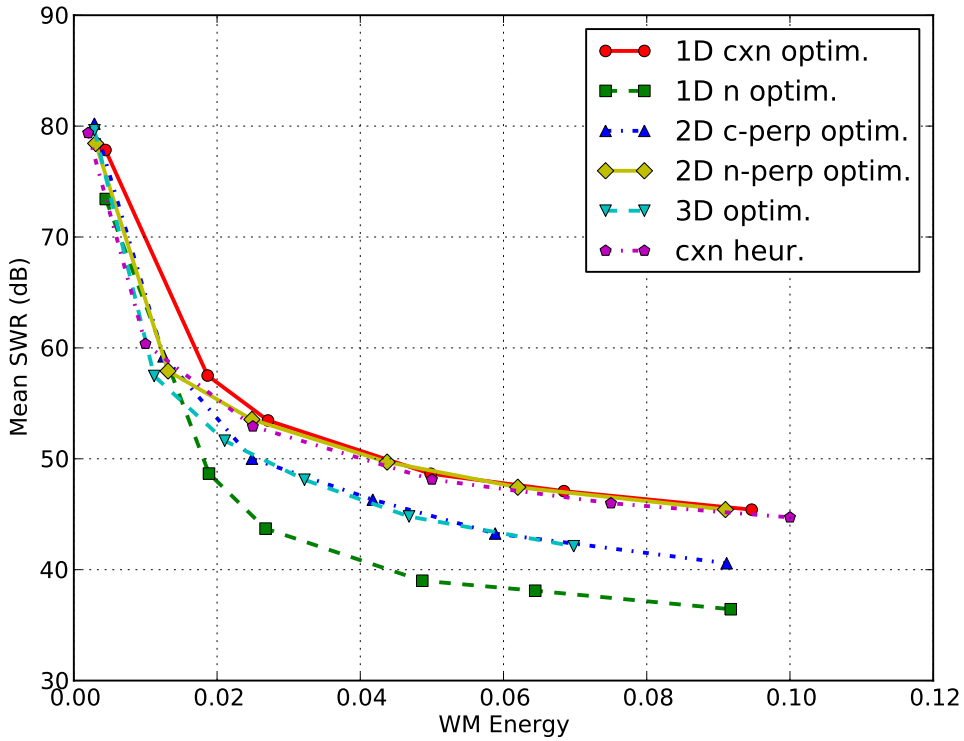


Figure 3.11: Signal to watermark ratio versus watermark energy bound or watermark energy (s_w or ϵ), plotted for different data embedding methods.

and improves other methods by a small margin.

3.3.3 Discussion of Results

In this chapter, a watermarking method that extracts data embedded in 3D models from arbitrary 2D views is presented. Proposed method utilizes a perspective invariant relating six minimally constrained 3D points and their 2D counterparts. Data is embedded by displacement of 3D interest points. Primary challenges of this watermarking method are identified to be noise sensitivity of the perspective invariant, and accurate 2D interest point detection.

An optimal data embedding method is developed to overcome the noise sensitivity of the perspective invariant. An objective function is defined considering the invariant function, and maximized to find optimal 3D interest point displacements.

A basic interest point detection scheme is devised to accurately judge perspective invariant performance. Interest points, which are selected to be face centers, are marked on mesh models by coloring faces with distinct hues. Interest point detection is accurate, since centers of triangular faces are perspective invariant.

Presented watermarking system is evaluated by simulations performed on point sets and mesh models. Simulations highlight a number of observations:

- Optimally embedded data in 3D mesh models can be extracted from 2D views with 99.0% watermark detection performance when one bit of information is embedded into two models.
- Optimization based methods have 10% higher watermark detection performance than heuristics based methods. In addition, peak detection rates are observed at lower watermark energies with optimization based methods, resulting in lower distortion in 3D meshes and 2D views.
- Detecting watermark on 2D views obtained with 15° camera angle does not produce significant performance degradation.
- Even with the basic detection scheme, accurately detecting interest points remains a factor in watermark detection performance. Especially self occlusions caused by data

embedding prevent watermark detection at high watermark energies.

The results provide a benchmark for optimal 3D-2D watermarking performance using this perspective invariant. Since, during the simulations, a single bit of information is hidden in models, avenues for improvement remain open for investigation, such as embedding data more than once on different interest point sets or employing error correction coding.

These results are a very promising indicator that perspective invariants based 3D-2D watermarking, a completely new area of research, is feasible.

CHAPTER 4

UTILIZATION OF PROJECTIVE INVARIANTS IN 3D-2.5D WATERMARKING

In the previous chapter, a feasible watermarking system was built on the idea of a perspective invariant remaining unchanged between a 3D model and its 2D representations. Existence of 3D projective invariants, as mentioned in Chapter 2, suggests that a similar watermarking scheme can be built between two 3D representations of a model.

Representation of a 3D model using range data in addition to a 2D image, or using a stereo pair of 2D images has become prevalent in broadcasting digital 3D video [16]. Moreover, developments in digital entertainment technology, specifically Microsoft Kinect, enabled easy and cheap access to depth sensors. As a result, many academic and engineering projects have appeared utilizing Kinect's depth sensing abilities [40, 41].

This chapter investigates feasibility of a watermarking system in which data embedded in a 3D model is extracted from its 2D plus depth representations. First, 3D plus depth representations and watermarking schemes operating on them are briefly reviewed. The requirements for a 2D plus depth representation to be suitable for watermarking with projective invariants is investigated next. A watermarking system that embeds data in 3D and extracts from 2D plus depth representations is presented afterwards.

4.1 2D Plus Depth Representations and Watermarking

Depth image based rendering (DIBR) is the process of synthesizing virtual views of a scene from still or moving images and associated per-pixel depth information [9, 42]. This process

consists of two steps: Reprojecting original 2D points to 3D space using the depth data, followed by projecting the reprojected 3D points to image plane of a virtual camera. This process of re-projection from 2D to 3D, and projection from 3D back to 2D is called *3D image warping* [9].

DIBR can be used to generate a stereo pair of images from a 2D image and associated depth map. Such a method of creating stereo view pairs using depth information broadcasted along with traditional video is proposed for 3D-TV systems [9]. Another application of the 2D plus depth representations, Kinect provides a depth image of the visual field in addition to RGB video. After calibration, actual depth of a point can be obtained [40]. 2D plus depth map representations of 3D scenes are expected to proliferate further in the future.

Watermarking methods were developed for 2D plus depth image representations. However, none of the methods in the literature extract data embedded in 3D models from 2D plus depth representations. Three recent examples for 2D plus depth watermarking methods are given below.

The first method [43] employs depth information to determine 2D image regions in which data is embedded. Background of a scene is thought to carry less information and more frequently be subjected to attacks. On the other hand, objects that are farther away are not distorted as much as closer objects, during 3D warping. Combining these two ideas, regions in the image that correspond to the farthest object that is not the background are deemed to be more suitable for embedding data. Depth map is processed to identify these regions during both watermarking and detection of the watermark. Data is embedded within DCT coefficients of identified regions on 2D images.

In another watermarking method for DIBR based free viewpoint video systems [44], watermark is embedded in the source cameras in such a way that it can be detected on artificially generated 2D images from arbitrary viewpoints. A random watermark pattern is generated according to the reference camera view point. The watermark pattern is warped to all the other source cameras so that effectively the 3D space is watermarked.

Similarly, In a system which generates stereo views from a central image and associated depth map, different watermark patterns for left and right stereo pairs are embedded in the center image, accounting for 3D warping [45]. Regions in which there are many holes due

to warping are left intact. Watermark is detected after the virtual view's projective camera parameters are determined.

4.2 2D Plus Depth Requirements for 3D-2.5D Watermarking

As demonstrated in Section 2.2.2, the mapping between a 3D point X and its related depth image point X_d,

$$X_d = H_{depth} \cdot X \tag{4.1}$$

is a projective transformation. Here, H_{depth} is defined as $H_{depth} = H \cdot T$ where T is the camera matrix and

$$H = \begin{pmatrix} \frac{1}{\omega} & 0 & 0 & 0 \\ 0 & \frac{1}{\omega} & 0 & 0 \\ 0 & 0 & -\frac{255 \cdot k}{z_{far} - z_{near}} & -\frac{255 \cdot z_{far}}{z_{far} - z_{near}} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.2}$$

as given in (2.33). Note that H depends on ω, which is related to actual point depth by a scalar factor (2.28). Therefore, H_{depth}, which is the product of H and the camera matrix T, depends not only on the camera parameters but also on the depth of the 3D point being mapped.

As a result of this, a group of 3D points cannot be mapped from 3D coordinates to depth image coordinates by a constant projective transform. That is, H_{depth} changes for each point according to its depth. This prevents using depth image coordinates directly in projective transformation based watermarking.

On the other hand, if actual depth of a point, or equivalently z_{near} and z_{far} values that are used in obtaining the depth image (see equation (2.27)) are known, H's dependence on ω can be removed by multiplying x' and y' in (2.34) with the depth, as shown below:

$$\begin{pmatrix} x' \cdot d \\ y' \cdot d \\ z' = 255 \cdot (\frac{z_{far}-d}{z_{far}-z_{near}}) \\ 1 \end{pmatrix} = H' \cdot T \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = H'_{depth} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{4.3}$$

where

$$H' = \begin{pmatrix} k & 0 & 0 & 0 \\ 0 & k & 0 & 0 \\ 0 & 0 & -\frac{255 \cdot k}{z_{far}-z_{near}} & -\frac{255 \cdot z_{far}}{z_{far}-z_{near}} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.4}$$

and $k = {}^{d}/_{\omega} = {}^{sign(\det M)}/_{\|m^3\|}$ is a constant scalar for given camera parameters. Hence, the new transform, $H'_{depth} = H' \cdot T$, lacks any dependent factor and is constant for all points being mapped.

Note that this new mapping is between a 3D point $(x, y, z, 1)^T$ and another 3D point $(x' \cdot d, y' \cdot d, z', 1)^T$. Here $(x', y', z', 1)^T$ is the depth image point X_d corresponding to the 3D point $X = (x, y, z, 1)^T$, and d is the actual depth of the 3D point. Unlike depth image coordinates, $(x' \cdot d, y' \cdot d, z', 1)^T$ is related to the 3D point through a constant projective transformation given camera parameters, and thus can be used in watermarking based on projective invariance.

The point $(x' \cdot d, y' \cdot d, z', 1)^T$ can easily be obtained from 2D image coordinates x', y', actual depth of the point, d, or depth image value z', and z_{near} and z_{far} values used in obtaining the depth image. In this chapter, three dimensional point $X'_d = (x' \cdot d, y' \cdot d, z', 1)^T$ is called the *modified depth image point* related to depth image point X_d.

Given these definitions, $H - depth'$ is a projective transform between a set of 3D points and related modified depth image points obtained from a 2D plus depth representation of the scene. This projective transformation relation between 3D models and their 2D plus depth representations explained in this section enables us to devise a watermarking scheme utilizing projective invariants. Next section provides a suitable 3D projective invariant to be used in this scheme.

4.3 3D Projective Invariant

It was stated in Section 3.1 that the coefficients $I = \{I_{12}, I_{13}, I_{14}\}$ obtained by (3.21a) are 3D projective invariants. In order to show that I coefficients remain unchanged under an arbitrary projective transform, it is useful first to introduce the concept of a *relative invariant* [11]. A quantity related to a geometrical configuration that changes under a transformation T by a factor of $|T|^{\omega}$, $\omega > 1$, is called a relative invariant under that transformation.

The determinant of a matrix formed by combining four 3D point homogeneous coordinates is a relative invariant of any linear transformation [11, 34, 33]. If the transformation is defined as $y = T \cdot x$, and the four points are $X = (x_1, x_2, x_3, x_4)$, transformation of all points can be written as $Y = T \cdot X$. Taking determinant of both sides, the following relation is obtained:

$$|Y| = |T| \cdot |X|. \tag{4.5}$$

As can be seen, between the input and transformed determinants, there is only a difference of $|T|$ [11], therefore, determinant of four points is a relative invariant under T. Since T is a non-singular 4x4 matrix, it is a projective transformation, and the determinant is a relative invariant under projective transformation.

An invariant (also called an *absolute* or *scalar* invariant) can be built using relative invariants [11]. For example, dividing the relative invariant above by determinant of another four point set X' provides the following invariant:

$$\frac{|Y|}{|Y'|} = \frac{|X|}{|X'|}. \tag{4.6}$$

Given the six 3D points $P_1, P_2, P_3, P_4, P_5, P_6$, relative invariants can be constructed by using any four of them. The following relative invariants are defined:

$$M_1^\alpha = |P_2, P_3, P_4, P_5|, \tag{4.7}$$

$$M_2^\alpha = |P_1, P_3, P_4, P_5|, \tag{4.8}$$

$$M_3^\alpha = |P_1, P_2, P_4, P_5|, \tag{4.9}$$

$$M_4^\alpha = |P_1, P_2, P_3, P_5|, \tag{4.10}$$

$$M_5^\alpha = |P_1, P_2, P_3, P_4|, \tag{4.11}$$

$$M_1^\beta = |P_2, P_3, P_4, P_6|, \tag{4.12}$$

$$M_2^\beta = |P_1, P_3, P_4, P_6|, \tag{4.13}$$

$$M_3^\beta = |P_1, P_2, P_4, P_6|, \tag{4.14}$$

$$M_4^\beta = |P_1, P_2, P_3, P_6|, \tag{4.15}$$

$$M_6^\beta = |P_1, P_2, P_3, P_4|. \tag{4.16}$$

Substituting the linear dependence (3.2) into (4.7) [33, 34], since the determinant is multilinear in columns,

$$M_1^\alpha = \alpha_1|P_2, P_3, P_4, P_1| + \alpha_2|P_2, P_3, P_4, P_2| + \alpha_3|P_2, P_3, P_4, P_3| + \alpha_4|P_2, P_3, P_4, P_4|. \tag{4.17}$$

Determinants with multiple columns vanish:

$$M_1^\alpha = \alpha_1|P_2, P_3, P_4, P_1| = -\alpha_1|P_1, P_2, P_3, P_4| = -\alpha_1 M_5^\alpha. \tag{4.18}$$

Similar relations between α_i, β_i and M_i^α and M_i^β can be derived as [33, 34];

$$\begin{aligned} \alpha_1 = -\frac{M_1^\alpha}{M_5^\alpha},\ \alpha_2 = \frac{M_2^\alpha}{M_5^\alpha},\ \alpha_3 = -\frac{M_3^\alpha}{M_5^\alpha},\ \alpha_4 = \frac{M_4^\alpha}{M_5^\alpha}, \\ \beta_1 = -\frac{M_1^\beta}{M_6^\beta},\ \beta_2 = \frac{M_2^\beta}{M_6^\beta},\ \beta_3 = -\frac{M_3^\beta}{M_6^\beta},\ \beta_4 = \frac{M_4^\beta}{M_6^\beta}. \end{aligned} \tag{4.19}$$

From (3.21a) and (4.19) the I coefficients can be written as ratios of determinants:

$$I_{12} = \frac{\alpha_1\beta_2}{\alpha_2\beta_1} = -\frac{M_1^\alpha M_2^\beta}{M_2^\alpha M_1^\beta}, \tag{4.20}$$

$$I_{13} = \frac{\alpha_1\beta_3}{\alpha_3\beta_1} = \frac{M_1^\alpha M_3^\beta}{M_3^\alpha M_1^\beta}, \tag{4.21}$$

$$I_{14} = \frac{\alpha_1\beta_4}{\alpha_4\beta_1} = \frac{M_1^\alpha M_4^\beta}{M_4^\alpha M_1^\beta}. \tag{4.22}$$

Since these coefficients are ratios of relative invariants, they are absolute invariants under projective transformation.

4.4 3D-2.5D Watermarking with Projective Invariant

The coefficients given in the previous section enable a watermarking scheme in which data embedded in a 3D model is extracted from any 2D plus depth representation. As was the case in 3D-2D watermarking utilizing perspective invariants, this watermarking scheme is camera parameter independent.

Data embedding process on a 3D model in 3D-2.5D watermarking is identical to that of 3D-2D watermarking (Section 3.2):

1. Select six interest points in 3D $P = \{P_i\}, i \in [1, 6]$.
2. Embed data on 3D interest points by modifying their relative positions; creating different versions of the 3D model.
3. Calculate I coefficients for each version of the 3D model ($S_I = \{I_i\}, i \in [1, n]$, equations (4.20), (4.21), (4.22)).

After 2D plus depth views of different models are generated by using various camera parameters, data is extracted by applying the following steps:

1. Given a 2D view, detect interest points, $p = \{p_i\}, i \in [1, 6]$.
2. Construct the modified depth image coordinates for each interest point. That is, concatenate 2D coordinates with depth image value as the third coordinate, and calculate modified depth image coordinates utilizing the actual depth information.

3. Calculate I coefficients using the modified depth image coordinates of interest points. These detected I coefficients are I_d.

4. Find the I_{min} value that gives the minimum Euclidean difference value in $\|I_i, I_d\|$ for $I_i \in S_I$.

$\|I_i, I_d\|$ above is the Euclidean distance between two sets of I coefficients. I_{min} value shows which version of the 3D model was used. Similar to the method given in the previous chapter, to extract data, watermark detector has to have prior knowledge of S_I.

4.4.1 Selection and Detection of Interest Points

Selection and detection are performed on 2D images. Therefore, problems related to interest point detection and selection explained in Chapter 3 remain. Again, in order to simplify interest point detection problem and accurately judge performance of the perspective invariant, a simple interest point detection scheme is implemented. The following sections briefly remind methods used to detect and select interest points.

4.4.1.1 Detection of Interest Points

Interest points are selected to be centroids of triangular faces. Selected faces are painted with distinct hues (red, green, blue, yellow, cyan, and magenta), while unselected faces are left white. Since triangle centroids are invariant under perspective projection, detection of interest points in 2D views is done by isolating regions of constant hue, and calculating their centers of gravity.

4.4.1.2 Selection of Interest Points

Marking interest points makes detection simpler but interest points still need to be selected carefully. This selection is inevitably related to camera position. Naturally, if an interest point is not visible in a 2D view then the watermark cannot be detected.

As in Chapter 3, a view direction is selected to represent a set of camera positions around that direction. Watermarked 2D views are to be produced in a range around the chosen view

direction. Interest points are selected according to this view direction. Therefore, calculated I coefficients are also related to this view direction. Watermark can be embedded multiple times using different view angles, if detection is required from 2D views that are obtained from camera angles far from the chosen view direction.

Among all the faces of a given model, the following constraints are applied to find a set of candidate interest points that are suitable for watermarking, for a given view direction:

- **Occlusion constraint:** Face centroids should not be occluded.
- **Direction constraint:** Angle between face normal and view angle, that is the vector connecting face centroid to center of a camera positioned at the view direction, should be less than 60 degrees.
- **Adjacency constraint:** Selected faces should not be adjacent, to prevent unwanted motion of adjacent faces during watermark embedding.
- **Invariant constraint:** First four selected faces' centroids should not be coplanar, due to the perspective invariant constraints.

After applying these constraints, remaining faces' centroids are candidates for being used as 3D interest points in watermarking.

4.4.2 Embedding Data

Similar to Chapter 3, data is embedded in a 3D model by changing positions of 3D interest points in a way that is not possible with a projective transformation. If this operation is shown as $P'_i = P_i + \vec{v_i}$, where $\vec{v_i}$ is the translation vector for i-th interest point, watermark energy can be defined as the average of translation vector magnitudes, as in (3.23). Watermark energy is related to 3D mesh distortion and watermark robustness.

Embedding data in a 3D model involves determining $\vec{v_i}$ for a given upper bound of watermark strength. As in Chapter 3, this can be done by methods utilizing heuristics or by optimization.

4.4.2.1 Data Embedding Utilizing Heuristics

In data embedding methods using heuristics, each interest point is translated by a fixed amount along a direction. Translation direction is selected according to view vector ($\vec{c_i}$) and face normal ($\vec{n_i}$). Both $\vec{c_i}$ and $\vec{n_i}$ are defined to be unit vectors.

The following displacement directions are tested:

- Move towards or away from the camera: $\vec{v_i} = s_w \cdot |\vec{c_i}|$.
- Move perpendicular to view vector, parallel to image plane: $\vec{v_i} = s_w \cdot |\vec{c_{i\perp}}|$.
- Move perpendicular to face normal (parallel to face surface) and perpendicular to view vector (parallel to image plane): $\vec{v_i} = s_w \cdot |\vec{n_i} \times \vec{c_i}|$.

4.4.2.2 Optimal Data Embedding

When data is embedded in a 3D model to produce different model versions M_i, and 2D plus depth data from model j, it is expected that $||I_i, I_j|| = 0$ only for $i = j$. In other cases, $||I_i, I_j||$ be greater than zero. To maximize watermark detection performance, this value should be maximized.

On the other hand, a constraint on the amount of displacement of interest points should be imposed to limit the amount of 3D mesh distortion due to watermarking. Thus, the objective function O, that is to be maximized, emerges as given below:

$$O = ||I_i, I_j|| \text{ where } i \neq j,\ |\vec{v_k}| < \epsilon,\ \text{ and } P'_k = P_k + \vec{v_k},\ k \in [1, 6]. \tag{4.23}$$

As in Chapter 3, heuristics can be utilized to reduce the search space dimensions. Different data embedding approaches based on optimization and application of heuristics to reduce search space dimension is given below. P_i denotes 3D interest points.

- **3D optimization:** Optimize in a 3D region centered around P_i. No heuristics applied.
- **2D optimization:** Optimize on the plane perpendicular to view vector, parallel to image plane, centered at P_i.

- **2D optimization:** Optimize on the plane perpendicular to face normal, parallel to face surface, centered at P_i.
- **1D optimization:** Optimize on face normal centered at P_i.
- **1D optimization:** Optimize on a line perpendicular to both face normal and view vector, centered at P_i.

4.4.3 Watermark Detection

As was noted in Section 4.2, mapping 3D points to modified depth image coordinates is a projective transformation. I coefficients, which are invariants under projective transformation, remain unchanged during this mapping. That is, if the 3D point X is mapped to modified depth image coordinates X'_d by the transformation $X'_d = H'_{depth} \cdot X$, and $I(P) = (I_{12}, I_{13}, I_{14})^T$ is a vector of I coefficients related to six 3D points $P = \{P_1, P_2, P_3, P_4, P_5, P_6\}$, then $I(P) = I(P_d)$, where P_d is the set of modified depth image coordinates defined by $P_d = H'_{depth} \cdot P$.

Modified depth image coordinates of an interest point having 2D image coordinates $p_i = (x_i, y_i)^T$ are $(x_i \cdot d_i, y_i \cdot d_i, z(x_i, y_i))^T$ where $z(x, y)$ is the depth map sample at coordinates (x, y) and d is the actual depth value calculated using the formula

$$d_i = z_{far} - (z_{far} - z_{near}) \cdot \frac{z(x_i, y_i)}{255}. \tag{4.24}$$

If the coefficients I_d are calculated from these modified depth image coordinates, I_{min} that minimizes the following relation determines which version of model was used to generate the 2D view depth map pair:

$$I_{min} = \underset{I_i \in S_I}{\arg\min} \, \|I_i, I_d\| \tag{4.25}$$

4.5 Experimental Results

Simulations are performed, similar to those of Chapter 3, in two phases. In the first phase, randomly sampled 3D points independent of a model are used as interest points. Interest

point detection is simulated by adding Gaussian noise to 2D interest point coordinates. In the second phase interest points are marked on 3D mesh models as explained in Section 3.2.1. Interest points are detected utilizing face colors.

Constrained Newton Conjugate-Gradient is used as the optimization method [35, 36]. Optimization regions are defined as cubes, squares and line segments depending on the number of dimensions. Longest axis of the optimization region is set to be watermark energy upper bound value, that is, ϵ in equation 4.23.

4.5.1 Interest Point Detection on Synthetic Data

For each iteration of the experiment, six 3D interest points are randomly selected from a uniform distribution around the 3D origin. A perspective camera, located on the negative z-axis looking towards the origin is used to obtain 2D interest point coordinates and interest point depths. 2D coordinates are then scaled to correspond to pixel coordinates of a square image of 800 pixels on one side. Depth values are quantized to 0-255 levels. Distribution of random 3D points and camera parameters are selected to be compatible with simulations with 3D mesh models. Gaussian noise is added to pixel coordinates and depths to simulate interest point detection and depth calculation.

In this phase, since there are no models and no faces, only the following data embedding methods are simulated:

- 3D Optimization
- 2D Optimization on the plane perpendicular to view vector ($2D - \vec{c_\perp}$)
- 1D Optimization along the view vector ($1D - \vec{c}$)
- Heuristics: Along the view vector ($\vec{c}$ heuristics)
- Heuristics: Perpendicular to view vector ($\vec{c_\perp}$ heuristics)

One bit of information is embedded by creating one additional version of the original model using selected data embedding method. Performance of the proposed system is investigated for a range of watermark energy upper bounds (0.05, 0.1, 0.15, 0.2), viewing angles (0°, 15°, 30°, 45°) and Gaussian noise levels (0.0, 0.5, 1.0, 1.5).

Watermark detection is declared as successful, if

$$\|I_2, I_2'\| < \|I_1, I_2'\| \tag{4.26}$$

where I_1 is obtained from the original model and I_2 from the modified model. I_2' is obtained from a 2D plus depth view of the modified model. For optimization based methods, the following objective function is used:

$$O = \|I_1, I_2\| \text{ where } |\vec{v_k}| < \epsilon, k \in [1, 6]. \tag{4.27}$$

If at any point in the iteration, selected interest points or their 2D counterparts fail to conform to perspective projection constraints given in Section 3.1, iteration is restarted using another set of random interest points.

In Figure 4.1, performance of five data embedding methods is plotted against standard deviation of Gaussian noise added to 2D interest point coordinates, at 0° viewing angle, across all watermark energy levels. Note that, all methods exhibit perfect performance when there is no noise added. Detection performance degrades as noise strength is increased. Optimization based methods have better performance than heuristics based methods for any noise level, with 3D and 2D optimization methods showing the best performance, and 1D optimization closely following. At noise levels of 0.5 and 1.5 pixels standard deviation, detection performance drops to 97.9% and 88.6% for 2D optimization, and 93.3% and 84.3% for $\vec{c_\perp}$ heuristics, which is the better of the two heuristics based methods.

In Figure 4.2, detection rate is plotted against viewing angle (from 0° to 45°) for a noise of 0.5 pixel standard deviation. Throughout this view range, none of the data embedding methods show significant performance change. Compared to 3D-2D watermarking, this is an advantage of using depth information in addition to 2D information in 3D-2.5D watermarking.

Finally, Figure 4.3 displays detection rate against watermark energy at 0° viewing angle and 1.5 pixel noise standard deviation. Heuristics based methods show better performance with increasing watermark energy. Optimization based methods are ahead of heuristics based methods, although they do not exhibit a clear relationship with detection rate. 3D, 2D and 1D optimization methods reach peak detection rates of 99.4%, 98.4% and 97.4%, respectively.

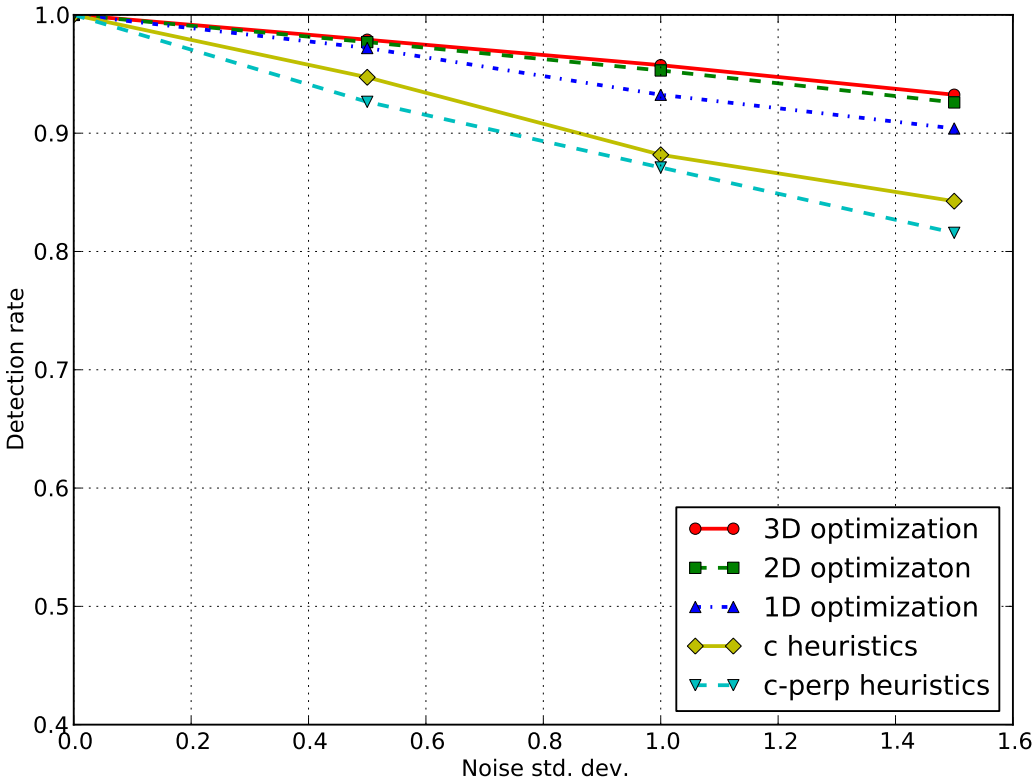


Figure 4.1: Detection rate versus Gaussian noise standard deviation, plotted for different data embedding methods.

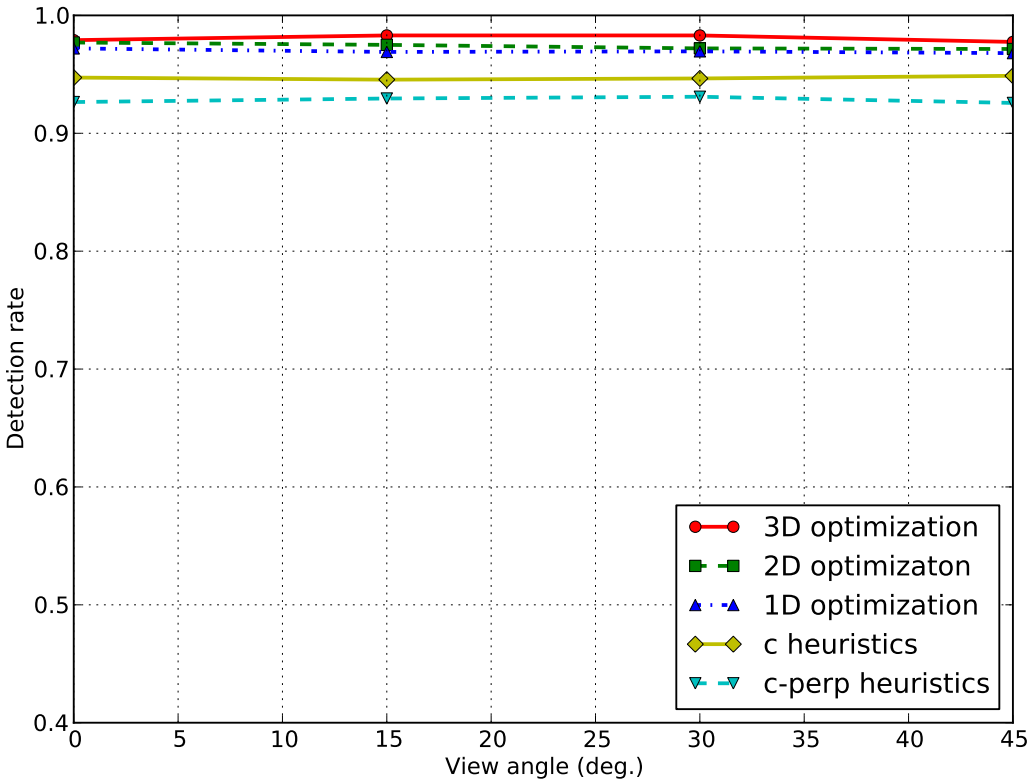


Figure 4.2: Detection rate versus viewing angle, plotted for different data embedding methods.

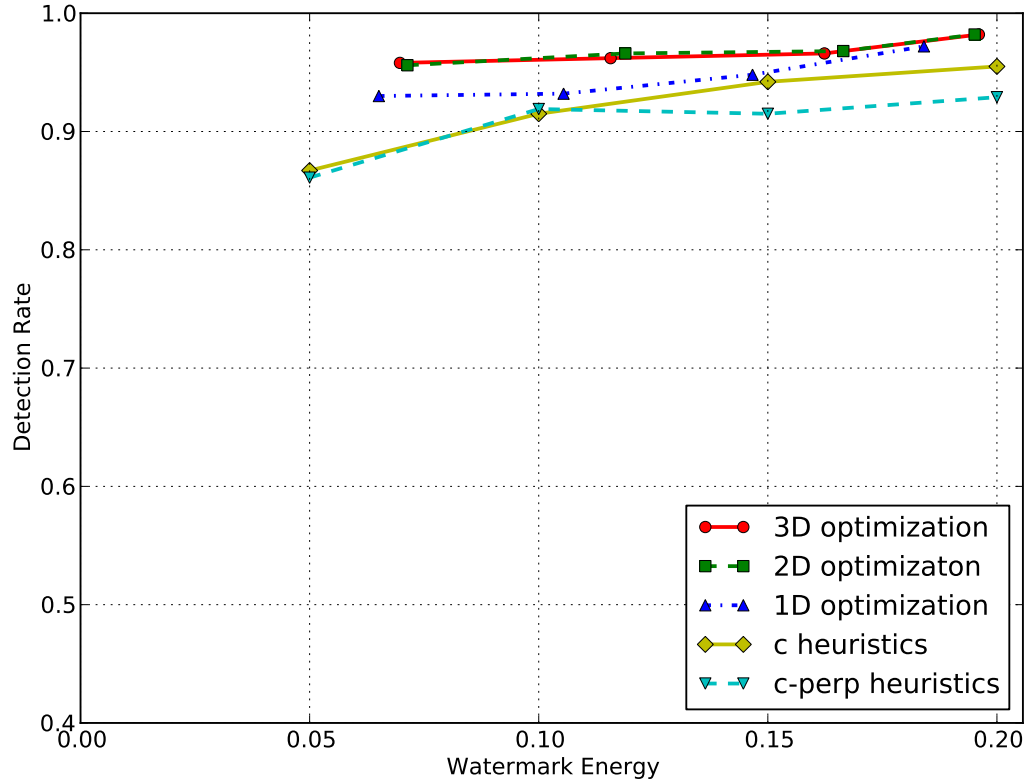


Figure 4.3: Detection rate versus watermark energy (s_w), plotted for different data embedding methods.

$\vec{c_\perp}$ heuristics and $\vec{c}$ heuristics follow with peak detection rates of 95.5% and 92.9%.

These results demonstrate that 3D-2.5D watermarking with the selected projective invariant is feasible. Using depth information has improved performance from different angles. Optimization based methods exhibit better performance, however, the disparity between them and heuristics based methods is smaller than that of 3D-2D watermarking.

4.5.2 Interest Point Detection on 3D Models

In this phase simulations are done on triangular faced mesh models with no textures. As in Chapter 3, mesh models are rendered in Blender 3D software. Rendered 3D scene contains the model to be rendered, a camera and a lamp. 2D views and depth images are rendered with a camera of 35 mm focal length. Camera is rotated around the model to the desired view angle.

All six 3D models introduced in Section 3.3.2 models from Stanford 3D Scanning repository [37] and Aim@Shape repository [38] are used in simulations in this section.

A single iteration of the simulation involves the following steps:

1. Interest points are randomly selected and marked on the model as described in Section 3.2.1. Model-0 is created.
2. Watermark is embedded by moving interest points according to selected watermark energy, as described in Section 4.4.2. Model-1 is created.
3. 2D views and depth images are obtained for the original model (Model-0) and the modified model (Model-1), after rotating the camera by the specified view angles.
4. I coefficients for Model-0 and Model-1 are calculated.
5. Mesh distance values are calculated between Model-1 and Model-0, using the Metro tool [39].
6. Signal to watermark ratios of 2D watermarked views to original model views are calculated.

If not exactly six interest points are detected, or detected interest points violate the perspective invariant constraint given in Section 3.1, the iteration is discarded. The iteration is considered a success, if watermark in a 2D plus depth view generated from Model-1 is successfully detected.

Data is embedded using the following methods:

- 3D Optimization
- 2D Optimization on the plane perpendicular to view vector ($2D - \vec{c_\perp}$)
- 2D Optimization on the plane perpendicular to face normal ($2D - \vec{n_\perp}$)
- 1D Optimization along the face normal ($1D - \vec{n}$)
- 1D Optimization the line perpendicular to face normal and view vector ($1D - \vec{n} \times \vec{c}$)
- Heuristics: Along the line perpendicular to face normal and view vector ($\vec{n} \times \vec{c}$)

Performance of the proposed system is investigated for watermark energy constraint values (for optimization based methods) and watermark energy values (for heuristics based method) of 0.002, 0.01, 0.025, 0.05, 0.075, and 0.1. View angles of 0°, 7.5°, 15° are simulated. Higher watermark energy values produce too pronounced distortion on models and thus were not simulated. Mesh distortion is measured by Hausdorff distance.

Detection rate of all data embedding methods for a viewing angle of 0° are given in Figure 4.4. The x-axis denotes watermark energy (s_w). Results are plotted by accumulating results from all iterations in 6 buckets according to watermark energy. Bucket sizes are selected so that there are equal number of samples in each bucket. Detection rate of each bucket is calculated by dividing the number of successful detections by the number of samples. Detection reates are plotted at average watermark energy for each bucket. This approach prevents erroneous results caused by small number of results in a bucket.

Overall, detection rates for optimization based methods are higher with lower watermark energies when compared to the ($\vec{n} \times \vec{c}$) heuristics method, reaching a performance plateau of 99.5% at 0.005 and 0.01 watermark energy for 3D and 2D optimization methods respectively. 1D optimizing methods have slightly lower peak performance. In contrast, the heuristics method exhibits at best 82.6% detection rate at 0.1 watermark energy. Figure 4.5 is zoomed in to show optimal data embedding methods.

As can be seen in Figure 4.6, similar to results of Chapter 3, data embedding methods that displace interest points perpendicular to face normal (1D optimization along $\vec{n} \times \vec{c}$, 2D optimization on $\vec{n_\perp}$, and $\vec{n} \times \vec{c}$ heuristics) produce the lowest mesh distortion, as expected. Optimization along the face normal results in largest mesh distortion. In general, mesh distortion increases with increasing watermark energy almost linearly.

Figure 4.7 shows detection rate versus view angle characteristics for all data embedding methods. None of the tested data embedding methods exhigit performance degradation in the simulated view range. Largest variation is displayed by $\vec{n} \times \vec{c}$ heuristics method: 1.6% decrease between 0° and 15°

Signal to watermark ratio in generated 2D views is inversely related to watermark energy (Figure 4.8), as expected.

These results demonstrate that optimization based data embedding methods have better wa-

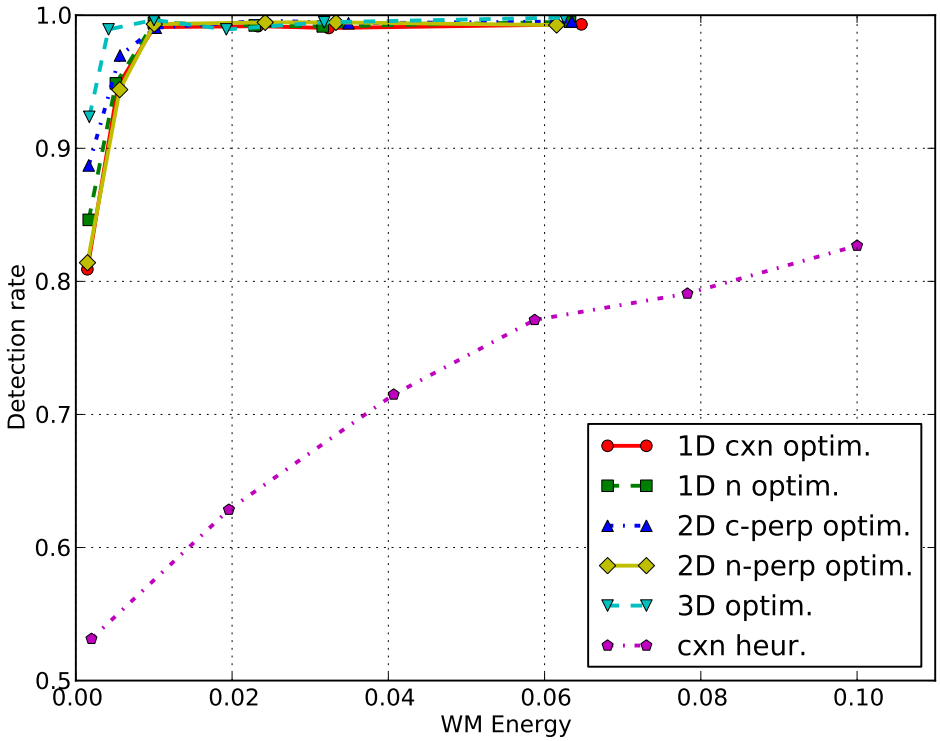


Figure 4.4: Detection rate versus watermark energy bound or watermark energy (s_w or ϵ), plotted for different data embedding methods.

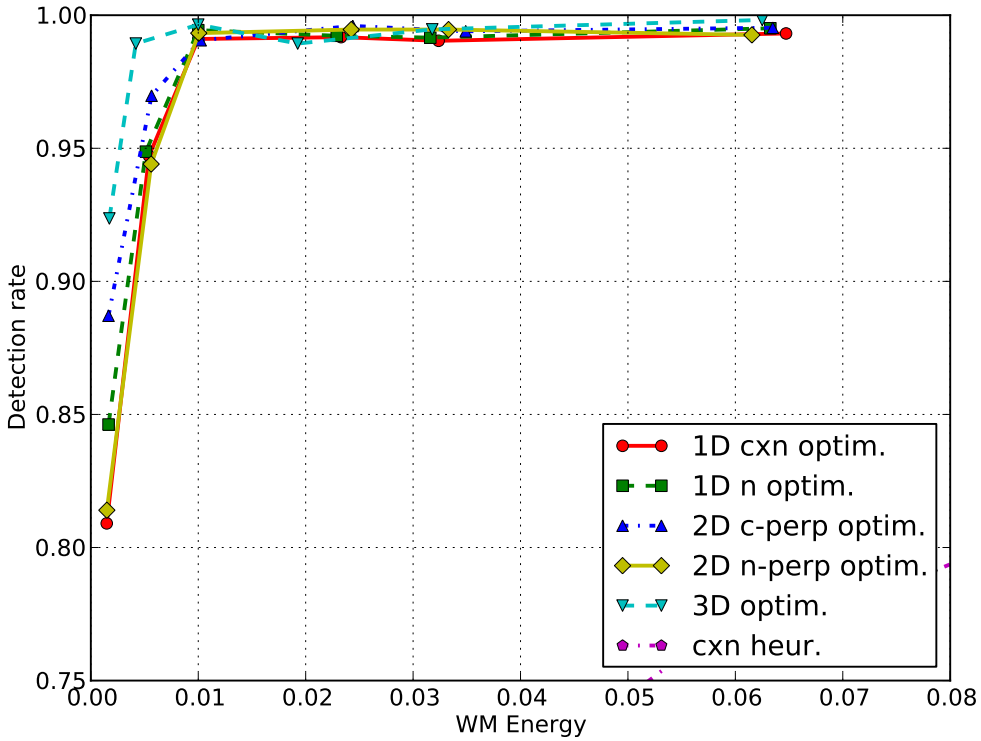


Figure 4.5: Detection rate versus watermark energy bound or watermark energy (s_w or ϵ), plotted for different data embedding methods, zoomed in to show optimization based methods.

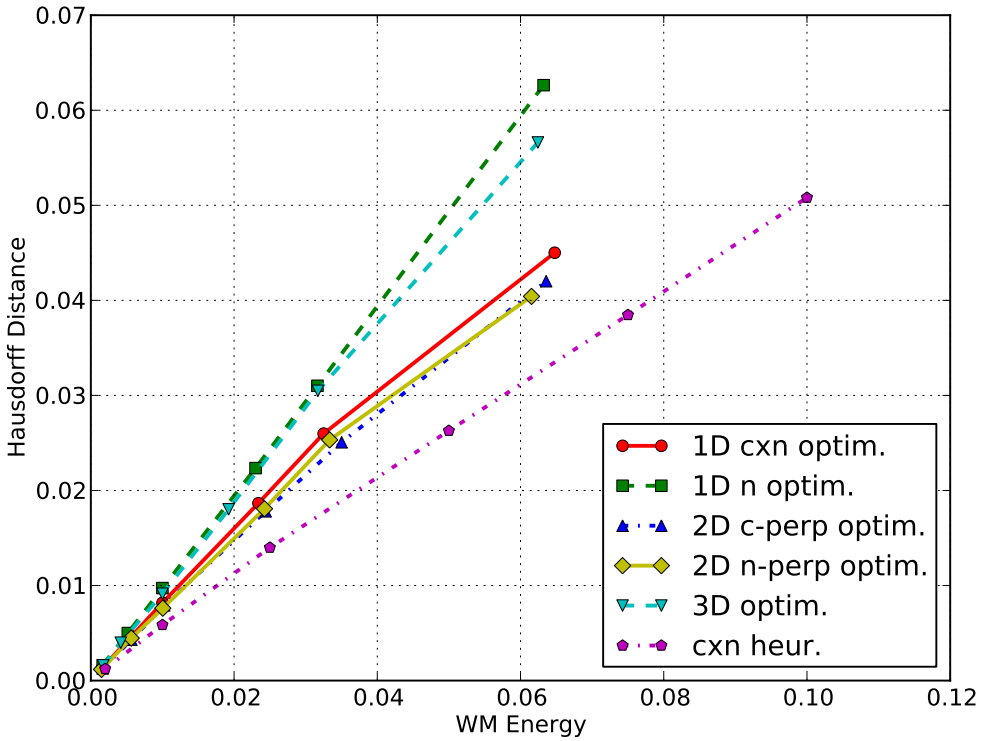


Figure 4.6: Mesh distortion versus watermark energy bound or watermark energy (s_w or ϵ), plotted for different data embedding methods.

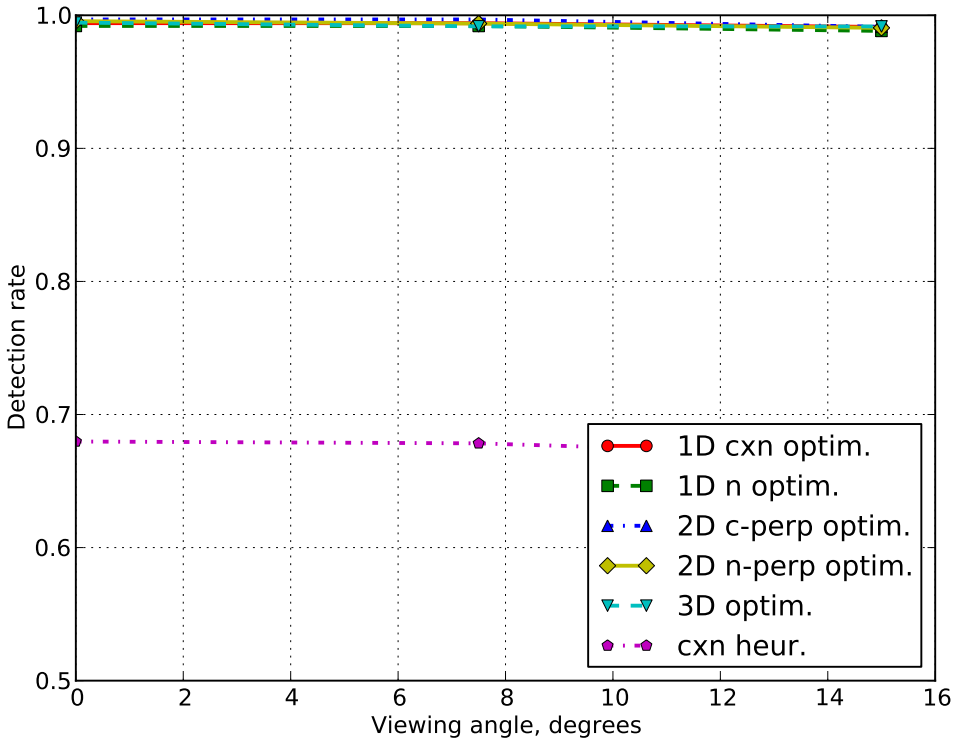


Figure 4.7: Detection rate versus viewing angle, plotted for different data embedding methods.

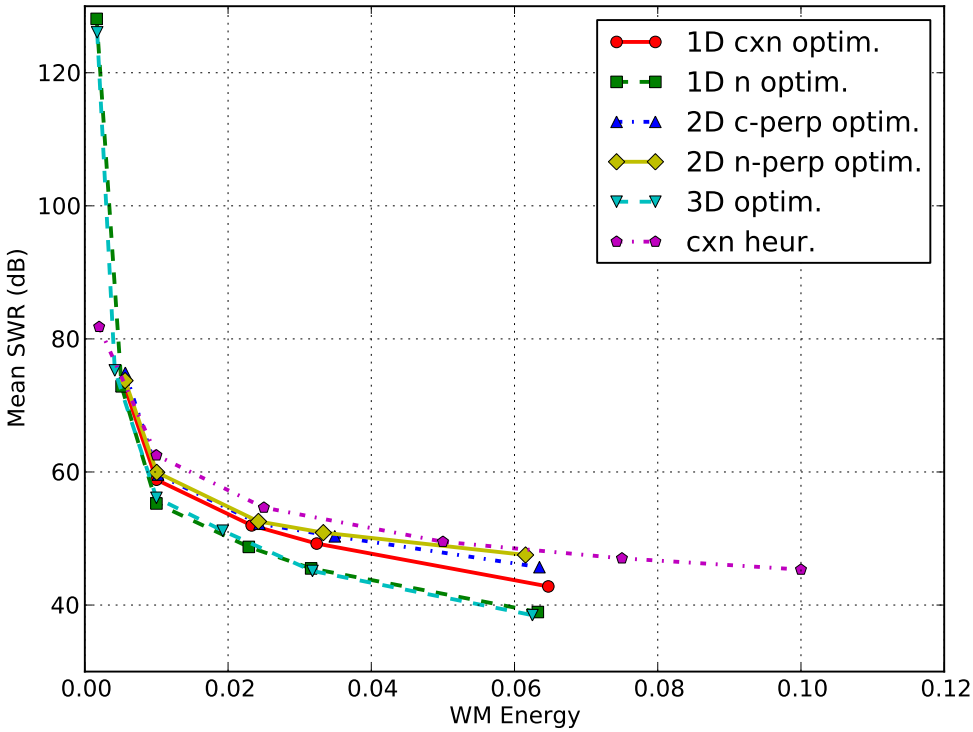


Figure 4.8: Signal to watermark ratio versus watermark energy bound or watermark energy (s_w or ϵ), plotted for different data embedding methods.

termarking performance compared to the heuristics based methods. 3D optimization has the best watermark detection performance, however results in more distortion in 3D mesh and 2D images.

4.5.3 Discussion of Results

In this chapter, as an extension to the 3D-2D watermarking system of Chapter 3, a system that extracts data embedded in 3D models from 2D plus depth representations is presented. A projective invariant that operates in three dimensions is utilized to establish an invariant relation between 3D interest points and corresponding 2D plus depth points. Similar to Chapter 3, data is embedded by displacing 3D interest points.

It was determined that a 3D plus depth representation has to provide actual depth information, for 3D-2.5D watermarking to be possible. Modern depth sensing technology is able to provide this.

Optimization and interest point detection techniques developed in Chapter 3 are utilized for 3D-2.5D watermarking system presented in this chapter. Simulations are done on point clouds and 3D mesh models.

Several observations can be made regarding simulation results:

- Optimally embedded data in 3D mesh models can be extracted from 2D plus depth representations with a detection rate of 99.5%. This performance increases that of 3D-2D watermarking utilizing perspective invariants. Additional information present in 3D plus depth representations raise the performance if invariant based watermarking.
- Optimal data embedding methods in 3D-2.5D watermarking utilize lower watermarking energies to reach better watermarking performance compared to 3D-2D watermarking methods.

These observations strongly suggest using projective invariants in optimal 3D-2.5D watermarking is feasible. Extension of projective invariant based watermarking to extract data from 2D plus depth representations is possible, and the additional depth information increases watermarking performance.

CHAPTER 5

EFFECTS OF REALISTIC 3D MODELS ON INTEREST POINT DETECTION

In Chapters 3 and 4, two watermarking methods, in which data embedded in 3D models were detected in 2D and 2D plus depth representations, were presented. Performance of both methods were demonstrated on synthetic data and 3D mesh models, however, evaluation has not been done on realistic, textured models. Moreover, in the previously presented methods, interest points were marked on models, which may not be suitable for some applications. In such cases, existing features on 3D models should be used as interest points.

Interest point detection for viewpoint independent watermarking is a challenging problem. Requirements for interest points in such a system were given briefly in Section 3.2.1 as being reliably detectable when viewed from different angles, when rendered under different illumination conditions, and after data is embedded. Affine invariant interest point detectors available in the literature may be suitable for viewing angle changes [46] . However, deformation of the model, and distortion in 2D interest point features caused by data embedding poses serious problems.

In addition to detection of interest points, accurate localization of interest points is crucial for watermark detection performance. As demonstrated in Chapter 3, less than 0.5 pixel localization error is necessary for satisfactory performance. Therefore, a successful interest point detector should be able to have good localization performance.

On the other hand, properties of realistic 3D models also influence interest point detection. A 3D model has other components that affect its appearance, apart from the polygonal mesh which gives its shape. Two most familiar components of a 3D model are material and texture

[16]. Material of a model determines its color, reflection model, albedo, transparency and similar properties. Texture of a model can be considered as an image wrapped around the mesh of the model, altering the color of pixels on the surface of the model (see Figure 5.1). A 3D model may also have other components, such as the bump map, which is a texture altering the normal directions of each pixel on the model surface, thereby giving it a bumpy, rough appearance. Parameters of the 3D scene, such as lights, shadows, and the selected rendering method, also affect the model's appearance in 2D views [16]. These properties of a 3D model complicates the problem of detecting embedded data in the polygonal mesh.

In this chapter, the term *viewpoint independent watermarking* is used to mean 3D-2D and 3D-2.5 watermarking utilizing perspective and projective invariants.

5.1 Interest Point Detection for Viewpoint Independent 3D Watermarking

The interest point detection method applied in Chapters 3 and 4, namely marking interest points on the model with distinct hues, is a solution that satisfies the requirements for an interest point detection scheme given above: Colored faces are detectable from different viewing angles, they are differentiable under variable illumination, and they remain detectable after data is embedded, as long as they are not occluded. Centroids of triangular faces are also invariant under perspective transformation, providing good localization performance.

In this chapter, another approach that employing existing features on realistic models as interest points is suggested: Many candidate interest points are identified using a 2D interest point detector on 2D views. These candidate interest points are then filtered according to the requirements given above, that is, interest points that are not detected in all possible views, and those that are not detected after data is embedded are discarded. In other words, interest points which cannot be detected, or from which watermark cannot be detected due to interest point localization errors, are discarded. Remaining interest points are suitable for watermarking. Descriptors of selected interest points are then sent to watermark detector. Watermark detector runs the same 2D interest point detection algorithm on received arbitrary views to detect selected interest points in order.

This approach is formalized in the procedure below. This procedure is given for embedding data with watermark energy s_w in a 3D model M that is to be extracted from 2D views gener-

ated in a view range $(0°, \theta)$.

1. Given the 3D model M, generate 2D views for the angles $0°$ and θ.

2. Run interest point detector on those two 2D views, obtain feature descriptors, and identify the repeatable interest points that are detectable in both views.

3. Simulate data embedding process: Displace repeatable interest points by s_w, determining displacement direction using the $\vec{n}\times\vec{c}$ heuristics explained in Section 4.4.2.1. Obtain a modified model M'.

4. Generate 2D views of M', for the angles $0°$ and θ.

5. Run interest point detector on 2D views of M', obtain feature descriptors, and identify those interest points that are detectable on all views.

6. Select n_{ip} interest points with the lowest detection localization errors as candidate interest points.

7. Obtain 6-permutations of n_{ip} interest points. Embed data in model M using each permutation, obtaining data embedded models $N_i, i \in [1, Perm(n_{ip}, 6)]$.

8. Generate 2D views of models N_i for the angles $0°$ and θ.

9. Attempt to extract embedded data from views. Interest point permutations for which data is successfully extracted from both views are suitable sets of interest points.

Steps between 3 and 6 in the proposed method are utilized to reduce the number of permutations to check in steps 7 to 9. Skipping the simulation steps and directly embedding data in Step 7 results in a prohibitive number of data extraction attempts.

After applying this procedure, a number of sets of interest points in which data can be safely embedded are obtained. More viewing angles can be used at any 2D view generation step to decrease the likelihood of missed interest points due to viewpoint differences, although computation time will be increased.

5.2 Interest Point Detectors

In the following sections, five interest point detector algorithms' eligibility for viewpoint independent watermarking is investigated:

- Scale invariant feature transform (SIFT) [47],
- Maximally stable extremal regions (MSER) [48],
- Features from accelerated segment test (FAST) [49],
- Harris-Laplace detector [50],
- Harris-Affine detector [50].

SIFT is a scale and orientation invariant image feature detector and descriptor [47]. Difference of Gaussians algorithm is applied to smoothed and resampled input image at different scales to detect interest points. Detection scale determines the scale of interest point, and dominant orientation of each interest point is determined. SIFT feature descriptors are obtained from local gradient data by histogramming gradient orientations around the interest point. SIFT feature descriptors are resilient against illumination changes and offer a partial invariance to affine distortion of interest points [47].

The connected components across all possible thresholdings of a gray level image constitute the maximal regions of the image [48]. Some image regions remain stable over a wide range of thresholds. These regions are invariant to affine transformation, are stable, covariant under adjacency preserving transformations, and allow multi scale detection [48]. MSER algorithm detects such maximally stable regions in an image.

FAST feature detection algorithm is an efficient implementation of SUSAN detector [49, 51]. SUSAN identifies corners in an image by calculating the ratio of pixels having different intensity values compared to the interest point in a circular region. SUSAN has good noise suppression capabilities [51].

Harris-Laplace and Harris-Affine are extensions to the Harris corner detection algorithm [52, 50]. Harris-Laplace detector utilizes a scale-adapted version of Harris corner detection to localize points in scale space. Those points which correspond to Laplacian of Gaussian

extrema over scale are selected as interest points [50]. Harris-Affine detector further builds on Harris-Laplace to obtain an affine invariant interest point detector. The algorithm automatically adapts location, shape, and scale of the neighborhood of an interest point to obtain affine invariance [50].

These five interest point detectors are evaluated in the following section.

5.3 Experimental Results

Experiments are conducted in two phases. The first half of the procedure given in Section 5.1, up to Step 6, is used to evaluate performances of five interest point detector algorithms in phase one. The second half, starting from Step 6, uses output from the first phase, and is utilized to evaluate watermarking performance of the suggested method.

5.3.1 Interest Point Detection Experiments

In this section, performance of interest point detection algorithms are tested on a realistic model. A textured and bump mapped 3D model is used during experiments (see Figure 5.1). All interest point detection algorithms are evaluated over different watermark strength values and viewpoints by attempting detection after interest points are displaced. For a given view range $(0°, \theta)$, viewpoint performance of detectors are evaluated by identifying repeatable interest points on views $0°$ and θ, and detecting repeatable interest points on views $0°, \alpha, \theta$ and β where $0° < \alpha < \theta < \beta$.

Performance of each interest point detection algorithm is evaluated through the procedure given below. This procedure is similar to the first half the procedure given in Section 5.1.

1. Given the 3D model M, generate 2D views from angles $0°, 5°, 10°, 15°$.
2. Run interest point detector on the views $0°, 10°$, obtain feature descriptors, and identify the repeatable interest points.
3. Simulate data embedding process: Translate repeatable interest points by s_w, in a direction perpendicular to face normal and view vector ($\vec{n} \times \vec{c}$ heuristics), obtaining a modified model M'.

Figure 5.1: Stone head model and its texture image.

4. Run interest point detector on the views $0°, 5°, 10°, 15°$ for 2D views of the models M and M'. Identify repeatable interest points in each view.

5. Record detection rate and localization performance of interest point detectors.

SIFT descriptors are used for all interest point detectors, and descriptors are matched using the nearest neighbor algorithm [47].

Tables 5.1, 5.2, and 5.3 summarize detection and localization performance of interest point detection methods with view angles of 5°, 10° and 15°, respectively, at 0.025 watermark energy. Number of repeatable interest points is given in the first column. Ratio of successfully detected repeatable interest points is given in the second column. Mean and standard deviation of detected interest point localization errors are given in the next two columns. The final two columns show the localization error values for the best localized and the sixth best localized interest points.

Detection rate of all interest point detectors are sufficient to obtain six interest points for data to be embedded. However, almost none of the interest point detectors have satisfactory localization error means and standard deviations values. This demonstrates that localization is the primary challenge in interest point detection for viewpoint independent 3D watermarking.

Interest point detection error for the best detected interest points is an important performance metric, since, while watermarking, it is possible to select the interest points in whose locations data is going to be embedded. Moreover, since the perspective invariant requires six interest points, the sixth lowest localization error gives a performance measure closely related to watermarking.

According to the last column of each table, FAST emerges as the most suitable interest point detection algorithm, with consistently lowest error values for the sixth interest point. Note that FAST does not have the best localization behavior in terms of the mean and standard deviation values. Good performance in all repeatable interest points is desired, but not required for interest point detection. However, low error values for six or more interest points is required for feasible watermarking.

It should be noted that corner detectors (FAST, Harris-Affine, and Harris-Laplace) have better localization performance when compared to region (MSER) and blob detectors (SIFT), espe-

cially in terms of “best error” and “sixth error” results, as can be seen in the tables. Larger features of blob and region detectors are more prone to localization errors and are less sensitive to small deformations caused by data embedding. Corner detectors, particularly FAST, provides more localized, smaller scaled features.

Table 5.1: Detection and localization performance of interest point detectors at $\mathbf{s_w = 0.025}$ and **5**° view angle.

Detector	Total	Det. Rate	Error Mean	Std. Dev.	Best Error	Sixth Error
MSER	97	0.602	0.820	0.944	0.463	0.977
Harris-Affine	95	0.578	0.990	1.81	0.348	0.661
Harris-Laplace	96	0.656	1.30	1.58	0.497	0.994
FAST	143	0.838	1.81	12.3	0.066	0.279
SIFT	33	0.844	1.77	2.69	0.932	1.20

Table 5.2: Detection and localization performance of interest point detectors at $\mathbf{s_w = 0.025}$ and **10**° view angle.

Detector	Total	Det. Rate.	Error Mean	Std. Dev.	Best Error	Sixth Error
MSER	97	0.649	2.71	15.8	0.353	0.678
Harris-Affine	95	0.500	1.76	3.36	0.197	0.983
Harris-Laplace	96	0.589	1.84	2.77	0.197	0.983
FAST	143	0.719	2.31	13.7	0.144	0.513
SIFT	33	0.727	2.95	5.71	0.786	1.25

Table 5.3: Detection and localization performance of interest point detectors at $\mathbf{s_w = 0.025}$ and **15**° view angle.

Detector	Total	Det. Rate	Error Mean	Std. Dev.	Best Error	Sixth Error
MSER	97	0.505	0.639	0.777	0.297	0.784
Harris-Affine	95	0.430	1.08	2.05	0.530	0.845
Harris-Laplace	96	0.407	1.05	1.39	0.530	0.931
FAST	143	0.570	1.16	6.05	0.220	0.597
SIFT	33	0.655	8.63	26.4	0.627	1.595

In Tables 5.4 and 5.5, detection and localization performances of MSER and FAST detectors at a view angle of 5° and watermark energies of 0.025, 0.05 and 0.1 are given. Best error and sixth error values are increasing with increasing watermark energy. Even with a watermark

energy of 0.05, localization errors reach one and a half pixels. These results indicate that watermarking with existing features on models is only feasible with low watermark energy levels.

Table 5.4: Detection and localization performance of MSER interest point detector at **5**° view angle.

$\mathbf{s_w}$	Total	Det. Rate	Error Mean	Std. Dev.	Best Error	Sixth Error
0.025	95	0.602	0.820	0.944	0.463	0.977
0.05	95	0.602	3.45	10.8	1.43	2.16
0.1	95	0.391	10.17	32.8	4.15	5.12

Table 5.5: Detection and localization performance of FAST interest point detector at **5**° view angle.

$\mathbf{s_w}$	Total	Det. Rate	Error Mean	Std. Dev.	Best Error	Sixth Error
0.025	143	0.838	1.81	12.3	0.066	0.279
0.05	143	0.734	2.02	0.643	1.30	1.66
0.1	143	0.561	4.58	0.672	3.14	4.76

5.3.2 3D-2D Watermark Detection Experiments

Watermarking performance of the FAST method, which is determined to be the most suitable interest point detector in Section 5.3.1, is investigated by embedding data in those interest points that have the least localization error.

Best performing interest points obtained from experiments of Section 5.3.1 are selected to be the candidate interest points in experiments in this section. After data is embedded on these interest points, watermark detection will be attempted from different views. Those interest points through which successful watermark detection occurs are assumed to be suitable for 3D-2D watermarking.

Experiment procedure, which is similar to the second half of the procedure in Section 5.1, is given below.

1. Select n_{ip} interest points with lowest detection localization errors as candidate interest points.

2. Obtain 6-permutations of interest points. Embed data in model M using each permutation, obtaining data embedded models N_i, where $i \in [1, Perm(n_{ip}, 6)]$.

3. Render 2D views of models N_i with angles $0°, 5°, 10°, 15°$.

4. On rendered views, detect interest points, obtain feature descriptors, and find nearest neighbor matches to get interest points.

5. Attempt to extract embedded data from interest points. Interest point permutations for which data is successfully extracted from both views are suitable sets of interest points.

Data is embedded using the $\vec{n} \times \vec{c}$ heuristics method in Step 2, since the same data embedding method is used in selecting the best performing interest points. The interest points that were detected using FAST interest point detector with 0.025 watermark energy are used as input to the experiment. The interest points that were detected both in 0° and 10° views are sorted according to interest point detection errors at 10°, and best n_{ip} are selected. In Step 4, SIFT descriptors obtained from 2D views are matched to descriptors of interest points obtained from 0° views of unmodified model M. n_{ip} is set to be 10, generating $Perm(10, 6) = 151.200$ number of watermarking attempts in Step 2. Random 250 permutations are selected to limit experiment duration.

Watermark detection is attempted on 2D views generated with camera angles of $0°, 5°, 10°$ and 15°. These attempts produce 31, 8, 13, and 9 successes, respectively. Among these successes, 6 are common in 0° and 5°, 5 are common in 0°, 5° and 10°, and only 1 is common in all four views (see Table 5.6, and Figures 5.2 and 5.3).

Table 5.6: Interest point sets enabling watermark detection.

View	**Number of successes**
0° and 5°	6
0°, 5° and 10°	5
0°, 5°, 10° and 15°	1
0° and 10°	10

This result, however pessimistic it seems, demonstrates that it is possible to find a set of interest points that is usable in extracting data embedded in a realistic 3D model from a range of 2D views. Searching for a longer duration is expected to yield more usable sets.

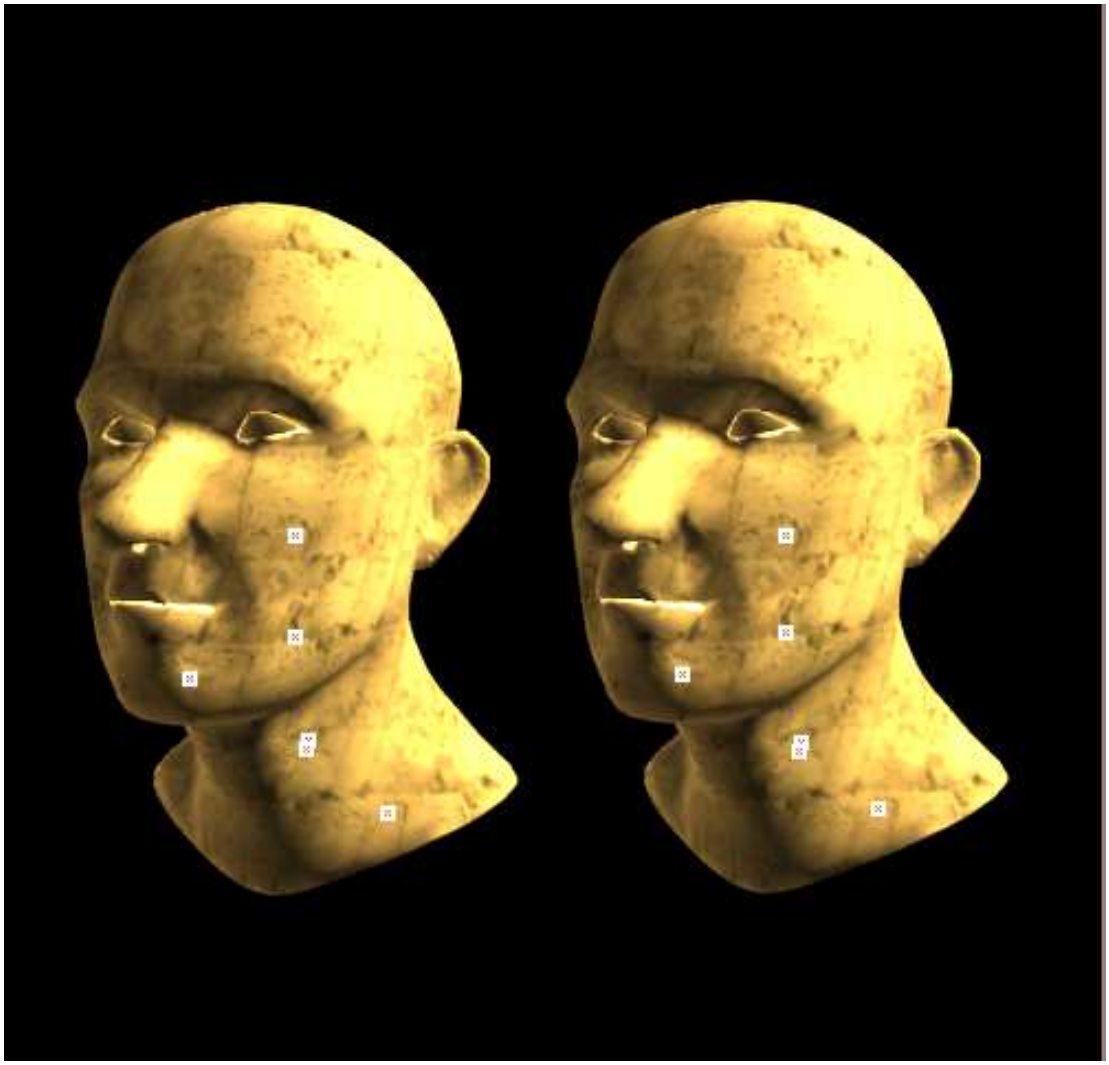


Figure 5.2: Original (left) and data embedded model.

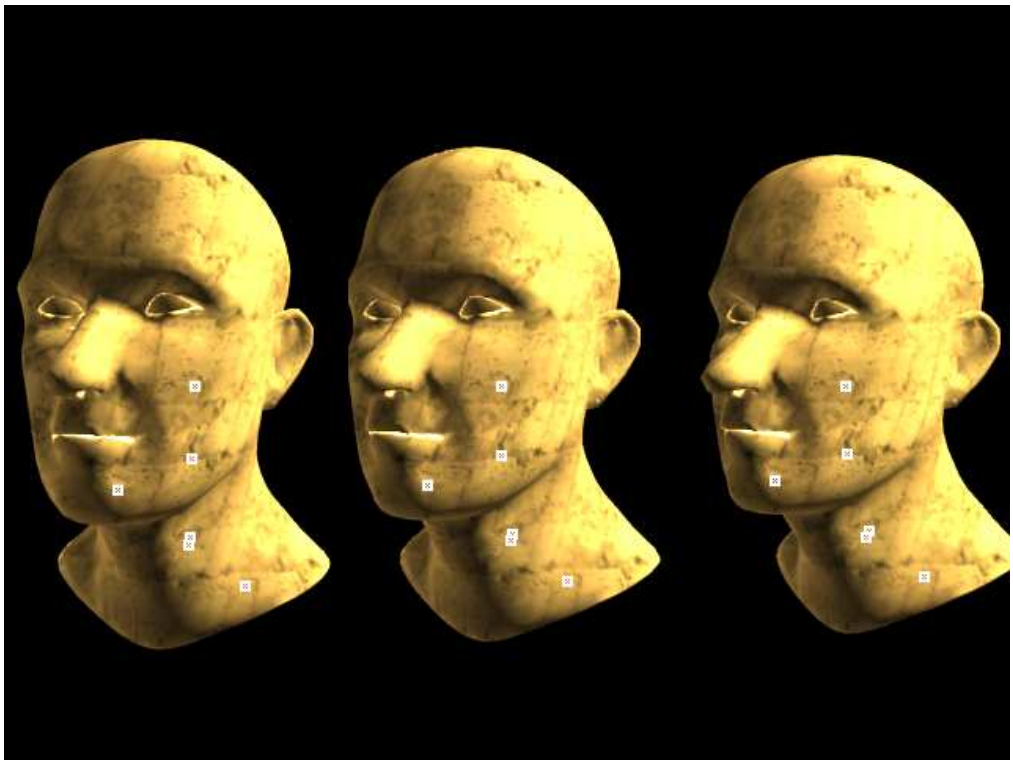


Figure 5.3: Data embedded model viewed from 5°, 10° and 15°.

A number of factors lead to the large number of watermark detection misses. First of all, a permutation of interest points may not be suitable for watermarking. They may violate perspective invariant constraints (Section 3.1), or they may be so close to each other that embedding data in them may prevent detection due to distortion. Moreover, even with the best interest point detector examined, interest point localization errors are still a degrading factor. The watermark energy level, kept low to reduce interest point localization errors, also reduces the watermark detection performance. And finally, the utilized data embedding method, chosen for consistency with the simulation step in the previous section, is the lowest performing data embedding method.

Since interest point descriptors are obtained at 0° view angle, a decrease in performance is observed as view angle increases. Another point to note is that since initial repeatable interest points are selected using the views 0° and 10° (in the previous section), these two views have more successes and common successes.

5.4 Discussion of Results

Interest point detection for viewpoint independent 3D watermarking is a challenging problem. The challenge mainly stems from the fact that interest points detected in 2D views are back projected to 3D models, and modified to embed data. It is difficult to detect these modified interest points using different viewpoints, under varying rendering conditions.

In this chapter, a number of interest point detectors are evaluated to determine those that are suitable for 3D-2D watermarking. Localization performance is observed to be the primary restraining factor, rather than repeatability. Among six methods, FAST corner detector is observed to have the most suitable interest point detection and localization performance. It is interesting that FAST corner detection does not detect all interest points with the smallest amount of error, but it produces a few interest points that have smaller localization errors than other methods.

An interest point selection method that filters out those interest points that are not detectable after view changes or data embedding, and then selects the points that have the smallest localization error is presented. Watermarking is attempted by the remaining interest points. Those interest points from which embedded data can be extracted are suitable for viewpoint independent watermarking on realistic models.

Simulation results show that data can be embedded in realistic models utilizing features already existing on the 3D model, in other words, without marking interest points on the model. This result is an indicator of feasibility of 3D-2D and 3D-2.5D watermarking on realistic models, and expands the application domain of 3D-2D and 3D-2.5D watermarking with perspective and projective invariants.

CHAPTER 6

SUMMARY and CONCLUSIONS

This dissertation introduces a novel approach to 3D-2D watermarking, based on using geometrical invariants to embed data. Two different classes of invariants are used throughout the dissertation: 3D projective invariants and 3D-2D perspective projection invariants. Literature on invariants is mainly concerned with using invariants in object detection. In the dissertation, it is demonstrated that invariants are usable in watermarking.

Projective invariants and perspective invariants are viewpoint independent. Therefore, watermarking methods developed in this dissertation, utilizing invariants, are also viewpoint independent. This is a major feature of the proposed methods, enabling data to be embedded in a 3D model once, and detected from any 2D view of it.

First method proposed, which is given in Chapter 3, is a 3D-2D watermarking method using a perspective invariant with minimal constraints. Data is embedded in relative positions of six 3D interest points, and detected from rendered views of the model, independent of camera position. Two main problems of a 3D-2D watermarking system utilizing invariants are identified: Noise sensitivity of the invariant, and repeatable interest point detection.

An objective function considering the invariant is developed to address the noise sensitivity. This objective function is maximized to find optimal 3D interest point displacements. To address the repeatable interest point detection problem, a basic interest point detection that marks interest points on models is utilized. Simulations are done on synthetic data and 3D meshes.

Next, an extension to the 3D-2D watermarking method that extracts data from 2D plus depth representations is proposed in Chapter 4. This watermarking method works on an image and

an associated range data of a 3D scene to extract data embedded on the 3D model. Interest point positions, detected from the 2D image, are combined with depth information obtained from range data to detect the watermark using projective invariants. Optimization and interest point detection techniques developed for the first method are utilized to enhance performance. Simulations are done on synthetic data and 3D meshes.

The problem of repeatable interest point detection in viewpoint independent watermarking is attacked in Chapter 5. Interest points have to be accurately detectable not only in multiple views of the 3D model, but also after data is embedded. Different 2D interest point detector algorithms' detection and localization performances are evaluated in a scenario adapted to requirements of a viewpoint independent watermarking system.

An interest point detection process tailored to viewpoint independent watermarking is presented, and its applicability is demonstrated on a realistic, textured 3D model. The interest point detection method developed in this chapter allows utilizing interest points available on 3D meshes.

The use case for 3D-2D and 3D-2.5D watermarking systems is justified when one considers that 3D content is often consumed in 2D. Embedding data in 3D models before generation of 2D images and videos is more efficient than watermarking generated 2D views. Moreover, in some use cases, like free viewpoint video or computer games, 2D content is rendered at the client. In such a case, watermarking 2D content in the client is less secure than embedding the watermark in 3D models before transmission. 3D-2D and 3D-2.5D systems will be more attractive because they can handle these use cases. The main contribution of this dissertation is to demonstrate that 3D-2D and 3D-2.5D watermarking using projective invariants is feasible, opening up an entirely new area of research.

6.1 Conclusions

The idea of utilizing projective and perspective invariants in watermarking systems that extract data embedded in 3D models from 2D and 2.5D representations is examined in this dissertation.

The 3D-2D watermarking method presented in Chapter 3 exhibits very strong performance on

3D mesh models when optimal data embedding methods are used. Optimal data embedding methods also reduce mesh and image distortion. Performance of the system indicates that perspective invariants based optimal 3D-2D watermarking, a completely new area of research, is feasible.

Extension of the 3D-2D watermarking method to 2D plus depth representations presented in Chapter 4 exhibits increased performance, by utilization of additional depth information. The results indicate that projective invariant based 3D-2.5D watermarking is also feasible.

The interest point detection method suggested in Chapter 5 is shown to be successful in utilizing existing features on realistic 3D models as interest points. This is the last necessary enhancement for the methods introduced in the previous chapters to be feasible in realistic application scenarios.

All methods presented in this dissertation point towards new directions in watermarking research.

6.2 Future Work

Although 3D-2D and 3D-2.5D watermarking using invariants is shown to be feasible, the proposed methods can be enhanced in various ways. First of all, proposed methods are not yet evaluated against attacks on 2D views. Techniques should be developed especially against attacks that aim to prevent accurate interest point detection. Cropping, blurring, video coding, and similar traditional attacks' effects on performance should be investigated. For use cases in which the model is transmitted to the client, techniques against 3D attacks on models should also be investigated.

Methods to embed more than one bit in a 3D model should be developed and evaluated. Such methods' effects on performance and mesh and 2D view distortion should be investigated.

The idea of using perspective invariants based watermarking is shown to be feasible using a general, weakly constrained invariant. However, for some use cases, other invariants may be more suitable. For example, when two views are transmitted to the receiver, an invariant that relates the two views can be used to embed data. Similarly, multi-view invariants can be used to hide data in successive frames of a video. For specific use cases, alternative, maybe more

constrained perspective and projective invariants should be researched.

REFERENCES

[1] A. Smolic, K. Mueller, N. Stefanoski, J. Ostermann, A. Gotchev, G.B. Akar, G. Triantafyllidis, and A. Koz. Coding algorithms for 3dtv - a survey. *Circuits and Systems for Video Technology, IEEE Transactions on*, 17(11):1606 –1621, nov. 2007.

[2] Alper Koz. *Watermarking for 3D representations*. PhD thesis, Middle East Technical University, August 2007.

[3] Aljoscha Smolic, Philipp Merkle, Karsten Müller, Christoph Fehn, Peter Kauff, and Thomas Wiegand. Compression of multi-view video and associated data. In Haldun M. Ozaktas and Levent Onural, editors, *Three-Dimensional Television*, Signals and Communication Technology, pages 313–350. Springer Berlin Heidelberg, 2008.

[4] Bloom J. A. Cox I. J., Miller M. L. *Digital Watermarking*. Morgan Kaufmann Publishers, USA, 2002.

[5] J.-L. Dugelay and Daoudi M. Baskurt A., editors. *3D Object Processing: Compression, Indexing and Watermarking*, chapter 4. 3D Object Watermarking. John Wiley & Sons Ltd., England, 2008.

[6] I.J. Cox, M.L. Miller, and J.A. Bloom. Watermarking applications and their properties. In *Information Technology: Coding and Computing, 2000. Proceedings. International Conference on*, pages 6 –10, 2000.

[7] G.C. Langelaar, I. Setyawan, and R.L. Lagendijk. Watermarking digital image and video data. a state-of-the-art overview. *Signal Processing Magazine, IEEE*, 17(5):20 –46, sep 2000.

[8] Tong Liu and Zheng ding Qiu. The survey of digital watermarking-based image authentication techniques. In *Signal Processing, 2002 6th International Conference on*, volume 2, pages 1556 – 1559 vol.2, aug. 2002.

[9] C. Fehn. A 3d-tv system based on video plus depth information. In *Signals, Systems and Computers, 2003. Conference Record of the Thirty-Seventh Asilomar Conference on*, volume 2, pages 1529 – 1533 Vol.2, nov. 2003.

[10] R.I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[11] Joseph L. Mundy, Deepak Kapur, Stephen J. Maybank, Patrick Gros, and Long Quan. Geometric invariance in computer vision. chapter Geometric interpretation of joint conic invariants, pages 77–86. MIT Press, Cambridge, MA, USA, 1992.

[12] Christopher Coelho, Aaron Heller, Joseph L. Mundy, David A. Forsyth, and Andrew Zisserman. Geometric invariance in computer vision. chapter An experimental evaluation of projective invariants, pages 87–104. MIT Press, Cambridge, MA, USA, 1992.

[13] C. Fehn, P. Kauff, M. Op De Beeck, F. Ernst, W. IJsselsteijn, M. Pollefeys, L. Van Gool, E. Ofek, and I. Sexton. An evolutionary and optimised approach on 3d-tv. In *In Proceedings of International Broadcast Conference*, pages 357–365, 2002.

[14] C. Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *ACM Trans. Graph.*, 23:600–608, August 2004.

[15] Jonathan Shade, Steven Gortler, Li-wei He, and Richard Szeliski. Layered depth images. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '98, pages 231–242, New York, NY, USA, 1998. ACM.

[16] Uğur Güdükbay and Funda Durupınar. Three-dimensional scene representations: Modeling, animation, and rendering techniques. In Haldun M. Ozaktas and Levent Onural, editors, *Three-Dimensional Television*, Signals and Communication Technology, pages 165–200. Springer Berlin Heidelberg, 2008.

[17] Suk-Hwan Lee, Tae-Su Kim, Seung-Jin Kim, Young Huh, Ki-Ryong Kwon, and Kuhn-Il Lee. 3d mesh watermarking using projection onto convex sets. In *Image Processing, 2004. ICIP '04. 2004 International Conference on*, volume 3, pages 1577 – 1580 Vol. 3, oct. 2004.

[18] Oliver Benedens and Christoph Busch. Towards blind detection of robust watermarks in polygonal models. *Computer Graphics Forum*, 19(3):199–208, 2000.

[19] M.G. Wagner. Robust watermarking of polygonal meshes. In *Geometric Modeling and Processing 2000. Theory and Applications. Proceedings*, pages 201 –208, 2000.

[20] Han Sae Song, Nam Ik Cho, and Jong Weon Kim. Robust watermarking of 3d mesh models. In *Multimedia Signal Processing, 2002 IEEE Workshop on*, pages 332 – 335, dec. 2002.

[21] Satoshi KANAI, Hiroaki DATE, Takeshi KISHINAMI, and Satoshi Kanai Hiroaki Date. Digital watermarking for 3d polygons using multiresolution wavelet decomposition. In *Proc. Sixth IFIP WG 5.2 GEO-6*, pages 296–307, 1998.

[22] F. Uccheddu, M. Corsini, and M. Barni. Wavelet-based blind watermarking of 3d models. In *Proceedings of the 2004 workshop on Multimedia and security*, MM&Sec '04, pages 143–154, New York, NY, USA, 2004. ACM.

[23] Bennour J. and Dugelay J.-L. Watermarking of 3d objects based on 2d apparent contours. In *Proc. of SPIE-IS&T Electronic Imaging*, volume 6072. SPIE, 2006.

[24] E. Garcia and J.L. Dugelay. Texture-based watermarking of 3d video objects. *Circuits and Systems for Video Technology, IEEE Transactions on*, 13(8):853 – 866, 2003.

[25] R.I. Hartley. Projective reconstruction and invariants from multiple images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(10):1036 –1041, October 1994.

[26] J. Brian Burns, Richard S. Weiss, and Edward M. Riseman. Geometric invariance in computer vision. chapter The non-existence of general-case view-invariants, pages 120–131. MIT Press, Cambridge, MA, USA, 1992.

[27] YuanBin Wang, Zhang Bin, and Yu Ge. The invariant relations of 3d to 2d projection of point sets. *Pattern Recognition Research, Journal of*, 3(1):14–23, 2008.

[28] I. Weiss. Projective invariants of shapes. In *Computer Vision and Pattern Recognition, 1988. Proceedings CVPR '88., Computer Society Conference on*, pages 291 –297, jun 1988.

[29] J. Brian Burns, Richard S. Weiss, and Edward M. Riseman. Geometric invariance in computer vision. chapter The non-existence of general-case view-invariants, pages 120–131. MIT Press, Cambridge, MA, USA, 1992.

[30] Y. Zhu, L. D. Seneviratne, and S. W. E. Earles. New algorithm for calculating an invariant of 3d point sets from a single view. *Image and Vision Computing*, 14(3):179 – 188, 1996.

[31] C. A. Rothwell, D.A. Forsyth, A. Zisserman, and J. L. Mundy. Extracting projective structure from single perspective views of 3d point sets. In *Views of 3-D Point Sets Proc. of 4:th ICCV*, pages 573–582, 1993.

[32] S. J. Maybank. Relation between 3d invariants and 2d invariants. *Image and Vision Computing*, 16(1):13 – 20, 1998. Geometric Modeling and Invariants for Computing Vision.

[33] Isaac Weiss and Manjit Ray. Model-based recognition of 3d objects from one view. In Hans Burkhardt and Bernd Neumann, editors, *Computer Vision -ECCV 98*, volume 1407 of *Lecture Notes in Computer Science*, pages 716–732. Springer Berlin / Heidelberg, 1998. 10.1007/BFb0054775.

[34] I. Weiss and M. Ray. Model-based recognition of 3d objects from single images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(2):116 –128, feb 2001.

[35] S. G. Nash. Newton-Type Minimization via the Lanczos Method. *SIAM Journal on Numerical Analysis*, 21:770–788, August 1984.

[36] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–.

[37] The stanford 3d scanning repository. http://graphics.stanford.edu/data/3dscanrep/, September 2011.

[38] Aim@shape shape repository. http://shapes.aimatshape.net, September 2011.

[39] P. Cignoni, C. Rocchini, and R. Scopigno. Metro: Measuring error on simplified surfaces. *Computer Graphics Forum*, 17(2):Pattern Recognition Research, Journal of, 1998.

[40] Zhengyou Zhang. Microsoft kinect sensor and its effect. *IEEE MultiMedia*, 19(2):4–10, April 2012.

[41] Igor Barnov, Dimitry Burov, et al. KinectHacks, 2010–.

[42] Leonard McMillan. *An Image-Based Approach To Three-Dimensional Computer Graphics*. PhD thesis, University of North Carolina at Chapel Hill, 1997.

[43] Ning Zhu, Guiguang Ding, and Jianmin Wang. A novel digital watermarking method for new viewpoint video based on depth map. In *Intelligent Systems Design and Applications, 2008. ISDA '08. Eighth International Conference on*, volume 2, pages 3 –7, nov. 2008.

[44] E. Halici and A.A. Alatan. Watermarking for depth-image-based rendering. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 4217 –4220, nov. 2009.

[45] Yu-Hsun Lin and Ja-Ling Wu. A digital blind watermarking for depth-image-based rendering 3d images. *Broadcasting, IEEE Transactions on*, 57(2):602 –611, june 2011.

[46] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65:2005, 2005.

[47] D.G. Lowe. Object recognition from local scale-invariant features. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 1150 –1157 vol.2, 1999.

[48] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761 – 767, 2004. British Machine Vision Computing 2002.

[49] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*, volume 1, pages 430–443, May 2006.

[50] Krystian Mikolajczyk and Cordelia Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[51] S. M. Smith and J. M. Brady. Susan - a new approach to low level image processing. *International Journal of Computer Vision*, 23:45–78, 1995.

[52] C Harris and M Stephens. A combined edge and corner detector. *Proc 4th Alvey Vision Conference*, 1988.

VITA

PERSONAL INFORMATION

Surname, Name: Yaşaroğlu, Yağız

Nationality: Turkish (TC)

Date and Place of Birth: 15 October 1980, Ankara

Email: yagiz.y.phd@gmail.com

EDUCATION

Degree	Institution	Year of Graduation
MS	METU Electrical and Electronics Engineering	2003
BS	METU Electrical and Electronics Engineering	2001
High School	ODTÜ Geliştirme Vakfı Özel Lisesi, Ankara	1997

WORK EXPERIENCE

Year	Place	Enrollment
2006-Present	Mobilus Ltd.	Founder
2002-2006	Tübitak Bilgi Teknolojileri Enstitüsü	Senior Researcher
2001-2002	ASELSAN	Research Engineer

FOREIGN LANGUAGES

Advanced English.

PUBLICATIONS

- Yağız Yaşaroğlu and A. Aydın Alatan, “Extracting Embedded Data in 3D Models from 2D Views Using Perspective Invariants”, 3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON), May 2011.

- Yağız Yaşaroglu and A. Aydın Alatan, “Summarizing Video: Content, Features, and

HMM Topologies", Visual Content Processing and Representation, 8th International Workshop, September 18-19 2003, Madrid, Spain.

- Özgür Önür and Ersin Esen and A. Aydın Alatan and Medeni Soysal and Serhat Tekinalp and Yağız Yaşaroğlu, "A MPEG-7 compliant Video Management System: BilVMS", Proceedings of the 4th European Workshop on Image Analysis for Multimedia Interactive Services, 2003.