University of Nevada, Reno

Third Generation 3D Watermarking:Applied Computational Intelligence Techniques

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science and Engineering

by

Mukesh C. Motwani

Dr. Frederick C. Harris, Jr./Dissertation Advisor

July, 2011

Abstract

With the explosion of multimedia content over Internet, there is a need for copyright protection of digital content. Whether it is music albums swapped over peer to peer networks or video files uploaded over websites such as `YouTube.com` or 3D models such as Shrek, artists need to protect their ownership of content. 3D Watermarking provides a deterrent to piracy of 3D models by embedding a hidden piece of information in the original content.

Watermarking algorithms have a basic requirement that the watermark should be imperceptible to avoid being detected and not cause visible distortion to the viewer. The watermark should also be robust to withstand unintentional attacks. It is also desired that the watermark insertion capacity should be as high as possible to withstand intentional attacks and to allow insertion of multiple or redundant or biometric watermarks. Insertion of high density imperceptible watermark will make it extremely difficult for an attacker to find the watermark and then make substantial changes in the 3D model to remove or overwrite the watermark. However, inserting large amounts of information as watermark can cause distortion. The design of watermarking algorithms involves a trade off between imperceptibility, capacity and robustness.

The first generation of 3D watermarking techniques inserted low capacity watermark based on spatial geometry and have poor robustness. The second generation of algorithms explored use of multi-resolution transform to insert the watermark and improve the robustness. This dissertation explores use of computational intelligence techniques to build third generation watermarking algorithms, that insert robust, high density watermarks and go the extra mile in terms of hiding more information than the first and second generation techniques. The focus of this study is to optimize the energy of the watermark and intelligently selecting information pockets in 3D model for watermark insertion and at the same time still maintaining randomness in the process to avoid detection. Use of Fuzzy Logic, Genetic Algorithms, and Artificial Neural Networks are proposed and assessed.

Acknowledgments

My sincere thanks go to my advisor, Dr. Frederick C. Harris, Jr. for his endless support and guidance over the past several years. I am also thankful to my committee members: Dr. Sergiu Dascalu, Dr. Bobby Bryant, Dr. William Kuechler, and Dr. Daulatram Lund for their time and feedback.

My gratitude goes to my younger sister, Rakhi, who dedicated countless number of hours in discussion and reading of my work to provide invaluable feedback and suggestions. I owe special thanks to my parents and friends for their unconditional love, encouragement and support. I would like to thank my cat, Billo for spending countless sleepless nights with me while writing this dissertation. I would also like to thank my students who contributed towards this work.

Last but not the least, I am thankful to God for giving me the potential to complete this work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

3D graphic models find widespread uses in movies, architecture, gaming, virtual reality, computer aided design (CAD), simulation of organs in medicine, military, and bioinformatics just to name a few. With the advent of 3D TVs into households and affordable 3D video cards for desktop computers, 3D models are becoming even more omnipresent. With the prospect of 3D printers becoming a household reality, its just a matter of time that the 3D digital content creation market will grow into a multi-billion dollar industry. There are websites [1, 3] which allow artists to upload their artwork and sell these 3D models. Princeton University even provides a search engine [7] to query for 3D models by shape descriptors. There are a number of tools to create these 3D models such as [4], [6] and [2]. However, designing and building high end 3D graphic models requires considerable skill and the use of specialized software and/or hardware such as laser scanners. With high demand and popularity of 3D models and considering the cost, time, and effort required to build such models comes the menace of widespread illegal copying of 3D models. Watermarking is a technique which deters illegal copying by inserting a hidden message in the 3D model.

1.1 Motivation

Digital Rights Management (DRM) techniques based on encryption technologies have been used to in past to prevent copying of digital multimedia only to be decrypted and unlocked within few months if not weeks or days. Encryption algorithms used are independent of the format of the multimedia, whether it is audio, video, album songs, movies, ebooks, or 3D models. Numerous copies of the 3D Shrek model (from the movie Shrek) can be found online. The 3D Gollum character from Lord of the Rings movie can also be found online. Hollywood suffers millions of dollars in lost revenue annually for movies which are pirated and sometimes distributed even before the movies are released in theaters. Prevention of multimedia copying has not only proven to be difficult but its also difficult to trace the pirates or the origin of breach in the distribution chain. In spite of strong legislation in United States with the introduction of Digital Millennium Copyright Act (DMCA) and the Recording Industry Association of America (RIAA) bringing lawsuits, illegal copying of multimedia continues till this day. DRM systems attempt to provide an anti-piracy framework that restricts the use of content to its rightful user. The failure of encryption based DRM can be highlighted with Apple abandoning its FairPlay DRM within iTunes in January 2009.

Watermarking is a technique which inserts a message or code within the digital content. Secure Digital Music Initiative (SDMI), a consortium of music-industry companies, held a challenge in 2000 to test the strength of their watermarking technologies. Edward Felton *et al.* and his team from Princeton and Rice University [8] defeated all four watermarking algorithms thereby proving watermarks by themselves are ineffective. However, there has been considerable amount of significant research since 2000. In 2008, Fox Studios started using an on-demand watermarking system to automatically and seamlessly embed imperceptible forensic information in every frame of video content to protect against piracy. Although encryption and watermarking techniques when used separately have been proven ineffective, when used together within a DRM framework, supplement each other and can be a formidable deterrent for illegal copying. For example, $X - MenOrigins : Wolverine$ movie in April 2009 was illegally distributed with visible watermarks ‘Rising Sun Pictures’ on Peer to Peer (P2P) networks and eventually an arrest was made for the person who uploaded the movie on $megaupload.com$ [85]. Although, the presence of the watermark did not apparently assist in the arrest, nevertheless it still serves as a deterrent. Its now common practice for movie studios to add watermarks to their pre-release, in-house versions of productions to track an original uploader.

The state of research of watermarking 3D models is still in its infancy as compared to published work in image and video watermarking. However, lessons learned from the movie industry indicate that watermarking is a viable technology which is here to stay and can be extended to 3D models. It is not a common practice for amateur artists today to insert watermarks when selling their original 3D content to the seller and they trust the seller to not violate the ownership rights of the artist. Although sellers of 3D models give royalties to artists for ownership of 3D models, there are no business models to support redistribution of 3D models which protect intellectual rights of the artist. There is not much work done related to insertion of multiple watermarks to support such a reseller distribution business model. Thus, there is a business need for strong 3D watermarking algorithms.

1.2 Objective

Typically, very small amount of random information is inserted as watermark. The lower the energy content of the watermark, the easier it is to potentially remove or destroy the watermark. If more information is inserted as a watermark, the more difficult it is to remove the watermark since the attacker will have to identify all the locations where the watermark was inserted and the amount of watermark inserted at each location. Inserting high density watermarks further denies the attacker ability to insert their own watermark to overwrite or destroy the original watermark without causing perceptible distortion. Recently, biometric prints [69, 67] have been explored to be used as a watermark for insertion in 3D models to implement a DRM solution to ensure fair rights management such that the needs of both the artists and consumers are balanced. Such biometric prints can be large in size and further justify the need for watermarking algorithms to have the capability to insert imperceptible large size watermarks. 3D models when streamed through noisy transmission channels can corrupt parts of the data thereby destroying the inserted watermark. There is a need to insert redundant copies of the watermark information for error handling in case of unintended modification during transmission. Inserting redundant copies implies that the watermarking algorithm has a high embedding capacity. The objective of this dissertation is to explore innovative ways to insert the maximum amount of secret information into 3D mesh models without causing perceptual distortion and also make it difficult for the attacker to guess where the watermark was inserted and the amount of watermark inserted.

1.3 Challenges

A watermark inserted in a 3D model is equivalent to adding noise to the 3D model, which in effect could cause perceptible distortion. Inserting multiple watermarks or biometric watermarks is a very challenging problem since higher amounts of secret information need to be inserted in the 3D models without causing perceptible distortion in viewing of the model. To ensure imperceptibility of the modification caused by watermark embedding, a perceptibility criterion needs to be used. However, the Human Visual System (HVS) for 3D models is not well understood and has not been mathematically modeled. It would be optimal to embed a watermark just below the threshold of perceptual distortion. However, this threshold is difficult to determine and can vary for individual to individual. As a consequence, the host data is usually modified by an amount relatively small to their average amplitude in order to avoid perceptible distortion.

It is also required that for the watermark to be robust, the watermark should survive unintentional attacks such as compression loss, affine transformations, smoothing, and noise. At the same time, the watermark should survive intentional attacks by an attacker to remove or overwrite the watermark. The watermark should be perceptually invisible and should be embedded in such a way so that the potential hacker should be forced to make substantial changes in the 3D multimedia content in order to destroy the watermark. For high robustness it is desirable that the watermark amplitude is as high as possible so that an attacker cannot simply remove the watermark by applying a noise filter or a low pass filter. It is also desired for high robustness that the watermark is spread through the model so that the attacker cannot easily identify the watermark locations.

The requirements of having high capacity and robustness are in direct conflict with the requirement of being imperceptible. Thus, there is a need for an algorithm to insert high energy robust watermarks which are imperceptible at the same time. Thus, there is a need to maximize the amount of watermark information to be inserted in 3D mesh without causing perceptible distortion, and without being detected by an attacker to determine where and how much the watermark information was inserted. This dissertation addresses this challenge and focuses on optimally inserting high density watermarks in 3D models without causing perceptible distortion to the human eye viewer.

1.4 Methodology

Invisible watermarking is achieved by insertion of secret binary data in the 3D model. To deter brute force attacks, information is typically inserted randomly and scattered throughout the 3D model. To insert higher amounts of information, computational intelligence techniques have been explored. The proposed approach not only exploits the geometry of the location where the secret information is to be inserted, but also exploits local geometry of the surrounding locations to optimize the amount of information to be inserted. The novelty of the proposed work lies in the approach of viewing watermarking as an optimization and classification problem and use of computational intelligence techniques such as fuzzy logic, genetic algorithms, and neural networks to optimally select the locations where the watermark needs to be inserted and the amount of information to be inserted by exploiting the local geometry of the 3D model. Fuzzy Logic has been proposed to use heuristic rules for identifying locations for the watermark and the amount of watermark to be inserted. Evolutionary techniques such as genetic algorithms have been explored to adaptively add the amount of information to be inserted based on the local geometry. Neural Networks have also been explored to select the locations for watermark insertion.

1.5 Dissertation Organization

This dissertation is organized as follows: Chapter 2 gives a brief overview of watermarking and categorizes related work into different generations of watermarking techniques. Chapter 3 gives background of watermarking and describes the rationale for using computational intelligence techniques in the proposed Third Generation watermarking. Chapter 4 describes the Fuzzy Logic proposed approach with a brief overview of the theory, detailed description of algorithm, experiments with analysis of heuristic rules used. Chapter 5 includes a brief introduction to Genetic Algorithms (GA) with detailed information on how GA's are used for watermarking and followed by parameter analysis. Chapter 6 gives a brief overview of Artificial Neural Network (ANN) architectures and explain how ANN's are used for watermarking and then concluding with experiments. Chapter 7 compares and analyzes the techniques used in the previous chapters. Limitations, conclusions, and future research directions are also covered in this chapter.

Chapter 2

Background and Related Work

2.1 Background

A 3D mesh consists of three combinational entities: vertices, faces, and the edges connecting the vertices. A vertex list gives the co-ordinates in 3D space of each and every vertex in the model and a face list which describes how the vertices are connected to each other. An edge list can be derived by traversing the face and vertex list. Figure 2.1 is an example of a wireframe mesh.

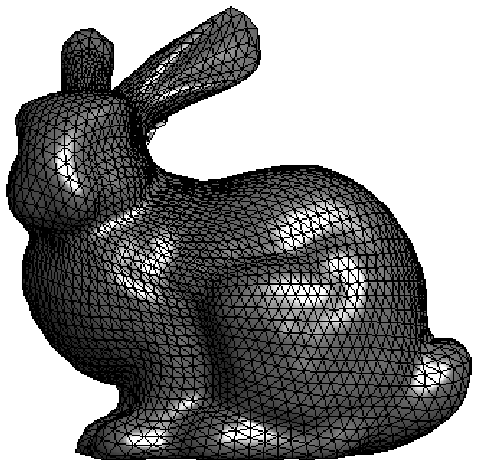


Figure 2.1: 3D Mesh

Watermarking techniques embed imperceptible data into the multimedia content. The covertly embedded data is called the watermark and may consist of a users unique ID, cryptographic keys, copyright ownership messages, access conditions of the content, logos, image, biometrics, or content-based information. The watermark embedding and retrieval process is assisted by a secret key, in which lies information on where and to what extent has the original content been modified in order to accommodate the watermark. Imperceptibility is a strong requirement of every watermarking scheme, because the watermark should not distort the original media or interfere with its intended use or function. Robustness is necessary to assure that common signal processing, geometric operations, and malicious modifications do not impact the detection or retrieval of the watermark. The objective is to facilitate content owners to prove their ownership by retrieving the watermark from a pirated media and then litigate against the offender. Figures 2.2 and 2.3 show the two components of a watermarking system: embedder and detector.

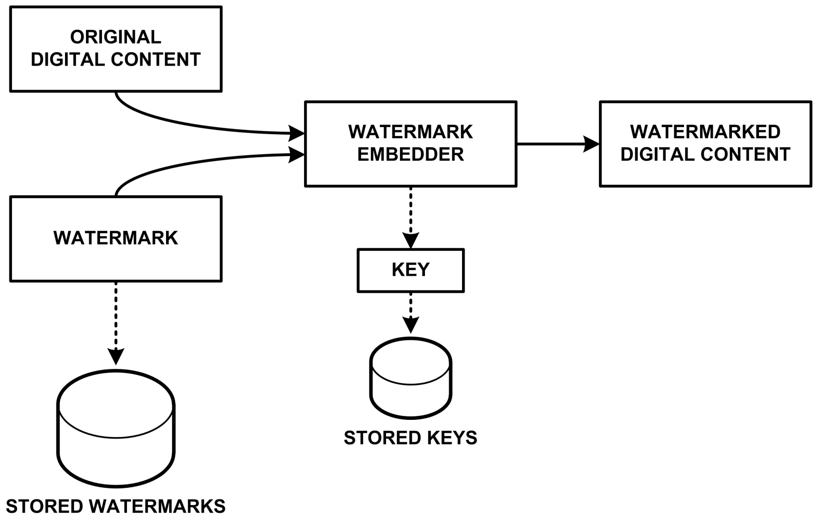


Figure 2.2: Watermark Insertion

The watermark detection process can be non-blind (retrieval process requires access to any of the original content), semi-blind (detector requires access to some side information and/or the watermark but not the original content), or blind (detection is performed without access to the original content). Non-blind detection methods are more robust but impractical for use in DRM systems. Since non-blind tech-

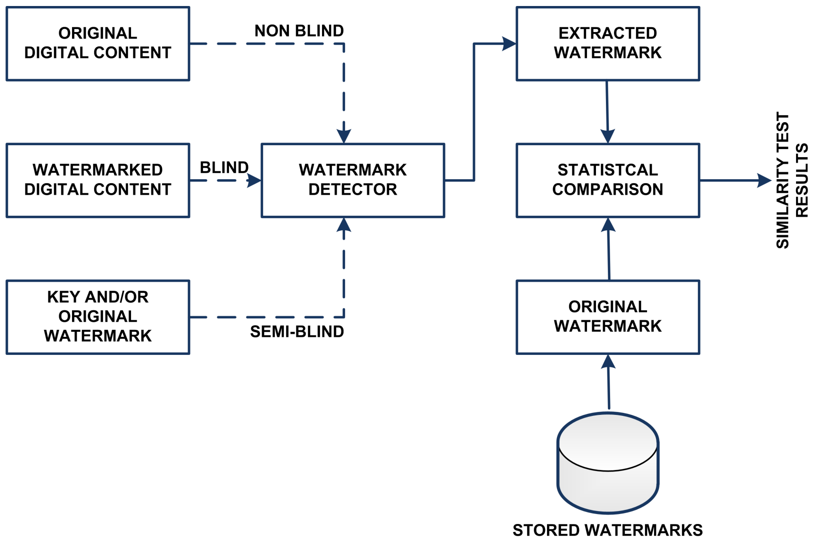


Figure 2.3: Watermark Detection

niques require that the original content is available to the detector, that necessitates access ability to the original content from the consumers end of the DRM system software, which creates a security hole in the system. Semi-blind techniques are most appropriate for use in this context as blind techniques compromise the robustness requirement.

Watermarking algorithms are classified into public/blind and private/non-blind depending on whether the original model is available to check for ownership infringement. The requirement of being able to detect the watermark without the original watermark introduces a very challenging problem especially if robustness is also desirable. This dissertation focuses on applications where the original watermark is available for detection. Such a scheme is practical for distribution of multiples copies of the watermarked model and identification of end users, where the content provider would like to identify the watermark in case of illegal use and trace the watermark back to the appropriate end-user. Our algorithm operates on 3D models that have a mesh representation.

Watermarking algorithms can be also be broadly classified into first, second and third generations of watermarking. Existing watermarking algorithms in this dissertation have been classified in two generations of watermarking algorithms. The generations are classified on the basis of capabilities of the algorithms. With advancing generations, watermarking algorithms have greater embedding capacity without causing perceptible distortion and still being robust.

Figure 2.4 shows classification of algorithms into generations based on the watermark insertion domain. Third generation techniques build on existing first and second generation algorithms and also include hybrid domains to allow information fusion from different domains. The proposed third generation algorithms explore the use of computational intelligence techniques to insert a high capacity watermark in both the spatial and transform domains.

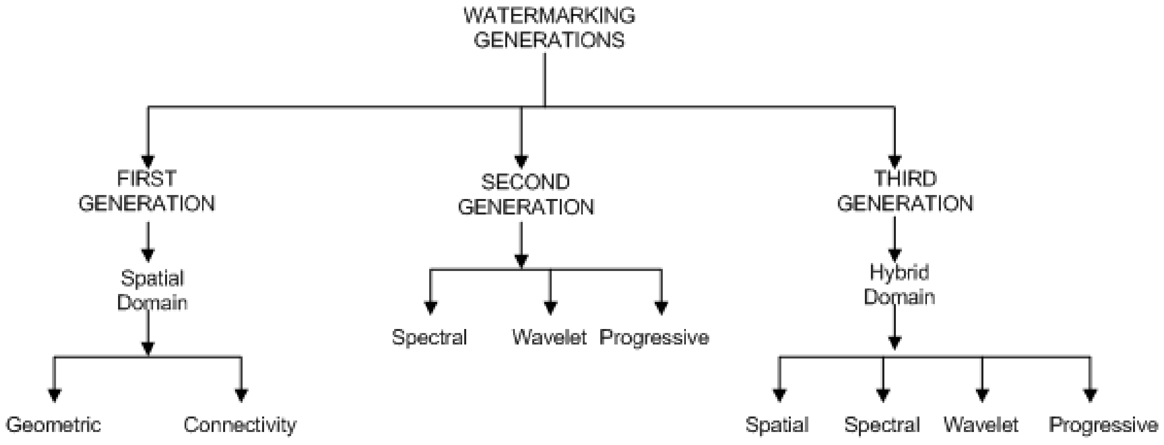


Figure 2.4: Watermarking Generations

Background of watermarking has been covered in depth with survey of 3D watermarking techniques [9, 93], so only a brief overview will be presented here.

2.2 First Generation Watermarking

Figure 2.5 shows block diagram of first generation watermarking algorithms. Features are extracted in the spatial domain. The 3D watermarking schemes which embed data in the spatial domain may be classified in two main categories: Connectivity-driven watermarking schemes and Geometry-driven watermarking schemes. The spatial domain watermarking schemes are usually less robust to attacks like compression and noise addition. They however survive cropping attack and are less complex. The first generation algorithms insert watermark in spatial domain by either modifying vertex positions or modifying the connectivity of the vertices.

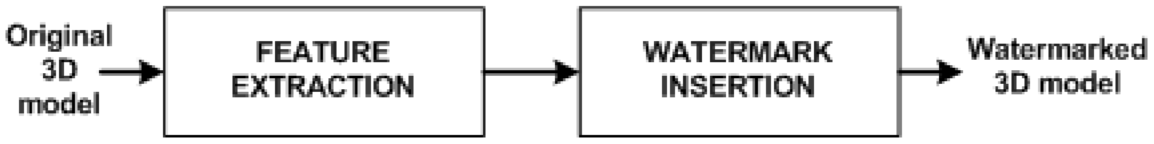


Figure 2.5: First Generation Watermarking

2.2.1 Connectivity Driven

Connectivity-driven watermarking algorithms are those which make an explicit use of the mesh connectivity (some authors also refer to these as topological features). These schemes are typically based on traversal of all the mesh triangles. For each triangle satisfying an admissibility function, slight modifications are introduced in local invariants by changing the adjacent point positions. As a consequence, these are sensitive to noise addition modifications. Among this class of watermarking schemes, Ohbuchi *et al.* [73] have proposed four different watermarking algorithms in the first work published on 3D watermarking. These schemes are respectively named Triangle Similarity Quadruple (TSQ), Tetrahedral Volume Ratio (TVR), Triangle Strip Peeling Sequence (TSPS) and Macro Density Pattern (MDP).

Mohsen Ashourian and Reza Enteshary [11] insert a random watermark based on a masking factor at vertex positions. The masking factor is based on the estimate of average difference between position and connected vertices. The capacity of inserted watermark is 100 bits and the algorithm was tested on two models with the number of faces ranging from 3,500 to 5,000 faces. The non-blind watermarking algorithm was proven to have robustness against additive noise, MPEG4 compression, and mesh simplification attacks. However, the robustness of the algorithm was due to the repeated insertion of the watermark and using error redundant codes and thus the low embedding capacity.

2.2.2 Geometry Driven

This section presents the 3D watermarking schemes which embed data in the geometry. These schemes modify the point positions and/or the face normal. Point normal is estimation of the local continuous surface normal and represents the local shape of the mesh.

Thomas Harte and Adrian G. Bors [41] presented the simplest spatial domain method for embedding a watermark in 3D models. Their algorithm shifts a vertex from its regular neighborhood away from the surface if bit 0 is to be embedded and towards surface if bit 1 is to be embedded. Though the algorithm is very simple, and does not require the original object in the detection, one vertex is used for embedding one bit which reduces the capacity of the watermark to a large extent. The watermark is also not robust to many geometric transformation attacks.

2D image watermarking techniques have inspired algorithms that project 3D information onto a 2D image. Wang Ying *et al.* [99] projected 3D meshes onto to 2D geometry images. This method is robust to remeshing and vertex reordering attacks. Bennour *et al.* [15, 16] proposed a framework for watermarking 3D object via their contour information.

Zhen Li, WeiMin Zheng, and ZheMing Lu [54] proposed an algorithm which allows embedding public watermarks containing copyright information into 3D models consisting entirely of triangle meshes. In this algorithm, watermarks are embedded into a 3D model by altering model vertices with weights and along directions that are all adaptive to the local geometry. The vertex selection is done on the basis of its distance from its neighboring vertices.

Oliver Benedens' [14] bin encoding algorithm is the most successful algorithm in this generation. Benedens [13] modifies the normal distribution to store invisible information in the model's geometry. An orientation histogram is constructed for this purpose by sampling the normals. The watermarks show significant robustness against mesh simplifying methods. In another approach based on curvature tensor, Alface and Macq [10] automatically selects robust feature points for watermark insertion. Bin Yang *et al.* [97] combined principal component analysis (PCA) with the construction of cone bins. Preliminary work using normals for curvature estimation for this dissertation was published in [68].

2.3 Second Generation Watermarking

There have been several enhancements since the first generation to improve the performance in terms of capacity, invisibility, and robustness of the watermark. Figure 2.6 shows block diagram of second generation watermarking algorithms. The second generation algorithms use various transformations to insert the watermark in the transform domain coefficients to enhance robustness. In the second generation, spectral decomposition and multi resolution techniques such as wavelet transform and progressive meshes are used to decompose a 3D model into a lower resolution and the watermark is inserted in the bit stream. Second generation algorithms thus enable applying the watermarking approach to streamed meshes and increased the robustness of the algorithm by inserting the watermark at multiple resolutions.

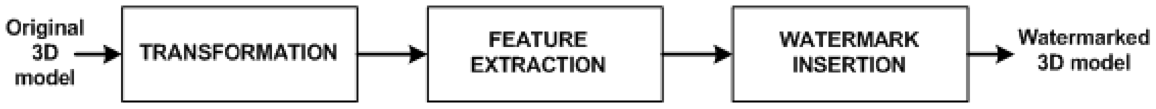


Figure 2.6: Second Generation Watermarking

2.3.1 Multiresolution Decomposition

The wavelet transform gives a multiresolution representation of a 3D model. The watermark is inserted in the 3D model at each level of the wavelet transform. Thus, the watermark exists at even lower resolutions of the 3D model. This gives two major advantages over conventional approaches: First, it helps in defending a down sampling attack. In a down sampling attack, the attacker down samples the 3D model and thus reduces the resolution in order to remove the watermark. But as the watermark is inserted even in the lower resolution of 3D model, it does not get destroyed. Thus, the multiresolution approach makes the watermark robust against attacks. Second, as the watermark is added even at lower resolutions the amount of watermark added is more than the watermark added without multiresolution analysis. This increases the capacity of the watermark which increases robustness against attacks such as smoothing, cropping, and addition of noise. As a limitation, wavelet based methods require the mesh to have 1-to-4 subdivision connectivity. Wavelet transforms can only be applied on semi-regular connectivity meshes because of the quaternary subdivision/simplification process. This drawback has been later solved by [90] who extend this scheme to irregular meshes.

Kanai *et al.* [49], in 1998, proposed the first watermarking method based on wavelet analysis with a non-blind detection scheme. Kanai is the first person to apply transformed domain approaches to 3D models. It works in the mesh's wavelet transformed domain. Kani's algorithm first decomposes a 3D polygonal mesh by using lazy wavelets induced on 3D polygonal meshes. He then modified the wavelet coefficients to embed a watermark. Their watermarks are resistant to affine transformation and random noise added to the vertices.

Yin *et al.* [98] have adopted the scheme to perform multiresolution decomposition. Watermark information can be embedded into some spatial kernels of the low-frequency component of the shape. This strategy actually deals with the low-resolution representation in the geometry hierarchy which, however, does not play the same role as the low-resolution components in the frequency domain. Unlike embedding a bit into the low-frequency domain, embedding a watermark bit into a vertex of the coarse mesh does not mean that the bit has been embedded globally into the low-frequency components of the whole mesh. For that reason the scheme is not robust against crop operations.

Ohbuchi *et al.* [74, 73] introduced several schemes for watermarking polygonal models. One scheme embeds information using groups of four adjacent triangles: they perturb the vertex coordinates to obtain certain desired values for ratios of edge lengths in the group or for ratios of triangle height over triangle base. Another scheme proposed uses ratios of tetrahedral volumes.

2.3.2 Spectral Decomposition

Spectral representation of mesh can be used for progressive transmission and is vastly superior to existing spatial domain techniques. In 2000, Karni and Gotsman [50] showed how spectral methods can be applied to 3D mesh data to obtain compact representations. To reduce complexity, the mesh is partitioned into sub-meshes. Due to partitioning there could be edge effects, which is a limitation of this technique. Since then several approaches have been explored for watermarking using spectral decomposition. Cayre *et al.* [26] introduced spatial overlapping for the spectral representation to overcome this limitation. Ohbuchi *et al.* in [75] embeds message bits by deforming the 'low-frequency' components of the mesh by using mesh spectral analysis. Ohbuchi *et al.* in [74] then extended mesh-spectral analysis to 3D point cloud. Since spectral mesh requires connectivity, a non-manifold mesh was generated to derive connectivity. In [56], and [57], Luo and Bors performed principal component analysis(PCA) of spectral coefficients and used integral moments as feature to insert 1 bit ensuring minimal distortion.

2.3.3 Progressive Mesh

Hoppe proposed progressive meshes in [42, 43] by generating lower resolution models of the mesh by performing a sequence of vertex split operations. Praun and Hoppe [34] reported robust mesh-watermarking algorithm that works using Hoppe's progressive meshes and is applicable to polygonal meshes having arbitrary vertex connectivity. Praun's method modified the shape of the mesh by using a spatial kernel to embed information in the low frequency component of the shape. Their watermarks are resistant against similarity transformation, smoothing, additive random noise, and other attacks. In addition, their watermarks are resistant against mesh simplification and other operations that preserve shape but modify vertex connectivity, by recreating the connectivity of the reference (i.e. original) mesh on the watermarked (and possibly attacked) mesh by means of mesh alignment followed by resampling. The embedding consists in the displacement of the points in normal or reverse-normal directions accordingly to the watermark value. Recently, in 2010 Chen and Chen [27] inserted watermarks in a progressive mesh.

Thus, in the second generation, multi resolution techniques such as wavelet transform and progressive meshes were used to decompose a 3D model in to multiple resolutions and the watermark is inserted in the bit stream. The model is then reconstructed from lower resolutions. Such capability allows for a watermark to be inserted in streamed data. With movies being download through broadband Internet from websites such as `netflex.com` gaining widespread acceptance, the progressive streaming of 3D meshes is going to gain more importance. Second generation watermarking algorithms are more suited for such kind of applications and overall are more robust and have higher capacity as compared to first generation techniques.

2.4 Third Generation Watermarking

Third generation watermarking techniques build on the first and second generation techniques by adding an intelligent layer of optimization for high density watermark insertion. Thus, these algorithms can be extended to be used on streaming meshes as well. Figure 2.7 shows block diagram of second generation watermarking algorithms. Watermarking can be viewed as an optimization problem where the objective is to maximize the number of vertices to watermark and also maximize the amount of watermark to insert without causing perceptible distortion. Third generation techniques also extends to algorithms for fragile watermarking. The novelty of this dissertation lies in evaluating computational intelligence techniques to solve high density watermark insertion as an optimization problem. Evolutionary techniques such as genetic algorithms have been included in this generation. Fuzzy Logic and Neural Network based watermarking algorithms are also included in this new breed of algorithms. There is not much published work in this generation class except for [45], in which Hu *et al.* used quadratic programming (QP) for constrained optimization of 3D meshes. Hu proposed a histogram-based method for watermarking 3D polygonal meshes by using quadratic programming to minimize the mean square error between the original mesh and the watermarked mesh. However, this method has difficulties in dealing with large meshes because of the complexity limitations of existing QP solvers. There is no published work which explores use of genetic algorithms, fuzzy logic or artificial neural networks for 3D watermarking applications. However, genetic algorithms (GA), fuzzy logic (FL), and artificial neural networks (ANN) have been used for image and video watermarking with partial success.

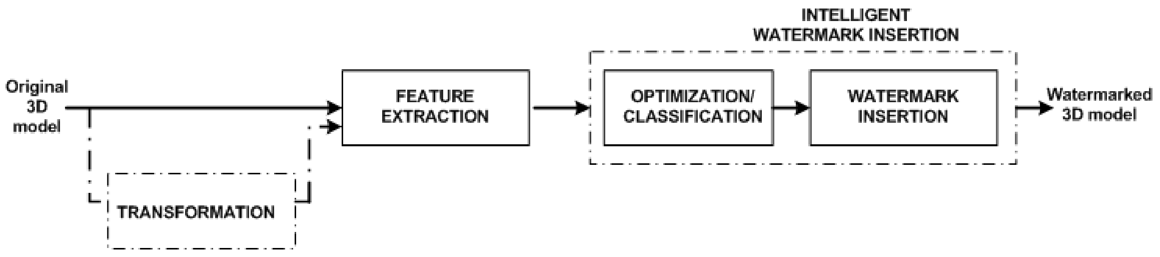


Figure 2.7: Third Generation Watermarking

Genetic Algorithms(GA) are widely used for watermarking of video sequences, audio signals and digital images. In audio watermarking [51], the algorithm uses wavelet coefficients of the audio signal as a chromosome. The fitness function used by the authors is the signal to noise ratio and the similarity between the original and the extracted watermark. The GA determines the best position for insertion of the watermark. For a GA used in image watermarking [96], chromosome coefficients of the image in the Discrete Cosine Transform (DCT) domain are modified to embed the watermark. Peak signal to noise ratio (PSNR) and normalized cross (NC) correlation is used for the fitness function and the GA identifies the ideal pixel candidates to embed the watermark. GA is also used for finding the optimal embedding locations in image watermarking using wavelet transforms [104]. In this paper, chromosomes are selected to be the values of the embedding strength and the number of times of Arnold transform. The normalized cross-correlation and the PSNR constitute the fitness function. GA have also been used in video watermarking [39]. In this algorithm, the watermark image is embedded into video frames by changing the position of some discrete wavelet transform coefficients. Different positions for watermark image embedding are simulated as chromosomes and the mean absolute difference between the intensity values before and after watermark is chosen to be the fitness function.

Recently there has been interest in using artificial neural networks (ANN) for image watermarking [60, 72, 46, 53, 47, 37, 103, 102]. Again, there is no published work which uses ANN for watermarking of 3D models. Neural networks have been trained to perform complex functions in various fields, including pattern recognition, identification, classification, speech, vision, and control systems. Der-Chyuan Lou *et al.* [55] used neural networks to emulate the human visual system for generating the suitable strength of the watermark to be embedded in an image. The watermark is adjusted by the neural network to provide maximum watermark strength, but keeping it imperceptible. The watermark is embedded in the DCT coefficients. The luminance, sensitivity, frequency sensitivity, texture sensitivity and entropy sensitivity are computed and used as the inputs of the neural network. The neural network outputs the weight and the length of the watermark. The proposed watermarking scheme has good peak signal to noise ration (PSNR) and detection response. Also, the watermark is not concentrated in one area of the image, thus improving resistance to cropping attack. Kenneth J. Davis and Kayvan Najarian [29] implemented an automated system of creating maximum-strength watermarks. The neural network accurately and reliably models the performance of the human visual system perception of the quality of the watermarked images. The input to the neural network provides the same information that is available to the human visual system. To model the frequency sensitivity and luminance sensitivity, the neural network is provided with the wavelet transforms, and with the pixel values of the image in the local neighborhood and texture parameters to model the contrast masking effect of the human visual system. The neural network must also be provided with the strength of the added watermark. Their experiments showed that when properly trained, the neural network can perform successfully, allowing it to be used in place of several human reviewers. Although ANNs haven't been used for 3D watermarking, they have been explored for surface reconstruction [48, 23, 30]. Radial basis functions have also been used for surface modelling [89]

Prior to exploring use of fuzzy logic in 3D models in this dissertation, the concept of using fuzzy logic for information fusion was applied to images by building a fuzzy perceptual mask. In [64], brightness, edge sensitivity, and texture was computed for each wavelet at different scales in an image by Motwani *et al.* to compute the watermarking strength. Around same time, similar approach was adopted in [44] to build fuzzy inference filter to select large entropy coefficients to embed watermark in DWT of image. In [84], Sakr *et al.* relies on a dynamic fuzzy inference system to extract the human eye sensitivity knowledge to adjust and select the watermark strength for each pixel in an image. Success of fuzzy perceptual mask in images inspired this research work to explore use of fuzzy logic for watermarking in 3D models and results are outlined in later chapters. Fuzzy logic is not much explored in the field of 3D computer graphics. Shuhong *et al.* [94] used fuzzy logic for adaptive finite element mesh generation.

Chapter 3 Approach

Computational Intelligence (CI) [71, 86] is the study of adaptive mechanisms to enable or facilitate intelligent behavior in complex, uncertain and changing environments. These adaptive mechanisms include those Artificial Intelligence (AI) paradigms that exhibit an ability to learn or adapt to new situations, to generalize, abstract, discover, and associate. Recently, these bio-inspired algorithms have been organized under the umbrella of computation intelligence. The scope of this dissertation is to however explore the use of ANNs, GAs, and Fuzzy Logic. It is intuitive to use bioinspired algorithms such as GAs and ANNs because the viewer of this 3D data is a human and the hacking process is also done potentially by a human to guess the positions of the watermark. Since there is a human element involved in the viewing and brute force attack process, it's logical or intuitive to use a bioinspired process to circumvent the attacks or counter attack the brute force thinking process or the logical process.

Watermarking can be considered as an optimization and classification problem. In a watermarking algorithm, vertices are randomly selected and information is added by perturbing the vertex positions without causing perceptible distortion. Vertices are randomly selected from the search space of the pool of vertices to increase complexity of a brute force attack. However, if the search space is sufficiently large and is possible to reduce the search space to a smaller search space which is made up of good candidates for watermark insertion and at the same time still have randomness in the reduced but high quality search pool. Thus, the watermarking insertion algorithm would randomly select vertices for watermark insertion from a pool of vertices which have good data hiding capacity. Therefore, watermarking can be considered a classification problem where vertices are marked as either good candidates or bad candidates for data insertion. Watermarking can also be considered as an optimization problem where we want to optimize the amount of data to be inserted.

Once vertices are selected for watermarking, the amount of watermark to be inserted in the vertex position is to determined so that there is no perceptible distortion. If a smaller watermark is added by modifying fewer bits in the vertex position, the watermark could be easily eroded or destroyed with a compression attack or other watermarking attacks. If a higher number of bits are modified, the watermarked vertex could potentially cause perceptible distortion. Thus, the amount of watermark to be added is an optimization problem where the algorithm needs to be able to make a determination as to the maximum amount of watermark or bits to modify in the vertex without causing perceptible distortion. Alternatively, the algorithm can choose to select only the vertices from the entire search space and watermark only those which have high data embedding capacity. Thus, the amount of watermark to add, depends on the vertex to be selected which in turn depends on the surrounding geometry of the vertex under consideration. HOW MUCH to insert we want to optimize using evolutionary algorithms such as a genetic algorithm. WHERE to insert can be regarding as a classification problem and can be learned using neural networks or use the key observation to determine the vertices to be watermarked. Motwani *et al.* [70] used Support Vector Machines (SVM) as a classifier to classify vertices as good or bad vertices for 3D watermark insertion. The goal of the watermarking evolutionary optimization process or algorithm is to know WHAT to maximize or minimize through the choice of a fitness function but with ignorance of the details of exactly HOW that goal is achieved since there is randomness involved in achieving that goal. In other words, we want to find the optimal location of the watermarked vertex without putting on the emphasis of how that position is obtained (through evolution).

This approach works for robust as well as fragile watermarking. Fragile watermarking is used to detect any kind of tamper i.e. unauthorized modifications in the model. The best and the simplest way to do this is by inserting a watermark at each and every vertex of the model.

3.1 Watermark Insertion

Figure 3.1 shows detailed block diagram of third generation watermarking algorithms.

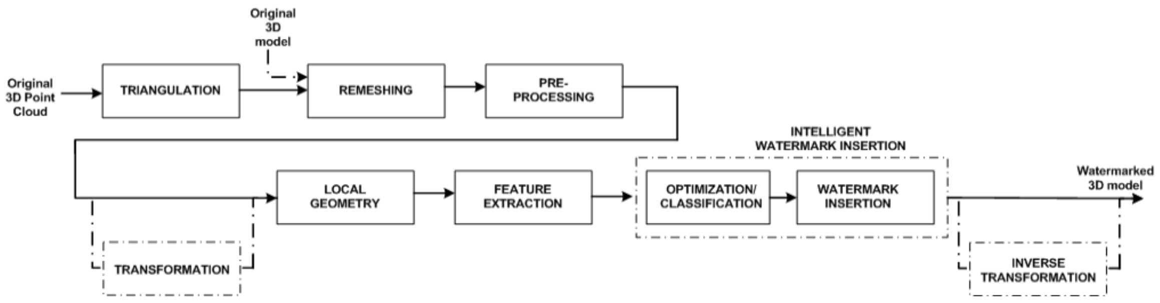


Figure 3.1: Third Generation Watermarking Block Diagram

3.1.1 Triangulation

3D surface can be represented a number of ways and has roots in a branch of mathematics called computational geometry. Computational Geometry is generalization of a set of tools and techniques developed that takes advantage of the structure provided by geometry. Triangulation is the process of determining the connectivity between the vertices to create triangular faces of a wireframe mesh. There are various techniques for surface reconstruction from point clouds. Some of the mesh generation popular techniques can be referred to in [19]. Voronoi diagrams and Delaunay triangulations [18] have many interesting properties, which are studied in depth in discrete computational geometry. Laser scanners scan a material object to create a cloud of 3D points. 3D point cloud can be registered using various techniques as shown in [58] and [20]. Discrete Computational Geometry algorithms such as Delaunay triangulations are then used to create a mesh which is further refined and processed. Ideally, the watermark should be inserted right at the source in the 3D point cloud. However, such a technique may not support a multiple distribution reseller model and the mesh may further be refined after connectivity between the vertices is determined. Making triangulation part of the watermarking process makes the watermarking technique invariant to vertex reordering and remeshing attacks. However, if the mesh is already generated, this part of the step is not required.

3.1.2 Remeshing

Since the proposed technique requires the mesh to be semi-regular, it is imperative that the mesh is remeshed into semi-regular. Praun and Hoppe in [78] gives details on remeshing 3D mesh into semi-regular mesh.

3.1.3 Pre-Processing

Normalization of 3D models as a pre-processing step before insertion of a watermark makes the watermark resilient to modifications in the 3D model due to affine and scaling transformations. The center of mass of the 3D model is shifted to the origin and the model is scaled to fit in a unit cube.

3.1.4 Transformation

Second generation watermarking techniques use various transformations such as spectral representation, wavelet transform, and progressive meshes. Applying a transform such as wavelet transform is performed in this step.

3.1.5 Local Geometry Representation

It is important that a good watermarking algorithm should not use connectivity information since mesh vertex reorder would easily destroy such a watermark. There is a need to define a feature vector to represent the local geometry of the mesh to insert the watermark. These feature vectors would be then determined over a surface which represent the local geometry. In this dissertation, 1-ring and 2-ring vertices have been selected to represent the local geometry of a 3D model. In future work, other local geometric structures such as Voronoi rings, fixed radius 3D balls, and voxels can be used to represent local geometry and feature vectors derived based on these geometric structures could be exploited using CI techniques to classify the vertices for selection of insertion of watermark and optimize the amount of watermark to be inserted.

The 1-ring neighborhood of a vertex V is defined as the surfaces formed by that vertex V with its neighbors as shown in Figure 3.2

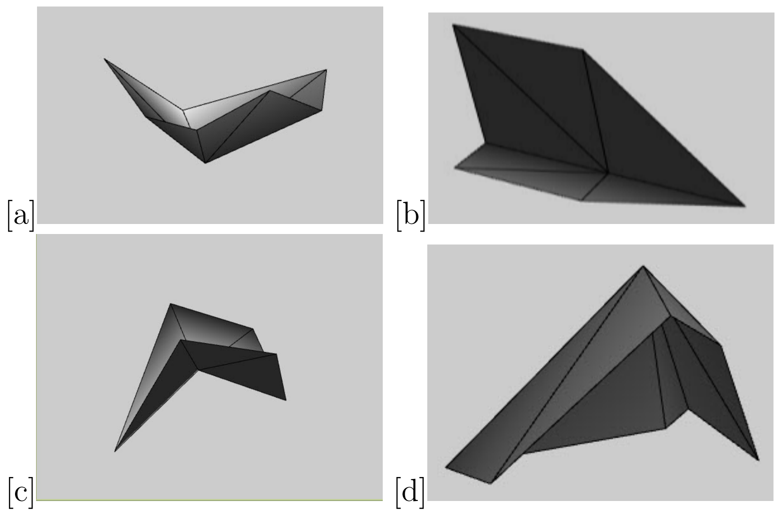


Figure 3.2: Various 1-ring vertices shown from different angles

3D triangular mesh models are represented by a set of vertices and a list of triangular faces formed by the vertices. A vertex v_i is a neighbor of another vertex v_j if an edge exists that connects v_i and v_j . The set of all the neighbors of a vertex v_i is called 1-ring of the vertex. The set of all neighbors of the 1-ring neighbors of a vertex v_i along with the set of their 1-ring neighbors is called 2-ring of the vertex, as shown in Figure 3.3. The number of neighbors of v_i in it's 1-ring neighborhood is the valence or degree of the vertex v_i. Figure 3.3 shows a vertex of valence 5 (1-ring) and vertices of valence 5 and 6 (in the 2-ring).

2-ring neighbourhood is used here since the level of accuracy increases with a 2-ring neighborhood because the comparison is done with a larger number of vertices.

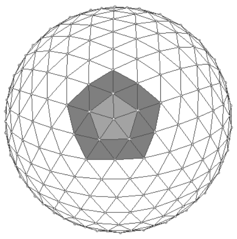


Figure 3.3: Neighborhood of a Vertex: 1-ring demonstrated by light gray patch, 2-ring demonstrated by light and dark gray patches

3.1.6 Feature Extraction

The basis of the use of these bio-inspired algorithms is based on the premise of two key observations listed below.

1. The perception of distortion caused by inserting watermark is influenced by the surrounding geometry of the vertices. For example, perturbation of an isolated vertex in 3D would not cause the eye to perceive any change in location of the vertex. However, if the isolated vertex is in the backdrop of a flat surface, the change is more visible. If the same vertex is on a bumpy surface, the change is imperceptible. Thus, the absolute location of the vertex is not important for watermarking, but its location relative to the local geometry determines whether the vertex is a good or bad candidate for watermark insertion. If information is inserted in a particular vertex, it may be perceivable as distortion, whereas if information is inserted in the neighborhood of the vertices along with the vertex under consideration, the distortion can be masked by the supporting geometry. The neighborhood of the vertices should also be considered in the selection process to determine their suitability or ”fitness” for selection.

2. If a vertex is located on a very bumpy surface then the change in location would be least perceptible. For the purpose of analysis of this key observation and using them as basis of the bio-inspired algorithms, flat, smooth, curved, bumpy and combinations of these types of surface patches are primarily analyzed. Flat Surface, Curved surface, Smooth surface, bumpy surface, combination of the

above surfaces as shown in Figure 3.2.

Thus, based on the above key observations, feature vectors are defined which quantitatively model these observation. Based on the type of local surface geometry, two features are defined.

CURVATURE is the amount by which a geometric surface deviates from being flat. Curved surface consist of a number of smaller triangles to give the perception of smooth surface as compared to what is needed for a flat surface. Normal variation usually gives a good indication of the surface curvature. For example, if the surface is flat, all the surface normals are parallel to each other and there is zero deviation of the average normal from each of the surface normals. If the surface is smooth, the deviation of the surface normal from the average normal is consistent with the deviation of the other surface normals from the average normal. If the surface has uneven curvature, the deviation of the surface normals from the average normal could be erratic.

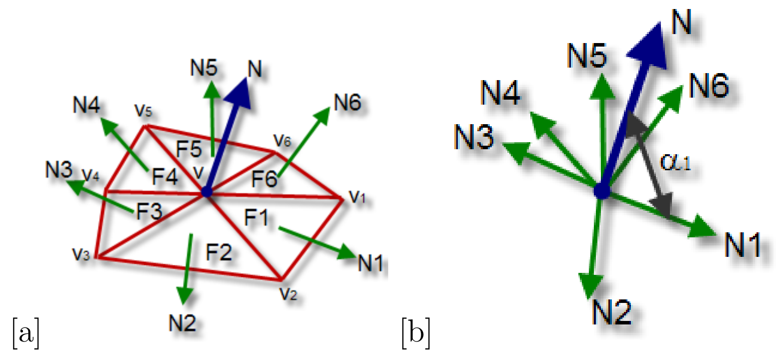


Figure 3.4: Surface normal's (in green) and average normal (in blue) for a 1-ring vertex neighborhood

BUMPINESS in the wavelet domain as used by Kanai [49] is calculated by dividing the scaler coefficient of a low resolution model with the length of vector between its adjacent scalar coefficients.

3.1.7 Intelligent Watermark Insertion

This part of the process can include either an optimization step or a classification step. Depending on the computational intelligence technique, details of the step vary. The Fuzzy Logic, Genetic Algorithms, and Neural Network chapters (Chapters 4, 5, and 6) discuss this part of the process.

3.2 Watermark Extraction

The watermark extraction process is shown in Figure 3.5. To make the watermark robust to remeshing attacks, the extracted watermark is remeshed using geometry images. The remeshed model is the normalized and rotated to align with the principal component of the 3D model data for mesh alignment. The extraction process is non-blind if an input model is available to subtract from the watermarked model. The process is semi-blind if a key is available to determine which vertices were watermarked and the amount of information inserted in each vertex.

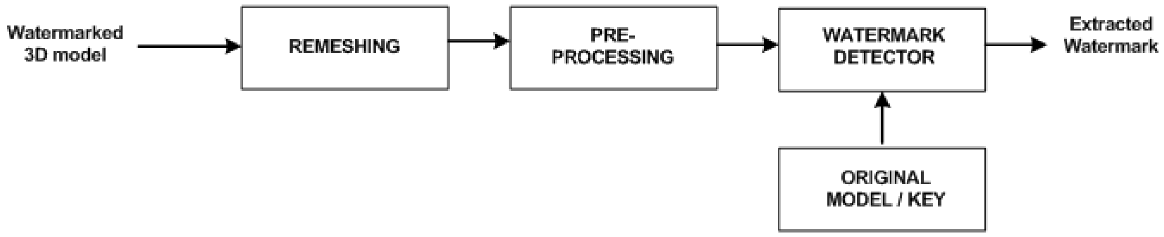


Figure 3.5: First Generation Watermarking Block Diagram

3.2.1 Similarity Measure

Similarity between the extracted watermark and the original watermark is computing using correlation. Correlation is found using equations below.

$$A(u) = A_x(i) + A_y(j) + A_z(k) \tag{3.1}$$

$$A(u)' = A_x'(i) + A_y'(j) + A_z'(k) \tag{3.2}$$

Where $A_x(i), A_y(j), A_z(k)$ are the x,y,z co-ordinates of the u' th vertex in the attacked model. $A_x^{'}(i), A_y^{'}(j), A_z^{'}(k)$ are the x,y,z co-ordinates of the corresponding u' th vertex in the watermarked model.

$$\text{Amount of correlation} = \frac{w+w^{'}*correlation}{W} \tag{3.4}$$

w = number of vertices not attacked, $w^{'}$= number of vertices attacked, W = total number of vertices in the watermarked model.

If the correlation > 80% implies that the model has not been attacked. For fragile watermarking, the correlation = 100% to detect tampering.

3.3 Performance Evaluation

The embedding capacity of the watermark can be determined by the number of vertices watermarked and the amount of watermark inserted at each vertex in the model. For fragile watermarking, the number of vertices watermarked is more important than the strength of the watermark (bits inserted/vertex is more important) since the goal is to identify tampering at each vertex. For robust watermarking, the number of vertices could vary depending on the size of the model and since there is randomness in the process the number of vertices is not fixed. Thus, for robust watermarking the strength of the watermark or the capacity is determined by the number of bits/vertex. However, the number of bits inserted per vertex could vary and thus the Signal to Noise Ratio (SNR) is good indication for the strength of the watermark. It is well known fact that SNR is not an indication of the perceptual distortion. Recently, Hausdorff distance has been used as a more accurate measure to compare the errors between two surfaces.

3.3.1 Perceptibility Distortion Measures

Watermarking algorithms typically use Signal to Noise Ratio (SNR) [17, 95] to measure perceptible distortion and Normalized Cross Correlation (NCC) to measure the robustness of the watermark. Hausdorff distances (HD) are generally used to find out the degree of mismatch between two sets of points and have been used as a distortion measure for image watermarking [62]. The use of Hausdorff distance as an error measuring parameter has been explained in MESH [12]. The equations below compute SNR and HD for the 1ring and 2ring local geometric structures.

The amount of distortion in the 1-ring can be measured by computing the signal to noise ratio (SNR) as follows:

$$SNR = \frac{\sum_{i=1}^{N}(X_i^2 + Y_i^2 + Z_i^2)}{\sum_{i=1}^{N}[(X_i - X_i')^2 + (Y_i - Y_i')^2 + (Z_i - Z_i')^2]} \tag{3.5}$$

where N is the number of vertices in the 1-ring or 2-ring neighborhood of the center vertex including the center vertex;

X_i, Y_i, and Z_i are the Cartesian coordinates of the vertices in the 1-ring or 2-ring neighborhood including the center vertex;

X_i', Y_i', and Z_i' are the modified coordinates of the center vertex.

$$VSNR = 20 * Log_{10}(SNR) \tag{3.6}$$

For given sets of points, the Hausdorff distance is defined as the maximum distance of a set to the nearest point in the other set. Thus, Hausdorff distance can be represented as a maximin function as shown in equation below

$$h(a,b) = \text{max}\{\text{min}(\text{d(a,b)})\} \tag{3.7}$$

where, 'a' and 'b' represent the sets of points between whom the Hausdorff distance is measured. In the proposed algorithms, 'a' is the vertex list formed by the 1-ring or 2-ring neighborhood of the vertex V and 'b' is the vertex list formed by the 2-ring neighborhood of V_i, which is the position of V with the watermark inserted.

It should, however be noted that $h(a,b) \neq h(b,a)$ Thus, the symmetrical hausdorff distance is defined as

$$H(a,b) = max\{h(a,b)h(b,a)\} \tag{3.8}$$

In this dissertation, we have used Hausdorff Distance and Vertex Signal to Noise Ratio (VSNR) to find the degree of mismatch between the 1-ring or 2-ring neighborhoods of the original vertex V and the watermarked vertex V_i.

3.3.2 Precision

The vertex is represented in IEEE double precision floating point form as a string of binary bits. It is composed of 64 bits, divided into a 52 bit mantissa M, 11 bit exponent E, and sign bit S, as seen in Fig. 3.6

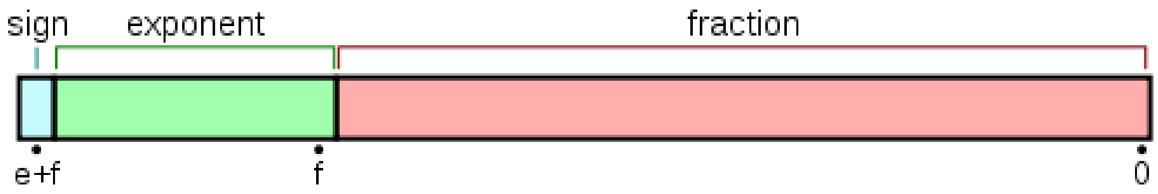


Figure 3.6: Floating Point Representation of Vertex Coordinate

The binary sequence is inserted into the 40 least significant bits (LSB) of the mantissa by performing Boolean OR operation. This ensures that the watermark is additive. A sequence of 40 bits is the maximum amount of information that can be inserted in the vertex without causing visible distortion.

Chapter 4

Fuzzy Logic

Fuzzy Logic [59, 81] is a superset of classical number theory or set theory. It allows us to model imprecise or vague information which otherwise is extremely complicated and difficult to represent using mathematical equations. Fuzzy Logic was introduced by Professor Lotfi Zadeh [101] in 1965 and has been widely used in control systems, washing machines, and other electronic systems. It is best suited for modeling non-linear data and represent imprecise data in easy to understand IF THEN ELSE linguistic rules. This makes them easy to design and provides a way to capture human knowledge which otherwise is difficult to represent in mathematical equations in to simple linguistic rules which otherwise are simple to define or describe using daily language sentences. Fuzzy Logic has been successfully used for solving classification problems using unsupervised data.

4.1 Algorithm

4.1.1 Block diagram

As we saw in Chapter 3, we have a block diagram of our framework. That diagram is shown again in Figure 3.1. We are now going to zoom in on the bottom row of that diagram and show the enhanced flow for a Fuzzy Logic watermark generation and embedding. This is shown in Figure 4.1.

In this chapter Fuzzy Logic is used to quantify the amount of watermark to insert in vertices. Thus, fuzzy logic determines how to classify vertices into separate bins

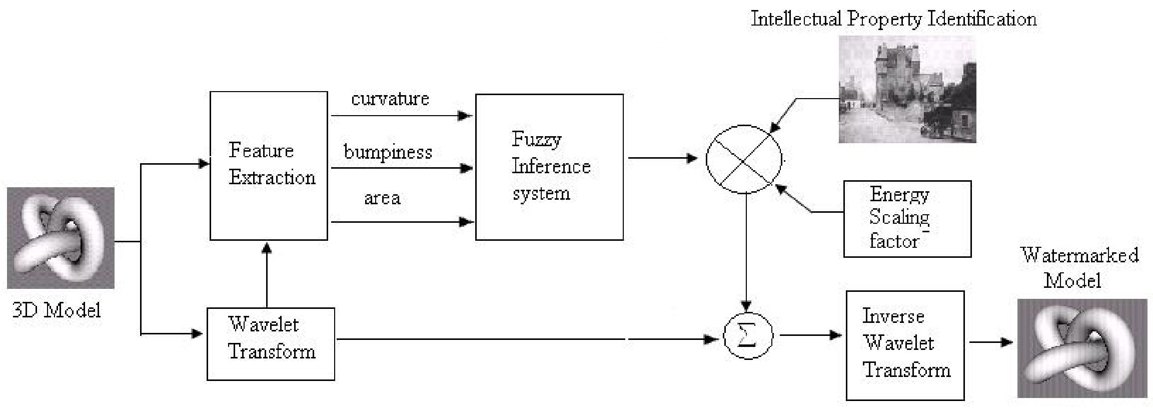


Figure 4.1: Fuzzy Logic Based Approach - Block Diagram for Watermark Embedding

identifying which vertices are good for insertion of watermark.

4.1.2 Watermark Insertion

Step 1 - Decomposition of 3D Model

As shown in Section 3.1.3, prior to applying the wavelet transform, all mesh vertices are normalized between 0 and 1 by placing an imaginary bounding box to provide robustness against scaling attacks. The wavelet transform [38] is then implemented using Lifting Scheme [87] and the Cohen-Daubechies-Feauveau CDF (2, 2) [28] wavelet is used. Lifting scheme requires that the input signal samples be classified into even and odd for computation of scalar and wavelet coefficients respectively.

Decompose the original 3D model using a 3D wavelet transform with up to three levels using a CDF (2, 2) wavelet. The basic aim of the application of wavelets to a 3D model is to bring about its multiresolution representation. Figure 4.2 shows the block diagram for wavelet decomposition.

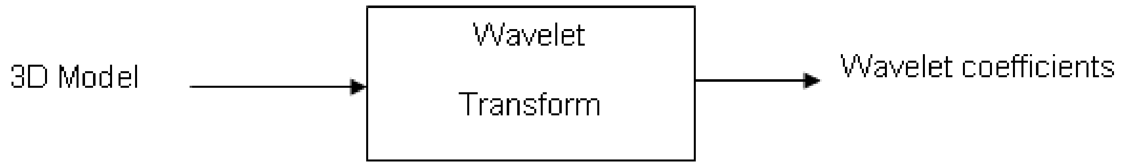


Figure 4.2: Block diagram for wavelet decomposition

Step 2: Computing fuzzy inputs

Fuzzy input variables are computed considering the geometry of the model such as area, curvature, and bumpiness of the surface corresponding for each vertex. The area of the triangular face formed by 3 vertices is computed by the magnitude of the normal to the triangular patch. Curvature is the amount by which a geometric object deviates from being flat. A curved surface consist of more smaller triangles when compared to a flat surface. Since only semi-regular meshes are considered in the current system, each regular vertex is connected to 6 other vertices. Thus, 6 surface normals are computed for each corresponding neighbor's vertex. Curvature is computed by taking average of the angles between surface normals and the average surface normal. A bumpy surface is a surface which is not smooth but is irregular and uneven. A bumpy surface has more details associated with it and thus has more watermark holding capacity. Bumpiness is calculated by dividing the wavelet coefficient magnitude by the length of vector joining two EVEN neighbors as shown in Figure 4.3. Bumpiness, Area, and Curvature are passed as fuzzy inputs to the Fuzzy Inference System (FIS) to compute a Fuzzy perceptual mask for each wavelet coefficient at each level. Curvature and area for the mesh vertices are computed in the spatial domain whereas bumpiness for the corresponding vertex is computed in the wavelet domain.

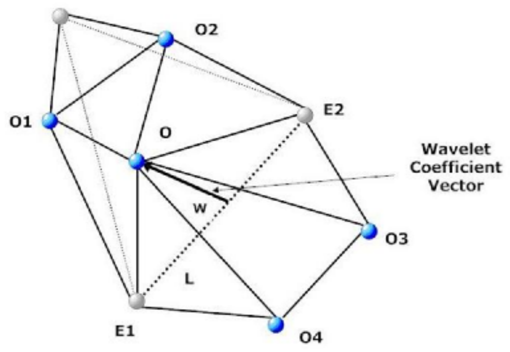


Figure 4.3: Bumpiness calculation using Wavelet Coefficient vector

Fuzzy input variables are computed considering the geometry of the models such as curvature and bumpiness of the surface for each wavelet coefficient. The output of the fuzzy system is a single value which gives a perceptual value for each corresponding wavelet coefficient. Thus, the fuzzy perceptual mask combines all these nonlinear variables to build a simple, easy to use model for spatial masking. Figure 4.4 shows block diagram of fuzzy inference system (FIS).

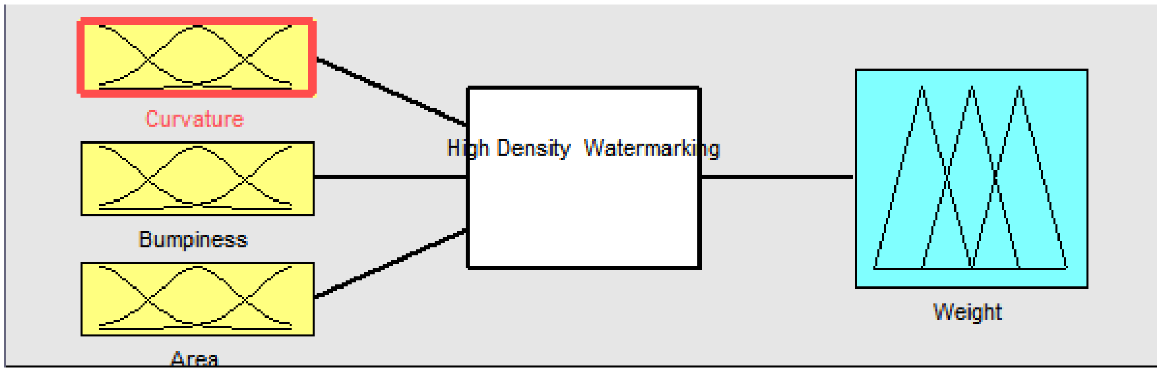


Figure 4.4: Block diagram of fuzzy inference system

Step 3: Computing Fuzzy Mask

The output of the fuzzy system is a single value which corresponds to a perceptual threshold for each corresponding wavelet coefficient. Thus, the fuzzy perceptual mask combines 3 nonlinear variables: Curvature, Bumpiness, and Area to build a simple, easy to use model. Although the fuzzy output has 7 membership functions as shown in Figure 4.5, only the HIGH and the HIGHER fuzzy output sets are used for insertion of the watermark in the 3D model. This is to make the watermark imperceptible and more robust.

A total of 15 fuzzy rules have been developed, described as follows:

1. IF [Curvature] is Low AND [Bumpiness] is Low THEN [Weighting factor] is Lowest

2. IF [Curvature] is Low AND [Bumpiness] is Medium THEN [Weighting factor] is Lower

3. IF [Curvature] is Low AND [Bumpiness] is High THEN [Weighting factor] is Low
4. IF [Curvature] is Medium AND [Bumpiness] is Low THEN [Weighting factor] is Lower
5. IF [Curvature] is Medium AND [Bumpiness] is High AND [Area] is Low THEN [Weighting factor] is Medium
6. IF [Curvature] is Medium AND [Bumpiness] is High AND [Area] is Medium THEN [Weighting factor] is High
7. IF [Curvature] is Medium AND [Bumpiness] is High AND [Area] is High THEN [Weighting factor] is Higher
8. IF [Curvature] is High AND [Bumpiness] is Low THEN [Weighting factor] is Low
9. IF [Curvature] is High AND [Bumpiness] is Medium AND [Area] is Low THEN [Weighting factor] is Medium
10. IF [Curvature] is High AND [Bumpiness] is Medium AND [Area] is Medium THEN [Weighting factor] is High
11. IF [Curvature] is High AND [Bumpiness] is Medium AND [Area] is High THEN [Weighting factor] is Higher
12. IF [Curvature] is High AND [Bumpiness] is High THEN [Weighting factor] is Higher
13. IF [Curvature] is Medium AND [Bumpiness] is Medium AND [Area] is Low THEN [Weighting factor] is Low
14. IF [Curvature] is Medium AND [Bumpiness] is Medium AND [Area] is Medium THEN [Weighting factor] is Medium

15. IF [Curvature] is Medium AND [Bumpiness] is Medium AND [Area] is High THEN [Weighting factor] is High

A number of experiments were performed in developing the fuzzy interference system. Different fuzzy rules were adopted and analyzed using fuzzy surface plots as shown in Figures 4.8 and 4.9. Figure 4.6 illustrates how the rules are evaluated for a given fuzzy input.

Once the weighting factor is computed through the FIS (Fuzzy Interference System), the fuzzy sets are further multiplied by the random binary sequence and further scaled by a salience factor, giving the fuzzy mask for the corresponding wavelet coefficient. The above procedure of computing the weighting factor is computed for the all the wavelet coefficients in each level. The weighting factor is obtained by defuzzifying the fuzzy output using Mamdani [100] type inference system.

Step 4: Embedding watermark sequence

The watermark is inserted by modifying the wavelet coefficient vectors in accordance with the fuzzy mask corresponding to that wavelet coefficient and the binary watermark sequence using the following equation:

$$W' = W + f(F, B, K) \tag{4.1}$$

where,

W' = modified wavelet coefficient vector,
W = original wavelet coefficient vector,
F = fuzzy perceptual mask,
B = 8 bit gray scale image,
K = energy scaling factor.

Step 5: Reconstruction of 3D model

Compute the inverse 3D wavelet transform of W' to get the watermarked model.

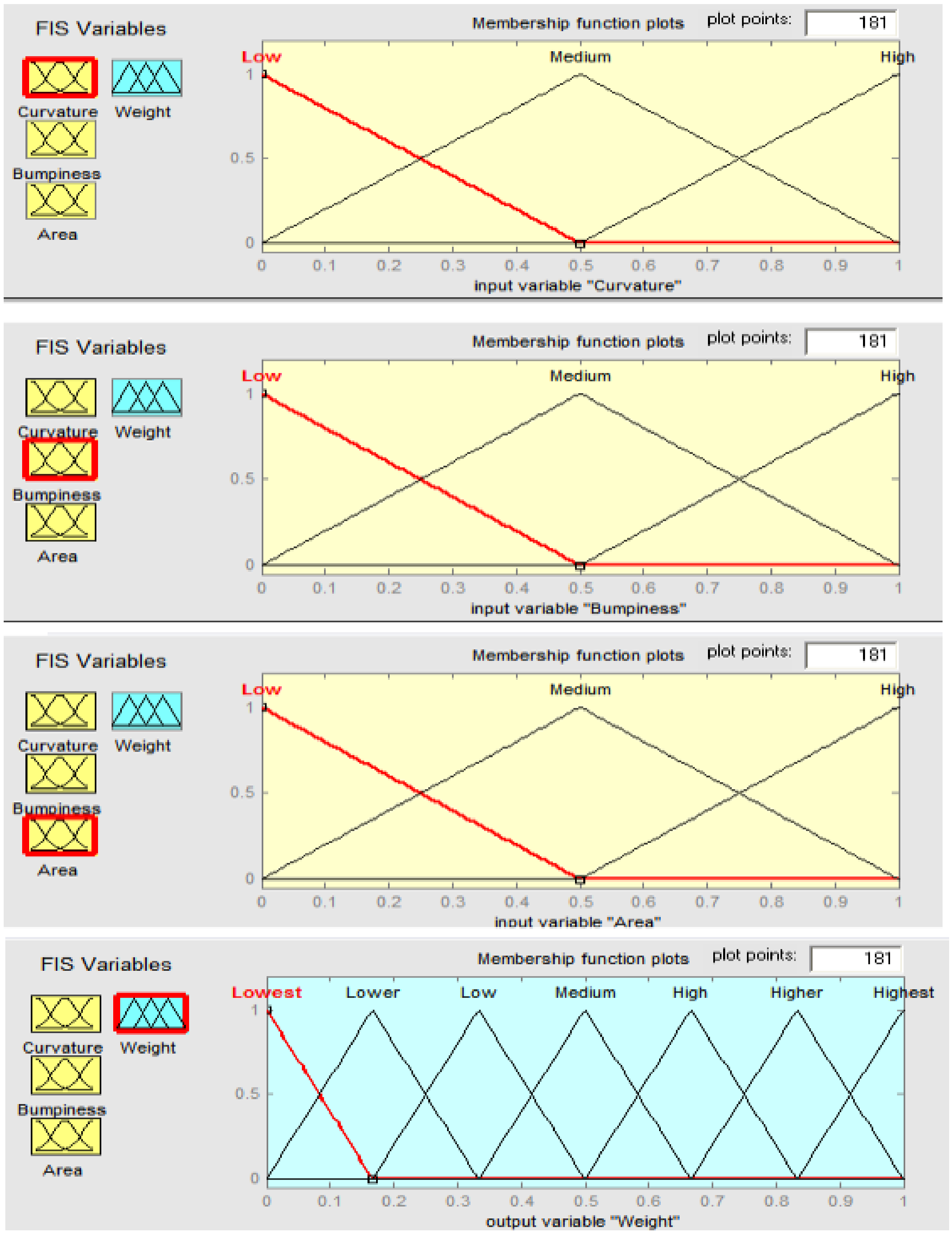


Figure 4.5: Membership functions for Fuzzy Inputs and Outputs

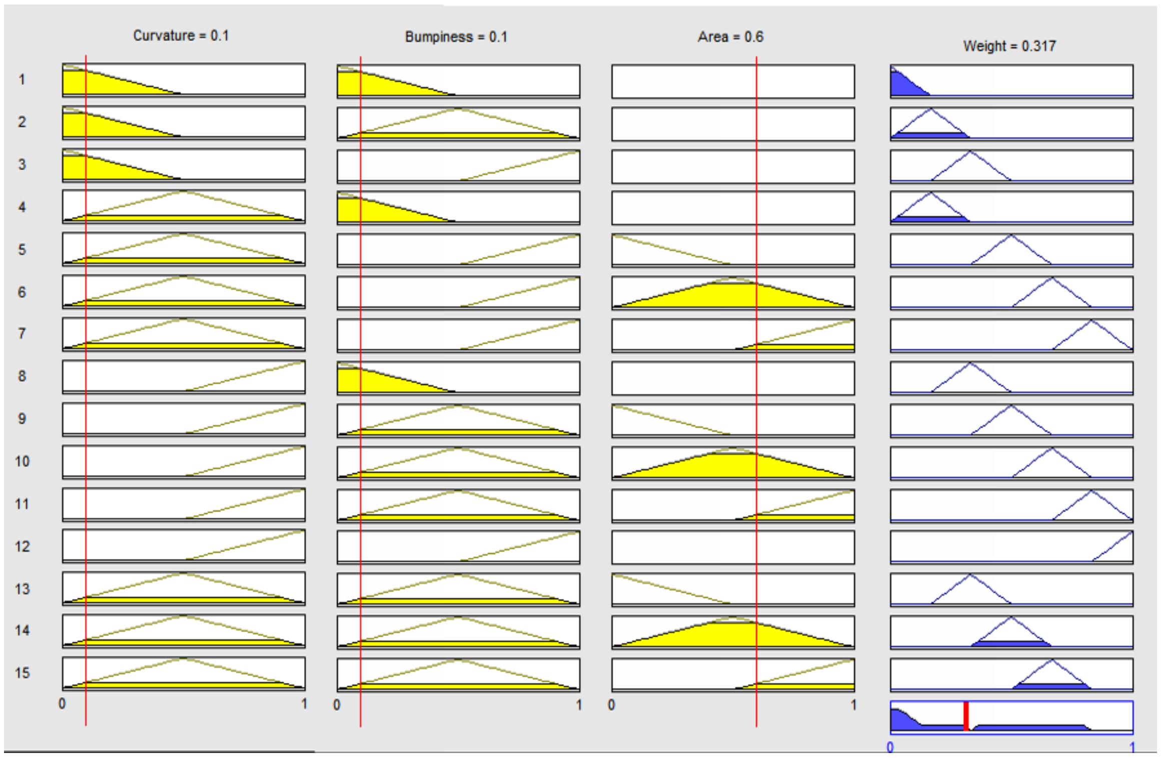


Figure 4.6: Fuzzy Rule Evaluation

4.1.3 Watermark Extraction

A non blind method is used to extract the watermark as illustrated in Figure 4.7. Algorithm for extracting the watermark is as follows

Step 1 - Decomposition of 3D Model

Compute the CDF (2, 2) 3D wavelet transform of the model which has to be tested for attacks and that of the original model.

Step 2 - Watermark Retrieval

Subtract the coefficients of the two models to obtain the watermark.

Step 3 - Watermark Correlation

Correlate the original watermark (B) with the recovered watermark (B*) to determine the authenticity.

Since our system is a non-blind watermarking scheme, the original model and original watermark are needed to extract the watermark from the attacked model. Correlation is computed for the original and attacked watermark and a threshold chosen to determine if the model is attacked.

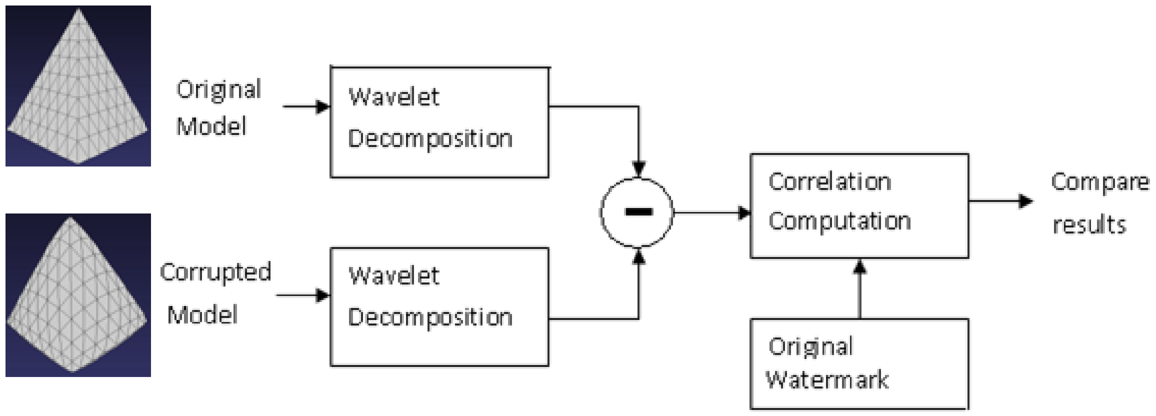


Figure 4.7: Block diagram for watermark extraction

4.2 Experimentation

The choice of wavelet depends on the nature of the data. In nature images, no particular wavelet seems to perform better in comparison to other wavelets. The same is assumed for models created using scanning of objects. However, if the mesh is synthetically created, the choice of wavelet could have a huge impact on the compact representation properties of the wavelet. A bi-linear interpolated wavelet would then perform better as compared to other wavelet to sparsely represent 3D data. In the proposed algorithm a CDF(2,2) wavelet was used to obtain multi-resolution representation of the model.

Generally the wavelet transform used in 3D multimedia uses a lifting scheme so as to make it memory and time efficient. There are various types of wavelet transforms such as spherical wavelets, Bsplines, CDF (2, 2) etc. Due to local support of CDF (2, 2) wavelet, the ratio w/l which is the ratio of wavelet vector to the vector joining two even vertices can be found out. This ratio is nothing but the bumpiness parameter of a mesh. Hence we use CDF (2, 2) wavelet transform. CDF (2, 2) wavelet transform is bi orthogonal wavelet transform that uses lifting scheme. Thus it is memory efficient and time efficient. The local support for CDF (2, 2) wavelet is 2. It uses only the neighboring even vertices of any ODD vertex. Thus due to this any changes made in wavelet coefficients reflect only locally. Fuzzy Surfaces are plotted as shown in Figure 4.8 and Figure 4.9 to show the non-linearity of the fuzzy variables.

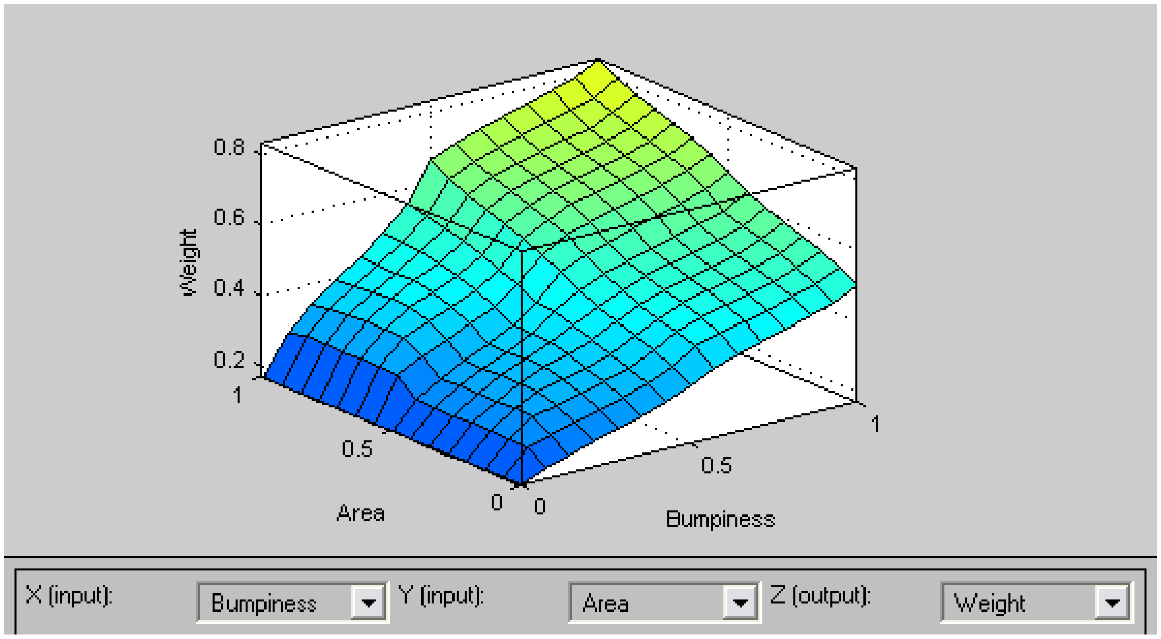


Figure 4.8: Fuzzy Surface for bumpiness and area

4.3 Performance Evaluation

The meshes as shown in Figure 4.10 used for analysis have different shapes to analyze the fuzzy mask behavior for all three fuzzy inputs: curvature, bumpiness, and area. Some of the specifications for these models are shown in Table 4.1 8-bit gray scale images are used for watermark insertion in the meshes. The algorithm was tested with various gray scale images of different sizes to evaluate attacks. As shown in Table 4.2 and Figure 4.11, the algorithm is robust to all kinds of attacks giving exceptional results.

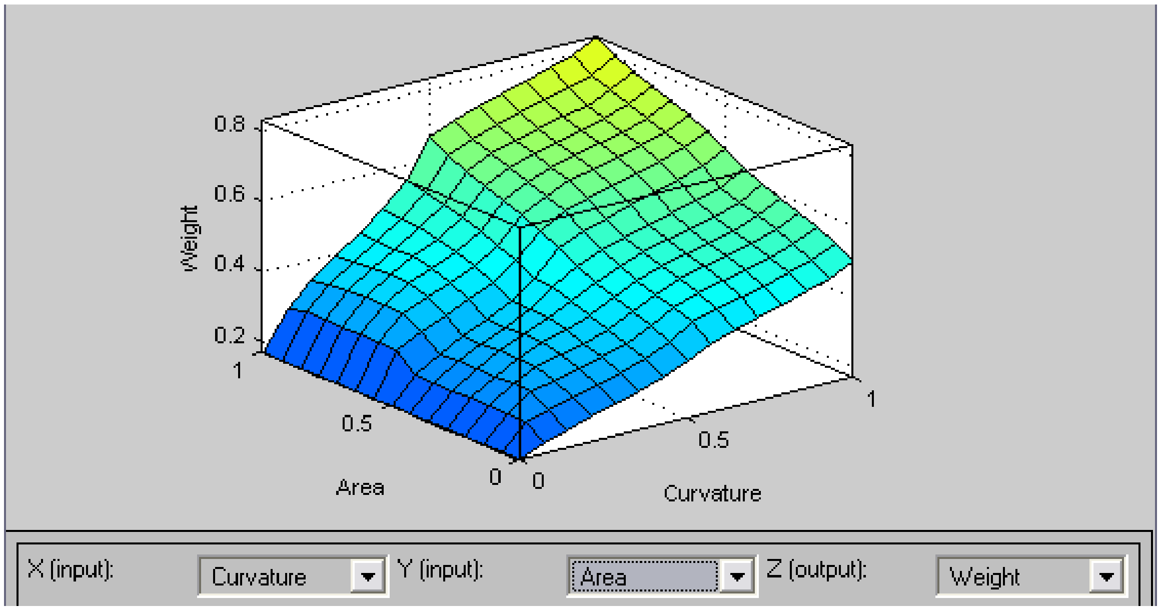


Figure 4.9: Fuzzy Surface for curvature and area

Table 4.1: Models used for Fuzzy Logic testing

Model	No. of Vertices	No. of polygons	No. of Vertices Watermarked
Smiley	1026	2048	576
Super Pyramid	16386	32768	10346
Doughnut	23040	46080	11780
Super Smiley	16386	32768	12046
Bumpy Doughnut	23040	46080	12364

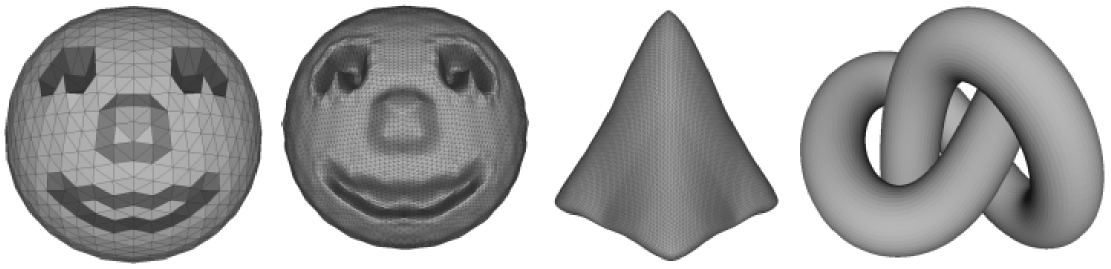


Figure 4.10: Models used for fuzzy testing

The meshes used for testing had different shapes to analyze the fuzzy mask behavior for all three fuzzy inputs curvature, bumpiness and area. A sphere shaped model ’smiley’ was used to evaluate firing of fuzzy rules for surfaces with relatively smooth degrees of curvature. A low and high resolution donut shaped mesh ’donut’ was used

Table 4.2: Correlation results with extracted fuzzy watermark

Model	Rotation, translation and scaling	Noise addition 0.2%	Smoothing (HC Laplacian Filter)	2nd watermark	Cropping
Smiley	1	0.9829	0.9913	0.9995	0.9891
Super Pyramid	1	0.9919	0.9989	0.7108	0.9442
Doughnut	1	0.7988	0.9586	0.8365	0.9264
Super Smiley	1	0.8671	0.8985	0.8323	0.8011
Bumpy Doughnut	1	0.7144	0.8168	0.8332	0.9409

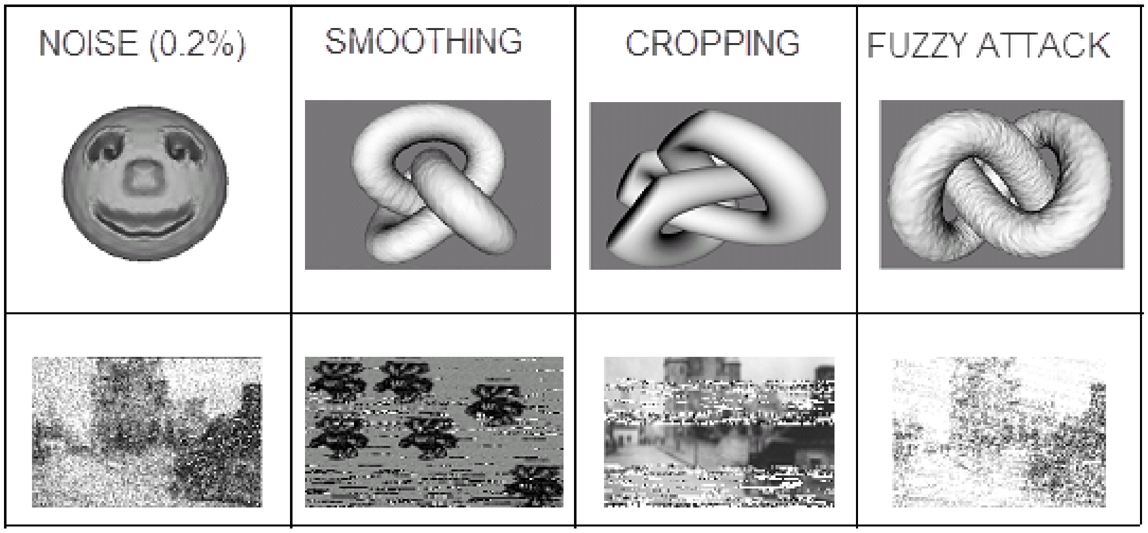


Figure 4.11: Attacks on Fuzzy Watermarked Models

to evaluate curvature with bumpiness. A pyramidal model was also used to analyze the fuzzy system response for surfaces with relatively flat geometry so that rules for larger area polygonal triangles could be tested. Even if an attacker uses subdivision to increase the number of triangular faces and thus reduce the area of triangular face, the curvature and bumpiness fuzzy input parameters remain unaffected because the geometry of the surface remains unchanged. Other meshes with arbitrary geometry to test firing of multiple fuzzy rules firing at the same time were also used. However, to evaluate results and analyze performance, models with just either just curvature or bumpiness were used to analyze the results and draw conclusion. To further evaluate performance on meshes with a higher number of faces, subdivision was used to create 'supersmiley'. A Similar procedure was adopted on lower resolution meshes like donut and pyramid. The following attacks were attempted on watermarked models:

Rotation and Translation: The algorithm is completely invariant to rotation and translation attacks. The change in parameters does not affect the relative distance between the vertices and thus the magnitude of the wavelet coefficients remains unchanged. Thus our algorithm is invariant to rotation and translation.

Scaling: Although scaling modifies the magnitude of wavelet coefficients but due to normalization of the model during watermark insertion and extraction process, the watermark is unaffected. Thus our algorithm is invariant to rotation, translation, scaling or combination of geometrical transformations.

Noise analysis: As shown in Table 4.2, noise affects the watermark the most in bumpy doughnut and doughnut. The watermark is less variant to noise in case of planar surfaces and more severely affected in case of bumpy surfaces. The reason is that during the watermark insertion process the change in entropy of the wavelet coefficients is most for surfaces with bumpiness and curvature and least for planar surfaces. In spite of the noise attack the extracted watermark is still recognizable in all the cases.

Smoothing: Smoothing attack affects the bumpiness of the model more than curvature. This is because it is nothing but low pass filter that removes the high frequency part which is bumpiness. Thus bumpy doughnut is more affected as visually shown in Table 4.2.

Cropping: In our system the watermark is inserted uniformly in the model. Thus even if the model is cropped the watermark is not completely destroyed. The amount of watermark destroyed depends upon the extent of cropping.

Second watermark / Fuzzy attack: A second watermark was inserted in a fuzzy watermarked model in an attempt to overwrite the existing watermark. The existing first watermark is randomly spread through the model thus making it difficult to completely overwrite the existing model. There was no distortion due to the insertion of the second watermark. If a condition is put on not to overwrite the existing watermarked vertices, the fuzzy system can support insertion of multiple watermarks.

The first watermark extracted as shown in Table 4.2 is still recognizable to some degree.

4.4 Analysis

Fuzzy Rules are based on key observations identified in Section 3.3.1 of Chapter 3. The proposed fuzzy system allows us to model the non-linearities of the HVS in the 3D domain. Thus, the fuzzy system can be used as a perceptual mask to identify the amount of watermark to be inserted.

Our algorithm takes advantage of both the spatial and the wavelet domain at the same time by using fuzzy logic. Curvature and area for the mesh vertices are computed in the spatial domain whereas bumpiness for the corresponding vertex is computed in the wavelet domain. Thus, fuzzy logic has been used for information fusion in hybrid domain.

This approach is not suitable for an irregular model because of the limitation that the mesh should be regular. Features such as bumpiness can be calculated in the spatial domain without using a wavelet transform. It is robust to compression since the watermark is inserted up to 4th or 5th significant digit without causing perceptual distortion.

Choice of membership functions and rules are heuristic. There is no well defined way to select membership functions best suited for a particular application. Also the number of rules and the rules used are subjective.

The advantage of wavelet based fuzzy rules is that the watermark can be inserted at multi-resolutions of the model making the algorithm robust. To overcome these limitations, fuzzy rules can be developed which are not based in the wavelet domain.

Due to the limitation of wavelet transforms being applied on semi-regular meshes only, it is a challenge obtaining test data. Only a semi-regular mesh can be used for experimentation since a wavelet transform cannot be performed on irregular meshes. Most of the test 3D meshes available freely are non semi-regular meshes. Models such as smiley were generated using Maya and due to limitations of the artistic skills of the author, the model is very simplistic and is basically a sphere with perturbations. The donut model was obtained as one of the sample models from MeshLab software [5]. Bumpy donut was created by modifying the smooth donut mesh. To create more 3D test models, it is proposed converting irregular meshes into semi-regular meshes. One of the techniques to do this is to project the irregular meshes into geometric images and then project the geometric images as semi-regular meshes.

4.5 Summary

Our proposed algorithm inserts upto 8 bits per vertex coordinate and is robust against smoothing, cropping, affine operations, compression, and noise attacks. Experimental results prove that Fuzzy based watermarking systems can work extremely well for robust watermarking. The proposed algorithm also inserts high capacity watermarks and allows multiple watermarks.

Chapter 5

Genetic Algorithms

Genetic Algorithms (GA) are a class of evolutionary algorithms that use evolution and Darwin's theory of survival of the fittest as a source of inspiration to solve optimization problems [40, 61, 33]. In a genetic algorithm, all the candidate solutions to solve the problem are called chromosomes. Each generation has a specific number of chromosomes, which constitute the population of that generation. The fundamental block of a GA is the fitness function. The fitness function defines the parameter that has to be optimized. Each chromosome is evaluated using the fitness function and returns a value known as the fitness value of that chromosome. According to the best fitness values, some chromosomes are selected to reproduce and they populate the next generation. The most common operators used for reproduction are selection, mutation, and crossover operators.

5.1 Algorithm

5.1.1 Watermark Insertion

The flowchart for genetic algorithm used for watermark insertion is shown in Figure 5.1. Each vertex in the model is represented by (X, Y, Z) coordinates in the Cartesian axis and also represents a chromosome. For each vertex in the 3D model, 1-ring neighborhood for the vertex is extracted. Local geometry of the mesh can be represented by 1-ring vertices. Modifying the coordinate position of center vertex of the 1-ring neighborhood can be considered as adding random noise with reference to the original

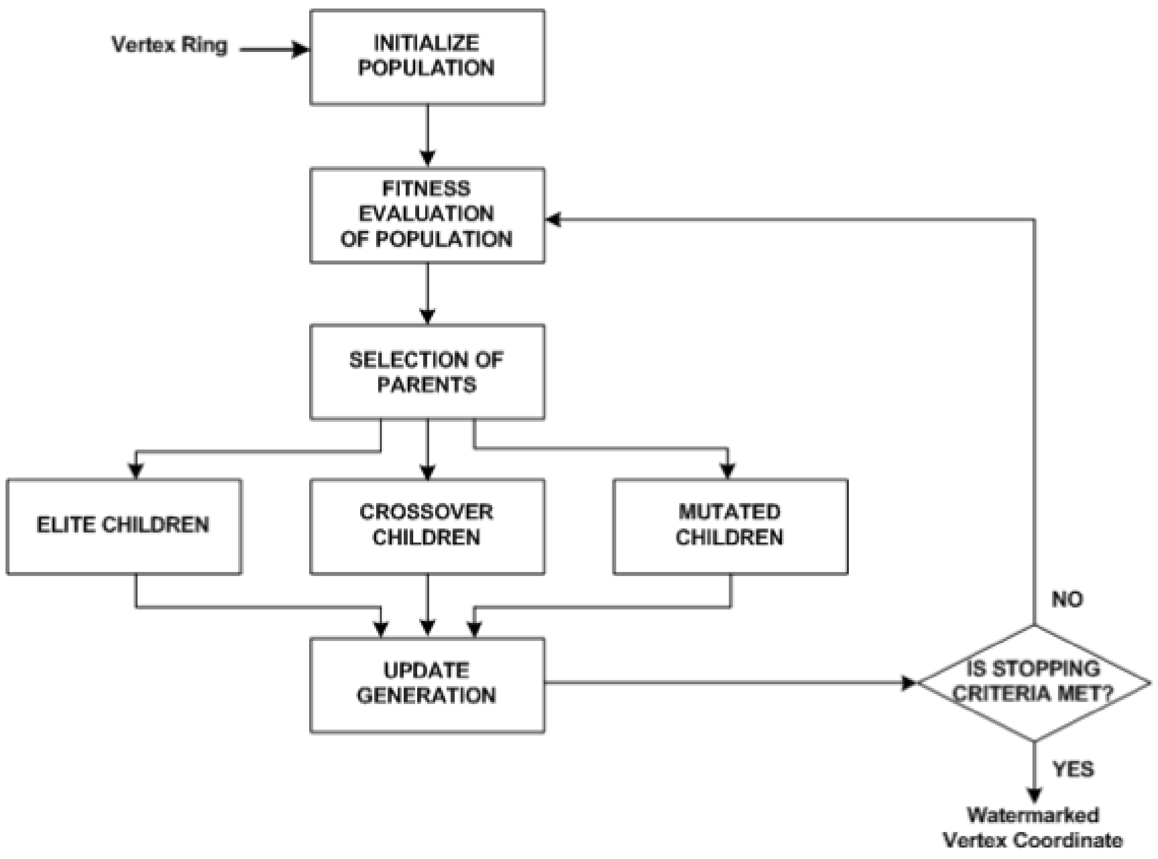


Figure 5.1: Flow chart for GA Approach

location of the vertex. Thus, the watermark added to a vertex is equivalent to noise being added to the vertex. The initial population of chromosomes or candidates watermark vertex positions around the centre vertex are created using uniform creation function. Then the fitness of all the chromosomes in the population is evaluated by sorting the values in ascending order. Parents of the current population are then selected from the sorted fitness values. Elite children, cross over children, and mutated children are then created using the parents from the current population to create the next generation of chromosomes. Until a stopping criteria is met, the same procedure is applied on the next generation. GAs is used to optimize the amount of watermark to be inserted by choosing the best fit chromosome. For fragile watermarking, a GA can be applied on all the 1-ring vertices of the 3D mesh. For robust watermarking, a random number of 1-ring vertices are selected. Once the vertex has been moved by an optimum amount, the old vertex list is replaced by the new vertex list and the face list remains the same. The combination of this new vertex list and face list creates the watermarked model. The model is then scaled back to the original size and shifted to its original position as well. The choice of fitness function determines what the parameter GA is going to optimize. The chromosome with the best fitness value at the end of pre-determined number of generations is considered as the optimized output of the algorithm. The best fitness value from a pool of chromosomes corresponds to the chromosome with the maximum signal to noise ratio (SNR). The amount of distortion in the 1-ring can be measured by computing the SNR as follows:

$$SNR = \frac{\sum_{i=1}^{N}(X_i^2 + Y_i^2 + Z_i^2)}{\sum_{i=1}^{N}[(X_i - X_i^{'})^2 + (Y_i - Y_i^{'})^2 + (Z_i - Z_i^{'})^2]} \tag{5.1}$$

where N is the number of vertices in the 1-ring neighborhood of the center vertex including the center vertex;

X_i, Y_i, and Z_i are the Cartesian coordinates of the vertices in the 1-ring neighborhood including the center vertex;

$X_i^{'}$, $Y_i^{'}$, and $Z_i^{'}$ are the modified coordinates of the center vertex.

The objective of the genetic algorithm is to minimize distortion by maximizing the signal to noise ratio (SNR) for the 1-ring neighborhood. The fitness function chosen is:

$$FitnessFunction = \frac{1}{SNR} \tag{5.2}$$

A genetic algorithm is used to compute the near optimal value of the amount of watermark to be added to the center vertex of the 1-ring neighborhood without causing any perceptible distortion.

5.1.2 Watermark Extraction

In a non-blind approach, the original model is subtracted from the watermarked model and thus the watermark is extracted. However, this approach is not restricted to non-blind watermarking. If a key is used to store the vertex index and the amount of watermark added by the above proposed algorithm using GA, the original model is not required. The extracted watermarked and the original watermark are then compared using correlation. For robust watermarking, the correlation threshold is 80%. For fragile watermarking, the threshold is 100%.

5.2 Experimentation

5.2.1 Parameters

The probability that the GA based optimization process gets caught in a local optimum depends on the parameter settings [83]. Parameters have been selected by trial and error so that the solution converges. However, even if the algorithm is stuck occasionally in a local minima, that is perfectly acceptable.

Each chromosome is represented by (X, Y, Z) Cartesian coordinates and a population of 100 such chromosomes is initially created near the center vertex of the 1-ring neighborhood. The uniform creation function creates a normally distributed random population within the initial range. The initial range defines the upper bound and the lower bound of the X, Y, and Z co-ordinates while creating the initial population. These limits have been set to the maximum and the minimum value of the respective co-ordinates of the 1-ring. This ensures that the optimal chromosome does not move outside the one ring to avoid perceptible distortion. The uniform creation function creates a normally distributed random initial population within the range. The selection operator used is stochastic uniform. Stochastic uniform is used because it considers a certain value below which it does not allow to reproduce, i.e., it follows the basic rule of GA that ability to reproduce is directly proportional to its fitness value. In this type of selection, threshold value is choosen which is inversely proportional to the number of parents. This threshold value is also called the step size and if the fitness value of any chromosome lies below this step size, it is considered to be weak and hence not allowed to reproduce. Similarly, if the fitness value of a chromosome is greater than or equal to M times the threshold value, then that chromosome is selected to reproduce M times. M is determined by

$$M = \lfloor\left(\frac{fitnessvalue}{stepsize}\right)\rfloor \tag{5.3}$$

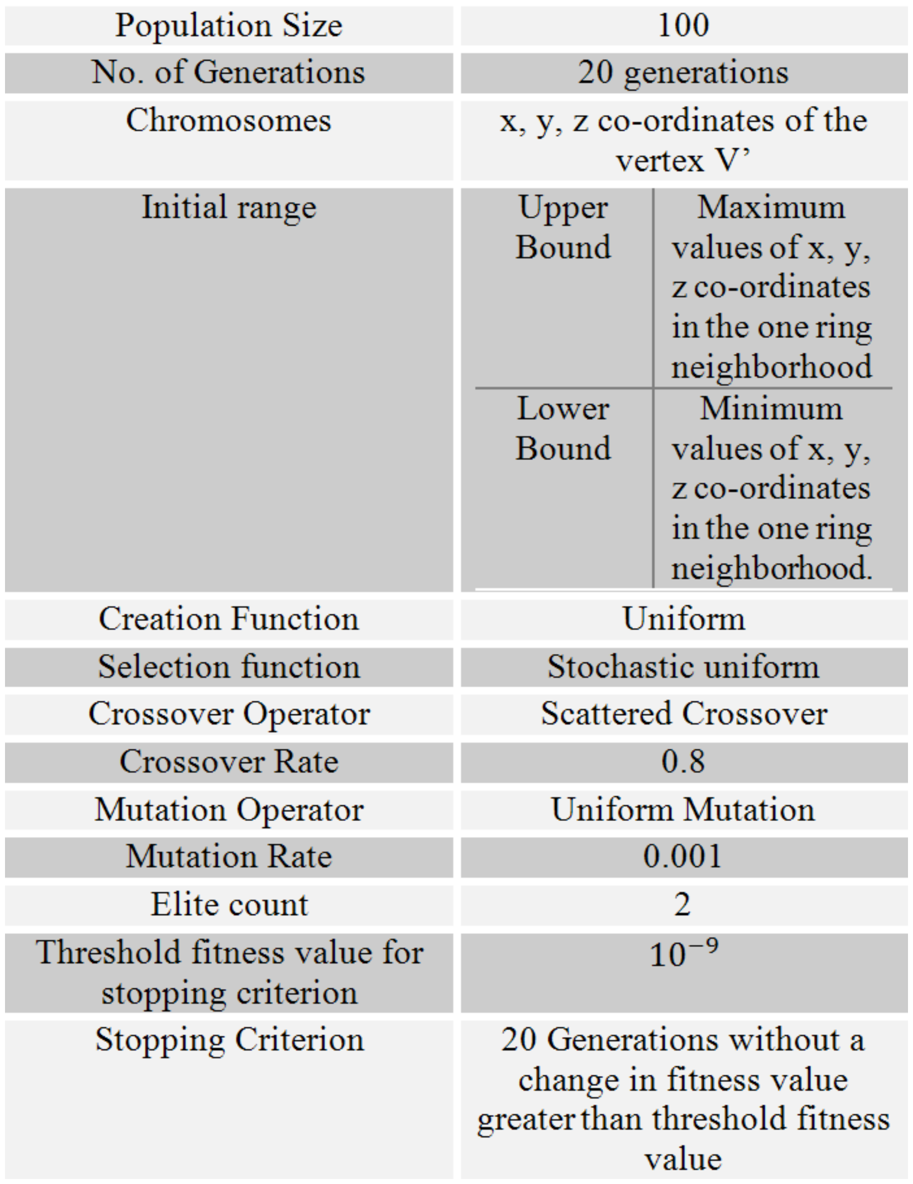
Scattered crossover is used as crossover operator. The mutation operator used is uniform mutation operator. In this type of mutation operator, the X, Y, and Z co-ordinates of the chromosomes are randomly replaced by a value within the range. Thus, mutation operator randomly flips one of the genes of the chromosome to give a new offspring. This new generation is then again evaluated using the fitness function and the process continues till stopping criteria is reached. Stopping criteria is met when the algorithm encounters one of the following two conditions.

1. Generation limit, i.e., the number of specified generations is exceeded, or
2. Average cumulative change in the fitness values between consecutive generations is less than the fitness threshold value.

The values of the above described parameters used in the algorithm have been tabularised in Table 5.2.

Population Size and Number of generations

The algorithm is tested with many different shapes of 1-ring neighborhoods for deciding the parameters for the GA. The advantage of a using a 1-ring representation of the vertex is that it is easy to visualize the change in location of the vertex and the computation of distortion is local to the 1-ring vertex. Different sets of experiments were conducted to determine the population size and the number of generations. Black markers in Figure 5.3 to Figure 5.6 correspond to the best fitness value from the pool of chromosomes for each generation whereas the mean fitness value is denoted by blue markers. 4 sets of experiments are performed on 3 models to determine the optimum combination of population size and the number of generations. Figure 5.3 to Figure 5.6 indicate that a population size of 100 run for 20 generations are the



Population Size	100	
No. of Generations	20 generations	
Chromosomes	x, y, z co-ordinates of the vertex V’	
Initial range	Upper Bound	Maximum values of x, y, z co-ordinates in the one ring neighborhood
	Lower Bound	Minimum values of x, y, z co-ordinates in the one ring neighborhood.
Creation Function	Uniform	
Selection function	Stochastic uniform	
Crossover Operator	Scattered Crossover	
Crossover Rate	0.8	
Mutation Operator	Uniform Mutation	
Mutation Rate	0.001	
Elite count	2	
Threshold fitness value for stopping criterion	10^{-9}	
Stopping Criterion	20 Generations without a change in fitness value greater than threshold fitness value	

Figure 5.2: GA Algorithm Parameters

most optimal values. When a GA is run for 20 generations with a population size of 20, the perceptible distortions is high. This is because the initial pool of chromosomes is scattered randomly within the range. Since the population size is just 20, the chances of these chromosomes being closer to the center vertex of the 1-ring are very low. When the algorithm with these parameters is run for only 20 generations, there is not enough time for the GA to converge at a value close to the center vertex. When the GA is run for 100 generations with population size of 100 the algorithm is computationally expensive. As seen in the Figure 5.9, the perceptible distortion with these parameters is very low. But the same degree of low perceptible distortion is achieved with a population size of 100 run for just 20 generations and this also decreases the computational costs. When the population size is 20 and is run for 100 generations, the initial pool of chromosomes is once again scattered within the range and the chances of them being close to the center vertex are very low. Thus, over a period of 100 generations, much of the newer chromosomes are produced as a result of mutation, i.e., mutation will take place at least 100 times, once for each generation. Mutation is replacement of a chromosome with a random value. This random value may or may not have a good fitness value. Thus, as the results show, the performance of this combination is not as good as the performance of the combination of population size 100 run for 20 generations. The graphs in Figure 5.3 to Figure 5.6 show that there is no considerable change in the best fitness value after 20 generations. Thus, the parameters chosen for embedding the watermark was population size is 100 and number of generations is 20.

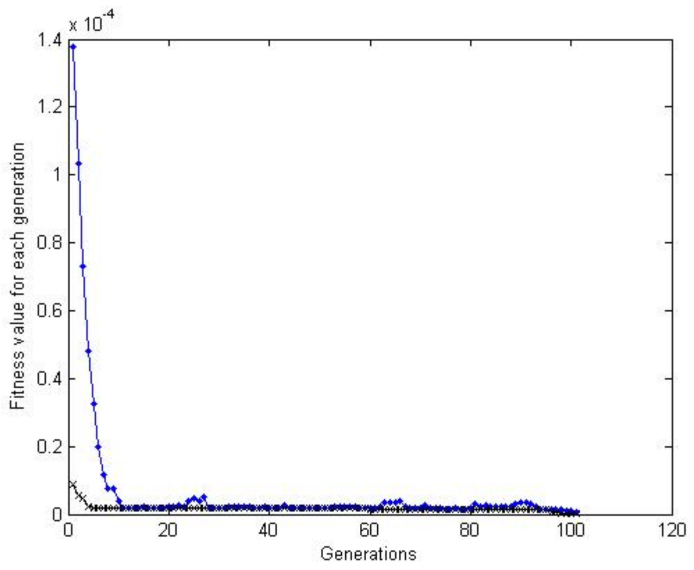
5.2.2 Fitness Function

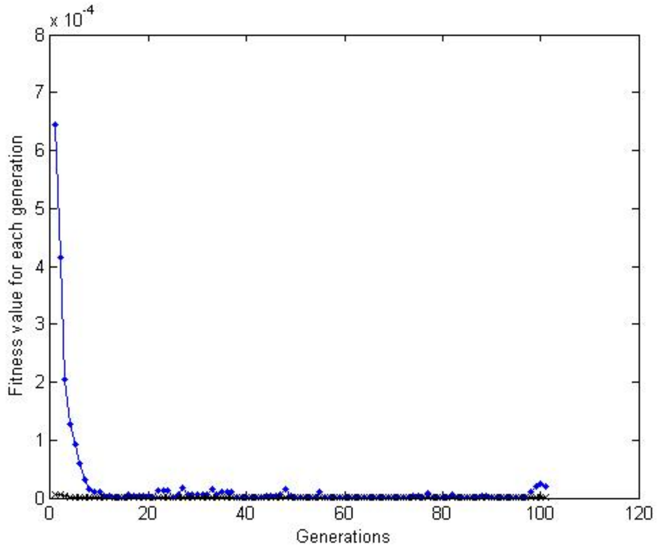
The fitness function plays the most important role in optimizing the amount of watermark and the distortion. Different fitness functions as listed below are evaluated.

Hausdorff distance based Fitness function

Hausdorff distance [82] is calculated to find the degree of mismatch between two polygons or broadly, two sets of points in space. In this fitness function, we have used the Hausdorff distance (HD) to find the degree of mismatch between the 1-ring neighborhoods of the original vertex V and the watermarked vertex $V^{'}$. This degree of mismatch is reduced by minimizing the Hausdorff distance and thus preventing perceptible distortion.

$$FitnessFunction = H(v, v^{'}) = max\{h(v, v^{'})h(v^{'}, v)\} \tag{5.4}$$





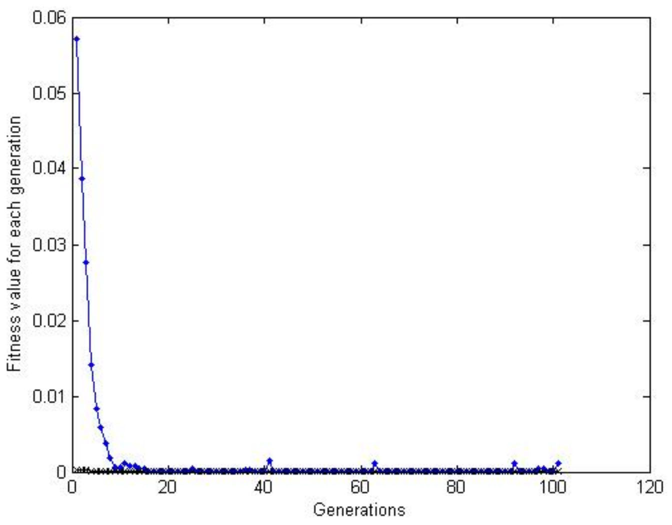


Figure 5.3: In running for 100 generations with a population size of 100, the change in best fitness value and the mean fitness value: (a) for vertex index 10 of the Smiley, (b) for vertex index 10 of the Mannequin, and (c) for vertex index 20 of the Mechanical.

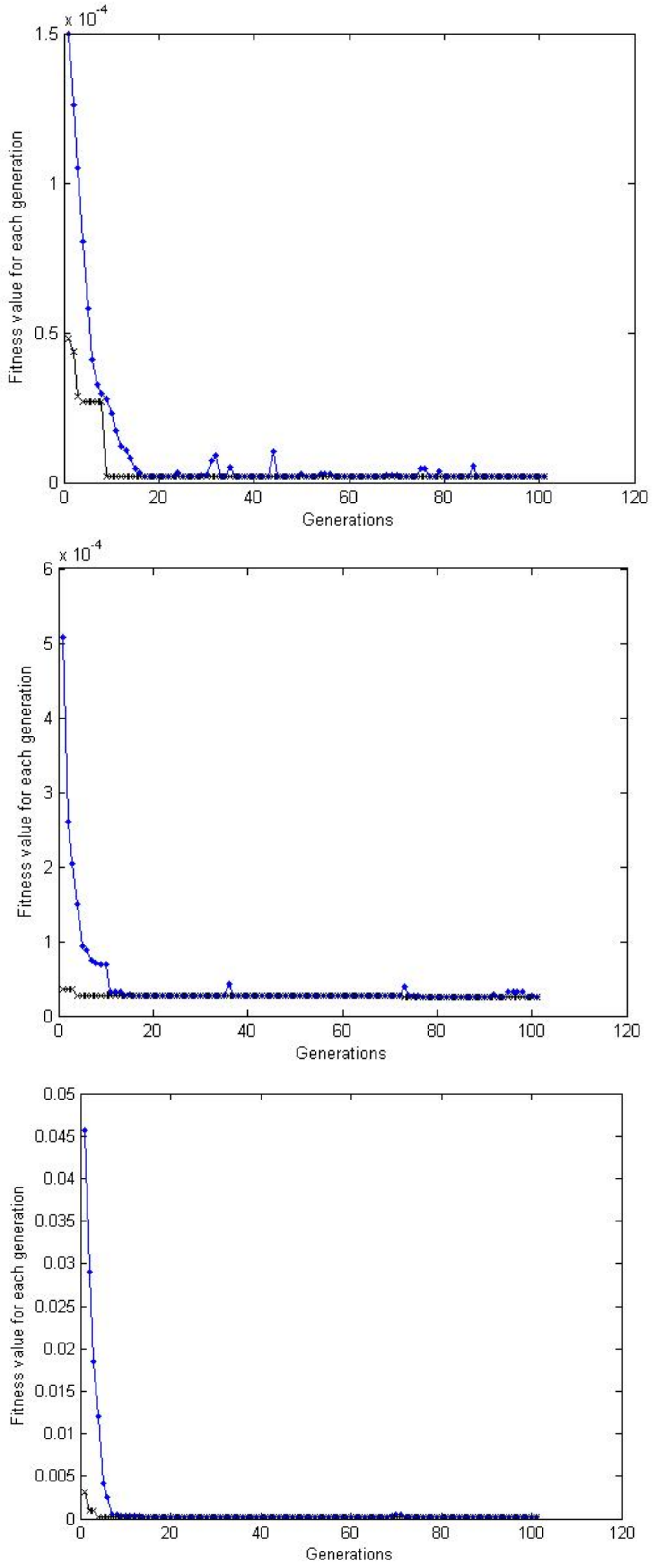


Figure 5.4: In running for 100 generations with a population size of 20, the change in best fitness value and the mean fitness value: (a) for vertex index 10 of the Smiley, (b) for vertex index 10 of the Mannequin, and (c) for vertex index 20 of the Mechanical

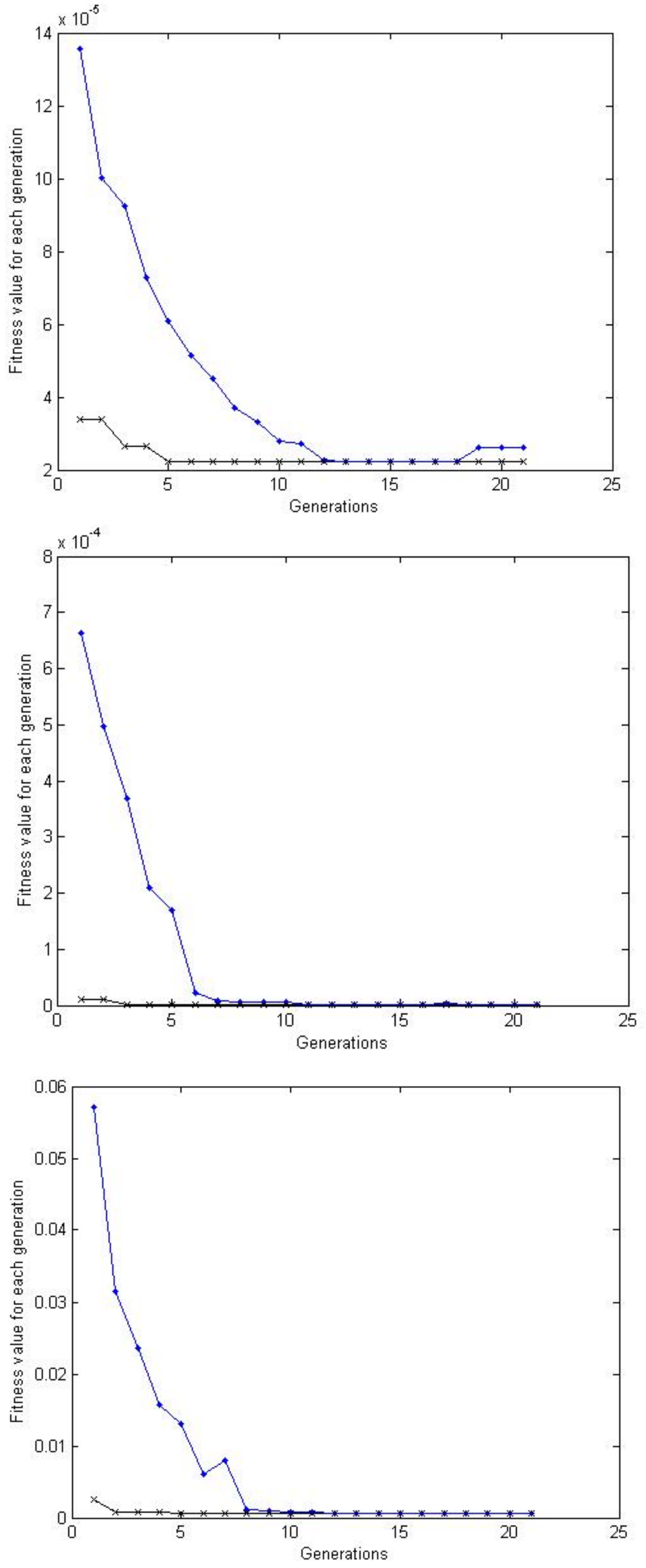


Figure 5.5: In running for 20 generations with a population size of 20, the change in best fitness value and the mean fitness value: (a) for vertex index 10 of the Smiley, (b) for vertex index 10 of the Mannequin, and (c) for vertex index 20 of the Mechanical

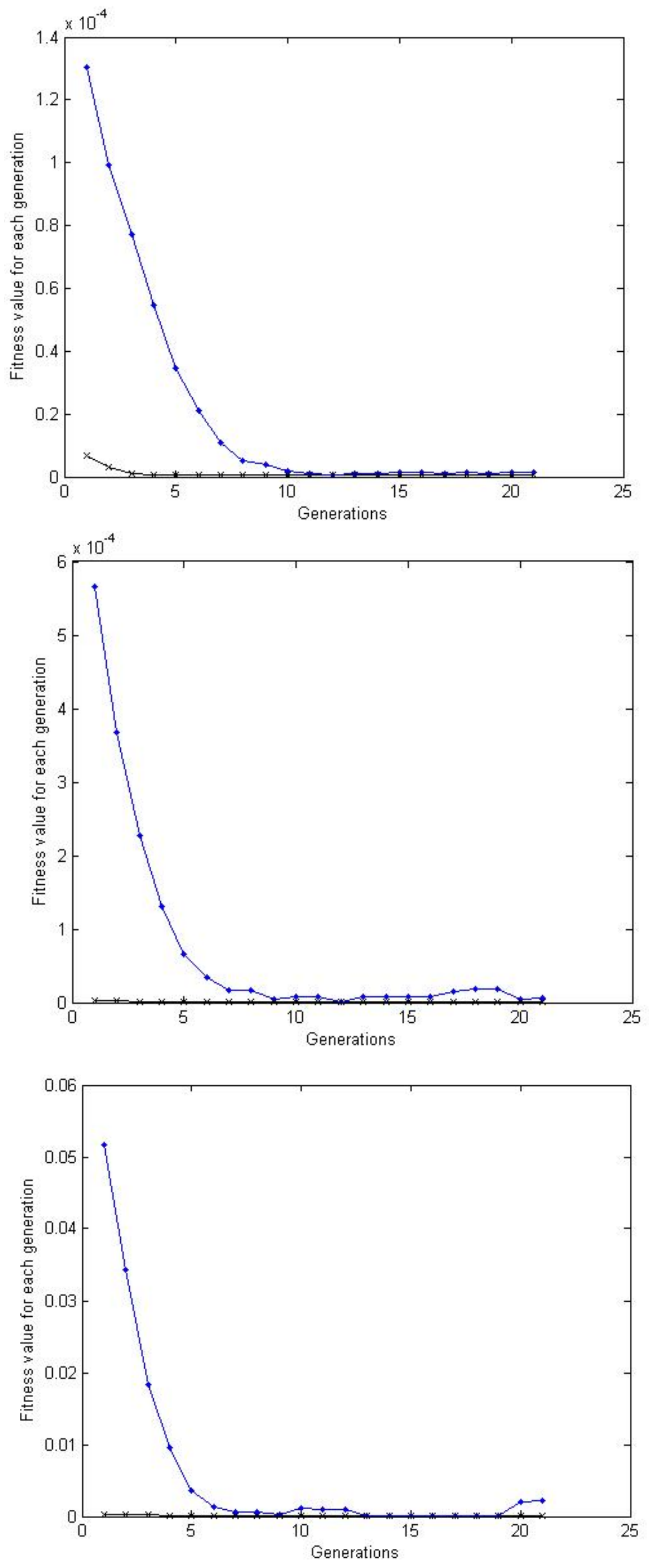


Figure 5.6: In running for 20 generations with a population size of 100, the change in best fitness value and the mean fitness value: (a) for vertex index 10 of the Smiley, (b) for vertex index 10 of the Mannequin, and (c) for vertex index 20 of the Mechanical

Curvature based Fitness function

The objective of a GA using the curvature based fitness function is to optimize the change in magnitude of cross product of the surface normals of the faces constituting the 1-ring neighborhood of the center vertex and the candidate watermarked vertex. The fitness function is choosen so that the change in direction of the surface normals of the local geometry due to the addition of the watermark in the vertex coordinate is minimal. To maximize the strength of the watermark, it is essential that the vertex V is moved by a considerable amount of distance to $V^{'}$. For this purpose we take the root mean square distance of $V^{'}$ from each and every triangle in the 1-ring neighborhood of V. To have minimal distortion, it is essential that the surface normals of the 1-ring neighborhood of $V^{'}$ have nearly the same direction as that of the surface normals of the one ring neighborhood of V. In this regard, we will take the summation of the magnitudes of the cross products of the one ring neighborhood surface normals corresponding to V and $V^{'}$. The point-to-triangle distance is obtained by considering the relative position of the point from the triangle and finding out if the projection of the point is closer to an edge or a vertex or if it projects inside the triangle and accordingly calculating the distance of the point from the triangle. If we consider only the distance parameter in our fitness function, though the Mean Square error is high, the amount of distortion is also considerable and in the case where only the sum of cross products in considered, the amount of watermark and hence the mean square error (MSE) is low.

Thus, the fitness function can be mathematically expressed as:

For non-flat surfaces

$$FitnessValue = \frac{(\sum_{i=1}^{n} \|(S(i) * S^{'}(i))\|)^{4}}{D^{1/4}} \tag{5.5}$$

where,

n= no. of faces in the 1 ring neighborhood,
S(i) = Surface normal of the ith face with the original vertex V,
$S^{'}(i)$ = Surface normal of the ith face with the moved vertex $V^{'}$,
D = root mean square of the distances between the vertex $V^{'}$
and the triangles in the 1-ring neighborhood of V.

Flat and Cylindrical Surfaces

The algorithm should insert a watermark in 3D models that have flat surfaces as well. This is difficult because in that condition the chromosome has to move only on a flat surface on a 2D plane. Since, the fitness function used is the VSNR, every time the chromosome moves out of the plane, the SNR decreases. Thus, the objective of the GA is to maximize this SNR, due to which the vertex, V stays in the 2D plane, as illustrated in Figure 5.7. For flat surfaces, the direction of the average normal will always be parallel to all the surface normals of the 1-ring. Thus, if the surface is flat the magnitude of the cross product between average normal and all the surface normals is equal to 0 i.e. along a 2 dimensional line, the visibility parameter will be the magnitude of cross product of the new line (formed by Vertex $V^{'}$ and V) with the desired direction and the distance is Euclidean distance between the new vertex $V^{'}$ and the old Vertex V. Point-to-triangle distance is obtained by considering the relative position of the point from the triangle and finding out if the projection of the point is closer to an edge or a vertex or if it projects inside the triangle and accordingly calculates the distance of the point from the triangle. Therefore, the fitness function for flat surfaces can be mathematically expressed as:

$$FitnessValue = \frac{\|(L * L^{'}\|}{D^{1/4}} \tag{5.6}$$

where,

L = line formed by $V^{'}$ and V
L' = desired direction of motion (formed by vertex V and the point which is farthest from V in its 1-ring neighborhood
d = Euclidean distance between V and $V^{'}$

Thus, by using the above parameters we add the watermark by moving each vertex by a particular value inside its 1-ring neighborhood. The watermarked vertex list is subtracted from the original vertex list and hence the result obtained is the amount of watermark. By using the above parameters, watermark is added by moving each vertex by a particular value inside its 1-ring neighborhood.

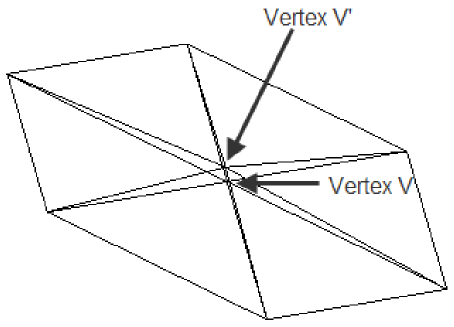


Figure 5.7: Movement of vertex on Flat surface using GA

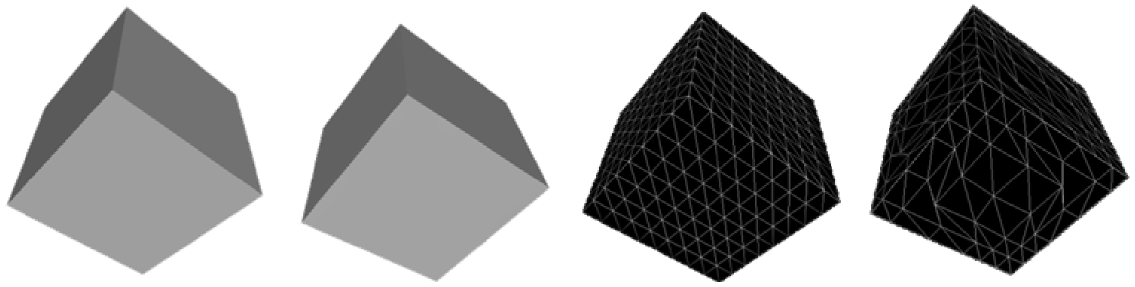


Figure 5.8: Cube (from left to right: -Original Model, Watermarked Model, Mesh structure original model, Mesh structure of the watermarked model)

Cylindrical surfaces are formed by the intersection of two planes. Thus, while watermarking the vertex of the cylinder it is required that the chromosome should move along the line of intersection of these two planes. Mechanical Figure 5.9 (e) is a combination of both cylindrical as well as flat surfaces. Figure 5.9 (e) shows genetic algorithm watermarks the mechanical model without any perceptible distortion. Thus, the GA works for flat as well as cylindrical surfaces.

5.3 Performance Evaluation

5.3.1 Perceptibility Experiments

The algorithm was run with population sizes population sizes of 20 and 100 and with 20 and 100 generations. The algorithm was tested for 5 different models: bunny, mannequin, smiley, horse, and mechanical. The results in Figure 5.9 indicate there is no perceptible distortion between watermarked and original models. Table 5.1 shows the change in SNR with the change in population and generation sizes.

Table 5.1: Average time per vertex and Vertex SNR

Model	Population Size	Number of Generations	Time (sec)	SNR (dB)
Mannequin	100	100	1.340	144.30
	100	20	0.480	126.60
	20	100	0.420	117.00
	20	20	0.240	103.00
Smiley	100	100	1.33	146.00
	100	20	0.320	121.25
	20	100	0.480	119.00
	20	20	0.200	97.88
Mechanical	100	100	1.090	126.73
	100	20	0.285	102.81
	20	100	0.325	101.34
	20	20	0.110	82.15
Horse	100	100	1.440	156.34
	100	20	0.370	133.50
	20	100	0.438	130.35
	20	20	0.218	113.56
Bunny	100	100	1.475	158.68
	100	20	0.430	137.75
	20	100	0.486	134.87
	20	20	0.220	109.36

5.3.2 Attacks

The attacks for fragile watermarking of 3D model differ from the attacks for robust watermarking. The attacks on the fragile watermarked 3D model deal only with unauthorized modification of any region in the model.

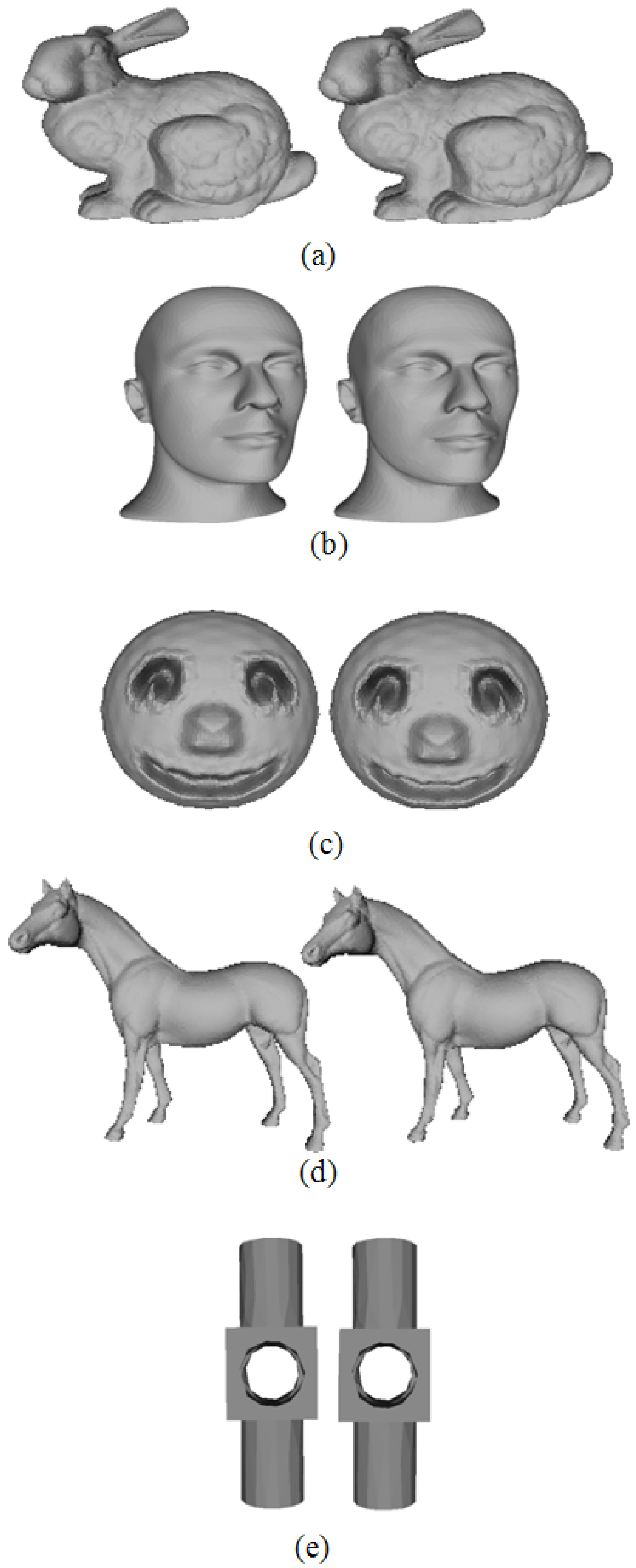


Figure 5.9: Perceptibility using GA. Original model is shown on the left and the watermarked model is on the right

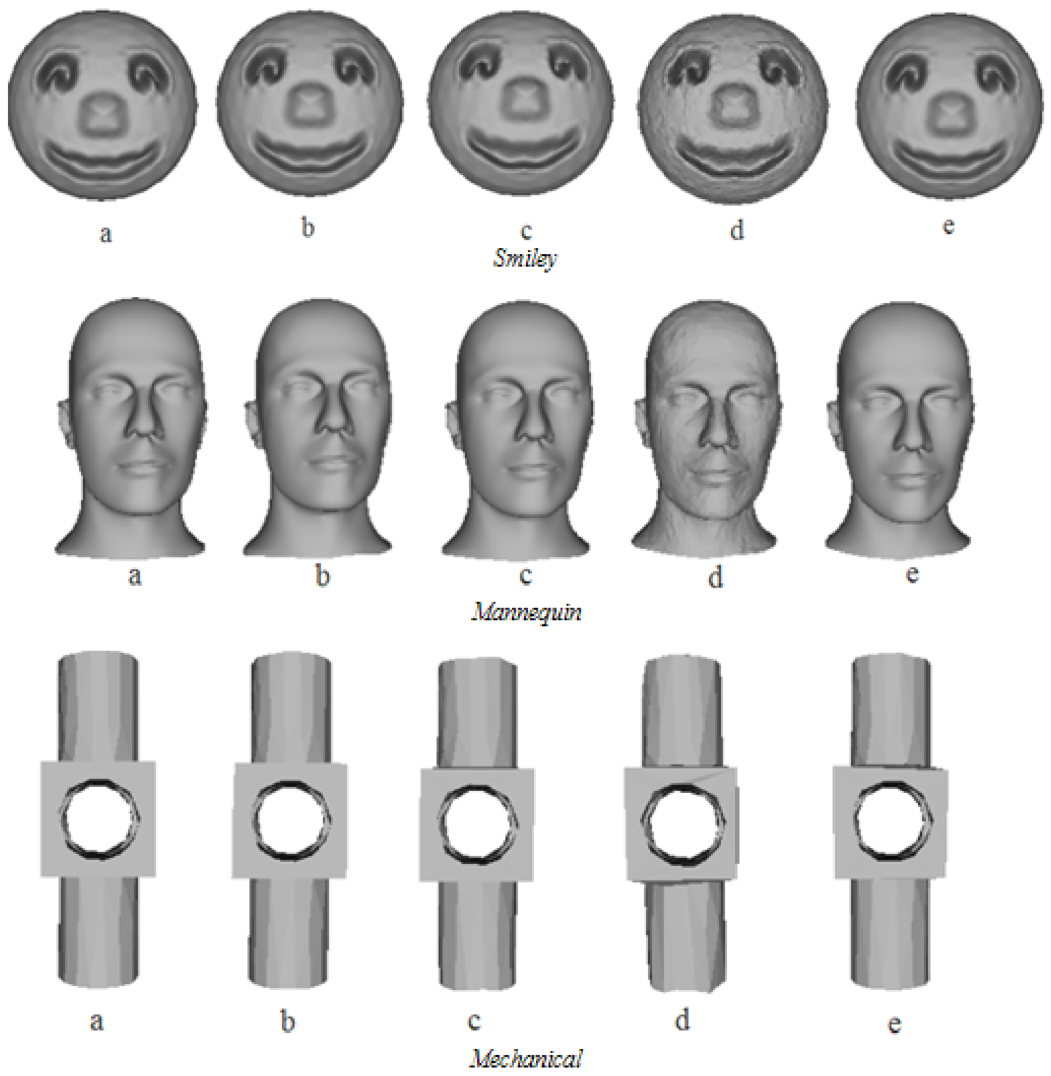


Figure 5.10: Watermarked 3D models: (a) original model, (b) watermarked model with a population size of 100, run for 100 generations, (c) watermarked model with a population size of 20, run for 100 generations, (d) watermarked model with a population size of 20, run for 20 generations, and (e) watermarked model with a population size of 100, run for 20 generations.

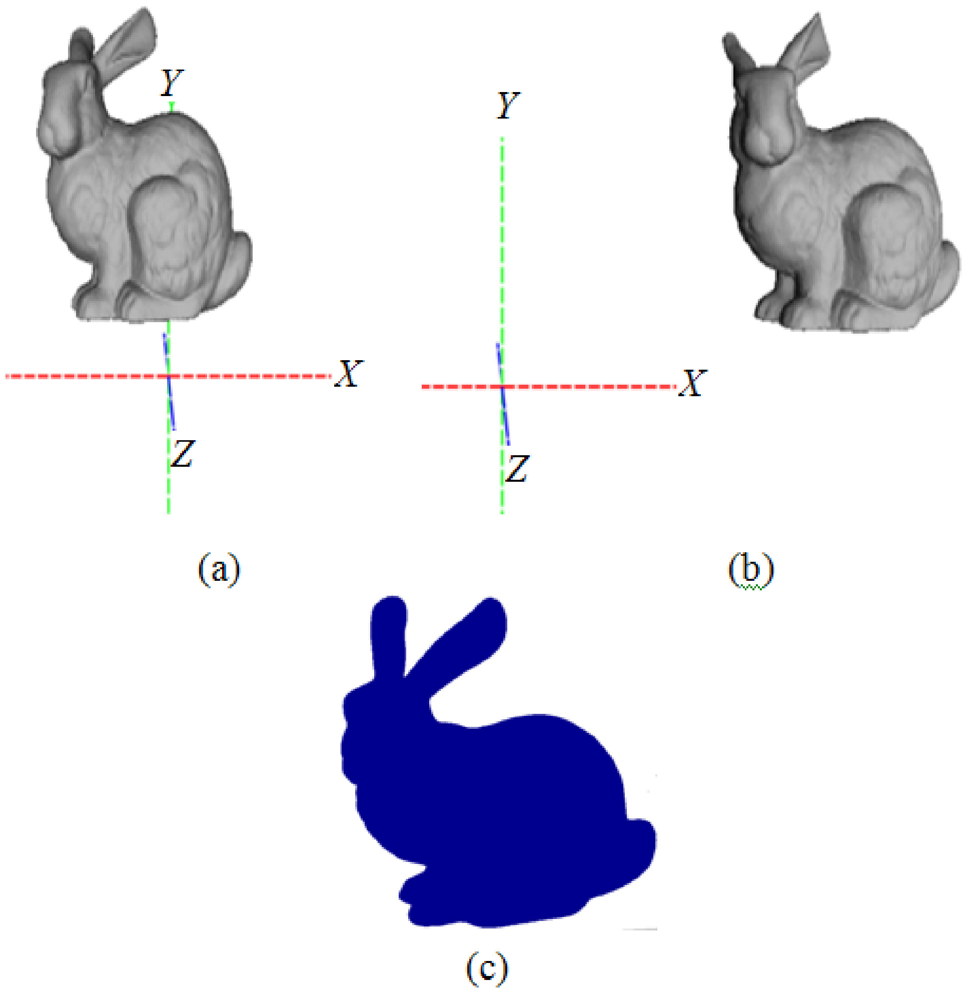


Figure 5.11: Attacks on bunny: a) GA watermarked bunny, (b) translated model, and (c) tamper region detection.

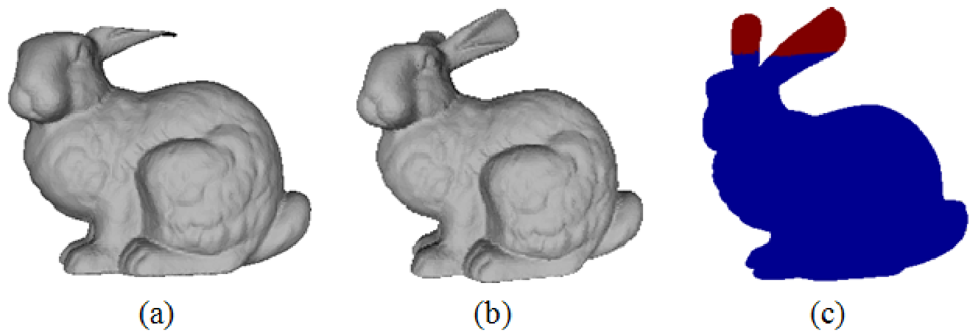


Figure 5.12: Attacks on bunny: (a) original bunny, (b) deformed bunny, and (c) modified region shown by colored (red) patch.

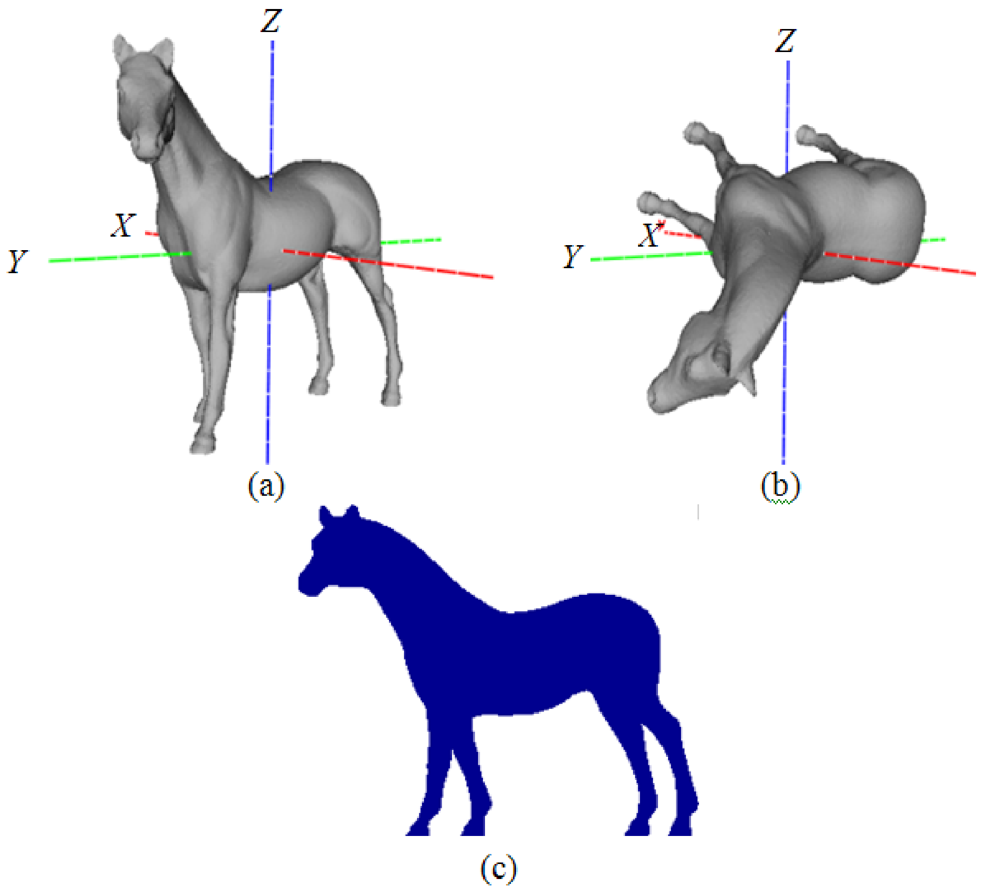


Figure 5.13: Attacks on horse: (a) original horse, (b) rotate by 250°, and (c) no red patches indicating the model has not been modified.

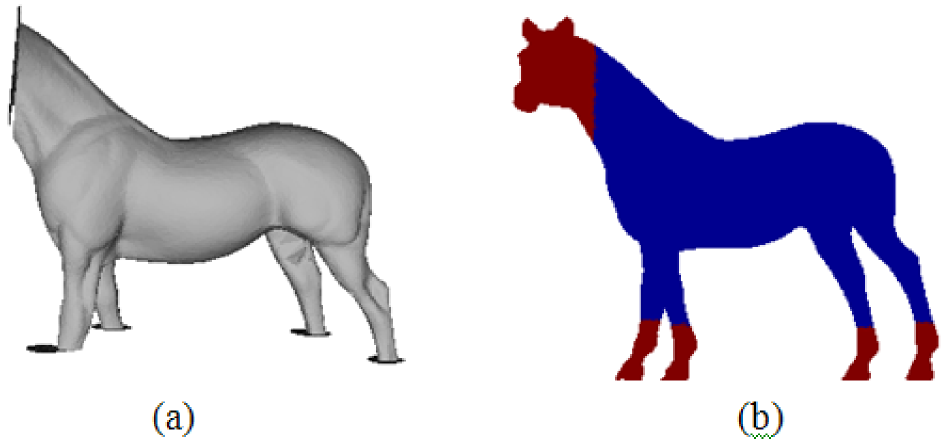


Figure 5.14: Modification detection: (a) horse deformed and (b) deformed regions.

Due to normalization of the model as a preprocessing step, the watermark is not affected by attacks such as translation, rotation and scaling as shown in Figure 5.11 to 5.14. Correlation of 100% is obtained for affine transformations. Note that the model has not been tampered with since there are no colored (red) patches in Figure 5.13. Similarly, Figure 5.14 shows red patches for the tampered regions in the model.

5.4 Analysis

In this chapter, novel fragile watermarking technique for tamper detection using Genetic Algorithms has been proposed. The algorithm generates and embeds a watermark in each and every vertex of the model. The use of signal to noise ratio as the fitness function prevents any perceptible distortion in the model. The use of the Hausdorff Distance as the fitness function is also effective in preventing any perceptible distortion in the model. Also, even though Genetic Algorithms are known to be slow, the approach used here reduces the computational costs significantly. The premature convergence was achieved by running the algorithm with a population size of 100 for 20 generations and was found to provide satisfactory results. The average time for watermarking each vertex was approximately 1.5 seconds. The algorithm has been analyzed for flat and cylindrical surfaces. It was shown that the mesh structure in case of flat surfaces is distorted though the model is not, which indicated insertion of a significant amount of watermark. The algorithm was tested for the unauthorized structural modifications in the model. It was shown that the algorithm detects the region of tamper effectively. The objective of the Genetic Algorithm is to minimize the vertex signal to noise ration or the Hausdorff distance between the 1-ring neighborhood of the original and the watermarked vertex. The other challenge of time complexity is overcome by running the Genetic Algorithm for just 20 generations and causing it to converge prematurely. This significantly reduces the computational cost. The experimental results indicate that the algorithm effectively detects any distortion in model. The Genetic Algorithm has to generate a watermark in 3D models that have flat surfaces as well. This is difficult because the vertex V can move only on a flat surface on a 2D plane. This is ensured by the fitness function. Figure 5.8 shows a cube where the mesh structure is seen to be distorted but the model remains undistorted.

5.5 Summary

This algorithm presented in this chapter is computationally expensive but achieves high watermark embedding capacity by allowing watermark to be inserted in all the vertices of the model without causing any perceptible distortion. Experimental results indicate that the algorithm can detect any tampering in the watermarked model. All the simulations were run on a machine with a 2GHz Intel Processor and the algorithm took approximately 1.5 seconds to watermark one vertex. Up to 50%-60% of the model's vertices are modified without causing any distortions for robust watermarking. This algorithm has demonstrated great potential to embed high capacity watermarks.

Chapter 6

Artificial Neural Networks

Artificial Neural Networks (ANN) [35, 36] are an algorithmic modeling of biological neural systems. The neuron, which is the basic component of an ANN, upon activation fires an output signal corresponding to a set of input signals. A neuron receives input signals p_i and aggregates these signals into a net input signal n by multiplying each input signal with corresponding numerical weights w_i and summing up all of these computed numerical values along with a bias b, as shown in Figure. 6.1.

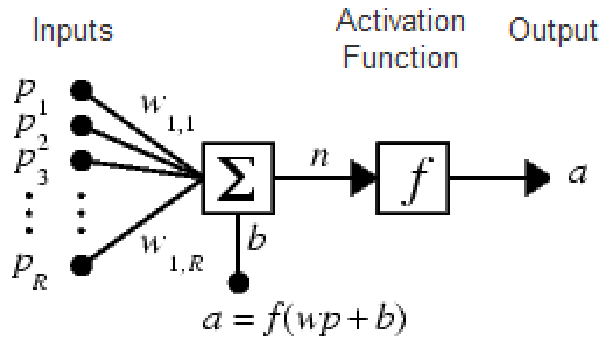


Figure 6.1: Neuron - The Building Block of a ANN [31]

The output signal a is computed by an activation function f that takes n as the input. An activation function can be linear or non-linear in nature. The most commonly used activation functions, such as *sigmoid* and *hyperbolic tangent*, map n to a in a non-linear way. The neuron transmits an output signal only when it is activated i.e. the net input signal falls within the working range of the activation function. The bias is used to change the threshold at which a neuron activates and is adjusted by the learning phase of the ANN. The weights w, that control the strengths of the input signals, are very critical in defining the behavior of the ANN and evolve during the learning phase as well. An ANN is a layered network of neurons (Figure. 6.2). The goal of the learning phase of ANN is to determine the best values for w and b from a given set of data and adjust these values until a certain criterion is satisfied.

Neural networks are trained to perform a particular function by adjusting the values of the connections (weights) between elements. Neural networks are trained so that a particular input leads to a specific target output. During training the network parameters or weights are adjusted, based on a comparison of the output and the target, until the network output matches the target. Typically many such input/target pairs are needed to train a network. Once trained, the neural network can be used to determine the output when input is fed.

In supervised learning, the neuron is provided with a training data set consisting of input vectors and a target (desired output) associated with each input vector. A supervised learning ANN uses the target vector to determine how well it has learned, and to guide adjustments to weight values to minimize the overall error between the real output of the neuron and the target output.

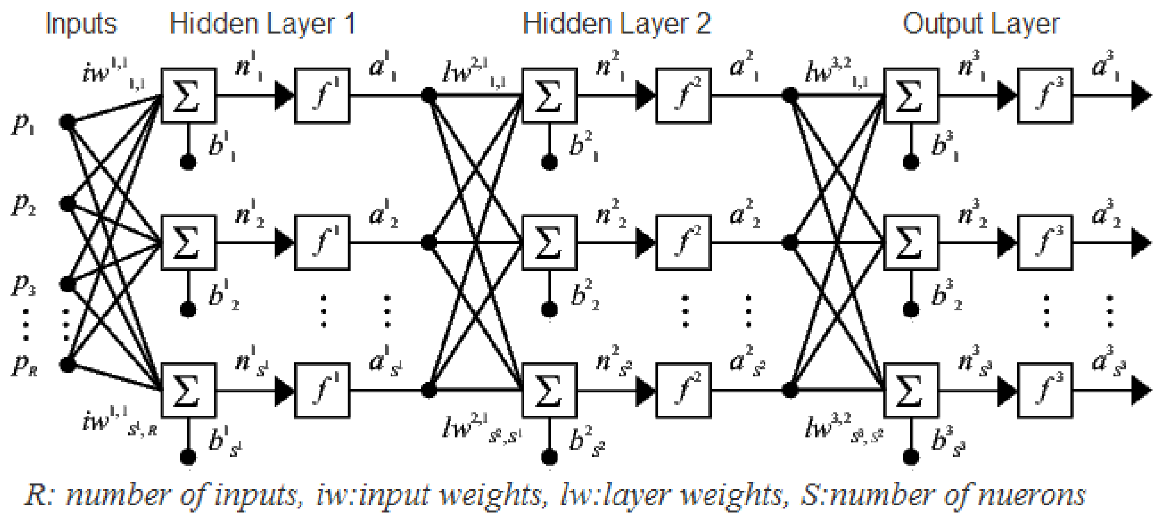

Inputs
Hidden Layer 1
Hidden Layer 2
Output Layer
R: number of inputs, iw:input weights, lw:layer weights, S:number of nuerons

Figure 6.2: Multi-Layer ANN [31]

A feedforward ANN propagates the signals through all the layers to obtain the result, which is the output of the last layer in the network. Backpropagation is one of the architectures used to train the ANN such that the output of the network is an accurate approximation of the target values. During the learning iterations, the output value of the ANN for each training pattern is computed and the error signal is propagated back from the output layer toward the input layer so that the weights are revised appropriately.

For the proposed approach, described in the following section, the neural network is used as a classifier to determine whether a vertex is suitable for watermark insertion.

6.1 Algorithm

6.1.1 Block Diagram

The objective of the artificial neural network is to watermark a 3D model in those locations which will produce imperceptible distortions in the final watermarked model. This task can be achieved by selecting vertices where addition of watermark data will not produce visible distortion. The artificial neural network needs to be trained to recognize different topologies of 1-rings of vertices of a model. Figure 6.3 shows the process of training the neural network by feeding the geometry of 1-ring vertices as feature vectors.

An artificial neural network is adopted for selecting vertices that are classified as suitable for watermark insertion. A variety of 3D models with varying degrees of surface curvature are used to train the neural network. An array of neural networks is used for vertices with different valences to achieve higher watermark embedding capacity. The watermark extraction process is informed and needs the original watermark and 3D model.

The input vectors that are used to train the neural network are derived from the 1-ring neighborhoods of each vertex. The 1-ring neighborhoods take into account the local geometry of the vertex. Based on analysis of the local curvature which is estimated by the angles between surface normals of a neighborhood, a neural network

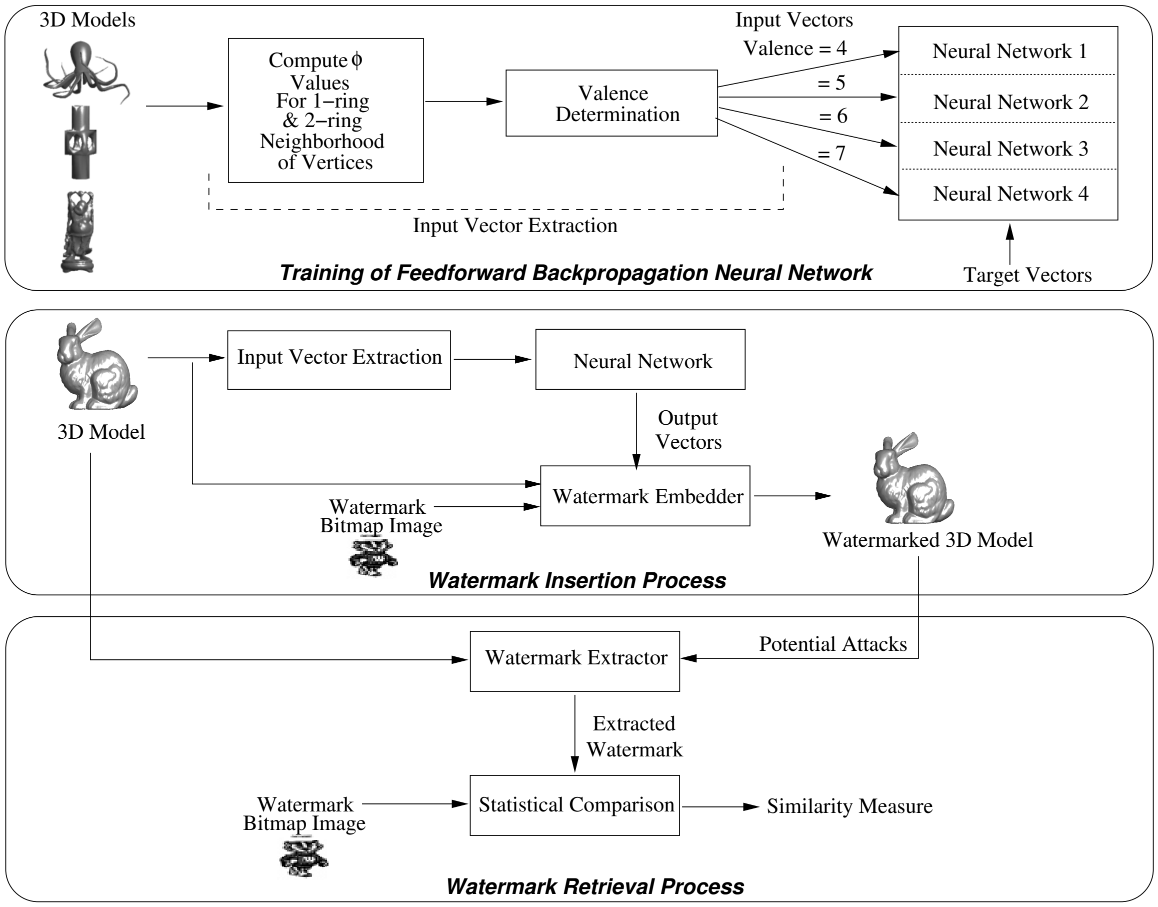


Figure 6.3: ANN System Block Diagram

is trained so that it can appropriately choose regions from any 3D model to embed the watermark. Figure 6.4 demonstrates the surface normals for 3D models with varying levels of curvature. For flat surfaces, the angles between surface normals are small in magnitude since these normals are almost parallel to each other. For regions representing edges, the angle between the neighboring normals is much larger in magnitude. Smoother regions like the *Mushroom* top have relatively smaller variations in orientation of the surface normals. Thus, angular difference between neighboring surface normals represents the curvature or local shape of a region and is an appropriate input vector for training and simulation of the artificial neural network. The neighborhood size is restricted to 1-ring so that local details of surfaces are not lost.

Figure 6.3 outlines the system block diagram and the following subsections de-

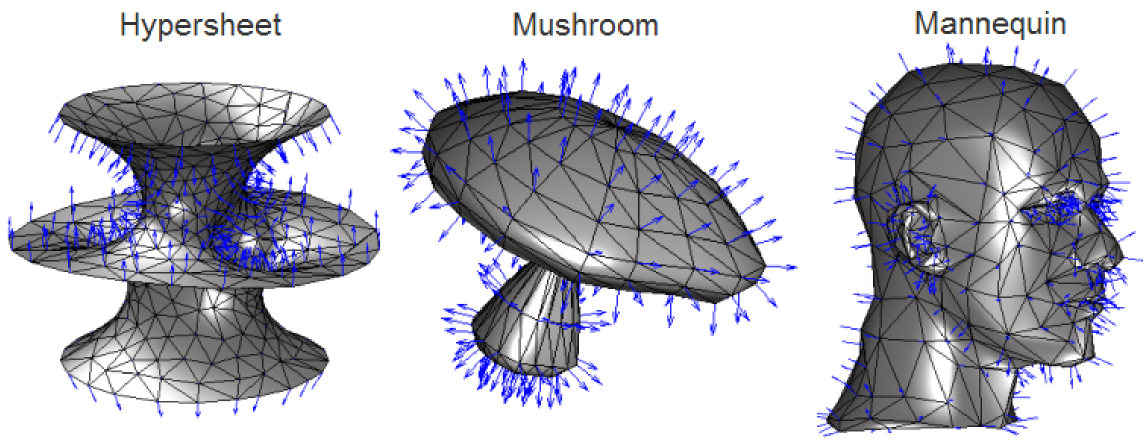


Figure 6.4: Curvature Estimation From Normal Vector Distribution

scribe the three blocks of the system - Neural Network Training, Watermark Insertion, and Watermark Extraction.

6.1.2 Feature Vector

A feature vector includes each of the angles formed between the surface normal corresponding the triangular face with v as one of the vertices and the average normal as shown in Figure 6.5. This feature vector represents the geometry of the 1-ring vertex neighborhood. Valency of a vertex is the count of how many other vertices the vertex is connected to in the 3D model. Thus, the length of the feature vector would be equal to the valency of the vertex. However, feature vectors used for training of ANN have to be of fixed length. Thus, vertices with only valency 6 are selected for feature extraction. Semi-regular meshes have vertices with valency of either 6 or 4. Valency 6 vertices are also called regular vertices since most of the vertices in a 3D model have valency 6. Thus, the feature vectors used in the proposed algorithm also have length of 6.

The following steps are implemented to compute the curvature value of a vertex:

Step 1: Consider a vertex v with valency equal to 6 from the mesh. Let M be the number of its adjacent faces which are equal to 6. Find normal's N_i to each face which is formed by v and its neighboring vertices v_i as shown in Equation 6.1

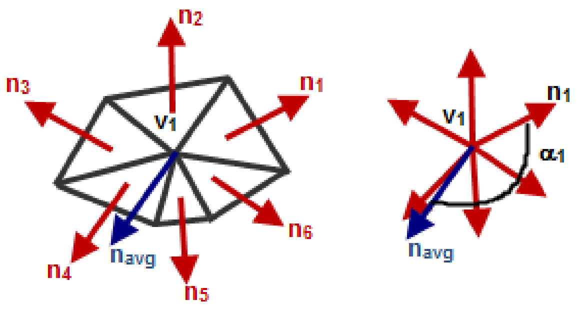


Figure 6.5: Surface normals (in red) and average normal (in blue) for a 1-ring vertex neighborhood.

Step 2: Find the average resultant vector N_{avg} of all the above normals passing through v.

$$N_{avg} = \frac{1}{M}\sum_{i=1}^{M} N_i \tag{6.1}$$

Step 3: Now compute angles α_i between each pair of N_i and N_{avg}.

$$\alpha_i = \cos^{-1}\left(\frac{N_i \cdot N_{avg}}{|N_i||N_{avg}|}\right) \tag{6.2}$$

Step 4: Compute angles which form the feature vector to be fed to ANN.

$$FeatureVectorF = [\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4\ \alpha_5\ \alpha_6] \tag{6.3}$$

Thus, if the region around the considered vertex is flat, the angles α_i will be small in magnitude since the face normals will be almost parallel to the average normal. However, if the region represents a peak, the angle between the face normal and the average normal through the vertex, will have a larger magnitude and so the smoothness measure's magnitude will be higher. Thus, the selected feature vector represents local geometry or shape of a surface or region and is fed as features vectors to ANN for training and simulation.

6.1.3 Training of Artificial Neural Network

The artificial neural network is trained to recognize which types of one-rings are suitable for insertion of watermark data, and which ones are not. The angle variations of each vertex having valency 6 as computed in Equation 6.3 will be fed to the neural network in order to train it to recognize different types of vertices. Figure 6.6 shows the block diagram for the training the artificial neural network.

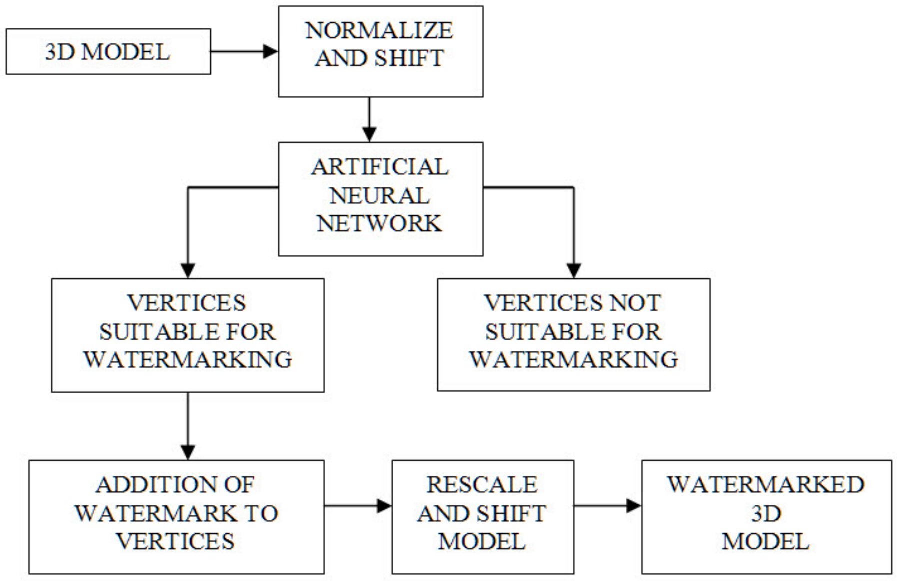

3D MODEL
NORMALIZE AND SHIFT
ARTIFICIAL NEURAL NETWORK
VERTICES SUITABLE FOR WATERMARKING
VERTICES NOT SUITABLE FOR WATERMARKING
ADDITION OF WATERMARK TO VERTICES
RESCALE AND SHIFT MODEL
WATERMARKED 3D MODEL

Figure 6.6: Training of Artificial Neural Network

Figure 6.7 shows the architecture of the back propagation neural network being used to select vertices for watermarking. The output of the neural network is trained to be either 1 for a suitable vertex, or -1 for a non-suitable vertex.

Training is done by manually adding noise to the vertices and the human operator determines if the addition of noise is perceptible or not. The vertices in which visible distortion is produced after adding a small amount of information are marked as unsuitable for watermark insertion. These will be vertices having a surrounding ring which is flat, or they could be the peaks of raised surfaces. On the other hand, the vertices in which addition of large amount of information does not produce visible distortion are marked as suitable for watermark insertion. 162 sets of vertex rings

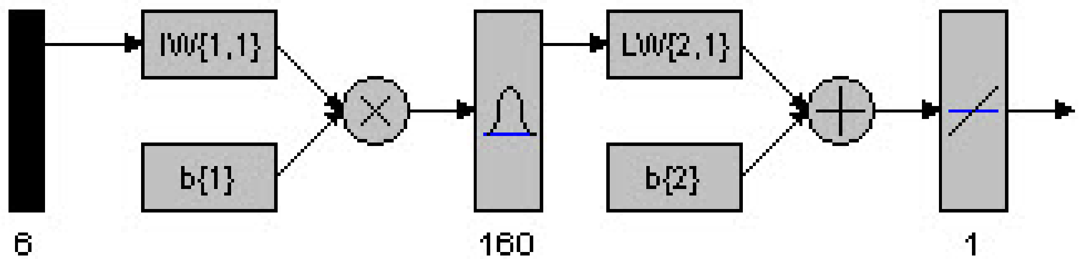


Figure 6.7: Neural Network Architecture

with different geometrical structures were extracted from 5 normalized 3D models and were evaluated by 2 human operators to reduce the human error of deciding whether to insert the watermark or not. Since there is no Human Visual System for 3D models, the labelling of output vectors is a manual process. Thus, human intelligence is transferred to the classifier by using using training data which was manually generated. The neural network was trained to recognize vertices suitable for watermarking according to the topology of the one-ring around the vertices. 20 neurons were in the hidden layer.

Figure 6.8 shows the output produced after testing of the trained neural network. The output produced by the neural network follows the actual output very closely. Thus, the neural network can be said to be successfully trained.

6.1.4 Insertion of watermark

Figure 6.6 gives the overview of the watermarking insertion process. Feature vectors extracted from vertices with valency equal to 6 are extracted from the entire model and fed as inputs to the neural network. The trained ANN then selects the vertices for watermarking. Fig. 6.9 shows these vertices (in dark red) in the models. It can be easily verified from these figures that the neural network is indeed choosing vertices on the mesh which are suitable for watermarking, and no vertices are selected from smooth regions such as the Mannequin's forehead. We insert a random sequence in the selected vertices. The difference between the watermarked vertices and the

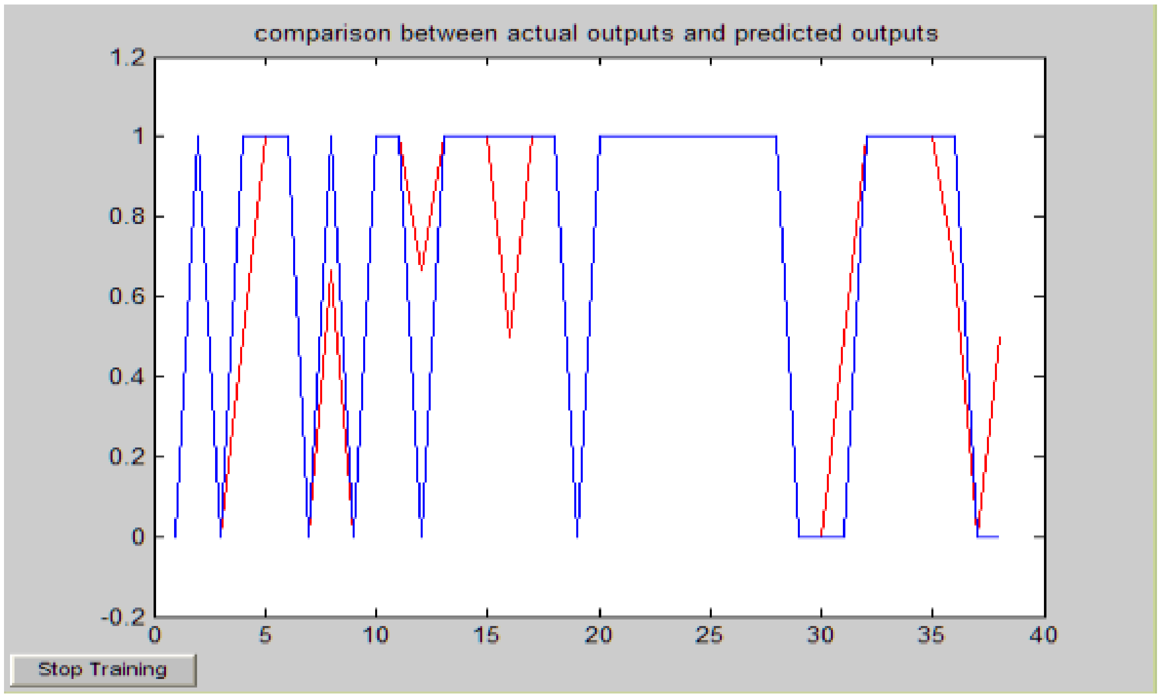


Figure 6.8: Testing of Neural Network (Blue: Expected Output, Red: Actual Output)

original vertices is the watermark. Thus, for each co-ordinate of a vertex selected to be modified, we have:

$v'(x, y, z) = v(x, y, z) + KW$ where,

v'=Watermarked Vertex,
K=Scaling Factor,
W=Watermark Data.

Finally, the model is re-shifted to its initial location in space and the co-ordinates are also re-scaled. Thus, the watermark is inserted in the geometry of the model and this model can be distributed for use by others. The watermark inserted can be the logo of a company, the designer's identification, the user's signature or any other intellectual property. This watermarking method modifies only the locations of vertices, without changing the connectivity of vertices.

This demonstrates that the neural network adds watermark data to those vertices where visible distortion will not be produced in the model.

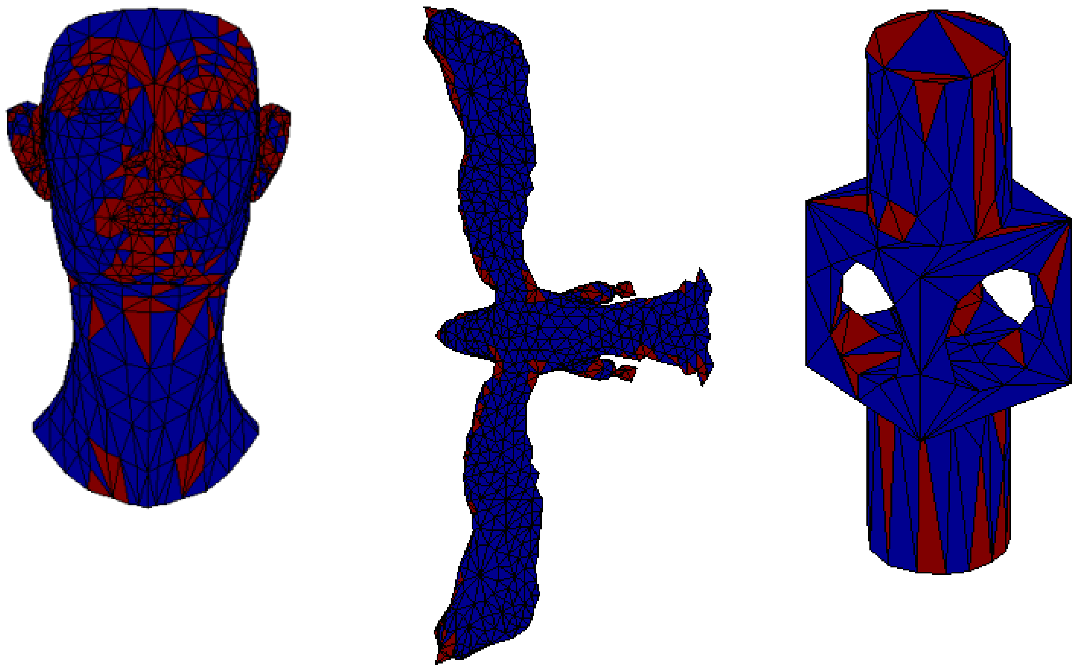


Figure 6.9: Vertices selected for ANN based watermarking in red

6.1.5 Extraction of Watermark

The process of semi-blind watermark extraction is as shown in Figure 6.10. In the watermark extraction process, the watermarked 3D model is normalized and shifted to the origin. The private key, which contains the indices of the watermarked vertices, is used to extract the watermark from those vertices. Finding the correlation is a common method used to determine the extent of similarity between the original watermark, and the extracted watermark, as seen in Chapter 3. The output of the correlator is a Pearson's correlation coefficient. The extracted watermark and the original watermark are correlated and a high ad-hoc value of correlation coefficient greater than 0.7 will prove the ownership of the 3D model.The correlation coefficient is a number between -1 and +1 which measures the degree to which two variables are linearly related.

The percentage of correlation between the recovered watermark and original watermark is 100% in the absence of any attacks on the watermarked model.

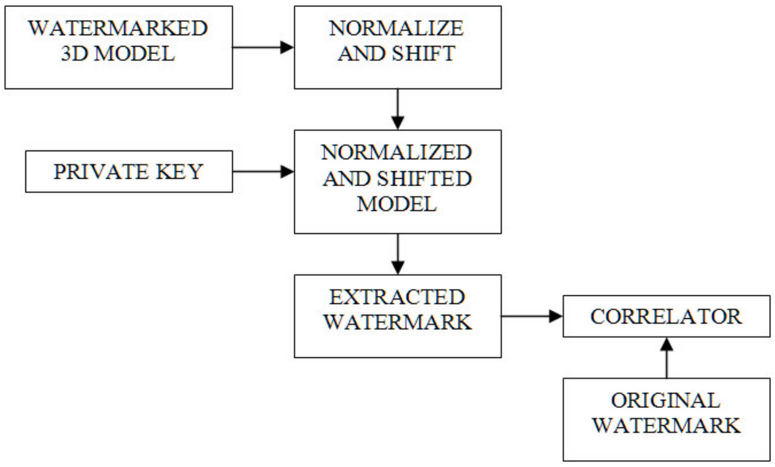


Figure 6.10: ANN Watermark Extraction Process

6.2 Experimentation

In addition to backpropogation networks, radial basis function networks were also evaluated.

6.2.1 Radial Basis Function Network

For training the neural network, 3D models with varying degrees of surface curvature are chosen. Input vectors are computed for vertices of valence 4, 5, 6 and 7 from each 3D model chosen for training. Input vectors are the values of angles ϕ between the face normals and the average normal for the 1-ring and 2-ring neighborhood of a vertex. These ϕ values are computed using the technique specified in Chapter 3 and published in [65]. The limitation in [65] was that it considered vertices of valence 6 only and the output vector specified whether a vertex was appropriate for watermark insertion or not. Here we use multiple neural networks, each trained with input vectors derived from vertices of a different valence. We use 4 neural networks, one each for vertices of valence 4, 5, 6, and 7. The output of the neural network is one of four levels(1, 2, 3, 4) used to represent the watermark embedding strength of a vertex. The watermarking algorithm selects vertices with levels 2, 3 & 4 to embed a watermark of different strength in each level. This approach increases the embedding capacity of the algorithm while maintaining imperceptibility of the watermark. To achieve rotational invariance, all permutations of the input vectors for each vertex are added to the training set.

Target vectors, for a set of input vectors, are derived from the original and watermarked models using the algorithm in [66] with the parameter for embedding capacity set at 100% (this ensures that all the vertices are modified). Subtracting vertices of the original model from the watermarked model gives the value of the watermark embedded into each vertex. The minimum and maximum value of the strength of this watermark is determined and the difference between the two values is divided into four intervals. The watermark values lying in these four ranges are assigned levels of 1, 2, 3, or 4. This level serves as the target vector for a set of input vectors corresponding to a vertex.

Input vectors and the corresponding target vectors are used to train the 4 neural networks such that the aggregate neural network can classify new input vectors by associating an appropriate output vector for each vertex.

The artificial neural network has 1 hidden layer with R=4, 5, 6, or 7 inputs and 1 output layer with S^1 inputs. The hidden layer uses a hyperbolic tangent sigmoid transfer function(f^1) while the output layer uses a linear transfer function(f^2). The hidden layer has $S^1 = 20$ neurons and the size $S^2 = 1$ of the output layer is determined by the target vector. LW denotes the layer weight matrices, IW represents the input weight matrices. 3D models different from those used in the training session are used for network simulation. The mean square error is used as the performance function to compute the error between the network outputs and the target vectors.

Watermark Insertion

The process for embedding the watermark in the 3D model is outlined in Figure 6.11. The 3D model is normalized before the feature extraction process to ensure that the curvature values are invariant of translation, scale, and orientation operations on the 3D model. The curvature values for the normalized 3D model are computed by the feature extraction block using the method discussed in the previous section. For a given 3D model that needs to be watermarked, the input vectors are computed for vertices of different valences and fed as input to the artificial neural network. The output of the artificial neural network specifies the watermark embedding capacity level of the vertex corresponding to the input vector. The watermark embedder selects vertices with level 2, 3, and 4 for watermark insertion. The algorithm perturbs a vertex by using a scaling factor for the watermark to be embedded. A scaling factor of $K_1 = 0.25 * 10^{-4}$ is used for vertex of level 2, $K_2 = 0.5 * 10^{-4}$ is used for vertices of level 3 and $K_3 = 10^{-4}$ is used for vertices of level 4. A gray scale bitmap image is used as a watermark. For each co-ordinate of a vertex selected to be modified, the modification to the vertex is determined by the following equation:

$$v'_{x',y',z'} = v_{x,y,z} + KW \tag{6.4}$$

where, v = Original Vertex,
v' = Watermarked Vertex,
K = Scaling Factor,
W = Watermark Data (image pixel value between 0-255).

The trained RBF neural network is tested with these features vectors as input for each vertex in the 3D model, to predict the watermark value at each vertex. The watermarked model is computed by randomly selecting vertices in the 3D model and adding the corresponding watermark value to those vertices. Finally, the watermarked model is de-normalized.

Watermark Extraction

The embedded watermark is retrieved from a watermarked model by subtracting the vertices of the normalized original model from the vertices of the normalized version of the watermarked model. This retrieved watermark is correlated with the originally inserted watermark to determine a measure of similarity between the two sets of watermark values. A correlation measure of 1.0 indicates a perfect match between

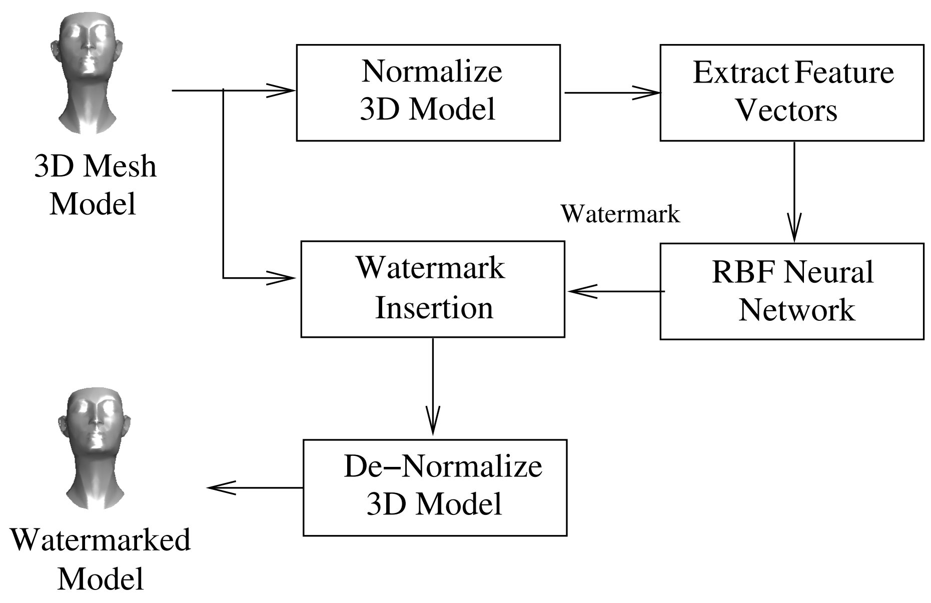


Figure 6.11: ANN Watermark Insertion Process

the original and extracted watermark. If the correlation measure is above 0.70, the test model is declared to be authentic and tamper-proof. Figure 6.12 sketches the block diagram for the non-blind retrieval process.

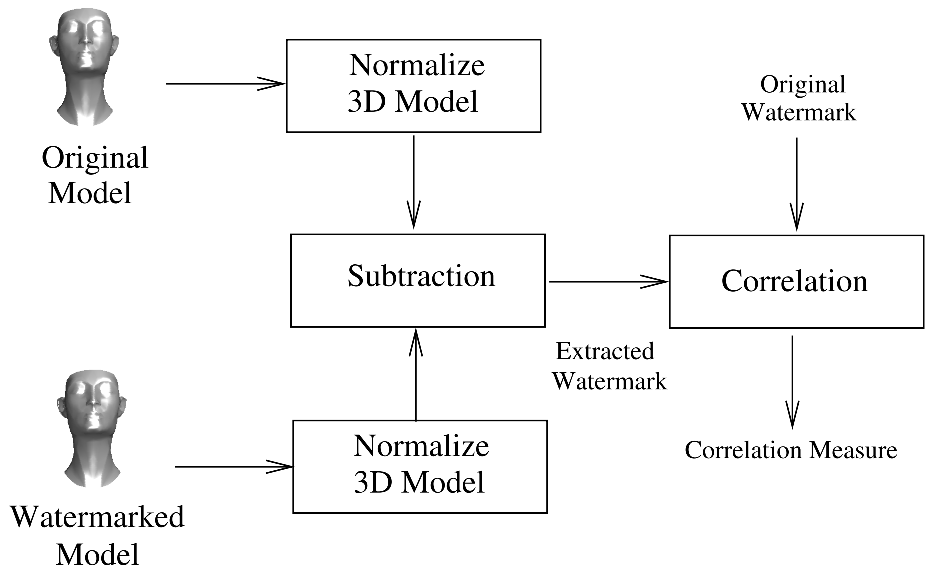


Figure 6.12: ANN Watermark Extraction Process

6.3 Performance Evaluation

An attack on a 3D model is an attempt to remove the watermark, but still retain enough of the model so that it can be used. 3D models are prone to operations like cropping, smoothing, noise addition, translation, rotation and scaling, which may destroy the watermark. This is not desired as the 3D model's ownership or copyright integrity inserted as watermark may be destroyed as well. Thus, it is important that the watermark inserted should be robust enough to handle such attacks. To prove the efficiency of our method, typical attacks were simulated on the watermarked models. Table 6.2 and Table 6.1 gives the summary of tests and results for some models.

Model Name	Number of Vertices	Number of Modified Vertices	VSNR (dB)
Cow	2904	883	98.75
Hypersheet	487	287	119.23
Mushroom	226	98	104.57
Mannequin	428	224	81.25
Eagle	1000	153	61.15
Mechanical	175	62	85.34

Table 6.1: Distortion of ANN Watermarked Algorithm

Noise This attack was simulated by adding normally distributed random numbers (with mean 0 and variance 0.3). Such an attack does affect the extracted watermark, but the correlation is still above the predetermined threshold of 0.7. This threshold was found after attacking the watermarked model, and finding how much of the original watermark remains after the noise attack.

Smoothing The HC smoothing algorithm is described in detail in [91]. Smoothing has a considerable effect on the watermarked model. By smoothing, large transitions in surface levels are minimized by shifting or removal of some vertices. This resulted in degradation of the watermark. In most cases more than 50% of the watermark was destroyed in this attack.

3D Model Name	Similarity Measure		
	Noise (% level)	HC Smoothing (# of steps)	Cropping (# of cropped vertices)
Cow	87.66% (10% level)	40.07% (2 steps)	67.57% (758 vertices)
Hypersheet	77.13% (10% level)	30.38% (2 steps)	62.41% (86 vertices)
Mushroom	85.06% (10% level)	42.42% (2 steps)	33.09% (105 vertices)
Mannequin	89.82% (10% level)	56.50% (2 steps)	92.91% (42 vertices)
Eagle	88.06% (10% level)	44.42% (2 steps)	65.10% (651 vertices)
Mechanical	82.62% (10% level)	52.50% (2 steps)	34.91% (50 vertices)

Table 6.2: Similarity Measure Results For Embedded and Extracted Watermarks after Various Attacks

Cropping Cropping refers to removal/chopping of a part or parts of a model. The amount of watermark destroyed depends upon the extent of cropping. This necessitates adequate presence of the watermark in various regions. The technique is robust against cropping; a high value of correlation is obtained between the original watermark and the extracted one. Since no watermark was inserted in the legs and part of the wings of the 'Eagle' model, no information was lost when the legs were cropped, thus giving a 95% correlation between the original watermark and extracted watermark.

Scaling, Translation, Rotation and Affine attacks This method is completely resistant to uniform scaling and affine attacks. The change in these parameters does not affect the relative orientation of the normal's at the vertices. Thus our algorithm is invariant to scaling and affine attacks and gives 100% correlation.

6.4 Analysis

Since the input feature vector to be fed to the ANN should of constant size, multiple neural networks have been adopted to accommodate vertices of different valences. The performance of the ANN is heavily dependant on the quality of the training data set. The initial training sets created by human operators did not yield satisfactory performance. The primary reason was that the training data quality was poor due to a number of reasons. There was human fatigue when labeling data and also there was ambiguity among the operators as to whether watermark should be inserted. To avoid data quality shortcomings, outputs of a high embedding capacity algorithm [66] are utilized for training the neural networks. Watermark invisibility is achieved through embedding the watermark with different scaling factors in vertices with higher embedding strengths. Experimental results show that the presented watermarking algorithm is of higher capacity than the formerly devised algorithm [65].

6.5 Summary

This chapter proposes a novel watermarking algorithm in which vertices are selected from the 3D model for watermarking by an ANN without causing perceptible distortion. Feedforward backpropagation and radial basis function neural networks were evaluated for selecting vertices for watermark insertion. A variety of 3D models with varying degrees of surface curvature were used to train and test the neural network. An array of neural networks was used for vertices with different valences to achieve higher watermark embedding capacity. The watermark extraction process is informed and needs the original watermark and 3D model. Experimental results evaluate the imperceptibility and robustness of the proposed algorithm and tested the algorithm against various attacks including noise addition, smoothing, and cropping. The proposed method uses an artificial neural network to select vertices based on the geometry of the 1-ring of vertices surrounding them, and gives good results (visual and analytical) for various types of surfaces (flat, curved, uneven, etc.) present on

3D models. Also, the system is robust against various possible attacks.

Chapter 7

Conclusions

7.1 Conclusions

The criteria of minimizing distortion or maximizing the amount of watermark was explored using computational intelligence (CI) techniques. Fuzzy logic is used for data fusion for spatial masking in wavelet domain. The scheme also adaptively varies the strength of the watermark signal with reference to the local geometry. The fuzzy input variables are computed for each wavelet coefficient in the model based on the local geometry. The output of the fuzzy system is a single value which gives a perceptual value for each corresponding wavelet coefficient. Thus, the fuzzy perceptual mask combines all these non-linear variables to build a model for spatial masking in 3D domain. We have demonstrated that this watermarking scheme is robust against attacks such as mesh simplification, addition of noise, model cropping, and at the same time achieve a high level of imperceptibility. Although the fuzzy based approach is not very suitable for fragile watermarking, it is the ideal choice for robust watermarking. 24 bits per vertex coordinate were inserted as a watermark. In 2008, Kai Wang *et al.* [92] claimed to be the first to insert multiple watermarks in 3D mesh. However, in 2007 [63], the proposed fuzzy logic in this dissertation was the first approach to insert multiple watermarks. Kai's work validates that wavelet transform has deep pockets of information for inserting watermarks.

In this dissertation, GAs have been used to find near-optimal solutions from infinite optimal solutions. In a GA based approach, by reducing the number of generations, optimality is traded for speed gains. In the case of optimizing a single criterion f, an optimum is either its maximum or minimum. A GA based approach works extremely well for fragile watermarking by inserting random numbers as watermark in all the vertices of a 3D mesh. Upto 120 bits per vertex coordinate can be inserted as watermark. This was never achieved before in 3D watermarking. The proposed algorithm works equally well for low and high resolutions models. The algorithm was equally effective in inserting information in flat and curved surfaces and was tested on CAM models as well. The GA based approach can also insert multiple watermarks as well for robust watermarking.

When compared to the Fuzzy Logic and Genetic Algorithms based approaches, performance of ann ANN is not exceptional. Lack of good quality training data is the primary reason. For the same reason, hybrid approaches such as neuro-fuzzy based approaches were not explored since such an approach too would require good quality data to learn the parameters of the network. However, an ANN based approach holds promise for other watermarking related applications such as stegnalysis. It can be concluded that unsupervised classifiers are a better choice when compared to supervised learning algorithms for 3D watermarking application.

7.2 Limitations

Each of the approaches explained in detail in previous chapters have limitations. For the Fuzzy based approach using wavelet transform, the limitation arises due to the use of semi-regular meshes required by the wavelet transform. That limitation is overcome by remeshing any irregular mesh into semi-regular mesh. However, thats adds another processing step. Another challenge with the fuzzy based approach is evaluating the quality of rules. Rough Sets [77] are proposed to evaluate if the the proposed 15 rules are sufficient and if the number of input and output membership classes are sufficient to differentiate classes.

For the GA based approach, computational resources can be a constraint since genetic algorithms are slow. However, genetic algorithms can be executed using parallel programming or implemented in parallel using hardware to speed the execution of the algorithm [24]. Also, in the current implementation only random information is inserted as watermark. In future work, images can be supported as a watermark by using binary representation of the chromosome. IEEE 754 floating point representation can be used as the binary representation for the vertex coordinates. Also, GA can be explored to insert information in the wavelet domain. Also, the current GA based implementation uses a single objective for optimization. Future work could use multiple objectives [79]. Multi-objective optimization algorithms have not been explored in the current scope of this dissertation. However, future work should explore multi-objective algorithms to minimize distortion and maximize the embedding capacity.

The biggest limitation for ANNs based approach is the lack of good quality training data. The performance of the ANN based approach is heavily dependent on the quality of the training data set. To overcome this limitation in the current ANN based implementation, 1-ring vertices from the original model are used as input and watermarked vertices using the GA approach are used as the output in the training dataset. Self-organizing maps [52] can also be evaluated for improved performance. Other features could also be evaluated.

7.3 Future Work

The goal of this dissertation was to build a framework using computational intelligence algorithms such as GAs, ANNs, and Fuzzy Logic to build a robust, fragile, and high density 3D watermarking system. There are other watermarking related applications such as fingerprinting and stegnalysis [22] which have not been explored in this dissertation using computational intelligence techniques. ANNs can be a very powerful tool for 3D stegnalysis since a neural network can be trained to learn which vertices could have been watermarked. Future work would explore and compare other computational intelligence and metaheuristics [88] techniques such as simulated annealing and tabu search. Computational Intelligence also includes Swarm intelligence [21] algorithms such as Particle Swarm Optimization [76] and Ant Colony Optimization [32]. Artificial Immune System [25] with algorithms such as Clonal Selection can also be used in this framework. There are several other directions in which the future work can be explored. Some of the extended capabilities in future could be exploring different applications some of which are listed below.

7.3.1 Third Generation Framework Capabilities

The capabilities of the third generation framework can be exploited to applications other than watermarking. Few of them are discussed below.

1. 3D Compression

   Watermarking and compression can be viewed as complementary problems. In watermarking, the goal is to add information without causing perceptible distortion, whereas in compression the objective is to remove information without causing perceptible distortion. Success of applying CI techniques can be extended to applying a similar approach to 3D compression [80].

2. New Watermarking Attacks

   A good watermarking technique can also be a considered as a good watermarking attack as well. For example, the approach described using genetic algorithms can also be used to insert multiple watermarks, or destroy the existing watermark as well.

The framework will result in a robust intelligent system that uses computational intelligence techniques to search optimal location for insertion or removal of information and optimize the amount of insertion or removal of the information to be added or removed. This system can also be applied to other systems such as surface reconstruction from point clouds and animations, images and video.

Bibliography

[1] 3D models, 3D modeling textures and plugins at turbosquid. http://www.turbosquid.com. Last Accessed Aug 17, 2009.

[2] blender.org - home. http://www.blender.org/. Last Accessed May, 2009.

[3] Buy 3D models, sell 3D models, free 3D models download at 3Dexport. http://www.3Dexport.com. Last Accessed May, 2009.

[4] Maya 3D animation, visual effects, and compositing software - autodesk. http://usa.autodesk.com/adsk/servlet/pc/index?id=13577897siteID=123112. Last Accessed May, 2009.

[5] Meshlab. http://meshlab.sourceforge.net/. Last Accessed Jul 17, 2011.

[6] Newtek lightwave. http://www.newtek.com/lightwave/. Last Accessed May, 2009.

[7] Princeton 3D model search engine. http://shape.cs.princeton.edu/search.html. Last Accessed May, 2009.

[8] Usenix security conference,2001. http://www.usenix.org/events/sec01/craver.pdf. Last Accessed May, 2011.

[9] P. Alface. *Perception and Re-Synchronization Issues for the Watermarking of 3D Shapes*. PhD thesis, Universite catholique de Louvain(UCL), Belgium, 2006.

[10] P. Alface and B.Macq. Blind watermarking of 3D meshes using robust feature points detection. In *Proceedings of IEEE International Conference on Image Processing (ICIP)*, volume 1, pages 693–696, September 2005.

[11] M. Ashourian and R. Enteshary. A new masking method for spatial domain watermarking of three-dimensional triangle meshes. In *TENCON 2003. Conference on Convergent Technologies for Asia-Pacific Region*, volume 1, pages 428 – 431 Vol.1, 2003.

[12] N. Aspert, D. Santa-Cruz, and T. Ebrahimi. Mesh: Measuring errors between surfaces using the hausdorff distance. In *Proceedings of the IEEE International Conference on Multimedia and Expo*, volume I, pages 705 – 708, 2002. `http://mesh.epfl.ch`.

[13] O. Benedens. Geometry-based watermarking of 3D models. *Computer Graphics and Applications, IEEE*, 19(1):46 –55, 1999.

[14] Oliver Benedens. Watermarking of 3D polygon based models with robustness against mesh simplification. In *Proceedings of SPIE Security and Watermarking of Multimedia Contents*, volume 3657, pages 329–340, 1999.

[15] J. Bennour and J.-C. Dugelay. Protection of 3D object visual representations. In *Multimedia and Expo, 2006 IEEE International Conference on*, pages 1113 –1116, 2006.

[16] Jihane Bennour and Jean-Luc Dugelay. Protection of 3D object through silhouette watermarking. In *ICASSP 2006, 31st International Conference on Acoustics, Speech, and Signal Processing, May 14-19, 2006, Toulouse, France*, 05 2006.

[17] Jihane Bennour and Jean-Luc Dugelay. Toward a 3D watermarking benchmark. In *MMSP 2007, IEEE International workshop on multimedia signal processing, October 1-3, 2007, Chania, Greece*, 10 2007.

[18] M. Berg, O. Cheong, and M. Kreveld. *Computational geometry: algorithms and applications*. Springer, 2008.

[19] Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Cl225;udio Silva, and Gabriel Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5:349–359, 1999.

[20] P.J. Besl and N.D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:239–256, 1992.

[21] C. Blum and D. Merkle. *Swarm intelligence: introduction and applications*. Natural computing series. Springer, 2008.

[22] R. Bohme. *Advanced Statistical Steganalysis*. Information Security and Cryptography Texts and Monographs. Springer, 2010.

[23] F. Boudjemai, P.B. Enberg, and J.-G. Postaire. Surface modeling by using self organizing maps of kohonen. In *Systems, Man and Cybernetics, 2003. IEEE International Conference on*, volume 3, pages 2418 – 2423 vol.3, Oct. 2003.

[24] E. Cantù-Paz. *Efficient and accurate parallel genetic algorithms*. Genetic algorithms and evolutionary computation. Kluwer Academic Publishers, 2000.

[25] L.N.D. Castro and J. Timmis. *Artificial immune systems: a new computational intelligence approach*. Springer, 2002.

[26] F. Cayre, P. Rondao-Alface, F. Schmitt, Benot Macq, and H. Matre. Application of spectral decomposition to compression and watermarking of 3d triangle mesh geometry. *Signal Processing: Image Communication*, 18(4):309 – 319, 2003.

[27] Hung-Kuang Chen and Yung-Hung Chen. Progressive watermarking on 3D meshes. In *Broadband Multimedia Systems and Broadcasting (BMSB), 2010 IEEE International Symposium on*, pages 1 –7, 2010.

[28] A. Cohen, Ingrid Daubechies, and J. C. Feauveau. Biorthogonal bases of compactly supported wavelets. *Communications on Pure and Applied Mathematics*, 45(5):485–560, 1992.

[29] K.J. Davis and K. Najarian. Maximizing strength of digital watermarks using neural networks. In *Neural Networks, 2001. Proceedings. IJCNN '01. International Joint Conference on*, 2001.

[30] A. de Medeiros Brito, A.D. Doria Neto, J. Dantas de Melo, and L.M. Garcia Goncalves. An adaptive learning approach for 3-d surface reconstruction from point clouds. *Neural Networks, IEEE Transactions on*, 19(6):1130 –1140, June 2008.

[31] Howard Demuth, Mark Beale, and Martin Hagan. *MATLAB Neural Network Toolbox 6: User's Guide.* The Mathworks, 2009.

[32] M. Dorigo and T. Stutzle. *Ant colony optimization.* Bradford Books. MIT Press, 2004.

[33] Jean-Luc Dugelay, Atilla Baskurt, and Mohamed Daoudi. *3D Object Processing: Compression, Indexing and Watermarking.* Wiley Publishing, 2008.

[34] A. Finkelstein E. Praun, H. Hoppe. Robust mesh watermarking. In *ACM SIGGRAPH Proceedings*, pages 69–76, 1999.

[35] Andries Engelbrecht. *Computational Intelligence - An introduction.* 2nd edition, Wiley Publishing, 2007.

[36] Alexander I. Galushkin. *Neural Network Theory.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

[37] Guangyong Gao and Guoping Jiang. Grayscale watermarking resistant to geometric attacks based on lifting wavelet transform and neural network. In *Intelligent Control and Automation (WCICA), 2010 8th World Congress on*, pages 1305 –1310, 2010.

[38] R.X. Gao and R. Yan. *Wavelets: Theory and Applications for Manufacturing.* Springer, 2010.

[39] Yang Gaobo, Sun Xingming, and Wang Xiaojing. A genetic algorithm based video watermarking in the dwt domain. In *Computational Intelligence and Security, 2006 International Conference on*, volume 2, pages 1209 –1212, 2006.

[40] David E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley Professional, 1 edition, January 1989.

[41] T. Harte and A.G. Bors. Watermarking graphical objects. In *Digital Signal Processing, 2002. DSP 2002. 2002 14th International Conference on*, 2002.

[42] H. Hoppe. Progressive meshes. In *SIGGRAPH 1996 Proceedings*, pages 99–108, 1996.

[43] H. Hoppe. Efficient implementation of progressive meshes. *Computers Graphics*, 22(1):27–36, 1998.

[44] Ming-Shing Hsieh. Image watermarking based on fuzzy inference filter. In *Machine Learning and Cybernetics, 2009 International Conference on*, volume 5, pages 3058 –3063, July 2009.

[45] R. Hu, P. Rondao-Alface, and B. Macq. Constrained optimisation of 3D polygonal mesh watermarking by quadratic programming. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 1501 –1504, 2009.

[46] Song Huang and Wei Zhang. Digital watermarking based on neural network and image features. In *Information and Computing Science, 2009. ICIC '09. Second International Conference on*, volume 2, pages 238 –240, May 2009.

[47] Song Huang, Wei Zhang, Wei Feng, and Huaqian Yang. Blind watermarking scheme based on neural network. In *Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on*, pages 5985 –5989, 2008.

[48] I.P. Ivrissimtzis, W.-K. Jeong, S. Lee, Y. Lee, , and H.-P. Seidel. Neural meshes: surface reconstruction with a learning algorithm. Research Report MPI-I-2004-4-005, Max-Planck-Institut für Informatik, Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany, October 2004.

[49] 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.

[50] Zachi Karni and Craig Gotsman. Spectral compression of mesh geometry. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '00, pages 279–286, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[51] M. Ketcham and S. Vongpraphip. Genetic algorithm audio watermarking using multiple image-based watermarks. In *Communications and Information Technologies, 2007. ISCIT '07. International Symposium on*, pages 1235 –1240, 2007.

[52] T. Kohonen. *Self-organizing maps.* Springer series in information sciences. Springer, 2001.

[53] Mao Li, Fan Yang-yu, Lv Guo-yun, and Wang Hui-qin. Adaptive color image watermarking algorithm based on fractal and neural networks. In *Image and Signal Processing, 2009. CISP '09. 2nd International Congress on*, pages 1 –5, 2009.

[54] Zhen Li, Wei-Min Zheng, and Zhe-Ming Lu. A robust geometry-based watermarking scheme for 3D meshes. In *Innovative Computing, Information and Control, 2006. ICICIC '06. First International Conference on*, 30 2006.

[55] Der-Chyuan Lou, Jiang-Lung Liu, and Ming-Chiang Hu. Adaptive digital watermarking using neural network technique. In *Security Technology, 2003. Proceedings. IEEE 37th Annual 2003 International Carnahan Conference on*, pages 325 – 332, 2003.

[56] Ming Luo and Adrian G. Bors. Principal component analysis of spectral coefficients for mesh watermarking. In *ICIP'08*, pages 441–444, 2008.

[57] Ming Luo, Kai Wang, Adrian Bors, and Guillaume Lavou. Local patch blind spectral watermarking method for 3D graphics. In *International Workshop on Digital Watermarking*, Lecture Notes in Computer Science, pages 211–226. Springer-Verlag, August 2009.

[58] Ameesh Makadia, Alexander IV Patterson, and Kostas Daniilidis. Fully automatic registration of 3D point clouds. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1:1297–1304, 2006.

[59] F. Martin McNeill and Ellen Thro. *Fuzzy logic: a practical approach.* Academic Press Professional, Inc., San Diego, CA, USA, 1994.

[60] Shi-chun Mei, Ren-hou Li, Hong-mei Dang, and Yun-kuan Wang. Decision of image watermarking strength based on artificial neural-networks. In *Neural Information Processing, 2002. ICONIP '02. Proceedings of the 9th International Conference on*, volume 5, pages 2430 – 2434 vol.5, 2002.

[61] Melanie Mitchell. *An Introduction to Genetic Algorithms.* MIT Press, Cambridge, MA, USA, 1998.

[62] Vishal Monga, Divyanshu Vats, and Brian L. Evans. Image authentication under geometric attacks via structure matching. In *ICME*, pages 229–232. IEEE, 2005.

[63] M. Motwani, N. Beke, A. Bhoite, P. Apte, and F.C. Harris. Adaptive fuzzy watermarking for 3D models. In *Computational Intelligence and Multimedia Applications, 2007. International Conference on*, volume 4, pages 49 –53, 2007.

[64] M.C. Motwani, R.C. Motwani, and F.C. Harris. Wavelet based fuzzy perceptual mask for images. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 4261 –4264, Nov. 2009.

[65] Mukesh Motwani, Sergiu Dascalu, Bobby Bryant, and Frederick Harris. 3D multimedia protection using artificial neural network. In *IEEE International Workshop on Digital Rights Management*, Las Vegas, USA, 2010.

[66] Mukesh Motwani, Balaji Sridharan, Rakhi Motwani, and Frederick Harris. Tamper proofing 3D models. In *IEEE International Conference on Signal Acquisition and Processing*, Bangalore, India, 2010.

[67] R. Motwani, S. Dascalu, and F. Harris Jr. A voice biometric watermark for 3D models. In *Proceedings of IEEE International Conference on Computer Engineering and Technology*, April 2010.

[68] R. Motwani and F. Harris Jr. Robust 3D watermarking using vertex smoothness measure. In *Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition*, July 2009.

[69] R.C. Motwani, F.C. Harris, and K.E. Bekris. A proposed digital rights management system for 3D graphics using biometric watermarks. In *Consumer Communications and Networking Conference (CCNC), 2010 7th IEEE*, pages 1 –6, 2010.

[70] R.C. Motwani, M.C. Motwani, B.D. Bryant, F.C. Harris, and A.S. Agarwal. Watermark embedder optimization for 3D mesh objects using classification based approach. In *Signal Acquisition and Processing, 2010. ICSAP '10. International Conference on*, pages 125 –129, 2010.

[71] C.L. Mumford and L.C. Jain. *Computational Intelligence: Collaboration, Fusion and Emergence.* Intelligent Systems Reference Library. Springer-Verlag Berlin Heidelberg, 2009.

[72] A.R. Naghsh Nilchi and A. Taheri. A new robust digital image watermarking technique based on the discrete cosine transform and neural network. In *Biometrics and Security Technologies, 2008. ISBAST 2008. International Symposium on*, pages 1 –7, 2008.

[73] R. Ohbuchi, H. Masuda, and M. Aono. Watermarking three-dimensional polygonal models through geometric and topological modifications. *Selected Areas in Communications, IEEE Journal on*, 16(4):551 –560, May 1998.

[74] R. Ohbuchi, A. Mukaiyama, and S. Takahashi. Watermarking a 3D shape model defined as a point set. In *Cyberworlds, 2004 International Conference on*, pages 392 – 399, 2004.

[75] Ryutarou Ohbuchi, Akio Mukaiyama, and Shigeo Takahashi. A frequency-domain approach to watermarking 3d shapes. *Computer Graphics Forum*, 21:373–382, 2002.

[76] K.E. Parsopoulos and M.N. Vrahatis. *Particle Swarm Optimization and Intelligence: Advances and Applications.* Premier Reference Source. Information Science Reference, 2010.

[77] L. Polkowski. *Rough sets: mathematical foundations.* Advances in soft computing. Physica-Verlag, 2002.

[78] Emil Praun and Hugues Hoppe. Spherical parametrization and remeshing. *ACM Transactions on Graphics*, 22:340–349, 2003.

[79] G.P. Rangaiah. *Multi-objective optimization: techniques and applications in chemical engineering.* Advances in process systems engineering. World Scientific, 2009.

[80] Flemming Friche Rodler. Wavelet based 3d compression for very large volume data supporting fast random access. Technical report, 1999.

[81] T.J. Ross. *Fuzzy Logic with Engineering Applications.* John Wiley & Sons, 2010.

[82] William Rucklidge. *Efficient visual recognition using the Hausdorff distance.* Lecture notes in computer science. Springer, 1996.

[83] Gnter Rudolph and Alexandru Agapie. Convergence properties of some multi-objective evolutionary algorithms. In *In Congress on Evolutionary Computation (CEC 2000*, pages 1010–1016. IEEE Press, 2000.

[84] N. Sakr, Jiying Zhao, and V. Groza. A dynamic fuzzy logic approach to adaptive hvs-based watermarking. In *Haptic Audio Visual Environments and their Applications, 2005. IEEE International Workshop on*, page 6 pp., Oct. 2005.

[85] Greg Sandoval. Fbi makes arrest in wolverine uploading case. http://news.cnet.com/8301-31001_3-10416372-261.html. Last Accessed July 17, 2011.

[86] S. Sumathi and S. Paneerselvam. *Computational Intelligence Paradigms: Theory & Applications Using MATLAB.* Taylor and Francis, 2009.

[87] Wim Sweldens. The Lifting Scheme: A Construction of Second Generation Wavelets. *SIAM Journal on Mathematical Analysis*, 29(2):511–546, 1998.

[88] E.G. Talbi. *Metaheuristics: from design to implementation.* Wiley Series on Parallel and Distributed Computing. John Wiley & Sons, 2009.

[89] G. Turk, Huong Quynh Dinh, J.F. O'Brien, and G. Yngve. Implicit surfaces that interpolate. In *Shape Modeling and Applications, SMI 2001 International Conference on.*, pages 62 –71, may 2001.

[90] S. Valette and P. Prost. Wavelet-based multiresolution analysis of irregular surface meshes. *Visualization and Computer Graphics, IEEE Transactions on*, 10(2):113 –122, 2004.

[91] J. Vollmer, R. Mencl, and H. Mller. Improved laplacian smoothing of noisy surface meshes. In *Computer Graphics Forum*, pages 131–138, 1999.

[92] Kai Wang, G. Lavoue, F. Denis, and A. Baskurt. Hierarchical blind watermarking of 3D triangular meshes. In *Multimedia and Expo, 2007 IEEE International Conference on*, pages 1235 –1238, 2007.

[93] Kai Wang, G. Lavoue, F. Denis, and A. Baskurt. A comprehensive survey on three-dimensional mesh watermarking. *Multimedia, IEEE Transactions on*, 10(8):1513 –1527, 2008.

[94] Shuhong Wang, Liang Yuan, Jie Qiu, and Semyung Wang. Feature-based fuzzy control adaptive finite-element mesh generation for electromagnetic fields. *Magnetics, IEEE Transactions on*, 41(5):1688 – 1691, May 2005.

[95] Kai Wanga, Guillaume Lavou?, Florence Denis, Atilla Baskurt, and Xiyan He. A benchmark for 3D mesh watermarking. In *Shape Modeling International*, pages 231–235, 2010.

[96] Zhicheng Wei, Jufeng Dai, and Jianxiong Li. Genetic watermarking based on dct domain techniques. In *Electrical and Computer Engineering, 2006. CCECE '06. Canadian Conference on*, pages 2365 –2368, May 2006.

[97] Bin Yang, Xiao-Qiang Li, and Wei Li. A robust mesh watermarking scheme based on pca. In *Image and Graphics, 2009. ICIG '09. Fifth International Conference on*, pages 778 –783, 2009.

[98] Kangkang Yin, Zhigeng Pan, Jiaoying Shi, and David Zhang. Robust mesh watermarking based on multiresolution processing. *Computers Graphics*, 25(3):409 – 420, 2001.

[99] Wang Ying, Zheng Xue-feng, and Liu Hai-Yan. Robust 3D watermarking based on geometry image. In *Wireless Communications, Networking and Mobile Computing, 2008. WiCOM '08. 4th International Conference on*, pages 1 –4, 2008.

[100] L. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 100:9–34, 1999.

[101] Lotfi A. Zadeh. Fuzzy sets. *Information and Control*, pages 338–353, 1965.

[102] Jun Zhang, Nengchao Wang, and Feng Xiong. Hiding a logo watermark into the multiwavelet domain using neural networks. In *Tools with Artificial Intelligence, 2002. (ICTAI 2002). Proceedings. 14th IEEE International Conference on*, 2002.

[103] Xinhong Zhang and Fan Zhang. A blind watermarking algorithm based on neural network. In *Neural Networks and Brain, 2005. ICNN B '05. International Conference on*, volume 2, pages 1073 –1076, 2005.

[104] Ning Zhong, Zunwen He, Jingming Kuang, and Zhihai Zhuo. An optimal wavelet-based image watermarking via genetic algorithm. In *Natural Computation, 2007. ICNC 2007. Third International Conference on*, volume 3, pages 103 –107, 2007.