3D Data Security by Means of Data Hiding and Encryption for the Fashion Industry

Bianca Jansen van Rensburg

THÈSE POUR OBTENIR LE GRADE DE DOCTEUR DE L'UNIVERSITE DE MONTPELLIER

En Informatique

École doctorale : Information, Structures, Systèmes

Unité de recherche LIRMM - Laboratoire d'Informatique, de Robotique et de Micro-électronique de Montpellier

Sécurisation des données 3D par insertion de données cachées et par chiffrement pour l'industrie de la mode

Présentée par Bianca JANSEN VAN RENSBURG

Le 9 novembre 2023

Sous la direction de William PUECH

Devant le jury composé de

Florent DUPONT, PU, Univ. Lyon / LIRIS, Lyon	Rapporteur
Kai WANG, CR CNRS, HDR, GIPSA-Lab, Grenoble	Rapporteur
Caroline FONTAINE, DR CNRS, ENS Paris-Saclay / LSV, Paris	Examinatrice
Pauline PUTEAUX, CR CNRS, CRIStAL, Lille	Examinatrice
William PUECH, PU, Univ. Montpellier / LIRMM, Montpellier	Directeur de thèse
Adrian BORS, Associate Professor, University of York, York, UK	Invité
Jean-Pierre PEDEBOY, PDG, STRATEGIES, Rungis	Invité

UNIVERSITÉ DE MONTPELLIER

REMERCIEMENTS

Tout d'abord, je tiens à remercier les membres du jury, Florent Dupont, Kai Wang, Caroline Fontaine, Pauline Puteaux et Adrian Bors pour l'intérêt qu'ils ont porté à mon travail. Je remercie en particulier Florent Dupont et Kai Wang d'avoir accepté d'être rapporteurs et d'avoir lu attentivement mon travail.

Je remercie tout particulièrement mon directeur de thèse, William Puech. Tout d'abord, je voudrais te remercier d'avoir accepté de m'encadrer en thèse. C'était un privilège de collaborer avec toi au cours de ces trois dernières années. Tu m'as toujours poussé à grandir, à travers nos discussions scientifiques, à travers les voyages, et en me donnant la confiance nécessaire pour affronter mes peurs. Grâce à toi, j'ai davantage confiance en moi et en mes capacités. C'était un privilège de collaborer avec toi non seulement au niveau scientifique, mais aussi au niveau humain. Je tiens donc à te remercier pour ta gentillesse, ton soutien et ton écoute tout au long de ma thèse. Ce n'est pas tout le monde qui aurait traversé Paris avec moi jusqu'à 1h du matin car j'ai mal aux dents.

Je tiens à remercier Stratégies de m'avoir donné cette opportunité en finançant ma thèse dans le cadre d'une collaboration CIFRE avec le LIRMM. En particulier, je voudrais remercier Jean-Pierre Pedeboy pour avoir cru en ce projet. Nos discussions ont toujours conduit à l'exploration de nombreuses nouvelles pistes scientifiques.

Je remercie l'équipe d'ICAR. J'ai rejoint l'équipe pour la première fois en 2019, à l'occasion d'un stage de M1. C'est la bonne ambiance, les conversations intéressantes et les gens ici qui ont contribué à alimenter ma passion pour la recherche.

Je remercie tout particulièrement mon fiancé, Benjamin Detourbet, pour avoir toujours été là pour moi au cours de ces 8 dernières années. Avec ton amour et ton calme, tu as toujours été ma principale source de soutien et je n'aurais pas pu faire tout cela sans toi. Je te remercie pour ton amour et ta volonté de me suivre où que j'aille.

Je remercie mes parents, Belinda et John Jansen van Rensburg, pour votre amour et votre soutien constants tout au long de ma vie et pour tous les sacrifices que vous avez faits pour moi. Vous m'avez toujours dit, depuis que je suis toute petite, que je pouvais accomplir tout ce que je voulais.

Je tiens également à remercier mes futurs beaux-parents, Isabelle et Francis Detourbet. Il y a 8 ans, vous m'avez accueillie dans la famille et depuis, je me suis toujours sentie chez moi avec vous. Je remercie tous les enfants de la famille, Lahja, Arja, Erwan et Liam pour leur bonheur et leur joie de vivre contagieuse.

Je tiens à remercier mon ange gardienne de thèse Pauline Puteaux et mon amie Laura Bertojo pour vos conseils, votre soutien et votre amitié.

Je tiens à remercier mes chiens, Shallan, Theoden et Noctis, ainsi qu'Oliver, Nash, Chaussettes et Zarko. Votre amour inconditionnel et votre empathie ont été un soutien essentiel. Vous avez bien montré pourquoi le chien est le meilleur ami de l'homme.

Enfin, merci à mon collègue de bureau, Nicolas Dibot, pour m'avoir toujours fait rire et à Erwan Reinders, que j'ai encadré en stage. C'était un plaisir de travailler avec vous.

Abstract

Over the last few decades, 3D objects have become an essential part of everyday life in both private and professional contexts. These 3D objects are often stored on the cloud and transferred over networks many times during their existence, where they are susceptible to malicious attacks. Therefore, 3D object security, such as encryption or data hiding, is essential. Encryption is used to protect the visual confidentiality of the 3D object's content. Selective encryption schemes can also be used, where part of a component, such as a part of each vertex, is encrypted. Data hiding is generally used to protect the copyright or the authenticity of the 3D object. However, when a 3D object is encrypted, a third party such as a server may need to embed data in the confidential 3D object. In this case, data hiding in the encrypted domain is performed. In many applications, 3D objects often consist of millions of vertices, and so storing and sharing them online is expensive, time consuming and not environmentally friendly. Consequently, 3D object compression is essential. In this work, we present three contributions in different research areas. First, we present our work on a new method to obtain a watermarked 3D object from high-capacity data hiding in the encrypted domain. Based on the homomorphic properties of the Paillier cryptosystem, our proposed method allows us to embed several secret messages in the encrypted domain with a high-capacity. These messages can be extracted in the plaintext domain after the 3D object decryption. To the best of our knowledge, we are the first to propose a data hiding method in the encrypted domain where the high-capacity watermark is conserved in the plaintext domain after the 3D object is decrypted. Both the encryption and the data hiding in the encrypted domain are format compliant and without size expansion, despite the use of the Paillier cryptosystem. Then, we present our work on an evaluation metric for the visual security level of selectively encrypted 3D objects. Based on a dataset composed of evaluated selectively encrypted 3D objects, we propose a model to determine the security parameters according to a desired security level. We detail our proposed *3DVS* score which serves to measure the visual security level of selectively encrypted 3D objects. We also present a method which allows us to hierarchically decrypt an encrypted 3D object according to a generated ring of keys. This ring consists of a set of keys that allow a stronger or weaker decryption of the encrypted 3D object. Each hierarchically decrypted 3D object has a different visual security level, where the 3D object is more or less visually accessible. Our method is essential when it comes to preventing trade secrets from being leaked from within a company or by exterior attackers. It is also ecologically friendly and more secure than traditional selective encryption methods. Finally, we present our work on joint security and compression methods based on Google's 3D object compression method Draco, where we integrate a security step in Draco, which is becoming the new industry standard. These security steps are encryption, selective encryption and watermarking.

Keywords

Multimedia security, Multimedia data hiding, 3D security, Encryption, Draco compression

RÉSUMÉ DE LA THÈSE

Au cours des dernières décennies, les objets 3D sont devenus des éléments essentiels de la vie quotidienne, tant dans le contexte privé que professionnel. Ces objets 3D sont souvent stockés sur le cloud et transférés sur des réseaux plusieurs fois au cours de leur existence, où ils sont susceptibles de faire l'objet d'attaques malveillantes. Par conséquent, des méthodes de sécurisation des objets 3D, comme le chiffrement ou l'insertion des données cachées, sont essentielles. Le chiffrement est utilisé pour protéger la confidentialité visuelle du contenu d'un objet 3D. Il est également possible d'utiliser des schémas de chiffrement sélectif, dans lesquels seulement une partie de l'objet 3D est chiffrée. L'insertion des données cachées est généralement utilisée pour protéger les droits d'auteur ou l'authenticité des objets 3D. Toutefois, lorsqu'un objet 3D est chiffré, un tiers, tel qu'un serveur, peut avoir besoin d'intégrer des données dans l'objet 3D confidentiel. Dans ce cas, les données sont cachées dans le domaine chiffré. Les objets 3D sont souvent constitués de millions de sommets, de sorte que le stockage et le partage en ligne sont coûteux. Par conséquent, la compression des objets 3D est essentielle. Dans ce travail, nous présentons trois contributions dans différents domaines de recherche. Premièrement, nous présentons notre travail sur une nouvelle méthode permettant d'obtenir un objet 3D marqué à partir d'une insertion de données cachées de haute capacité dans le domaine chiffré. Basée sur des propriétés homomorphiques du cryptosystème de Paillier, notre méthode permet d'insérer plusieurs messages secrets dans le domaine chiffré avec une haute capacité. Ces messages peuvent être extraits dans le domaine en clair après le déchiffrement de l'objet 3D. À notre connaissance, nous sommes les premiers à proposer une méthode d'insertion de données cachées dans le domaine chiffré où les données cachées de haute capacité sont conservées dans le domaine en clair après le déchiffrement de l'objet 3D. Le chiffrement et l'insertion de données cachées dans le domaine chiffré sont conformes au format et sans expansion de taille, malgré l'utilisation du cryptosystème de Paillier. Nous présentons ensuite notre travail sur une mesure d'évaluation du niveau de sécurité visuelle des objets 3D chiffrés sélectivement. Basé sur une nouvelle base de données composée d'objets 3D chiffrés sélectivement et évalués, nous proposons un modèle pour déterminer les paramètres de sécurité en fonction du niveau de sécurité souhaité. Nous détaillons notre score *3DVS* qui sert à mesurer le niveau de sécurité visuelle des objets 3D chiffrés sélectivement. Nous présentons également, à notre connaissance, la première méthode permettant de déchiffrer hiérarchiquement un objet 3D chiffré en fonction d'un trousseau de clés généré. Ce trousseau se compose d'un ensemble de clés qui permettent un déchiffrement plus ou moins fort de l'objet 3D chiffré. Chaque objet 3D déchiffré hiérarchiquement a un niveau de sécurité visuelle différent, où l'objet 3D est plus ou moins accessible visuellement. Notre méthode est essentielle lorsqu'il s'agit d'empêcher des fuites des secrets commerciales au sein d'une entreprise ou par des at-

taquants extérieurs. Elle est également écologique et plus sécurisée que les méthodes traditionnelles de chiffrement sélectif. Enfin, nous présentons notre travail sur des méthodes conjointes de sécurité et de compression basées sur la méthode de compression d'objets 3D de Google, Draco, dans laquelle nous intégrons une étape de sécurité dans Draco, qui est en train de devenir la nouvelle norme de l'industrie. Ces étapes de sécurité sont le chiffrement, le chiffrement sélectif et le tatouage.

Mots-clés

Sécurité multimédia, Insertion de données cachées, Sécurité 3D, Chiffrement, Compression Draco

Table of Contents

INTRODUCTION

Context and applications

In recent years, multimedia accounts for the majority of data shared online. With the rise of social media and the emergence of the meta-verse, multimedia sharing has become an integral part of our everyday lives in both personal and professional contexts. However, once the multimedia is online, it becomes inherently vulnerable to malicious attacks such as copying or theft. Consequently, there is a growing need for multimedia security. In the context of this thesis, we focus on the security of 3D objects by means of encryption and data hiding.

In certain industries, such as the fashion or entertainment industries, these 3D objects are considered as important assets, and their theft can result in a great financial loss, or a leak of trade secrets. In some cases, such as in healthcare, these 3D objects represent patient information, and their theft could result in the sharing of private medical information.

Generally, these 3D objects are of a high quality are therefore very large, often consisting of millions of vertices. Consequently, their processing, storage and sharing consume many resources. They therefore need to be compressed as well as secured. This can be resolved using joint security and compression methods.

In an industrial context, 3D objects often represent the designs for new products. First, the designer manually creates the 3D object, after which the 3D object is sent along the digital production line. This production line contains many stages, where the 3D object is stored and shared online, and then downloaded and processed by many different parties. Due to the nature of the 3D object, it has a high financial value. Therefore, many security problems can arise during its lifespan in the production line. This 3D object therefore needs to be secured against malicious attackers. These attackers include competing companies, as well as pirates seeking to create counterfeit products. The 3D object can be obtained by unauthorized parties either through illegally attacking the 3D object when it is stored online, or through internal leaks.

This CIFRE thesis is the result collaboration between the company STRATEGIES, Rungis, France and the Laboratoire d'Informatique, de Robotique et de Micro-électronique de Montpellier (LIRMM), Université de Montpellier, Centre National de la Recherche Scientifique (CNRS). STRATEGIES create the software suite of design tools *Romans CAD Software* for luxury brands in the fashion industry, and in particular, the shoe industry. Fig. 1 shows a screenshot of their design software *RCS 3D*, taken from their

website[1], where a shoe is in the process of being designed. STRATEGIES also propose an online viewing platform called *Showcase*, where brands can publish their 3D models online (Fig. 2a), along with all variants of the 3D model (Fig. 2b).

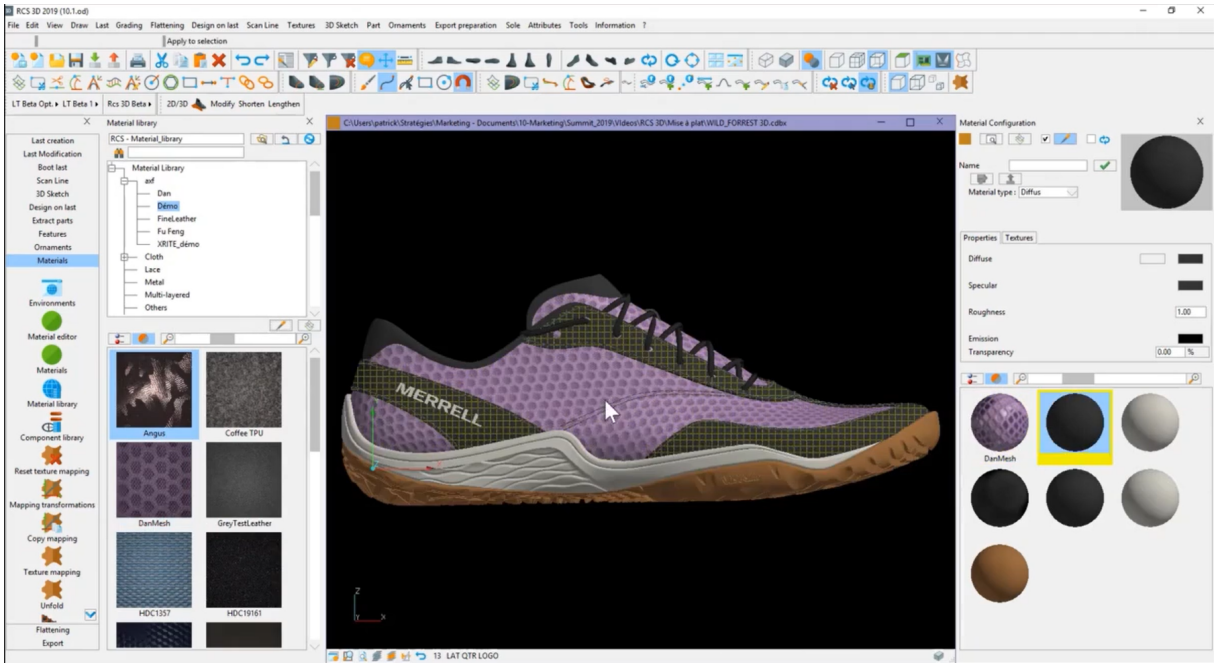
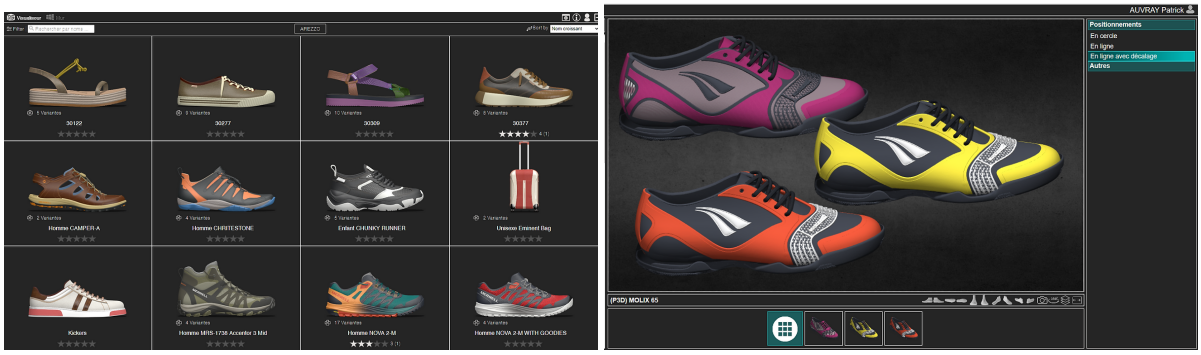


Figure 1: *RCS 3D software.*



(a) **(b)**

Figure 2: *The online viewing platform Showcase, where a) Different shoes for a brand, and b) The variants of the same shoe.*

There are different ways to secure the 3D objects, notably by encryption or by data hiding. Encryption is used to secure the content of the 3D objects, while data hiding is used to ensure the data rights management, authenticate the data, or identify the source of internal leaks. In some cases, these methods can be combined with other techniques in order to create joint security methods. For example, these joint security methods can be joint encryption and data hiding (data hiding in the encrypted domain), joint encryption and compression (crypto-compression), or joint data hiding and compression.

[1] www.romans-cad.com

Structure

The structure of this thesis is divided into two main parts. In the first part, we present a state of the art and in the second, we describe our three main contributions.

In Part I, we present a current state of the art of multimedia security. In Chapter 1, we present a state of the art of multimedia encryption. We present the foundations of modern cryptography, we describe the structure of an image and a 3D object, and we present different types of encryption for images and 3D objects. Then, in Chapter 2, we present a state of the art of multimedia data hiding. We introduce the fundamentals of data hiding, and then describe the foundations and applications of the three main categories of data hiding, which are steganography, watermarking and high capacity data hiding. Finally, in Chapter 3, we present a state of the art of joint multimedia encoding. We present the compressed domain, which includes image compression, videos, and 3D object compression. We then present joint compression and security, as well as data hiding in the encrypted domain.

In Part II, we detail our three main contributions. In Chapter 4, we describe two variants of a reversible data hiding in the encrypted domain method for 3D objects, which is based on the Paillier homomorphic cryptosystem. Then, in Chapter 5 we present two contributions relating to the visual security level of 3D objects. We present a method for a security parameter estimation as well as a subjective metric to determine the visual security level, before describing a new encryption method which allows for a hierarchical decryption of 3D objects. Then, in Chapter 6, we detail three contributions for joint security and compression for 3D objects based on Draco. These methods are a crypto-compression method, a selective crypto-compression method, and a joint watermarking and compression method.

Finally, we conclude this thesis and present perspectives for future work concerning 3D object security.

Part I

State of the Art

CHAPTER 1

MULTIMEDIA ENCRYPTION

Chapter Contents

1.1 Introduction

Over the last few years, multimedia has come to play an essential role in everyday life, in both private and professional contexts. Today, the manufacturing, entertainment and healthcare industries, among many others, are now reliant on online multimedia storage and sharing. This multimedia is often regarded as important assets, as it represents trade secrets or personal information. For example, in the manufacturing industry, 3D objects often represent design templates whose theft could result in the loss of trade secrets, and ultimately counterfeit products. In healthcare, multimedia often represents a patient's private medical information. The storage and sharing of multimedia in a private context has also largely increased, due the growing popularity of social media and media sharing platforms, as well as the meta-verse. Consequently, it is essential that multimedia stored and shared online is secured from unauthorised access. The first category of multimedia security methods is encryption, which assures the visual confidentiality of the multimedia so that its content is unrecognizable.

In this chapter, we present a current state of the art of multimedia encryption techniques. First, in Section 1.2, we introduce the foundations of modern cryptography, where we describe a brief history of encryption and present Kerckhoffs' principle, as well as symmetric and asymmetric cryptography. Then, in Section 1.3, we present images, the main encryption methods, as well as visual quality assessment and visual security assessment applied to images. Finally, in Section 1.4, we first present the representation of 3D objects and their fundamentals such as their encryption and evaluation. We also present a specific type of 3D object encryption, which is selective encryption, and introduce the notion of visual security levels constructed for 3D objects.

1.2 Modern Cryptography

Cryptography is described by Katz *et al.* in their book "Introduction to modern cryptography" as *the scientific study of techniques for securing digital information, transactions, and distributed computations* (1). This definition applies to modern times, where cryptography is mainly applied to digital information.

Historically, this has not always been the case, and cryptography is considered as *the art of writing or solving codes*, as defined by the Oxford dictionary (2). Securing sensitive or secret information from unauthorized access is indeed a practice that dates back to 600 BC, where the Spartans a scytale to transfer secret messages during battle. The scytale consists of a leather strip with engraved letters wrapped around a wooden rod. If the rod has the correct diameter, the letters on the leather strap line up to reveal a hidden message.

The Romans also used cryptography, since around 60 BC, Julius Caesar invented the Caesar cipher. This cipher aligns an alphabet with the same alphabet shifted by a certain number of places. The message is encrypted by substituting each letter by the corresponding shifted letter. For example, if there is a shift of 2, then "A" is replaced with "C", "B" with "D", and "C" with "E", etc.

Until recently, encryption methods were based on the secrecy of the encryption method. In modern times, methods such as these can no longer be considered secure. Once the encryption method is known, not only does the cryptosystem become obsolete and therefore have to be discarded, but all data previously encrypted with this method can be now decrypted by anyone with knowledge of the cryptosystem and is therefore compromised.

1.2.1 Kerckhoffs' Principle

To combat this problem, in 1883, Kerckhoffs presented *Kerchoffs' principal*. This introduces the notion that in order for a cryptosystem to be considered secure, it must remain secure if all information about the cryptosystem is known and only the key remains secret (3). Kerckhoffs' principle states:

- The security of a cryptosystem must rely on the key and not the encryption method.
- Decrypting an encrypted message without a key must be materially, if not mathematically, impossible.
- Deducing the key with the use of the encrypted text and the plaintext must be impossible.

Later 1949, Shannon reformulated Kerckchoff's principle by stating that in order for a cryptosystem to be considered secure, we must assume that *the enemy knows the system being used* (4). This is known as Shannon's maximum. In this same article, Shannon stated that a cryptosystem must introduce both diffusion and substitution. Diffusion refers to the dissipation of the statistical structure and eliminating redundancy in the ciphertext. Consequently, if a single bit in the plaintext (resp. ciphertext) is modified, multiple bits in the ciphertext (resp. decrypted plaintext) must change. In the case of images, this means that the encrypted pixels must have no correlation with one another. Diffusion can be introduced with a permutation step. Confusion ensures that the relationship between the ciphertext and the key is complex, and therefore the key cannot be deduced from the ciphertext. In other words, each bit of the ciphertext must depend on multiple bits of the key. Confusion can be introduced with a substitution step.

Modern cryptography is based upon these concepts. Therefore, in modern cryptosystems, the encryption method is always known and the security relies on the key. There are two different types of key schemes which are symmetric cryptography and asymmetric cryptography.

1.2.2 Symmetric Cryptography

Symmetric cryptography, which is also known as secret key cryptography, describes cryptosystems where a single secret key is used to encrypt and decrypt a plaintext. This

secret key is therefore shared between the data sender and the data receiver. Fig. 1.1 illustrates an overview of a symmetric encryption scheme.

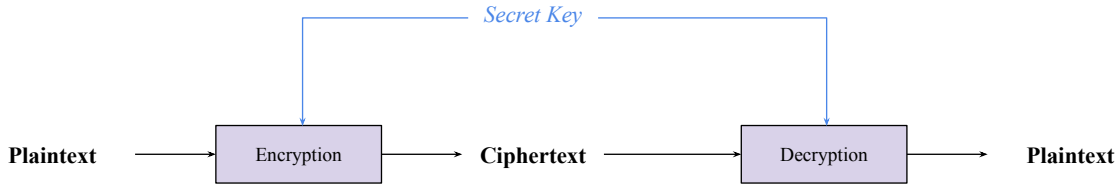


Figure 1.1: *Overview of a symmetric encryption scheme.*

There are two categories of ciphers used in symmetric encryption scheme. The first category of ciphers is a block cipher, where the plaintext is grouped into blocks of a fixed size. These blocks, which often have a size $size_b$ of 64, 128 or 256 bits, are then encrypted consecutively, where each encrypted block has a size of $size_b$. In the case where the plaintext is not a multiple of $size_b$ and the last block is consequently smaller than the required size $size_b$, then this smaller block is padded with zeros in order to achieve the required size. In block ciphers, each block is encrypted separately, and therefore two identical blocks will have the same ciphertext if they are encrypted with the same key.

The second category is a stream cipher, first presented by Vernam in 1926, where the plaintext is encrypted element by element (5). We note that these elements can be either bits or blocks. A pseudo-random binary sequence bit_seq is generated with the use of a pseudo-random number generator (PRNG) initialized with a secret key K:

$$bit\_seq = PRNG(K). \tag{1.1}$$

A plaintext m is then encrypted with an exclusive-or (XOR) between m and the pseudo-random binary sequence bit_seq:

$$c = m \oplus bit\_seq, \tag{1.2}$$

where c is the resulting ciphertext and $\oplus$ is the binary operator XOR.

The manner in which the different blocks or elements interact and thus influence each other during the encryption and decryption process depends on the mode of operation. There are five different block cipher modes of operation:

- **Electronic Code Book (ECB)**: In this mode, each block B_i is encrypted in parallel:

  $$B'_i = \mathcal{E}_\mathcal{K}(B_i), \tag{1.3}$$

  where B'_i is the resulting ciphertext and $\mathcal{E}_\mathcal{K}(\cdot)$ is the encryption method based on the secret key K.

  This mode is simple and very fast as parallel encryption is possible, but is it also the weakest mode in terms of security, as a direct relationship between the plaintext and the ciphertext is established. This implies that if there are multiple identical blocks, then information can be deduced about the plaintext once encrypted. This mode and the CTR mode are the only block ciphers.

- **Cipher Block Chaining (CBC)**: In this mode, an XOR is performed between the current block to encrypt B_i and the previous encrypted block B'_{i-1} before the encryption takes place:

$$B'_i = \mathcal{E}_\mathcal{K}(B_i \oplus B'_{i-1}), \tag{1.4}$$

  where B'_i is the resulting ciphertext and $\mathcal{E}_\mathcal{K}(\cdot)$ is the encryption method based on the secret key K.

  This mode improves the security of ECB, as there is no longer a relationship between the plaintext and the ciphertext. We note that this mode requires an initialization vector IV, where $B_0 = IV$.

- **Cipher Feedback (CFB)**: In this mode, an XOR operation is performed between the current block B_i and the encryption of the previous block B'_{i-1}:

$$B'_i = B_i \oplus \mathcal{E}_\mathcal{K}(B'_{i-1}), \tag{1.5}$$

  where B'_i is the resulting ciphertext and $\mathcal{E}_\mathcal{K}(\cdot)$ is the encryption method based on the secret key K. We note that like the CBC mode, this mode requires an initialization vector IV, where $B_0 = IV$.

- **Output Feedback (OFB)**: In this mode, an XOR operation is performed between the current block B_i and a sequence S_i.

$$B'_i = B_i \oplus S_i, \tag{1.6}$$

  where B'_i is the resulting ciphertext and S_i is given by:

$$S_i = \mathcal{E}_\mathcal{K}(S_{i-1}), \tag{1.7}$$

  where $\mathcal{E}_\mathcal{K}(\cdot)$ is the encryption method based on the secret key K. This mode also requires an initialization vector IV, where $S_0 = IV$.

- **Counter (CTR)**: In final mode, an XOR operation is performed between the current block B_i and the encryption of a pseudo-random sequence S_i:

$$B'_i = B_i \oplus \mathcal{E}_\mathcal{K}(S_i), \tag{1.8}$$

  where $\mathcal{E}_\mathcal{K}(\cdot)$ is the encryption method based on the secret key K. The sequence S_i is generated with the use of a PRNG initialized by a counter c_i, where:

$$c_i = c_{i-1} + 1, \tag{1.9}$$

  and c_0 is a random seed, and therefore S_i is given by:

$$S_i = PRNG(c_i). \tag{1.10}$$

The first standard for block ciphers is the Data Encryption Standard (DES), which was approved in 1976 (6). However, DES requires a 56 bit key and is considered outdated by modern cryptography standards, as the key can be found with brute force using modern computers. The Advanced Encryption Standard (AES) algorithm, proposed by Daemen and Rijmen (7) in 1999, replaced DES and Triple DES, and is considered to be the standard for modern block cipher encryption.

The AES encryption function consists of a set of processing operations which are performed iteratively on a block of $4 \times 4 = 16$ bytes (or 128 bits). The number of iterations, called rounds, depends on the size of the key. A 128 bit key has 10 rounds, and a 256 bit key has 14 rounds. For each round, a round key is generated according to the original 256 bit key by using a key expansion. Each processing operation depends on its corresponding round key.

First, the operation $AddRoundKey$ is applied to the current block. This operation consists of applying a round key by means of an XOR binary operation. Then, for all rounds except the last, there are four operations which are applied to the block, which are $SubBytes, ShiftRows, MixColumns$ and $AddRoundKey$. In the final round, $MixColumns$ is not applied. The first operation, $SubBytes$ consists of substituting each byte of the 16 byte matrix according to a non-linear look-up table, the Rijndael S-box. This step introduces confusion. In the $ShiftRows$ operation, each byte in each row is cyclically shifted by a certain offset $shift = i_r$, where i_r is the row's index and $i_r \in [0, 3]$. This operation introduces diffusion. The operation $MixColumns$ applies an invertible linear transformation to each column with the use of transformation matrix that combines the 4 bytes of each column.

1.2.3 Asymmetric Cryptography

Asymmetric cryptography describes cryptosystems where two keys, known as a key pair, are required. If one key is required to perform an encryption, then the other is required to perform the decryption. The first key, which is generally used for the encryption step, is a known public key, and the other, which is generally used for the decryption step, is an unknown private key. An overview of an asymmetric scheme is illustrated in Fig. 1.2. The two keys of the key pair are mathematically related, where the public key is calculated from the private key by means of a one-way function. It is therefore impossible to calculate the private key from the public key in a polynomial time. Generally, it is the private key holder who generates both keys, as the public key can then be distributed without the need for secrecy. While asymmetric cryptosystems solve the symmetric cryptosystems' problem related to the sharing of keys, they require much larger keys in order to be secure (at least 1000 bits).

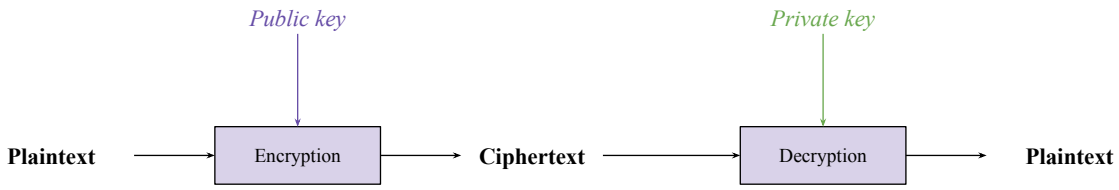


Figure 1.2: *Overview of an asymmetric encryption scheme.*

In 1978, Rivest *et al.* proposed the first asymmetric cryptosystem, Rivest-Shamir-Adleman (RSA) (8). In this cryptosystem, both the private key and the public key are derived from a set of two large prime numbers, where the public key is a multiplication of these prime numbers. The security of this method is based on the complexity of factorising large numbers.

Homomorphic Cryptosystems

Homomorphic cryptosystems are asymmetrical cryptosystems that are beneficial in signal processing as they translate a mathematical operation in the plaintext domain to another operation in the encrypted domain:

$$\mathcal{D}(\mathcal{E}(m_1) \circ \mathcal{E}(m_2)) = \mathcal{D}(\mathcal{E}(m_1 \square m_2)), \tag{1.11}$$

where $\mathcal{E}(\cdot)$ is a homomorphic encryption function, $\mathcal{D}(\cdot)$ is a homomorphic decryption function, $\circ$ and $\square$ are two mathematical operations which can be, but are not necessarily, the same operations, and m_1 and m_2 are the two plaintexts to be encrypted.

The advantage of homomorphic cryptosystems is that they allow a third party to modify content in the plaintext domain without the need to decrypt the content and therefore without compromising the security. What is more is that unlike non homomorphic cryptosystems which are deterministic, homomorphic cryptosystems are probabilistic and therefore for each plaintext, there exists multiple ciphertexts.

The literature describes two types of homomorphic encryption methods. The first category is fully homomorphic encryption methods, which translates multiple mathematical operations in the plaintext domain to operations in the encrypted domain. However, fully homomorphic encryption methods cannot yet be applied to real life scenarios, as they are inefficient due to their high computational complexity. In 2009, Gentry was the first to propose a fully homomorphic encryption scheme, which is based on ideal lattices (9).

The second category is partially homomorphic encryption methods, which translates a single operation between the plaintext and encrypted domains. In 1985, ElGamal proposed the partially homomorphic ElGamal cryptosystem based on discrete logarithms (10). Both the ElGamal encryption scheme and the RSA (8) encryption scheme are multiplicative homomorphic encryption schemes, as they have the property:

$$\mathcal{D}(\mathcal{E}(m_1) \times \mathcal{E}(m_2)) = \mathcal{D}(\mathcal{E}(m_1 \times m_2)), \tag{1.12}$$

where $\mathcal{E}(\cdot)$ is the encryption function, $\mathcal{D}(\cdot)$ is the decryption function, and m_1 and m_2 are the two plaintexts to be encrypted.

In 1999, Paillier introduced the homomorphic cryptosystem, the Paillier cryptosystem (11). This cryptosystem converts a multiplication in the encrypted domain to an addition in the plaintext domain. To generate the keys, we choose two prime numbers p, q such that:

$$gcd(pq, (p-1)(q-1)) = 1. \tag{1.13}$$

Set n and λ such that:

$$n = pq \text{ and } \lambda = lcm((p-1), (q-1)). \tag{1.14}$$

Choose $g \in (\mathbb{Z}/n^2\mathbb{Z})^*$ such that:

$$\exists\, \mu \mid \mu \ = (L(g^{\lambda} \bmod (n^2)))^{-1} \bmod (n), \tag{1.15}$$

where $L(\cdot)$ is defined as:

$$L(x) = \frac{x-1}{n}, \text{ where } x \in \mathbb{N}^*. \tag{1.16}$$

The public key is given by (n, g) and the private key by (λ, μ). If m is a plaintext to be encrypted, where $0 \leq m < n$, r is randomly generated, where $r \in (\mathbb{Z}/n\mathbb{Z})^*$, and $\mathcal{E}(\cdot)$ the Paillier encryption function, then the ciphertext c is:

$$c = \mathcal{E}(m) = g^m \times r^n \bmod n^2. \tag{1.17}$$

Consequently, the maximum size of the ciphertext is n^2, whereas the maximum size of the plaintext is n. It is the random value of r which guarantees the cryptosystem's probabilistic property. This property indicates that the encrypted value of a plaintext is not unique.

From the ciphertext c, the initial message m is retrieved:

$$m = \mathcal{D}(c) = L(c^{\lambda} \bmod n^2) \times \mu \bmod n, \tag{1.18}$$

where $\mathcal{D}(\cdot)$ is the Paillier decryption function.

The Paillier cryptosystem has multiple homomorphic properties which we exploit in our proposed method presented in this paper. The first of which is the Paillier additive homomorphic property which converts a multiplication in the encrypted domain to an addition in the plaintext domain:

$$\mathcal{D}((\mathcal{E}(m_1) \times \mathcal{E}(m_2)) \bmod n^2) = (m_1 + m_2) \bmod n, \tag{1.19}$$

where m_1 and m_2 are the two plaintexts to be encrypted.

As the homomorphic cryptosystems are probabilistic, by definition there exists multiple values of $\mathcal{E}(m)$ for every m. We can then modify $\mathcal{E}(m)$ such that:

$$\mathcal{D}(\mathcal{E}(m) \times (t^n \bmod n^2) \bmod n^2) = m \bmod n, \tag{1.20}$$

where t is relatively prime to n.

This property is termed the self-blinding property.

Concerning its security against attacks, the Paillier cryptosystem is IND-CPA secure (*i.e.* indistinguishable under chosen-plaintext attacks). It can be IND-CCA1 secure (*i.e.* indistinguishable under non-adaptive chosen ciphertext attack) depending on the parameters used. However, like all homomorphic cryptosystems – which are known to be malleable – it cannot be IND-CCA2 secure (*i.e.* indistinguishable under adaptive chosen ciphertext attack) (12).

1.3 Image Encryption

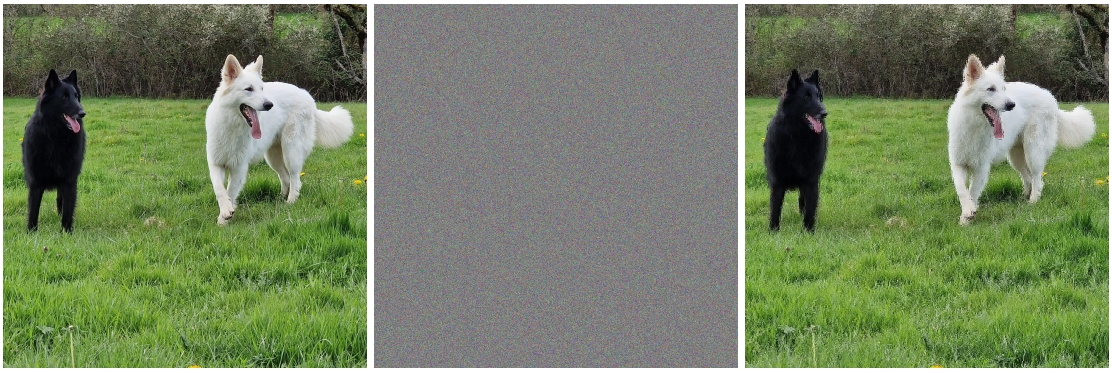
Images, which can be defined as a matrix of pixels, and are most commonly in greyscale or color, though many other types of images also exist. Each pixel in a greyscale image, typically encoded with 8 bits per pixel, represents a shade of gray, where 0 corresponds to black and 255 corresponds to white. Whereas pixels in color images are composed of three different channels. These three channels construct a color space, where each pixel is typically encoded with 8 bits per pixel per channel. While the most common type of color space is the red, green, blue (RGB) color space, other color spaces such as the HSL, CIELab, XYZ or YCrCb are also common. Fig. 1.3 presents an example of an RGB image (Fig. 1.3a) and the same image in greyscale (Fig. 1.3b), along with their respective color spaces.

(a) *RGB* **(b)** *Greyscale*

Figure 1.3: *Example of a color image (RGB) and its corresponding greyscale* image.

Image encryption methods aim to protect the visual confidentiality of the image, so that the content is illegible. Methods presented in Section 1.2 can be adapted in order to encrypt images. Fig. 1.4a presents the original image, and Fig. 1.4b presents the corresponding image encrypted with the AES encryption method with the CFB operation mode. In this case, the AES encryption method has been adapted for images. Fig. 1.4c presents the corresponding selectively encrypted image, where part of each pixel is encrypted. The last category of encryption is partial encryption methods, which encrypts a designated area of an image.

However, over the years, many encryption methods have been specifically designed for images. In 2005, Guan *et al.* (13) proposed a chaos-based image encryption method based on the Arnold cat map and the chaotic system developed by Chen and Ueta (14). Then, in 2007, Usman *et al.* developed an encryption method designed for medical images, based on pixel permutation (15). In 2019, Ghadirli *et al.* presented a comprehensive overview of encryption algorithms for color images (16). Also in 2019, Wang *et al.* described a parallel image encryption method based on diffusion and permutation (17). Recently, in 2022, Xian *et al.* proposed a chaotic image encryption scheme with a spiral-transform-based fractal sorting matrix (18).



(a) *Original* **(b)** *Encrypted* **(c)** *Selectively encrypted*

Figure 1.4: *The original image and the corresponding encrypted and selectively encrypted images.*

Many objective image quality assessment methods exist (19), notably the peak signal to noise ratio (PSNR) metric, which measures the difference between two images by means of a pixel by pixel comparison based on the root mean squared error (RMSE), and the structural similarity (SSIM) (20) which compares the luminance, contrast and structural differences between two images. However, these objective methods do not take into account how aesthetically pleasing an image is for the human visual system (HVS). In 2012, Murray *et al.* established a database AVA for aesthetic visual image analysis (21). Later, in 2015, Lu *et al.* showed that a deep neural network can be used to estimate image aesthetics (22). In 2018, Talebi and Milanfar presented a no-reference subjective neural image quality assessment NIMA (23). Contrary to previous methods, NIMA does not aim to predict the mean opinion score (MOS), but the distribution of opinion scores. Then in 2020, Zhai and Min presented a survey on perceptual image quality assessment methods (24).

Many visual security metrics have been also been proposed for images over the years. These metrics aim to measure how secure the content of an encrypted image is. In 2009, Yao *et al.* introduced a visual security assessment based on neighborhood similarity (25). Then in 2014, Jenisch and Uhl (26) described a visual security evaluation based on SIFT (27). In 2016, Xiang *et al.* proposed a perceptual visual security index by analyzing the texture and extracted edges of an image (28). Xiang *et al.* later described in 2020 a visual security index based on spatial contrast and texture features of an image (29). Abraham *et al.* suggested a visual security evaluation method using edge, texture and wavelet frequency information from the original and encrypted image (30). In 2020, Guo *et al.* proposed a perceptually encrypted image database for visual security evaluation (31). Even more recently, in 2021, Yang *et al.* described a visual security index based on a convolutional neural network (CNN) (32).

1.4 3D Object Encryption

We define a 3D object as a digital approximation of a continuous surface. There are two main ways in which 3D objects are created. Synthetic 3D objects are created by designers with the use of computer aided design (CAD) software. 3D objects can also

be acquired from real life data. The first way of creating these 3D objects is by scanning real life data. We note that many different types of scanners exist. This is common in the healthcare industry, for example. Another popular way of generating 3D objects from real life data is with photogrammetry, which consists of constructing a 3D object from multiple 2D acquisitions. Recently, in 2023, Verykokou and Ioannidis present an overview of image-based and scanner-based 3D modeling technologies (33).

1.4.1 3D Object Representation

While many representations of 3D objects exist, both continuous and discrete, our work focuses solely on 3D polygon meshes and in particular, 3D triangle meshes. Polygon meshes are also one of the most popular ways in which a 3D object can be represented. This is especially common in industries such as the manufacturing industry or the entertainment industry for example, where 3D objects are generally synthetically created by designers with the use of CAD software.

Polygon meshes are discrete representations of a continuous surface, and are composed of two main components, which are the geometry and the connectivity. We note that other information can also be associated with these meshes, such as textures, vertex normals, among other characteristics. A polygon mesh $\mathcal{M}$ can therefore be defined as:

$$\mathcal{M} = (\mathcal{V}, \mathcal{F}), \tag{1.21}$$

where $\mathcal{V}$ represents the geometry and the $\mathcal{F}$ the connectivity.

The geometry is represented by a set of vertices $\mathcal{V} = \{v_0, ..., v_{|\mathcal{V}|-1}\}$, which defines a point cloud. Each vertex $v \in \mathcal{V}$ consists of three coordinates x, y and z, where each of which can be represented by a 32-bit floating point. According to the IEEE 754 standard, a 32-bit floating point $fp \in \{x, y, z\}$ consists of a sign s represented with 1 bit, an exponent e represented with 8 bits and a mantissa m represented with 23 bits (from MSB to LSB) where:

$$fp = (-1)^s \times m \times 2^{e-127}. \tag{1.22}$$

Fig. 1.5 illustrates how a 32-bit floating point fp is divided into s, e and m.

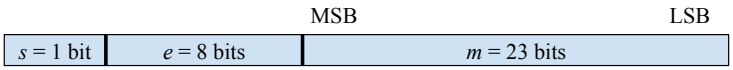
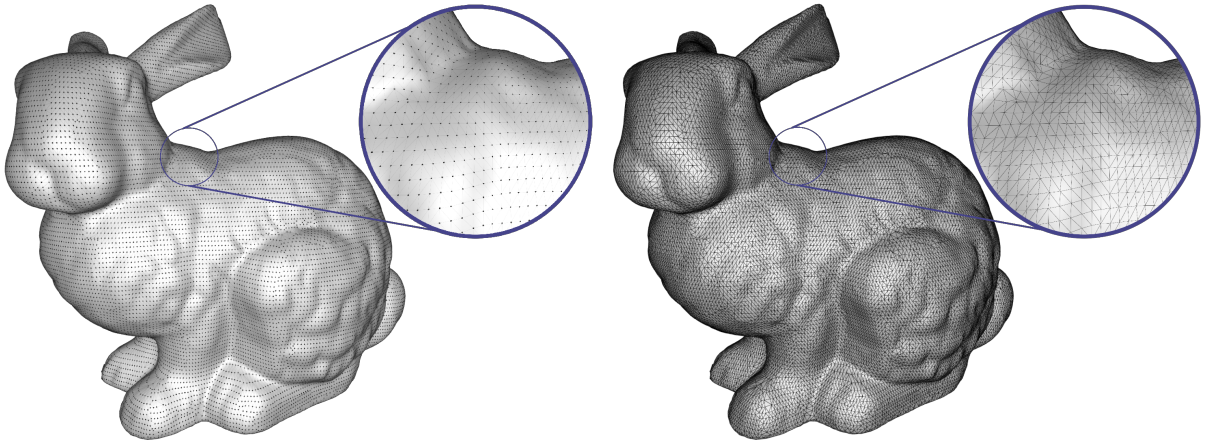


Figure 1.5: *Representation of a 32-bit floating point according to the IEEE 754 standard.*

This point cloud represents points on the surface which is being approximated. Intuitively, the number of vertices in the point cloud defines the resolution of the 3D object, as well as its storage size. Therefore, the higher the number of vertices, the greater the storage size and resolution of the 3D object. The point cloud is then connected by polygons, typically triangles, which represent the 3D object's connectivity. Fig. 1.6 presents the geometry (Fig. 1.6a) and the connectivity (Fig. 1.6b) of the 3D object *Bunny* taken from the Stanford dataset (34).



(a) *Geometry* **(b)** *Connectivity*

Figure 1.6: *The geometry and connectivity of the 3D object Bunny taken from the Stanford dataset (34).*

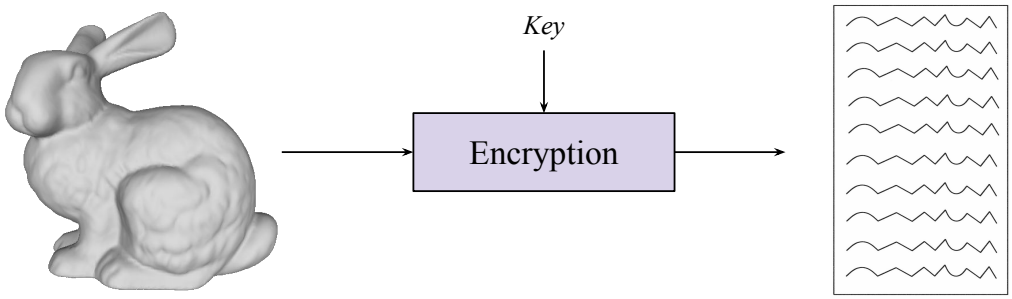
1.4.2 Fundamentals

The first category of encryption method for 3D objects is total encryption approaches, which consider the 3D objects as binary data and encrypt them using a standard encryption scheme such as RSA (8) or AES (7). However, these approaches do not take into account the structure of the 3D object. Therefore, once the 3D object is encrypted with such a method, it can no longer be visualized with the use of a standard 3D object viewer and is thus no longer format compliant. Format compliant methods aim to preserve the structure of the 3D object, as well as its original size. Fig. 1.7a illustrates a non format compliant encryption of *Bunny*, while Fig. 1.7b illustrates a format compliant encryption of *Bunny*.

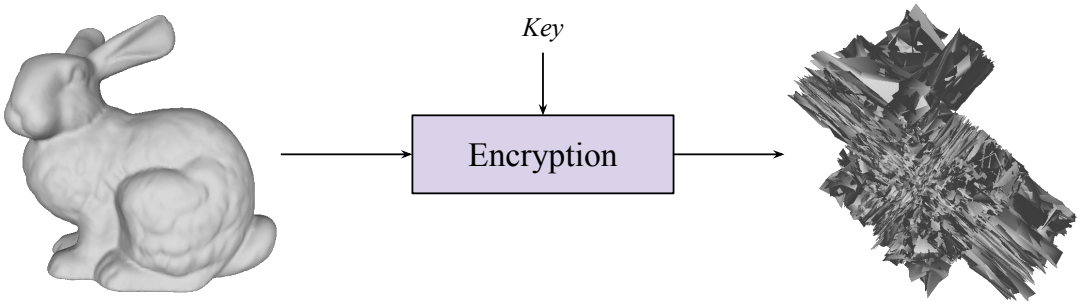
Recently, some format compliant encryption methods for 3D objects have been proposed. In 2015, Jolfaei *et al.* proposed an encryption scheme for 3D objects designed to conserve dimensional and spatial stability (35). Then in 2019, Wang *et al.* designed a fast 3D encryption method based on a chaos system, where the 3D object is transformed into a 2D object similar to an image (17). Very recently, in 2023, Gao *et al.* designed an encryption scheme for 3D objects based on a cascaded chaotic system, illustrated in Fig. 1.8 (36).

Visual Quality Metrics

Many different metrics have been proposed dedicated to evaluating the visual quality of 3D objects after they have undergone different operations. These operations include lossy compression, decryption, or attacks such as smoothing or zeroing among others. These metrics are mainly full reference ones. We present five quality metrics specifically applied to 3D objects. These quality metrics are the Root Mean Square Error (RMSE) (37), the Hausdorff Distance (HD) (38; 37), the PSNR (39), the Mesh Structural Distortion Measure (MSDM2) (40) and finally the Dihedral Angle Measure Error (DAME) (41).



(a) *Non format compliant*



(b) *Format compliant*

Figure 1.7: *The differences between non format compliant and format compliant encryption methods for 3D objects.*

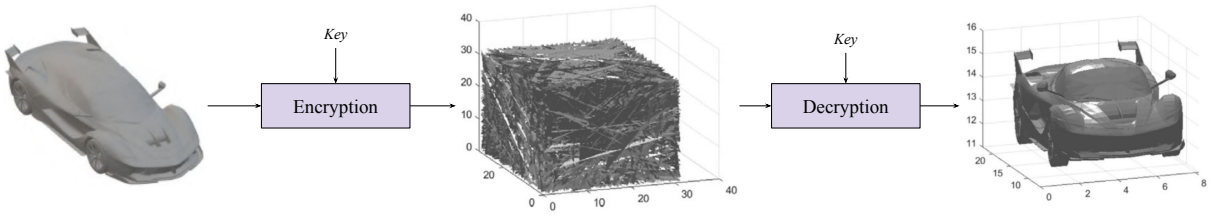


Figure 1.8: *Results of the method proposed by Gao et al. (36).*

- **The RMSE** is computed between two 3D objects O and O' by using a known correspondence between vertices:

$$\text{RMSE}(O, O') = \sqrt{\frac{1}{\mathcal{V}} \sum_{i=1}^{\mathcal{V}} ||v_i - v_i'||^2}, \tag{1.23}$$

  where $\mathcal{V}$ is the set of vertices and v_i (resp. v_i') the coordinates of the i-th vertex of O (resp. O').

- **The HD** is based on the distance between a vertex v from a 3D object O to the nearest vertex v' of a second 3D object O':

$$\begin{aligned} \text{HD}(O, O') &= \max(d(O, O'), d(O', O)), & (1.24)\\ d(O, O') &= \max_{v \in O} d(v, O'), & (1.25)\\ d(v, O') &= \min_{v' \in O'} \|v - v'\|_2. & (1.26) \end{aligned}$$

  While this metric is more costly in terms of time complexity than the RMSE, the HD is particularly useful when the correspondence between the vertices of the two 3D objects is unknown.

- **The PSNR** was initially a reference metric mainly used to evaluate the quality of 2D images. Some authors have proposed using it for 3D objects. Chao *et. al.* (39) proposed quantifying the distortion of 3D vertex positions or vertex normals using RMSE:

$$\text{PSNR}(O, O') = 20 \log_{10} \frac{D_{\text{max}}}{RMSE(O, O')}, \tag{1.27}$$

  where D_{max} is the length of the diagonal of the bounding box of the reference 3D object O.

- **MSDM2** (40) is an improvement of the metric MSDM (42) which adapts the 2D metric Structural SIMilarity (SSIM) (43) for 3D objects. In (40) and (42) Lavoué proposes replacing pixel values by the mean curvature of a 3D object and proposes a local measure defined as:

$$LMSDM(x, y) = (\alpha L(x, y)^a + \beta C(x, y)^a + \gamma S(x, y)^a)^{\frac{1}{a}}, \tag{1.28}$$

  where x and y are two local 3D windows, L represents a normalized curvature distance, C is based on the standard deviations σ_x and σ_y which reflect the roughness of the surfaces and S, by considering the covariance between local windows, which aims to detect changes in salient features.

  The MSDM between two 3D objects O and O' is defined by a Minkowski sum of their local window distances. MSDM2 provides two main improvements. The first allows users to compare 3D objects with different connectivities. It uses a step to determine the correspondence between the vertices of the reference 3D object O and those of the compared 3D object O'. The second improvement concerns the evaluation of visual differences by multi-resolution. The results of the metric are therefore more correlated with subjective assessments. With this approach, all the scores calculated on the surface area are combined into a single global score.

- **DAME** proposed by Vása and Rus (41) is used to analyze distortions in the dihedral angles of 3D object triangles with the same connectivity:

$$\text{DAME}(O, O') = \frac{\sum\limits_{\{t_1, t_2\} \in \Omega} ||D_{t_1,t_2} - \overline{D_{t_1,t_2}}|| m_{t1,t2}(w_{t_1} + w_{t_2})}{||\Omega||}, \tag{1.29}$$

  where Ω is the set of all pairs of triangles t_1, t_2 sharing the same edge, D_{t_1,t_2} the dihedral angle between t_1 and t_2 in O, $\overline{D_{t_1,t_2}}$ the dihedral angle between t_1 and t_2 in O', and $m_{t1,t2}$ the visual masking coefficient:

$$m_{t_1,t_2} = e^{k \times D_{t_1,t_2}}, \tag{1.30}$$

  where $k = 7$ is chosen empirically by the experiments of Vása and Rus (41).

  The visibility terms w_{t_1} and w_{t_2} are deduced by calculating the density of the pixels representing the triangles at different viewing angles. Vása and Rus proposed calculating this term by generating synthetic images of the 3D object from different angles in order to count the number of pixels in each image representing a triangle. The authors also proposed an approximation of this term, which is much faster. It consists of calculating the ratio between the area of the triangle and the area of the surface of the 3D object.

1.4.3 Selective Encryption

In certain contexts, such as in a digital production line in the manufacturing industry, a user or environment may have the right to access certain information about the 3D object, for example the shape but not the content, or the content but not the high quality information. Standard encryption methods presented in Section 1.4.2, which encrypt all content of the 3D object, cannot take into account different hierarchical access rights.

Selective Encryption Methods

As a solution to this problem, selective encryption methods for 3D objects have been proposed over the years (44; 45; 46; 47). These methods encrypt a specific set of data in order to protect it in accordance with the user's access rights, while leaving some data in the plaintext domain. Consequently, selective encryption methods are format compliant.

In 2006, Cho *et al.* proposed encrypting the 3D object's connectivity in order to generate a surface which contains holes, without modifying the geometry, as well as applying a *fingerprint* to the 3D object (44). Then in 2009, Gschwandtner and Uhl proposed using a progressive mesh to represent a 3D object and suggested encrypting a subset of the data (such as positions, colors or indices) in layers of this structure (45). In 2014, Eluard *et al.* presented multiple geometry-preserving encryption schemes based on permutations of vertices or coordinates (46).

Recently, in 2018, Beugnon *et al.* proposed selectively encrypting the binary representation of floating values used for each coordinate of every vertex of a 3D object (47). Fig. 1.9 illustrates an overview of the encoding process of (47). In their approach, they encrypt only the 23 bit mantissa presented in Eq. 1.22. They use a degradation level parameter p, which designates the first bit of the mantissa to be encrypted. This allows them to control the impact of the encryption on the geometry.

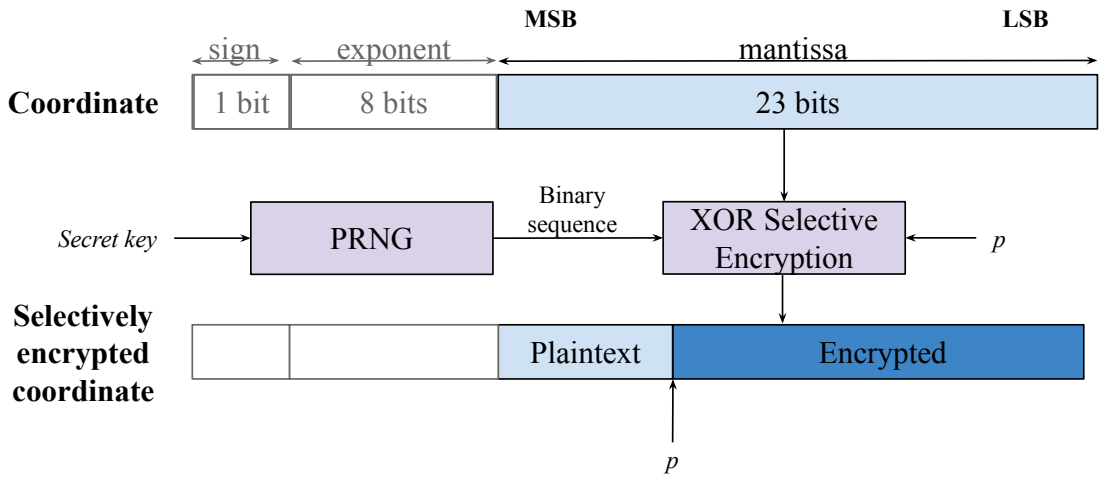
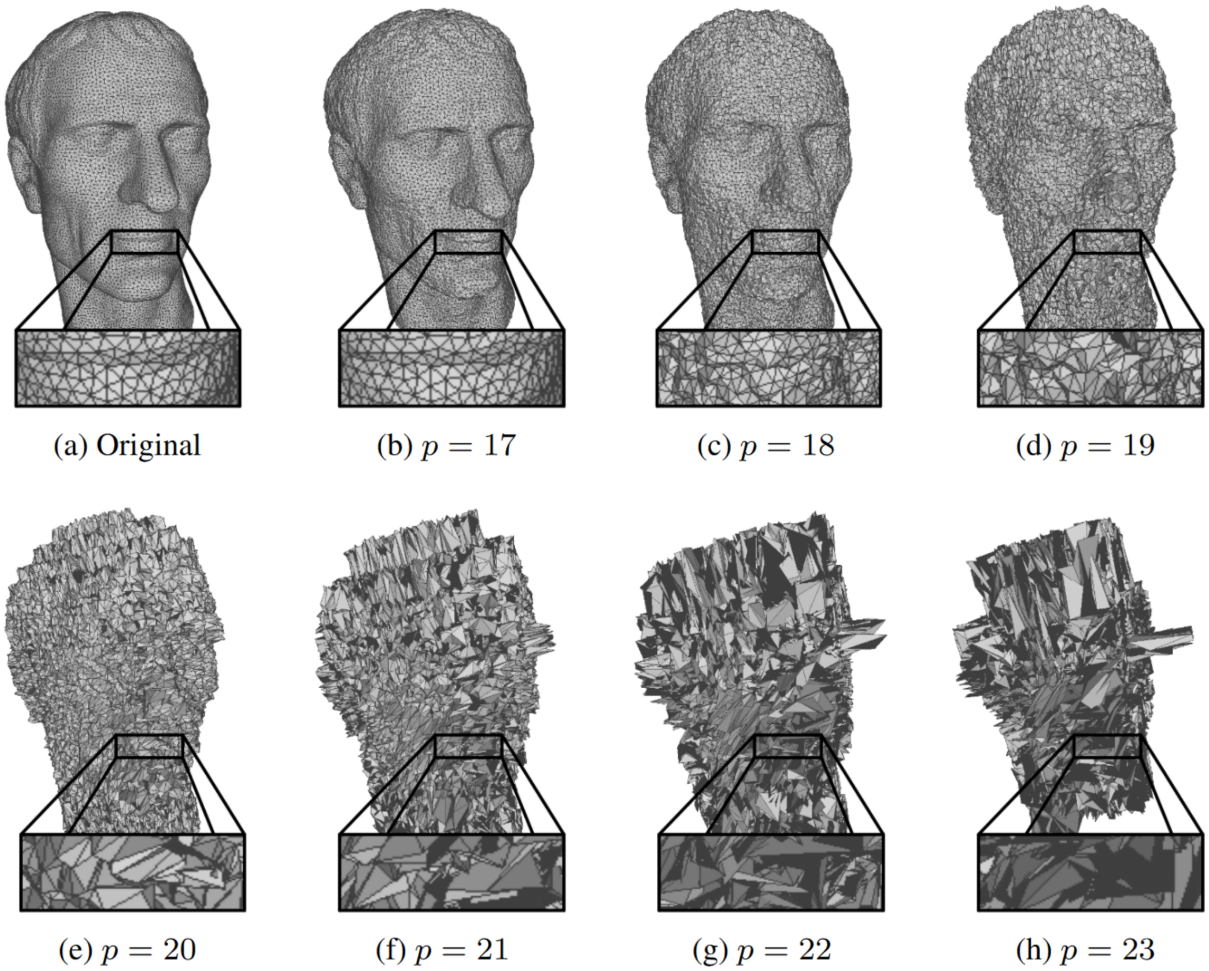


Figure 1.9: *Overview of the encoding process of the encryption scheme presented in (47).*

Fig. 1.10 shows the results of (47) according to the degradation level parameter p.

In this paper, they also show that up to 16 LSB of the mantissa of each coordinate can be lost before there is any impact on the visual quality of the 3D object.



(a) Original (b) $p = 17$ (c) $p = 18$ (d) $p = 19$

(e) $p = 20$ (f) $p = 21$ (g) $p = 22$ (h) $p = 23$

Figure 1.10: *The results of (47) according to the degradation parameter* p.

Visual Security Assessment

To the best of our knowledge, very few methods for visual security evaluation exist for 3D objects. However in 2002, Pommer and Uhl described applications for selective encryption of visual data (48). Based on this work, Beugnon *et. al* defined three different levels of visual security for selectively encrypted 3D objects: the **transparent** level, the **sufficient** level and finally the **confidential** level. These three levels are established based on the accessibility of the visual content of the 3D objects for the HVS. To the best of our knowledge, this is the only classification for 3D visual security that exists.

The **confidential level** defines the most secure level where neither the shape nor the content of a 3D object are accessible to the HVS. Fully encrypted 3D objects usually have a confidential visual security level. The **sufficient** level corresponds to 3D objects whose shape is accessible to the HVS, but not its content. Finally, the **transparent level** corresponds to a 3D object whose content and shape are recognizable, but the quality of the 3D object remains low enough to prevent it from being used, for example, for 3D printing.

Fig. 1.11 illustrates two examples of the three visual security levels applied to two 3D objects of shoes[1]. The first shoe is from the brand *Clarks*© and the second shoe from the brand *New Balance*©. At the **transparent level**, we can recognize that the objects are shoes and recognize the brand but we do not have access to the high quality information, while at the **sufficient level**, the brand is not identifiable. Finally, at the **confidential level**, we cannot even recognize that they are shoes.

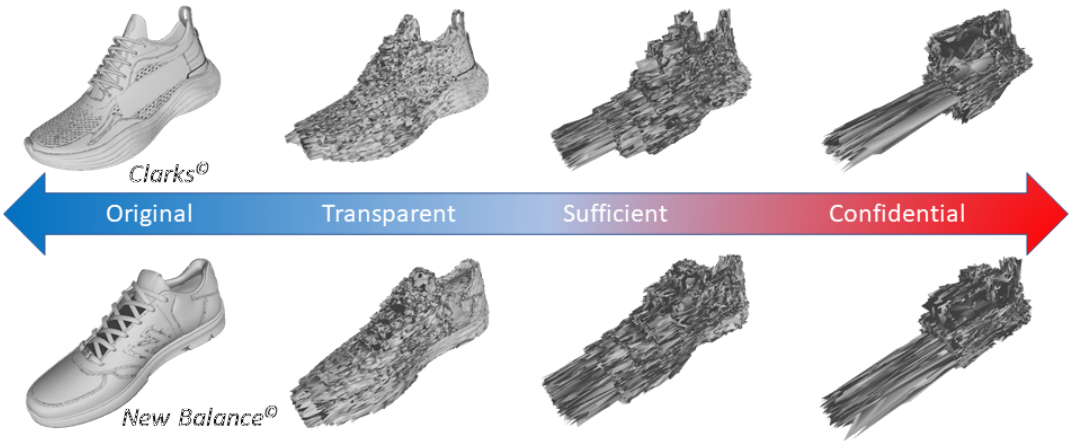


Figure 1.11: *Examples of the three visual security levels:* ***transparent****,* ***sufficient*** *and* ***confidential*** *(47) applied to two manufactured 3D objects of shoes provided by STRATEGIES (https://www.romans-cad.com/). The first shoe is from the brand Clarks*© *and the second from the brand New Balance*©*.*

Metrics designed for visual quality assessment, such as those presented in Section 1.4.2, are not well adapted to measuring the visual security of 3D objects, as visual quality assessment in 3D objects is very different from visual security assessment. While quality assessment constrains the evaluations to small visual distortions by comparing the visual quality of two 3D objects, visual security assessment tries to study a larger and broader spectrum of 3D objects where distortions intend to hinder the usage or the comprehension of the 3D object. The differences between visual quality and visual security are illustrated in Fig. 1.12. Visual quality metrics indeed cannot distinguish between the three different visual security levels (**transparent, sufficient** and **confidential**). Generally, a transparent level 3D object as well as a confidential level 3D object will both be labeled as a bad quality 3D object by a visual quality metric.

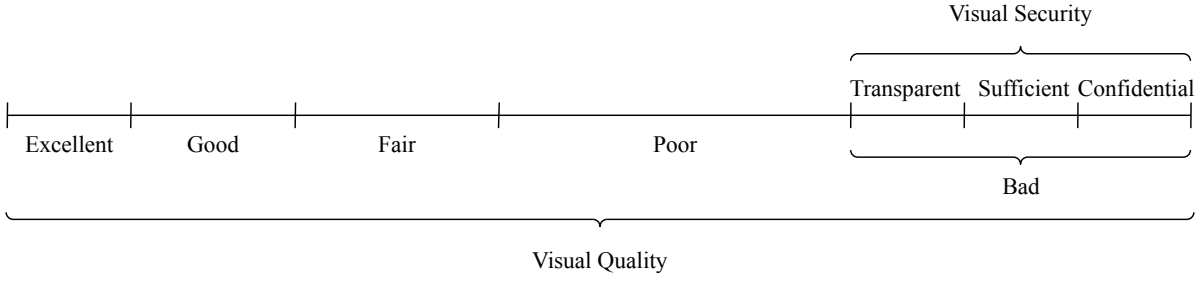
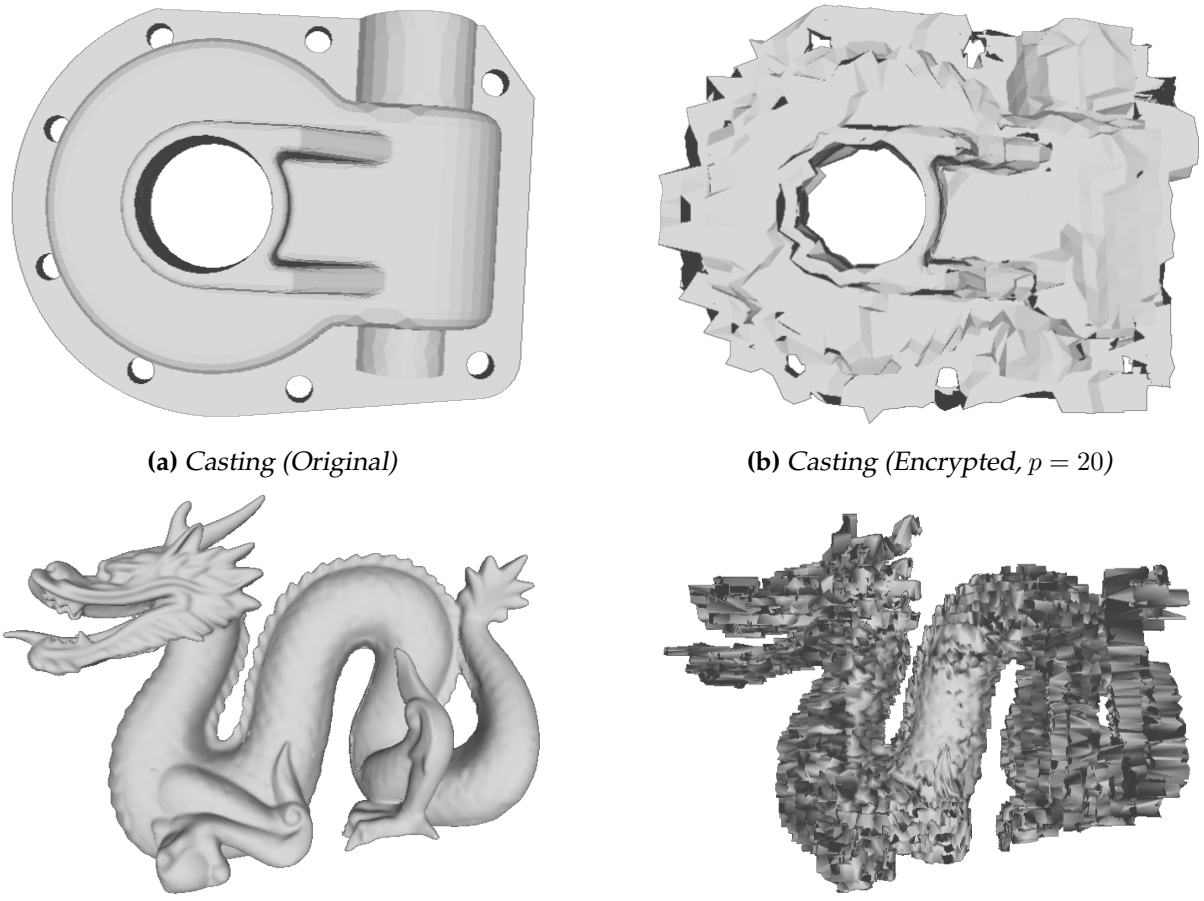


Figure 1.12: *Visual quality compared to visual security.*

[1]Provided by STRATEGIES (https://www.romans-cad.com/)

However, when evaluating the visual security level of selectively encrypted 3D objects, it is essential that HVS is taken into account. Objective metrics alone cannot accurately determine the visual security level according to the HVS, as the HVS is very subjective. Two different objects encrypted with the same parameters can indeed have different visual security levels. Many of the 3D object's characteristics, such as the form, density and smoothness of the 3D object, influence its recognizability according to the HVS.

Fig. 1.13 presents an example of this phenomenon. Fig. 1.13a presents the original 3D object *Casting*, composed of smooth surfaces and $5,096$ vertices, and Fig. 1.13c presents the original 3D object *Dragon*, composed of a more textured surface and $50,000$ vertices, where both *Casting* and *Dragon* are taken from the Stanford dataset (34). Fig. 1.13b and Fig. 1.13d present the 3D objects *Casting* and *Dragon* respectively, where both are encrypted with the method (47) using the parameter $p = 20$. In Fig. 1.13b, both the shape and content can be recognized, but the high quality details remain secure, and therefore it has a transparent visual security level. However, despite being selectively encrypted with the same parameter, in Fig. 1.13d, the shape is recognizable, but not the content and therefore it has a sufficient visual security level. To the best of our knowledge, other than our contribution which is described in Section 5, no metric exists for evaluating the visual security level of 3D objects.



(a) *Casting (Original)*

(b) *Casting (Encrypted, $p = 20$)*

(c) *Dragon (Original)*

(d) *Dragon (Encrypted, $p = 20$)*

Figure 1.13: *Comparison of two 3D objects Casting and Dragon, both taken from the Stanford dataset (34).*

1.5 Conclusion

In this chapter, we have presented a state of the art of multimedia encryption. First, we described a brief history of cryptography and how it evolved into modern cryptography. We presented Kerckhoffs' principle (3), which is the foundation of modern cryptography. We then presented symmetric cryptography and the different operation modes, and the current standards for modern cryptography. We described asymmetric cryptography, which requires larger keys but has a more efficient key sharing process, and in particular, we presented homomorphic cryptosystems.

In our work, which will be presented in Chapter 4, we take advantage of the homomorphic properties of the Paillier cryptosystem in order to design a data hiding method in the encrypted domain for 3D objects. Chapter 2 presents a state of the art of data hiding and a state of the art of data hiding in the encrypted domain will be discussed in Chapter 3.

We then detailed image encryption, where we presented the structure of an image as well as different image encryption methods developed over the years. We examined image quality assessment methods, as well as visual security metrics for images.

Finally, we presented 3D object encryption. We first detailed the representation of 3D objects. We then presented some 3D object encryption methods and the notion of format compliance, as well as some visual quality metrics for 3D objects. We then described selective encryption methods for 3D objects, where we introduced the notion of visual security for 3D objects and highlighted the differences between visual quality metrics and visual security metrics.

We have shown that while many encryption methods designed for images have been proposed over the years, very few encryption methods have been proposed for 3D objects. In particular, even fewer methods for selective encryption have been proposed for 3D objects. We also note that while a classification of different visual security levels have been proposed for 3D objects, no method exists for 3D object visual security evaluation.

In Chapter 5 we describe our proposed visual security metric for selectively encrypted 3D objects, which to the best of our knowledge, is the first metric for visual security for 3D objects. We also describe our proposed encryption method which allows for a hierarchical decryption of an encrypted 3D object to different visual security levels depending on the key used.

In Chapter 6, we detail our proposed encryption method and selective encryption method in the compressed domain for 3D objects. We note that encryption methods in the compressed domain will be discussed in Chapter 3.

CHAPTER 2

MULTIMEDIA DATA HIDING

Chapter Contents

2.1 Introduction

Another form of multimedia security is multimedia data hiding. Where cryptography seeks to secure the multimedia with the use of encryption, data hiding embeds hidden messages in the multimedia. We note that the embedded data can be in the form of text such as metadata, or multimedia, such as images, video or 3D objects. Cryptography is used when the content of the multimedia is confidential, whereas data hiding is used to ensure the multimedia's integrity. Data hiding can be used to hide sensitive information in a support, such as in the case of steganography, for high capacity data hiding, to embed copyright information, such as in the case of robust watermarking, or to detect whether the multimedia has been subjected to unauthorized alterations or sharing, such as in the case of fragile watermarking.

In this chapter, we present a state of the art of multimedia data hiding. First, in Section 2.2 we present an overview of data hiding fundamentals. We discuss an overview of the embedding phase and the extraction phase of data hiding. We also discuss the different criteria which need to be considered during data hiding and the different domains in which data hiding can be performed. We also present different evaluation metrics for the resulting multimedia. Then, in Section 2.3, we present a state of the art of steganography. We present the prisoners' problem which accurately represents steganography. We then describe a brief history of steganography and present a state of the art of image and 3D steganography. We also present steganography's counterpart, steganalysis, which aims to detect the presence of embedded messages in a multimedia support. Then, in Section 2.4, we present a state of the art of multimedia watermarking. We describe robust watermarking, which secures multimedia by embedding copyright information or by embedding user information so that the multimedia can be traced. We then present fragile watermarking, which assures the multimedia's authenticity by detecting any unauthorized modifications to the multimedia. Then, in Section 2.5, we describe high capacity data hiding, where additional data is embedded in the multimedia instead of having to be transferred in large separate files. Finally, in Section 2.6, we conclude this chapter.

2.2 Data Hiding Fundamentals

2.2.1 Data Hiding Overview

Multimedia data hiding is performed in two phases. The first is the embedding phase, in which the sender embeds a hidden message in the multimedia. The second is the extraction phase, in which the receiver extracts the hidden message from the multimedia.

Embedding Phase

The embedding phase is performed in two main steps, as described by Cox *et al.* (49). During this embedding phase, which is illustrated Fig. 2.1, a message is embedded in a multimedia support, such as an image or a 3D object. The first step is a synchronization step, which consists of aligning the message with the support. During this synchronization step, the order in which the pixels or vertices are processed is established. However, this processing order needs to be reestablished during the synchronization step of the extraction phase. While images are composed of an ordered 2D matrix of pixels, 3D objects are composed of an unordered set of vertices. Where embedding a message in the pixels of an image does not change the processing order of the pixels, embedding a message in a 3D object by changing the value of the vertices can impact the established relationship between the vertices, and therefore the original vertex processing order can be lost. This is known as the causality problem and has to be considered when establishing the processing order in the synchronization step during 3D object data hiding. In order to insure the security of the data hiding method, the synchronization step is also based on a secret key.

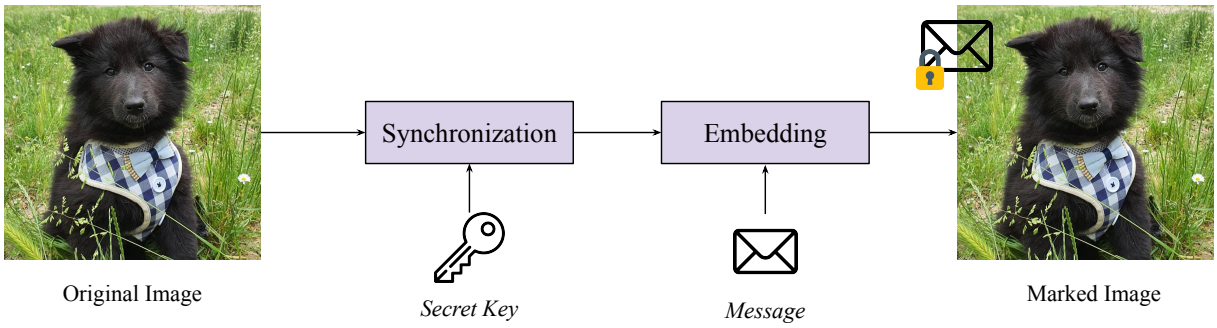


Figure 2.1: *Overview of the message embedding phase.*

Extraction Phase

During the data hiding extraction phase, which is illustrated in Fig. 2.2, the embedded message is extracted from the multimedia support. As during the embedding phase, first step of the extraction phase is the synchronization step, where the same processing order as the embedding phase is reestablished. The extraction phase synchronization step uses the same secret key as the embedding phase. The embedded message is then retrieved in one of two ways depending on the method. It is either read directly from the bits of the marked image and in this case, the message is extracted without error, or the embedded message is estimated according to an extraction criteria, which could result in errors in the extracted message. The quality of the extracted message is determined by the bit error rate (BER):

$$BER = \frac{m_{error}}{m_{length}}, \tag{2.1}$$

where m_{error} is the number of erroneous bits in the extracted message, and m_{length} is the total number of bits in the extracted message.

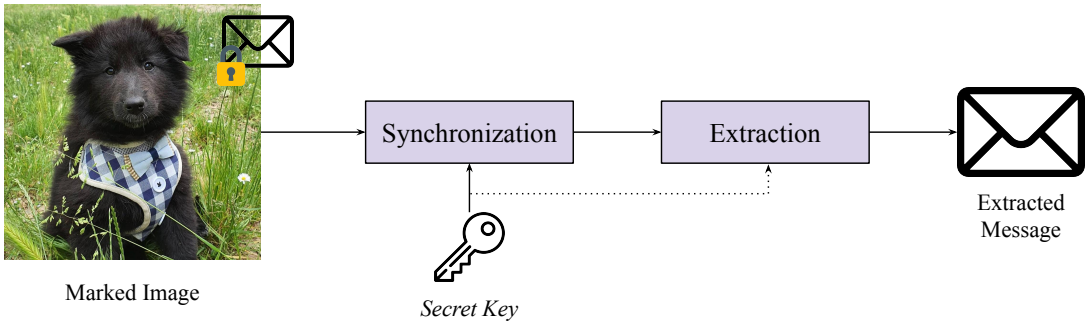


Figure 2.2: *Overview of the message extraction phase.*

During this extraction phase, the original multimedia support may or may not be required in order to extract the embedded message, depending on the data hiding method. In the case where the original multimedia support is not required, then the method is considered to be blind.

Once the message has been extracted, the original image is reconstructed. If the reconstructed image has a PSNR ≥ 50 dB, then the method is considered to be reversible.

2.2.2 Data Hiding Criteria

When constructing or evaluating different data hiding methods, five main criteria concerning the overall security of the multimedia data hiding method need to be considered. These criteria are:

- **Imperceptibility:** The embedded message is visually and statistically invisible in the support.
- **Robustness:** The resistance of the embedded message against standard or malicious processing operations applied to the support.
- **Capacity:** The quantity of bits which can be embedded in the support.
- **Security:** The security of the embedded message against attacks.
- **Complexity:** The computational complexity of the data hiding method.

The three main criteria among these five are the imperceptibility, the robustness and the capacity. However, data hiding methods are subjected to a compromise between these criteria, as prioritizing one criteria leads to sacrificing another. Fig. 2.3 presents the different categories of data hiding, where each category prioritizes different criteria. There are three main categories of data hiding, which are steganography which prioritizes imperceptibility, watermarking which prioritizes robustness, and high capacity data hiding which prioritizes capacity. We note that in the case of watermarking, two subcategories exist, which are robust watermarking and fragile watermarking. While the former aims to be as robust as possible, the latter purposefully sacrifices robustness with the objective of detecting unauthorized alterations to the multimedia.

The choice of data hiding category depends on the application, which is discussed in detail for each category in Section 2.3, Section 2.4 and Section 2.5 respectively.

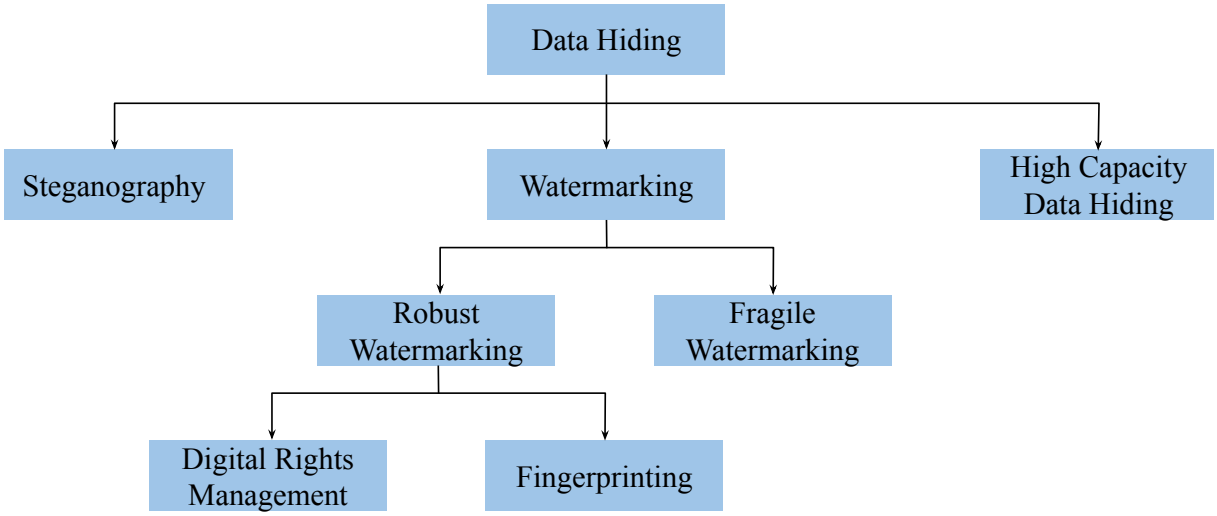


Figure 2.3: *Overview of the different data hiding categories.*

2.2.3 Data Hiding Techniques

Different multimedia data hiding methods embed the hidden message in different domains of the multimedia support. These domains include two main categories which are the spatial domain and the frequency domain. Data hiding in the spatial domain consists of modifying the multimedia's characteristics such as the pixel values in images. In 3D objects, this constitutes modifying the 3D object's geometric or topological characteristics. Frequency domain data hiding methods can modify the multimedia's characteristics in multiple different frequency domains.

Spatial domain data hiding methods generally have a higher capacity and a lower computational complexity than frequency domain data hiding methods. However, frequency domain data hiding methods are generally more robust and more imperceptible than spatial domain data hiding methods. We note that certain data hiding methods have been developed to embed data in other domains such as the encrypted domain or the compressed domain, however this will be discussed in detail in Chapter 3.

2.2.4 Evaluation

The first evaluation category is the visual quality of the resulting marked image or 3D object, or the reconstructed image or 3D object after the embedded message is extracted. For images, the peak signal-to-noise ratio (PSNR) is most commonly used. This PSNR is similar to the 3D object PSNR presented in Section 1.4.2. The PSNR is defined as:

$$\text{PSNR}(I, I') = 20 \log_{10} \frac{\text{max}^2}{RMSE(I, I')} \text{dB}, \tag{2.2}$$

where I is the original image, I' the resulting marked image, and max the maximum possible value of a pixel.

For images, the RMSE can be defined as:

$$\text{RMSE}(I, I') = \sqrt{\frac{1}{n} \sum_{i=1}^{n} ||p_i - p'_i||^2}, \tag{2.3}$$

where I is the original image, I' the resulting marked image, n the number of pixels, p_i the ith pixel of I and p'_i the ith pixel of I', with $i \in [0, n-1]$.

Concerning 3D objects, the visual quality metrics presented in Section 1.4.2, notably the RMSE and the HD metrics are most commonly used.

The second evaluation category is the capacity. For images, this is measured in the number of embedded message bits per pixel (*bpp*). For 3D objects, this is the number of embedded bits per vertex (bpv). This is also known as the embedding rate or the payload.

The final evaluation category is the extraction error of the embedded message, the BER (Eq. 2.1). This corresponds to the number of false bits extracted divided by the total number of bits extracted.

We note that as for the different security criteria, there is often a compromise between the embedding rate and the visual quality of the resulting multimedia. We also note that these evaluation metrics measure the performance and not the security of the multimedia data hiding method.

2.3 Steganography

The word steganography comes from the Greek *steganographia*, which means *covered writing*. Steganography refers to embedding messages in a support, where the embedded messages are visually and statistically imperceptible. This implies that the characteristics of the support, in particular the statistical distribution, must not be modified. Format compliant properties such as the size and format of the support must also be respected. Steganography methods however do not need to be robust or have a high embedding rate. Contrary to watermarking, in steganography, the multimedia support has no other value other than serving as a support and it is only the embedded message that needs to be secured. This is a more discrete way of securing information than cryptography, as when cryptography is used, it is evident that the sender wishes to conceal certain information, whereas steganography aims to deceive an attacker or the environment.

2.3.1 The Prisoners' Problem

The concept of steganography can be best described by the prisoners' problem, illustrated in Fig. 2.4, which was presented by Simmons in 1984 (50). There are two prisoners, Alice and Bob, who are locked in separate prison cells and their only means of communication is through messages delivered by agents of the warden Eve. In multimedia data hiding, these messages take the form of multimedia such as images or 3D objects, which are known as the support. Eve suspects that Alice and Bob are trying to coordinate an escape plan and so all communication, *ie.* the support, is examined by Eve. Alice and Bob are forced to accept these conditions if they wish to communicate.

Alice and Bob therefore need to communicate with one another by embedding imperceptible messages in the multimedia support. They need to deceive Eve, as she is trying to intercept these embedded messages. If Eve suspects that the support contains embedded messages, then Eve may also try to deceive Alice and Bob by introducing false messages. Consequently, Alice and Bob need to authenticate the message received.

In a real-life scenario, Eve can represent the environment or a malicious attacker from the perspective of the sender. Alice and Bob use steganography to exchange embedded messages, while Eve uses steganalysis to detect whether a support contains an embedded message. Steganalysis will be further described in Section 2.3.5.

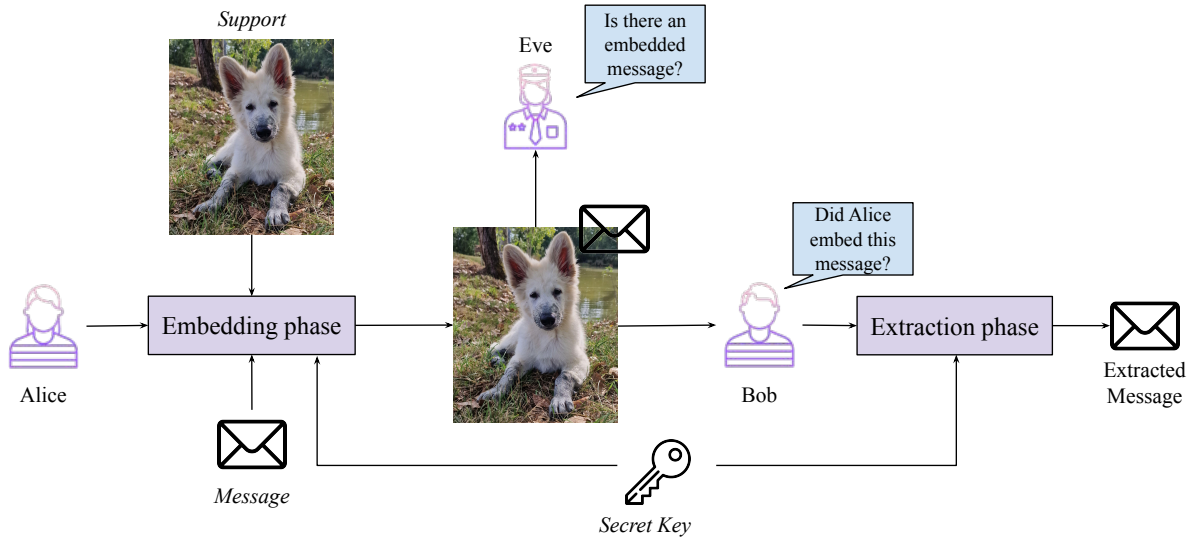


Figure 2.4: *An overview of the prisoners' problem.*

2.3.2 A Brief History

One of the first recorded uses of steganography dates back to around 500 BC, where the Greek historian Herodotus describes the tyrant Histiaeus shaving the head of a trusted slave. Histiaeus then tattooed a secret message on the back of the slaves head. Once the slave's hair had grown back, he was sent to deliver the message to Histeaus's son-in-law to warn him about an impending attack of the Persian army. The message could be read when the slave's head was shaved once more.

Historically, steganography has also been commonly used during war, as different divisions of the same army would wish to communicate with one another without arising the suspicions of the enemy. Notably, the Germans used a microdot technique during World War II. This technique consists of reducing an image or a document to the size of a dot, which is then embedded on a support, such as common paperwork or other everyday items. Fig. 2.5 presents a doll used as a support. Carrier pigeons are also considered to be steganography, as they carried hidden messages.

Figure 2.5: *A doll used as a support for the microdot technique.*

While encryption techniques render the message illegible, if an opponent intercepts the message, the opponent would not allow the message to reach its destination and would try to decipher the encrypted message. However, steganography aims to deceive the opponent into believing that no message exists. We note that in modern day steganography, the messages are generally encrypted before they are embedded.

2.3.3 Image Steganography

Image steganography consists of embedding hidden messages in the form of text such as metadata or bit streams, images or video inside a support image. In 1999, Marvel *et al.* first introduced spread spectrum image steganography method, where a message is embedded in inherent noise in the support image (51). In 2005, Potdar *et al.* conducted a survey of steganography and watermarking methods for images (52). Recently, in 2021, Subramanian *et al.* discussed recent deep learning image steganography techniques (53). Then, in 2022, Pan *et al.* developed an image steganography method where the support image first undergoes a multi-wavelet transformation and the embedding region is selected in the wavelet domain (54). The message is then embedded by intensity factor after the embedding region is processed.

2.3.4 3D Object Steganography

Like the concept of image steganography, 3D object steganography consists of embedding hidden messages in the form text, images or video inside a support 3D object. Although the concept is the same as that of image steganography, 3D object steganography can be more challenging due to the causality problem discussed in Section 2.2.

In 2003, Cayre and Macq proposed a 3D object steganography method with the use of an enhansed triangular strip peeling sequence (55). In 2010, Amat *et al.* proposed a lossless steganography method for 3D objects (56). They solve the causality problem as the vertex positions are never modified, but they instead modify the vertex connectivity in selected areas. This method is based on the concept of a minimum spanning tree. Fig. 2.6 presents an example of this method, where Fig. 2.6a illustrates the original point cloud (43,039 vertices) and Fig. 2.6b its corresponding minimum spanning tree. Then in 2017, Li *et al.* proposed a high capacity steganography method based on a Hamiltonian path. This method increases the resistance to steganalysis, which will be discussed in Section 2.3.5. In 2019, Farrag and Alexan proposed a 3D object steganography method where the binary bits of the message are embedded in the 3D object by manipulating the 3D object's polygons (57).

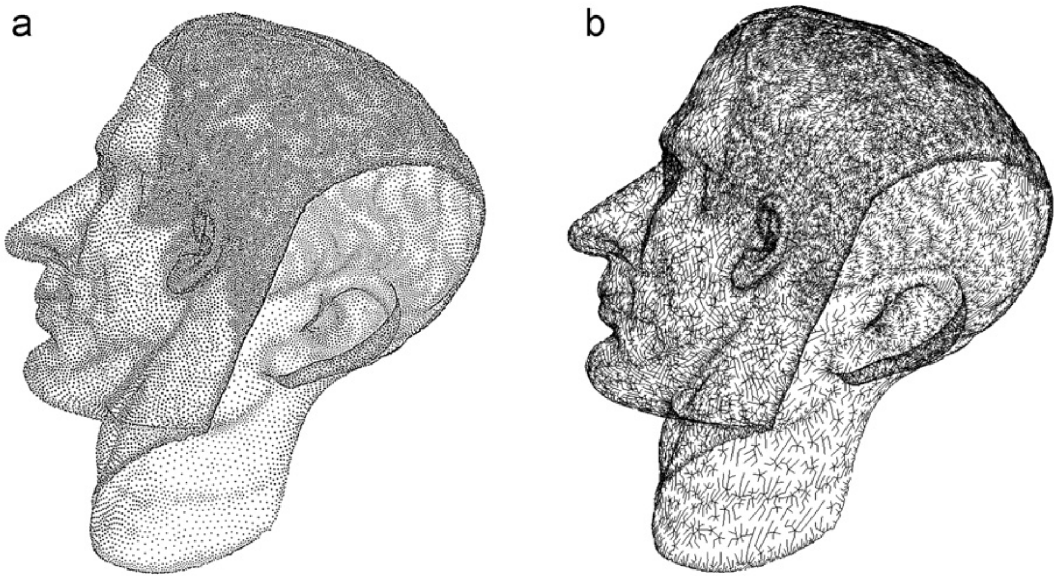


Figure 2.6: *An example of method described in (56) with a) The original point cloud and b) The corresponding minimum spanning tree.*

2.3.5 Steganalysis

Steganalysis is the complementary domain to steganography. While steganography aims to embed hidden messages in a support, steganalysis aims to detect whether embedded messages exist in a support, and thus detect steganography. In the prisoners' problem, Eve is the steganalysist. If an embedded message is detected, the steganalysist either blocks the message from reaching its destination, or they introduce interference in the form of processing operations or noise in order to corrupt the embedded message.

While Eve or the opponent is considered to be an attacker or the environment, Eve's intent is not necessarily malicious. Many viruses are spread using a subset of steganography, such as viruses known as Trojan horses, as described by Cole (58). Trojan horses are viruses which are hidden in a seemingly harmless support which the recipient installs. This virus is named after the seemingly harmless mythical giant wooden horse in which the Greeks hid in order to invade the city of Troy. We note that Trojan horses are not considered to be true steganography, as the true nature of the embedded message is unknown to the recipient. Steganalysis is also used to detect other forms of criminal activity, such as other communications with malicious intent.

We note that in real-life scenarios, steganalysis is generally more challenging than steganography. The steganalysist has no information on what characteristics of the multimedia have potentially been exploited to embed the hidden messages, or the parameters which have been used. However, in most steganalysis methods, some information is assumed to be known. In this case, steganalysis methods are also very useful for evaluating the security of steganography methods. 3D object steganalysis is also generally more challenging than image steganalysis due to the irregular structure of 3D objects.

Some steganalysis methods have been proposed for images over the years. In 2009, Pevny *et al.* proposed a steganalysis method for the detection of spatial domain image steganography methods by modeling the difference between adjacent pixels (59). In 2020, You *et al.* proposed a steganalysis method for images with the use of a Siamese CNN (60). In 2022, Eid *et al.* performed a survey on image and 3D object steganalysis (61). They show that most steganalysis methods are also formulated as a binary classification problem.

Very few steganalysis methods have been proposed for 3D objects. It was not until 2014 that Yang and Ivrissimtzis proposed the first steganalysis method for 3D objects (62). In 2017, Li and Bors proposed a steganalysis method for 3D objects with the use of local geometric features of 3D objects (63). Then in 2020, the same authors, Li and Bors, developed a 3D object steganalysis method with the use of a 3D wavelet multi-resolution analysis (64). Later in 2021, Zhou *et al.* proposed a 3D object steganalysis method based on a neighborhood-level representation-guided tensor voting model (65).

2.4 Watermarking

Steganography aims to protect the embedded message by rendering it as imperceptible as possible, however the support itself has no value, whereas watermarking seeks to protect the support with the use of embedded messages. There are two main categories of watermarking. The first is robust watermarking, where robustness is prioritized in order to insure that the intellectual property of the creator or the copyright of the owner is conserved. The second is fragile watermarking, where robustness is purposefully sacrificed in order to ensure the authenticity of the multimedia. It can also be used to embed user information in the multimedia so that the multimedia can be traced. Fragile watermarking methods also aim to localize the tampered zone.

2.4.1 Robust Watermarking

Robust multimedia watermarking methods are used to embed essential information which allows for the tractability of the multimedia. In these robust watermarking methods, the embedded message is still recoverable after the multimedia, such an image or a 3D object, has undergone different processing operations or attacks. These processing operations can include scaling, transformations or compression, among others. In the case of images, attacks can include blurring or zeroing where a certain number of LSB are replaced with zeros, among others. In 3D objects, attacks can include smoothing or zeroing, among others.

Digital Rights Management

Digital rights management (DRM) refers to ensuring that the digital rights of the data are respected. Robust watermarking is used to ensure DRM by embedding a watermark in the form of copyright or intellectual property information in the multimedia.

In 1994, van Schyndel *et al.* presented one of the first digital watermarking schemes for images (66). This work discussed the feasibility of an imperceptible watermark and proposed two different watermarking methods. In 2002, Gunsel *et al.* proposed a robust watermarking technique designed for images of fingerprints, which does not corrupt the features of the fingerprints (67). In 2011, Poljicak *et al.* proposed a watermarking method based the discrete Fourier transform of an image, where the watermark is embedded in the magnitude coefficients of the Fourier domain (68). Recently, in 2021, Alshoura *et al.* established a review of hybrid singular value decomposition based watermarking schemes for images (69). Very recently, in 2022, Begum *et al.* developed a hybrid blind watermarking method based on a combination of a discrete cosine transform, a discrete wavelet transform, and singular value decomposition (70).

Over the last two decades, many methods for 3D object watermarking have been proposed. In 1999, the first watermarking scheme for 3D objects was proposed by Benedens (71). Then, in 2005, Zafeiriou *et al.* proposed a two blind robust watermarking methods for 3D objects (72). Notably in 2007, Cho *et al.* proposed a statistical 3D object watermarking method, where data is embedded by using the distribution of vertex norms (73). Then in 2010, Wang *et al.*, established a benchmark for 3D object watermarking (74). In 2013, Bors and Luo proposed a 3D object watermarking method which aims to minimize surface distortion, where the distribution of the distances from the vertices to the center of the 3D object are used to embed the watermark according to the Levenberg-Marquardt optimization for the spherical coordinates of the 3D object (75). Very recently, in 2022, Yoo *et al.* developed a 3D watermarking method, where the watermark can be extracted from 2D renderings (images) of the 3D object with the use of a novel end-to-end learning framework (76). This method is illustrated in Fig. 2.7, where the watermark "CVPR" is extracted from 2D renderings of a 3D object under different camera angles and lighting conditions.

Over the years, watermarking methods robust to 3D printing have also been proposed. In 2017, Hou *et al.* proposed a blind watermarking method for 3D printing

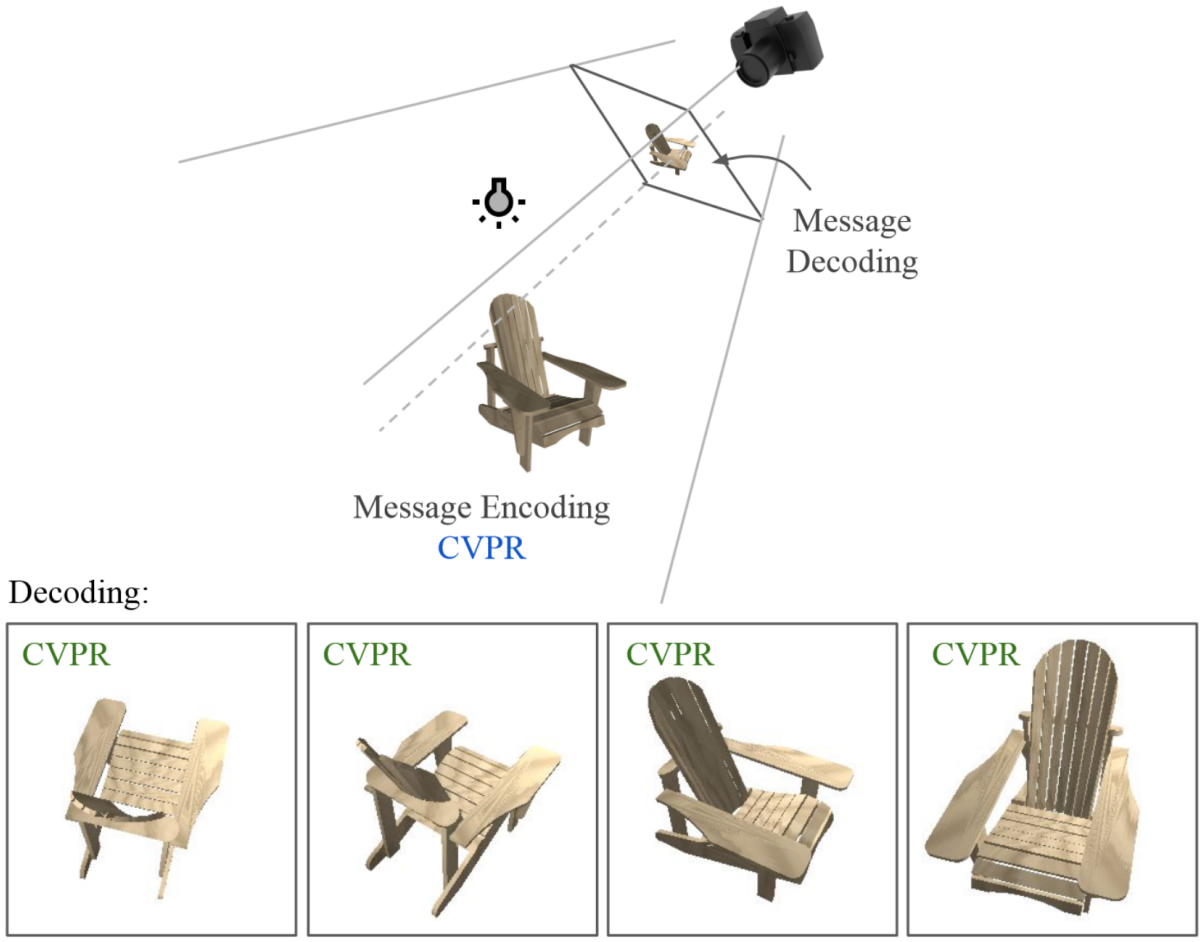


Figure 2.7: *A watermark is extracted from a 2D rendering of a 3D object under different camera angles and lighting conditions (76).*

based on analyzing the layering artifact (77). Then in 2021, Delmotte *et al.* described a blind watermarking method for 3D printing with the use of 3D moments for synchronization, and surface norm distribution (78). In 2022, Windolf *et al.* designed a blind watermarking method for the 3D printing of medical tablets (79).

Fingerprinting

While robust watermarking is often used to embed the owner's copyright information, it can also be used to identify internal leaks. This sub category is called fingerprinting, also known as traitor tracing, where a unique digital fingerprint of each user of the multimedia is embedded. Multimedia fingerprinting can therefore be viewed as a combination of a robust watermarking method and a fingerprinting code which is embedded in the multimedia. For example, in a digital production line, multimedia such as images or 3D objects are considered to be important assets as they often represent future products. Each time the multimedia is transferred, the recipient's fingerprint is embedded in the multimedia. In the case where the multimedia leaked, all the recipients of the multimedia can be traced. Fingerprinting is also used in the case where multiple people, for example employees, have access to the same multimedia. Each

person is given access to their own unique copy of the multimedia, fingerprinted with their identification information.

However, in the case where two people collude in order to leak private information, they can compare their versions of the same multimedia in order to locate the fingerprinting code and modify it, and consequently protecting their identities. In 1998, Boneh and Shaw proposed the fingerprinting code *Boneh-Shaw* based on randomization of a code matrix, which is robust against this type of attack (80). In 2003, Tardos then proposed a fingerprinting code *Tardos* which uses randomization in a less restrictive way than Boneh-Shaw (81). In 2008, Xie *et al.* described a multimedia fingerprinting method based on a zero-bit side-informed watermarking technique and a Tardos fingerprinting code (82). In 2011, Desoubeaux *et al.* designed a multimedia fingerprinting method which selects suspicious users and is based on a probabilistic traitor tracing code and an orthogonal zero-bit informed watermark (83). Recently, in 2022, Baaouni *et al.* proposed an image traitor tracing scheme where they convert a Tardos fingerprinting code to a QR code which they embed as a watermark (84).

Fingerprinting is also very useful for 3D printing, where the 3D object is printed with a fingerprint associated with the 3D printer, which allows the origin of the printed 3D object to be traced. Very few fingerprinting methods have been proposed for 3D objects. In 2018, Li *et al.* suggested that all 3D printers have their own unique fingerprint, and proposed a method, which they named PrinTracker, for detecting the origin of a printed 3D object using these fingerprints (85). In 2021, Gao *et al.* suggested equipping 3D printers with a ThermoTag, which watermarks printed 3D objects using the printer's thermodynamic properties (86).

2.4.2 Fragile Watermarking

Fragile multimedia watermarking is used to ensure the authenticity of multimedia such as images or 3D objects. These watermarks are designed to be modified or destroyed if the multimedia undergoes processing operations or attacks. Consequently, if the embedded watermark is modified or destroyed, this signifies that the multimedia has been altered and is no longer considered to be authentic. These methods also aim to localize the altered zone of the multimedia. These types of alterations can include processing operations, forgery of the multimedia, or attacks. Fragile multimedia watermarking is particularly useful in domains such as law, defense, healthcare and journalism, among others. Fragile watermarking methods can be classified into two categories, which are semi-fragile watermarking and complete fragile watermarking. Semi-fragile watermarking methods allow certain predefined modifications such as compression, which is not the case with complete fragile watermarking.

Over the last few decades, many methods have been proposed for fragile image watermarking. In 1999, Lin and Delp conducted a review of fragile image watermarks (87). In their review, they established different criteria which fragile watermarking should consider. Notably, fragile image watermarks should distinguish altered areas from non-altered areas without the use of the original multimedia content. In 2007, Liu *et al.* proposed a fragile image watermarking method which embeds a

watermark with the use of a generated chaotic map (88). An example of this method is illustrated in Fig. 2.8. Fig. 2.8a illustrates the original image *Lena* to be watermarked. Fig. 2.8b illustrates the watermark to be embedded and Fig. 2.8c illustrates the chaotic map generated. Fig. 2.8d presents the resulting watermarked image. Then in 2018, Shehab *et al.* designed a fragile singular-value decomposition based fragile watermarking scheme for medical images (89). Recently, in 2022, Lefèvre *et al.* proposed a semi-fragile watermarking method for image tampering localization with the use of error control codes (90).

(a) *Image Lena to be watermarked.* **(b)** *The watermark to be embedded.*

(c) *The chaotic map.* **(d)** *The watermarked image Lena.*

Figure 2.8: *An illustration of (88): a) The original 256 × 256 size image Lena, b) The watermark to be embedded, c) The chaotic map, d) The resulting watermarked image Lena with a PSNR of* 51.13 *dB.*

Fragile watermarking methods have also been proposed for 3D objects over the last few decades. In 2005, Wu and Cheung proposed a 3D object watermarking method which adjusts the vertex positions, while the 3D object's topology remains unchanged (91). In 2008, Wang *et al.* established a comprehensive survey on different watermarking techniques for 3D objects (92). This survey includes both robust and fragile watermarking methods. Recently, in 2021, Peng *et al.* proposed a reversible semi-fragile watermarking method for 3D objects which generalizes 2D regional nesting to an n-dimensional space (93).

2.5 High Capacity Data Hiding

The final main category of data hiding is high capacity data hiding, which as the name would suggest, prioritizes the greatest possible embedding capacity. This is particularly useful when there is metadata to be transferred along with the multimedia. Instead of transferring the metadata in multiple files, it can be embedded directly in the multimedia. This metadata can be information such as textures or vertex normals for 3D objects, or patient information and diagnoses in healthcare.

While all data hiding methods seek to be imperceptible, a certain category of high capacity data hiding also seeks to embed data while conserving the quality of the original multimedia. This category of reversible data hiding is known as high capacity *reversible* data hiding. Reversibility is essential in some domains such as healthcare, where information loss could lead to a misdiagnosis for example. Concerning images, a data hiding method is considered reversible if the resulting marked image has a PSNR ≥ 50 dB.

Many high capacity data hiding methods have been proposed for images over the years. Notably, in 2005, Ni *et al.* proposed a reversible data hiding method for images by implementing a histogram shift (94). An overview of a histogram shift is illustrated in Fig. 2.9. This method consists of finding the maximum point in the histogram (the peak) and the minimum point of the histogram (the zero). The histogram is then shifted between the peak and the zero by one place in the direction of the zero. The peak is then used to embed the message bit by bit. For each pixel in the peak, if the bit to embed is one, then the value is unchanged, otherwise it is incremented by one. Over the years, many methods using variations of this histogram shifting method have been proposed (95; 96; 97; 98).

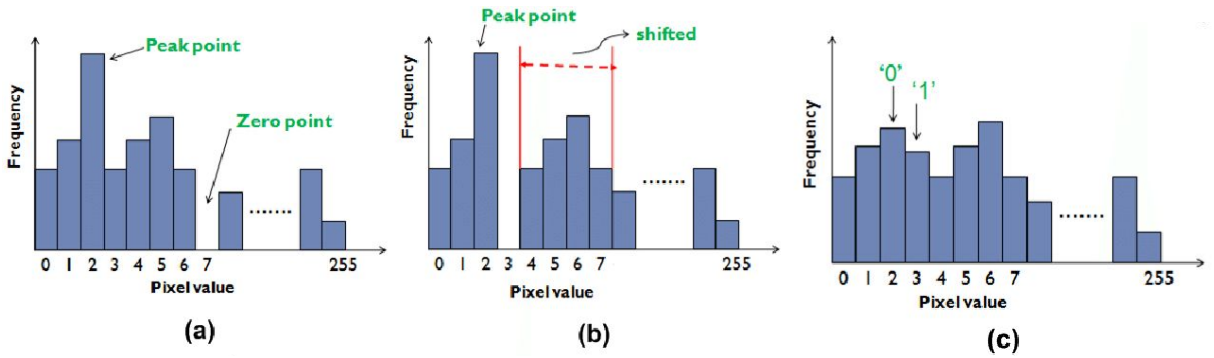


Figure 2.9: *Overview of a histogram shift as illustrated in (95).*

While many methods for high capacity data hiding in images have been proposed over the years, fewer have been developed for 3D objects. In 2017, Itier and Puech proposed a high capacity data hiding method with the use of a Hamiltonian path (99). An example of this method is illustrated in Fig. 2.10, where Fig. 2.10a presents the original 3D object which consists of 1002 vertices, Fig. 2.10b presents the associated point cloud, Fig. 2.10c the original Hamiltonian path, Fig. 2.10d the watermarked Hamiltonian path, Fig. 2.10e the watermarked 3D object, and finally Fig. 2.10f presents the RMSE between the original 3D object (Fig. 2.10a) and the watermarked 3D object (Fig. 2.10e). Then, in 2018, Jiang *et al.* described a reversible data hiding method for 3D objects where the

3D object is divided into an embedding set and a reference set and reconstructed using an optimal 3D prediction-error histogram (100). Also in 2018, Zhang *et al.* proposed a reversible data hiding method for 3D objects based on prediction-error expansion and sorting, where cells of the 3D object are sorted according to smoothness in order to find the best location to embed data (101).

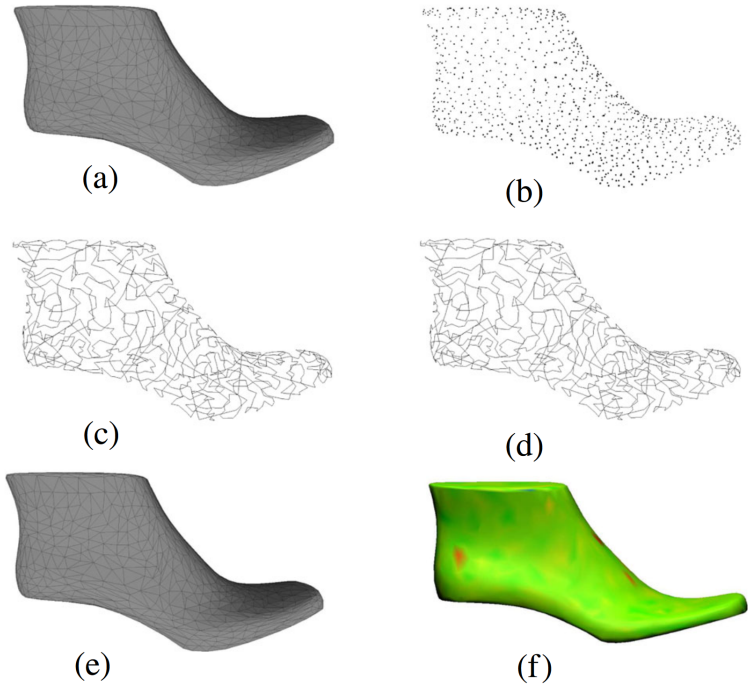


Figure 2.10: *An example of the high capacity data hiding method presented in (99) where a) The original 3D object, b) The associated point cloud, c) The Hamiltonian path, d) The watermarked Hamiltonian path, e) The watermarked 3D object, f) The RMSE between the original 3D object and the watermarked 3D object.*

2.6 Conclusion

In this chapter, we presented a state of the art of multimedia data hiding, in particular for images and 3D objects. We first presented an overview of the fundamentals of data hiding, where we detailed the embedding and extraction phases of data hiding. We then discussed the different criteria for data hiding as well as an overview of the different categories of data hiding. We detailed the different data hiding techniques and different evaluations of the results of data hiding methods.

We then established a state of the art of steganography, which we illustrated with the prisoners' problem. We detailed a brief history of steganography and then discussed image and 3D steganography. We then presented steganography's complementary domain, steganalysis. We then presented a state of the art of watermarking, where we detailed both robust watermarking and fragile watermarking for images and 3D objects, as well as their applications. Finally, we detailed a state of the art of image and 3D object high capacity data hiding, where we examined the notion of reversible data hiding in images and 3D objects.

While many methods have been proposed for image data hiding over the years, fewer have been proposed for 3D object data hiding. 3D object data hiding is generally more challenging than image data hiding due to the causality problem. In some types of data hiding, such as fingerprinting, methods developed for 3D objects are almost non-existent.

In Chapter 4, we present our proposed method for high capacity data hiding in the encrypted domain for 3D objects. In Chapter 6, we present our method for data hiding in the compressed domain for 3D objects. We note that a state of the art of data hiding in the encrypted domain and the compressed domain will be presented in Chapter 3.

CHAPTER 3

JOINT MULTIMEDIA ENCODING

Chapter Contents

3.1 Introduction

Online storage and sharing has become an integral part of everyday life, in both private and professional contexts. However, when storing and sharing multimedia on the cloud or over networks, it becomes susceptible to theft. In certain industries, such as the fashion or entertainment industries, these 3D objects are considered as important assets, and their loss or theft can result in a great financial loss, or a leak of trade secrets. In some cases, such as in healthcare, these 3D objects represent patient information, and their loss or theft could result in the sharing of private medical information. It is therefore essential that 3D objects are secured when they are stored on the cloud or transferred over networks.

There are two main categories of methods for securing 3D objects. The first is multimedia encryption, which consists of protecting the visual confidentiality of multimedia by rendering it illegible with the use of a secret key, as presented in Chapter 1. The second is multimedia data hiding, presented in Chapter 2, which aims to embed a secret message in the 3D object in an invisible way. Encryption is generally used when the content of the 3D object needs to be confidential, whereas data hiding is generally used to ensure the 3D object's integrity.

However, compression may be needed too, due to the generally high resolution of the multimedia being stored and shared. However, it is not as simple as performing different techniques one after the other, as generally, they both modify the same domains. A solution to this problem is joint multimedia encoding, where different techniques, such as encryption and compression, encryption and data hiding, or compression and data hiding, are performed simultaneously.

In this chapter, we establish a state of the art of joint multimedia encoding. First, in Section 3.2, we examine the compressed domain. We first present the JPEG image compression method, which is the industry standard image compression, video coding, and the Draco 3D object compression method, which is rapidly becoming the industry standard for 3D object compression. Then, in Section 3.3, we present a state of the art of joint compression and security, such as data hiding in the compressed domain, as well as crypto-compression. Then, in Section 3.4, we examine the encrypted domain. We present data hiding in the encrypted domain (DH-ED). We first establish a criteria for DH-ED. Then, we present a state of the art of DH-ED methods for images and in particular, homomorphic based DH-ED methods for images. We then present a comprehensive state of the art of DH-ED for methods for 3D objects. Finally, in Section 3.5, we conclude this chapter.

3.2 Compressed Domain

With the increasing popularity of social media over the last two decades, multimedia sharing has become an essential part of everyday life. In professional contexts, images and 3D objects are considered as important assets and are often stored on the cloud and shared over networks many times during their existence. However, these images

and 3D objects are often of very high resolution and can therefore be very large. This is particularly true for 3D objects, as they are often composed of millions of vertices. For example, a 3D object which has not been compressed composed of about 3 million vertices and 4.4 million faces has a size of about 528 MB. Therefore their online storage and sharing can be very expensive, time consuming and not environmentally friendly. A solution to this is multimedia compression.

3.2.1 JPEG Image Compression

The standard for image compression is the joint photographic expert group (JPEG) compression method, which was proposed by Wallace in 1991 (102). The JPEG image compression method is based on the principle that the human visual system (HVS) is much more sensitive to the luminosity value of a pixel, rather than its chromatic value. The HVS is also much more sensitive to changes in the low frequency domain of an image, as opposed to changes in the high frequency domain.

Encoding

Fig. 3.1 presents an overview of the encoding process of the JPEG image compression method. First, the color space of the original image, typically an RGB image composed of 8 bits per pixel per component, is converted to the YCrCb color space in order to isolate the image's luminosity, which is contained in the Y component. The two remaining color components, the Cr and Cb components which are the chrominance components, are then downsampled as the HVS is not very sensitive to these components. The image is then decomposed into square blocks of 8×8 pixels, where each block is then processed separately. The pixel values of each block are then shifted so that their range is centered on zero.

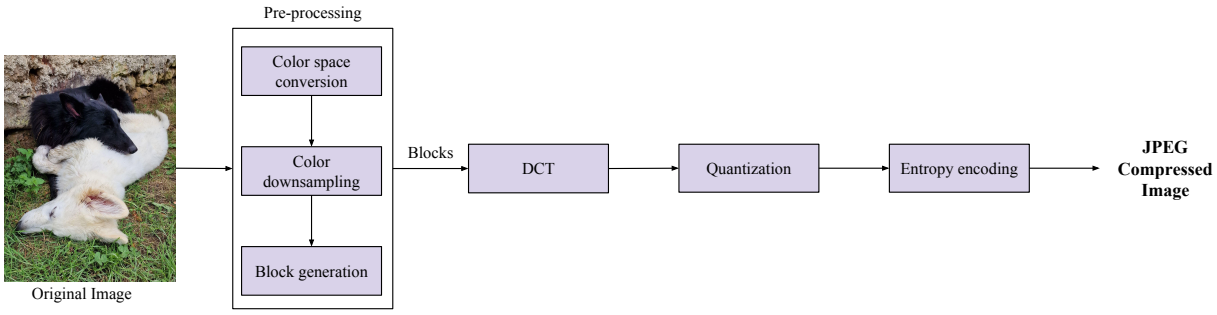


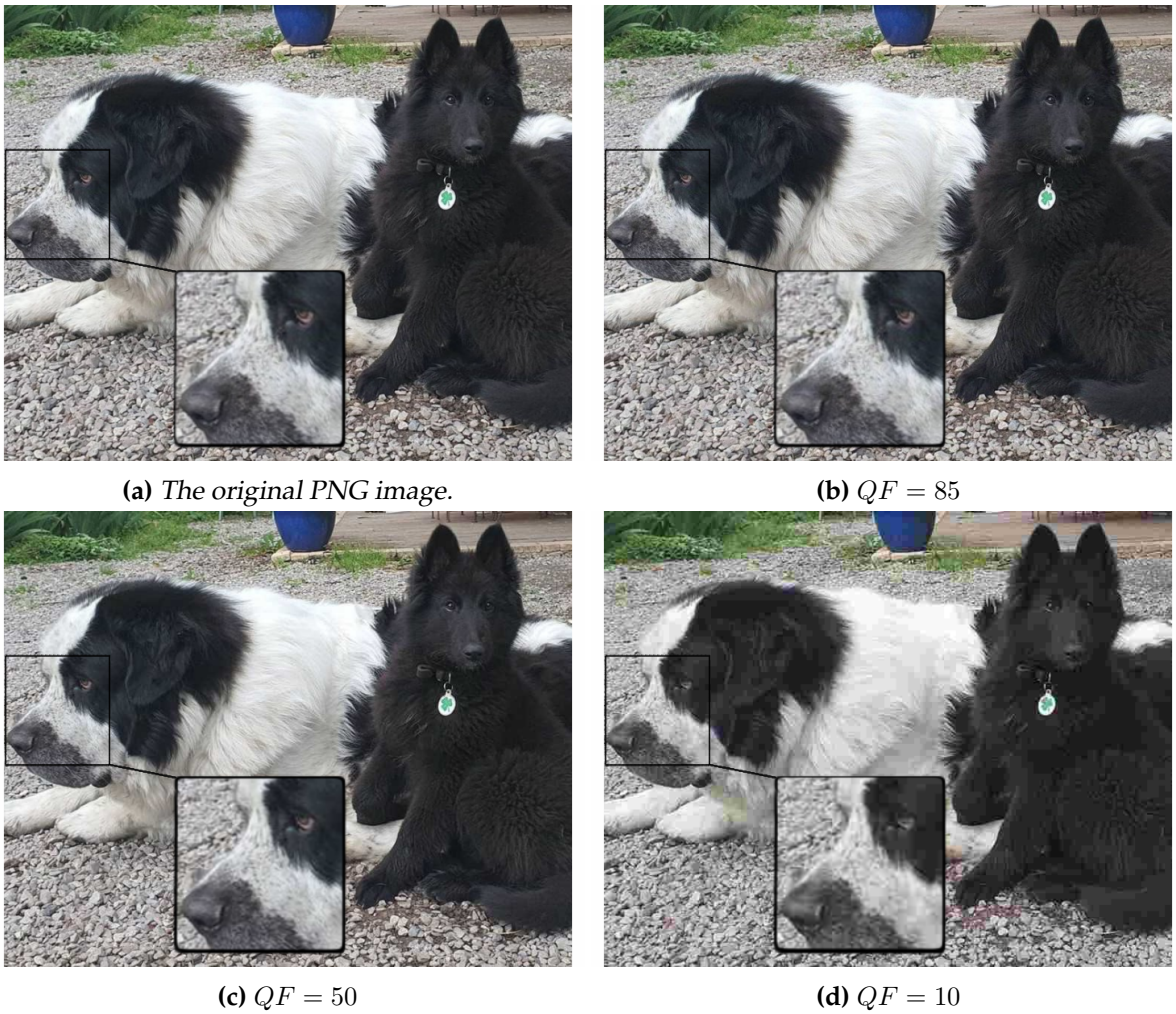
Figure 3.1: *Overview of the encoding process of the JPEG image compression method.*

Each block then undergoes a discrete cosine transform (DCT), which transforms the block from an 8×8 spatial domain matrix to an 8×8 frequency domain matrix F. There are two different types of frequency domain coefficients. The first coefficient $F(0,0)$ of the 8×8 frequency matrix is the low frequency DC coefficient, which is proportional to the average value of the original pixel block. The other 63 values of F are the high frequency AC coefficients. As the HVS is less sensitive to the AC coefficients, these coefficients are then quantized according to an 8×8 quantization table Q_{QF}. We note

that the DC coefficient is also quantized, according to $Q_{QF}(0,0)$, however the quantization is less significant. The parameter QF is the quality factor, which determines the standardized quantization table used. We note that the greater the quality factor $QF \in [0,100]$, the lower the coefficients of the quantization table Q_{QF}. In the case where QF has the maximum value of 100, then all coefficients of Q_{100} have the value of 1. The coefficients are quantized with:

$$F^Q(u,v) = round(\frac{F(u,v)}{Q_{QF}(u,v)}), \tag{3.1}$$

where F^Q is the quantized frequency matrix, $F(0,0)$ the DC coefficient and $F(u,v)$ the AC coefficient, where $u,v \in [0,7]$ and $u,v \neq (0,0)$, and $round$ a function which rounds the value to the nearest integer.



(a) *The original PNG image.* **(b)** $QF = 85$

(c) $QF = 50$ **(d)** $QF = 10$

Figure 3.2: *Example of JPEG according to different values of the parameter QF where a) The original PNG image, b) $QF = 85$, c) $QF = 50$, d) $QF = 10$.*

We note that choice of the quality factor QF is a trade-off between the visual quality and the compression rate of the the resulting JPEG compressed image. A high value of QF results in a high quality reconstructed image, however the compression rate is lower, whereas a low value of QF results in a high compression rate, however artifacts in the form of blocks appear in the reconstructed JPEG image. Fig. 3.2 illustrates an example of different values of QF. We note that while $QF = 85$ (Fig. 3.2b) has a

quality similar to that of the original image (Fig. 3.2a), artifacts are present for $QF = 50$ (Fig. 3.2c). Many artifacts are present for the very low value of $QF = 10$ (Fig. 3.2d).

The quantized coefficients are then mapped to a vector according to a zig-zag sequence illustrated in Fig. 3.3. This vector then undergoes an entropy encoding step according to a pre-defined standardized Huffman encoding table, which results in the JPEG compressed image.

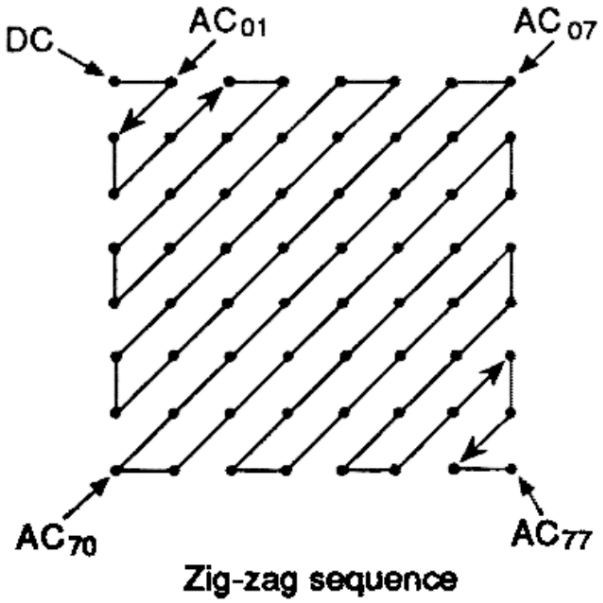


Figure 3.3: *An illustration of the JPEG zig-zag sequence, as presented in (102).*

Decoding

The JPEG decoding process, presented in Fig. 3.4, performs the inverse steps of the JPEG encoding process. First, an entropy decoding step is performed on each block. The AC coefficients of each of these blocks is then dequantized with:

$$F'^{Q}(u,v) = F^{Q}(u,v) \times Q_{QF}(u,v), \tag{3.2}$$

where F'^{Q} is the reconstructed dequantized frequency matrix, $F(0,0)$ the DC coefficient and $F(u,v)$ the AC coefficient, where $u,v \in [0,7]$ and $u,v \neq (0,0)$.

We note that the original AC coefficients cannot be reconstructed losslessly, even in the case where $QF = 100$ due to the integer rounding operation $round()$ in Eq. 3.1. The inverse DCT is performed, which transforms the frequency domain blocks into spatial domain blocks. These blocks are then recombined into a single pixel matrix which forms the reconstructed image. The Cr and Cb components are then upsampled, and then the image is reconverted from the YCrCb color space, back to the RGB color space. We note that like the dequantization step, the Cr and Cb components cannot be upsampled losslessly.

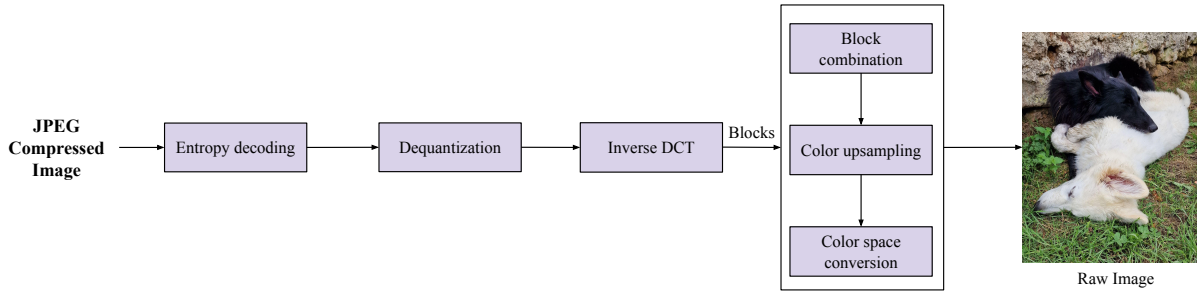


Figure 3.4: *Overview of the JPEG decoding process.*

JPEG2000

In 2000, Cristopoulos *et al.* presented the image compression method JPEG 2000 (103), which was designed to be the successor of JPEG. This image compression method is based on a discrete wavelet transform (DWT) and also has a higher compression rate and greater reconstruction quality than JPEG. However, because of a more complex file type, JPEG2000 is much less popular than the original JPEG. Therefore, today JPEG2000 is used in for certain applications, such encoding the frames in professional films displayed in movie theaters. We note that JPEG2000 was designed to be lossless or lossy, depending on the parameters.

3.2.2 Video

Videos are a form of image compression, as a video is an ordered sequence of images which are compressed in relation to one another. Videos are thus composed of sequences of multiple images per second, called frames. The video coding method, the high efficiency video coding (HEVC) method, also known as H.265, was developed in collaboration by multiple global organizations, and was standardized in 2013. This is the successor to the video coding method, the advanced video coding (AVC), also known as H.264.

In these video coding methods, frames are first divided into macroblocks. These macroblocks are then predicted according to other macroblocks. Frames can be classified into three categories:

- **Intra frames (I-frames)**: do not require any other frames in order to be decoded. These frames contain either the original macroblocks, or macroblocks predicted according to a macroblock in the same frame, known as an intra-macroblock.
- **Predicted frames (P-frames)**: require the previous frame in order to be decoded. These frames can contain both image information and vector motion information. Macroblocks in P-frames can be predicted according to the previous frame.
- **Bidirectional frames (B-frames)**: require both previous and subsequent frame in order to be decoded. These frames can contain information predicted from the previous frame, the subsequent frame and the current frame. We note that unlike P and I frames, not all video coding methods contain B-frames.

3.2.3 Draco 3D Object Compression

Over the past few years, multiple solutions for 3D compression have been proposed. In 2017, Dong *et al.* described a progressive compression algorithm that uses the 3D object's attributes (104). In 2019, Doumanoglou *et al.* detailed a comparison of real-time 3D compression methods (105). Then, in 2020, Liu *et al.* described a comparison of 3D point cloud compression methods (106). Recently, in 2021, Que *et al.* proposed an octree-based deep learning framework for point cloud compression (107).

Most notably in 2014, Google released their 3D compression method Draco, which is rapidly becoming the industry standard for 3D object compression (108). Fig. 3.5 presents the main steps of the encoding phase of the Draco 3D object compression method. Draco is separated into two main phases which are performed in parallel. These two phases are the connectivity encoding phase and the geometry encoding phase. The connectivity encoding phase is based on the 3D compression method Edgebreaker (109), whereas the geometry encoding phase is based on encoding the prediction errors after a vertex prediction step. We note that Draco is rapidly becoming the new industry standard for 3D compression. In our work, we are interested in the geometry encoding and decoding phase.

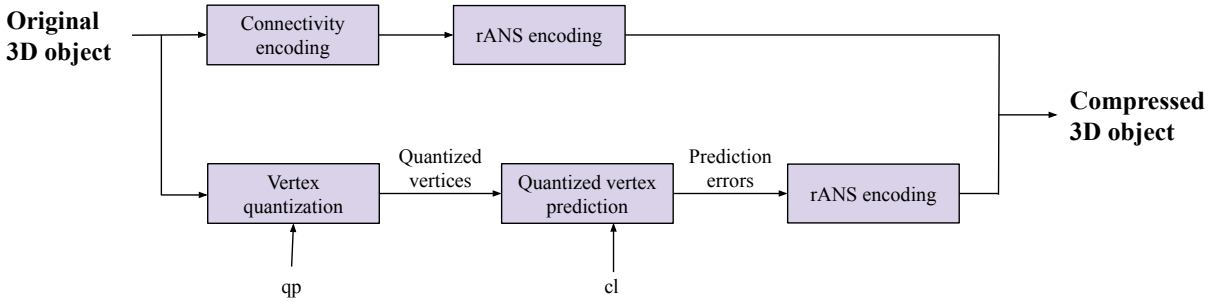


Figure 3.5: *Overview of the Draco 3D object encoding phase.*

We note that Draco is a very large system, with many different parameters. Certain options in the Draco encoding and decoding phases are chosen automatically by Draco depending on the 3D object's topology or number of vertices. Draco is also capable of compressing other characteristics of the 3D object such as textures, vertex normals, among others. In this work, we present an overview of the main steps of the encoding and decoding phase for the geometry and connectivity of a 3D object.

Edgebreaker

The Draco 3D object connectivity encoding phase is mainly based on the 3D object compression method Edgebreaker, proposed by Rossignac in 1999 (109). The Edgebreaker method encodes the 3D object's connectivity by encoding each triangle of a manifold 3D object with less than 2 bits.

The Edgebreaker method consists of traversing the connectivity of the 3D object step-by-step by means of a depth-first spiraling triangle spanning tree. At each step, a new triangle is visited and encoded according to which of its neighboring triangles

have already been visited. The current triangle is then encoded with a single character *C, L, E, R* or *S*, which is known as the *CLERS* string. These characters serve to describe how the current triangle can be reattached to set of already reconstructed triangles during the reconstruction, which takes place during the decoding phase.

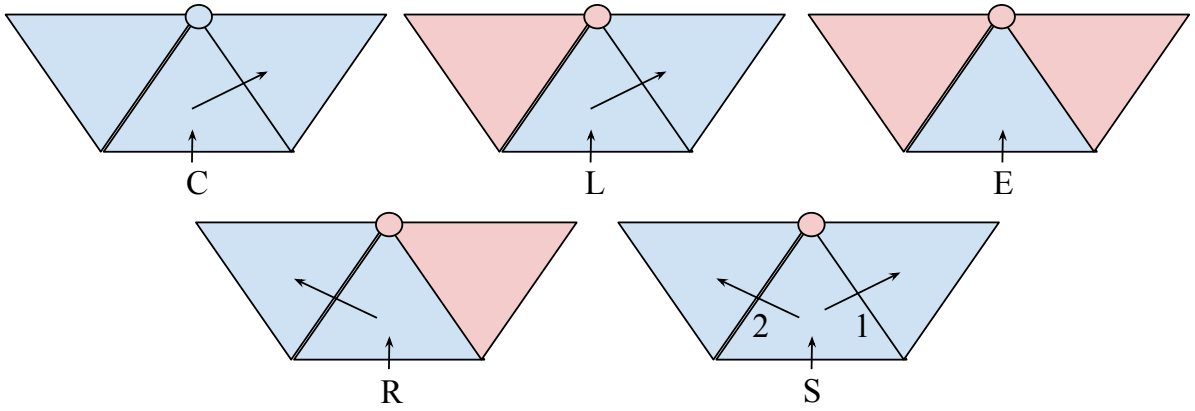


Figure 3.6: *The CLERS string allocation for the Edgebreaker compression method.*

Fig. 3.6 illustrates the *CLERS* string allocation, where the triangles or vertices in blue are not yet visited, while those in red have been visited, and the arrow indicates the next triangle to visit. There are five different scenarios possible for each triangle:

- **C:** the vertex is not yet visited and we move to the right.
- **L:** the left triangle is visited and we move to the right.
- **E:** both triangles are visited, the current triangle is labeled 'E' and we end the current loop.
- **R:** the right triangle is visited and we move to the left.
- **S:** neither neighboring triangles are visited and we move to the right first and then recursively to the left.

Vertex Quantization

During the geometry encoding phase, the vertices are first quantized according to the Draco quantization parameter $qp \in [0, 30]$, where qp is defines the number of bits conserved per coordinate during the quantization process, except for the special case $qp = 0$ which signifies that there is no quantization. Each coordinate c of each vertex v is transformed from a 32-bit floating point to an unsigned integer c' of qp bits:

$$c' = (c - c_{min}) \times \frac{2^{qp}}{range}, \tag{3.3}$$

where c is the the original floating point x, y or z coordinate, c' is the corresponding quantized coordinate, c_{min} is the minimum corresponding x, y or z coordinate, and

$range$ is the greatest edge of the bounding box. Consequently, the quantized coordinate $c' \in [0, 2^{qp}]$.

For example, if $qp = 15$, then during the quantization phase, each 32-bit floating point coordinate of the 3D object will be transformed into a 15 bit unsigned integer. The quantization parameter qp is one of the two main parameters for the Draco 3D object compression method. We note that the value of qp is a trade-off between the compression rate and visual quality of the decoded 3D object, as the vertex quantization is the only lossy process in the geometry encoding step of Draco. If a low value of qp is chosen, then fewer bits per coordinate are conserved, and so the quality of the reconstructed 3D object diminishes, however the compression rate increases. The value of qp is set by the user, however the default value recommended by Google is $qp = 11$, as there is no visual degradation.

Fig. 3.7 illustrates the effects of different values of qp on the 3D object *Bunny* (Fig. 3.7a), taken from the Stanford dataset (34). We observe in Fig. 3.7b that there is a large degradation when $qp = 7$ which has an RMSE of 6.160×10^{-4}, whereas there is no visible degradation when $qp = 11$ which has an RMSE of 0.380×10^{-4} (Fig. 3.7c), which is the recommended value for qp, or for $qp = 20$ which has an RMSE of 7.476×10^{-8} (Fig. 3.7d).

During the decoding process, each floating point coordinate c'' of each vertex can then be reconstructed using the inverse of Eq. 3.3:

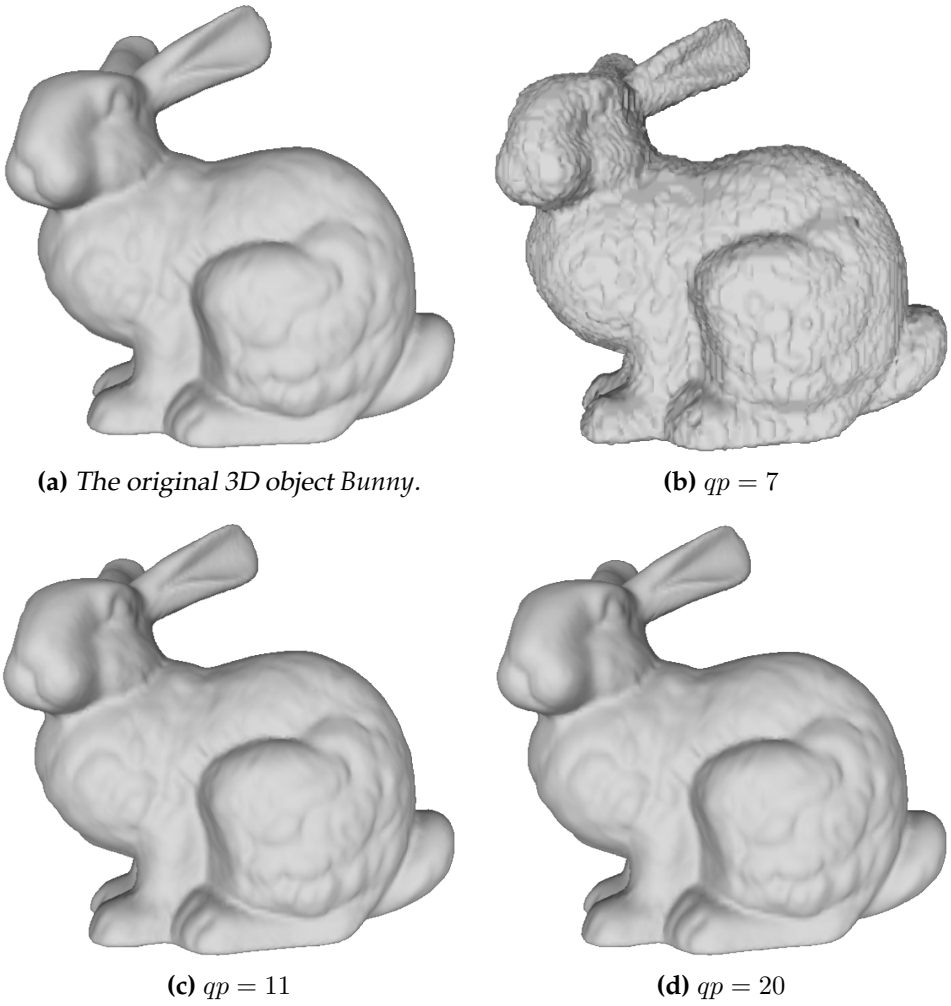
$$c'' = c' \times \frac{range}{2^{qp}} + c_{min}, \tag{3.4}$$

where c'' is the the reconstructed floating point x, y or z coordinate, c' is the corresponding quantized coordinate, c_{min} is the minimum corresponding x, y or z coordinate, and $range$ is the size of the greatest edge of the bounding box of the 3D object.

Vertex Prediction

The quantized vertices are then subjected to a vertex prediction step. Consequently, only the prediction errors are conserved. Fig. 3.8 presents the general framework of the vertex prediction step, where v_i is the vertex to be predicted according to previously constructed vertices v_j and v_k ($j \leq i$ and $k \leq i$), v_{ip} is the predicted vertex, and p_{error} is the resulting prediction error which is conserved and subsequently encoded by Draco.

This vertex prediction step relies on the compression level parameter cl, where $cl \in [0, 10]$. The value $cl = 0$ corresponds to no compression, and $cl = 10$ to the highest level of compression. Depending on the compression level cl, the vertex prediction method used is either a delta prediction, a parallelogram prediction, or a constrained multi-parallelogram prediction. We note that the value of cl is a trade-off between the compression rate and processing time. The default value recommended by Google is $cl = 7$.



(a) *The original 3D object Bunny.* **(b)** $qp = 7$

(c) $qp = 11$ **(d)** $qp = 20$

Figure 3.7: *The original 3D object Bunny, as well as the corresponding 3D objects decoded according to different values of the parameter* qp.

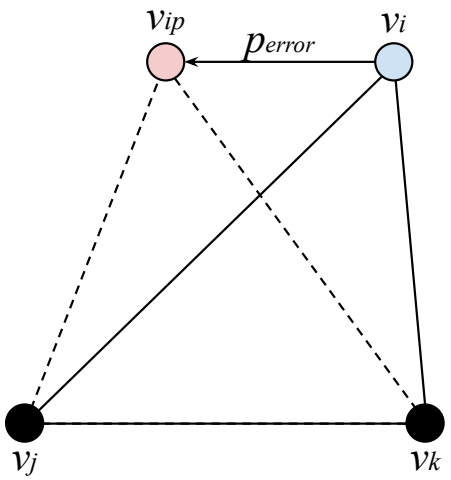


Figure 3.8: *General framework of the vertex prediction step.*

Entropy Encoding

After the geometry encoding step and the connectivity encoding step, both the encoded geometry and the encoded connectivity undergo an entropy encoding step sep-

arately. The Asymmetric Numeral System (ANS) encoding is an entropy encoding scheme proposed by Duda *et al.* in 2015 (110). In Draco, the range variant of ANS (rANS) encoding, also described in (110), is performed.

The Huffman encoding method and the arithmetic encoding method are the two most common entropy encoding methods. While Huffman encoding has the advantage of a low complexity, the compression rate is often suboptimal, whereas arithmetic encoding has a more optimal encoding rate, but it also has a greater complexity. The rANS entropy encoding method has the advantages of both the Huffman and the arithmetic encoding methods, as it has a more optimal encoding rate with a low complexity.

We define an alphabet $\mathcal{A}$, where $|\mathcal{A}| = l$ and $a_i \in \mathcal{A}$ where $i \in [0, l-1]$. We define s as a symbol in the sequence $\mathcal{S}$ to be encoded by $x \in \mathbb{N}$. The concept of ANS is to encode the symbols of $\mathcal{S}$ according to the distribution their probabilities. We note f_{a_i} as the frequency of a_i. The alphabet $\mathcal{A}$ is chosen so that $\sum_{i<l} f_{a_i} = 2^n$.

If s_t is the current symbol to be encoded, x_t is the current state, and f_{s_t} is the frequency of s_t, then the rANS encoding $C(\cdot)$ is performed with:

$$C(s_t, x_{t+1}) = \lfloor \frac{x_t}{f_{s_t}} \rfloor << n + c(s_t) + mod(x_t, f_{s_t}), \tag{3.5}$$

where $c(\cdot)$ is a variant of the cumulative distribution function:

$$c(a_i) = \sum_{j<i} f_{a_j}. \tag{3.6}$$

We note that contrary to a standard cumulative distribution function, $c(a_i)$ does not take into account f_{a_i}, which is the frequency of a_i.

During the rANS decoding process, the current symbol s_t and the new state x_t are retrieved at each step. First, s_t is extracted with:

$$s_t = c'(m), \tag{3.7}$$

where $a_i = c'(\cdot)$ is the inverse of $c(\cdot)$ such that:

$$c(a_i) < m < c(a_{i+1}), \tag{3.8}$$

and m is given by:

$$m = mod(x, 2^n). \tag{3.9}$$

The rANS decoding function is then given by:

$$D(x_{t-1}, s_t) = f_{s_t} \times (x_t >> n) + m - c(s_t). \tag{3.10}$$

Draco Decoding

Fig. 3.9 presents an overview of the Draco decoding phase. As during the encoding phase, the connectivity decoding and the geometry decoding are performed in parallel. We note that in this work, we are interested in the geometry decoding phase. The

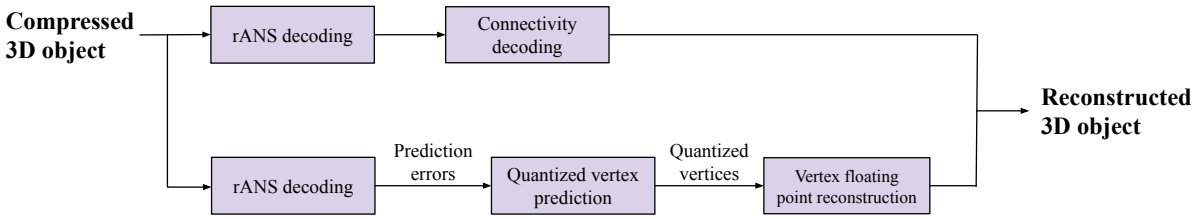


Figure 3.9: *An overview of the Draco decoding phase.*

compressed connectivity is decoded with rANS according to Eq. 3.10 and then reconstructed based on the Edgebreaker decoding method. During the geometry decoding process, the compressed geometry first undergoes a rANS decoding process in order to retrieve the vertex prediction errors. After the same vertex prediction step as during the encoding process is performed, the quantized vertices are retrieved. These qp bit quantized vertices are then reconstructed into 32-bit floating points according to Eq. 3.4. Consequently, the reconstructed 3D object is retrieved.

3.3 Joint Compression and Security

With the increasing quantity of multimedia stored online or transferred over networks, it is essential that multimedia is both compressed and secured. The solution to this problem is more complicated than first performing a compression step and then an encryption step, or first performing an encryption step then a compression step. If the multimedia is first compressed and then encrypted, the format compliance is lost as the compressed file can no longer be decoded by a viewer. However, if the multimedia is first encrypted and then compressed, the compression is no longer efficient, since compression methods use the redundancy in the multimedia in order to compress it, whereas encryption methods eliminate redundancy. If the multimedia is secured by means of data hiding, then the data hiding method needs to be robust against compression. In the case where the compression is performed first, then the data hiding method must respect the format of the compression method.

A solution to this compression and security problem is joint compression and encryption (crypto-compression), or joint compression and data hiding, where the compression and security are performed simultaneously. We note that in the case of 3D objects, joint compression and security is very challenging to achieve because they both produce changes in the same domains of the 3D object representation that interfere with each other. This is known as the causality problem.

3.3.1 Crypto-compression

During the last twenty years, several JPEG image crypto-compression methods have been proposed where an encryption step is integrated into the JPEG compression method. Van Droogenbroeck and Benedett suggested encrypting only the AC coefficients after the DCT transformation (111). Puech and Rodrigues proposed encrypting the DC co-

efficients and the lowest frequency AC coefficients (112). Gmira *et al.* proposed adding a dynamic Hill-Cipher encryption to the quantization step of JPEG compression (113). Hajji *et al.* (114) and Dridi *et al.* (115) both proposed crypto-compression schemes for images based on chaos.

Crypto-compression methods have also been developed for videos. Dufaux and Ebrahimi described two scrambling methods to hide private data in regions of interest (116). In 2011, Shahid *et al.* proposed an encryption method for H.264/AVC by selectively encrypting CAVLC and CABAC for P and I frames (117). Then, in 2017, Hamidouche *et al.* proposed an encryption scheme based on the chaos system in the scalable extension of HEVC (118).

While many crypto-compression methods have been proposed for images and videos, to the best of our knowledge, before our work, no methods for 3D object crypto-compression exist in the literature.

3.3.2 Data Hiding in the Compressed Domain

DH-ED allows data to be embedded in the support without revealing information about the content of the original support and therefore ensuring its visual confidentiality.

Many methods for data hiding in the compressed domain have been proposed for images, particularly for data hiding in JPEG images. In 2007, Xuan *et al.* proposed a reversible data hiding method in JPEG images based on histogram pairs (119). Then in 2015, Huang *et al.* presented a joint JPEG and histogram shifting based reversible data hiding method for images (120). Very recently, in 2023, Weng *et al.* detailed a reversible data hiding method for JPEG images based on adaptive 2D mappings for 2D histograms (121).

Some data hiding methods have been proposed for video. In 1997, Swanson *et al.* presented a data hiding method for video according to a perception based projection and quantization (122). In 2010, Ma *et al.* detailed a data hiding method for H.264/AVC video where they address the problem of intra-frame distortion drift by exploiting paired coefficients of a DCT block. Recently, in 2021, Konyar *et al.* proposed a data hiding method for videos based on adaptive inverted LSB332 (123).

Over the years, very few methods for joint compression and data hiding for 3D objects have been proposed. In 2009, Abdallah *et al.* proposed a method a joint compression and watermarking method for 3D objects (124). They propose using a Laplacian spectral mesh compression and embedding data in the spectral coefficients of the sub-mesh. Then, in 2011, Lee *et al.* described a joint reversible watermarking method for progressive 3D object compression (125). In their proposed method, a watermark is embedded in each level of detail of the 3D object.

3.4 Encrypted Domain

When transferring multimedia secured by encryption, the environment or another third party who has no right to access the content of the multimedia may need to embed data in the encrypted multimedia. For example, a server may need to embed identification information, or in healthcare, an employee other than a certified medical professional may need to embed patient identification information in the encrypted private medical information. A solution to this is data hiding in the encrypted domain (DH-ED), where a third party or the environment can embed data in the secured multimedia without the need to access the content of the multimedia.

DH-ED methods can be broken down into two main categories, which are reserving room before encryption (RRBE) and vacating room after encryption (VRAE). In RRBE methods, the content owner liberates space for a future embedded message in the multimedia during a preprocessing step, before the multimedia is encrypted. While in VRAE methods, the multimedia is first encrypted by the owner and the data hider can then embed the message by modifying the encrypted media accordingly.

There are also two main categories of multimedia reconstruction processes for DH-ED methods. The first category seeks to reconstruct the original multimedia without the embedded message, whereas the second category reconstructs the multimedia marked with the hidden message which was embedded in the encrypted domain. While many methods for DH-ED for images fall into the second category, only a single DH-ED method for 3D objects, which will be presented in Section 3.4.3, falls into this category.

3.4.1 Criteria

DH-ED methods are subjected to the same evaluation criteria as encryption and data hiding methods. DH-ED methods are ideally format compliant and without message extraction errors. Concerning the security of the data hiding method, the same criteria as data hiding in the plaintext domain, which are the imperceptibility, the robustness, the capacity, the security and the complexity, as discussed in Chapter 2.2 are applied.

Like data hiding in the plaintext domain, DH-ED is a trade off between the embedding rate and the quality of the reconstructed multimedia. When the multimedia can be reconstructed with very little or no distortion, this method is considered to be a reversible DH-ED (RDH-ED) method. In this work, we are particularly interested in RDH-ED methods. We note that for images, a method is considered to be reversible if the reconstructed image has a PSNR ≥ 50 dB. However, for 3D objects, there is currently no precise standard for reversibility, and this notion is still in dispute. In this work, we consider a method to be reversible if the reconstructed 3D object has no distortion according to the HVS.

3.4.2 Reversible Data Hiding in the Encrypted Domain in Images

RDH-ED for images is a recent subject of interest for researchers. Puteaux *et al.* show in their survey of the first twelve years of RDH-ED, which they established in 2021, the evolution of RDH-ED (126). Fig. 3.10 shows the number of papers published per year according to (126).

While many methods for RDH-ED exist today, it was not until 2008 that Puech *et al.* proposed the first RDH-ED method (127). Their method is based on the analysis of the local standard deviation of the marked encrypted images. Then, in 2011, Zhang proposed a RDH-ED method based on the spatial correlation of natural images (128). In 2018, Puteaux and Puech proposed a high capacity RDH-ED method for images, illustrated in Fig. 3.11 (129). They use the high local correlation between a pixel and its neighbors in the plaintext domain in order predict the MSB values of a pixel based on the previously decrypted neighboring pixels. In order to correct any prediction errors, they use an error location binary map.

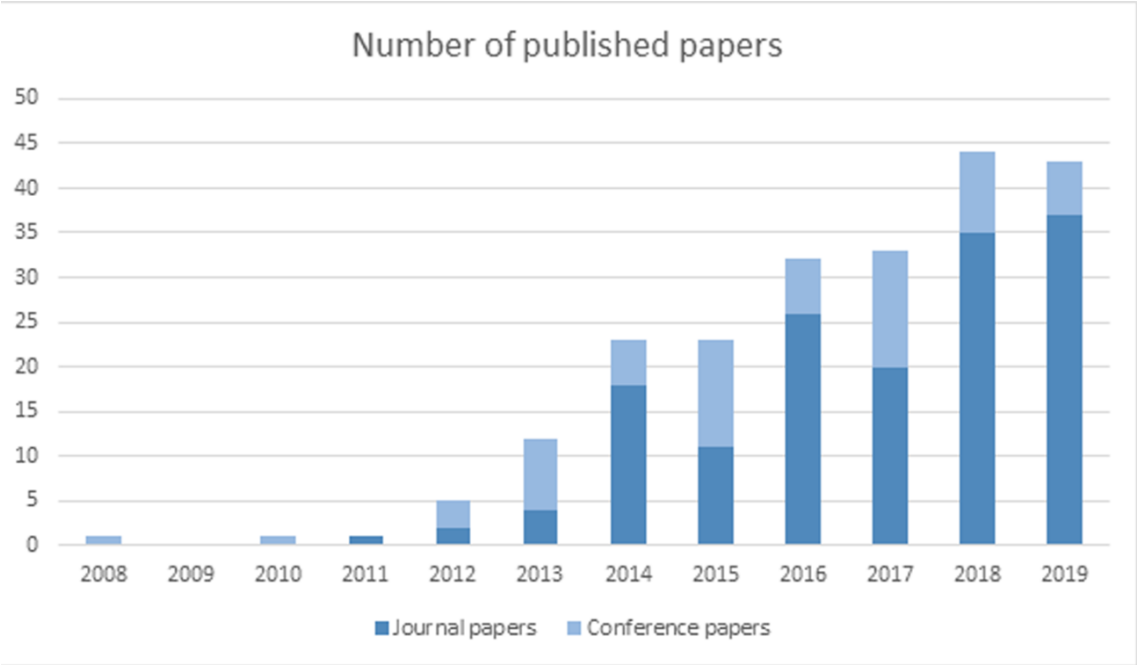
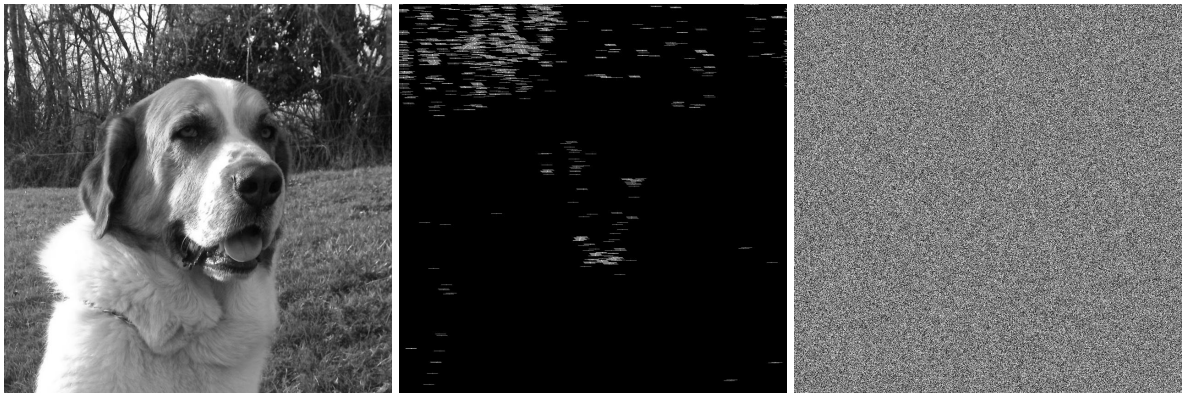


Figure 3.10: *Evolution of RDH-ED for images according to (126).*

Reversible Data Hiding in the Homomorphic Cryptosystems in Images

Several RDH-ED methods based on public key homomorphic cryptosystems have been proposed. These methods are based on either the Paillier cryptosystem (11) or cryptosystems involving the learning with errors (LWE) problem (130).

In 2014, Chen *et al.* were the first to propose a data hiding scheme based on the Paillier cryptosystem (131). Later, in 2018, Xiang and Luo described a method where an image is divided into sections for self-embedding before encryption (132). In 2019, Zheng *et al.* described a lossless, high-capacity data hiding method based on efficient



(a) *Original* **(b)** *Error Map* **(c)** *Encrypted*

Figure 3.11: *An illustration of (129) a) The original image, b) The corresponding error location binary map and c) The corresponding encrypted and marked image.*

mapping and use of expanded pixel values (133). Puteaux *et al.* proposed a high-capacity data hiding scheme in images that is based on a least significant bit (LSB) substitution (134). In fact, in this paper, Puteaux *et al.* perform a histogram shrinking function so that the pixel values are in the range $[0, n-1]$, where $n-1$ is the product of two integers. This is done in order to avoid pixel value overflows. Once the image is encrypted, there is a size expansion of 2.

3.4.3 Data hiding in the Encrypted Domain in 3D Objects

Just as DH-ED methods for images is a recent subject of interest for researchers, the same is particularly true concerning DH-ED methods for 3D objects. Until very recently, no methods existed for DH-ED for 3D objects. The first method was proposed by Jiang *et al.* in 2018 and since then, there has been a greatly increasing interest in this subject. DH-ED for 3D objects can be divided into three main categories of methods, which are vertex prediction based on partitioning, homomorphic cryptosystems, and 3D object subdivision.

Vertex Prediction Based on Partitioning

The first category of consists of partitioning the vertices of a 3D object into an embedding set and a reference set, where the reference set serves to predict the original bits of the coordinates of the embedding set. The majority of DH-ED methods for 3D objects is based upon this partition. The difference between these methods lies mainly in the manner in which the vertices of the 3D object are partitioned.

Fig. 3.12 presents a general framework for DH-ED methods for 3D objects based on vertex partitioning. First, the vertex coordinates are transformed from 32-bit floating points to integers of a variable size according to a quantization parameter set by the user. We note that this step is a quantization step. The vertices of the 3D object are then divided into an embedding set and a reference according to a criteria defined by the DH-ED method. We note that the main variation between these methods is

the manner in which the vertices are divided into these two sets. The vertices in the reference set are not used for embedding, but are used to predict the embedded bits of the embedding set of vertices, and consequently reconstruct these vertices, during the decoding process. We note that these methods generally require auxiliary information, and in some cases, a part of the payload is sacrificed in order to embed this auxiliary information, and in other cases, this auxiliary information is transferred in a separate auxiliary file.

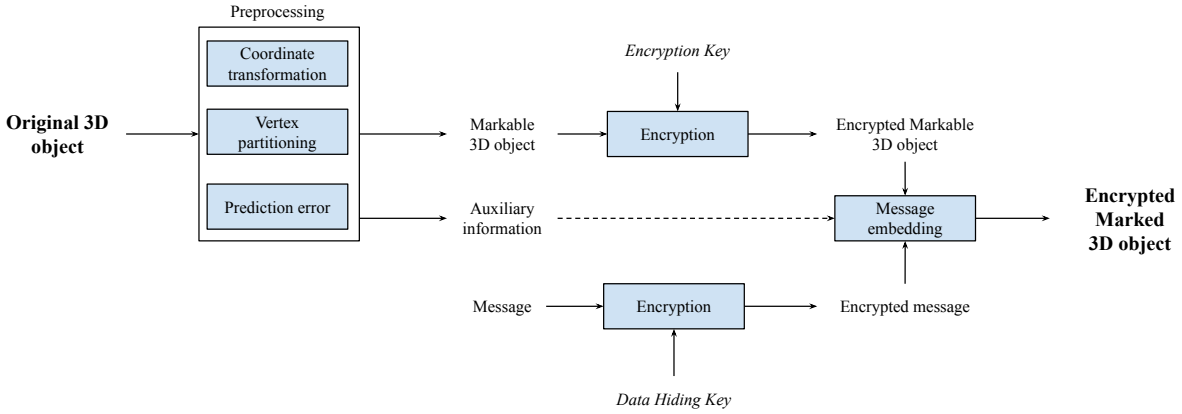


Figure 3.12: *The general framework for DH-ED methods for 3D objects based on vertex partitioning.*

Jiang *et al.*, who first proposed this method in 2018, divide the vertices by adding a vertex to the embedding set and its one-ring to the reference set (135). Multiple LSB of each coordinate are then used for embedding. This method has the disadvantage of a low payload, a distorted reconstructed 3D object and a high error rate when extracting the embedded data. Xu *et al.* then used the higher correlation between the MSB in the plaintext domain to perform a vertex prediction on the MSB instead of multiple LSB (136). This method has the disadvantage of a low payload and the use of an auxiliary file. Yin *et al.* improved upon these previous two methods in 2019 by embedding t MSB, where t is the minimum embedding capacity for all embeddable vertices (137). Very recently, in 2022, Tsai and Liu proposed randomly choosing a percentage of neighbors according a given threshold and basing the prediction on the center of gravity of the neighboring vertices (138). Also in 2022, Lyu *et al.* proposed optimizing the distribution between the embedding set and the prediction set by using the vertices' parity as the division criteria (139). The data is embedded by substituting t MSB of the embedding set, where t has a variable length. In 2023, Tang *et al.* then improved upon the method of Lyu *et al.* by adding another parameter in addition to the parity, which allows vertices to be transferred from the prediction set to the embedding set (140). The parity distribution of this method before the additional parameter is applied is illustrated in Fig. 3.13, where Fig. 3.13a presents the 3D object *Mushroom*, and Fig. 3.13b presents the corresponding parity distribution of the vertices.

The 3D object is then encrypted by performing an exclusive-or (XOR) on the quantized coordinates with the use of a pseudo-random bit stream which generated with the embedding key. The encrypted data is then embedded in the embedding set of vertices also by performing an XOR. We can also note that the payload depends on the 3D object's characteristics. During the decoding process, the embedded message is first extracted before the vertices are reconstructed using the reference set. Once the

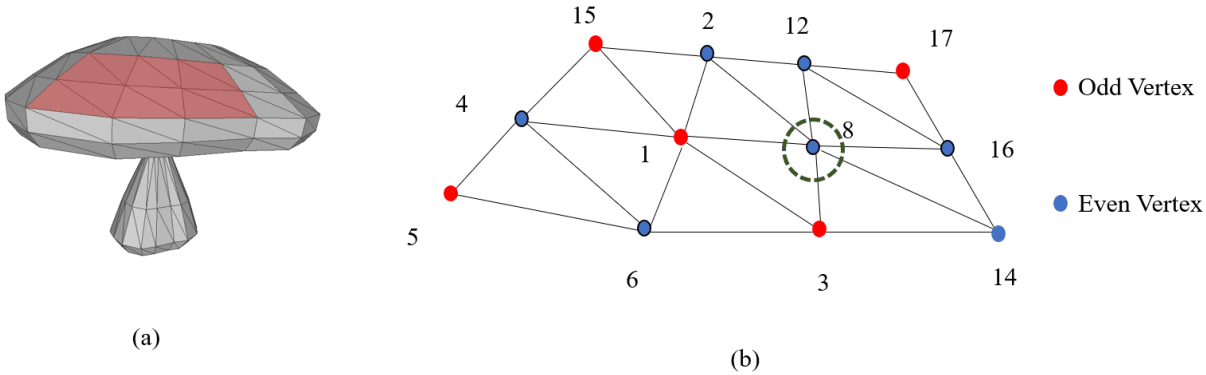


Figure 3.13: *The parity distribution of (140), with a) The 3D object Mushroom and b) The corresponding parity distribution of the vertices.*

3D object is decrypted, the reconstructed 3D object is no longer marked.

Homomorphic Cryptosystems

As presented in Chapter 1.2.3, the properties of homomorphic cryptosystems allow an operation in the plaintext domain to be transformed into the same or a different operation in the encrypted domain. Notably, the Paillier cryptosystem transforms a multiplication in the encrypted domain to an addition in the plaintext domain. However, there is a significant size expansion associated with the Paillier cryptosystem, as n in the plaintext domain becomes n^2 in the encrypted domain.

In 2018, Shah *et al.* proposed a DH-ED method for 3D objects using the Paillier cryptosystem (141). This method describes a two tier homomorphic DH-ED scheme, and to the best of our knowledge, is the only method based on a homomorphic cryptosystem before our contribution. The floating point vertex coordinates are first mapped to positive integers so the Paillier cryptosystem is able to process them. The 3D object is then encrypted using the Paillier cryptosystem. The first tier of data hiding is completed by using the Paillier cryptosystem's homomorphic properties to perform a histogram expansion and shifting in the encrypted domain. This results in a significant size expansion. The second tier data embedding is done by using the Paillier self-blinding property (Eq. 1.20). This is the only state-of-the-art method that preserves the embedded message once the 3D object is decrypted. We note that only the first tier of data hiding is preserved.

3D Object Subdivision

In 2020, Tsai proposed a DH-ED method for 3D objects based on spatial subdivision and space encoding of the 3D object (142). In this method, the vertices are represented as a ratio in relation to the 3D object's bounding box. After the 3D object is encrypted, it object then undergoes a spatial subdivision in the form of a grid, where each cell defines the limits of which a vertex can be displaced. The vertex is then displaced within the cell according to the bits of the message to be embedded. We note that

errors may occur during the embedded message extraction if the parameters are not chosen well.

3.5 Conclusion

In this chapter, we first examined the compressed domain. We presented the industry standard for image compression, the JPEG image compression method, video coding methods, as well as the industry standard for 3D object compression, the Draco compression method. We then presented a state of the art of joint compression and security for multimedia, including crypto-compression and data hiding in the compressed domain.

While many methods exist for image crypto-compression, to the best of our knowledge, no method exists for 3D object crypto-compression, or 3D object selective crypto-compression. In Chapter 6 of our contributions, we present the first 3D object crypto-compression method, as well as the first 3D object selective crypto-compression method. Both of these proposed methods is based on the Draco 3D object compression method.

While multiple methods for data hiding in JPEG images exist, very few methods exist for joint compression and data hiding in 3D objects. In Chapter 6 of our contributions, we also propose a joint compression and watermarking method for 3D objects, based on Draco. We are the first to propose such a method based on Draco, which is rapidly becoming the industry standard for 3D object compression.

In this chapter, we also examined the encrypted domain. We presented different criteria which needs to be taken into account, and then we presented a state of the art of RDH-ED in images. We then presented a detailed state of the art of DH-ED in 3D objects, where we described three main categories of methods.

Many DH-ED methods exist for images today, although this is a recent subject of interest for researchers. This subject of interest applied to 3D objects is even more recent, as the first method for 3D objects dates back to only 2018. Among these methods today, none of the reconstructed 3D objects remain marked with the high capacity embedded message once the 3D object is decrypted. Only Shah *et al.* are capable of reconstructing a marked 3D object once the 3D object is decrypted, however it is marked with a very low payload. In Chapter 4 of our contributions, we propose a DH-ED method which conserves the high capacity embedded message in the reconstructed 3D object after decryption. We note that this contribution is one of the first high capacity DH-ED methods for 3D objects.

Part II

Contributions

CHAPTER 4

3D DATA HIDING IN THE ENCRYPTED DOMAIN

Chapter Contents

4.1 Introduction

As described in Chapter 3, data hiding in the encrypted domain (DH-ED) methods for 3D objects is a very recent subject of interest for researchers, as the first paper dates back to only 2018 (100). In this chapter, we present in detail our proposed high capacity data hiding in the encrypted domain (HCDH-ED) method for 3D objects. Our method is based on the Paillier cryptosystem and uses its homomorphic properties (Eq. (1.19) and Eq. (1.20)) in order to embed messages in a homomorphically encrypted 3D object, without changing the connectivity of the 3D object. We present two variants of our proposed HCDH-ED method for 3D objects. We note that our first variant is considered to be one of the first DH-ED methods for 3D objects, preceded by only (100; 141; 137).

In this chapter, we present both variants of our proposed HCDH-ED method in Section 4.2. We then present our experimental results in Section 4.3. Finally, we conclude this chapter and present our perspectives in Section 4.4.

4.2 The Proposed HCDH-ED Method for 3D Objects

As presented in Chapter 3, in current state of the art methods, the messages which are embedded in the encrypted domain are lost once the 3D object is decrypted and reconstructed. Only Shah *et al.* conserve a part of the embedded message once the 3D object is decrypted and reconstructed, but the conserved payload has a low capacity of 3 bits per vertex (bpv). In this work, we present a HCDH-ED method which conserves a high capacity payload of 13.5 bpv once the 3D object is decrypted and reconstructed. We note that this payload is embedded in the encrypted domain. This is achieved with the use of the Paillier cryptosystem's homomorphic properties. The proposed HCDH-ED method is also format compliant, which implies that despite the disadvantage of the large size expansion associated with homomorphic cryptosystems, we avoid all size expansion. This is achieved by regrouping the 3D object's vertices into blocks whose size is determined by the desired key size. When compared to state of the art methods, our proposal is the only one to avoid size expansion, an auxiliary file and data errors which refer to errors in the retrieved message.

In Section 4.2.1, we first present an overview of the proposed HCDH-ED method and describe the differences between the two variants. Then, in Section 4.2.2, we analyze the necessary key size as a function of the desired block size. In Section 4.2.3, we describe the preprocessing step where the vertices are grouped into blocks. Then, in Section 4.2.4, we present our 3D object encryption method based on the Paillier cryptosystem. In Section 4.2.5, we present the data hiding step in the encrypted domain. Finally, in Section 4.2.6, we present how the 3D object is reconstructed and then how the embedded messages are extracted in the plaintext domain.

4.2.1 Overview

The main contributions of the proposed HCDH-ED method are summarized as follows:

1. The proposed method allows us to obtain a high capacity watermarked 3D object in the plaintext domain for which messages have been embedded in the encrypted domain. To the best of our knowledge, we are the only method that can achieve this;
2. The method is format compliant and there is no size expansion in the encrypted domain;
3. Very large key sizes can be used by grouping vertices into blocks;
4. In our first variant, a two tier data hiding is performed;
5. In our second variant, several messages can be embedded in the same encrypted 3D object. This process has no impact on the reconstruction.
6. No auxiliary information is required for message extraction in the plaintext domain.

We note that the main difference between the two variants is in the way the Paillier cryptosystem's self-blinding property is used. In the first variant, a second tier message can be embedded. Indeed, in the encrypted domain, we exploit the Paillier probabilistic property, which is known as the self-blinding property (Eq. (1.20)), in order to flag the used vertex blocks without impacting the reconstruction of the 3D object in the plaintext domain. In the second variant, we adapt the HCDH-ED method in order to integrate multi-message embedding. The Paillier cryptosystem's self-blinding property is used to flag the vertex blocks. Therefore, each time a new message is embedded in the encrypted 3D object, flags are used to indicate whether a vertex block has been watermarked, allowing for the construction of a binary location map.

Two Tier HCDH-ED Variant

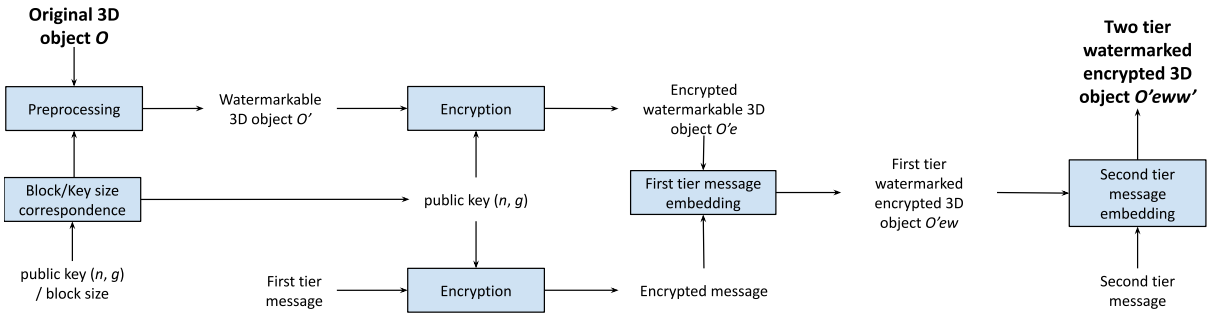


Figure 4.1: *Overview of the encoding phase of the proposed two tier HCDH-ED method for 3D objects.*

Fig. 4.1 presents an overview of the encoding phase of the two tier HCDH-ED variant. First, the 3D object undergoes a preprocessing step where the vertices are grouped into blocks, noted B, of size b vertices per block. The block size is directly proportional to the key size. Bits which we wish to use to embed the first tier message are designated according to the size b as well as the payload per block α. These designated bits are then set to zero since the embedded first tier message will be added to these bits during the embedding step. This embedding step is performed by first encrypting the first tier message with the same public key used for the 3D object encryption. Both the message and the 3D object are then multiplied in the encrypted domain. This is equivalent to an addition in the plaintext domain. The first tier watermarked encrypted 3D object can then be watermarked a second time with a smaller second tier message. We note that this second tier message exists only in the encrypted domain. With our proposed method, the visual quality of the watermarked 3D object in the plaintext domain is very high when compared to the corresponding original 3D object.

Multi-Message Embedding Variant

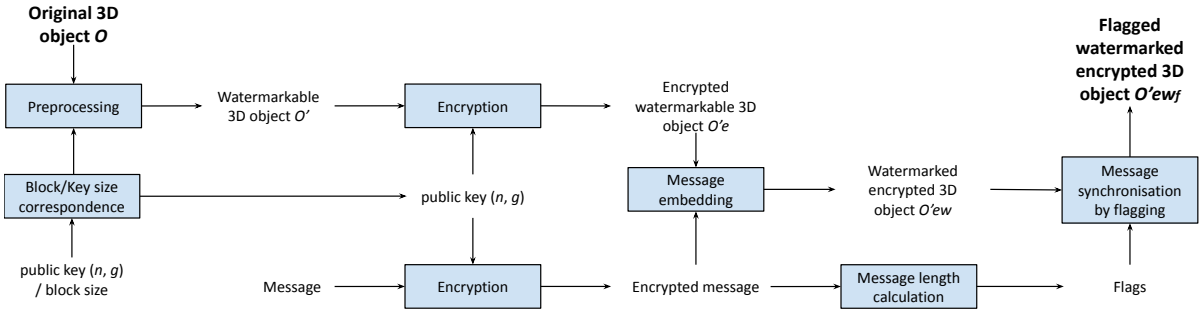


Figure 4.2: *Overview of the encoding phase of the proposed multi-message HCDH-ED method for 3D objects.*

We then improved upon the first variant by adapting our method to integrate multi-message embedding. Fig. 4.2 presents the overview of the encoding phase of this variant of the proposed method. We note that what was previously indicated as the first tier message in the first variant is now referred to as the message, as the second tier message in the first variant is now adapted to serve as flags. These flags allow us to synchronize a large number of messages, which in turn allows us to determine which blocks are already watermarked and to clearly separate each message. Indeed, with these flags, each time a new message is embedded in the encrypted 3D object, the location of this message is highlighted by adding flags. These flags indicate where additional messages can be placed in the 3D object, and allow the messages to be retrieved during the decryption process. This flag embedding process is fully reversible and has no impact on the decoding phase or on the decrypted 3D object.

4.2.2 Key Size Analysis

We consider each vertex block to have a size of $2k + 1$ bits. We note that the block size is determined by the size of the key. If k is the number of bits per block we want to

encrypt, then according to the constraints imposed by the Paillier cryptosystem, the value of n of the public key (n, g), should be represented with at least $k + 1$ bits. Therefore we have:

$$2^{2k} \leq n^2. \tag{4.1}$$

Due to the modulus n^2 in Eq. (1.17), the size of the encrypted data is at most n^2. In order to limit the size of the encrypted data to $2k + 1$, and consequently avoid size expansion, we impose the following constraint:

$$2^{2k} \leq n^2 < 2^{2k+1}. \tag{4.2}$$

Therefore, n is constrained by:

$$2^k \leq n < \sqrt{2} \cdot 2^k. \tag{4.3}$$

The relationship between n and the size b of a block B is deduced in Section 4.2.3.

4.2.3 Preprocessing

As presented in Chapter 1.4, we note that each vertex consists of three coordinates x, y and z, where each of which can be represented by a 32-bit floating point fp, which consists of a sign s (1 bit), an exponent e (8 bits) and a mantissa $mant$ (23 bits) where:

$$fp = (-1)^s \times mant \times 2^{e-127}. \tag{4.4}$$

Homomorphic cryptosystems cannot process floating point values due to the complexity of simple mathematical operations which are used in the encryption and data hiding processes. Therefore the encryption is performed exclusively on the mantissas. Additionally, encrypting only the mantissa allows the encrypted 3D object to remain format compliant. This does not compromise the security because the mantissa contains the most relevant information, while s and e contain mainly structural information. The 23 bits of the mantissa of each coordinate are transformed into an integer. This means that the part of each vertex v we want to encrypt is encoded with $23 \times 3 = 69$ bits.

In order to have a key sufficiently large to be secure, vertices are grouped into blocks B of size b vertices per block. Each block therefore consists of $69b$ bits. A block of vertices is then constructed by first grouping the MSB-0 of each vertex coordinate, then the MSB-1, until finally the LSB, as illustrated in Fig. 4.3. We note that due to the nature of the Paillier cryptosystem, the size of the block cannot exceed the size of the key. The size of the key is in turn limited by the complexity of the Paillier cryptosystem. The size of the block is therefore determined by the size of the key. Dividing the vertices

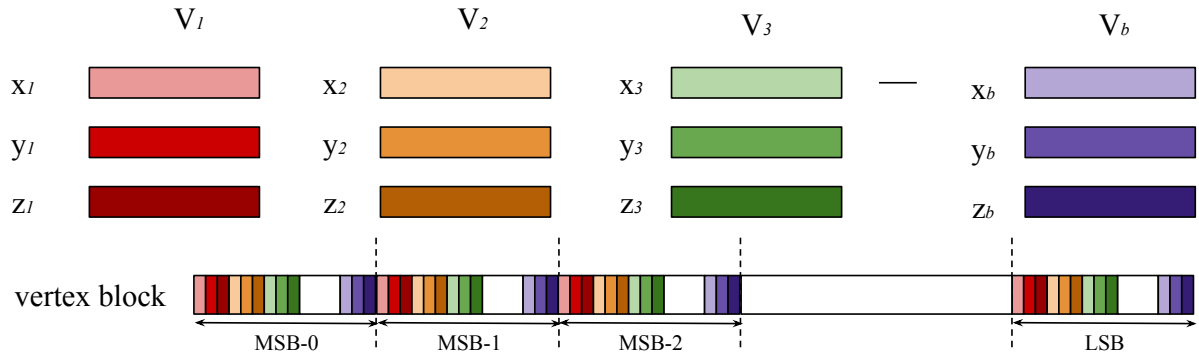


Figure 4.3: *Construction of a block B composed of b vertices.*

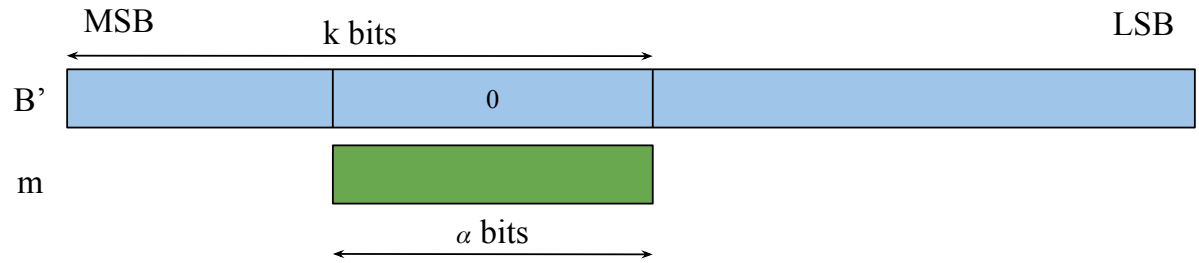


Figure 4.4: *Preprocessing of a vertex block B in the plaintext domain.*

into blocks allows the embedding of multiple messages, as each block can only contain a single message.

We note α the payload in bits per block. Each message to embed is divided into segments of size α bits. To avoid a bit overflow when we embed a segment of a message in a block B, as illustrated in Fig. 4.4, α bits of the block B are set to zero in the plaintext domain. If k is the number of bits to encrypt in a block B, then the α LSB among the k MSB are set to zero, as illustrated in Fig. 4.4. We note B' the watermarkable vertex block and O' the corresponding watermarkable 3D object.

4.2.4 Encryption

To avoid a size expansion of the encrypted vertex block in relation to the plaintext vertex block, we set the size of the encrypted vertex block $69b = 2k+1$ bits. This means that, as illustrated in Fig. 4.4, we should then encrypt:

$$k = \frac{69b - 1}{2} \text{ bits.} \tag{4.5}$$

We note that in order for $69b = 2k + 1$, then the block size b has to be odd.

To encrypt the k MSB of the block B', which we note $B'_{k_{MSB}}$, we use the Paillier homomorphic encryption function (Eq. (1.17)). We then obtain the $2k+1$ bits as illustrated in Fig. 4.5. The resulting $2k + 1$ encrypted bits substitute the bits of B'. The encrypted block B'_e is then divided into individual vertices in order to respect the original format

of the 3D object. We note B'_e the encrypted watermarkable vertex block and O'_e the corresponding encrypted watermarkable 3D object.

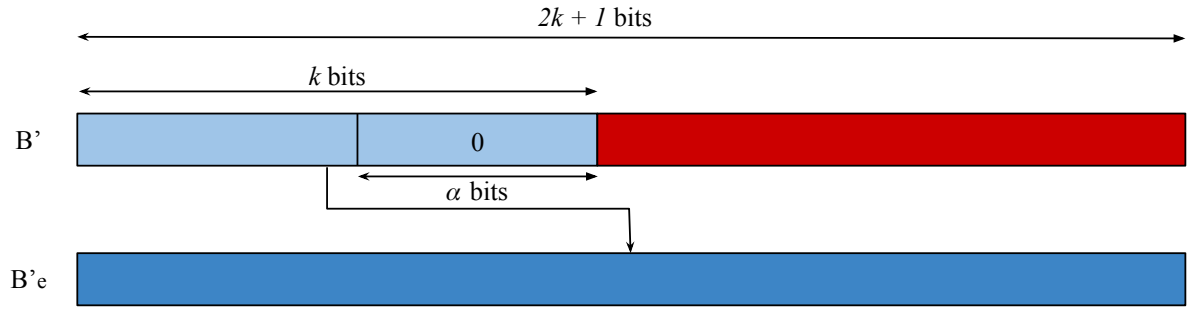


Figure 4.5: *Encryption process of a watermarkable vertex block B'.*

We note that the remaining $k+1$ LSB of B' are not included in the encryption step. They are then lost. As described in Chapter 1, according to an analysis proposed by Beugnon *et al.* (47), we assume that we can lose up to 16 LSB in the mantissa of each vertex coordinate, without visual degradation according to the human visual system (HVS). Just as images with a PSNR $\geq 50\ dB$ are considered reversible because there is no visual degradation according to the HVS, these 3D objects are considered to have no visual degradation because of their very small Hausdorff distances. This signifies that in each vertex there can be a loss of $3 \times 16b = 48b$ bits per block B before there is visual impact on the decrypted 3D object. Therefore, losing $k+1$ LSB is not a problem, since $k+1 < 48b$ according to Eq. (4.5).

4.2.5 Data Hiding in the Encrypted Domain

In this section, we describe the data hiding process for the messages embedded in the encrypted domain.

Message Embedding

In order to embed a message segment m in each block B'_e of the encrypted watermarkable 3D object O'_e, we use the Paillier additive homomorphic property of Eq. (1.19), which indicates that a multiplication in the encrypted domain is equivalent to an addition in the plaintext domain. We note that in the two tier reversible data hiding (HCDH-ED) variant, this message segment m is noted as the first tier message. Therefore, we embed the message segment m, in the encrypted block B'_e such that:

$$B'_{ew} = \mathcal{E}(B'_{k_{MSB}}) \times \mathcal{E}(m) \bmod n^2, \tag{4.6}$$

where B'_{ew} is the watermarked encrypted block, $\mathcal{E}(\cdot)$ is the Paillier encryption function and $\mathcal{E}(B'_{k_{MSB}}) = B'_e$.

We note O'_{ew} the corresponding watermarked encrypted 3D object. Note that since this multiplication in the encrypted domain is equivalent to an addition in the plaintext

domain, and since we have already cleared space for m by setting the α bits of the payload to 0, this operation is equivalent to an α LSB substitution in the plaintext domain. We can then reduce Eq. (1.19) to:

$$\mathcal{D}(\mathcal{E}(B'_{k_{MSB}}) \times \mathcal{E}(m) \bmod n^2) = B' + m. \tag{4.7}$$

As indicated in Section 4.2.4, Beugnon *et al.* show that we need to conserve at least $23 - 16 = 7$ useful bits per coordinate (u), which results in $3u = 21$ MSB per vertex (47). By respecting this, we do not compromise the visual quality of the decrypted 3D object. Therefore, α, the payload of a block B in bits is:

$$\begin{aligned} \alpha &= k - 3u \times b \\ &= k - u \times \frac{2k+1}{23} \text{ bits.} \end{aligned} \tag{4.8}$$

This results in a payload p, in bpv of:

$$\begin{aligned} p &= \frac{\alpha}{b} \\ &= \frac{k}{b} - 3u \\ &= \frac{69k}{2k+1} - 3u \; bpv. \end{aligned} \tag{4.9}$$

Fig. 4.6 shows the payload p in bpv as a function of the value of b. We observe that the curve quickly converges towards 13.5 bpv.

Second Tier Data Hiding

In the two tier HCDH-ED variant, we embed a second tier message m' which exists only in the encrypted domain. This means that when the 3D object is decrypted, this second tier message m' is lost. There are two different ways in which the second tier message m' can be embedded. Both of which use the Paillier probabilistic property, which is a property which indicates that the encrypted value of m is not unique.

If the second tier message m' is known when we embed the first tier message m, then when encrypting m, we choose r in Eq. (1.17) such that:

$$B'_{ew_f} \bmod 2^{nb} = m', \tag{4.10}$$

where nb is the number of bits used to encode m' and B'_{ew_f} the two tier watermarked encrypted block.

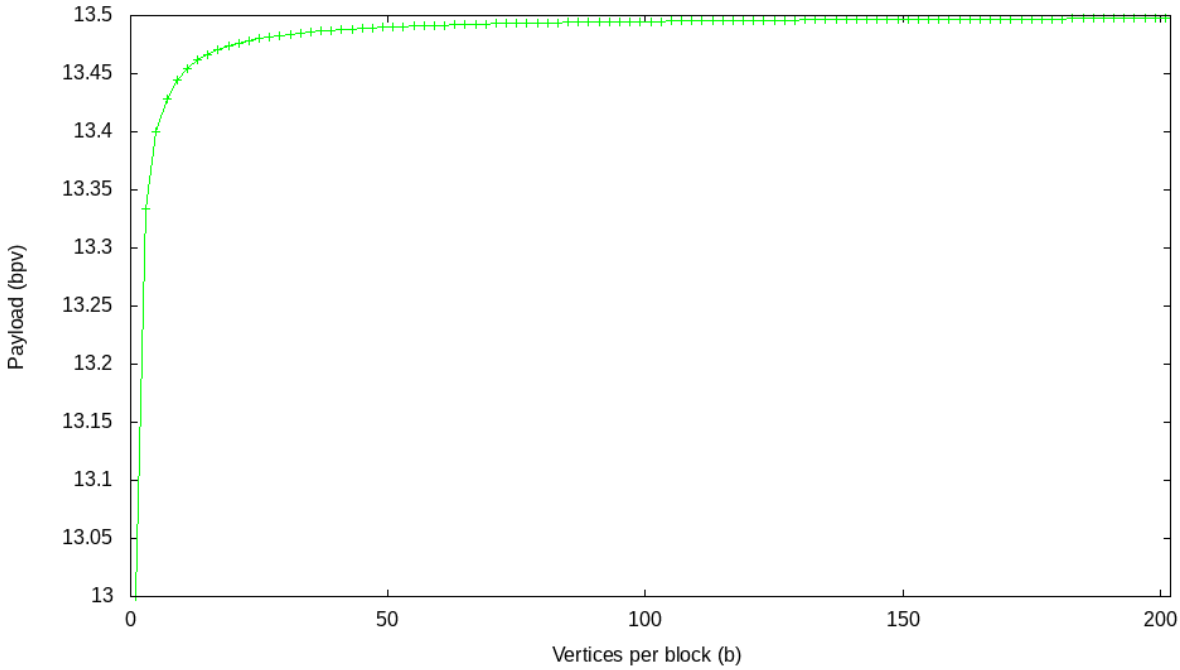


Figure 4.6: *Payload in bits per vertex as a function of the value of the block size b.*

The method developed by Shah *et al.* for a second tier data hiding can also be applied in our approach (141). In particular in the case of embedding m' without knowing its value when m is embedded. This method uses the Paillier cryptosystem's self-blinding property, Eq. (1.20), where we choose t relatively prime to n such that:

$$(B'_{ew} \times (t^n \bmod n^2) \bmod n^2) \bmod 2^{nb} = m'. \tag{4.11}$$

The complexity of both methods can be expressed by the probability of choosing the correct r and t respectively so that a modulus 2^{nb} results in m':

$$P(X = m') = \frac{1}{2^{nb}}. \tag{4.12}$$

Fig. 4.7 illustrates that the number of possible values which can be obtained by a modulus 2^{nb} is exponential. The choice of nb is then a trade-off between time complexity and the embedding capacity. We note that the payload of m' is inversely proportional to the number of vertices per block B.

Message Synchronization by Flagging

In the multi-message embedding variant of our proposed approach, when a message is embedded in the encrypted domain, the corresponding vertex blocks are flagged in order to synchronize this message with all the previously embedded messages. This flagging is necessary in order to extract the embedded messages. Concretely, the flags indicate which blocks are still available when another message is to be embedded. The flags replace the second tier message when comparing this variant to the previous two

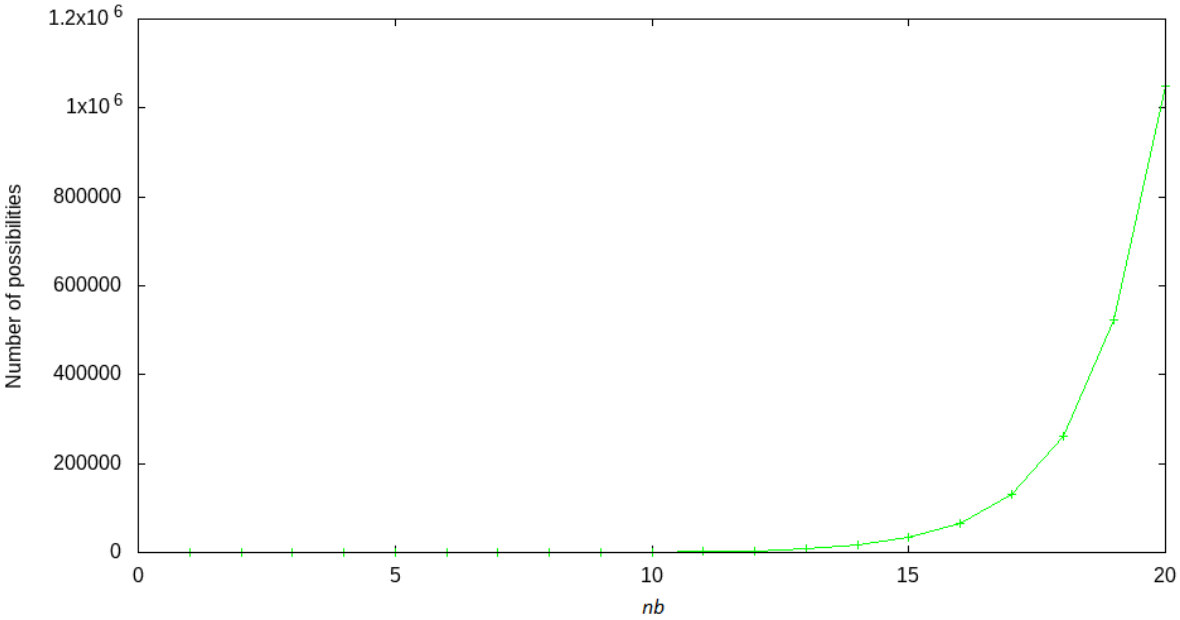


Figure 4.7: *Number of possibilities as a function of the value of* nb.

tier HCDH-ED variant. A flag f is therefore embedded using the Paillier probabilistic property which indicates that the encrypted value of the message m is not unique.

During the 3D object encryption, all the encrypted blocks B'_e are then flagged to 0. Based on Eq. (1.17), we choose r such:

$$B'_e \bmod 2 = 0, \tag{4.13}$$

where B'_e is the the encrypted watermarkable vertex block where all the flags are initialized at zero.

When a message is embedded in the encrypted 3D object, all the watermarked encrypted blocks B'_{ew} needed to embed this message are flagged as 1, except for the second to last one which is flagged as 0, so that two consecutive messages can be separated.

To do this, we propose using the Paillier cryptosystem's self-blinding property, Eq. (1.20), where we choose t relatively prime to n such that:

$$(B'_{ew} \times (t^n \bmod n^2) \bmod n^2) \bmod 2 = f, \tag{4.14}$$

where f is the corresponding flag, with $f \in \{0, 1\}$.

We note B'_{ew_f} the flagged watermarked encrypted block and O_{ew_f} the corresponding flagged watermarked encrypted 3D object. The complexity of our method can be expressed by the probability of choosing the correct r and t respectively so that a modulus 2 results in f:

$$P(X = f) = \frac{1}{2}. \tag{4.15}$$

4.2.6 3D Object Decryption and Message Extraction in the Plaintext Domain

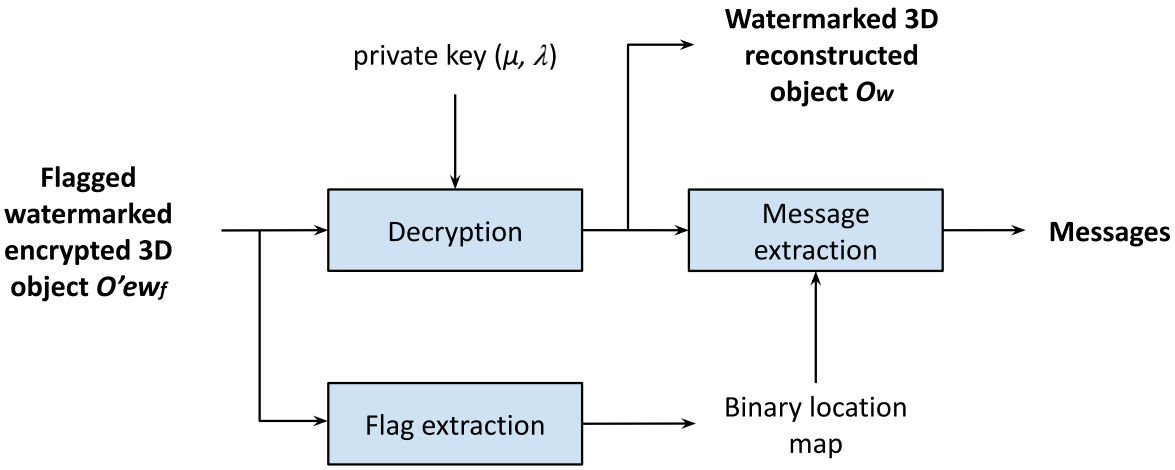


Figure 4.8: *The decryption and message extraction phases.*

In this section, we present the reconstruction of the 3D object and then the extraction of the embedded messages in the plaintext domain. Fig. 4.8 shows an overview of the decryption and the message extraction steps for the multi-message embedding variant. The flagged watermarked encrypted 3D object O'_{ew_f} is decrypted using the private key (μ, λ) (Eq. (1.18)) to give us the reconstructed watermarked 3D object O_w. We note that the data receiver needs only the private key and no other additional information in order to decrypt the 3D object, as the block size is determined by the key size. For each flagged watermarked encrypted block B'_{ew_f}, we obtain a decrypted watermarked block:

$$B_w = \mathcal{D}(B'_{ew_f}) = L({B'_{ew_f}}^{\lambda} \bmod n^2) \times \mu \bmod n, \tag{4.16}$$

where $\mathcal{D}(\cdot)$ is the Paillier decryption function.

In parallel to the decryption, the flag extraction from the flagged watermarked encrypted 3D object is performed for each block:

$$f = B'_{ew_f} \bmod 2, \tag{4.17}$$

which allows us to generate a binary location map that indicates which blocks contain messages.

All the messages can then be extracted from O_w and a binary location map generated from the extracted flags as illustrated in Fig. 4.8.

In the case of the two tier HCDH-ED variant, the second tier message m' is extracted using the same principle:

$$m' = B'_{ew_f} \bmod 2^{nb}, \tag{4.18}$$

where nb is the number of bits used to encode m'.

Fig. 4.9 illustrates the reconstruction of a watermarked vertex block B_w, which is retrieved from Eq. (1.18). The decryption of the $2k+1$ bits of the block B'_{ew_f} results in the original k MSB of the block B'. These bits replace the k MSB in the encrypted vertex

block to construct B_w. We extract the α LSB among the k MSB of the vertex block to retrieve the original message segment m.

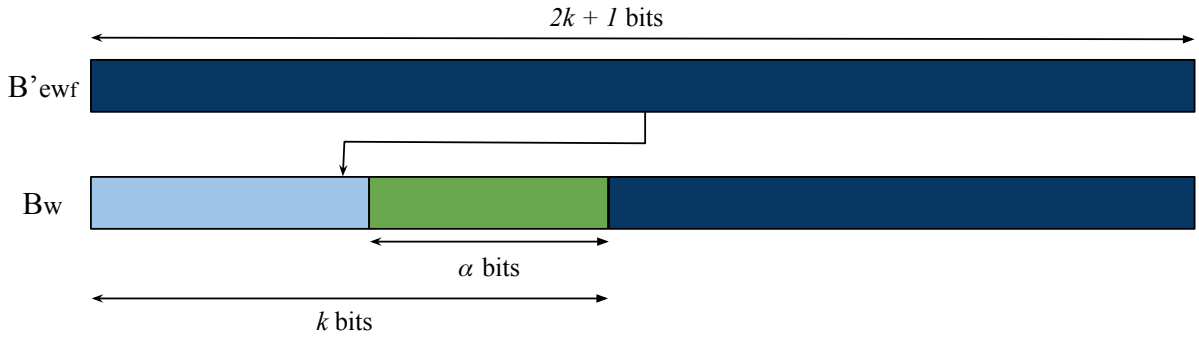


Figure 4.9: *Decryption of a block B'_{ew_f} in order to reconstruct a watermarked block B_w in the plaintext domain.*

4.3 Experimental Results

In this section, we present experimental results obtained with our method. First, in Section 4.3.1, we analyze if the key choice and the block size have an effect on the visual degradation of the decrypted 3D object. In Section 4.3.2, we present results on a large dataset and in Section 4.3.3, we compare our method with existing state-of-the-art methods. Finally, in Section 4.3.4, we present an application of our method to a real-life scenario.

We note that we present our results according to the multi-message embedding variant. There is indeed no difference between the reconstructed watermarked 3D object of the multi-message embedding variant and that of the two tier HCDH-ED variant. The only difference in the results presented is that the two tier HCDH-ED variant has a higher payload in the encrypted domain.

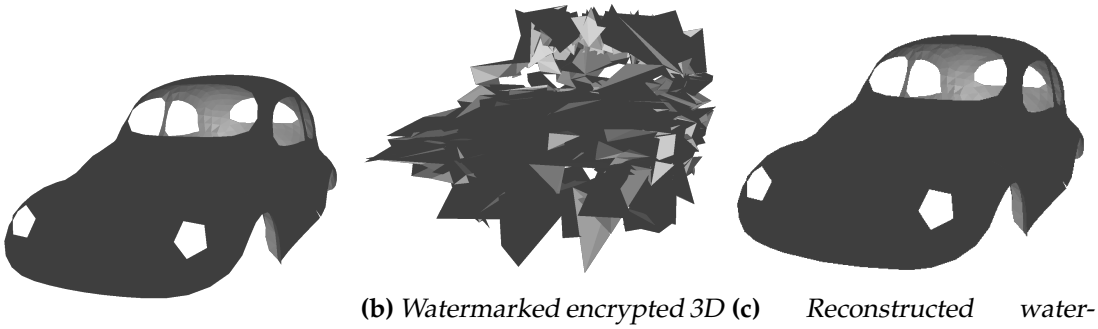
In order to be secure and for real life applications, we need a public key (n, g) where the size of n is at least an estimated 1000 bits[1]. Therefore, we group the vertices into blocks of size $b = 29$ vertices per block and so we have $69 \times 29 = 2k + 1$, which means that $k = 1000$. The value of n is therefore constrained by $2^{1000} \leq n < 2^{1000.5}$. Thus, n is represented by 1001 bits, while respecting the previous constraint.

We note O the original 3D object, O' the watermarkable 3D object, O'_e the encrypted watermarkable 3D object, O'_{ew} the watermarked encrypted 3D object, O'_{ew_f} the flagged watermarked encrypted 3D object and O_w the watermarked decrypted 3D object.

[1] A size of 1000 bits is just an example to illustrate our method in this work. We can apply our method with much larger key sizes.

4.3.1 Key and Block Size Analysis

Fig. 4.10a illustrates the original 3D object *Beetle*, Fig. 4.10b represents *Beetle* when it is encrypted and watermarked with messages with a payload of 13.5 bpv and Fig. 4.10c represents the watermarked reconstruction.



(a) *Original 3D object Beetle.* **(b)** *Watermarked encrypted 3D object Beetle.* **(c)** *Reconstructed watermarked 3D object Beetle.*

Figure 4.10: *Obtained results when the 3D object Beetle is watermarked with a payload of* 13.5 bpv *(block size of 29 vertices).*

The 3D object *Beetle*, Fig. 4.10a, has been encrypted, watermarked and then decrypted using 50 different keys of 1001 bits (corresponding to blocks of 29 vertices) drawn at random from a list of eligible keys. Table 4.1 presents the obtained statistical results between the watermarked decrypted 3D objects O_w and the original 3D object O.

Table 4.1: *Comparison between 50 watermarked decrypted instances of the 3D objects Beetle* O_w *and the original 3D object Beetle* O*.*

Beetle (O, O_w)	RMSE (10^{-3})	Hausdorff (10^{-3})
Mean	0.6933	1.741
St. Deviation	0.00169	0.000185
Median	0.6934	1.739
Minimum	0.6920	1.713
Maximum	0.6943	1.774

From the standard deviations of the RMSE and Hausdorff distances which are of the order 10^{-6} and 10^{-7} respectively, we can conclude that the key does not influence the quality of the watermarked decrypted 3D objects. We can also note that there are no outliers, since the minimum and maximum values are very similar to one another. The minimum RMSE value is 0.6920×10^{-3} compared to the maximum value of 0.6943×10^{-3}, and the minimum Hausdorff distance is 1.713×10^{-3} whereas the maximum is 1.774×10^{-3}.

Because of the self-blinding homomorphic property, when we embed a flag f, the decrypted value of the vertex block watermarked with m does not change. Therefore the flag embedding does not affect the quality of the watermarked decrypted 3D objects.

Fig. 4.11 and Fig. 4.12 illustrate the RMSE and the Hausdorff distances respectively for different values of the block size b ($b = 1, b = 5, b = 9, b = 29$ vertices per block)

according to the payload. We can conclude that the block size b does not influence the distortion of the watermarked decrypted 3D object.

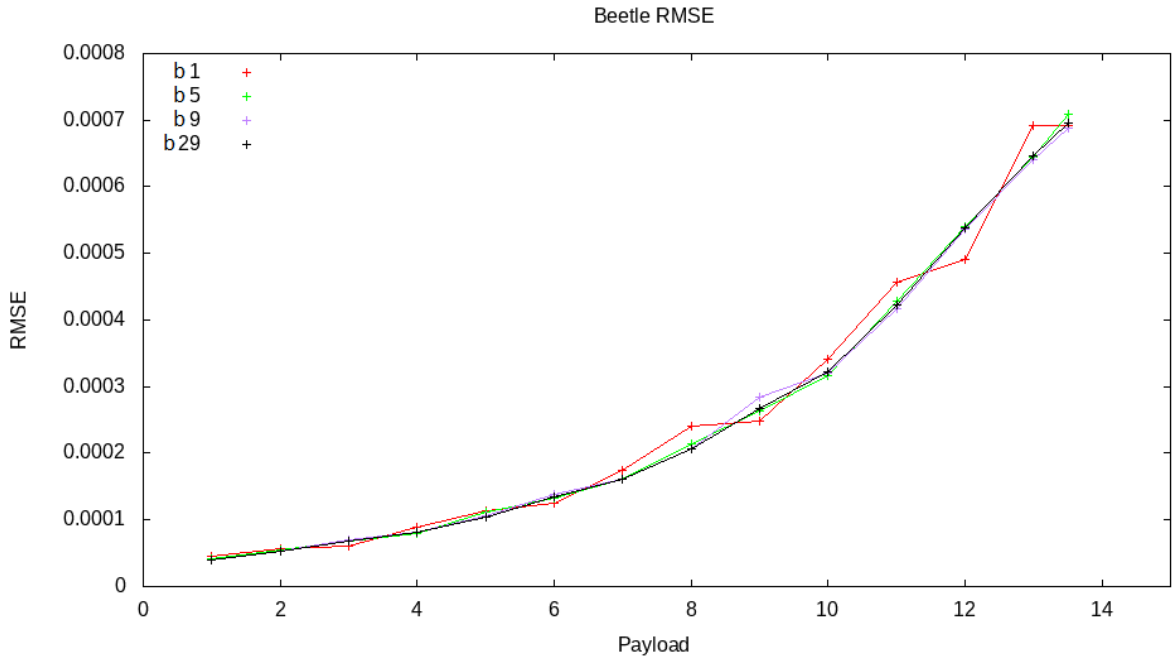


Figure 4.11: *RMSE between the original 3D object Beetle and the reconstructed one as a function of the payload in bpv and the block size b vertices per block.*

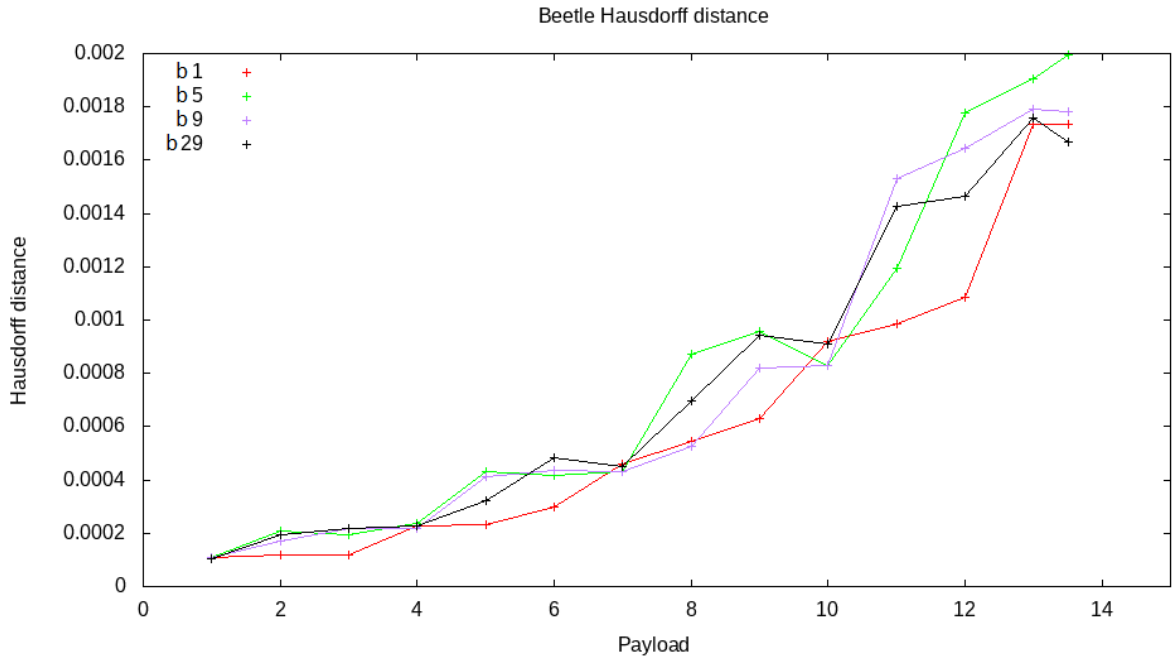


Figure 4.12: *Hausdorff distance between the original 3D object Beetle and the reconstructed one as a function of the payload in bpv and the block size b vertices per block.*

4.3.2 Performance on a Large Dataset

We tested our method on the Princeton dataset (143) which consists of 380 different 3D objects. As in Section 4.3.4, vertices are grouped into blocks of size 29 vertices per block, resulting in a secure key size of 1001 bits.

Table 4.2 and Table 4.3 present the Hausdorff distance and RMSE values respectively. We compare the original 3D object O with the encrypted 3D object O'_e, the watermarked encrypted 3D object O'_{ew} and finally the watermarked decrypted 3D object O_w. We also compare O'_e with O'_{ew}.

Table 4.2: *Hausdorff distances obtained when our proposed method is applied to the Princeton dataset (143).*

Princeton	O/O'_e	O/O'_{ew}	O'_e/O'_{ew}	O/O_w
Mean	0.4677	0.4686	0.1392	3.769×10^{-3}
St. Deviation	0.1101	0.1100	0.0531	0.443×10^{-3}
Median	0.4833	0.4830	0.1288	3.744×10^{-3}
Minimum	0.1127	0.1124	0.0129	2.580×10^{-3}
Maximum	0.6949	0.6734	0.4181	5.267×10^{-3}

Table 4.3: *RMSE obtained when our proposed method is applied to the Princeton dataset (143).*

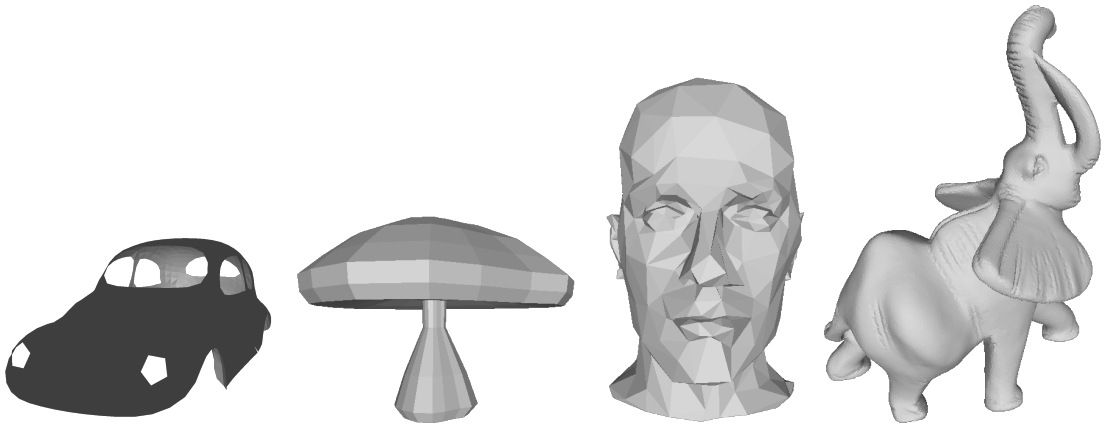
Princeton	O/O'_e	O/O'_{ew}	O'_e/O'_{ew}	O/O_w
Mean	0.1698	0.1698	0.1668	1.303×10^{-3}
St. Deviation	0.0290	0.0290	0.0255	0.199×10^{-3}
Median	0.1636	0.1637	0.1615	1.263×10^{-3}
Minimum	0.1156	0.1173	0.1166	0.903×10^{-3}
Maximum	0.2679	0.2671	0.2381	2.079×10^{-3}

We observe that while O/O'_e and O/O'_{ew} have very similar Hausdorff distances and RMSE, represented in Table 4.2 and Table 4.3 respectively, the Hausdorff distance and the RMSE of O'_e/O'_{ew} remain large. Therefore we can conclude that the content of the 3D object remains secure independently of whether there is an embedded message or not. Moreover, the median Hausdorff distance and RMSE of O/O_w are 3.744×10^{-3} and 1.263×10^{-3} respectively, which indicates that the resulting watermarked 3D object O_w is similar to the original 3D object O. We note that the mean distances are similar to the median distances. With a maximum Hausdorff distance and RMSE of 5.267×10^{-3} and 2.079×10^{-3} respectively, these 3D objects remain visually identical to the original.

4.3.3 Comparisons with Previous Work

In this section we compare the results of our method with those of existing work Jiang *et al.* (135), Shah *et al.* (141), Yin *et al.* (137), Lyu *et al.* (139) and Xu *et al.* (136). In order to compare our obtained results with previous work, we develop our experimentation using four standard test 3D objects: *Beetle* (988 vertices, Fig. 4.13a), *Mushroom* (226 vertices, Fig. 4.13b), *Mannequin* (428 vertices, Fig. 4.13c) and *Elephant* (24,955

vertices, Fig. 4.13d). For this experiment, in order to make a comparison with other state-of-the-art methods, we encrypt these four 3D objects vertex by vertex. Indeed, while our method can reach a payload of 13.5 bpv depending on the block size, we set the block size $b = 1$ vertex per block and the maximum payload for $b = 1$ which is 13 bpv.



(a) *Beetle* **(b)** *Mushroom* **(c)** *Mannequin* **(d)** *Elephant*

Figure 4.13: *Standard 3D objects used to compare our results with other state-of-the-art methods.*

Table 4.4: *Feature comparison between our proposed method and other existing state-of-the-art methods.*

	Encrypted Domain					Plaintext Domain	
Methods	Encryption	Size expansion	Auxiliary file	Payload (in bpv)	Data error	Marked 3D object	HC Marked 3D object
Jiang *et al.* (135)	Exclusive-or	No	No	0.37	Yes	No	No
Shah *et al.* (141)	Paillier cryptosystem	Yes	No	6 (3+3)	No	Yes	No
Yin *et al.* (137)	Exclusive-or	No	Yes	16.25	No	No	No
Lyu *et al.* (139)	Exclusive-or	No	No	22.83	No	No	No
Xu *et al.* (136)	Exclusive-or	No	Yes	1.07	No	No	No
Proposed $b = 1$	Paillier cryptosystem	No	No	13	No	**Yes**	**Yes**

Table 4.4 presents a feature comparison between our proposed method and five existing state-of-the-art methods Jiang *et al.* (135), Shah *et al.* (141), Yin *et al.* (137), Lyu *et al.* (139) and Xu *et al.* (136). Our proposed method is the only one to avoid size expansion, an auxiliary file and data error. Note also that our method is able to generate a watermarked 3D object in the plaintext domain.

We note that the payloads of the methods of Jiang *et al.* (135), Yin *et al.* (137), Lyu *et al.* (139) and Xu *et al.* (136) are the average payloads of the four 3D objects, as the payloads of these methods depend on the number of vertices eligible for embedding. The

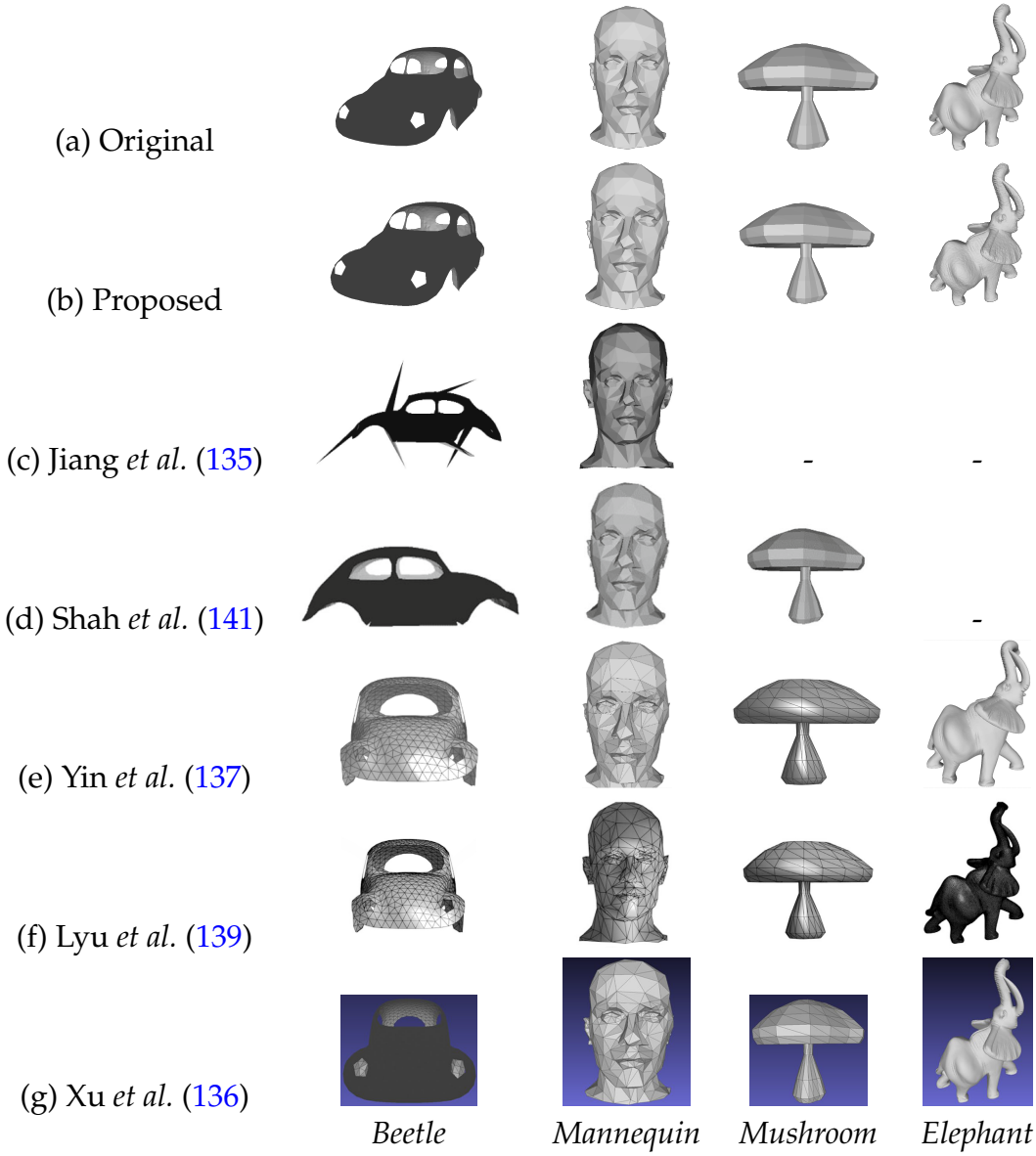


Figure 4.14: *Visual results of Beetle, Mannequin, Mushroom and Elephant with the proposed method compared to current state-of-the-art methods (135; 141; 137; 139; 136).*

payload of Shah *et al.* is divided into two parts: the payload in the plaintext domain and the possible payload in the encrypted domain. While both the proposed method and the method of Shah *et al.* (141) produce a watermarked 3D object in the plaintext domain, our proposed method has no size expansion and achieves a significantly higher payload. Indeed, the method we propose is the only one which allows us to obtain a high capacity payload in both plaintext and encrypted domains.

Fig. 4.14 presents visual results of the proposed method and those of current state-of-the-art methods. Fig. 4.14.a presents the original 3D objects *Beetle*, *Mannequin*, *Mushroom* and *Elephant*. Fig. 4.14.b presents the visual results of the proposed method while Fig. 4.14.c, Fig. 4.14.d, Fig. 4.14.e, Fig. 4.14.f and Fig. 4.14.g present visual results from previous work (taken from (135; 141; 137; 139) and (136) respectively). We observe that despite a generally higher Hausdorff distance than (137; 141; 139) and (136), like (137; 141; 139) and (136), the results of our proposed method are visually similar to the

3D object	Methods	Encrypted domain payload (*bpv*)	Plaintext domain payload (*bpv*)	HD (10^{-3})
Beetle	Jiang *et al.* (135)	0.35	0	0.977
	Shah *et al.* (141)	6 (3+3)	3	0.034
	Yin *et al.* (137)	16.51	0	8.60×10^{-3}
	Lyu *et al.* (139)	23.55	0	8.66×10^{-3}
	Xu *et al.* (136)	0.98	0	0.866×10^{-3}
	Proposed	1	1	0.108
	Proposed	7	**7**	0.461
	Proposed	13	**13**	1.73
Mushroom	Jiang *et al.* (135)	0.45	0	0.960
	Shah *et al.* (141)	6 (3+3)	3	0.400
	Yin *et al.* (137)	16.72	0	8.10×10^{-3}
	Lyu *et al.* (139)	21.76	0	8.12×10^{-3}
	Xu *et al.* (136)	1.34	0	75.3×10^{-3}
	Proposed	1	1	0.209
	Proposed	7	**7**	0.881
	Proposed	13	**13**	3.18
Mannequin	Jiang *et al.* (135)	0.34	0	1.01
	Shah *et al.* (141)	6 (3+3)	3	0.370
	Yin *et al.* (137)	13.66	0	4.00×10^{-3}
	Lyu *et al.* (139)	18.05	0	4.00×10^{-3}
	Xu *et al.* (136)	0.95	0	4.00×10^{-3}
	Proposed	1	1	0.655
	Proposed	7	**7**	2.70
	Proposed	13	**13**	8.04
Elephant	Jiang *et al.* (135)	0.34	0	1.08
	Shah *et al.* (141)	6 (3+3)	3	0.0339
	Yin *et al.* (137)	18.12	0	8.60×10^{-3}
	Lyu *et al.* (139)	27.96	0	8.64×10^{-3}
	Xu *et al.* (136)	1.02	0	8.66×10^{-3}
	Proposed	1	1	0.149
	Proposed	7	**7**	0.543
	Proposed	13	**13**	2.82
Average	Jiang *et al.* (135)	0.37 ± 0.05	0	1.01 ± 0.046
	Shah *et al.* (141)	6 (3+3)	3	0.209 ± 0.176
	Yin *et al.* (137)	16.25 ± 1.62	0	$(7.325 \pm 1.93) \times 10^{-3}$
	Lyu *et al.* (139)	22.83 ± 4.12	0	$(7.36 \pm 2.25) \times 10^{-3}$
	Xu *et al.* (136)	1.07 ± 0.18	0	$(22.21 \pm 35.54) \times 10^{-3}$
	Proposed	1	1	0.280 ± 0.219
	Proposed	7	**7**	1.15 ± 0.911
	Proposed	13	**13**	3.94 ± 2.43

Table 4.5: *Comparison of the payload in both encrypted and plaintext domains, and of the distortion between our method and five significant current state-of-the-art approaches for the four 3D objects Beetle, Mushroom, Mannequin and Elephant.*

original 3D objects. However, to the best of our knowledge, we are the only method to achieve a high capacity data hiding in the resulting decrypted 3D object, which remains watermarked with hidden messages.

Table 4.5 represents comparisons between the payloads in both the plaintext and encrypted domains, and the Hausdorff distances of the results of our proposed method and those of the existing state-of-the-art methods. We note that while the state-of-the art methods seek to reconstruct the original 3D object, in the proposed method we retrieve a 3D object which remains watermarked with the hidden messages that were embedded in the encrypted domain. Therefore we do not seek to be statistically identical to the original 3D object. With our method, note that the reconstructed watermarked 3D object remains visually very similar to the original 3D object, as shown in Fig. 4.16. Our method is the only one that achieves a high payload in both the plaintext and the encrypted domains. With a block size of $b = 1$, once the 3D object is reconstructed, it remains watermarked with a message of up to $13\ bpv$.

4.3.4 Application to a Real-Life Scenario

In this section, we present results obtained with our method when applied to a real-life scenario. We propose to apply our method on the 3D object of a *New Balance*© brand shoe, which we call *Shoe*, provided by the Stratégies[2] company.

(a) *Original 3D object O of a New Balance*© *brand shoe.* **(b)** *Encrypted watermarkable 3D object O'_e.* **(c)** *Watermarked encrypted 3D object O'_{ew}.*

(d) *Flagged watermarked encrypted 3D object O'_{ew_f}.* **(e)** *Reconstructed watermarked 3D object O_w.*

Figure 4.15: *Obtained results on a 3D object O of a New Balance*© *brand shoe, with a payload of $13.5\ bpv$ (block size of 29 vertices).*

Fig. 4.15 illustrates the 3D object *Shoe* at different stages of the proposed method, with a payload of $13.5\ bpv$. Fig. 4.15a presents the original 3D object which a designer creates. Before sending it along the production line, the 3D object is encrypted (Fig. 4.15b) in order to produce an encrypted watermarkable 3D object O'_e. This encrypted 3D object is then sent to multiple different parties in the production line over a network. Each time the encrypted 3D object is sent, the server, which does not have the right to access the original 3D object, watermarks the 3D object with information such as the IP address of the sender. Fig. 4.15c illustrates this watermarked encrypted

[2]Stratégies (https://www.romans-cad.com/)

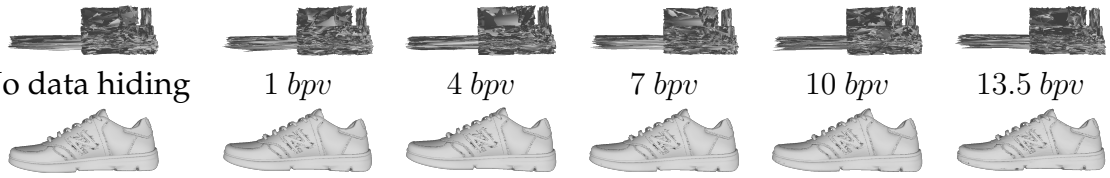


Figure 4.16: *Comparison between the watermarked encrypted 3D object O'_{ew_f} and the corresponding reconstructed watermarked 3D object O_w according to the payload from $1\ bpv$ to $13.5\ bpv$.*

3D object. The server also flags the watermarked 3D object in Fig. 4.15d so that other information can be hidden. We observe that the encrypted 3D object's content remains secure after each message is embedded. Finally, the 3D object is decrypted, resulting in a watermarked 3D object presented in Fig. 4.15e. We note that the original 3D object O, Fig. 4.15a, and the resulting watermarked 3D object O_w, Fig. 4.15e, are visually very similar.

From the 3D object *Shoe*, the top row of Fig. 4.16 illustrates the watermarked encrypted 3D object O'_{ew} according to the payload from $1\ bpv$ to $13.5\ bpv$, while the bottom row illustrates the corresponding watermarked decrypted 3D object O_w. We observe that while the content of the 3D object is secure when encrypted, there are no visual differences between the resulting watermarked decrypted 3D objects.

Table 4.6: *Hausdorff distance measurements when our proposed method is applied to a 3D object of a New Balance© brand shoe.*

Payload in bpv	O/O'_e	O/O'_{ew}	O/O'_{ew_f}	O/O_w
1	0.2332	0.2344	0.2313	0.1167 10^{-3}
4	0.2317	0.2317	0.2317	0.2342 10^{-3}
7	0.2306	0.2306	0.2306	0.4601 10^{-3}
10	0.2317	0.2317	0.2310	0.9558 10^{-3}
13.5	0.2315	0.2315	0.2305	1.9337 10^{-3}

Table 4.6 represents the Hausdorff distances when our method is applied to the 3D object *Shoe*. We observe that for each payload the values of O/O'_e, O/O'_{ew} and O/O'_{ew_f} are similar, which indicates that the content of the 3D object is secure in O'_e, O'_{ew} and O'_{ew_f}, while the content remains clear in O_w.

4.4 Conclusion

In this chapter we proposed a new high capacity HCDH-ED method for 3D objects based on the Paillier cryptosystem. We describe a method which conserves the original format and avoids both size expansion and the use of an auxiliary file, while maintaining the visual quality of the 3D object. Our method uses a large key size, which makes it suitable for real life applications. Most importantly, our approach is a method in which the message can be extracted in the plaintext domain, producing a reconstructed 3D object watermarked with up to 13.5 *bpv*. To the best of our knowledge,

our method is the only one that achieves a high payload both in the plaintext and encrypted domains. We have proposed two variants for this method. In the first variant, a second tier message can be embedded in the encrypted domain, whereas in the second variant, the watermarked encrypted vertex blocks are flagged, which allows us to have multi-embedding in the encrypted domain.

The proposed method could be further improved by ordering the coordinates within the vertex block B according to the ascending order of the three exponents e of the vertex coordinates in Eq. 1.22. This would lead to less distortion in the case where the same number of bits are not watermarked in every coordinate.

We are currently working on a reversible data hiding in the encrypted domain method for 3D objects with the use of a Hamiltonian path. The vertices in the plaintext domain are first ordered according to a Hamiltonian path, before the 3D object is encrypted and the message is embedded by substituting the MSB of each coordinate. After decryption, the Hamiltonian path is used to predict the MSB of each coordinate in order to restore the original 3D object.

The work presented in this chapter is presented in two different international publications. The first variant, the two tier HCDH-ED variant, was presented during the international conference IEEE ICIP 2021 (144). The second variant, the multi-message embedding variant, was presented in detail in the international journal ACM Transactions on Multimedia Computing, Communication and Applications in 2023 (145).

CHAPTER 5

3D OBJECT VISUAL SECURITY

Chapter Contents

5.1 Introduction

In this chapter, we present two contributions. As described in Chapter 1, there are three levels of 3D visual security which correspond to the accessibility of the content according to the human visual system (HVS). A 3D object whose shape and content are not recognizable has a **confidential** level. In the case where only the shape and not the content of the 3D object is accessible, the 3D object has a **sufficient** level. Finally, a 3D object whose shape and content are accessible, but the high quality details remain secure and cannot be used, for example for 3D printing, corresponds to a **transparent** level. In this chapter, we present two contributions relating to the visual security levels.

While there are many different methods proposed in the state of the art for 3D object visual quality assessment, these cannot be used for visual security assessment. The two assessment categories are indeed very different, as secured 3D object are simply labeled as having a bad quality by quality assessment metrics.

The first contribution is a new 3D visual security metric, called 3D Visual Security (3DVS) score, which is designed to measure the visual security level of selectively encrypted 3D objects. This score is constructed with a combination of classic linear regression and logistic regression. We also construct a polynomial regression model in order to estimate the selective encryption parameters according to the desired visual security level. This metric is based on a dataset which consists of 50 3D objects which are each selectively encrypted with 10 different encryption parameter levels. This dataset, called *Selectively Encrypted 3D Object* (SE3DO) dataset is composed of 550 3D objects and is accompanied by opinion scores (OS) gathered from 54 different observers.

The main novelties of this contribution are summarized as:

1. A new model to estimate the adequate security parameters according to the desired visual security level, based on the SE3DO dataset;
2. A new metric based on full reference 3D metrics, called 3D Visual Security (3DVS) score, designed to measure the visual security level of selectively encrypted 3D objects.

Selective encryption methods are useful when a user is permitted to access part of the 3D object, such as the shape, but not the content. However, these 3D objects may need to be transferred via a third party, such as a network or a production line, where it should be confidentially secured.

The second contribution is a new format compliant 3D object encryption method, which allows for a hierarchical decryption of a single encrypted 3D object. An encrypted 3D object can be hierarchically decrypted so that the result is a 3D object with the desired visual security level (from confidential to transparent or sufficient). Encrypted 3D objects can also be fully decrypted in order to recover the original clear level 3D object. During the encryption phase, our proposed method encrypts a 3D object so that it has a confidential visual security level. From a master key, a ring of hierarchical keys is established during the encryption phase. Each key corresponds to a

different visual security level. Users are then able to decrypt the confidential 3D object where the resulting 3D object has a visual security level according to the hierarchical level of the key used for decryption. To the best of our knowledge, we are the first to propose a 3D object encryption method which allows for a hierarchical decryption of 3D objects.

Our method is therefore an essential function in industry as it prevents the sharing of trade secrets. For example, during the manufacturing process, different users may have different access rights. A certain user may have the right to only identify what type of object a 3D object represents (for example, a shoe), and therefore the sufficient level. Another user may have the right to access the specific design model, but not the original, and therefore the transparent level, as to prevent 3D printing. The 3D object is therefore secure against unauthorized sharing.

Our method is indeed more ecologically friendly and secure than selective encryption. Instead of having to encrypt, store and send multiple selectively encrypted 3D objects according to each user, a process which consumes many resources, a 3D object is confidentially encrypted, stored and shared only once. As this 3D object has a confidential level, it can be securely transferred since third parties such as the environment, or an attacker, will not be able to access any information about the 3D object. This is not the case when sharing selectively encrypted 3D objects.

The main novelties of this contribution are summarized as:

1. We propose a new format compliant 3D object encryption method;

2. The method allows for a hierarchical decryption of a single encrypted 3D object;

3. The result of the hierarchical decryption is a 3D object with the desired visual security level (confidential, sufficient or transparent);

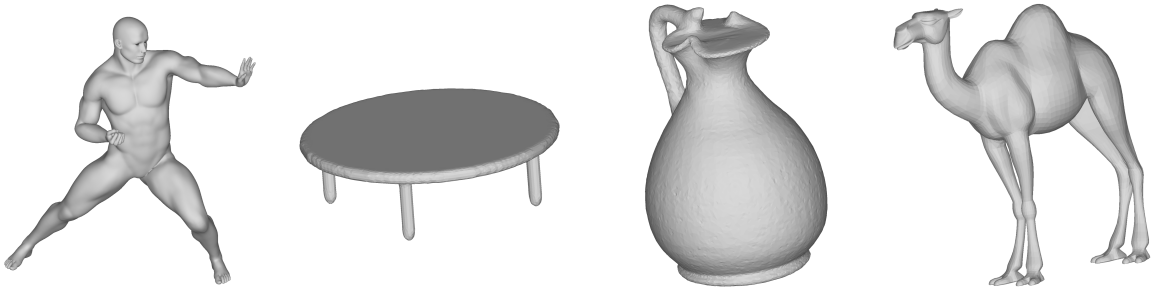
4. From a master key, a ring of hierarchical keys is established during the encryption phase where each key corresponds to a different visual security level.

This chapter is organized as follows. First, in Section 5.2, we present the SE3DO dataset upon which our first contribution is based. Then, in Section 5.3 we detail the security parameter prediction. In Section 5.4, we describe the proposed 3DVS score for visual security assessment for 3D objects. Then, in Section 5.5, we present our proposed format compliant encryption method which allows for hierarchical decryption of 3D objects. In Section 5.6 we detail the results of our experimentation and carry out a sensitivity analysis. Finally, in Section 5.7 we conclude our work.

5.2 The SE3DO Dataset

5.2.1 The Original 3D Objects

To the best of our knowledge, this is the first dataset of subjective evaluations of the visual security of encrypted 3D objects. All previous 3D object datasets have been constructed to assess quality in various contexts such as acquisition, processing, compression, watermarking or even segmentation applications. The SE3DO dataset is constructed with 3D objects from existing datasets, such as the Princeton dataset (143), *SHREC-12* and *SHREC-14* for 3D segmentation (146; 147) and *Thingi10k* (148). From these datasets, 50 3D objects are selected, which serve as references in the SE3DO dataset. Four examples of these 3D objects are presented in Fig. 5.1



(a) *Object #5* **(b)** *Object #12* **(c)** *Object #24* **(d)** *Object #32*

Figure 5.1: *Examples of reference 3D objects from the Selectively Encrypted 3D Object (SE3DO) dataset.*

As illustrated in Fig. 5.1, the reference 3D objects are either from CAD (Fig. 5.1.a) or from scanning systems (Fig 5.1.b). Some of these 3D objects represent living entities named *organic* objects, whereas the complement represents *inert* objects (helmet, statue, vase, *etc.*). By varying the origins of the reference 3D objects, as well as the represented content, a wide variety of 3D objects is evaluated. This is in order to study the effects of 3D selective encryption and to determine not only where it is most effective in protecting the content, but also where it is most adapted to the density of the vertices, the size of the 3D object and the local curvature which can influence its numerical form.

Fig. 5.2 presents the distribution of the reference 3D objects: 31 are 3D objects generated by scanning, whereas 19 of the 3D objects are produced using CAD tools. Among these 3D objects, 21 of them represent organic 3D objects, while the other 29 represent inert 3D objects. We note that the reference 3D objects are distinguished by properties such as the number of vertices, the number of faces, the average length of an edge, as well as other characteristics, as illustrated in Fig. 5.3.

5.2.2 The Encrypted 3D Objects

These 50 reference 3D objects are encrypted using the 3D selective encryption method from (47), described in Chapter 1. In (47), it is only the 23 bit mantissa that is encrypted. The degradation level is based on the variable p, where p indicates the first

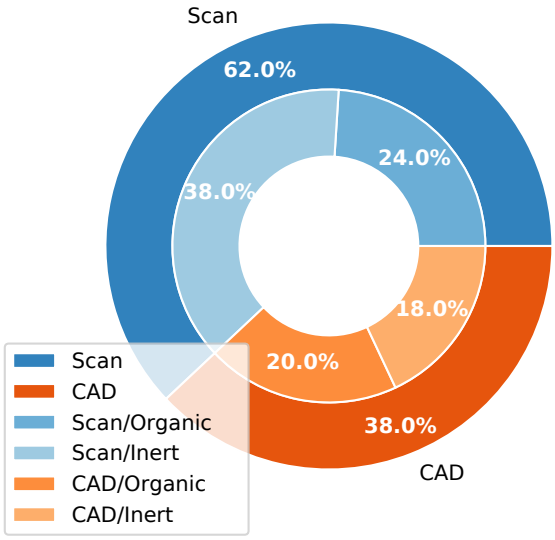
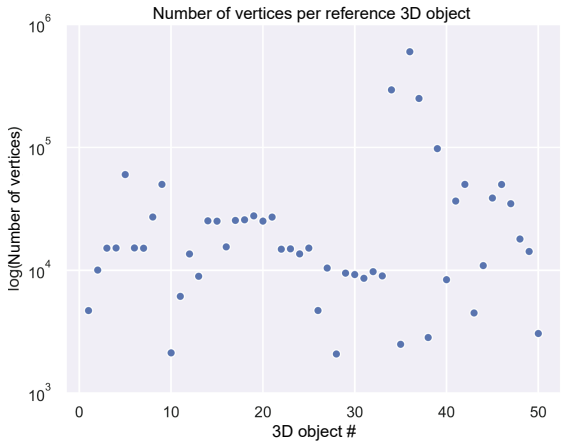
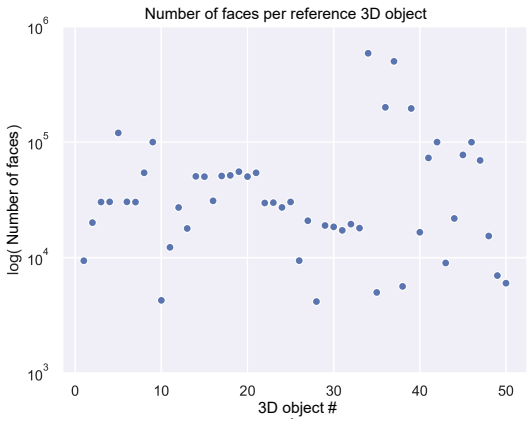


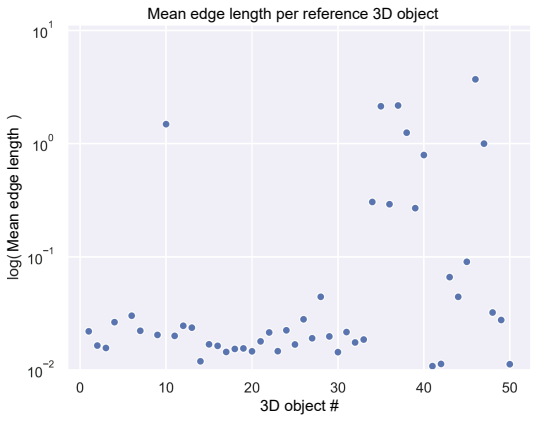
Figure 5.2: *Distribution of reference 3D objects in the Selectively Encrypted 3D Object (SE3DO) dataset depending on the origin (CAD or Scan) and type of content (Organic or Inert).*



(a) *Number of vertices*



(b) *Number of faces*



(c) *Mean edge length*

Figure 5.3: *Characteristics of reference 3D objects of the SE3DO dataset.*

bit of the mantissa to be encrypted. Then the 50 3D objects are encrypted using the degradation levels where $p \in [13, 22]$, as illustrated in Fig. 5.4 for the 3D object #18. According to a study by Beugnon *et al.*, when the parameter $p < 13$, the RMSE between the original 3D object and the selectively encrypted 3D object is negligible (47). When $13 \leq p \leq 16$, even though there is no visual difference between the original and selectively encrypted 3D object (Fig. 5.4), the RMSE is no longer negligible. Thus the degradation levels where $p \in [13, 22]$ are used. For each encryption, the secret key K is changed to avoid bias. Finally, the SE3DO dataset is obtained. This dataset is composed of 550 3D objects, where 500 of them are selectively encrypted and the remaining 50 are the reference 3D objects.

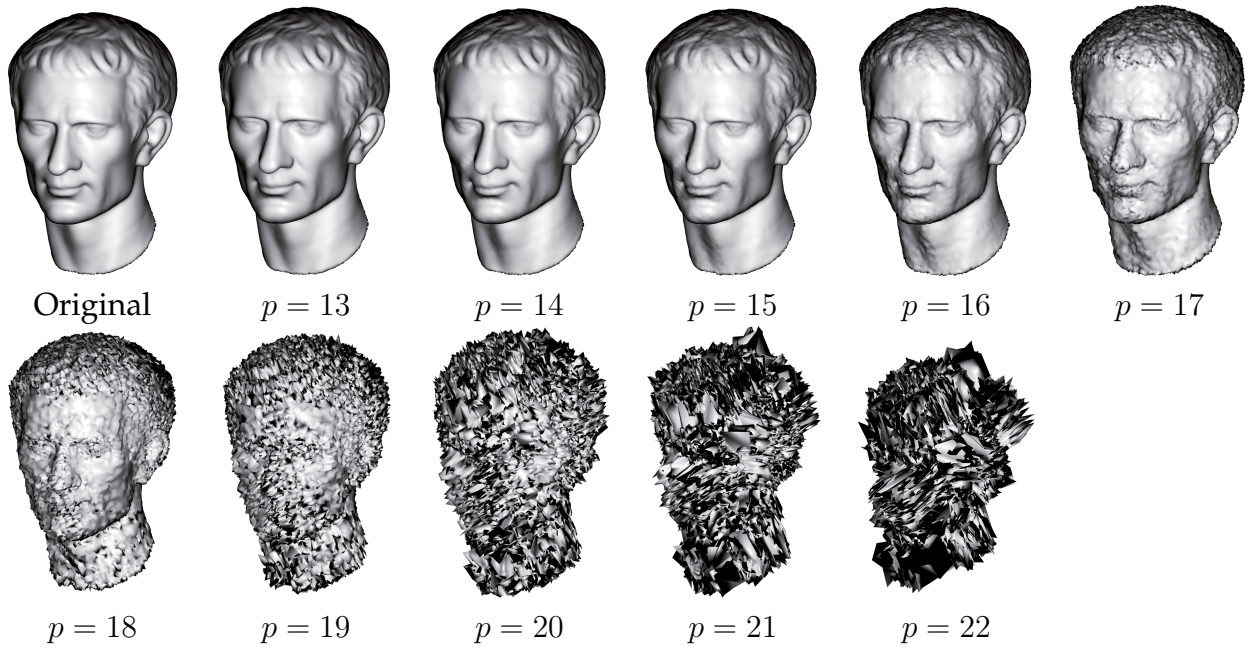


Figure 5.4: *3D objects representing the selective encryption of the 3D object #18 of the SE3DO dataset, according to the different selective encryption levels where the parameter* $p \in [13, 22]$.

5.2.3 Evaluation Protocol

Generally, images are subject to large varieties of distortions related to acquisition, processing, compression, storage, transmission, reproduction or data hiding. This results in a degradation of their visual quality. The same observations can be made for 3D objects. Initially, subjective evaluations are generally used to evaluate the performance of objective metrics. With the arrival of machine learning methods, some metrics are constructed from the data produced by the evaluations (149).

Evaluation System

The most common measurement for creating subjective metrics is the Mean Opinion Score (MOS) which usually varies between 1 and 5. Table 5.1 presents the scale defined for the study of the visual security of 3D objects for the subjective evaluations. Values

MOS	Signification	Shape	Content	Quality
1	**Confidential level**	Confidential	Confidential	Poor
2	**Sufficient level**	Accessible	Confidential	Poor
3	**Transparent level**	Accessible	Accessible	Low
4	Noisy 3D object	Accessible	Accessible	Medium
5	Clear 3D object	Accessible	Accessible	High

Table 5.1: *Values and significations of the MOS as part of the visual security assessment.*

1, 2 and 3 correspond to the three different security levels defined by (48) and (47) as part of the scenarios for selective encryption of visual data, namely the confidential, sufficient and transparent level. Thus, a MOS value of 1 is associated with the confidential level, where the selective 3D object encryption methods generate 3D objects whose shape and content are confidentially protected. The MOS value of 2 is associated with the sufficient level, where only the shape of the 3D object is recognizable, but not its content. Finally, a MOS value of 3 corresponds to the transparent level, this allows recognition of the form and the content, but the high quality of the 3D object is protected. A MOS value of 4 allows observers to differentiate between noisy 3D objects and 3D objects that are of high quality with a MOS value of 5. The value of MOS 5 corresponds to 3D objects that have no obvious defects and therefore are high quality, unencrypted 3D objects.

Stimulus Mode

Among the different subjective quality assessment protocols there are 4 pre-dominant modes, namely *single-stimulus*, *double-stimulus*, *forced-choice pairwise comparison* and *similarity judgments*. Each of these modes has its advantages as well as its disadvantages (150). However, in this subjective assessment, it is very clear that the single-stimulus protocol must be used. Indeed, as the aim is to create a metric to evaluate the visual security of a selectively encrypted 3D object, the use of protocols using two 3D objects (the original 3D object and the encrypted one, for example) provides information on the shape and content of the 3D object whose visual security is analyzed. Thus, other modes of *stimulus* seeking to compare two 3D objects of different qualities would not provide any information on the visual security. Due to the nature of the data, in this case 3D objects, observers are allowed to interact with these 3D objects with the use of camera motions (translation, rotations, zooms) so that they can observe the shape of the 3D object from all angles.

Evaluation Environment

Subjective assessments are carried out with a standardized protocol to provide correct and universal results. However, they require a controlled environment and many

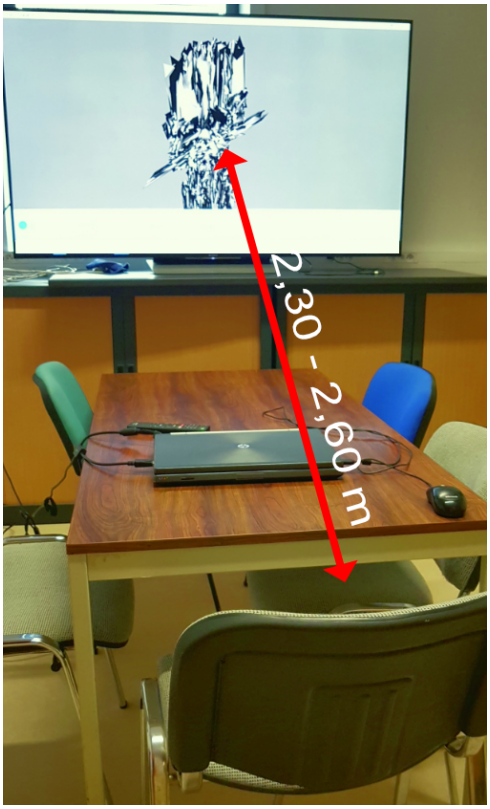


Figure 5.5: *The room used for subjective evaluation of the visual security of selectively encrypted 3D objects.*

limitations due to human judgment that can vary significantly depending on external conditions and individuals.

The evaluations are therefore conducted in a specialized room behind closed doors to maintain control over essential elements such as light, screen resolution and distance from the screen. As shown in Fig. 5.5, the observers are positioned in front of a professional LCD screen *Sony FW-75XD8501* 4K Ultra HD of $190.5\ cm$, based on LED technology with a resolution of 3840 $\times$ 2160 pixels and a brightness of about $450\ cd/m^2$. The observers are seated at a distance of between 2.30-2.60 meters from the screen.

The 3D objects are displayed on a uniform grey background, without texture, using a *shader* based on the Phong lighting model (151) with light grey material turning white in specular areas as illustrated in Fig. 5.6.

Evaluation Procedure

As previously presented, there are 50 3D reference objects and for each of these objects, as well as 10 additional variations generated with the encryption parameters ($13 \leq p \leq 22$). Each observer evaluates 50 distinct 3D objects which can be original or selectively encrypted with ($13 \leq p \leq 22$). To prevent observers from learning to recognize 3D objects, they are showed only one random variant of each of the 50 3D reference objects. As illustrated in Fig. 5.7, before the evaluation phase, all 5 different levels of a 3D object are presented to the observer in order to introduce the problem

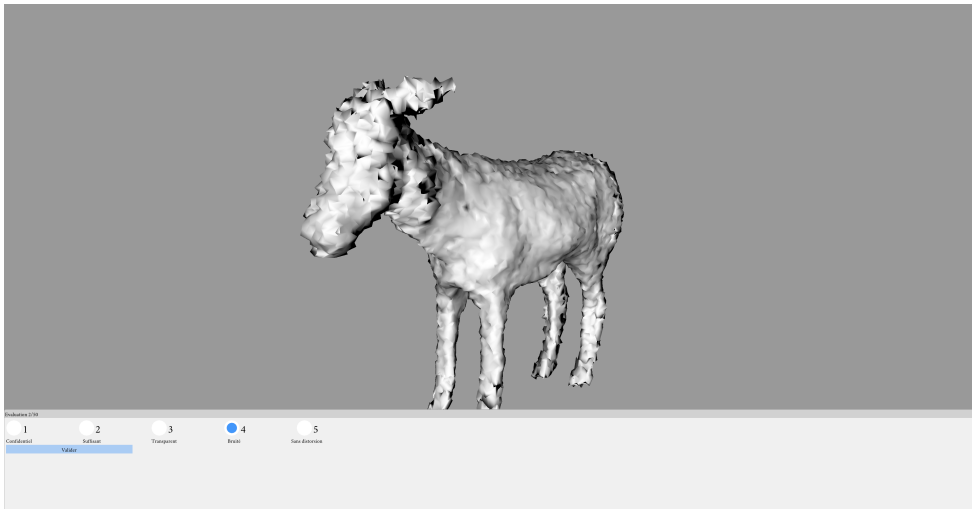
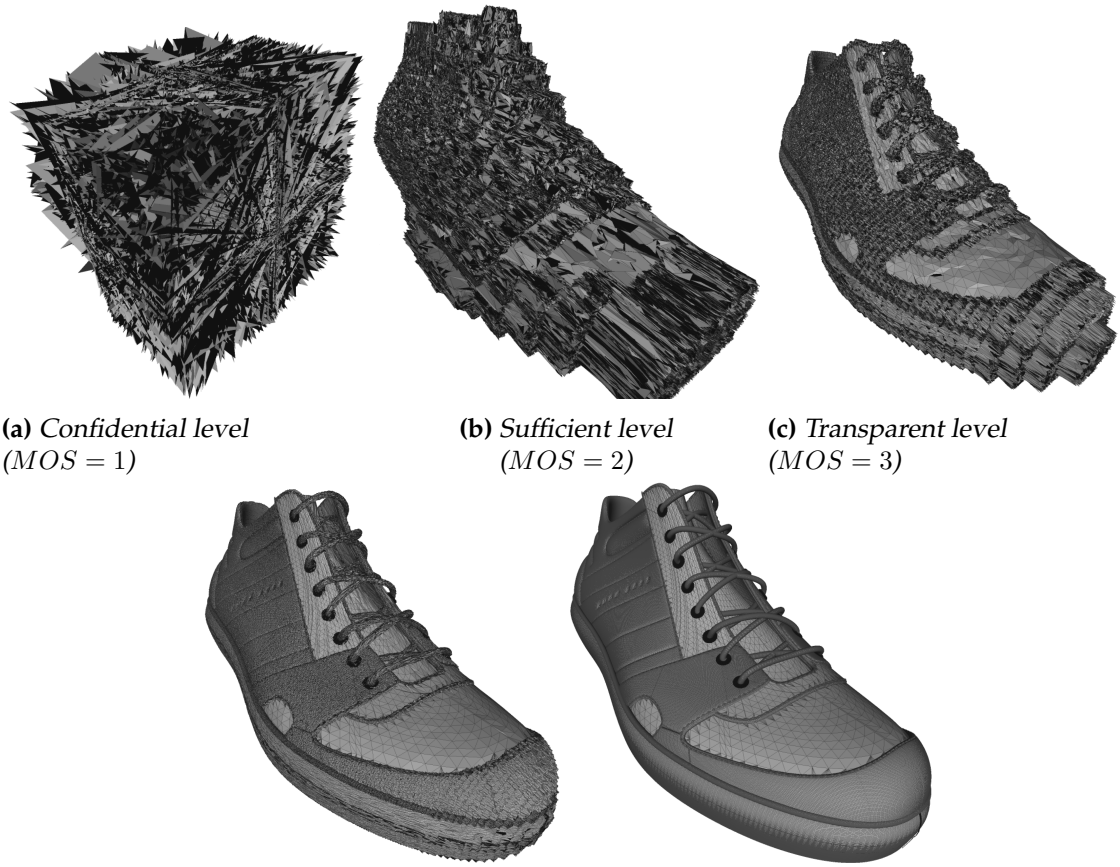


Figure 5.6: *Evaluation of the visual security of selectively encrypted 3D objects. The observer selects an opinion score (OS) value for the selectively encrypted 3D object.*



(a) *Confidential level* $(MOS = 1)$

(b) *Sufficient level* $(MOS = 2)$

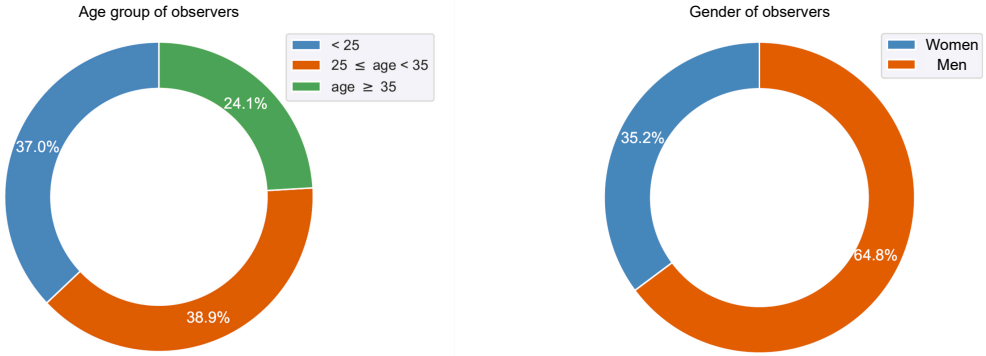
(c) *Transparent level* $(MOS = 3)$

(d) *Noisy 3D object* $(MOS = 4)$

(e) *Clear 3D object* $(MOS = 5)$

Figure 5.7: *Selectively encrypted 3D objects for the observer initiation phase at the different levels of visual security of 3D objects.*

of visual security level evaluation. The order in which the selectively encrypted 3D objects are presented is crucial, as it serves to show the evolution of the shape and content of the 3D object from confidentially encrypted into something recognizable. During this demonstration, the users are showed how the opinion score (OS) should



(a) *Age* **(b)** *Gender*

Figure 5.8: *Observer distributions by : a) Age, b) Gender.*

be assigned during each evaluation phase.

Observer Group Analysis

The group of observers who participated in the subjective assessments is diverse. It is composed of experts in the field of computer graphics, image processing, as well as other so-called non-experts.

Fig. 5.8 represents the distribution of observers by age and gender. As illustrated in Fig. 5.8.a, the observers are divided into three age groups: under 25 (20), 25-35 (21) and over 35 (13). We note that 19 observers are women and 35 are men according to Fig. 5.8.b.

5.2.4 Evaluation Analysis

The 54 observers generated 2700 opinion score (OS) values distributed over all 550 3D objects in the dataset. Among the 2700 OS, approximately 250 are those of the 3D reference objects. These scores on 3D reference objects are mainly used to identify ambiguous 3D reference objects, *i.e.* objects with a naturally distorted appearance for observers. This makes it possible to analyze the perception of the quality of 3D objects created from the digitization of real-world objects. In addition, more information is obtained on the threshold of sensitivity to distortions of different observers. The approximate 2450 other OS are those of the evaluation of selectively encrypted 3D objects. On average, each selectively encrypted 3D object has been evaluated 5 times.

In the SE3DO dataset, for each selectively encrypted 3D object, for any value of p, the assigned OS values can generally vary around two or three values. Thus, there is no specific value of p where a unique value for an OS is obtained for whatever the encrypted 3D object. Fig. 5.9 shows the distribution of the percentages of the OS values according to the parameter p. More precisely, the majority of observers gave an OS

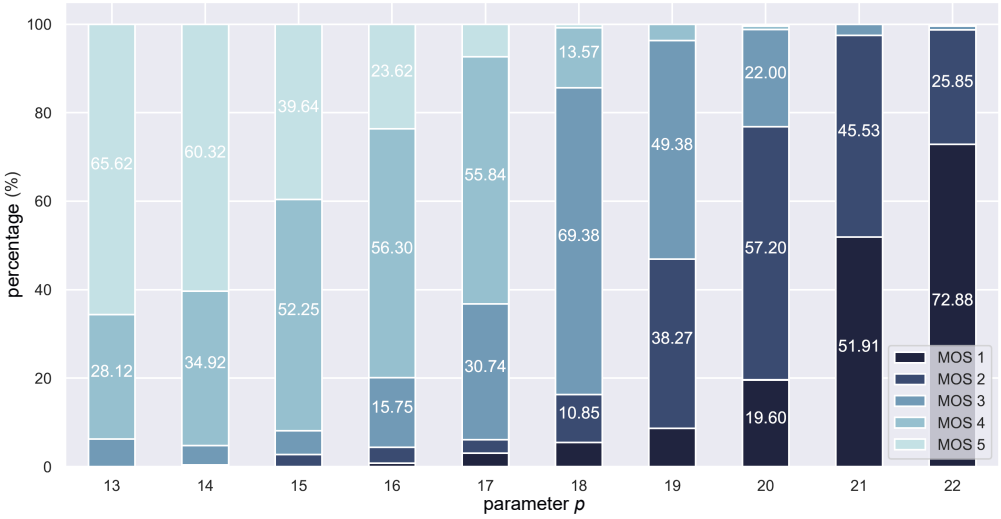


Figure 5.9: *Distribution of the percentages of the opinion score values according to the parameter p.*

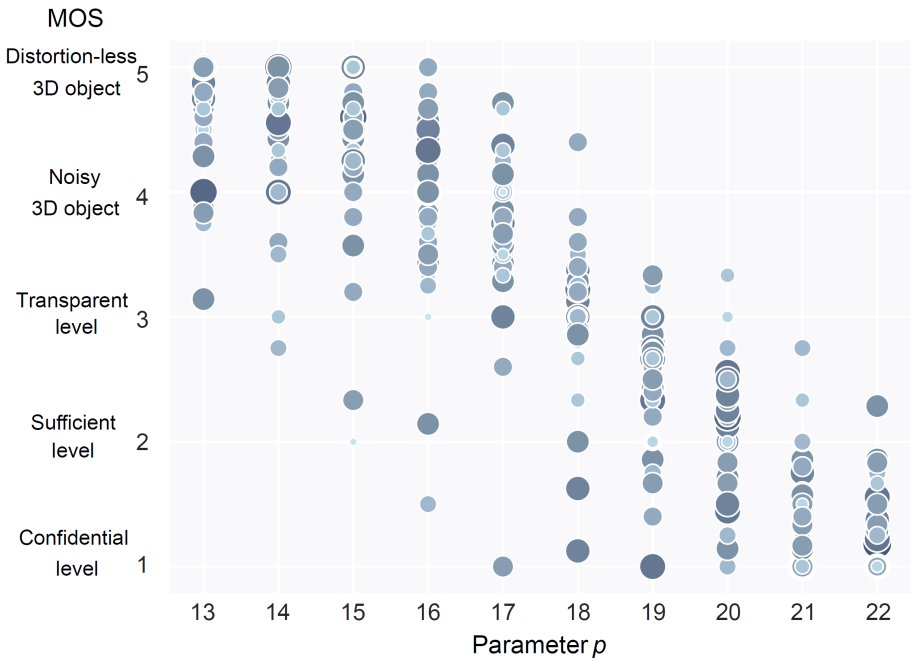


Figure 5.10: *Distribution of the MOS values for each encrypted 3D object according to the parameter p.*

of 5, for $p \in \{13,\ 14\}$ (65.62%, 60.32%), an OS of 4, for $p \in \{\ 16,\ 17\}$ (56.30%, 55.84%), an OS of 3, for $p \in \{18\}$ (69.38%), an OS of 2, for $p \in \{20\}$ (57.20%) and an OS of 1, for $p \in \{\ 22\}$ (72.88%). However, observers are more varied for $p \in \{15,\ 19,\ 21\}$. Indeed, for p =15, although observers voted for an OS of 4, with 52.25%, there are still 39.64% who chose a score of 5. A similar situation arises for $p = 19$ where no OS value exceeds 50%, with 49.38% for an OS of 3, 38.27% for an OS of 2 and 12.35% for the other OS values. Finally, for $p = 21$, an OS of 1 is 51.91%, compared to 45.53% with an OS of 2. From these results, specific intervals representing different levels of visual security start to emerge. Thus, observers consider objects selectively encrypted with a parameter p equal to 18 or 19 as transparent, 19 or 20 as sufficient and finally 21 or 22 as confidential. Despite an encryption with p equal to 13 or 14, most observers consider

that the 3D objects do not have visible geometric distortions and that they only appear from $p = 15$. To summarize, for $p \in \{13,\ 14\}$, the majority of observers preferred an OS of 5, for $p \in \{15,\ 16,\ 17\}$ an OS of 4, for $p \in \{18,\ 19\}$ an OS of 3, for $p \in \{20,\ 21\}$ an OS of 2 and finally a OS of 1 for $p \in \{22\}$. In addition, for some values of p, OS values are assigned almost uniformly between two values, particularly for $p \in \{15,\ 19,\ 21\}$. The values of p pivots, where observers consider that a visual change occurs for a large part of the selectively encrypted 3D objects. There are very rare value pairs (p, OS), but they do still exist. For example, the first evaluation giving an OS of 1 appears for $p = 16$, while for an OS of 5, the last encryption parameter p is 18.

Fig. 5.10 represents the MOS values calculated from the OS values provided by the observers for each encrypted 3D object in the SE3DO dataset. It can be observed in Fig. 5.10 that, despite the OS given mainly by observers for each value of the encryption parameter p presented in Fig. 5.9, some 3D objects are considered to be confidentially encrypted despite a low p value. Or on the contrary, 3D objects are considered to be of a transparent or sufficient level, despite a high value of p.

5.3 Security Parameter Estimation

In this section, we present our method for estimating the visual security parameter for 3D objects which we wish to selectively encrypt. An overview of the method is illustrated in Fig. 5.11. The visual security parameter is estimated using a polynomial function obtained by regression based on the desired security level $MOS_{desired} \in [1, 5]$, where 1 corresponds to a confidential level, 2 a sufficient level, and 3 a transparent level. The estimated visual security parameter p is then used to selectively encrypt the 3D object.

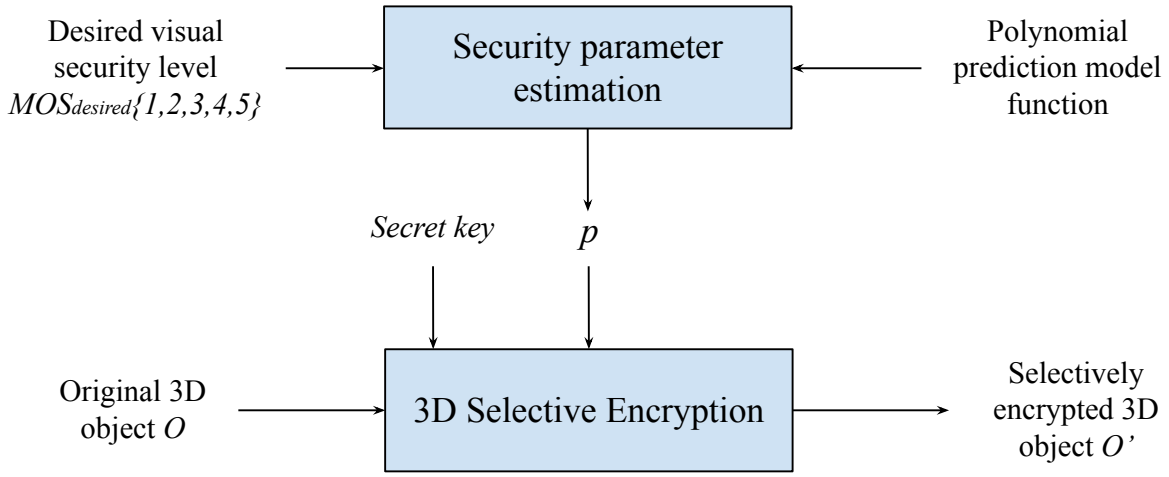


Figure 5.11: *Overview of the visual security parameter estimation.*

In order to be able to estimate the encryption parameter p, we calculate correlations between the values of p and the values of the MOS obtained from the observers. A correlation coefficient is a statistical measurement describing the linear relationship between two variables. These correlation coefficients are between -1 and $+1$. A correlation coefficient close to $+1$ indicates that the two variables have a very high positive

Correlation coefficients	Visual security parameter p
Pearson	-0.906
Spearman	-0.904

Table 5.2: *Correlation coefficients between the parameter p and the MOS values obtained from the observer evaluations of the SE3DO dataset.*

Regression metrics	Train	Test
R^2 score	0.8651	0.7443
Explained Variance Score	0.8651	0.7508
Mean Absolute Error	0.7774	1.0745
Mean Squared Error	1.1128	2.1094
Max Error	4.7248	5.2846
Median Absolute Error	0.5838	0.7303

Table 5.3: *Results of the polynomial regressions according to the MOS values of the observers on both datasets.*

linear relationship, while a correlation coefficient close to -1 shows that the two variables have a very high negative linear relationship. A correlation coefficient close to 0 indicates that the variables are independent and therefore there is no relationship between the two.

Table 5.2 presents the obtained correlation coefficients between parameter p and the MOS values of the observers. At first we chose to use the data from the evaluations in three different ways: by directly using all the values of OS given by observers (raw values), by using the median values of OS for each 3D object, and finally by using the MOS values for each 3D object. The first approach (raw values) uses all the OS given (approximately 2450 evaluations) which makes it possible to calculate a value as close as possible to reality. The other two approaches (median and mean values) use the OS values assigned to the 500 selectively encrypted 3D objects. We find that there is a strong relationship between the parameter p and the OS values of the observers. With this analysis, we observe that the parameter p is strongly correlated to the OS values of the observers. As a result, we can build a model to estimate the value of p based on a desired level of visual security. To do this, we have decided to apply a polynomial regression in order to build a statistical learning model. So, we separate the data from the SE3DO dataset into two distinct 3D object datasets, namely a 3D object dataset for the training phase and a 3D object dataset for the test phase. The goal is to train the model on a representative subset of the data and test the validity of the model on the rest of the 3D objects, which have never been observed by the model. To do this, we use 30 3D reference objects (and their 300 associated encrypted versions) for the training phase and 20 3D reference objects (and their 200 associated encrypted versions) for the testing phase.

Table 5.3 presents the results of the estimation models of the parameter p as a function of the MOS values for the training base and the test base. We test our model on the MOS obtained for each 3D object. We first calculate the determination coefficient

Parameter p	13	**14**	15	**16**	17	**18**	**19**	20	21	**22**
Observed MOS	5.0	5.0	4.7	4.0	3.6	3.2	2.6	1.0	1.0	1.0
$f(D)$	13.55	13.55	14.06	15.66	16.49	17.63	19.00	21.61	21.61	21.61
$[f(D)]$	14	**14**	14	**16**	16	**18**	**19**	22	22	**22**

Table 5.4: *Estimations of the parameter p according to the desired level of visual security for the 3D object #18 shown in Fig. 5.4.*

(or R^2 score):

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=0}^{n-1} (y_i - \hat{y}_i)^2}{\sum_{i=0}^{n-1} (y_i - \overline{y})^2}, \tag{5.1}$$

where y is the score vector of the field truth, $\hat{y}$ the score vector obtained by the model and $\overline{y}$ the mean of the scores of y.

The R^2 score gives information about the quality of the model, for example a model giving the right predictions without taking into account the input data receives a score of 0.0. The score can become negative if the model is bad, while a model giving good results by taking into account the input data has a score that tends towards 1.0.

The explained variance score is a metric used to evaluate the quality of predictions based on a relationship between the difference in variances of the prediction and the field truth:

$$EVS(y, \hat{y}) = 1 - \frac{\text{variance}(y - \hat{y})}{\text{variance}(y)}, \tag{5.2}$$

where variance is the square of the standard deviation for y and $\hat{y}$, respectively variance(y) and variance$(\hat{y})$.

We also calculate the mean absolute error (Mean-AE), the mean squared error (MSE), the maximum absolute error (Maximum-AE), and the median absolute error (Median-AE).

Table 5.3 presents the results when we train our model with the MOS values to estimate the value of p according to a $MOS_{desired}$ between 1 and 5. Indeed, the R^2 score has a value around 0.8651. In addition, we note that the median absolute error is only 0.5832, which means that the estimated values of p are mostly close to what is expected. Table 5.3 also presents the results of the estimation models of the parameter p as a function of MOS for the test data. We observe a decrease in scores during the test phase. Indeed, the best results obtained, with the mean values, are 0.7443 compared to 0.8651 during the training phase for the R^2 score and the explained variance score is about 0.7508, compared to 0.8651 during the training phase. These results show the robustness of the model using the MOS values for learning.

In Fig. 5.12, we present the polynomial regressions for the estimation of p as a function of the desired visual security level and the MOS values of the 3D object used to train the model. The blue curve represents the obtained polynomial following the polynomial regression for each model. The size of the markers and their color correspond to the absolute error of the estimation, so the smaller and darker the circle, the higher

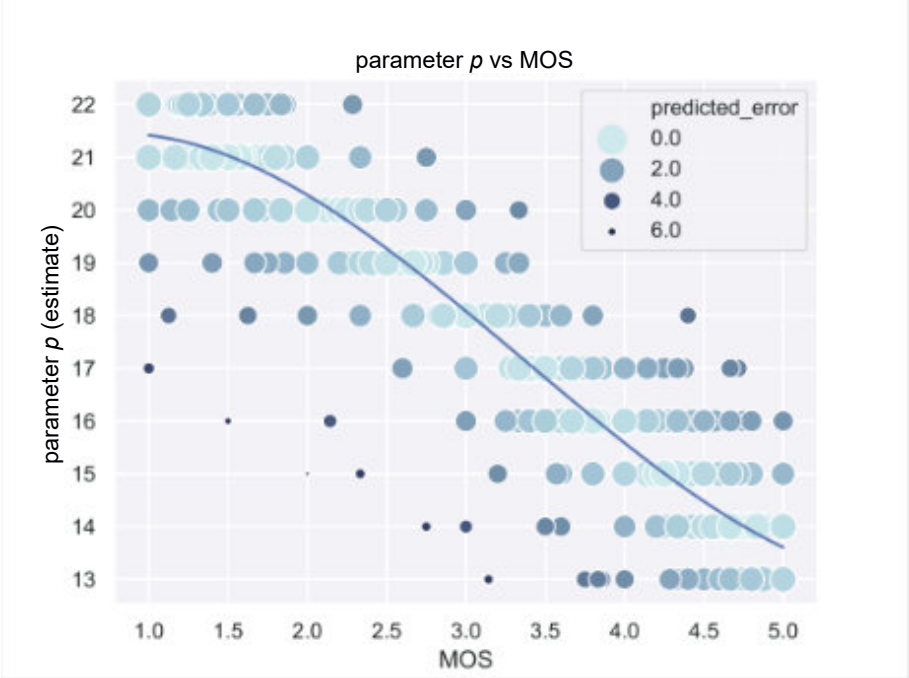


Figure 5.12: *Polynomial regressions for estimating the visual security parameter p.*

the error. The coefficients of the polynomial obtained by regression are:

$$f(D) = 21.0404 + \begin{bmatrix} \mathrm{D} & \mathrm{D}^2 & \mathrm{D}^3 \end{bmatrix} \times \begin{bmatrix} 1.6931 \\ -1.2442 \\ 0.1212 \end{bmatrix}. \tag{5.3}$$

So, thanks to our subjective evaluations we were able to establish a model for the estimation of the encryption parameter p according to a desired level of visual security. We note that the best way to estimate p is obtained with a model using MOS values due to the high scores for R^2 and the explained variance, this method also benefits from low errors, as is presented in Table 5.3.

Table 5.4 compares the results obtained for the estimations of the parameter p with the model $f(\cdot)$ according to the MOS values for the 3D object #18 of the SE3DO dataset and its 10 encrypted versions with p ranging from 13 to 22, as shown in Fig. 5.4. We find that we can estimate an encryption parameter p that is relatively close to the one used during the construction of the dataset from the desired level of visual security. The various estimations give slightly lower values for p when the desired level of visual security is greater than or equal to 3. From a desired visual security level of 2, the estimated parameters are higher than the expected encryption parameter. So, our models are able to offer adapted values for the encryption parameter, especially when the desired visual security score is less than or equal to 2. Indeed, the estimated value of p is greater than the expected value.

In this section, we have shown that it is possible to estimate the parameter p from a desired visual security level. We can therefore automatically propose a value for this encryption parameter.

5.4 The Proposed 3DVS Score

In this section, we develop our proposed regression based 3D Visual Security (3DVS) score, illustrated in Fig. 5.13. In Section 5.4.1, we detail the correlation between the MOS values and our different metrics. Then in Section 5.4.2, we construct a regression model based on the MOS values. In Section 5.4.3, we present the construction of our 3DVS score based on regression, and finally in Section 5.4.4 we apply the proposed 3DVS score to another 3D selective encryption method in order to verify its effectiveness.

Still based on our subjective evaluations, we can now build a metric to estimate the level of visual security of selectively encrypted 3D objects as a score. As explained in Chapter 1, estimating the visual security level is very different to estimating the visual quality. Each 3D object in the dataset is studied using the objective metrics with full reference. Each object is then compared to its reference 3D object. Then we can analyze the efficiency of the metrics used to study the visual security level of selectively encrypted 3D objects.

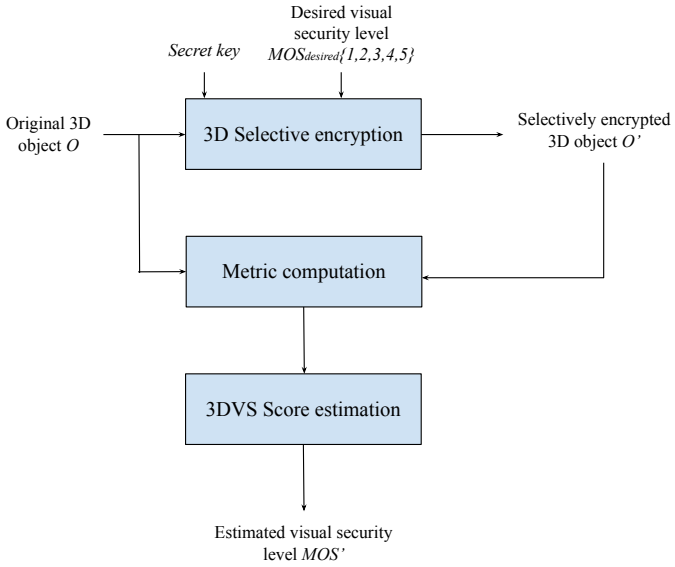


Figure 5.13: *Overview of the proposed 3D-Visual security (3DVS) score.*

5.4.1 Correlation

Table 5.5 shows the correlation coefficients calculated for the objective metrics log(RMSE), log(HD), DAME, MSDM2 and PSNR, when using the MOS values of each 3D object of the SE3DO dataset.

We can clearly see in Table 5.5 that the PSNR metric has the highest correlation with a Pearson value of 0.903 and a Spearman value of 0.930. The log(RMSE), log(HD) and MSDM2 metrics also have interesting correlation coefficients (above 0.70 in absolute terms), with the log(RMSE) being the highest. Only the DAME metric is totally independent of the scores given by the observers. We suspect that this is because the DAME

	Objective metrics				
Correlation coefficients	log(RMSE)	log(HD)	DAME	MSDM2	PSNR
Pearson	-0.775	-0.743	-0.032	-0.757	0.903
Spearman	-0.815	-0.794	-0.116	-0.776	0.930

Table 5.5: *Correlation coefficients between the observers' MOS values and objective metrics.*

metric is based on the mean of the differences in dihedral angles weighted by the area of the triangles. Seeing as how the geometric positions of the vertices vary greatly, we can assume that the areas of the triangles formed by these vertices also vary greatly.

5.4.2 Regression Model Construction

We notice that the MOS values are distributed in the form of a sigmoid function, in particular in relation to the PSNR which has the highest correlation with the MOS out of all the full reference metrics used. It is for this reason that we wish to construct our linear regression model using a sigmoid function. We therefore use a combination of classic linear regression and logistic regression. We do this by fitting a sigmoid function to the data, without classifying the data into binary categories.

In order to fit the sigmoid, and consequently construct our model, the MOS values which vary between 1 and 5 have to be mapped to values between 0 and 1:

$$y' = \frac{(y-1)}{4}, \tag{5.4}$$

where y is the original MOS and y' the mapped MOS.

The input data is normalised in order for it to be possible to construct a multi-feature regression model. The sigmoid function is then fit to the data in the same way as is logistic regression:

$$\hat{y'} = \frac{1}{1+\exp^{-z}}, \tag{5.5}$$

where $\hat{y'}$ is the output value of our model, $z = w_0 + w_1 \times x_0 + w_2 \times x_1 ...$, with w the weights and x the features, and $w_i \in w$ and $x_i \in x$.

Instead of interpreting $\hat{y'}$ as a percentage likelihood as we would in logistic regression, we convert $\hat{y'}$ to a value between 1 and 5, which represents the MOS estimated by our model:

$$\hat{y} = 1 + 4 \times \frac{1}{1+\exp^{-z}}. \tag{5.6}$$

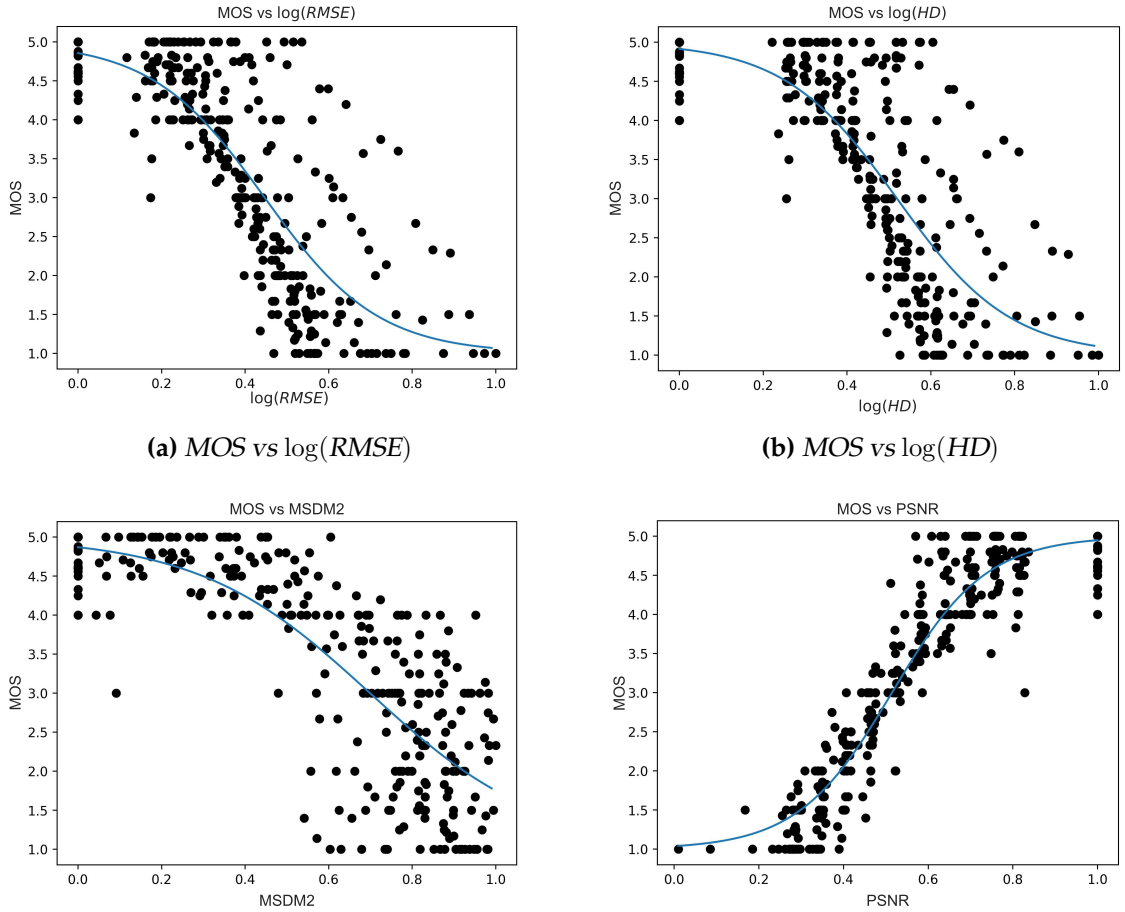
5.4.3 Construction of the Proposed 3DVS Score

We use the sigmoid-based regression model described in Section 5.4.2 to construct the proposed 3DVS score. Fig. 5.14 illustrates the sigmoid-based regression model fit to the MOS values for each 3D object of the SE3DO dataset that we have used to train our

	3D metrics							
	Training				Testing			
Regression metrics	log(RMSE)	log(HD)	MSDM2	PSNR	log(RMSE)	log(HD)	MSDM2	PSNR
R^2 score	0.6450	0.5971	0.6482	**0.8991**	0.6641	0.6533	0.5582	**0.9166**
Explained Variance Score	0.6450	0.5971	0.6482	**0.8991**	0.6696	0.6653	0.5583	**0.9197**
Mean Absolute Error	0.1509	0.1627	0.1527	**0.0794**	0.1435	0.1490	0.1693	**0.0733**
Mean Squared Error	0.0395	0.0449	0.0390	**0.0112**	0.0363	0.0375	0.0477	**0.0090**
Max Error	0.6578	0.6521	0.6155	**0.4439**	0.7850	0.7437	0.6346	**0.2808**
Median Absolute Error	0.1180	0.1232	0.1302	**0.0629**	0.1137	0.1290	0.1451	**0.0572**

Table 5.6: *Results of the polynomial regressions of the selected metrics for the MOS values.*

model. Visually, we observe that the curves closely fit the given data, especially in the case of the PSNR.



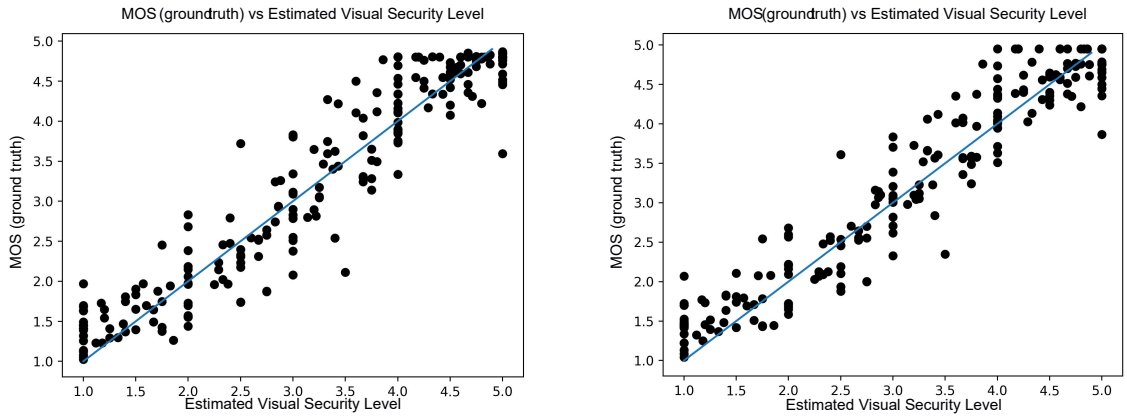
(a) *MOS vs* $\log(RMSE)$ **(b)** *MOS vs* $\log(HD)$

(c) *MOS vs MSDM2* **(d)** *MOS vs PSNR*

Figure 5.14: *Distribution and regression of the MOS values for each encrypted 3D object according to the metric values* $\log(RMSE)$, $\log(HD)$, *MSDM2 and PSNR.*

In Table 5.6, we present the different regression metric scores obtained from the training and test phases of the regression models constructed with $\log(\text{RMSE})$, $\log(\text{HD})$, PSNR and MSDM2. We note that while the PSNR has the best scores for R^2 with 0.9166 and for the explained variance with 0.9197, the other three metrics produce interesting results. Therefore, our metric is largely based on the PSNR, but we use the other three metrics in order to render the 3DVS score more robust.

Fig. 5.15a shows the calculated visual security levels in relation to the ground truth MOS values of the 3D objects of the test dataset (20 reference 3D objects and their 200 variations) according to a regression based on $\log(\text{RMSE})$, $\log(\text{HD})$, PSNR and



(a) **(b)**

Figure 5.15: *Results from the measurement of the visual security levels from a regression based on the metrics: a)* $\log(RMSE)$, $\log(HD)$, *PSNR and MSDM2, b) PSNR and log(RMSE).*

MSDM2. Visually, the determined visual security levels seem to correspond well to the MOS values of the observers. This model, noted g_{All}, can be formulated as:

$$\begin{cases} g_{All}(O,O') = \alpha + \beta \times \text{PSNR}(O,O') \\ +\gamma \times log(\text{RMSE}(O,O')) + \delta \times log(\text{HD}(O,O')) \\ +\eta \times \text{MDSM2}(O,O'), \\ \alpha = -2.4087, \\ \beta = 6.8352, \\ \gamma = -1.3183, \\ \delta = 0.4188, \\ \eta = -1.0538. \end{cases} \tag{5.7}$$

Fig. 5.15b illustrates the determined visual security levels in relation to the ground truth MOS values of the 3D objects of the test base (20 reference 3D objects and their 200 variations) according to a regression based only on the metrics PSNR and log(RMSE). Visually, the determined visual security levels seem to correspond well to the MOS values of the observers. So, this second model, noted $g_{\text{PSNR,log(RMSE)}}$ can be formulated as:

$$\begin{cases} g_{\text{PSNR},log(\text{RMSE})}(O,O') = \alpha + \beta \times \text{PSNR}(O,O') \\ +\gamma \times log(\text{RMSE}(O,O'), \\ \alpha = -4.0341, \\ \beta = 8.3373, \\ \gamma = -0.5585. \end{cases} \tag{5.8}$$

Table 5.7 presents the results of the regression constructed with the four metrics, as well as those constructed with only the PSNR and log(RMSE). We observe that the regression using the PSNR and log(RMSE) gives the best all round results during the testing phase, but in order for our model to be as robust as possible, we can also construct our 3DVS score using g_{All} as it produces similar results.

Table 5.8 details the visual security level estimated with the proposed regression model for the object #18 from the SE3DO dataset, based on the different full reference metrics. We note that it is mostly the PSNR as well as the combined PSNR and log(RMSE) that produce the most accurate results.

	3D metrics			
	Training		Test	
Regression metrics	g_{All}	$g_{\text{PSNR,RMSE}}$	g_{All}	$g_{\text{PSNR,RMSE}}$
R^2 score	**0.8993**	0.8966	0.9127	**0.9168**
Explained Variance Score	**0.8993**	0.8966	0.9142	**0.9201**
Mean Absolute Error	**0.0792**	0.0801	0.0754	**0.0728**
Mean Squared Error	**0.0112**	0.0115	0.0094	**0.0090**
Max Error	0.4540	**0.4412**	0.3192	**0.2833**
Median Absolute Error	**0.0624**	0.0641	0.0596	**0.0589**

Table 5.7: *Results of the regression constructed with the MOS values of the observers and the different combinations of metrics.*

	Observed MOS						
p	MOS	$PSNR$	$MSDM2$	$log(RMSE)$	$log(HD)$	All	$PSNR/RMSE$
0	5.00	**4.95**	4.87	4.86	4.92	**4.95**	**4.95**
13	5.00	**4.75**	4.27	4.49	4.46	4.73	**4.74**
14	5.00	**4.59**	3.86	4.34	4.33	4.54	**4.58**
15	4.71	**4.35**	3.32	4.15	4.15	4.26	**4.34**
16	4.00	**4.01**	2.73	3.92	3.94	3.85	**4.01**
17	3.67	3.57	2.28	**3.66**	3.73	3.35	**3.58**
18	3.22	3.06	2.09	**3.37**	3.44	2.87	**3.08**
19	2.40	**2.54**	2.03	3.06	3.09	**2.43**	2.58
20	1.00	2.08	**2.01**	2.76	2.83	**2.05**	2.12
21	1.00	**1.71**	2.07	2.46	2.50	**1.76**	**1.76**
22	1.00	**1.45**	2.11	2.17	2.26	1.52	**1.48**

Table 5.8: *Determined visual security levels based on the MOS values and the metrics for the 3D object #18 shown in Figure 5.4.*

5.4.4 Application of the Proposed 3DVS Score to Another 3D Selective Encryption Method

In this section, in order to verify the effectiveness of the proposed 3DVS score, we propose to apply it to another 3D selective encryption method. Indeed, while the proposed dataset is constructed using the selective encryption method developed by Beugnon *et al.* (47), the proposed 3DVS score is able to evaluate the visual security level of 3D objects selectively encrypted using other methods.

Another 3D selective encryption method is used to perform a comparison. This selective encryption method encrypts the vertices of a 3D object by adding pseudo-random values to the three coordinates of each vertex $v_i\{x_i, y_i, z_i\}$. These pseudo-random values have a Gaussian distribution centered in 0 with a standard deviation σ.

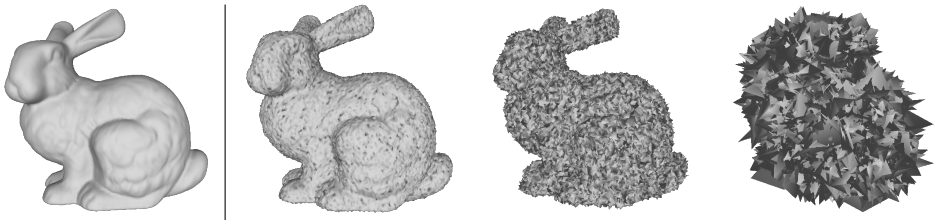
From a 3D object O which is composed of V vertices v_i, a selective encrypted 3D object O' is obtained by adding a pseudo-random Gaussian noise to each vertex:

$$v'_i = E_K(v_i), \tag{5.9}$$

where $E_K()$ is the encryption function, with K the secret key and $i \in [0, V-1]$ such as:

$$\begin{cases} x'_i = x_i + \mathcal{N}_{K,\sigma}(i \times 3) \\ y'_i = y_i + \mathcal{N}_{K,\sigma}(i \times 3 + 1) \\ z'_i = z_i + \mathcal{N}_{K,\sigma}(i \times 3 + 2), \end{cases} \tag{5.10}$$

where $\mathcal{N}_{K,\sigma}()$ is a pseudo-random Gaussian number generator with a standard deviation σ and based on the secret key K. This method is fully reversible. For the decryption, the same sequence of pseudo-random values is generated again using the same key, and are subtracted from the encrypted 3D object. The level of selective encryption depends on the value of σ used for the Gaussian distribution.



(a) Original (b) Transparent (c) Sufficient (d) Confidential

Figure 5.16: *Selectively encrypted 3D object Bunny (Object #44) with the encryption method based on a pseudo-random Gaussian number generator.*

Fig. 5.16 presents the 3D object *Bunny*, which is selectively encrypted with this method. The visual security level is increased by increasing the standard deviation σ of the pseudo-random Gaussian number generator. Fig. 5.16.a. presents the original 3D object included in the proposed dataset. Fig. 5.16.b. illustrates the selectively encrypted 3D object when using a standard deviation σ of 3×10^{-3}. In this case, the obtained 3DVS score is $[3.163] = 3$, which corresponds to a transparent level. Fig. 5.16.c. illustrates the selectively encrypted 3D object with a standard deviation σ of 10×10^{-3}. In this case, the obtained 3DVS score is $[1.657] = 2$, corresponding to a sufficient level. Finally, Fig. 5.16.d. illustrates the selectively encrypted 3D object with a standard deviation $\sigma = 100 \times 10^{-3}$. In this case the obtained 3DVS score is 1, meaning that we have a confidential level.

These experimental results confirm that our 3DVS score performs well on 3D objects which were selectively encrypted with another encryption method.

5.5 The Proposed 3D Object Encryption Method

In this section, we detail our proposed encryption scheme for 3D objects which allows a hierarchical decryption. We consider two levels of visual security for the hierarchically decrypted 3D objects: the **transparent level** and the **sufficient level**. The fully

encrypted **confidential** level 3D object can also be fully decrypted, so that the original 3D object is retrieved. We note that the original 3D object has a **clear level**. Fig. 5.17 presents an overview of the encryption phase of the proposed method. From a 3D object and a secret key, called the master key, the proposed method generates a fully encrypted 3D object and a ring of hierarchical keys. With these keys, a hierarchical decryption can be performed, whose strength depends on that of the key.

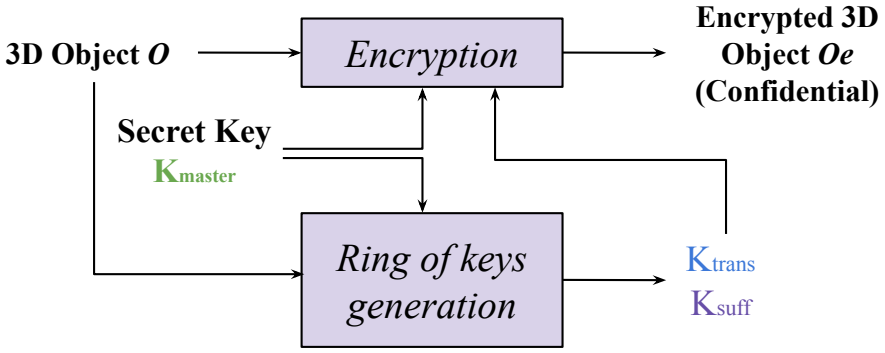


Figure 5.17: *Overview of the encryption phase of a 3D object.*

First, in Section 5.5.1, we show how we divide the original 3D object's vertices into blocks, where the bits of these blocks are sorted from the most significant bits (MSB) to the least significant bits (LSB). The blocks are then divided into sub blocks, where each sub block is to be encrypted and subsequently decrypted with a key of a different hierarchical level. Then, in Section 5.5.2, we detail the key generation process where the hierarchical keys are created according to the encryption performed with its hierarchically superior key. Here we define three keys K_{master}, K_{trans} and K_{suff} which allow us to obtain the fully decrypted original 3D object which has a clear level, or a selectively encrypted 3D object with either the transparent or sufficient visual security level. In Section 5.5.3, we describe the encryption process where we use the Advanced Encryption Standard (AES) scheme with a cipher feedback (CFB) mode to encrypt each of the sub blocks. Finally, in Section 5.5.4, we present the decryption process where some or all sub blocks are decrypted depending on the hierarchical level of the key that is used.

5.5.1 Block Generation

We note O the original 3D object. As discussed in Chapter 1, a 3D object can be represented by a set of vertices $\mathcal{V} = \{v_0, ..., v_{|\mathcal{V}|-1}\}$ and faces $\mathcal{F} = \{f_0, ..., f_{|\mathcal{F}|-1}\}$, where $\mathcal{F}$ describes the 3D object's connectivity. Each vertex $v \in \mathcal{V}$ consists of three coordinates x, y and z, where each of these can be represented by a 32-bit floating point.

According to the IEEE 754 standard, a 32-bit floating point $f \in \{x, y, z\}$ consists of a sign s represented with 1 bit, an exponent e represented with 8 bits and a mantissa m represented with 23 bits (from MSB to LSB) where:

$$f = (-1)^s \times m \times 2^{e-127}. \tag{5.11}$$

Fig. 5.18 illustrates how a 32-bit floating point f is divided into s, e and m.

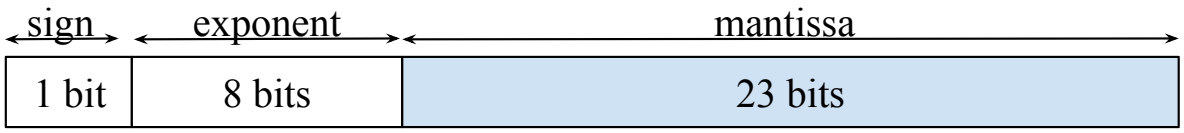


Figure 5.18: *Representation of a 32-bit floating point according to the IEEE 754 standard.*

As presented in Chapter 1, Beugnon *et al.* (47) described a selective encryption method where only certain bits of the mantissa of each of the coordinates are encrypted. Likewise, in our proposed method, we only consider the mantissa. We aim to divide the mantissa into sub blocks, with α bits per sub block within the mantissa, as shown in Fig. 5.19. Each sub block corresponds to a different hierarchical level to be encrypted with its respective key.

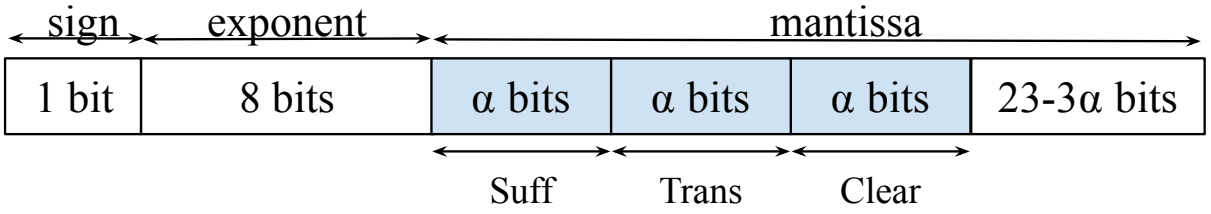


Figure 5.19: *The bits of the mantissa are divided into sub blocks.*

The size of α, and therefore the size of each sub block, determine the strength of the encryption. For example, if $\alpha = 0$, then no encryption takes place. To apply the weakest encryption possible, we take the value $\alpha = 1$. We note that $\alpha \leq \lfloor \frac{23}{3} \rfloor = 7$.

However, we wish to do this for a block of vertices, in order to construct a square block of 16 bytes (128 bits) for the AES encryption scheme described in Chapter 1. A 3D object O contains N vertices, and therefore $3N$ coordinates. These $3N$ coordinates are grouped into M blocks B_i where $i \in [0, M-1]$. Each block therefore consists of h bits:

$$h = 3n \times 3\alpha \text{ bits}, \tag{5.12}$$

where $n = \frac{N}{M}$ is the number of vertices per block. These bits b_{ij} of the block B_i, where $j \in [0, h-1]$, are then sorted from the most significant bits (MSB) to the least significant bits (LSB) as shown in Fig. 5.20.

The block B_i is then sliced into sub blocks B_{i_suff}, B_{i_trans} and B_{i_master}, as illustrated in Fig. 5.21. The block B_i is sorted and sliced in this manner so that the sub block B_{i_suff} contains the most significant bits of each coordinate, and B_{i_master} contains the least significant bits of each coordinate. Therefore, each vertex is encrypted in the same manner. We note α_B the number of bits per sub block:

$$\alpha_B = 3n \times \alpha. \tag{5.13}$$

Therefore, B_{i_suff} is comprised of the α_B MSB of B_i. We impose the constraint:

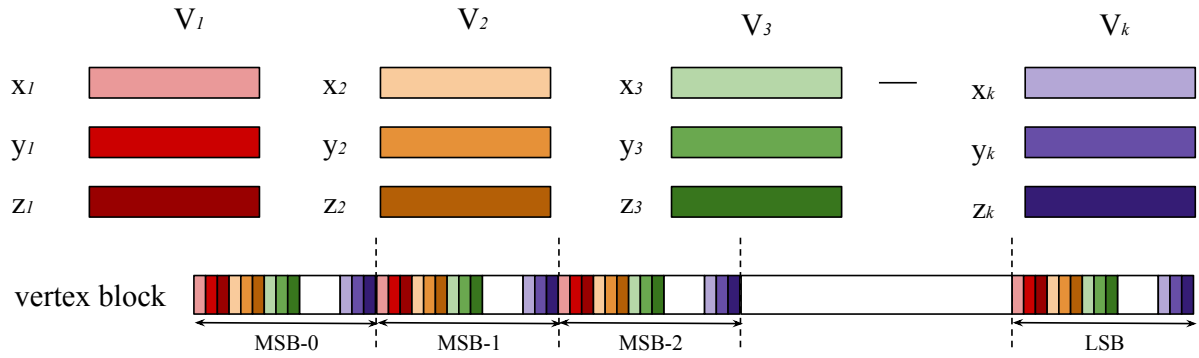


Figure 5.20: *The bits of a block B_i are sorted from the MSB to the LSB.*

$$\alpha_B \leq 128 \text{ bits}, \tag{5.14}$$

so that each sub block can be padded to correspond to an Advanced Encryption Standard (AES) block which is described in Section 5.5.3.

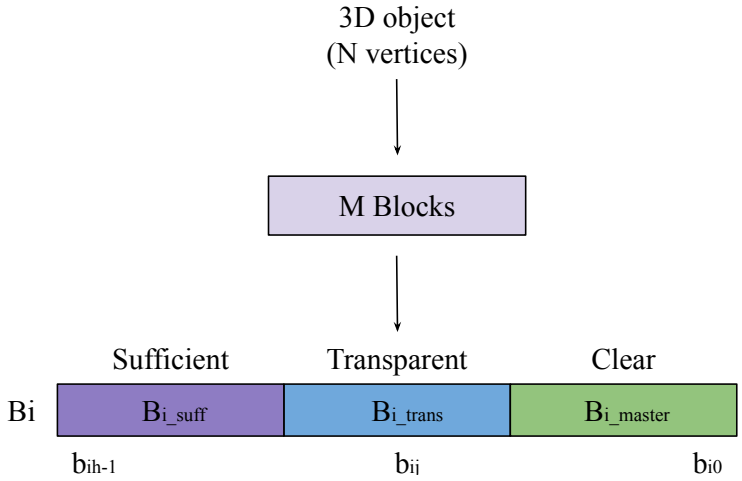


Figure 5.21: *Division of M blocks into sub blocks of size α_B.*

We can also redefine h:

$$h = 3\alpha_B \text{ bits}. \tag{5.15}$$

5.5.2 Key Ring Generation

In this work, we propose a 3D object encryption method which allows for hierarchical decryption. In order to obtain a selectively encrypted 3D object with the desired security level, the fully encrypted 3D object can be hierarchically decrypted using a secret key taken from a ring of hierarchical keys. We consider two levels of visual security: the transparent level and the sufficient level. The encrypted 3D object can also be fully decrypted in order to retrieve the original clear level 3D object. We therefore have three different keys: the master key K_{master} which fully decrypts the confidential 3D object in order to retrieve the original clear level 3D object, K_{trans} which results in a transparent level selectively encrypted 3D object, and finally K_{suff} which results in a sufficient level selectively encrypted 3D object. We define the hierarchy thus:

$$K_{master} > K_{trans} > K_{suff}. \tag{5.16}$$

The key generation occurs twice; once during the encryption phase and once during the decryption phase. During the encryption phase, all the keys are generated and each key K_x is used to encrypt its corresponding sub block B_{i_x} where $x \in master, trans, suff$. Encrypting each sub block with its corresponding key is what allows a hierarchical decryption. A user possessing the transparent key K_{trans} is not able to decrypt the block B_{i_master} and consequently is not able to retrieve the original clear level 3D object, only a transparent level 3D object. This process is described in Section 5.5.3. During the decryption phase, a user owns a single key K_x and therefore the inferior keys are necessary and need to be generated. With this key K_x the user can decrypt the corresponding sub block B_{i_x}, and thanks to the generated inferior keys, the inferior blocks B_{i_y} can also be decrypted, where $x \in master, trans, suff$ and $y < x$ according to the hierarchy defined in Eq. 5.16, where $y = x - 1$ or $y = x - 2$. This process is further detailed in Section 5.5.4. It is for this reason that we impose the constraint: a key K_x is able to generate its hierarchically inferior key K_{x-1} but cannot generate its hierarchically superior key K_{x+1}.

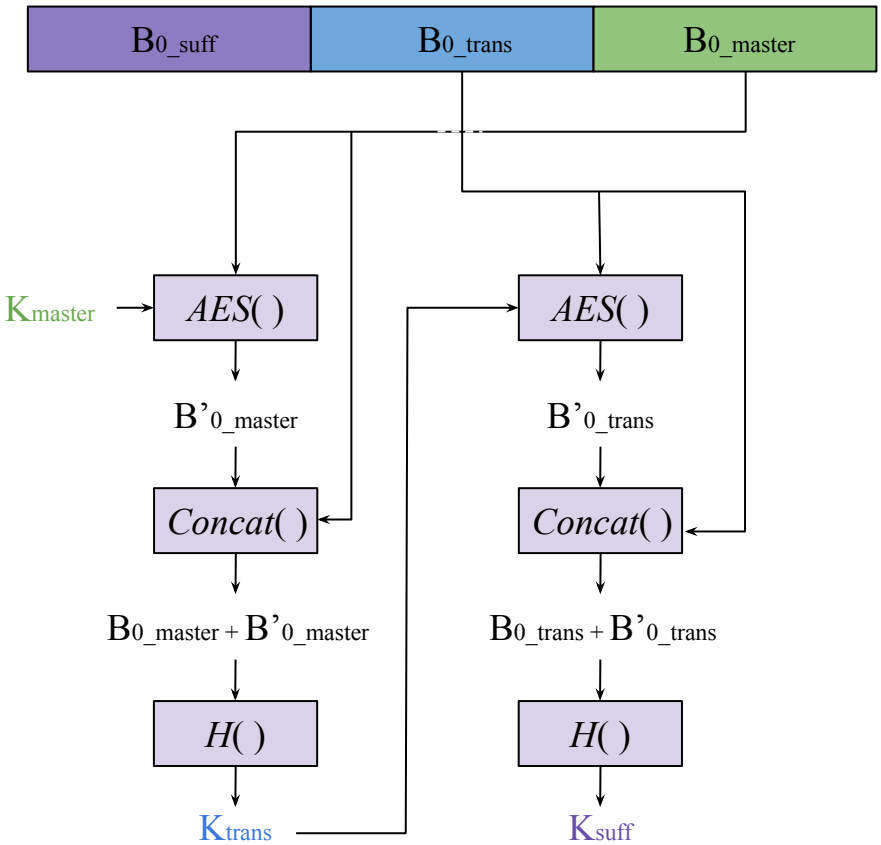


Figure 5.22: *Overview of the key generation process.*

Fig. 5.22 illustrates the key ring generation process. During this process, the hierarchically superior block and its encryption are concatenated and processed by a hash function in order to generate the hierarchically inferior key. This concatenation allows a single master key to be used to encrypt and hierarchically decrypt many 3D objects,

while the transparent key and the sufficient key are unique to each 3D object. This improves the security of the method, as a user possessing a sufficient key may have the right to access a single sufficient level 3D object. As the user's sufficient key is unique to the 3D object, this user is unable to use their key to hierarchically decrypt other 3D objects encrypted with the same master key used to generate the confidential 3D object. A hash function is used to generate the hierarchically inferior keys, because by definition of a hash function, it is irreversible. This guarantees that the hierarchically superior keys cannot be retrieved from the inferior keys. The hierarchically superior keys, and consequently the 3D objects with hierarchically superior visual security level, are therefore secure against attackers possessing hierarchically inferior keys.

We note that K_{master} is the original 256 bit secret key provided by the user during the encryption process, which serves as the master key. In order to generate the transparent key K_{trans}, we first encrypt the first clear sub block B_{0_master}:

$$B'_{0\_master} = \mathcal{E}_{K_{master}}(B_{0\_master}), \tag{5.17}$$

where $\mathcal{E}$ is the AES encryption function and K_{master} is the secret key used to perform the encryption.

B_{0_master} and B'_{0_master} are then concatenated. This concatenated block is then processed with a hash function, which provides the 256 bit transparent key K_{trans}:

$$K_{trans} = H(B_{0\_master} + B'_{0\_master}), \tag{5.18}$$

where H is a hash function. We note that the choice of the hash function is left to the user, provided that it is capable of generating a key with a minimum size of 256 bits. As previously discussed, by definition of a hash function, it is irreversible. This guarantees that the hierarchically superior keys cannot be retrieved from the inferior keys. In our experimental results in Section 5.6, we have used the hash function SHA-3 (152). Likewise, B'_{0_trans} is generated by:

$$B'_{0\_trans} = \mathcal{E}_{K_{trans}}(B_{0\_trans}), \tag{5.19}$$

where $\mathcal{E}$ is the AES encryption function and K_{trans} is the transparent key used to perform the encryption.

K_{suff} can then be generated from a hash of the concatenation of the sub block B_{0_trans} and its encrypted B'_{0_trans}:

$$K_{suff} = H(B_{0\_trans} + B'_{0\_trans}), \tag{5.20}$$

where H is a hash function.

For the encryption and decryption, we use the cipher feeback (CFB) mode so therefore a 128 bit initialization vector is also required. Each sub block B_{i_x} has its own corresponding initialization vector IV_x. This IV_x serves to initialize the AES encryption function and is used in the place of a previous sub block in the case of the first sub block. Like the master key K_{master}, IV_{master} is the original secret initialization vector

given by the user during the encryption process. The initialization vectors IV_{trans} and IV_{suff} are generated in a similar way to the keys K_{trans} and K_{suff}. To generate the inferior initialization vector IV_{x-1}, the initialization vector IV_x is concatenated with its encryption and then processed by a hash function. The encrypted initialization vectors are calculated. IV_{master} is encrypted with the AES encryption function:

$$IV'_{master} = \mathcal{E}_{K_{master}}(IV_{master}), \tag{5.21}$$

where $\mathcal{E}$ is the AES encryption function and K_{master} is the master key used to perform the encryption.

The concatenation is used as an input to a hash function, where IV_x is the first 128 bits of the output. The transparent level initialization vector IV_{trans} is given by:

$$IV_{trans} = H(IV_{master} + IV'_{master}). \tag{5.22}$$

Likewise, the sufficient level IV IV_{suff} can then be generated by first encrypting IV_{trans}:

$$IV'_{trans} = \mathcal{E}_{K_{trans}}(IV_{trans}), \tag{5.23}$$

where $\mathcal{E}$ is the AES encryption function and K_{trans} is the transparent key used to perform the encryption.

IV_{suff} is then given by:

$$IV_{suff} = H(IV_{trans} + IV'_{trans}). \tag{5.24}$$

5.5.3 Encryption

We use the AES algorithm to encrypt each sub block B_{i_x} where $x \in master, trans, suff$. For security reasons, we have chosen to use the Cipher Feedback (CFB) mode. We note that with this mode, only the AES encryption function, and not the decryption function, is used.

In our proposed method, each sub block B_{i_x} is encrypted using its corresponding 256 bit key K_x. Fig. 5.23 illustrates the encryption of a block B_i. First, the previous encrypted sub blocks B'_{i-1_x} are padded if needed, so that each has a size of 128 bits, in order to form a square block for the AES encryption scheme. This is possible because of the constraint imposed in Eq. 5.14. Each of these padded sub blocks B'_{i-1_x} are then encrypted using the AES scheme:

$$B''_{i-1\_x} = \mathcal{E}_{K_x}(pad(B'_{i-1\_x})), \tag{5.25}$$

where $\mathcal{E}_{K_x}$ is the AES encryption function which uses the key K_x and $x \in \{master, trans, suff\}$, and pad the function which pads the blocks so that it contains 128 bits. We note that when $i = 0$, B'_{i-1_x} is replaced by its corresponding 128 bit initialization vector IV_x.

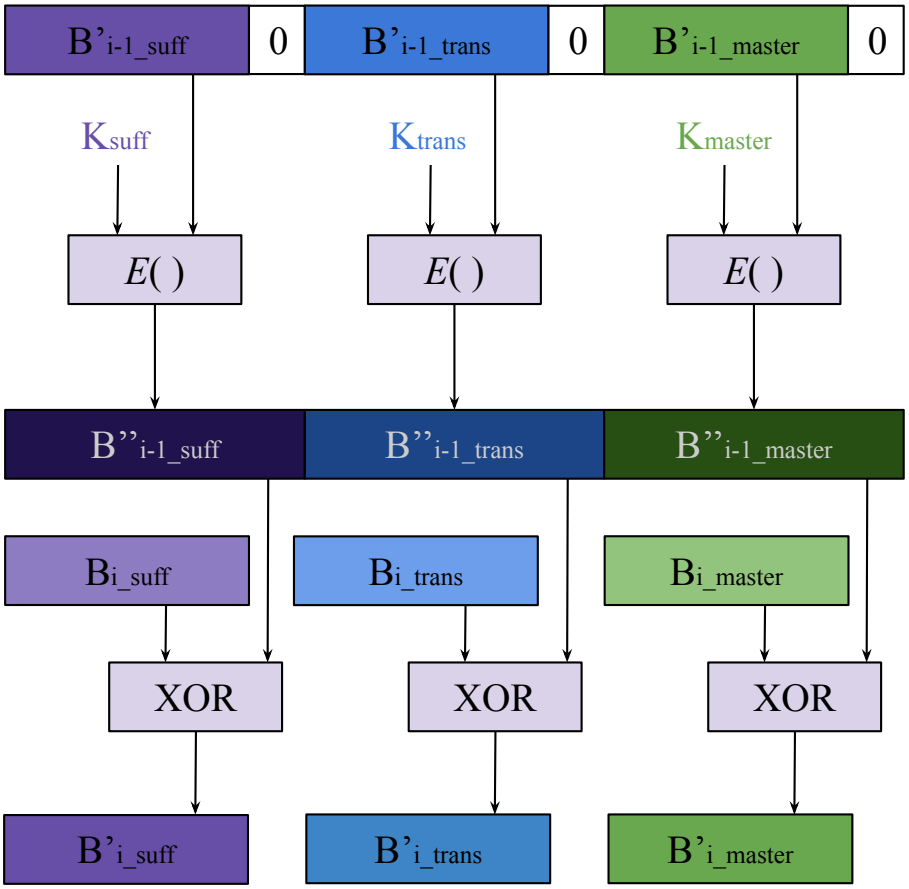


Figure 5.23: *Encryption of a block B_i which is composed of 3 parallel encryptions.*

An exclusive or (xor) is then performed between the AES encrypted sub block B''_{i-1_x} and the clear sub block B_{i_x} to produce the encrypted sub block B'_{i_x}:

$$B'_{i\_x} = B''_{i-1\_x} \bigoplus B_{i\_x}. \tag{5.26}$$

We note that if each block B_{i_x} is originally composed of α_B bits, then we conserve only the first α_B bits of the xor, resulting in an encrypted block B'_{i_x} of α_B bits. Consequently, there is no size expansion.

5.5.4 Hierarchical Decryption

Fig. 5.24 illustrates an overview of the hierarchical decryption phase. In order to decrypt the fully encrypted 3D object O_e which has a confidential level, one of the hierarchical keys in the ring is used. If the master key K_{master} is used, then the original 3D object O is retrieved. If the transparent key K_{trans} is used, then the resulting selectively encrypted 3D object O_{trans} will have a transparent level. Otherwise, if the sufficient key K_{suff} is used, the resulting selectively encrypted 3D object O_{suff} will have a sufficient level.

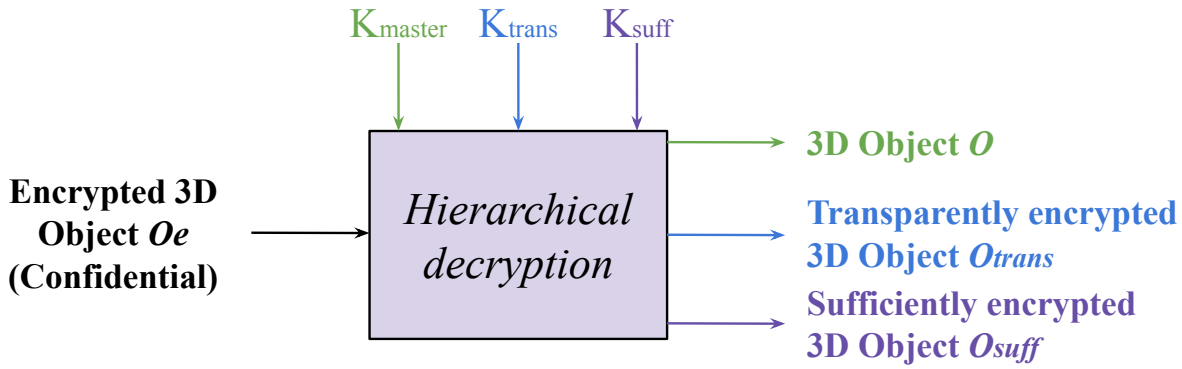


Figure 5.24: *Overview of the decryption phase as a function of the used key.*

As indicated in Section 5.5.2, to decrypt a confidential 3D object O_e, the user has a single secret key K_x which is used to decrypt the corresponding sub blocks B'_{i_x}. Each key K_x decrypts their corresponding sub block B_{i_x} and then subsequently generates the inferior key K_{x-1} using Eq. 5.18 or Eq. 5.20, in order to decrypt the inferior blocks.

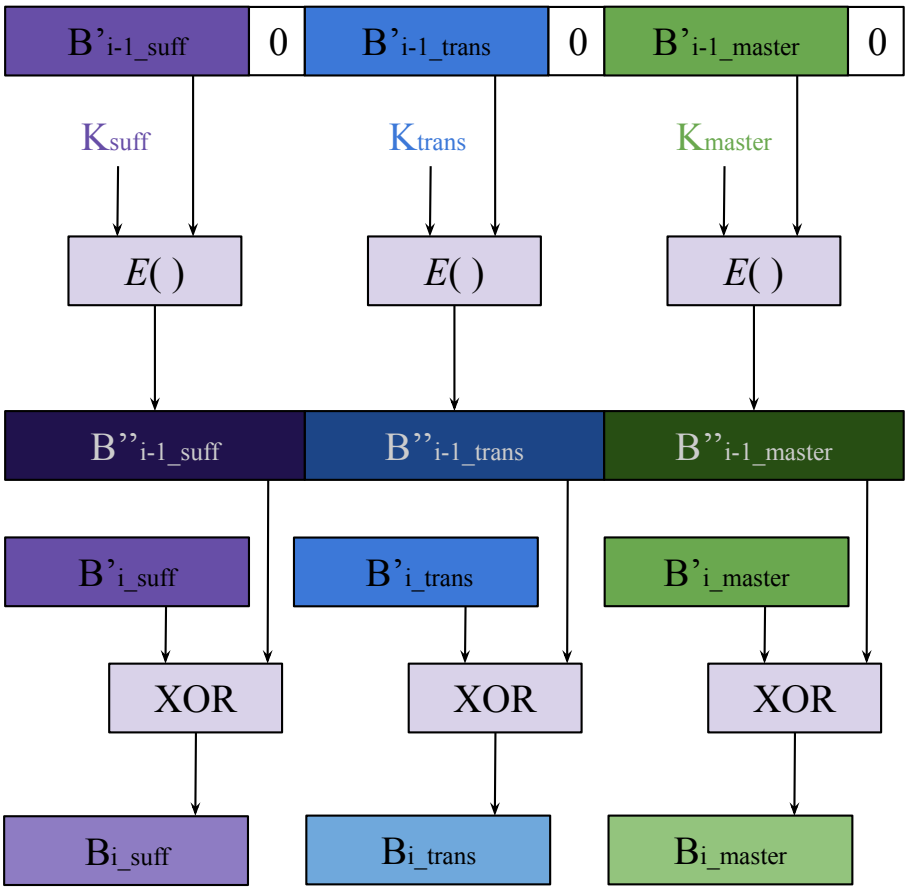


Figure 5.25: *Hierarchical decryption of a block B'_i*

Fig. 5.25 illustrates the decryption process for a block B'_i. The decrypted sub block B_{i_x} can be retrieved using an xor between its corresponding encrypted block B'_{i_x} and the doubly encrypted sub block B''_{i-1_x}:

$$B_{i\_x} = B''_{i-1\_x} \bigoplus B'_{i\_x}, \tag{5.27}$$

where $x \in \{master, trans, suff\}$.

For example, if a user has the master key K_{master}, the encrypted 3D object can be fully decrypted. The master key K_{master} is first used to decrypt the sub block B'_{0_master}. From this, the transparent key K_{trans} can be generated and the sub block B'_{0_trans} can then be decrypted. Then, the sufficient key K_{suff} can be generated. The three sub blocks B'_{i_x} corresponding to each visual security level can then be decrypted in parallel.

We note that like during the encryption process, the block B''_{i-1_x} is generated using Eq. 5.25.

5.6 Hierarchical Decryption Experimental Results

In this section, we present experimental results obtained with the proposed method. Our experiments were performed on 10 3D objects from the Stanford dataset (34), as well as the 380 3D objects which make up the Princeton dataset (143). We note that all the keys as well as the initialization vectors were drawn at random. In Section 5.6.1, we show experimental results obtained by the proposed format compliant encryption method which allows for a hierarchical decryption of 3D objects, with a full example. Then, in Section 5.6.2, we detail the results obtained when the proposed method is applied to different datasets. We describe the results of individual 3D objects of the Stanford dataset (34), as well as the average results on the larger Princeton dataset (143). Finally, in Section 5.6.3 we present the effects of different attacks on our method, such as Laplacian smoothing and zeroing. We note the original 3D object O, the encrypted confidential level 3D object O_e, the sufficient level selectively encrypted 3D object O_{suff}, and the transparent level selectively encrypted 3D object O_{trans}.

5.6.1 Full Example

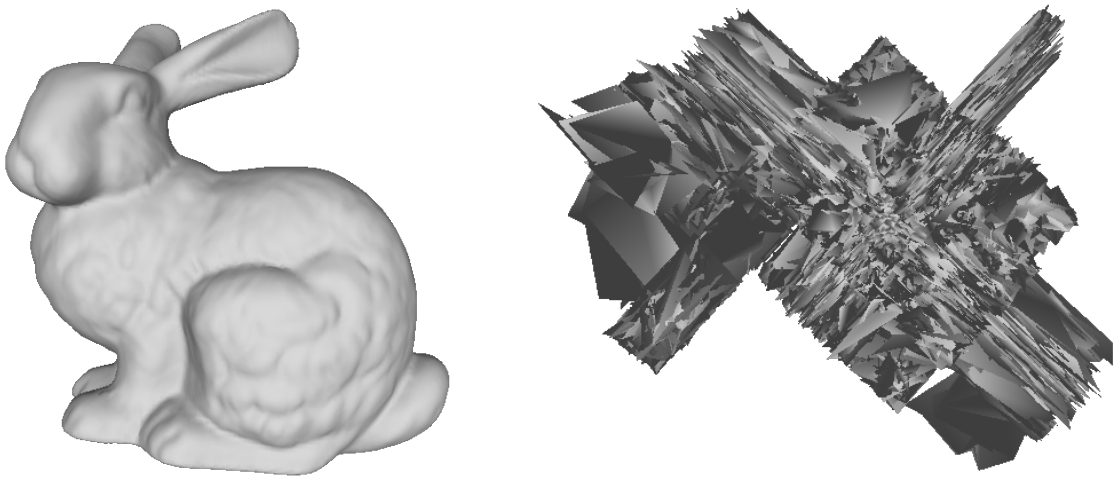
Fig. 5.26 illustrates the original 3D object *Bunny* O (Fig. 5.26a), taken from the Stanford dataset (34), as well as its encryption O_e using the proposed method (Fig. 5.26b). In this example, we used the parameter $\alpha = 2$, where α is the number of bits encrypted per sub block corresponding to a certain visual security level in a single mantissa. We also use the 256 bit master key:

- $K_{master} = 357538782f413f44\ 28472b4b62506553\ 68566d5970337336\ 7639792442264529$,

and the 128 bit initialization vector:

- $IV_{master} = 472d4a614e645267\ 556b587032733576$.

The root mean squared error (RMSE) between the original 3D object O and the



(a) *Original 3D object O.* **(b)** *Confidential 3D object O_e.*

Figure 5.26: *a) The original 3D object O Bunny, b) Its encryption O_e with the proposed method.*

encrypted confidential level 3D object O_e is 0.162 and is visually secure, as illustrated in Fig. 5.26b.

From the secret key K_{master} and the initialization vector IV_{master}, based on the hash function SHA-3 (152), we generate the transparent key:

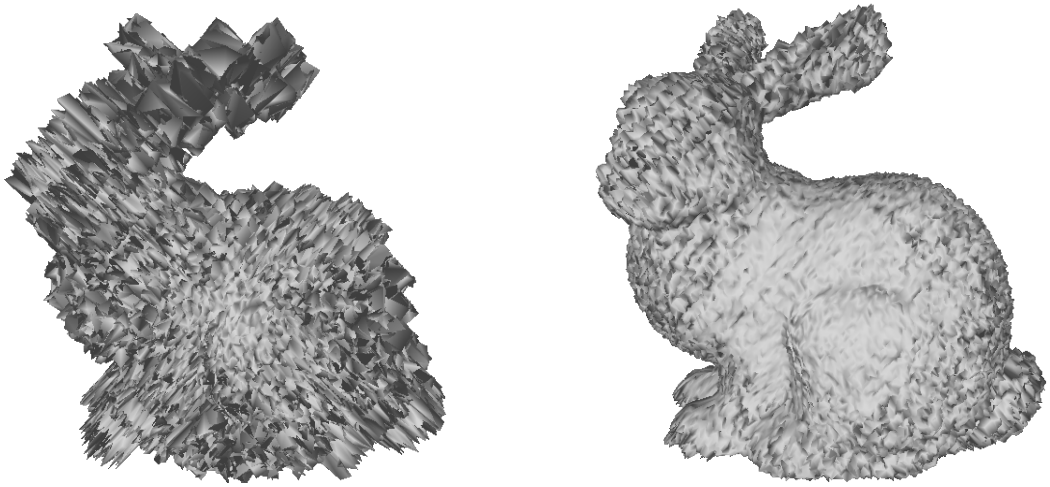
- $K_{trans} = c47caae3bb1aeb13\ c8b6e21986a0f4ee\ 085559067bf40108\ 3a733350e256e786$,
- $IV_{trans} = 7bb5000ab802c01f\ c9b71f5b57262a23$,

and then subsequently the sufficient key:

- $K_{suff} = 0ed37fc1cc7f4ab3\ 9313a3243947da08\ 08a0266447a985f4\ 70928568a0a6767b$,
- $IV_{suff} = 37a956434e90d5e0\ 8f62a8801e26245d$.

During the decryption phase, the generated hierarchical keys K_{suff} or K_{trans}, or the secret key K_{master} are used. Fig. 5.27 illustrates the results of the sufficient level 3D object O_{suff} (Fig. 5.27a) and the transparent level 3D object O_{trans} (Fig. 5.27b). The RMSE between the original 3D object O and the hierarchically decrypted sufficient 3D object O_{suff} is 39.769×10^{-3} and the RMSE between O_{trans} and O is 9.507×10^{-3}. We note that if the secret key K_{master} is used, then the original 3D object is perfectly reconstructed, and therefore has an RMSE of 0.

We note that the discussion around the choice of parameter for α is complex, as it depends on the HVS. A less detailed 3D object which is selectively encrypted with a certain parameter will generally be more recognizable than a more detailed 3D object selectively encrypted with the same parameters. We can therefore use our proposed 3DVS score in order to determine the optimal value of the parameter α for each visual security level.



(a) *Reconstructed sufficient 3D object* O_{suff}. **(b)** *Reconstructed transparent 3D object* O_{trans}.

Figure 5.27: *Hierarchical decryption of the confidentially encrypted 3D object Bunny: a) Sufficient level 3D object* O_{suff}*, b) Transparent level 3D object* O_{trans}.

Fig. 5.28 represents the visual results of our proposed method for the 3D object *Bunny*. We can observe that the parameters $\alpha = 4$ and $\alpha = 5$ are too high, as we can access too much information for the sufficient level and the transparent level. The most visually accurate results are obtained with the parameters $\alpha = 2$ or $\alpha = 3$ for *Bunny*. These parameters are confirmed by using the 3DVS score. We note that the optimal 3DVS score is 3 for the transparent level, and 2 for the sufficient level. For the sufficient level, the 3DVS score has a value of 1 for $\alpha = 2$ and 1.65 for $\alpha = 3$. For the transparent level, the 3DVS score has a value of 2.47 for $\alpha = 2$ and 3.76 for $\alpha = 3$. These values remain close to the optimal 3DVS score for each level. In particular, the best parameter for a sufficient level 3D object *Bunny* is $\alpha = 3$, and $\alpha = 2$ for the transparent level. Therefore, the user should choose $\alpha = 2$ or $\alpha = 3$ depending on their desired use. We note that the content of the confidential level 3D object O_e is always secure regardless of the value of α, as this corresponds to an encryption, and that the original 3D object is always reconstructed regardless of the value of α. Fig. 5.29 illustrates the RMSE values according to the desired visual security level for $\alpha \in [1, 5]$, for the 3D object *Bunny*.

5.6.2 Results on Large Datasets

In this section, we present the results of the proposed method on large datasets: the Stanford dataset (34) and the Princeton dataset (143). Fig. 5.30 presents the hierarchical decryption of three 3D objects *Dragon, Ramses* and *Venus* from the Stanford dataset. Table 5.9, Table 5.11 and Table 5.13 present the RMSE between the original 3D objects O of the Stanford dataset (34) and the confidentially encrypted 3D objects O_e, the sufficient level hierarchical decryption O_{suff}, and the transparent level hierarchical decryption O_{trans} respectively according to α. Table 5.10, Table 5.12 and Table 5.14 present the Hausdorff distances. Table 5.15 and Table 5.16 present the mean RMSE and Hausdorff distances with the standard deviation between the original 3D objects O of the Princeton dataset (143) and the hierarchical decryptions and encryption according to α. We note that a clear level 3D object decrypted with the secret master key K_{master} is iden-

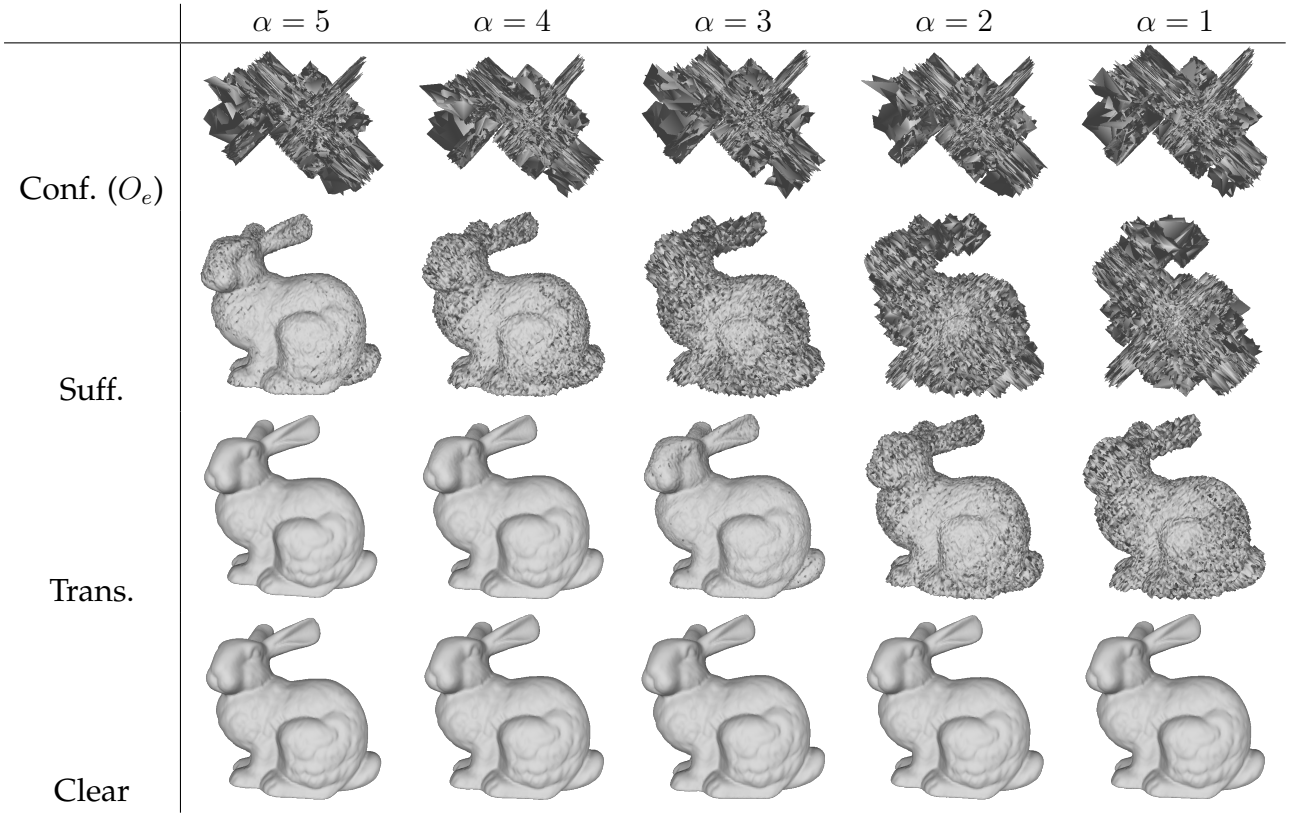


Figure 5.28: *The resulting hierarchical decryption for Bunny according to the desired visual security level with different α parameters.*

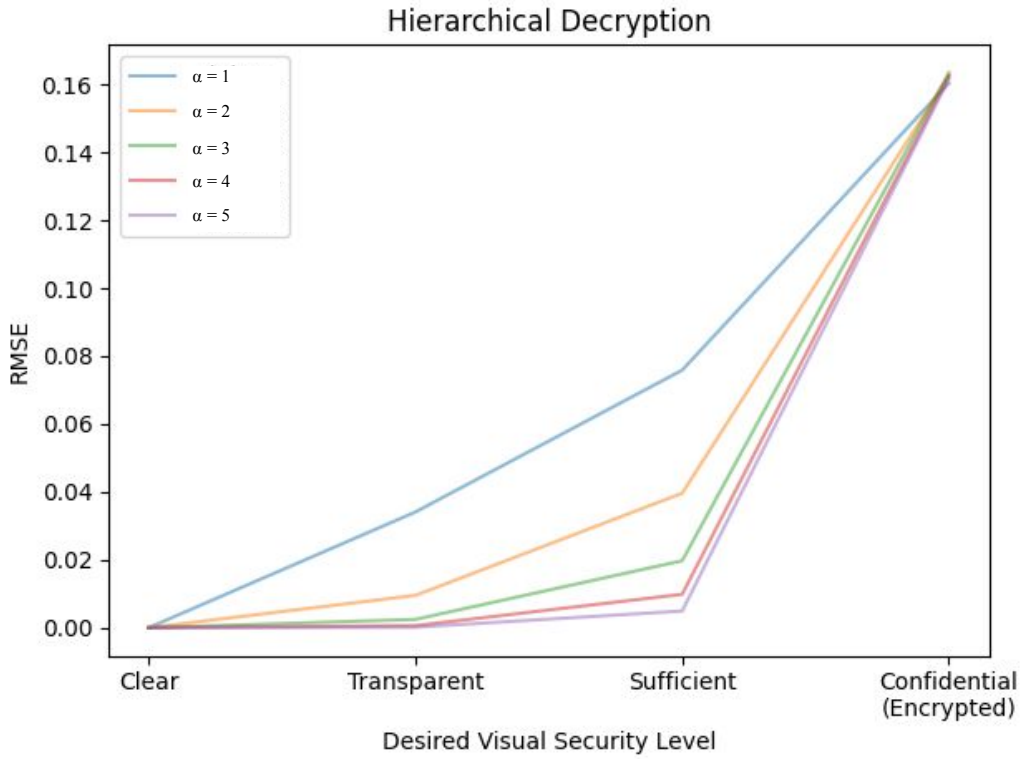


Figure 5.29: *RMSE of the original 3D object Bunny and the 3D objects with different visual security levels according to α.*

tical to the original 3D object O, and so it has an RMSE and a Hausdorff distance of 0.

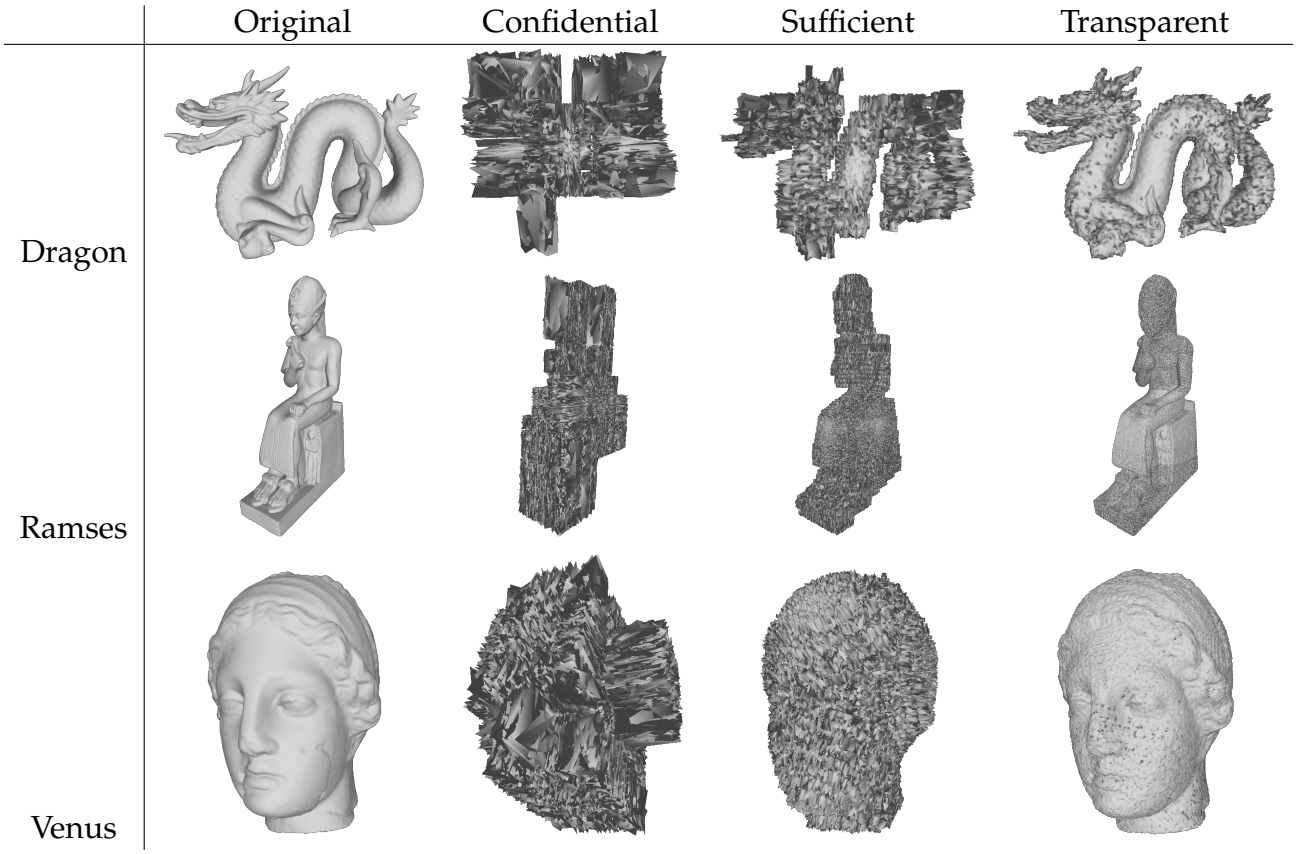


Figure 5.30: *The resulting hierarchical decryption of three different 3D objects from the Stanford dataset.*

3D Object	$\alpha = 1$	$\alpha = 2$	$\alpha = 3$	$\alpha = 4$	$\alpha = 5$
Bunny	0.160	0.162	0.163	0.163	0.162
Casting	0.180	0.182	0.182	0.184	0.182
Cow	0.187	0.189	0.187	0.188	0.189
Crank	0.184	0.184	0.184	0.185	0.184
Dragon	0.177	0.179	0.180	0.180	0.180
Hand	0.162	0.164	0.163	0.163	0.163
Horse	0.156	0.158	0.158	0.157	0.158
Rabbit	0.161	0.162	0.162	0.162	0.161
Ramses	0.147	0.148	0.149	0.148	0.149
Venus	0.224	0.225	0.225	0.225	0.225

Table 5.9: *RMSE between the original 3D objects O and the **encrypted (confidential level)** 3D object according to α.*

As we decrease α, the RMSE and the Hausdorff distances increase for the sufficient and transparent levels. This is to be expected, because more significant bits remain encrypted, which has a larger impact on the visual security of the 3D object. The standard deviations for the sufficient and transparent levels of the Princeton dataset (143) remain low for the RMSE (Table 5.15) and the Hausdorff distances (Table 5.16). For the

3D Object	$\alpha = 1$	$\alpha = 2$	$\alpha = 3$	$\alpha = 4$	$\alpha = 5$
Bunny	0.160	0.162	0.163	0.163	0.162
Casting	0.180	0.182	182.309	0.184	0.182
Cow	0.187	0.189	0.187	0.188	0.189
Crank	0.183	0.184	0.184	0.185	0.184
Dragon	0.176	0.179	0.180	0.180	0.180
Hand	0.161	0.164	0.163	0.163	0.163
Horse	0.157	0.158	0.158	0.157	0.158
Rabbit	0.161	0.162	0.162	0.162	0.161
Ramses	0.147	0.148	0.149	0.148	0.149
Venus	0.224	0.225	0.225	0.225	0.225

Table 5.10: *Hausdorff distances between the original 3D objects O and the **encrypted (confidential level)** 3D object according to α.*

3D Object	$\alpha = 1$	$\alpha = 2$	$\alpha = 3$	$\alpha = 4$	$\alpha = 5$
Bunny	75.864	39.610	19.725	9.839	4.920
Casting	86.839	45.381	23.121	11.389	5.720
Cow	91.681	45.202	23.715	11.800	5.837
Crank	90.105	45.574	24.080	11.614	5.748
Dragon	85.751	43.663	21.773	10.799	5.403
Hand	80.142	40.931	20.526	10.242	5.119
Horse	78.823	39.011	19.686	9.837	4.918
Rabbit	75.687	37.494	18.686	9.396	4.736
Ramses	71.570	148.457	18.449	9.283	4.631
Venus	107.965	55.212	27.643	13.737	6.891

Table 5.11: *RMSE (10^{-3}) between the original 3D objects O and the **sufficient level** hierarchical decryption according to α.*

3D Object	$\alpha = 1$	$\alpha = 2$	$\alpha = 3$	$\alpha = 4$	$\alpha = 5$
Bunny	75.864	39.610	19.725	9.839	4.920
Casting	86.839	45.3812	23.121	11.3891	5.720
Cow	91.681	45.202	23.715	11.800	5.837
Crank	90.105	45.574	24.080	11.614	5.748
Dragon	85.751	43.663	21.773	10.799	5.403
Hand	80.142	40.931	20.526	10.242	5.119
Horse	78.823	39.011	19.686	9.837	4.918
Rabbit	75.687	37.494	18.686	9.396	4.736
Ramses	71.570	148.457	18.449	9.283	4.631
Venus	107.965	55.212	27.643	13.737	6.891

Table 5.12: *Hausdorff distances (10^{-3}) between the original 3D objects O and the **sufficient level** hierarchical decryption according to α.*

encrypted 3D objects, the level of α does not greatly impact the RMSE or the Hausdorff distances. We can also see this by looking at the mean RMSE and standard de-

3D Object	$\alpha = 1$	$\alpha = 2$	$\alpha = 3$	$\alpha = 4$	$\alpha = 5$
Bunny	34.088	9.527	2.454	0.615	0.154
Casting	39.703	11.063	2.833	0.715	0.178
Cow	40.258	11.361	2.913	0.729	0.181
Crank	40.077	11.231	2.859	0.721	0.180
Dragon	37.370	10.474	2.690	0.675	0.169
Hand	35.640	9.968	2.557	0.648	0.160
Horse	34.137	9.496	2.437	0.615	0.154
Rabbit	32.731	9.105	2.345	0.591	0.148
Ramses	32.125	8.979	2.295	0.578	0.145
Venus	47.677	13.357	3.421	0.861	0.216

Table 5.13: *RMSE (10^{-3}) between the original 3D objects and the **transparent level** hierarchical decryption according to α.*

3D Object	$\alpha = 1$	$\alpha = 2$	$\alpha = 3$	$\alpha = 4$	$\alpha = 5$
Bunny	34.088	9.512	2.454	0.615	0.154
Casting	39.703	11.063	2.833	0.715	0.178
Cow	40.258	11.361	2.913	0.729	0.181
Crank	40.077	11.231	2.859	0.721	0.180
Dragon	37.370	10.474	2.690	0.675	0.169
Hand	35.640	9.968	2.557	0.648	0.160
Horse	34.137	9.496	2.437	0.615	0.154
Rabbit	32.731	9.105	2.345	0.591	0.148
Ramses	32.125	8.979	2.295	0.578	0.145
Venus	47.677	13.357	3.421	0.861	0.216

Table 5.14: *Hausdorff distances (10^{-3}) between the original 3D objects and the **transparent level** hierarchical decryption according to α.*

	Confidential	Sufficient	Transparent
$\alpha = 1$	169.380 ± 0.076	81.556 ± 0.034	36.300 ± 0.014
$\alpha = 2$	170.747 ± 0.077	41.817 ± 0.016	10.137 ± 0.004
$\alpha = 3$	170.888 ± 0.077	20.873 ± 0.007	2.601 ± 0.001
$\alpha = 4$	170.787 ± 0.076	10.467 ± 0.004	0.653 ± 0.0003
$\alpha = 5$	170.781 ± 0.077	5.236 ± 0.002	0.163 ± 0.00006

Table 5.15: ***Average** RMSE (10^{-3}) between the original 3D objects O of the Princeton dataset and the hierarchical decryption according to α.*

	Confidential	Sufficient	Transparent
$\alpha = 1$	462.474 ± 0.283	215.414 ± 0.103	79.316 ± 0.032
$\alpha = 2$	472.955 ± 0.297	117.892 ± 0.052	27.022 ± 0.009
$\alpha = 3$	475.827 ± 0.300	59.655 ± 0.021	7.33667 ± 0.002
$\alpha = 4$	474.405 ± 0.297	29.716 ± 0.010	1.910 ± 0.0006
$\alpha = 5$	473.821 ± 0.297	14.997 ± 0.005	0.480 ± 0.0001

Table 5.16: ***Average** Hausdorff distances (10^{-3}) between the original 3D objects O of the Princeton dataset and the hierarchical decryption according to α.*

viations (Table 5.15) for the Princeton dataset (143) which remain around 170×10^{-3} and 77×10^{-6} respectively. The same is true for the Hausdorff distances, which remain around 473×10^{-3} and 30×10^{-6}.

Table 5.9, Table 5.11 and Table 5.13 are summarized by Fig. 5.31, which illustrates the mean RMSE values and the standard deviation of the 10 objects from the Stanford dataset (34) according to the desired visual security level for each value of $\alpha \in [1, 5]$.

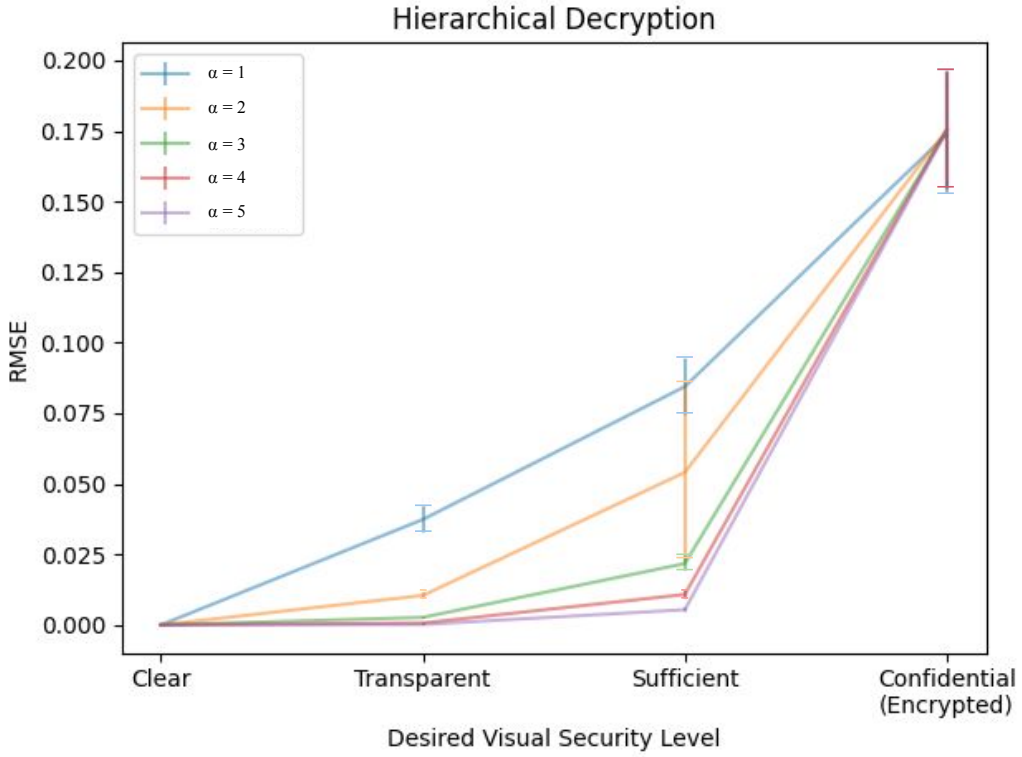


Figure 5.31: *Mean RMSE of the original 3D objects of the Stanford dataset and the 3D objects with different visual security levels according to α.*

5.6.3 Sensitivity Analysis

As shown by Said (153), methods where only a part of the data is encrypted can be susceptible to attacks aiming to retrieve the content. These attacks consist of using the clear data in order to reconstruct the content. To analyze the security of the hierarchically decrypted 3D objects, we first apply a Laplacian smoothing (154) to the 3D objects in order to attempt to recover the original 3D object. Then we use a zeroing attack, where all the encrypted bits are set to zero.

Fig. 5.32 and Fig. 5.33 present the visual results of the attacks for $\alpha = 2$ and $\alpha = 3$ respectively, where α is the number of encrypted bits per visual security level per single mantissa. When applied to the encrypted confidential level 3D object O_e, the smoothing and zeroing has no effect on the visual security. The 3D object's content remains secure. Although the RMSE slightly improves (Table 5.17 and Table 5.18), it remains very high, and no content or shape is revealed.

The smoothing attack improves the RMSE because by the nature of this kind of attack, it eliminates strong variations in the 3D object's polygons generated by the encryption. However, this does not in fact bring the 3D object closer to the original in practical use as it does not change the visual security level of the 3D object.

We observe that for the sufficient level, a slightly higher quality 3D object can be reconstructed, however the high quality details remain protected. Likewise, an attack on the transparent level 3D object also results in a slightly higher quality 3D object, but the high quality content remains secure. In fact, when the 3D object *Bunny* is zeroed, the RMSE increases for the sufficient and transparent levels, as seen in Table 5.17 for $\alpha = 2$ (49.2×10^{-3} and 11.033×10^{-3} respectively) and Table 5.18 for $\alpha = 3$ (26.658×10^{-3} and 3.145×10^{-3} respectively). We note that these values are confirmed by the 3DVS score. The confidential level 3D objects (Fig. 5.32 and Fig. 5.33) always have a 3DVS score of 1. The sufficient level 3D objects in both Fig. 5.32 and Fig. 5.33 always have a 3DVS score less than or equal to 2.10. For $\alpha = 2$ (Fig. 5.32), the transparent level 3D objects have a very stable 3DVS score varying from 2.47 to 2.51. For $\alpha = 3$ (Fig. 5.33), the 3DVS score varies for the transparent level from 3.76 to 4.58. This confirms that the optimal parameter for the transparent level for *Bunny* is $\alpha = 2$. We note that in order to perform an attack on the sufficient or transparent level 3D objects, the person would already have to possess a hierarchical key.

	Encrypted	Smoothed	Zeroed
Confidential	162.499	143.416	154.080
Sufficient	39.610	32.364	**49.200**
Transparent	9.512	6.425	**11.033**

Table 5.17: *RMSE (10^{-3}) between the original 3D object Bunny and the attacked 3D object for $\alpha = 2$.*

We note that we are the first to propose an encryption scheme which allows for a hierarchical decryption. Even Gschwandtner and Uhl (45), who partially encrypt 3D objects in layers using a progressive mesh representation, cannot hierarchically decrypt the encrypted 3D objects. Once the 3D objects are encrypted using (45), they can only be decrypted to reconstruct the original 3D object.

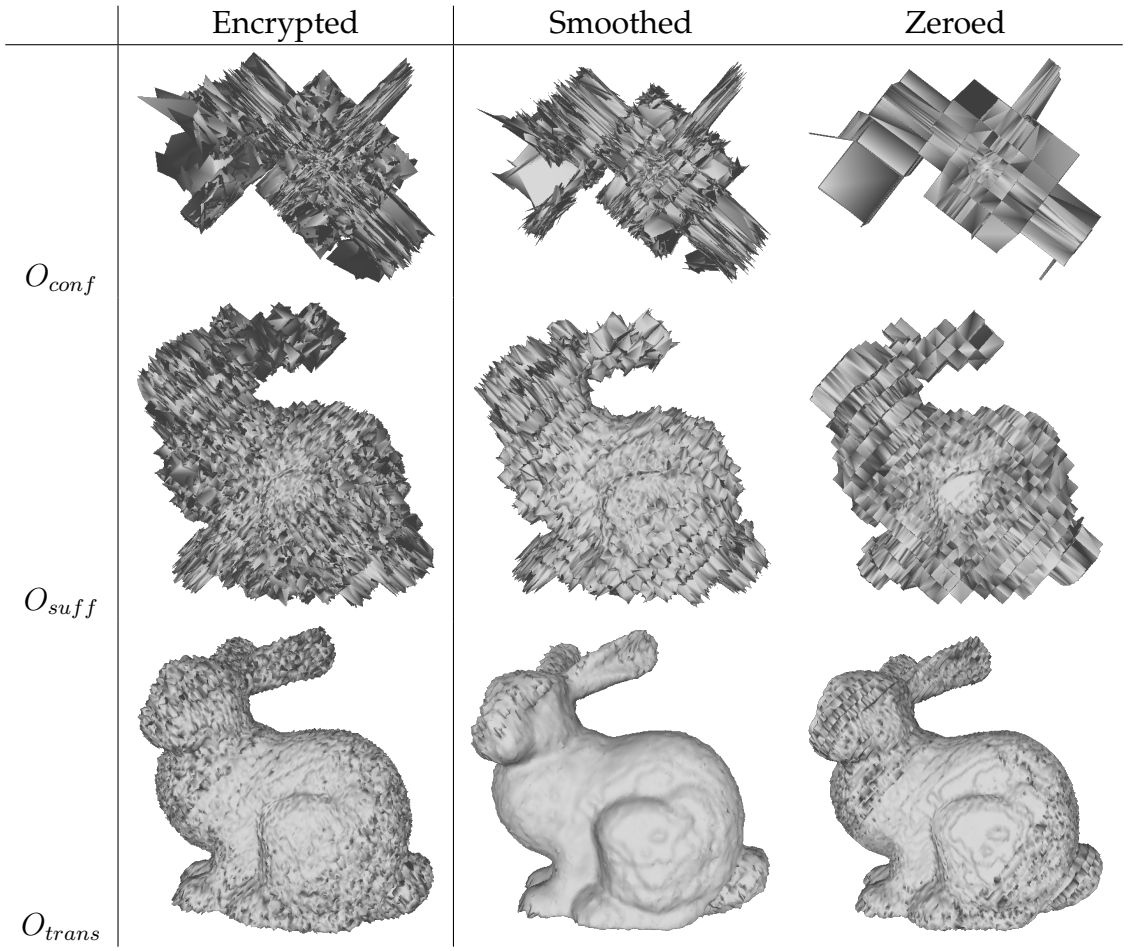


Figure 5.32: *Visual results of the attacks on the 3D object Bunny where* $\alpha = 2$.

	Encrypted	Smoothed	Zeroed
Confidential	163.478	144.449	156.159
Sufficient	19.725	14.677	**26.658**
Transparent	2.454	2.164	**3.145**

Table 5.18: *RMSE* (10^{-3}) *between the original 3D object Bunny and the attacked 3D object for* $\alpha = 3$.

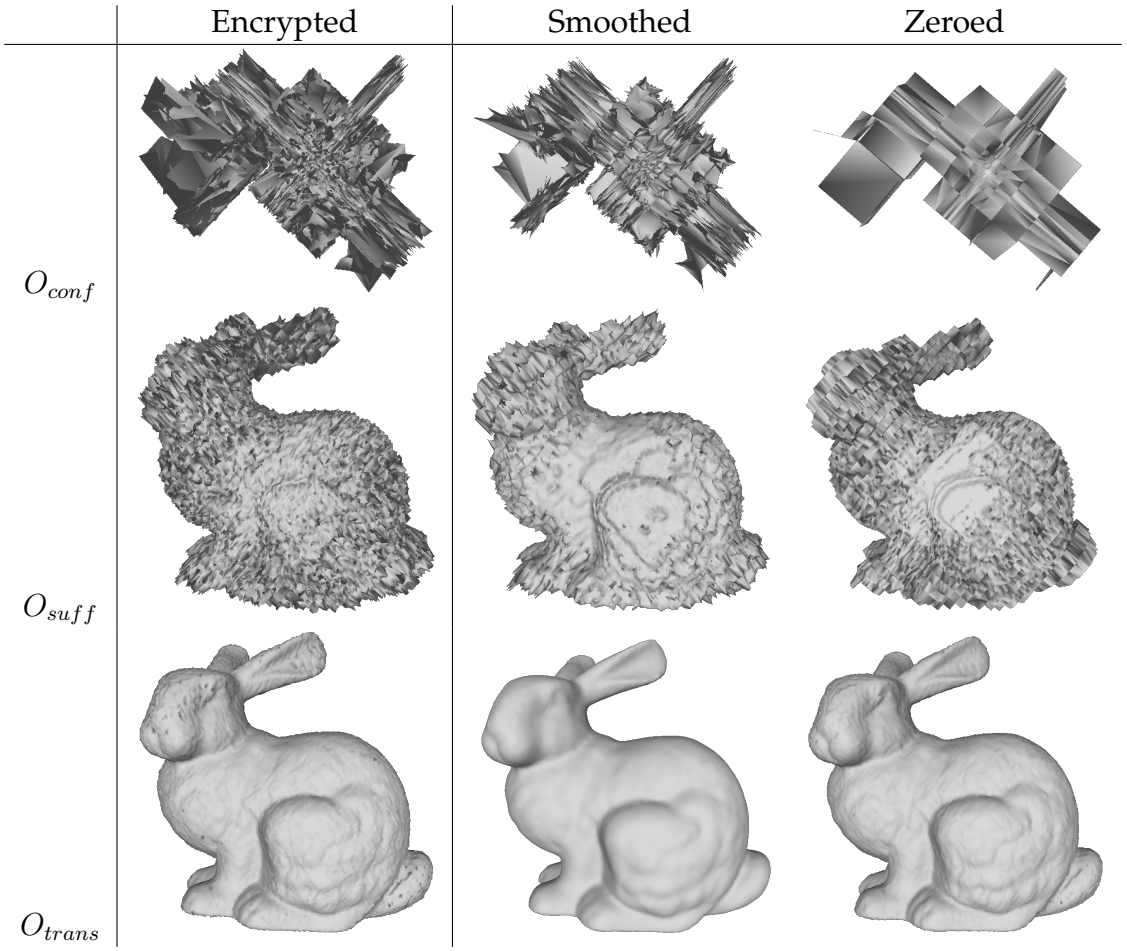


Figure 5.33: *Visual results of the attacks on the 3D object Bunny where* $\alpha = 3$.

5.7 Conclusion

In this chapter, we first presented a dataset of selectively encrypted 3D objects and their opinion scores (OS) upon which our first contribution is based. Then, with this dataset and full reference metrics, we used a polynomial regression model to estimate the encryption parameter for selectively encrypted 3D objects. Then we developed a new metric, 3DVS score, which serves to evaluate the visual security level of selectively encrypted 3D objects.

Methods based on selective encryption approaches generate 3D objects that can have a transparent, sufficient or a confidential visual security level. However, it is difficult to establish the pivotal thresholds for change between these three levels when using objective methods. Furthermore, depending on the encryption parameters, the geometry, and the connectivity of the 3D object, the results can vary significantly. We used the SE3DO dataset in order to create a linear regression model designed to estimate encryption parameters according to the desired visual security level for selectively encrypted 3D objects based on the SE3DO dataset. Finally, with the SE3DO dataset, we developed a new visual security metric 3DVS score for selectively encrypted 3D objects. The 3DVS score was constructed with sigmoid-based regression models and remains effective when used with another 3D selective encryption method.

In future work, we wish to further exploit subjective data assessment by using learning-based approaches such as neural networks, convolutional neural networks (2D and 3D) and by taking advantage of the inherent characteristics of 3D objects. This type of approach should allow us to propose solutions in the case of encrypted 3D objects desynchronized with the original 3D object. In future work, we propose also to develop new visual security metrics for encrypted 3D objects based on metrics with reduced-reference or no-reference methods.

The SE3DO dataset, the encryption parameter estimation method, as well as the 3DVS score are presented in the international journal Signal Processing: Image Communication (155).

We then presented a format compliant 3D object encryption method that allows for hierarchical decryption. The proposed method is eco-friendly, as only a single 3D object is encrypted, stored and subsequently transferred. We established a ring of hierarchical keys where each key allows us to decrypt an encrypted confidential level 3D object to a selectively encrypted 3D object with a different level of visual security, or to fully decrypt the 3D object to a clear level. We considered two different levels of visual security: the **sufficient** level and the **transparent** level. In our experimental results, we have seen that the choice of the encryption parameter α is a complex discussion, as it depends on both the desired visual security level and the 3D object's characteristics. We have shown through a sensitivity analysis that the proposed method is also resistant against attacks such as smoothing or zeroing. To the best of our knowledge, this is the first encryption method for 3D objects capable of a hierarchical decryption.

A limitation of our method is that once a key has been used, a hierarchically superior key cannot be used on the same decrypted 3D object. For example, if the sufficient key is used, the confidentially encrypted 3D object will be hierarchically decrypted to

a sufficient visual security level. A hierarchically superior key, for example, the master key, cannot be used to decrypt the hierarchically decrypted sufficient level 3D object in order to retrieve the original 3D object. In future work, we aim to develop a method where a second hierarchical key can be applied to a selectively decrypted 3D object.

This work is presented in two international publications. We first presented this method in the international conference IEEE MMSP 2022, where it won a Top 10% Best Paper Award (156). We then presented an improved, detailed paper in the international journal IEEE Transactions on Multimedia (157).

CHAPTER 6

JOINT SECURITY AND COMPRESSION BASED ON DRACO

Chapter Contents

6.1 Introduction

Nowadays, 3D objects often represent important assets which need to be of a high quality. Consequently, these 3D objects are often composed of millions of vertices. As presented in Chapter 3, it therefore expensive, time consuming and not environmentally friendly to store these 3D objects on the cloud or transfer them over networks. In addition, with online storage and sharing, these 3D objects are vulnerable to malicious attacks such as theft or copying. It is therefore important that they are both compressed and secured jointly.

However, 3D object encryption first and compression afterwards is not optimized, as while compression methods rely on using the redundancies in the 3D object, encryption eliminates redundancies. Therefore compression is not effective if performed after an encryption. In addition, if the compression is lossy, then the 3D object cannot be correctly decrypted. However, performing compression first and encryption afterwards is not format compliant.

A solution to this problem is crypto-compression methods, where encryption and compression are performed jointly. These methods are beneficial as they combine the encryption and compression into a single process. This adds an extra layer of security as an additional level of complexity and uncertainty is added, making it more difficult for an attacker to break the system. As presented in Chapter 3, this is also true for selective crypto-compression methods, as well as joint data hiding and compression methods, as both security methods and compression methods generally modify the same domains.

In this chapter, we present three contributions based on joint security and 3D object compression. We note that all three contributions are based on the 3D object compression method Draco (108), proposed by Google in 2014, which is presented in detail in Chapter 3.

The first contribution is a crypto-compression method, where an encryption step is embedded in the entropy encoding step during the Draco geometry encoding process. More precisely, we propose encrypting the encoded vertex prediction errors during the range Asymmetric Numeral System (rANS) (110) encoding step of the Draco 3D object compression method. This encryption is performed with the AES encryption scheme where the cipher feedback (CFB) mode is used. The proposed method is format compliant and has no size expansion in the encrypted or plaintext domain. It has a low complexity and does not require additional information such as an auxiliary file.

In the second contribution, we propose a format compliant selective crypto-compression method for 3D objects, based on Draco. We propose integrating the selective encryption step in the geometry encoding phase of Draco, between the vertex quantification and the vertex prediction. This selective encryption step is based on an exclusive-or (XOR) encryption.

In the final contribution of this chapter, we propose a joint watermarking and compression method for 3D objects based on Draco. As for the selective encryption step in the second contribution, we propose integrating the watermarking step between the

vertex quantization step and the vertex prediction step of the encoding phase of the Draco 3D object compression method. As a result, the Draco format file (.drc) is watermarked. This watermark is then extracted during the Draco decoding phase, before the vertices are reconstructed. As the watermark is not removed during the extraction phase, it is also possible to retrieve the watermark after the 3D object is reconstructed.

To the best of our knowledge, we are the first to propose joint security and compression methods for Draco. We note that a crypto-compression method based on Draco was requested on Github and was subsequently added by Google to their list of calls for Draco enhancements (158).

This chapter is organized as follows. First, in Section 6.2, we present the overall concept of integrating security methods in the Draco 3D object compression method. Then, in Section 6.3, we present our first contribution, which is a crypto-compression method based on Draco. In Section 6.4, we detail our second contribution, which is a selective crypto-compression method based on Draco. Finally, in Section 6.5, we present our third contribution, which is a joint watermarking and compression method based on Draco.

6.2 Security Integration in Draco

We note that the Draco 3D compression method is described in detail in Chapter 3. Draco is composed of two main processes, which are performed in parallel. These processes are the connectivity encoding (resp. decoding) process and the geometry encoding (resp. decoding) process. The connectivity encoding process is largely based on the connectivity encoding method Edgebreaker (109), which traverses the connectivity of a 3D object step-by-step by means of a depth-first spiraling triangle spanning tree. At each step, a new triangle is visited and encoded with single character C, L, E, R or S, according to which of its neighboring triangles have already been visited. These characters are known as the CLERS string.

As the proposed joint security and compression methods for 3D objects must be format compliant, we decide not to encrypt the CLERS string data. As the CLERS string serves to describe how the current triangle can be reattached to set of already reconstructed triangles during the reconstruction, modifying this data leads to impossible connectivity configurations. Consequently, the secured and compressed 3D object becomes illegible for the Edgebreaker decoder and therefore it loses its format compliance. Thus, in our contributions, the 3D object's connectivity is not modified. All security steps are embedded in during the geometry encoding (resp. decoding) process.

6.3 Crypto-compression

In this section, we detail our first contribution, the 3D object crypto-compression method based on Draco. First, in Section 6.3.1, we detail the proposed Draco 3D object crypto-compression method. We describe the geometry encoding phase in which the encryption step is integrated, as well as the Draco decoding phase in which the decryption step is integrated. Then, in Section 6.3.2, we present experimental results, as well as a detailed security analysis of the proposed Draco 3D object crypto-compression method.

6.3.1 The Proposed 3D Object Crypto-compression Method

In this work, we propose integrating an encryption step during the rANS encoding of the prediction errors of the quantized vertices during the Draco geometry encoding process. Fig. 6.1 illustrates an overview of the encoding phase of the proposed method. During the geometry encoding step, the vertices are first quantized according to the parameter qp. These quantized vertices then undergo a vertex prediction step according to a prediction scheme. This prediction scheme is determined according to the Draco compression level parameter cl, as detailed in Chapter 3.

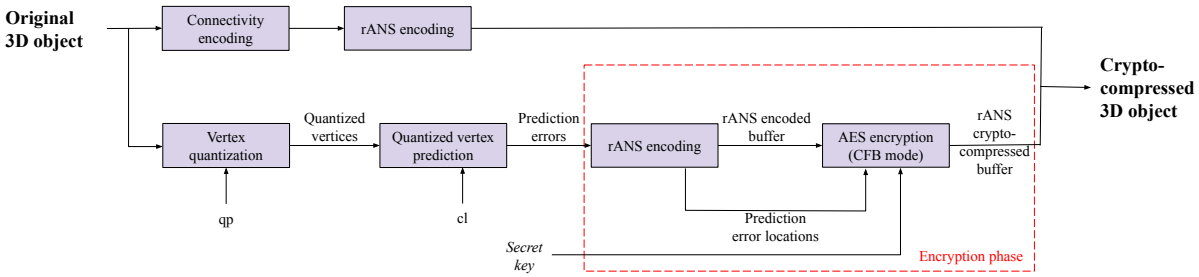


Figure 6.1: *Overview of the encoding phase of the proposed Draco 3D object crypto-compression method.*

The encryption step is then integrated in the rANS encoding of the prediction errors of the quantized vertices. The prediction errors, along with other information needed to decode the vertices, are encoded with the rANS encoding scheme, and then stored in the Draco output buffer. The rANS encoded prediction errors are then substituted by their encryption. We note that since the encryption takes place after the rANS entropy encoding, there is no size expansion and therefore Draco's original compression rate is maintained. The encryption would be less effective if performed before the entropy encoding step, since encryption eliminates redundancy and therefore increases the entropy.

Fig. 6.2 illustrates the detailed encryption of the prediction errors during the Draco geometry encoding phase. We note that the AES encryption scheme in CFB mode, described in Chapter 1, is used to encrypt the encoded prediction errors. However, in addition to the prediction errors, the prediction data contains other information, such as decoder initialization information for example, which is needed to correctly decode the vertices and consequently remain format compliant. Markers are therefore

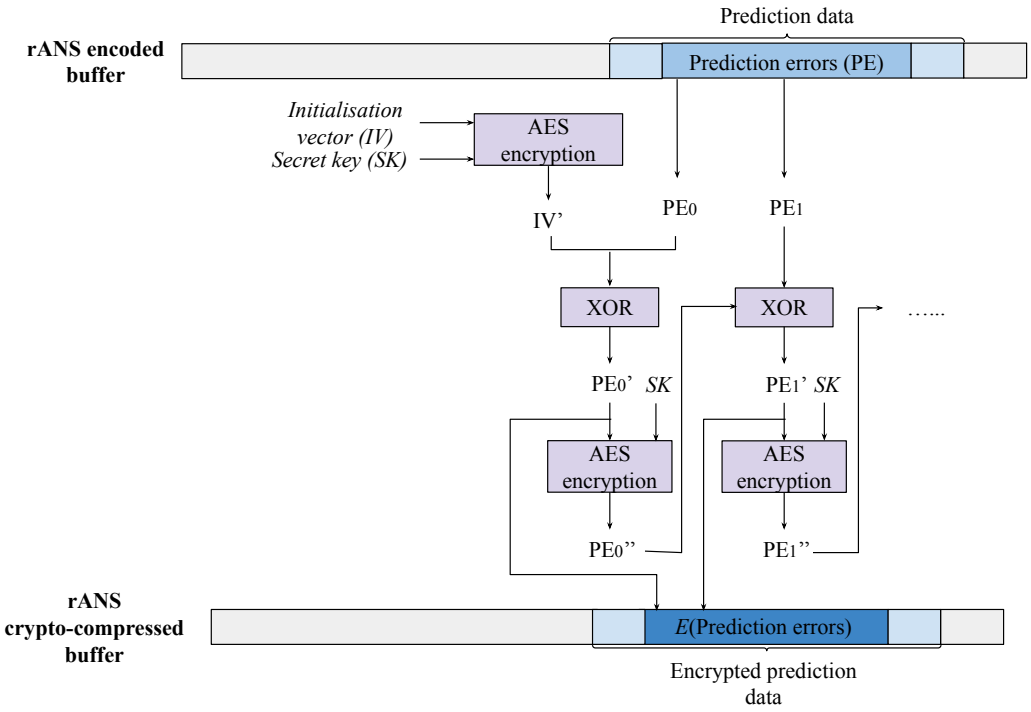


Figure 6.2: *Encryption of prediction errors using the AES in CFB mode.*

necessary to locate the prediction errors in the Draco output buffer. The structure of the prediction data and therefore the location of the prediction errors depend on several parameters such as qp and cl, but also on the characteristics of the 3D object. From these markers, we are then able to encrypt the correct segment of information after the rANS encoding.

Decoding phase

In order to successfully decode the 3D object, the decryption is performed jointly during the Draco decoding phase. Fig. 6.3 presents an overview of the decoding phase of the proposed Draco 3D object crypto-compression method. In order to avoid adding to the output buffer and therefore causing a size expansion, the decryption step is performed during the rANS decoding step. We note that the AES decryption in CFB mode is carried out before prediction errors are decoded by rANS (Fig 6.4). During the rANS decoding step, the decoder is first initialized using prediction error data that remains in the plaintext domain. Once the decoder has extracted the prediction errors, they are decrypted using the AES algorithm using the CFB mode. The decrypted prediction errors can then be decoded by the rANS decoder. After which, the decrypted prediction errors are then used to reconstruct the decrypted 3D object using a vertex prediction step and a vertex floating point reconstruction step.

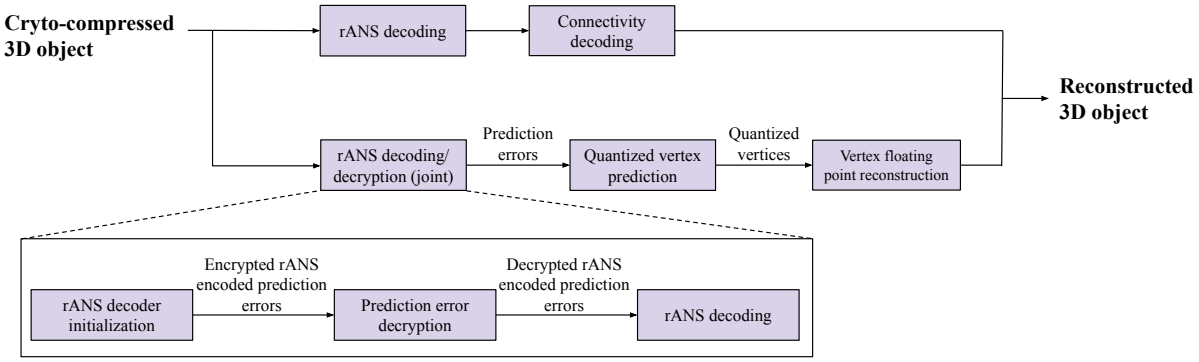


Figure 6.3: *Overview of the Draco decoding phase of the proposed Draco 3D object crypto-compression method.*

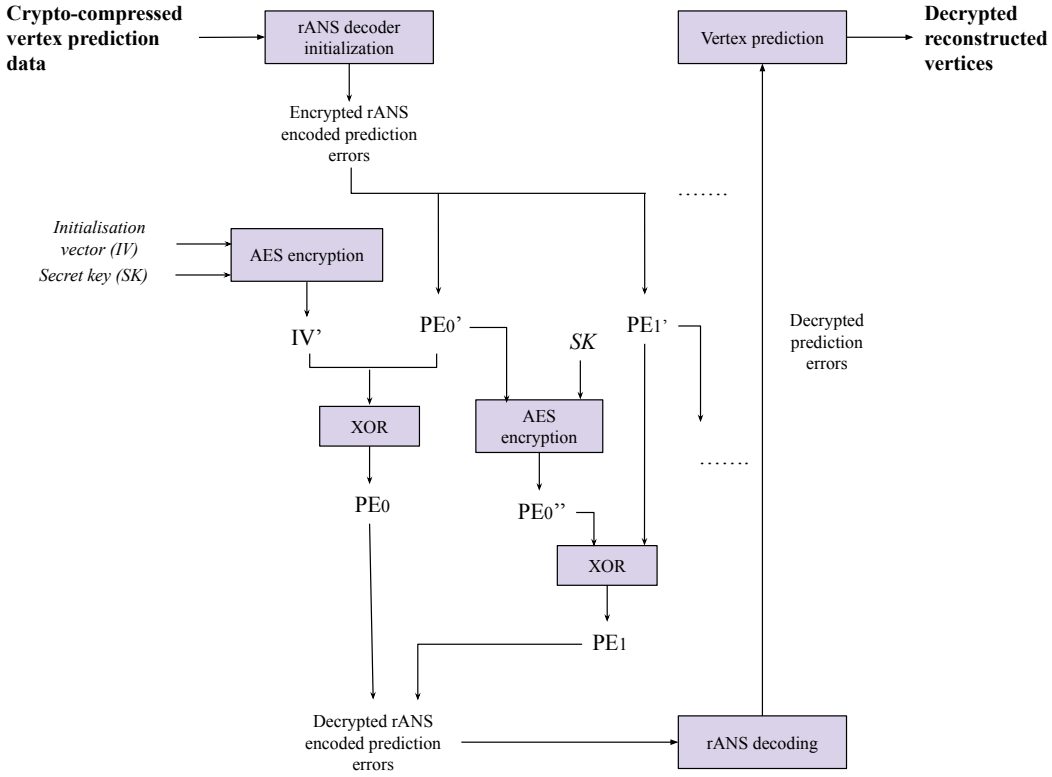


Figure 6.4: *The vertex reconstruction process.*

6.3.2 Experimental results

In this section, we detail our experimental results. First, we describe our results when our proposed method is applied to the Stanford dataset (34). We then perform a security analysis on our proposed method by performing a differential analysis, comparing the correlation between the Draco compressed 3D object and the crypto-compressed 3D object and analyzing the entropy.

We note an original 3D object O, O' corresponds to the decoded 3D object after a standard Draco compression, and O'_e corresponds to the decoded 3D object after a Draco crypto-compression.

Crypto-compression Results

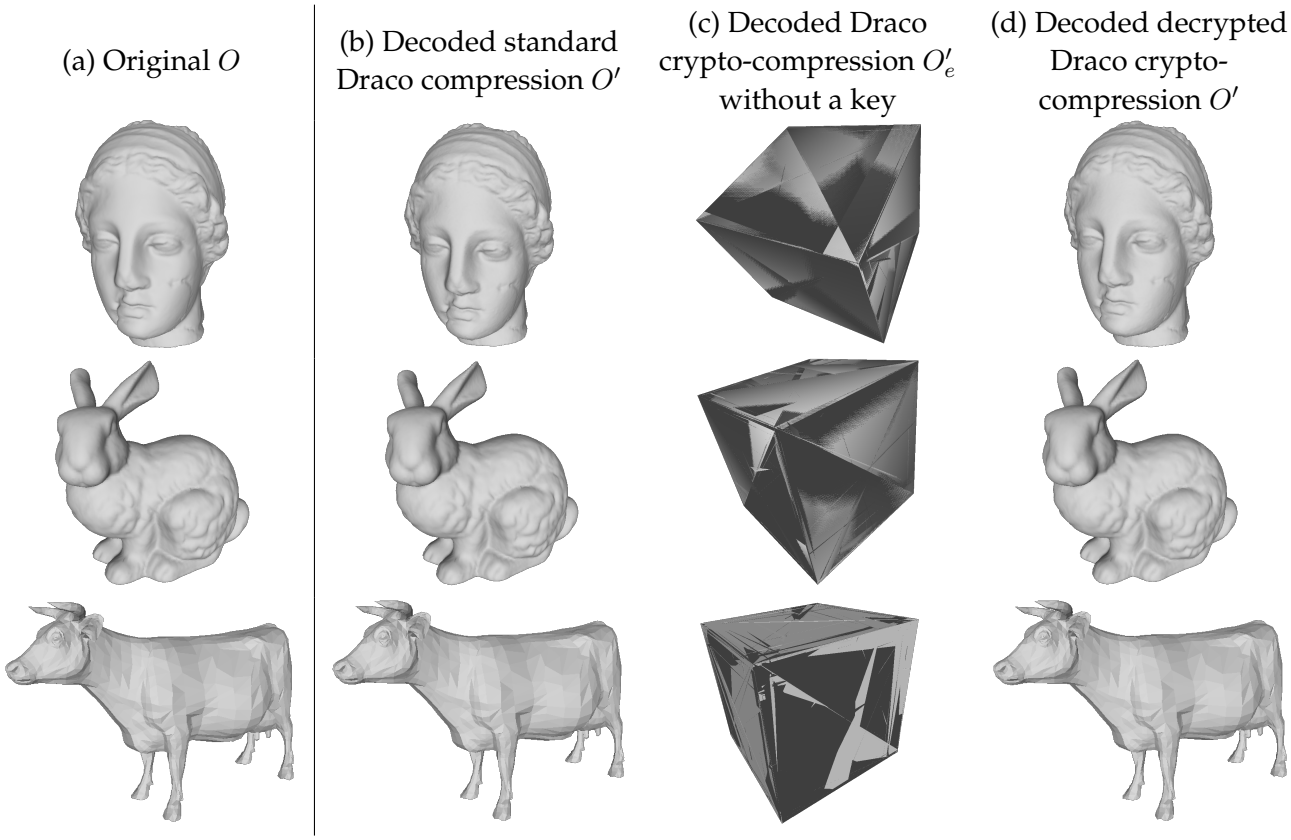


Figure 6.5: *Illustrations of the Draco 3D object crypto-compression scheme: a) The original 3D object O, b) The decoded standard Draco compressed 3D object O', c) The decoded Draco crypto-compressed 3D object O'_e without a key, and d) The decoded decrypted 3D object O'. The default Draco parameters $cl = 7$ and $qp = 11$ are used.*

Our experimentation was performed on the Stanford dataset (34). Each combination of cl and qp is applied to the 11 objects, resulting in a dataset of 3630 different 3D objects. Each 3D object is encrypted with a different pseudo randomly generated key.

Fig. 6.5 illustrates three examples of our proposed method, *Venus* ($100,759$ vertices), *Bunny* ($35,947$ vertices) and *Cow* (2,904 vertices). Fig 6.5.a shows the original 3D object O, then the decoded standard Draco compressed 3D object O' is shown in Fig 6.5.b, followed by the decoded Draco crypto-compressed 3D object O'_e without a key in Fig 6.5.c. Then, the decoded decrypted Draco crypto-compressed 3D object O' is shown in Fig 6.5.d. For these examples, the default parameters provided by Google for Draco are used: $cl = 7$ and $qp = 11$.

Our proposed method is fully reversible. There is no size expansion or data loss. The root mean squared error (RMSE) between a decoded standard Draco compressed 3D object O' and its corresponding decoded decrypted Draco crypto-compressed 3D object O'_e is always 0. This means that these obtained 3D objects are identical and our proposed crypto-compression method is fully reversible in relation to the Draco compression scheme. Therefore, we note both the decoded standard Draco compressed 3D object and its corresponding decoded decrypted Draco crypto-compressed 3D object O'.

Table 6.1: *Compression rates and Hausdorff distances (HD) between the original 3D objects of the Stanford dataset O and their decoded standard Draco compression O' and decoded Draco crypto-compression O'_e.*

	Compression rate	HD $(O, O')(10^{-4})$	HD (O, O'_e)
Mean	40.808	6.116	1.148
Std. dev	19.645	2.892	0.642
Min	14.399	0.805	0.100
Max	68.816	11.712	1.942

Table 6.1 presents compression rates and Hausdorff distances (HD) when comparing the original 3D objects of the Stanford dataset and their corresponding decoded standard Draco compressed O' and decoded Draco crypto-compressed O'_e 3D objects. We define the compression rate as the ratio between the size of the original 3D object and the size of the crypto-compressed 3D object. We note that our proposed Draco crypto-compression scheme has no size expansion in relation to the standard Draco compression scheme. Therefore the compression rate remains constant for O' and O'_e. The HD between O and O' remains low with an order of 10^{-4}.

We note that the proposed encryption method is efficient, as the AES encryption scheme is linear. The average time of the standard Draco compression encoding process of the 11 3D objects is $157.455\ ms$, while that of the proposed Draco crypto-compression method is $157.545\ ms$. We note that both were tested under the same conditions[1]. The average time of the standard Draco decoding process is $33.636\ ms$, while the average decoding and decryption time of the proposed Draco crypto-compression method is $34.818\ ms$. Therefore we conclude that the time added by the AES encryption and decryption is negligible.

Fig. 6.6 illustrates a full example of the compression rate according to the quantization level qp for each compression level cl for the 3D object *Venus*. The curve represents the decoded standard Draco compressed 3D object O', while the points represent the decoded Draco crypto-compressed 3D object. We observe that there are two major trends according to cl. This is explained by the type of encoding method used during Edgebreaker. For the parameters $cl \in [1-5]$ where the compression rate starts at about 75, the standard Edgebreaker encoding is used, while for parameters $cl \in [6-10]$, a valence encoding is used and the compression rate starts at about 150. We can observe that our proposed crypto-compression method has absolutely no size expansion, which means that the compression rate is preserved.

Fig. 6.7 illustrates the average RMSE between the Draco compressed 3D objects O' of the Stanford dataset and their corresponding crypto-compression O'_e according to the quantization level qp. We observe that apart from the low quantization levels where the 3D object is unrecognizable, the average RMSE remains similar around 1.25 when qp increases. We note that in Fig. 6.7, the default value of $cl = 7$ is used. However, we also note that the RMSE does not vary with the compression level cl.

[1] Algorithms were tested on a 6-core work station with 32 GB of RAM and a Windows based operating system.

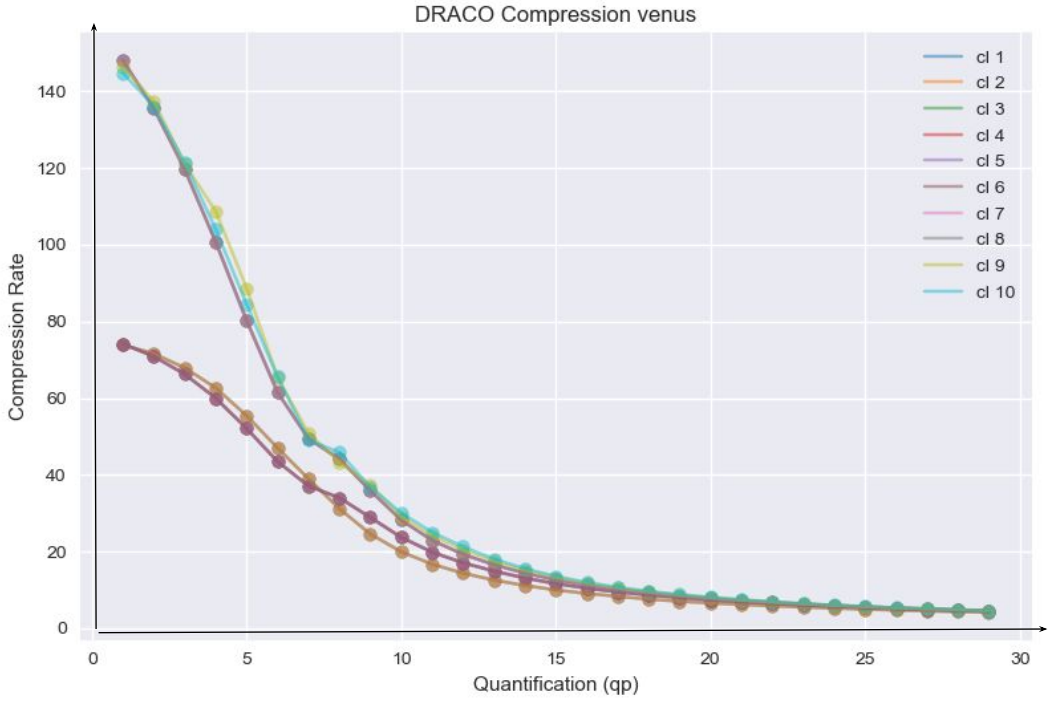


Figure 6.6: *Compression rate according to the quantization parameter qp for each compression level cl for both the decoded standard Draco compression and the decoded Draco crypto-compression of the 3D object Venus.*

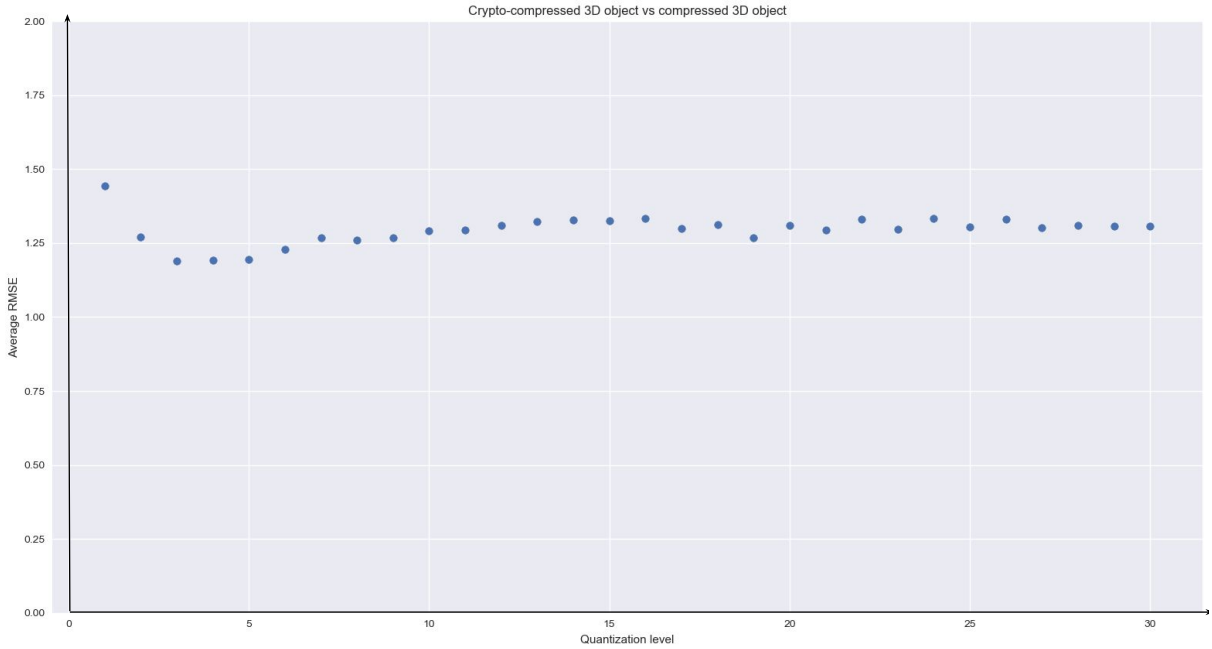
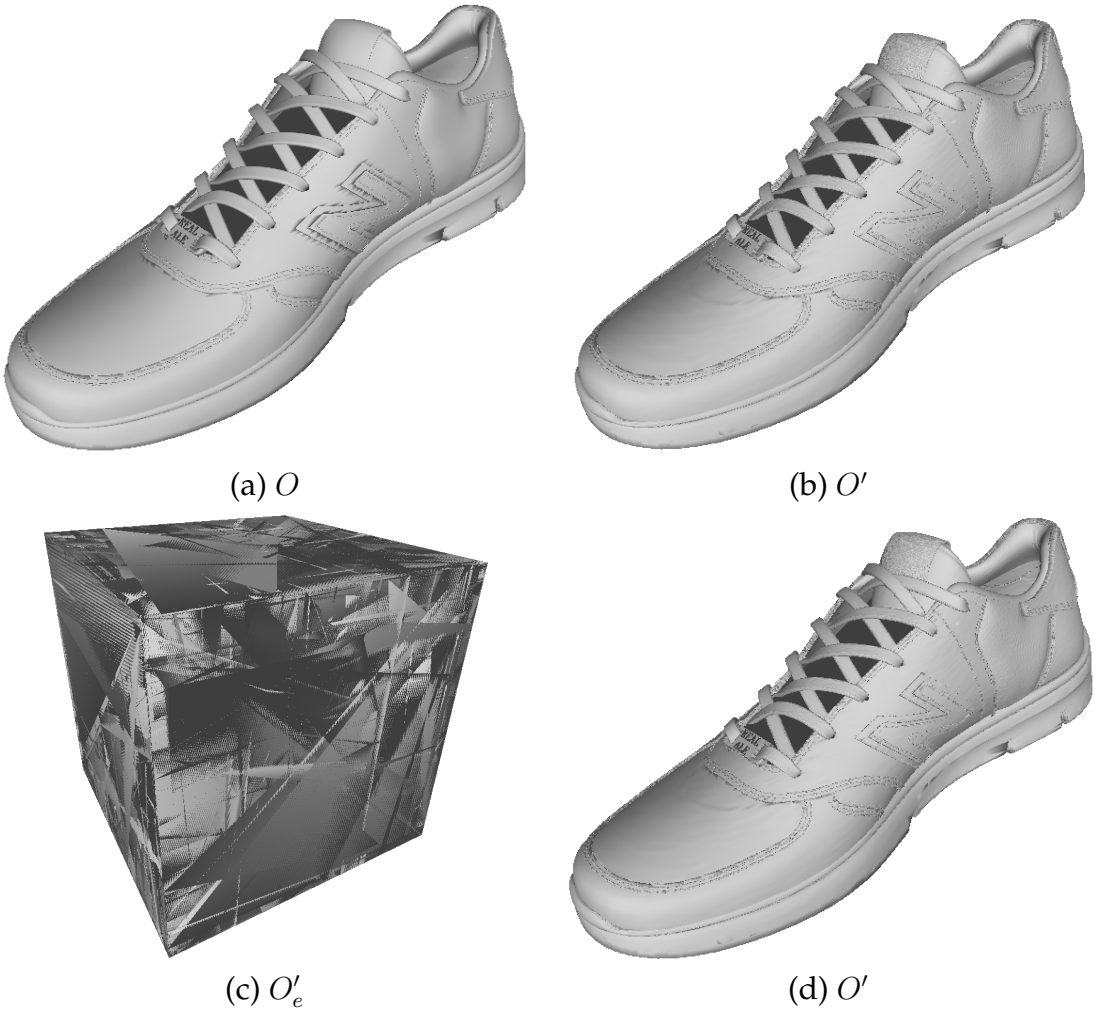


Figure 6.7: *Average RMSE between the decoded standard Draco compressed 3D objects of the Stanford datset O' and their corresponding decoded crypto-compression O'_e according the quantization level qp.*

Fig. 6.8 illustrates the results of the proposed Draco crypto-compression method on a manufactured 3D object *Shoe*[2] composed of $976,943$ vertices. Fig. 6.8.a presents the original manufactured 3D object and Fig. 6.8.b the decoded standard Draco com-

[2]Provided by STRATEGIES (https://www.romans-cad.com/)



(a) O (b) O'

(c) O'_e (d) O'

Figure 6.8: *Results of the proposed Draco crypto-compression method applied to a manufactured object Shoe (Provided by STRATEGIES (https://www.romans-cad.com/)): a) The original manufactured 3D object O, b) The decoded standard Draco compressed 3D object O', c) The decoded Draco crypto-compressed 3D object O'_e, d) The decoded decrypted Draco crypto-compressed 3D object O'.*

pressed 3D object O'. Fig. 6.8.c illustrates the decoded Draco crypto-compressed 3D object O'_e and Fig. 6.8.d the decoded decrypted Draco crypto-compressed 3D object O'. There is a compression rate of 34.75 and the RMSE between O' and O'_e is 273.817.

Security Analysis

We examine the security of the proposed crypto-compression scheme in terms of confidentiality. We first perform a differential analysis and measure the 3D object's correlation. Then we study the entropy of the decoded standard Draco compressed 3D object O' and the decoded Draco crypto-compressed 3D object O'_e.

Fig. 6.9 illustrates the differential analysis for the 3D object *Venus* with the default parameters $cl = 7$ and $qp = 11$. A Key_2 test is performed, where the 3D object is decrypted using keys where only a single bit of the correct 256 bit private key is flipped.

The x axis represents the index of the flipped bit in the private key, and the y axis represents the RMSE between the resulting 3D object and the decoded standard Draco compressed 3D object. We note that the key used in position 128 is the correct private key.

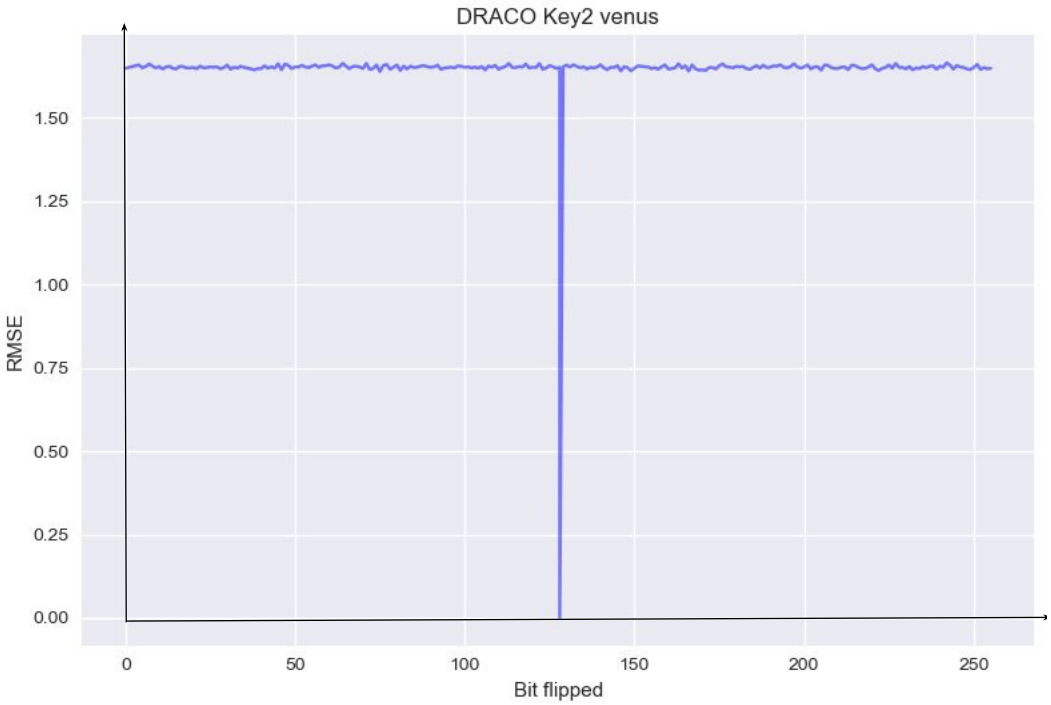


Figure 6.9: *Differential analysis of the 3D object Venus.*

We observe from Fig. 6.9 that while the correct key results in an RMSE of 0, and therefore a perfect decryption, any incorrect key results in an encrypted 3D object with an RMSE of around 1.65. This stable value means that an attacker cannot try to converge to the correct value. Therefore an incorrect key which has only a single false bit does not reveal any information about the 3D object. This is to be expected, since the AES encryption scheme is used.

Fig. 6.10 illustrates the correlation of the euclidean norms of vertices belonging to the decoded standard Draco compressed 3D object *Venus* in red, and the correlation of the euclidean norms of the vertices belonging to the decoded Draco crypto-compressed 3D object *Venus* in blue. Fig. 6.11 presents the same for the 3D object *Cow* and Fig. 6.12 the same for *Bunny*. We note that the default Draco parameters of $cl = 7$ and $qp = 11$ were used. The correlation was performed between pairs of neighboring vertices, where the neighbors are defined by the Edgebreaker scheme and are separated into two groups u and v. We note that this division into groups was performed in the same manner for both the decoded standard Draco compressed 3D object O' and the decoded Draco crypto-compressed 3D object O'_e.

We observe that there is a strong correlation between neighboring vertices in the decoded standard Draco compressed 3D object O'. However, we can also note that there is a large portion of uncorrelated data. This can be explained by the vertex order defined by the the Edgebreaker scheme. Once a loop is completed, there can be a jump to the start of the next loop. In comparison, we observe that when the 3D object is

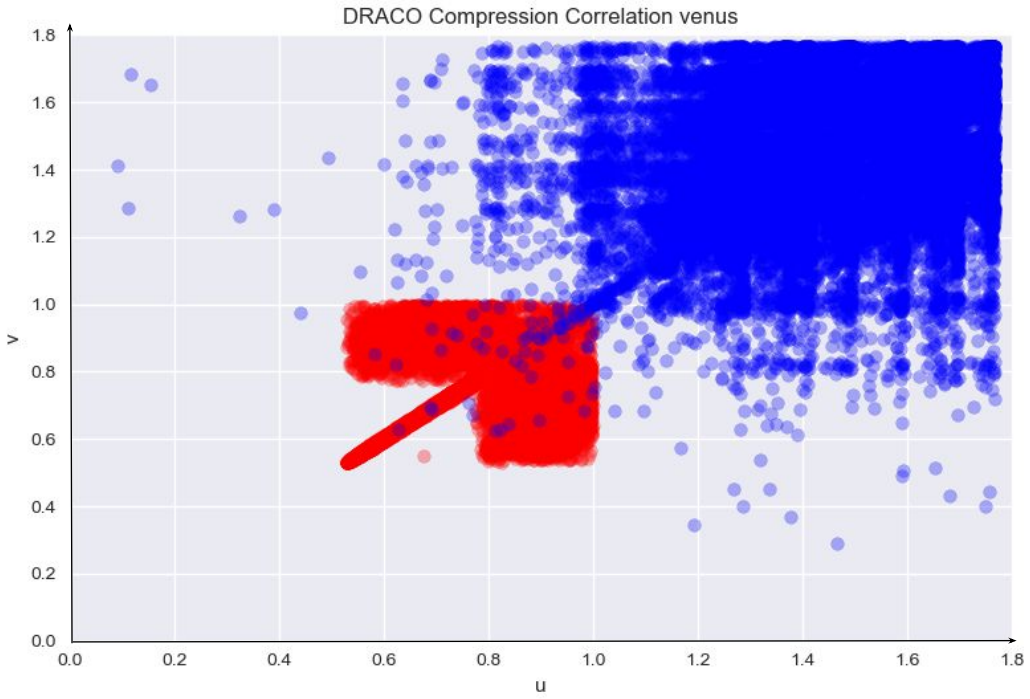


Figure 6.10: *Correlation of the decoded standard Draco compressed 3D object Venus (red), and the correlation of the corresponding decoded Draco crypto-compressed 3D object (blue) with the parameters* $cl = 7$ *and* $qp = 11$.

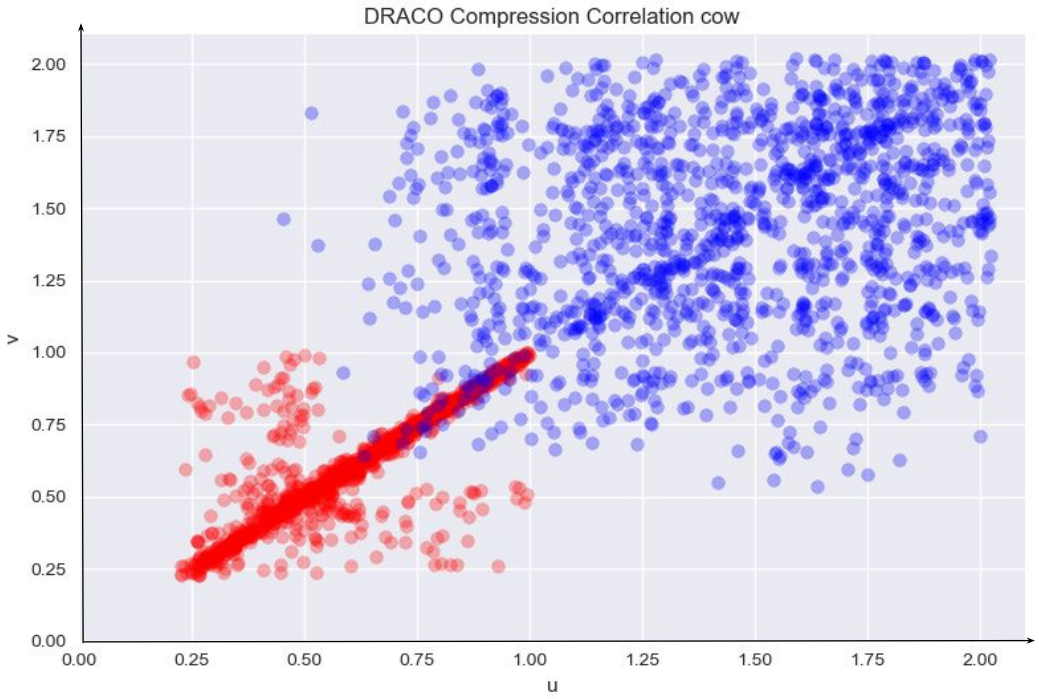


Figure 6.11: *Correlation of the decoded standard Draco compressed 3D object Cow (red), and the correlation of the corresponding decoded Draco crypto-compressed 3D object (blue) with the parameters* $cl = 7$ *and* $qp = 11$.

crypto-compressed, the neighboring vertices are no longer correlated.

The Pearson correlation coefficient between the vertices of O' and O'_e is 0.004 for

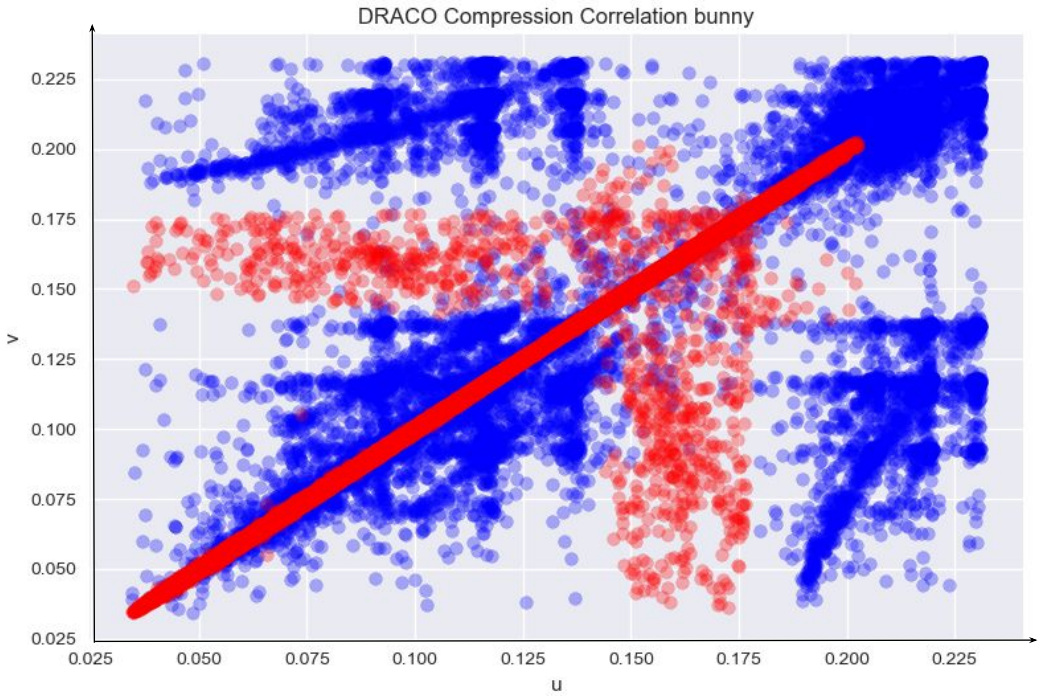


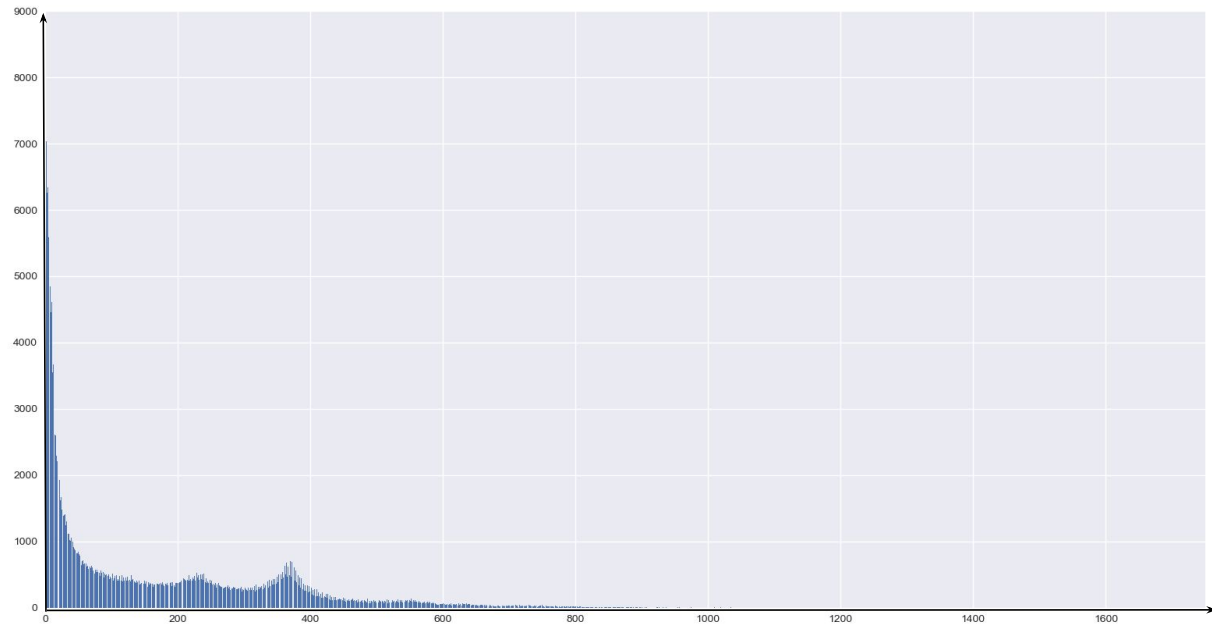
Figure 6.12: *Correlation of the decoded standard Draco compressed 3D object Bunny (red), and the correlation of the corresponding decoded Draco crypto-compressed 3D object (blue) with the parameters* $cl = 7$ *and* $qp = 11$.

Venus, 0.043 for *Cow*, and 0.510 for *Bunny*. These results show that the Draco crypto-compressed 3D objects O'_e is not correlated with its corresponding 3D object O' in the plaintext domain.

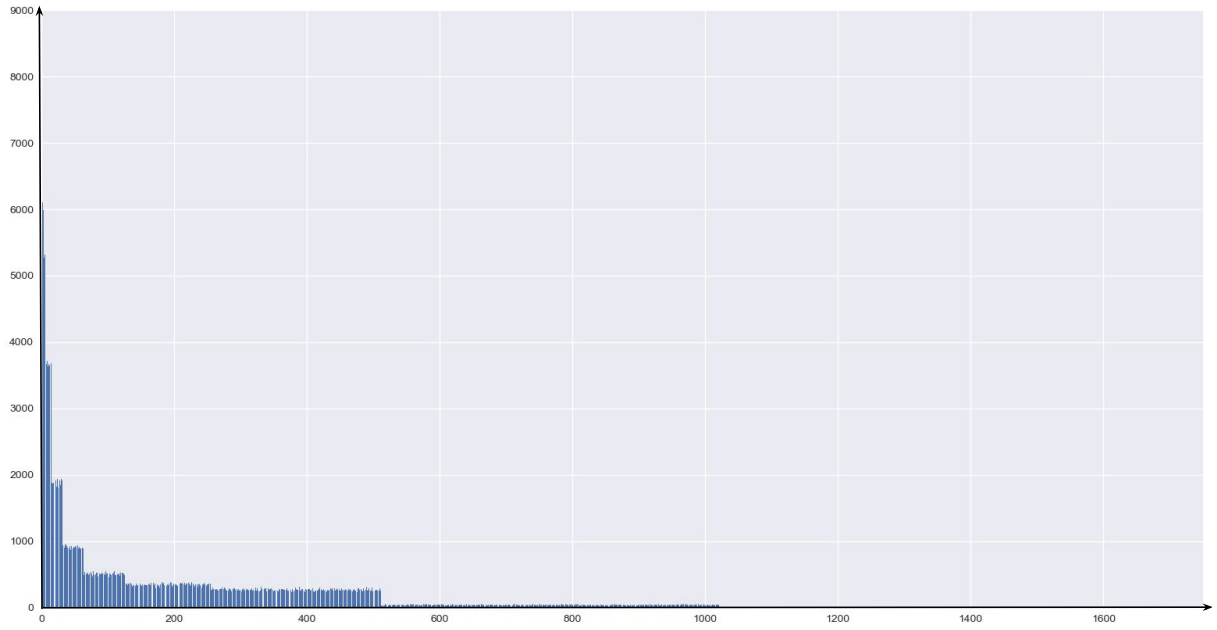
Fig. 6.13.a presents the prediction error distribution of the decoded standard Draco compressed 3D object *Venus* while Fig. 6.13.b presents the prediction error distribution of the decoded Draco crypto-compressed 3D object *Venus*, where the parameters are $cl = 7$ and $qp = 16$. Fig. 6.14.a shows the prediction error distribution of the decoded standard Draco compressed 3D object *Bunny* and Fig. 6.14.b shows the prediction error distribution of the decoded Draco crypto-compressed 3D object *Bunny*, where the parameters are $cl = 7$ and $qp = 16$.

We observe in Fig. 6.13.b and Fig. 6.14.b that after the 3D object has been encrypted, we obtain a histogram which is uniform within intervals of powers of 2. The entropy between the interval $2^8 + 1$ and 2^9 is 7.79 bits/prediction error for the standard Draco compressed 3D object *Venus* and 7.90 for *Bunny*, and a near maximum value of 7.99 bits/prediction error for the Draco crypto-compressed 3D object *Venus* and 7.97 for *Bunny*.

From this security analysis, we can conclude that our proposed Draco crypto-compression method is secure in terms of confidentiality.

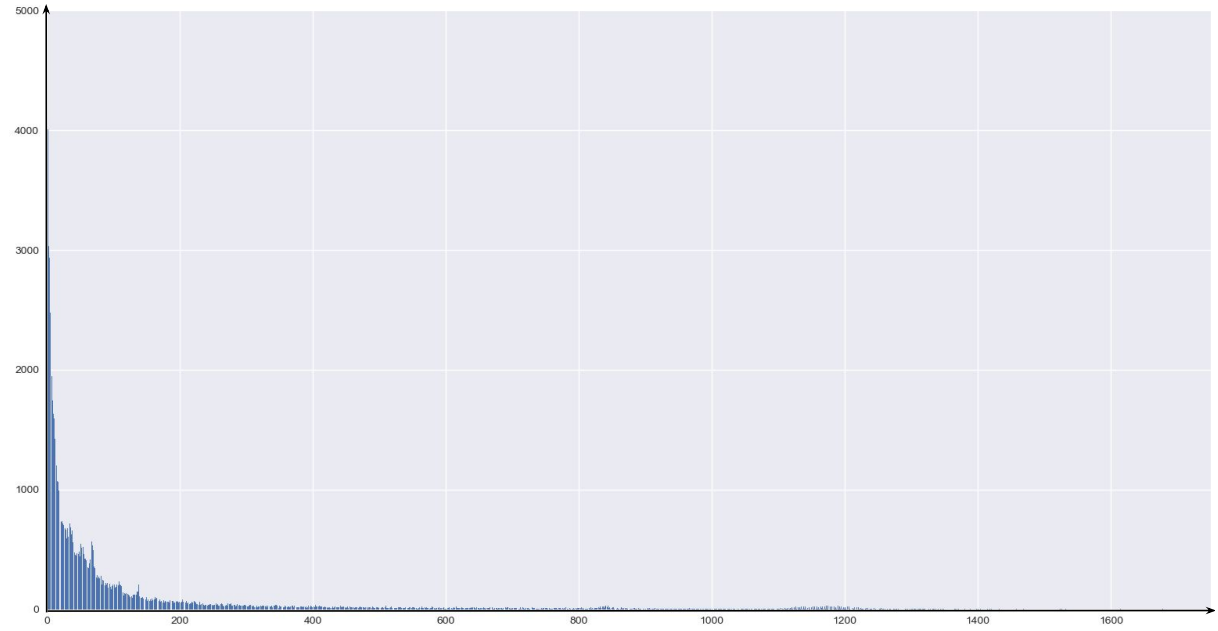


(a) *Venus* O'

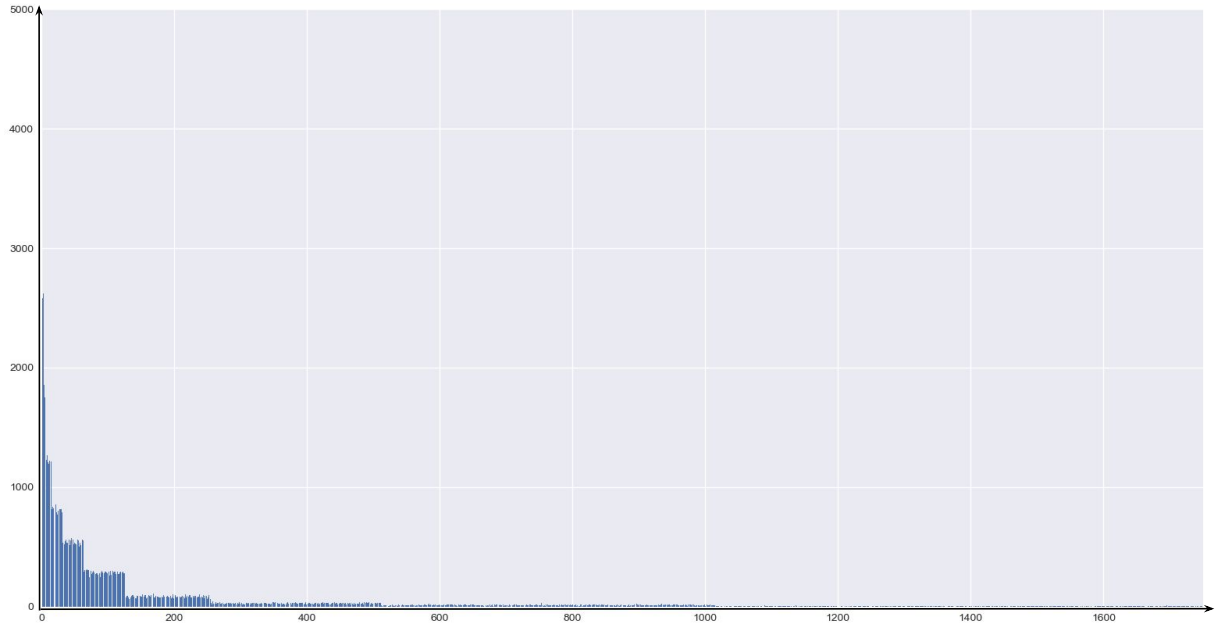


(b) *Venus* O'_e

Figure 6.13: *Prediction error distribution for: a) The decoded standard Draco compressed 3D object Venus* O'*, b) The corresponding crypto-compressed 3D object* O'_e *(parameters* $cl = 7$ *and* $qp = 16$*).*



(a) *Bunny* O'



(b) *Bunny* O'_e

Figure 6.14: *Prediction error distribution for: a) The decoded standard Draco compressed 3D object Bunny* O'*, b) The corresponding crypto-compressed 3D object* O'_e *(parameters* $cl = 7$ *and* $qp = 16$*).*

6.4 Selective Crypto-compression

In this section we present our second contribution. As described in Chapter 1, selective encryption methods are used to encrypt the 3D object according to a desired visual security level. In this contribution, we integrate a selective encryption step in the 3D object compression method Draco. First, in Section 6.4.1, we describe the proposed Draco 3D object selective crypto-compression method, where a selective encryption step is embedded in the Draco geometry encoding phase, between the vertex quantization step and the vertex prediction step. The Draco selectively crypto-compressed 3D object is then decrypted during the geometry decoding phase. Then, in Section 6.4.2, we present the experimental results for our proposed method.

6.4.1 The Proposed 3D object Selective Crypto-compression Method

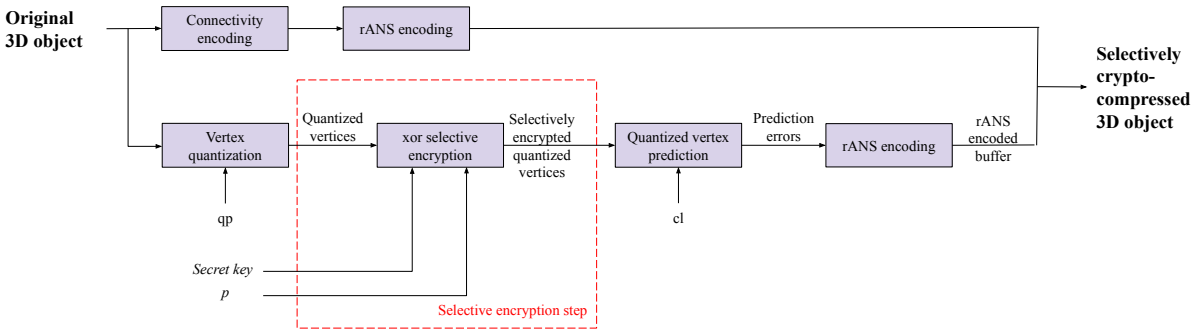


Figure 6.15: *Overview of the proposed Draco-based selective crypto-compression method of 3D objects.*

Fig. 6.15 presents an overview of the proposed selective crypto-compression method for 3D objects based on Draco. Like for the first contribution, only the geometry encoding phase is modified. After the 32 bit floating point vertices have been quantized, a selective encryption step is integrated. This XOR selective encryption method is largely based on the selective encryption method described in (47), which is presented in detail in Chapter 1 and relies on a secret key and a degradation parameter p.

Selective Encryption

During the vertex quantization step which takes place during the Draco geometry encoding phase, the vertices v are transformed from 32 bit floating points to unsigned integers v' according to the quantization parameter qp:

$$v' = (v - v_{min}) \times \frac{2^{qp}}{range}, \tag{6.1}$$

where v is the the original floating point x, y or z coordinate, v' is the corresponding quantized coordinate, v_{min} is the minimum corresponding x, y or z coordinate, and $range$ is the greatest edge of the bounding box. We note the quantization parameter

qp, where $qp \in [0, 30]$, corresponds to the number of bits conserved per coordinate, except for $qp = 0$, which signifies that there is no quantization. Consequently, after quantification, $v' \in [0, 2^{qp}]$.

In order to perform a selective encryption, we want to encrypt p LSB of each quantized coordinate v'. Therefore, we propose then dividing the quantized coordinates v' into two parts:

- v'_p are the p LSB of v' which are to be encrypted.
- v'_c are the $qp - p$ MSB of v' which remain in the plaintext domain.

Therefore, we have:

$$v' = v'_c + v'_p, \tag{6.2}$$

where $+$ corresponds to a concatenation.

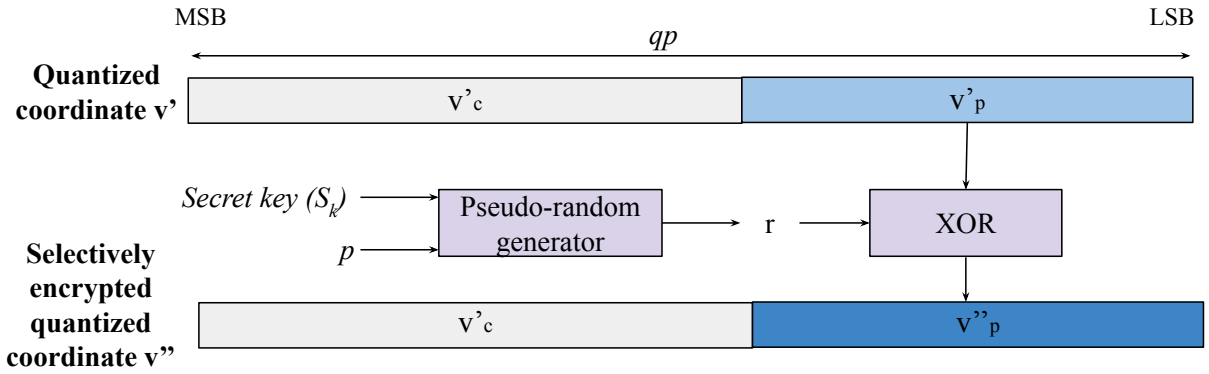


Figure 6.16: *The selective encryption of a quantized coordinate v'.*

Fig. 6.16 illustrates the selective encryption of a quantized coordinate v'. The selective encryption is performed with an XOR between v'_p, which is the p LSB of a coordinate, and a pseudo-random sequence r generated according to a 256 bit secret key:

$$v''_p = v'_p \oplus r, \tag{6.3}$$

where v'_p is the p LSB of v', r is generated by the pseudo-random generator and v''_p is the corresponding encryption. We then obtain the selectively encrypted quantized coordinate v'' by substituting v'_p with v''_p:

$$v'' = v'_c + v''_p, \tag{6.4}$$

where $+$ corresponds to a concatenation. We note that $p \in [0, qp]$. When $p = 0$, this is equivalent to a standard Draco compression. When $p = 1$, only the LSB of each coordinate is encrypted and when $p = qp$, this is equivalent to a full encryption.

After the selective encryption step, there is a vertex prediction step in Draco, where only the prediction errors are conserved (Fig. 6.15). These prediction errors then undergo a rANS encoding step. It is because of this entropy encoding step that the selective encryption becomes a trade-off between the compression rate and the security in terms of confidentiality. An analysis of this trade-off is detailed in the experimental results presented in Section 6.4.2.

Decryption

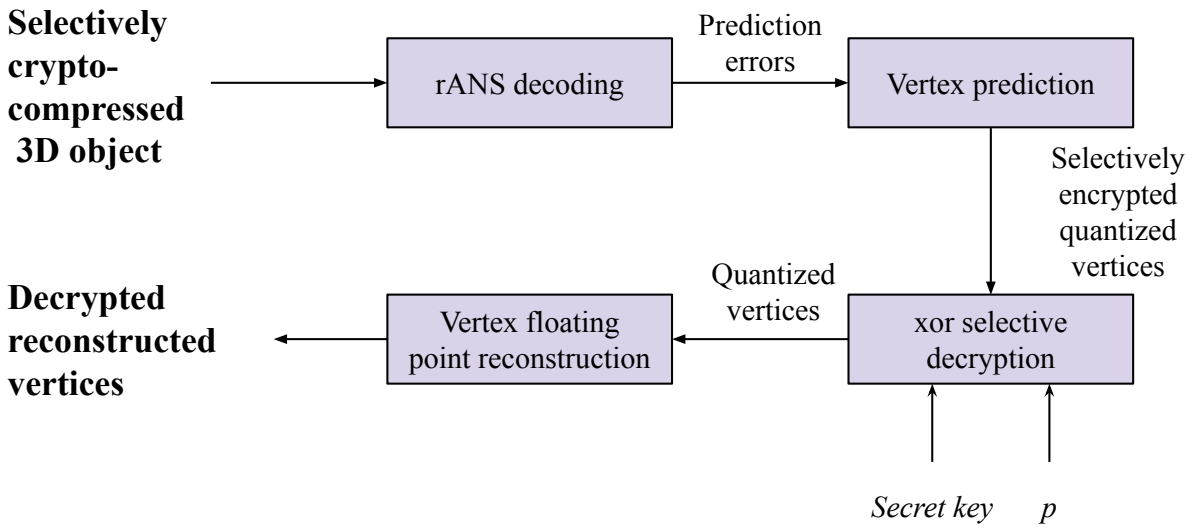


Figure 6.17: *The proposed Draco-based selective crypto-compression geometry decoding process.*

Fig. 6.17 presents an overview of the geometry decoding process of the proposed Draco-based selective crypto-compression system. First, the selectively crypto-compressed 3D object undergoes a rANS decoding step where the prediction errors are retrieved. Using these prediction errors, the selectively encrypted quantized vertices can be reconstructed with a vertex prediction step.

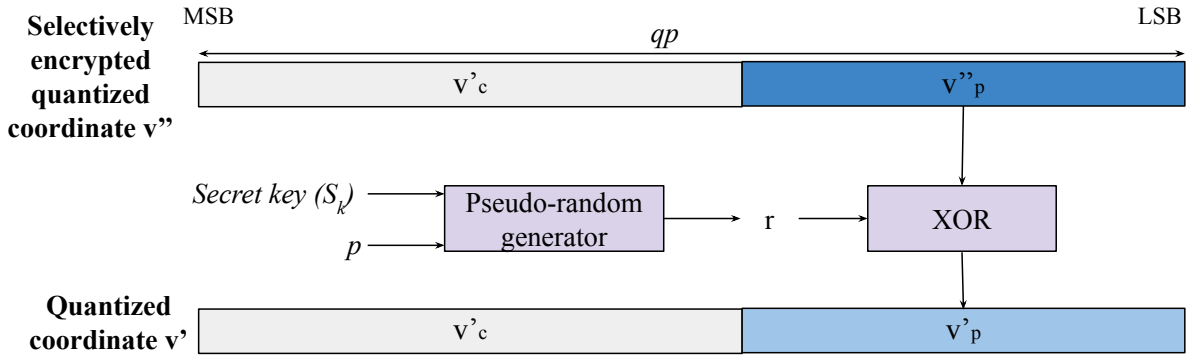


Figure 6.18: *The decryption of a selectively encrypted quantized coordinate v'.*

As illustrated in Fig. 6.18, in order to decrypt a selectively encrypted quantized coordinate v'', we use the same XOR operation as Eq. 6.3 to decrypt v''_p:

$$v'_p = v''_p \oplus r. \tag{6.5}$$

The decrypted p LSB v'_p are then substituted into v'', which results in the decrypted quantized coordinate v'. The floating point coordinate v_f can then be reconstructed using the inverse of Eq. 6.1:

$$v_f = v' \times \frac{range}{2^{qp}} + v_{min}, \tag{6.6}$$

where v_{min} is the minimum corresponding x, y or z coordinate, and $range$ is the greatest edge of the bounding box.

6.4.2 Experimental Results

In this section, we detail the experimental results of the proposed Draco 3D selective crypto-compression method. Our experimentation was carried out on the Stanford dataset which is composed of 11 3D objects (34). First, we present our results on the Stanford dataset using the default parameters given by Google ($qp = 11$, $cl = 7$). Then, we analyze the trade-off between compression rate and the 3D object's distortion.

Selective Crypto-compression

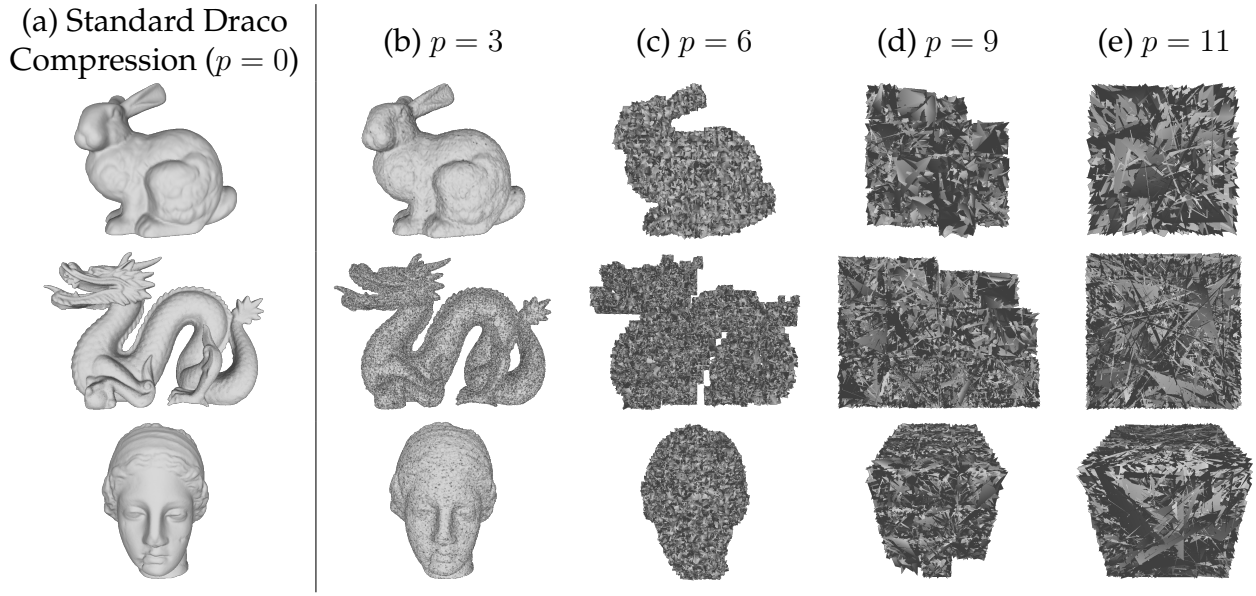


Figure 6.19: *Illustrations of the Draco 3D object selective crypto-compression scheme: a) The standard Draco compressed 3D objects, where $p = 0$, p corresponding to the number of LSB encrypted, b-e) The decoded Draco-based selectively crypto-compressed 3D object according to the selective encryption parameter p ($p = 3, p = 6, p = 9, p = 11$). The default Draco parameters $cl = 7$ and $qp = 11$ are used.*

Fig. 6.19 presents our results on three 3D objects from the Stanford dataset *Bunny*, *Dragon* and *Venus*. In Fig. 6.19.a we can see the standard Draco compressed 3D objects. We note that the default Draco parameters given by Google $cl = 7$ and $qp = 11$ are used. The selectively encrypted 3D objects with the respective selective encryption parameters $p = 3$ (Fig. 6.19.b), $p = 6$ (Fig. 6.19.c), $p = 9$ (Fig. 6.19.d) and $p = 11$ (Fig. 6.19.e) are presented. Since we have used $qp = 11$, $p = 11$ is therefore equivalent to a full encryption. We can observe that when the selective encryption parameter p increases, the distortion of the 3D object, and therefore security in terms of confidentiality, also increases. We note that the proposed method is fully reversible in relation to the standard

Draco compression method and therefore, upon decryption, we retrieve the decoded standard Draco compressed 3D object (Fig. 6.19.a).

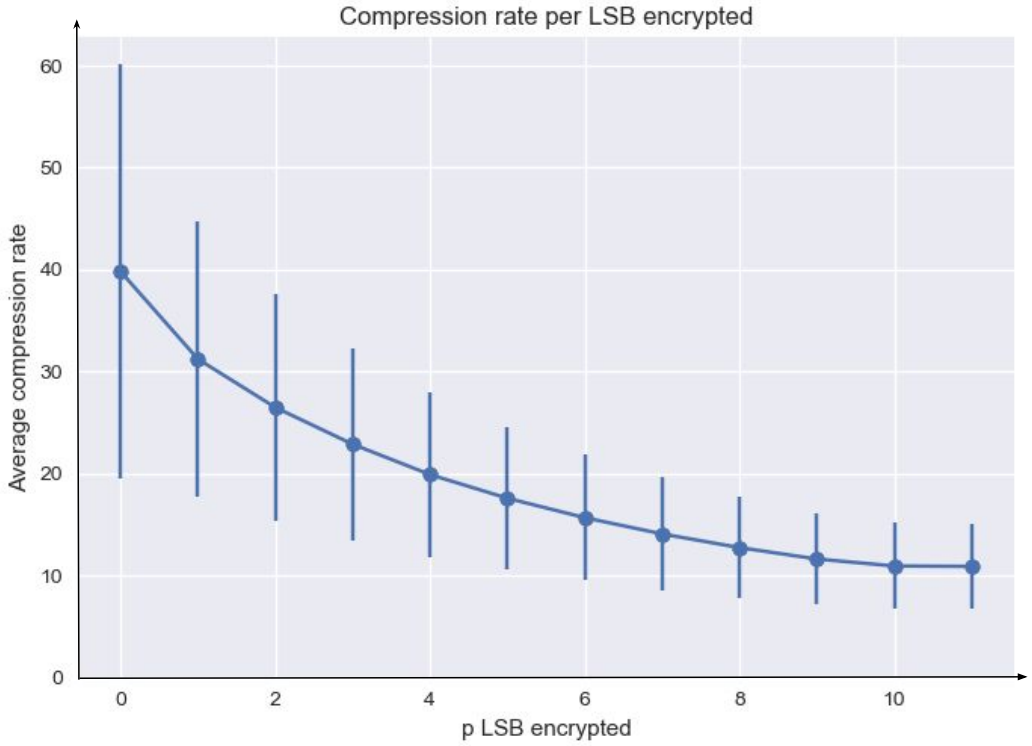
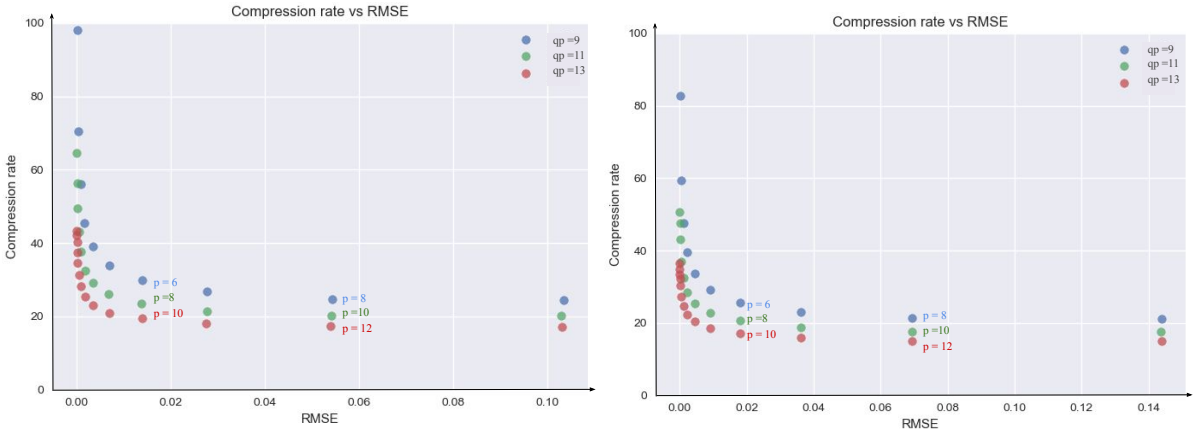


Figure 6.20: *Average compression rate and standard deviation of the Stanford dataset according the number p of LSB encrypted, where the default Draco parameters $cl = 7$ and $qp = 11$ are used.*

Fig. 6.20 illustrates the average compression rate of the 11 3D objects of the Stanford dataset, along with the standard deviations. We observe that when $p = 0$, and so the number of LSB encrypted is 0, the compression rate is about 40 with a standard deviation of about 20. This 3D object corresponds to the standard Draco compression. When $p = 4$, the average compression rate is about 20 with a standard deviation of about 8, however when $p = 11$, the average compression rate is about 11, with a lower standard deviation of about 4. This corresponds to a full encryption.

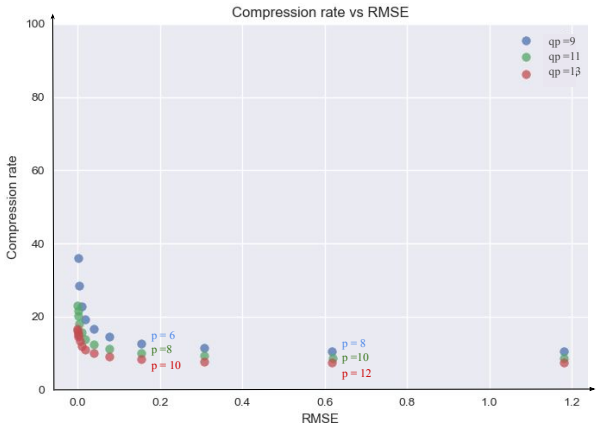
Analysis

Fig. 6.21a, Fig. 6.21b and Fig. 6.21c present the compression rate according to the RMSE for each selective encryption of *Bunny, Dragon* and *Venus* respectively with the selective encryption parameter $p \in [0, qp]$, for $qp = 9$ in blue, $qp = 11$ in green and $qp = 13$ in red. We observe that the selective encryption is a trade off between the compression rate and the RMSE. For a slight increase of p, when the value of p is small, the compression rate is quickly lost for a slight increase in the RMSE. However, when the value of p is near that of qp, the RMSE can be largely increased without sacrificing the compression rate. The choice of the parameter p depends on whether the user prefers to favor the



(a) *Bunny*

(b) *Dragon*



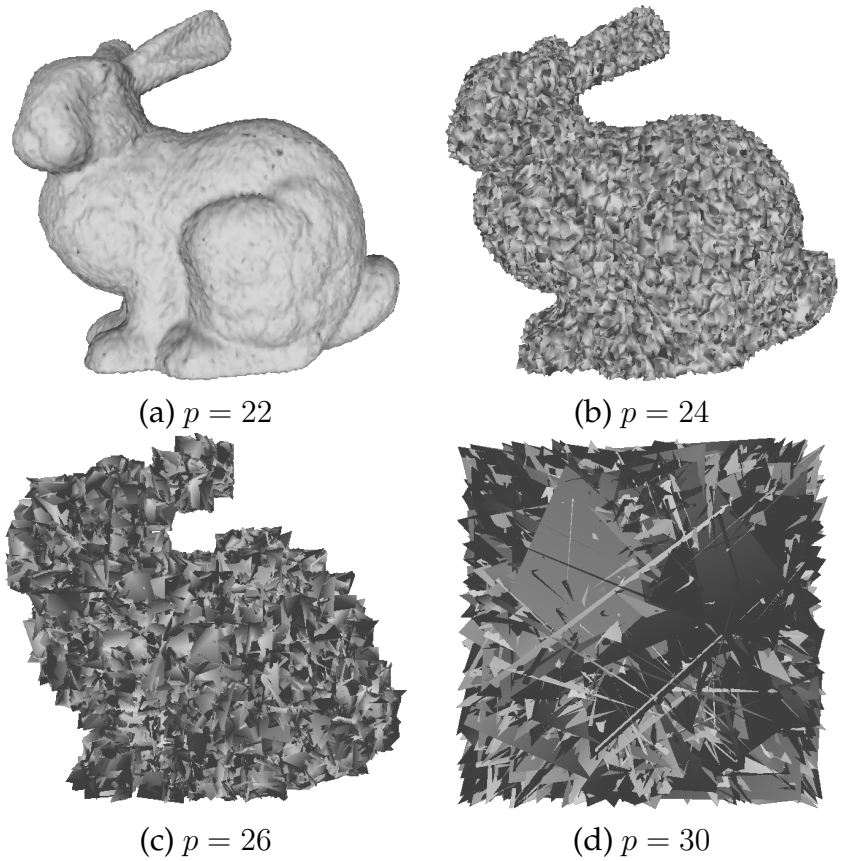
(c) *Venus*

Figure 6.21: *Compression rate according to RMSE for the 3D objects Bunny, Dragon and Venus respectively, where the parameters used are* $cl = 11$, $qp = 9$ *(blue),* $qp = 11$ *(green), and* $qp = 13$ *(red), and the value of the encryption parameter* $p \in [0, qp]$.

security or the compression rate. We also observe when qp is high, we sacrifice less of the compression rate when p is increased.

In Fig. 6.22, we observe that the effect of the LSB parameter p is strongly correlated to the value of qp. In Fig. 6.22.b, we see that the resulting decoded 3D object when $p = 22$ and $qp = 30$ which has an RMSE of 0.432×10^3 is visually similar to the decoded selectively crypto-compressed 3D object when $p = 3$ and $qp = 11$ (Fig. 6.19.a) which have RMSE values of 0.426×10^{-3} for *Bunny*. This is because 8 MSB are conserved when $p = 22$ and $qp = 30$ (Fig. 6.22.a) and when $p = 3$ and $qp = 11$ (Fig. 6.19.b). When the number of MSB conserved is near 0, the visual confidentiality of the decoded 3D object rapidly increases. The visual security does not depend on the number of LSB encrypted, but rather on the number of MSB left in the plaintext domain.

Fig. 6.23 presents the average compression rate of the 11 3D objects of the Stanford dataset according to the number of LSB encrypted p. The parameters $cl = 11$ and **qp = 30** are used. We can observe that when the quantization parameter is high, in this case $qp = 30$, the average compression is lower. The standard Draco compression has an average compression rate of about 5.75 and a standard deviation of about 2. However, when it is fully encrypted ($p = 30$), there is an average compression rate of



(a) $p = 22$ (b) $p = 24$

(c) $p = 26$ (d) $p = 30$

Figure 6.22: *Illustrations of the Draco 3D object selective crypto-compression scheme where the parameters $cl = 7$,* ***qp = 30*** *and p corresponds to the number of LSB encrypted: a-d) the decoded Draco-based selectively crypto-compressed 3D object according to the selective encryption parameter p ($p = 22$, $p = 24$, $p = 26$, $p = 30$).*

about 4 and a standard deviation of about 1.5.

Fig. 6.24a presents the compression rate and Fig. 6.24b the RMSE of the 3D object *Bunny* according to qp. The points in blue show the compression rate when $p = 0$, which corresponds to a standard Draco compression, the points in red show the compression rate when $p = \lfloor \frac{qp}{2} \rfloor$, and the points in green show the compression rate when $p = qp$, which corresponds to a full encryption. In these illustrations, we use a minimum value of $qp = 5$, as we consider that for our purposes, $qp < 5$ produces a 3D object with too poor a quality to be of interest. We observe in Fig. 6.24a that the two curves converge towards one other when we increase qp. This signifies that the trade-off between visual security in terms of confidentiality and the compression rate becomes less significant as qp increases. In Fig. 6.24b we observe that when p is small, the RMSE are more similar and when qp increases, the value of p has to be much higher in order for it to have an impact on the RMSE. However, when p is large, a slight increase of p results in a much higher RMSE. These observations made from Fig. 6.24a and Fig. 6.24b confirm what we observed in Fig. 6.21a, Fig. 6.21b and Fig. 6.21c.

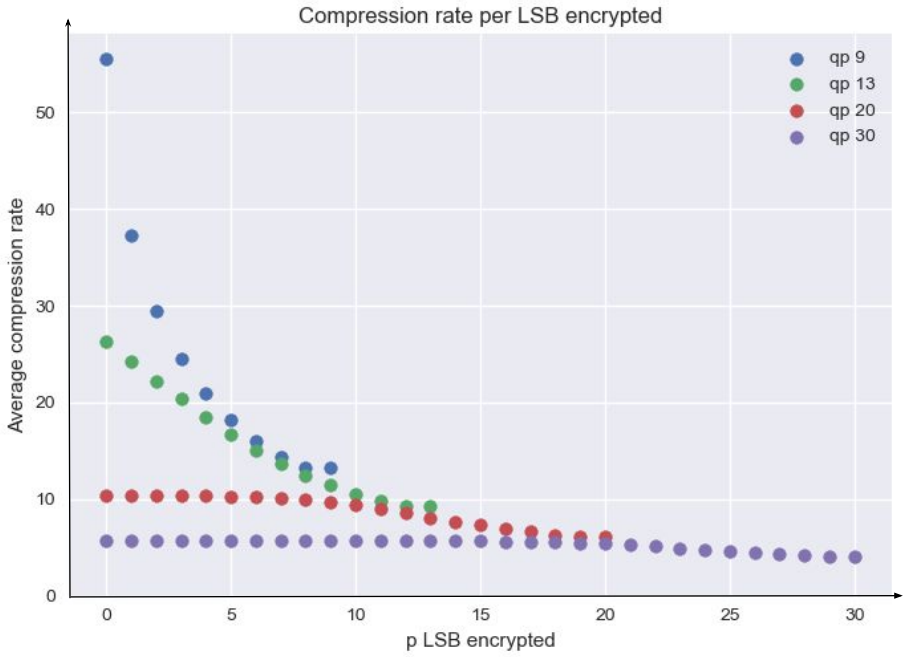
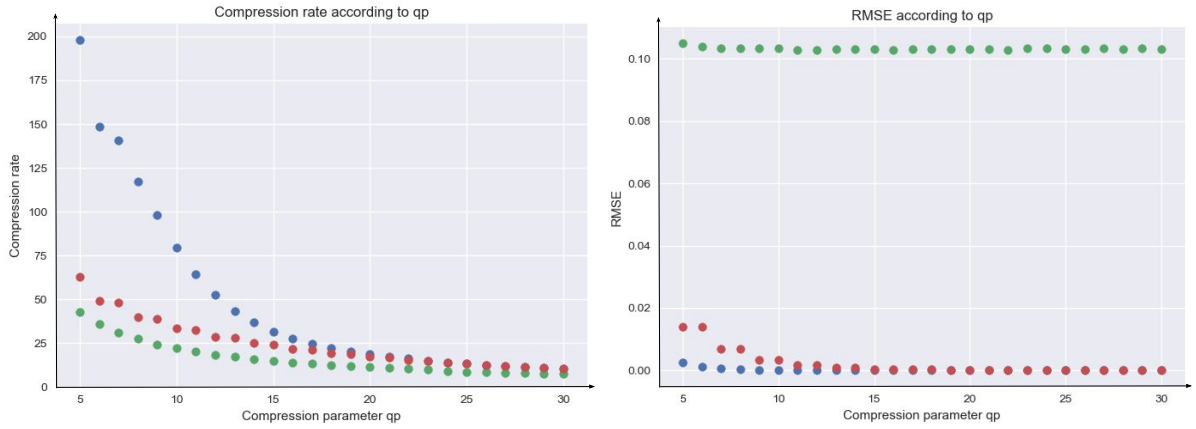


Figure 6.23: *Average compression rate of the Stanford dataset according to the number p of LSB encrypted, with the parameters $cl = 7$ and $qp = 9$ (blue), $qp = 13$ (green), $qp = 20$ (red) and $qp = 30$ (purple).*



(a) *Compression rate* **(b)** *RMSE*

Figure 6.24: *Compression rate and RMSE of the 3D object Bunny, according to qp, where $p = 0$ (blue, no encryption), $p = \frac{qp}{2}$ (red) and $p = qp$ (green, full encryption).*

6.5 Joint Watermarking and Compression

In this third contribution, we propose a joint watermarking and compression method for 3D objects based on Draco. We propose integrating the watermarking step between the vertex quantization step and the vertex prediction step of the encoding phase of the Draco 3D object compression method. We note that this proposed method results in a watermarked Draco format file. The watermark is then extracted during Draco's decoding phase, before the vertices are reconstructed. However, due to the fact that the watermark is not removed from the Draco format file, it is also possible to extract

the watermark once the 3D object is reconstructed.

First, in Section 6.5.1, we describe the proposed joint watermarking and Draco 3D object compression method. Experimental results are then provided in Section 6.5.2.

6.5.1 The proposed joint 3D object watermarking and compression method

Fig. 6.25 illustrates an overview of the encoding phase of the proposed method. Like for the first two contributions, we propose integrating the watermark embedding step in the geometry encoding phase and we do not interfere with the connectivity encoding phase. And like for the selective crypto-compression method, the watermarking step is integrated after the vertex quantization step. After embedding the watermark, the quantized vertices undergo a vertex prediction step, thus preserving the information embedded in the 3D object. Then, the watermarked and Draco compressed 3D object is produced after the entropy encoding.

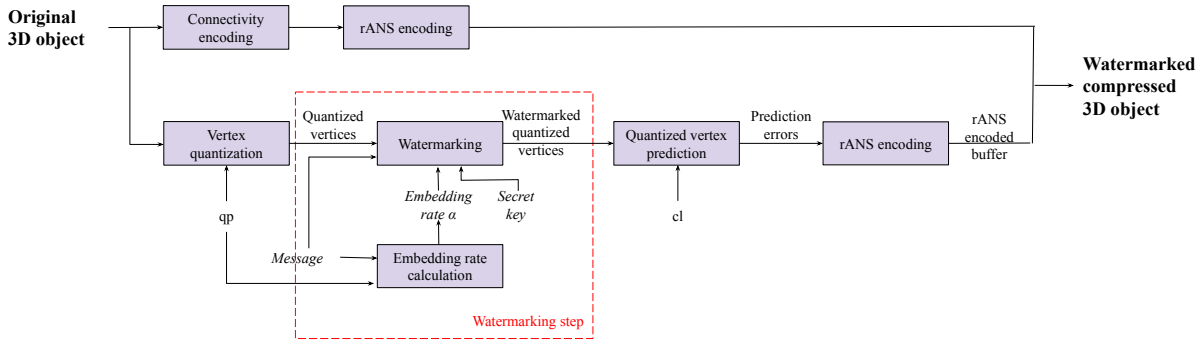


Figure 6.25: *Overview of the encoding phase of the proposed joint Draco 3D object compression and watermarking method.*

During the geometry encoding phase, the coordinates x, y and z of each vertex are first quantized according to the Draco quantization parameter $qp \in [0, 30]$ (Eq. 6.1). We note that the value of qp is a trade-off between the compression rate and the reconstructed 3D object's quality. The quantized coordinates are then subjected to a vertex prediction step according to the compression level parameter $cl \in [0, 10]$. A rANS encoding step is performed after the connectivity step and the geometry encoding step, resulting in a watermarked Draco compressed 3D object.

Watermarking and Encoding Phase

We propose integrating a watermarking step during the geometry encoding phase of the Draco 3D object compression method, which has no effect on the connectivity encoding phase that is performed separately. More precisely, the watermarking is performed between the vertex quantization step and the vertex prediction step. First, as illustrated in Fig. 6.25, the length of the message to embed is calculated in order to

determine the embedding rate per coordinate, which is limited by the total number of bits per quantized coordinate qp.

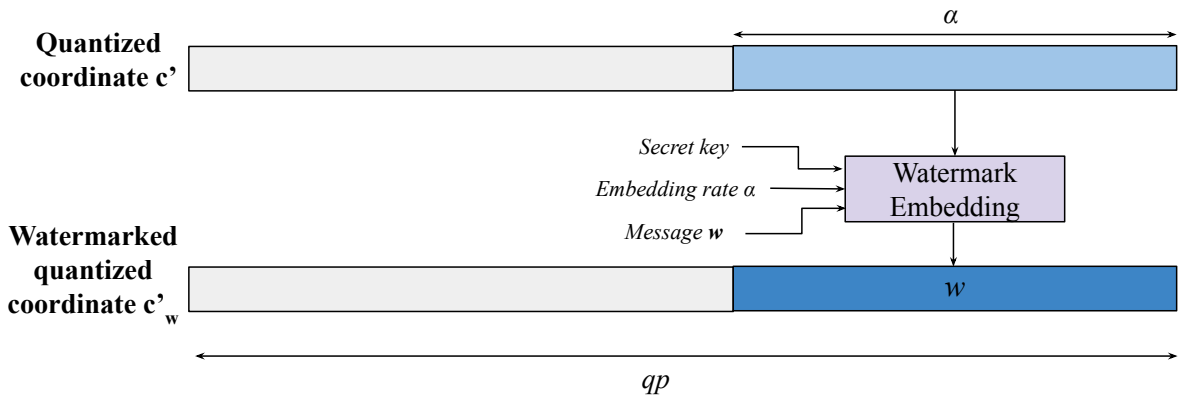


Figure 6.26: *The watermark embedding step for a single quantized coordinate c'.*

Fig. 6.26 presents the watermark embedding step for a single quantized coordinate c', which has a size of qp bits. Each quantized coordinate c' is watermarked with α bits according to a secret key which determines the order in which each message segment is embedded (*ie.* the message is permutated). We assume, as per convention, that the message has already been encrypted. The watermarking is performed by means of an LSB substitution:

$$c'_w = \left\lfloor \frac{c'}{2^\alpha} \right\rfloor \times 2^\alpha + w, \tag{6.7}$$

where c'_w is the watermarked coordinate, and w the message to be embedded in c'_w, composed of α bits.

After watermarking the quantized coordinates, they undergo the vertex prediction step. The prediction errors of the watermarked quantized vertices are then encoded with the rANS encoding method and a watermarked Draco compressed 3D object is obtained.

Watermark Extraction and Decoding

Fig. 6.27 presents an overview of the joint watermarking and Draco 3D object compression decoding phase. The watermark can be extracted in two places, either after the prediction and vertex reconstruction steps during the decoding phase, or from the reconstructed 3D object in the spatial domain after the entire Draco decoding.

First, the watermarked compressed 3D object undergoes a rANS decoding process in order to retrieve the vertex prediction errors. The watermarked quantized vertices are then reconstructed and the watermark can then be retrieved during the decoding step. Fig. 6.28 illustrates the watermark extraction process for a single reconstructed quantized coordinate. The message w is extracted by reading the α's LSB of each watermarked reconstructed quantized coordinate, where α corresponds to the embedding rate:

$$w = c'_w \bmod 2^\alpha, \tag{6.8}$$

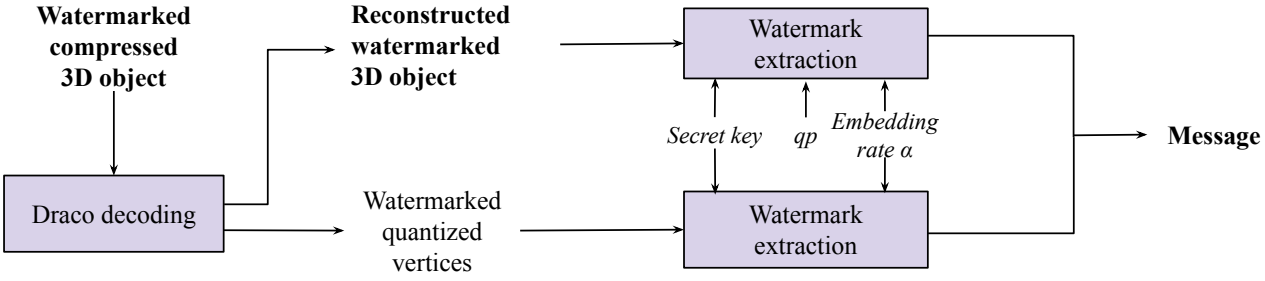


Figure 6.27: *An overview of the joint Draco 3D object compression and watermarking decoding phase.*

where c'_w is a watermarked quantized coordinate which has a size of qp bits.

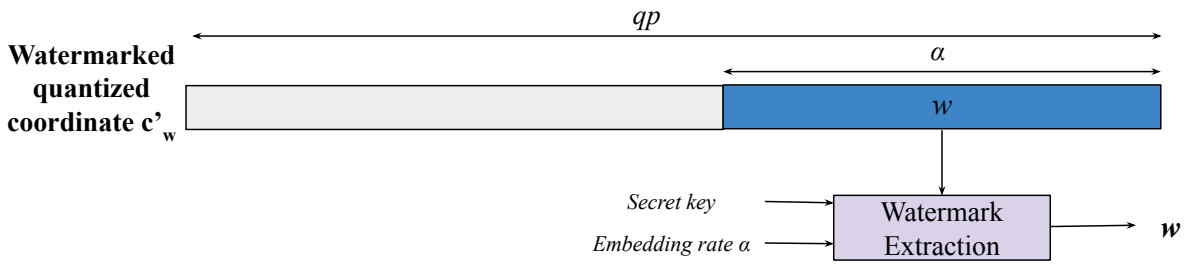


Figure 6.28: *Extraction of the watermark from a single watermarked quantized coordinate.*

After the watermark is retrieved, the floating point vertices are then reconstructed in order to finalize the Draco decoding process. A watermarked 3D object is then reconstructed, from which the watermark can also be retrieved. In this case the quantization parameter qp is necessary.

6.5.2 Experimental Results

In this section, we present experimental results of our joint watermarking and Draco 3D object compression method. First, in Section 6.5.2 we detail results of our method when applied to a single 3D object. Then, in Section 6.5.2, we present the results of our method when applied to the Standford dataset (34). Finally, we discuss the security of our proposed method in Section 6.5.2.

Full Example

We first apply our proposed method to the 3D object *Bunny*, from the Stanford dataset (34), as illustrated in Fig. 6.29. We note that α represents the embedding rate in bits per quantized coordinate. Consequently, with $\alpha = 0$, we obtain a 3D object decoded without any watermark embedding.

Fig. 6.29, shows the results of the joint watermarking and 3D Draco compressed Bunny object for various combinations of embedding rates α and quantization parameters qp. The embedding rate α must always be lower than qp, since qp represents the

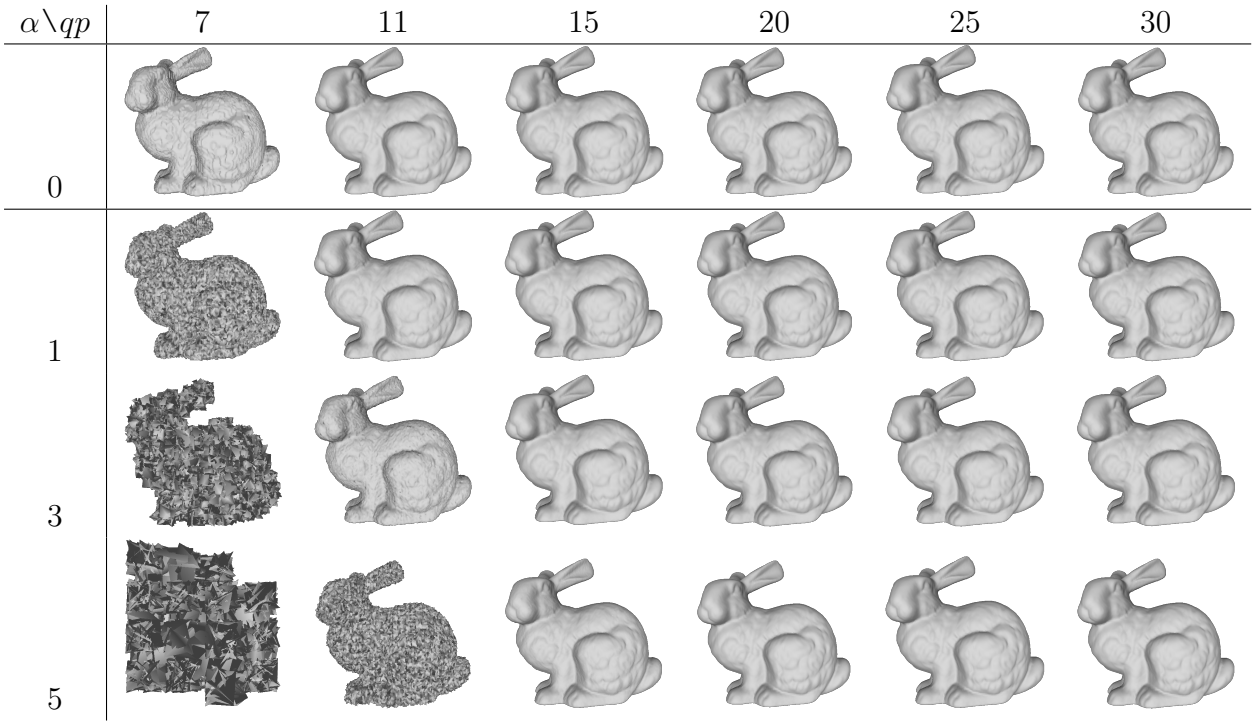
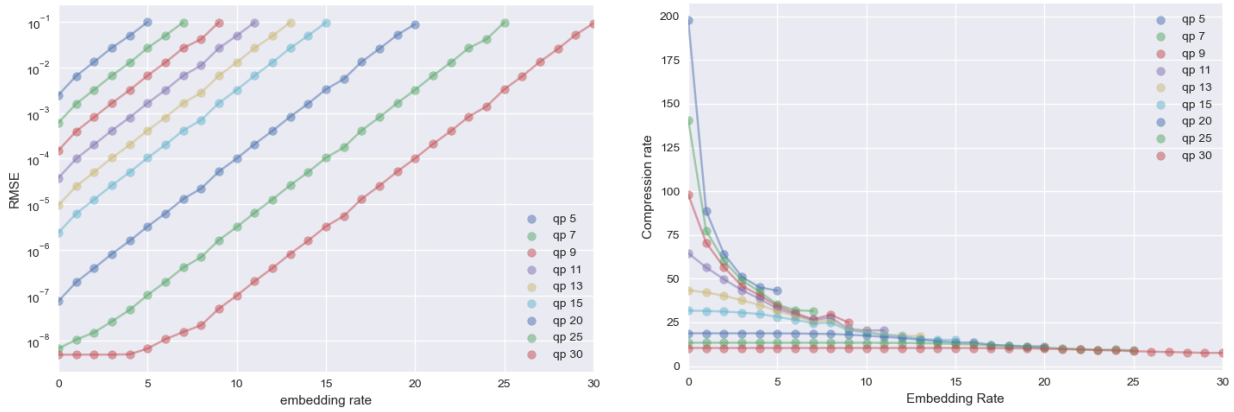


Figure 6.29: *Visual results of the proposed joint watermarking and 3D Draco compression method when applied on Bunny, for various payloads α, in bits per quantized coordinate and values of the Draco parameter qp.*

number of bits of a quantized coordinate. In other words, when qp increases, α can increase too, as more LSB bits become available for embedding. As the value of α approaches that of qp, the visual degradation increases. We can observe in Fig. 6.29 that there is no visual degradation when $(qp - \alpha) \geq 7$, as Draco 3D compression does not result in any visual degradation when at least 7 MSB are unchanged.

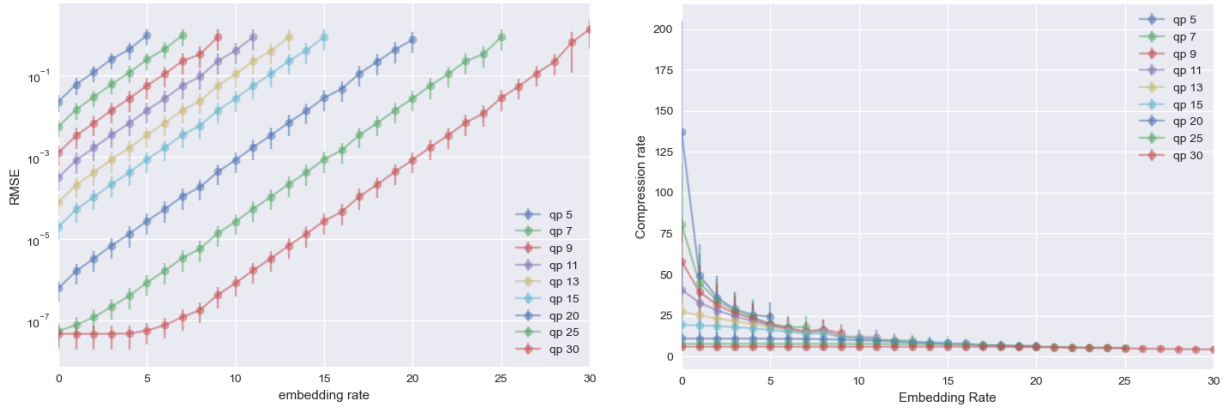


(a) RMSE (b) Compression rate

Figure 6.30: *RMSE and compression rate obtained for the 3D object Bunny watermarked and compressed with our proposed method as a function of the embedding rate α for various $qp \in [1, 30]$.*

Fig. 6.30(a) illustrates the RMSE obtained between the original 3D object *Bunny* and the watermarked and compressed one with our proposed method as a function of the embedding rate α for various $qp \in [1, 30]$. We can observe that the RMSE does not depend only on qp, but on the relationship between qp and α, *i.e.* the number of MSB

left unchanged. For example, when the number of MSB left unchanged is at least 7, *i.e.* $(qp - \alpha) \geq 7$, the RMSE is negligible, as it always has an order of 10^{-3}, regardless of the value of qp. This confirms the results illustrated in Fig. 6.29. Fig. 6.30(b) illustrates the compression rate obtained for the 3D object *Bunny* which has been watermarked and compressed with our proposed method, as a function of the embedding rate α for various $qp \in [1, 30]$. We note that $\alpha \in [0, qp]$, where $\alpha = 0$ corresponds to the original 3D compression method Draco, without watermarking. We observe that when qp increases, the compression rate decreases, this is due to the standard compression of Draco which is illustrated when $\alpha = 0$. This is to be expected, since qp defines the number of bits per coordinate of each vertex after quantization. We also observe that as qp increases, the compression rate reduction becomes less significant as α increases. From this detailed example of the 3D object *Bunny*, we can conclude that a trade-off between the quantization parameter qp and the embedding rate α is necessary in order to optimize both the compression rate and the RMSE.



(a) RMSE (b) Compression rate

Figure 6.31: *Mean RMSE and compression rate obtained for the entire Stanford dataset (34) watermarked and compressed with our proposed method as a function of the embedding rate α for various $qp \in [1, 30]$.*

Application on the Stanford dataset

When applying the proposed method on the entire Stanford dataset (34), composed of 11 3D objects, the mean RMSE and compression rate as a function of the embedding rate α, for various qp values, is provided in Fig. 6.31. Fig. 6.31(a) illustrates the mean RMSE with the standard deviation between the original 3D objects of the Stanford dataset and their corresponding reconstructed 3D objects which have been jointly watermarked and compressed using the proposed method. We note that the standard deviation is negligible, and therefore the RMSE does not greatly vary for different 3D objects. Like for *Bunny*, the RMSE does not depend only on the quantization parameter qp, but on the relationship between qp and α, *i.e.* the number of MSB's left unchanged. In Fig. 6.31(b), like for *Bunny*, we observe that when qp increases, the compression rate loss when α increases becomes less significant. Therefore, we conclude that the change in the compression rates between different 3D objects is minor, particularly when qp increases.

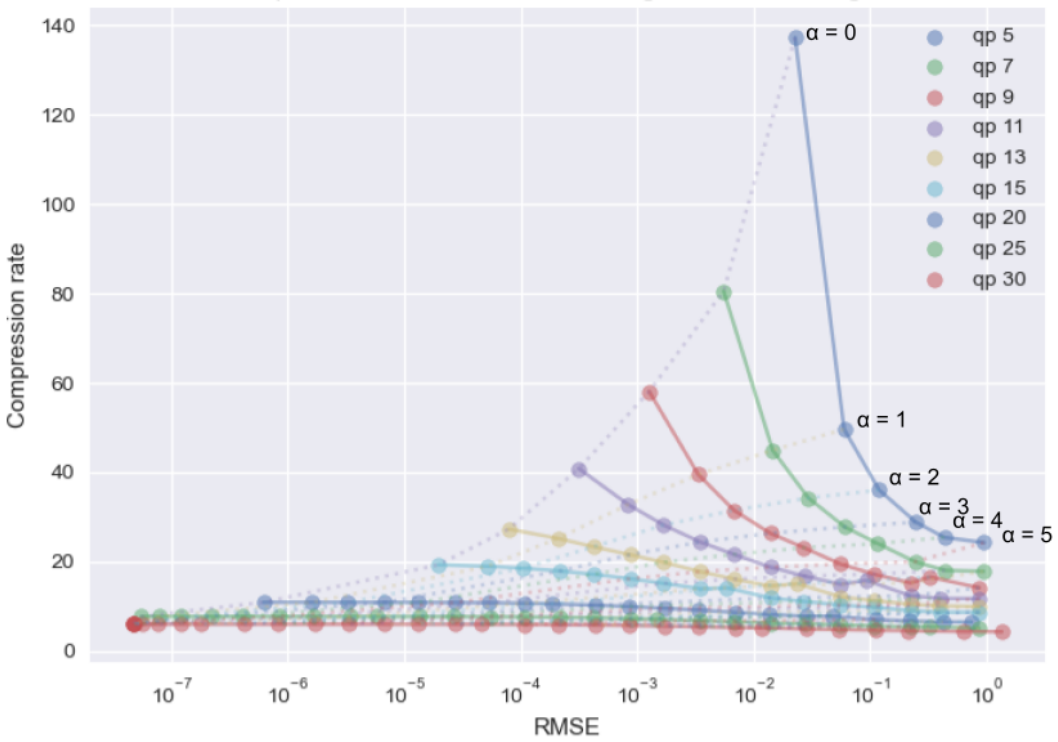


Figure 6.32: *Mean compression rate as a function of the RMSE for various qp and α, for the entire Stanford dataset (34).*

Like for the standard 3D compression method Draco, there is a trade-off between the compression rate and the distortion. Fig. 6.32 illustrates the mean compression rate as a function of the mean RMSE for various values of $qp \in [0, 30]$ for the Stanford dataset. We observe that changing the quantization parameter qp has a great impact on the compression rate and a minor impact on the RMSE. Therefore, the choice of qp should be made according to the desired embedding rate α. For example, if the desired embedding rate is $\alpha = 1$ bit per coordinate (3 bits per vertex), and the user wishes to minimize the distortion, then $qp = 15$ is recommended, as the average RMSE is 5.22×10^{-5}. If, for another example, the desired embedding rate α is 3 bits per coordinate (9 bits per vertex), then a good trade-off between the compression rate and the distortion is $qp = 11$. Indeed, with $qp = 11$, which is the default parameter for Draco given by Google, we achieve an average compression rate of 24.38 and an average RMSE of 3.47×10^{-3}. In conclusion, for these two examples ($qp = 15, \alpha = 1$ and $qp = 11, \alpha = 3$), we observe in Fig. 6.29 that the reconstructed 3D objects have a high quality without visual degradation.

Security Analysis and Discussion

In this section, we presented a joint watermarking and Draco compression method for 3D objects, which results in a watermarked Draco compressed 3D object as well as a watermarked Draco decoded 3D object, as the 3D object remains watermarked after the reconstruction. Joint encoding methods add an extra level of security, as an attacker needs to take into account the architecture of the joint encoding method. Indeed, if an attacker wishes to attack the 3D object with a method such as smoothing or zeroing,

they would need to design an attack specific to the architecture of the Draco decoding phase. We note that the watermarked Draco compressed 3D object cannot be attacked without destroying the decoded 3D object, as the Draco encoding process transforms the vertices so that they are irretrievable without performing the Draco decoding process.

While watermarking by LSB substitution is a classic method, its integration in Draco is challenging. With this approach, we have proven that it is possible to develop a joint watermarking and Draco compression method for 3D objects. We note that security for Draco compressed 3D objects is essential, as Draco is rapidly becoming the industry standard for 3D objects. We also note that very few methods for joint watermarking and compression for 3D objects exist in the state-of-the-art, as joint watermarking and compression is challenging to achieve, as both watermarking and compression modify the same domains of the 3D object representation that interfere with one another.

6.6 Conclusion

In this chapter, we described three contributions based on joint security and compression for 3D objects. In all three of these contributions, a security step is embedded in the geometry encoding phase of Google's 3D object compression method Draco.

The first contribution is a format compliant crypto-compression method for 3D objects based on Draco. We have integrated an AES encryption step using the CFB mode during the rANS encoding step, where we encrypted the vertex prediction errors. This encryption step is performed jointly with the 3D object compression. The proposed crypto-compression scheme has no size expansion and is completely reversible. We have shown with our experimental results that the proposed method is efficient in terms of time and compression rate. Based on an AES encryption with a 256 bit key, we have also shown with a security analysis that the proposed method is secure in terms of confidentiality. In future work, we want to change the form of the decoded crypto-compressed 3D object.

In the second contribution, we proposed a 3D object selective crypto-compression method based on Draco. We integrated a selective encryption step between the vertex quantization and the vertex prediction. From our experimental results, we can conclude that there is a trade-off between the compression rate and the RMSE of the 3D object. The choice of encryption parameter p depends on whether the user prefers to favor the compression rate or the security in terms of visual confidentiality. In future work, we aim to integrate a selective encryption step during the entropy encoding of the prediction errors, so as to conserve the compression rate of the standard Draco compression method.

Finally, in the third contribution, we proposed a joint watermarking and Draco 3D object compression method. The watermarking step is integrated during the geometry encoding process of the Draco 3D object compression method. As a result, a watermarked Draco compressed 3D object is produced. The watermark can then be

extracted either during the Draco decoding process, or after the 3D object is decoded and reconstructed, as the reconstruction still contains the watermark. Experimental results show that we can achieve a large watermark embedding rate, which can be used for embedding a significant amount of information including information related to the copyright. We have proposed an optimal value for the Draco and watermarking parameters in different scenarios. We have also tested the proposed method on a large dataset. In our security analysis and discussion, we showed that in order to perform an attack on the 3D object, an attack such as smoothing or zeroing has to be specifically designed according to the architecture of Draco, which is challenging. We have shown that it is possible to embed a watermarking step in the 3D object compression method Draco. In future work, we want to integrate the watermark, while trying to reduce the distortion as much as possible and keeping the original compression rate.

To the best of our knowledge, we are the first to propose a joint security and compression methods based on Draco. The first contribution was published in the international journal IEEE Access (159). The second was presented in the international conference IEEE IPTA 2022 (160). The third has been accepted and will be presented in the international conference IEEE ICIP 2023 (161).

Conclusion and Perspectives

In this final chapter, we summarize our work presented in this thesis and we present our perspectives. First, we summarize the contents of this manuscript where we have described three main categories of contributions relating to the security of 3D objects by means of data hiding or encryption. We then present possible perspectives for this work.

Conclusion

In this work, we have presented new methods for 3D object security based on data hiding and encryption. We first presented a state of the art of multimedia encryption. We detailed the fundamental notions and the evolution of modern cryptography. In particular, we presented cryptography applied to both images and 3D objects. We then presented a state of the art of multimedia data hiding. We presented the fundamentals of data hiding before describing the three main categories of data hiding, which are steganography, watermarking, and high capacity data hiding, as well as their applications. Finally, we presented a state of the art of joint multimedia encoding. We first described multimedia compression applied to images, video and 3D objects. We then detailed a state of the art of joint multimedia compression and security, as well as a state of the art of multimedia data hiding in the encrypted domain.

Reversible data hiding in the encrypted domain (RDH-ED) for 3D objects is a very new research topic, as the first method was proposed in 2018. However, these methods are essential as they allow third parties who are not authorized to access the original plaintext 3D object to embed data in the encrypted 3D object without the need for decryption. Thus, in our first contribution, we propose two variants of a RDH-ED method for 3D objects. In this method, we exploit the homomorphic properties of the Paillier cryptosystem which transform a multiplication in the encrypted domain into an addition in the plaintext domain. We first group the vertices of the 3D objects into blocks whose size depends on that of the key. Our method is therefore suitable for real-life applications, as it is designed for large key sizes. In the first variant, a two tier data hiding is performed using the Paillier cryptosystem's non deterministic property, known as the self blinding property, for the second tier data hiding. In the second variant, we use this same self-blinding property to flag each vertex block with the aim of creating a binary location map. This allows for a multi-message embedding. Despite our use of the Paillier cryptosystem, we avoid all size expansion. Our method is format compliant, avoids all auxiliary data and, to the best of our knowledge, is the only RDH-

ED method to produce a reconstructed 3D object which remains watermarked with a high capacity once decrypted.

Contrary to visual security level metrics for images, the notion of visual security level for 3D objects is a relatively unexplored domain. While metrics exist for 3D object quality assessment, to the best of our knowledge, before our contribution, no method for visual security assessment for 3D objects exists. Thus, in our second contribution, we detail two different contributions relating to the visual security levels of 3D objects. There are three visual security levels for 3D objects, which are the transparent level, the sufficient level and the confidential level. First, we developed a model to estimate the selective encryption parameters needed in order to achieve a desired visual security level, based on a dataset of selectively encrypted 3D objects called the SE3DO dataset. Based on this same dataset, we then detailed a subjective metric designed to measure the visual security level of an encrypted or, in particular, a selectively encrypted 3D object. We note that visual quality metrics cannot determine the visual security of a 3D object, as visual quality metrics simply label encrypted 3D objects has having a bad quality. Thus, they provide no information on the visual security. Finally, we described a new encryption method for 3D objects. This 3D object encryption method is the first to allow for a hierarchical decryption, which is based on a generated ring of hierarchical keys. We note that the visual security level of the decrypted 3D object depends on the hierarchy of the key. With this method, we have provided a more secure and environmentally friendly alternative for 3D object selective encryption, since only a single 3D object is confidentially encrypted, stored and shared.

Draco, the compression method for 3D objects developed by Google, is rapidly becoming an industry standard. In industry, 3D objects are often stored and shared online, and so Draco is often integrated into online tools, such as in the case of STRATEGIES, where 3D objects are inherently vulnerable to malicious attacks. However, to the best of our knowledge, no attention has been given to securing the 3D objects compressed with Draco. Thus, in our third and final contribution, we described three different contributions where we propose joint security and compression methods based on Draco. These three methods are a crypto-compression method, a selective crypto-compression method, and a joint watermarking and compression method for 3D objects based on Draco. To the best of our knowledge, we are the first to propose joint security methods for the Draco. In these joint security and compression methods, we integrate a security step in the Draco geometry encoding phase. In order to remain format compliant, the connectivity remains unaltered. In our proposed crypto-compression method, we integrate an AES encryption step during the Draco entropy encoding step. With the use of a secret key, the Draco compressed 3D object is decrypted during the geometry decoding phase. This decryption takes place jointly with the entropy decoding step. In the selective crypto-compression method and the joint watermarking and compression method, an exclusive-or encryption step and an LSB substitution step are respectively integrated between the Draco vertex quantization step and the Draco vertex prediction step. With the use of a secret key, the 3D object is decrypted or the message is extracted respectively, after the quantized vertices are reconstructed.

Perspectives

RDH-ED

Since the first RDH-ED method for 3D objects was proposed in 2018, the number of proposed methods has been growing exponentially. We present two possible directions for future contributions.

Exponent Optimization

Our first contribution could be improved by optimizing the quantity of bits embedded in a coordinate according to the coordinate's exponent value. We note that each coordinate is represented by a 32-bit floating point composed of a sign (1 bit), an exponent (8 bits) and a mantissa (23 bits), according to Eq. 1.22. The value of the exponent influences the impact of the bits of the mantissa. Therefore prioritizing the coordinates with small exponent values for data hiding would lead to less distortion in the reconstructed 3D object.

This can be achieved by ordering the coordinates within a vertex block according to the ascending order of the three exponents of each of the coordinates of a vertex. It can also be achieved by transforming the 3D object so that the exponent values of each coordinate are decreased.

MSB-based Prediction Scheme Using a Hamiltonian Path

Another perspective, on which we are currently working, is an RDH-ED method which is based on an MSB-based prediction scheme. In this method, we use a Hamiltonian path which defines a unique processing order for the vertices of a point cloud or a 3D object. To construct a Hamiltonian path, a starting vertex is given by the user. The vertex with the smallest euclidean distance to the previous vertex in the Hamiltonian path is chosen from the set of remaining vertices which are not already included in the Hamiltonian path. While Hamiltonian paths are traditionally used to define a synchronization order for a 3D object or a point cloud, in this method, we use a Hamiltonian path to predict the correct the MSB values of the vertex coordinates by exploiting the small distance between two connected vertices in the path.

In this method, the 3D object is encrypted with exclusive-or encryption and the message is then embedded by means of an MSB substitution. Before decrypting the vertices of the 3D object, the embedded message is read from the MSB of each coordinate. Then, once the vertices of the 3D object have been decrypted, the MSB of each coordinate needs to be corrected, as it was substituted in the encrypted domain. Each vertex therefore has $2^{1\times 3} = 8$ possible values. Thus, we use the Hamiltonian path to determine the correct value of the vertex. We achieve this by correcting the vertices in the order of the Hamiltonian path, and exploit the small distance between two connected vertices. Indeed, we use the previous vertex in the Hamiltonian path to predict

the correct coordinate values of the vertex to be reconstructed. The correct coordinate values are in general the closest value to the previous vertex in the Hamiltonian path. We mark any exceptions by sacrificing a part of the embedding space in order to embed this data. The number of exceptions can be reduced by translating the 3D object and therefore increasing the value of the exponent and consequently increasing the impact of the MSB.

Visual Security Metric

In this work, we described our proposed subjective visual security level metric for 3D objects which we have named the 3DVS score. This score is constructed with a linear regression based on the mean opinion scores (MOS) of a group of participants, and objective quality metrics. In future work, we want construct a new visual security metric based on the characteristics of the 3D object such as the smoothness or the density. We would also like to broaden our survey by creating an online version which will allow for many more participants and consequently, a greater quantity of MOS.

Another possible method of creating a subjective visual security level metric is by training a convolutional neural network (CNN). While many exist for the visual quality of 3D objects, none exist for the security of the 3D objects. The CNN can be trained using the SE3DO dataset which contains both the selectively encrypted 3D objects and the MOS scores.

Hierarchical Decryption

A limitation of our encryption method for a hierarchical decryption is that once a key has been used, a hierarchically superior key cannot be used on the same decrypted 3D object. For example, if the sufficient key is used, the confidentially encrypted 3D object will be hierarchically decrypted to a sufficient visual security level. A hierarchically superior key, for example the master key, cannot be used to decrypt the hierarchically decrypted sufficient level 3D object in order to retrieve the original 3D object.

In future work, we aim to develop a method where multiple hierarchical keys can be applied to a hierarchically decrypted 3D object. For example, if the sufficient key is used, a hierarchically decrypted sufficient level 3D object is retrieved. Someone with a hierarchically superior key, such as the master key, will then be able to fully decrypt this sufficient level 3D object and consequently retrieve the original clear level 3D object. This can be achieved by adding a marker which is conserved during the encryption and decryption process. This marker is therefore used to indicate if a sub block is already encrypted or decrypted, and therefore avoiding an encryption of an already encrypted sub block, or a decryption of an already decrypted sub block. We note that in this case, the same set of hierarchical keys must be used. This will also allow the master key to be reapplied to the hierarchically decrypted 3D object with a transparent or sufficient level, where the same transparent and sufficient keys will be able to hierarchically decrypt the 3D object once more.

Draco Security

In this work, to the best of our knowledge, we have presented the very first joint security methods for the Draco 3D object compression method. Consequently, we have shown that joint security and compression methods for Draco, in particular crypto-compression, selective crypto-compression and joint watermarking and compression methods, are possible. We believe that due to the popularity of Draco, the need for joint security and Draco compression methods will only increase. We therefore have many perspectives for joint security and Draco compression.

In future work, we aim to further analyze the behavior of the encrypted data in the case where the encrypted Draco file is decoded without a key. We note that when this encrypted decoded 3D object is visualized, it takes the form of a cube where most of the encrypted vertices are reconstructed on the border of this cube (Fig. 6.5c). This is due to the way the Draco decoder interprets the encrypted Draco compressed file, as the encrypted rANS encoded prediction error values reconstruct the vertices outside the limits of the Draco bounding box. These vertices are then brought back within the bounding box by Draco due to Draco's overflow management. We note that this is purely aesthetic as it corresponds to the way Draco interprets and displays the encrypted values. However, by analyzing this behavior, we aim to better understand the relationship between the encrypted decoded values and the vertex reconstruction process.

With these analyses, we wish to create a more efficient selective crypto-compression method and joint watermarking and compression method, where there is no loss in compression rate and therefore no size expansion.

In future work, we want to improve the robustness and imperceptibility of the joint watermarking and compression method. We aim to create a joint watermarking and compression method based on the statistics of 3D object by embedding a watermark in the histogram of the quantized vertices.

List of Publications

International Journals

- (159) **B. Jansen van Rensburg**, W. Puech and J. -P. Pedeboy, "The First Draco 3D Object Crypto-Compression Scheme," in IEEE Access, vol. 10, pp. 10566-10574, 2022.

- (155) S. Beugnon, **B. Jansen van Rensburg**, N. Amalou, W. Puech, J. -P. Pedeboy, "A 3D Visual Security (3DVS) score to measure the visual security level of selectively encrypted 3D objects," Signal Processing: Image Communication, vol. 108, 2022.

- (157) **B. Jansen van Rensburg**, W. Puech and J. -P. Pedeboy, "A Format Compliant Encryption Method for 3D Objects Allowing Hierarchical Decryption," in IEEE Transactions on Multimedia, 2022.

- (145) **B. Jansen van Rensburg**, P. Puteaux, W. Puech and J. -P. Pedeboy, "3D Object Watermarking from Data Hiding in the Homomorphic Encrypted Domain," ACM Transactions on Multimedia Computing Communications and Applications, vol. 19, no. 5s, 2023.

International Conferences

- (144) **B. Jansen van Rensburg**, P. Puteaux, W. Puech and J. -P. Pedeboy, "Homomorphic Two Tier Reversible Data Hiding In Encrypted 3D Objects," 2021 IEEE International Conference on Image Processing (ICIP), Anchorage, AK, USA, 2021, pp. 3068-3072.

- (160) **B. Jansen van Rensburg**, W. Puech and J. -P. Pedeboy, "Draco-Based Selective Crypto-Compression Method of 3D objects," 2022 Eleventh International Conference on Image Processing Theory, Tools and Applications (IPTA), Salzburg, Austria, 2022, pp. 1-6.

- (156) **B. Jansen van Rensburg**, W. Puech and J. -P. Pedeboy, "A Hierarchical Decryption Method for an Eco-Friendly Securing of 3D Objects," 2022 IEEE 24th International Workshop on Multimedia Signal Processing (MMSP), Shanghai, China, 2022, pp. 1-6.

- (161) **B. Jansen van Rensburg**, A. G. Bors, W. Puech and J. -P. Pedeboy, "Simultaneous Watermarking and Draco 3D Object Compression Method," 2023 IEEE International Conference on Image Processing (ICIP), Kuala Lumpur, Malaysia, 2023, **ACCEPTED**.
- E. Reinders, **B. Jansen van Rensburg**, P. Puteaux, W. Puech, "MSB-based Reversible Data-Hiding in Encrypted 3D Object using a Hamiltonian Path," 2023 IEEE 25th International Workshop on Multimedia Signal Processing (MMSP), Poitiers, France, 2023, **SUBMITTED**.

National Conferences

- (162) **B. Jansen van Rensburg**, A. G. Bors, W. Puech and J. -P. Pedeboy, "Méthode Jointe de Tatouage et Compression Draco pour les Objets 3D," COmpression et REprésentation des Signaux Audiovisuels 2023, 2023.
- (163) E. Reinders, **B. Jansen van Rensburg**, P. Puteaux, W. Puech, "Analyse d'images secrètes bruitées," COmpression et REprésentation des Signaux Audiovisuels 2023, 2023.

BIBLIOGRAPHY

[1] J. Katz and Y. Lindell, Introduction to modern cryptography. CRC press, 2020. 8

[2] A. Stevenson, Oxford dictionary of English. Oxford University Press, USA, 2010. 8

[3] A. Kerckhoffs, "La cryptographie militaire," J. Sci. Militaires, vol. 9, no. 4, pp. 5–38, 1883. 9, 25

[4] C. E. Shannon, "Communication theory of secrecy systems," The Bell System Technical Journal, vol. 28, no. 4, pp. 656–715, 1949. 9

[5] G. S. Vernam, "Cipher printing telegraph systems: For secret wire and radio telegraphic communications," Journal of the AIEE, vol. 45, no. 2, pp. 109–115, 1926. 10

[6] R. Davis, "The data encryption standard in perspective," IEEE Communications Society Magazine, vol. 16, no. 6, pp. 5–9, 1978. 11

[7] J. Daemen and V. Rijmen, The design of Rijndael, vol. 2. Springer, Berlin, Heidelberg, 2002. 11, 18

[8] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," Communications of the ACM, vol. 21, no. 2, pp. 120–126, 1978. 12, 13, 18

[9] C. Gentry, "Fully homomorphic encryption using ideal lattices," in Proceedings of the forty-first annual ACM symposium on Theory of computing, pp. 169–178, 2009. 13

[10] T. Elgamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," IEEE Transactions on Information Theory, vol. 31, no. 4, pp. 469–472, 1985. 13

[11] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in International conference on the theory and applications of cryptographic techniques, pp. 223–238, Springer, 1999. 13, 59

[12] F. Armknecht, S. Katzenbeisser, and A. Peter, "Group homomorphic encryption: characterizations, impossibility results, and applications," Designs, codes and cryptography, vol. 67, no. 2, pp. 209–232, 2013. 14

[13] Z.-H. Guan, F. Huang, and W. Guan, "Chaos-based image encryption algorithm," Physics letters A, vol. 346, no. 1-3, pp. 153–157, 2005. 15

[14] G. Chen and T. Ueta, "Yet another chaotic attractor," International Journal of Bifurcation and chaos, vol. 9, no. 07, pp. 1465–1466, 1999. 15

[15] K. Usman, H. Juzoji, I. Nakajima, S. Soegidjoko, M. Ramdhani, T. Hori, and S. Igi, "Medical image encryption based on pixel arrangement and random permutation for transmission security," in 2007 9th international conference on e-health networking, application and services, pp. 244–247, IEEE, 2007. 15

[16] H. M. Ghadirli, A. Nodehi, and R. Enayatifar, "An overview of encryption algorithms in color images," Signal Processing, vol. 164, pp. 163–185, 2019. 15

[17] X. Wang, L. Feng, and H. Zhao, "Fast image encryption algorithm based on parallel computing system," Information Sciences, vol. 486, pp. 340–358, 2019. 15, 18

[18] Y. Xian, X. Wang, X. Wang, Q. Li, and X. Yan, "Spiral-transform-based fractal sorting matrix for chaotic image encryption," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 69, no. 8, pp. 3320–3327, 2022. 15

[19] U. Sara, M. Akter, and M. S. Uddin, "Image quality assessment through fsim, ssim, mse and psnr—a comparative study," Journal of Computer and Communications, vol. 7, no. 3, pp. 8–18, 2019. 16

[20] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," IEEE transactions on image processing, vol. 13, no. 4, pp. 600–612, 2004. 16

[21] N. Murray, L. Marchesotti, and F. Perronnin, "Ava: A large-scale database for aesthetic visual analysis," in 2012 IEEE conference on computer vision and pattern recognition, pp. 2408–2415, IEEE, 2012. 16

[22] X. Lu, Z. Lin, H. Jin, J. Yang, and J. Z. Wang, "Rating image aesthetics using deep learning," IEEE Transactions on Multimedia, vol. 17, no. 11, pp. 2021–2034, 2015. 16

[23] H. Talebi and P. Milanfar, "Nima: Neural image assessment," IEEE Transactions on Image Processing, vol. 27, no. 8, pp. 3998–4011, 2018. 16

[24] G. Zhai and X. Min, "Perceptual image quality assessment: a survey," Science China Information Sciences, vol. 63, pp. 1–52, 2020. 16

[25] Y. Yao, Z. Xu, and J. Sun, "Visual Security Assessment for Cipher-Images based on Neighborhood Similarity," Informatica (Slovenia), vol. 33, pp. 69–76, 2009. 16

[26] S. Jenisch and A. Uhl, "Visual Security Evaluation Based on SIFT Object Recognition," in Artificial Intelligence Applications and Innovations, pp. 624–633, Springer Berlin Heidelberg, 2014. 16

[27] D. Lowe, "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision, vol. 60, p. 91, 2004. 16

[28] T. Xiang, S. Guo, and X. Li, "Perceptual Visual Security Index Based on Edge and Texture Similarities," IEEE Transactions on Information Forensics and Security, vol. 11, no. 5, pp. 951–963, 2016. 16

[29] T. Xiang, Y. Yang, H. Liu, and S. Guo, "Visual Security Evaluation of Perceptually Encrypted Images Based on Image Importance," IEEE Transactions on Circuits and Systems for Video Technology, vol. 30, no. 11, pp. 4129–4142, 2020. 16

[30] A. S. Abraham, L. R. Nair, and M. S. Deepa, "A novel method for evaluation of visual security of images," in 2017 International Conference on Networks Advances in Computational Technologies (NetACT), pp. 387–391, 2017. 16

[31] S. Guo, T. Xiang, X. Li, and Y. Yang, "PEID: A Perceptually Encrypted Image Database for Visual Security Evaluation," IEEE Transactions on Information Forensics and Security, vol. 15, pp. 1151–1163, 2020. 16

[32] Y. Yang, T. Xiang, H. Liu, and X. Liao, "Convolutional neural network for visual security evaluation," IEEE Transactions on Circuits and Systems for Video Technology, vol. 31, no. 8, pp. 3293–3307, 2021. 16

[33] S. Verykokou and C. Ioannidis, "An overview on image-based and scanner-based 3d modeling technologies," Sensors, vol. 23, no. 2, p. 596, 2023. 17

[34] M. Levoy, J. Gerth, B. Curless, and K. Pull, "The Stanford 3D scanning repository," URL http://graphics.stanford.edu/data/3Dscanrep/, vol. 5, no. 10, 2005. 17, 18, 24, 53, 118, 120, 125, 136, 137, 149, 156, 158, 159

[35] A. Jolfaei, X.-W. Wu, and V. Muthukkumarasamy, "A 3D Object Encryption Scheme Which Maintains Dimensional and Spatial Stability," IEEE Transactions on Information Forensics and Security, vol. 10, no. 2, pp. 409–422, 2015. 18

[36] S. Gao, R. Wu, X. Wang, J. Wang, Q. Li, C. Wang, and X. Tang, "A 3D model encryption scheme based on a cascaded chaotic system," Signal Processing, vol. 202, p. 108745, 2023. 18, 19

[37] N. Aspert, D. Santa-Cruz, and T. Ebrahimi, "Mesh: Measuring errors between surfaces using the hausdorff distance," in Proceedings of the 2002 IEEE International Conference on Multimedia and Expo (ICME), vol. 1, pp. 705–708, IEEE, 2002. 18

[38] P. Cignoni, C. Rocchini, and R. Scopigno, "Metro: Measuring error on simplified surfaces," in Computer Graphics Forum, vol. 17, pp. 167–174, Wiley Online Library, 1998. 18

[39] M.-W. Chao, C.-h. Lin, C.-W. Yu, and T.-Y. Lee, "A high capacity 3D steganography algorithm," IEEE Transactions on Visualization and Computer Graphics, vol. 15, no. 2, pp. 274–284, 2009. 18, 20

[40] G. Lavoué, "A Multiscale Metric for 3D Mesh Visual quality assessment," Computer Graphics Forum, vol. 30, no. 5, pp. 1427–1437, 2011. 18, 20

[41] L. Vása and J. Rus, "Dihedral Angle Mesh Error: a fast perception correlated distortion measure for fixed connectivity triangle meshes," Computer Graphics Forum, vol. 31, no. 5, pp. 1715–1724, 2012. 18, 20

[42] G. Lavoué, E. Gelasca, F. Dupont, A. Baskurt, and T. Ebrahimi, "Perceptually driven 3D distance metrics with application to watermarking," Proceedings of SPIE - The International Society for Optical Engineering, vol. 6312, 2006. 20

[43] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600–612, 2004. 20

[44] M. Cho, S. Kim, M. Sung, and G. On, "3D Fingerprinting and Encryption Principle for Collaboration," in International Conference on Automated Production of Cross Media Content for Multi-Channel Distribution (AXMEDIS), pp. 121–127, IEEE, 2006. 21

[45] M. Gschwandtner and A. Uhl, "Protected Progressive Meshes," in Advances in Visual Computing, pp. 35–48, Springer, 2009. 21, 126

[46] M. Éluard, Y. Maetz, and G. J. Doërr, "Impact of geometry-preserving encryption on rendering time," in 2014 IEEE International Conference on Image Processing (ICIP), pp. 4787–4791, IEEE, 2014. 21

[47] S. Beugnon, W. Puech, and J. Pedeboy, "From Visual Confidentiality To Transparent Format-Compliant Selective Encryption Of 3D Objects," in 2018 IEEE International Conference on Multimedia and Expo Workshops (ICME Workshops), pp. 1–6, IEEE Computer Society, 2018. 21, 22, 23, 24, 73, 74, 92, 94, 95, 108, 111, 146

[48] A. Pommer and A. Uhl, "Application Scenarios for Selective Encryption of Visual Data," in Multimedia and Security Workshop (ACM Multimedia), pp. 71–74, 2002. 22, 95

[49] I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker, Digital watermarking and steganography. Morgan kaufmann, 2007. 29

[50] G. J. Simmons, "The prisoners' problem and the subliminal channel," in Advances in Cryptology: Proceedings of Crypto 83, pp. 51–67, Springer, 1984. 33

[51] L. M. Marvel, C. G. Boncelet, and C. T. Retter, "Spread spectrum image steganography," IEEE Transactions on image processing, vol. 8, no. 8, pp. 1075–1083, 1999. 34

[52] V. Potdar, S. Han, and E. Chang, "A survey of digital image watermarking techniques," in INDIN '05. 2005 3rd IEEE International Conference on Industrial Informatics, 2005., pp. 709–716, 2005. 34

[53] N. Subramanian, O. Elharrouss, S. Al-Maadeed, and A. Bouridane, "Image steganography: A review of the recent advances," IEEE Access, vol. 9, pp. 23409–23423, 2021. 34

[54] P. Pan, Z. Wu, C. Yang, and B. Zhao, "Double-matrix decomposition image steganography scheme based on wavelet transform with multi-region coverage," Entropy, vol. 24, no. 2, p. 246, 2022. 34

[55] F. Cayre and B. Macq, "Data hiding on 3-D triangle meshes," IEEE Transactions on Signal Processing, vol. 51, no. 4, pp. 939–949, 2003. 35

[56] P. Amat, W. Puech, S. Druon, and J.-P. Pedeboy, "Lossless 3D steganography based on MST and connectivity modification," Signal Processing: Image Communication, vol. 25, no. 6, pp. 400–412, 2010. 35

[57] S. Farrag and W. Alexan, "A high capacity geometrical domain based 3D image steganography scheme," in 2019 International Conference on Advanced Communication Technologies and Networking (CommNet), pp. 1–7, IEEE, 2019. 35

[58] C. Eric, "Hiding in plain sight, stegnography and the art of covert communication," Wiley, Indianapolis, Indiana, ISBN, vol. 10, p. 0471444499, 2003. 36

[59] T. Pevnỳ, P. Bas, and J. Fridrich, "Steganalysis by subtractive pixel adjacency matrix," in Proceedings of the 11th ACM workshop on Multimedia and security, pp. 75–84, 2009. 36

[60] W. You, H. Zhang, and X. Zhao, "A siamese cnn for image steganalysis," IEEE Transactions on Information Forensics and Security, vol. 16, pp. 291–306, 2020. 36

[61] W. M. Eid, S. S. Alotaibi, H. M. Alqahtani, and S. Q. Saleh, "Digital image steganalysis: Current methodologies and future challenges," IEEE Access, vol. 10, pp. 92321–92336, 2022. 36

[62] Y. Yang and I. Ivrissimtzis, "Mesh discriminative features for 3D steganalysis," ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), vol. 10, no. 3, pp. 1–13, 2014. 36

[63] Z. Li and A. G. Bors, "Steganalysis of 3D objects using statistics of local feature sets," Information Sciences, vol. 415, pp. 85–99, 2017. 36

[64] Z. Li and A. G. Bors, "Steganalysis of meshes based on 3D wavelet multiresolution analysis," Information sciences, vol. 522, pp. 164–179, 2020. 36

[65] H. Zhou, K. Chen, W. Zhang, C. Qin, and N. Yu, "Feature-preserving tensor voting model for mesh steganalysis," IEEE transactions on visualization and computer graphics, vol. 27, no. 1, pp. 57–67, 2019. 36

[66] R. G. Van Schyndel, A. Z. Tirkel, and C. F. Osborne, "A digital watermark," in Proceedings of 1st international conference on image processing, vol. 2, pp. 86–90, IEEE, 1994. 37

[67] B. Gunsel, U. Uludag, and A. M. Tekalp, "Robust watermarking of fingerprint images," Pattern Recognition, vol. 35, no. 12, pp. 2739–2747, 2002. 37

[68] A. Poljicak, L. Mandic, and D. Agic, "Discrete fourier transform–based watermarking method with an optimal implementation radius," Journal of Electronic Imaging, vol. 20, no. 3, pp. 033008–033008, 2011. 37

[69] W. H. Alshoura, Z. Zainol, J. S. Teh, M. Alawida, and A. Alabdulatif, "Hybrid SVD-Based Image Watermarking Schemes: A Review," IEEE Access, vol. 9, pp. 32931–32968, 2021. 37

[70] M. Begum, J. Ferdush, and M. S. Uddin, "A hybrid robust watermarking system based on discrete cosine transform, discrete wavelet transform, and singular value decomposition," Journal of King Saud University-Computer and Information Sciences, vol. 34, no. 8, pp. 5856–5867, 2022. 37

[71] O. Benedens, "Geometry-based watermarking of 3D models," IEEE Computer Graphics and Applications, vol. 19, no. 1, pp. 46–55, 1999. 37

[72] S. Zafeiriou, A. Tefas, and I. Pitas, "Blind robust watermarking schemes for copyright protection of 3D mesh objects," IEEE Transactions on Visualization and Computer Graphics, vol. 11, no. 5, pp. 596–607, 2005. 37

[73] J.-W. Cho, R. Prost, and H.-Y.Jung, "An Oblivious Watermarking for 3-D Polygonal Meshes Using Distribution of Vertex Norms," IEEE Transactions on Signal Processing, vol. 55, pp. 142 – 155, 02 2007. 37

[74] K. Wang, G. Lavoué, F. Denis, A. Baskurt, and X. He, "A Benchmark for 3D Mesh Watermarking," in 2010 Shape Modeling International Conference, pp. 231–235, 2010. 37

[75] A. G. Bors and M. Luo, "Optimized 3D Watermarking for Minimal Surface Distortion," IEEE Transactions on Image Processing, vol. 22, no. 5, pp. 1822–1835, 2013. 37

[76] I. Yoo, H. Chang, X. Luo, O. Stava, C. Liu, P. Milanfar, and F. Yang, "Deep 3D-to-2D Watermarking: Embedding Messages in 3D Meshes and Extracting Them from 2D Renderings," in 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 10021–10030, 2022. 37, 38

[77] J.-U. Hou, D.-G. Kim, and H.-K. Lee, "Blind 3D mesh watermarking for 3D printed model by analyzing layering artifact," IEEE Transactions on Information Forensics and Security, vol. 12, no. 11, pp. 2712–2725, 2017. 38

[78] A. Delmotte, K. Tanaka, H. Kubo, T. Funatomi, and Y. Mukaigawa, "Blind 3d-printing watermarking using moment alignment and surface norm distribution," IEEE Transactions on Multimedia, vol. 23, pp. 3467–3482, 2021. 38

[79] H. Windolf, R. Chamberlain, A. Delmotte, and J. Quodbach, "Blind-Watermarking—Proof-of-Concept of a Novel Approach to Ensure Batch Traceability for 3D Printed Tablets," Pharmaceutics, vol. 14, no. 2, p. 432, 2022. 38

[80] D. Boneh and J. Shaw, "Collusion-secure fingerprinting for digital data," IEEE Transactions on Information Theory, vol. 44, no. 5, pp. 1897–1905, 1998. 39

[81] G. Tardos, "Optimal probabilistic fingerprint codes," Journal of the ACM (JACM), vol. 55, no. 2, pp. 1–24, 2008. 39

[82] F. Xie, T. Furon, and C. Fontaine, "On-off keying modulation and tardos fingerprinting," in Proceedings of the 10th ACM workshop on Multimedia and security, pp. 101–106, 2008. 39

[83] M. Desoubeaux, G. Le Guelvouit, and W. Puech, "Probabilistic fingerprinting codes used to detect traitor zero-bit watermark," in Media Watermarking, Security, and Forensics III, vol. 7880, pp. 334–342, SPIE, 2011. 39

[84] J. Baaouni, H. Choura, F. Chaabane, T. Frikha, and M. Baklouti, "Design of Multiprocessor Architecture for Watermarking and Tracing Images Using QR Code," in Intelligent Decision Technologies: Proceedings of the 14th KES-IDT 2022 Conference, pp. 109–122, Springer, 2022. 39

[85] Z. Li, A. S. Rathore, C. Song, S. Wei, Y. Wang, and W. Xu, "PrinTracker: Fingerprinting 3D printers using commodity scanners," in Proceedings of the 2018 ACM sigsac conference on computer and communications security, pp. 1306–1323, 2018. 39

[86] Y. Gao, W. Wang, Y. Jin, C. Zhou, W. Xu, and Z. Jin, "ThermoTag: A Hidden ID of 3D Printers for Fingerprinting and Watermarking," IEEE Transactions on Information Forensics and Security, vol. 16, pp. 2805–2820, 2021. 39

[87] E. T. Lin and E. J. Delp, "A review of fragile image watermarks," in Proceedings of the Multimedia and Security Workshop at ACM Multimedia, vol. 99, pp. 35–39, 1999. 39

[88] S.-H. Liu, H.-X. Yao, W. Gao, and Y.-L. Liu, "An image fragile watermark scheme based on chaotic image pattern and pixel-pairs," Applied Mathematics and Computation, vol. 185, no. 2, pp. 869–882, 2007. 40

[89] A. Shehab, M. Elhoseny, K. Muhammad, A. K. Sangaiah, P. Yang, H. Huang, and G. Hou, "Secure and Robust Fragile Watermarking Scheme for Medical Images," IEEE Access, vol. 6, pp. 10269–10278, 2018. 40

[90] P. Lefèvre, P. Carré, C. Fontaine, P. Gaborit, and J. Huang, "Efficient image tampering localization using semi-fragile watermarking and error control codes," Signal Processing, vol. 190, p. 108342, 2022. 40

[91] H.-T. Wu and Y.-M. Cheung, "A fragile watermarking scheme for 3D meshes," in Proceedings of the 7th workshop on Multimedia and security, pp. 117–124, 2005. 40

[92] K. Wang, G. Lavoue, F. Denis, and A. Baskurt, "A comprehensive survey on three-dimensional mesh watermarking," IEEE Transactions on Multimedia, vol. 10, no. 8, pp. 1513–1527, 2008. 40

[93] F. Peng, B. Long, and M. Long, "A general region nesting-based semi-fragile reversible watermarking for authenticating 3D mesh models," IEEE transactions on circuits and systems for video technology, vol. 31, no. 11, pp. 4538–4553, 2021. 40

[94] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," IEEE Transactions on Circuits and Systems for Video Technology, vol. 16, no. 3, pp. 354–362, 2006. 41

[95] P. Nagarju, R. Naskar, and R. S. Chakraborty, "Improved histogram bin shifting based reversible watermarking," 2013 International Conference on Intelligent Systems and Signal Processing (ISSP), pp. 62–65, 2013. 41

[96] P. Tsai, Y.-C. Hu, and H.-L. Yeh, "Reversible image hiding scheme using predictive coding and histogram shifting," Signal Processing, vol. 89, no. 6, pp. 1129–1143, 2009. 41

[97] G. Xuan, Q. Yao, C. Yang, J. Gao, P. Chai, Y. Q. Shi, and Z. Ni, "Lossless data hiding using histogram shifting method based on integer wavelets," in Digital Watermarking: 5th International Workshop, IWDW 2006, Jeju Island, Korea, November 8-10, 2006. Proceedings 5, pp. 323–332, Springer, 2006. 41

[98] V. Kelkar and H. Nemade, "Reversible watermarking in medical images using histogram shifting method with improved security and embedding capacity," in 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), pp. 971–975, IEEE, 2016. 41

[99] V. Itier and W. Puech, "High capacity data hiding for 3D point clouds based on static arithmetic coding," Multimedia Tools and Applications, vol. 76, pp. 26421–26445, 2017. 41, 42

[100] R. Jiang, W. Zhang, D. Hou, H. Wang, and N. Yu, "Reversible data hiding for 3D mesh models with three-dimensional prediction-error histogram modification," Multimedia Tools and Applications, vol. 77, pp. 5263–5280, 2018. 42, 68

[101] Q. Zhang, X. Song, T. Wen, and C. Fu, "Reversibility improved data hiding in 3D mesh models using prediction-error expansion and sorting," Measurement, vol. 135, pp. 738–746, 2019. 42

[102] G. K. Wallace, "The JPEG still picture compression standard," Communications of the ACM, vol. 34, no. 4, pp. 30–44, 1991. 47, 49

[103] C. Christopoulos, A. Skodras, and T. Ebrahimi, "The JPEG2000 still image coding system: an overview," IEEE transactions on consumer electronics, vol. 46, no. 4, pp. 1103–1127, 2000. 50

[104] Y. Dong, X. Yu, and P. Li, "3D model progressive compression algorithm using attributes," in 4th International Conference on Smart and Sustainable City (ICSSC 2017), pp. 1–5, IET, 2017. 51

[105] A. Doumanoglou, P. Drakoulis, N. Zioulis, D. Zarpalas, and P. Daras, "Benchmarking Open-Source Static 3D Mesh Codecs for Immersive Media Interactive Live Streaming," IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 9, no. 1, pp. 190–203, 2019. 51

[106] H. Liu, H. Yuan, Q. Liu, J. Hou, and J. Liu, "A Comprehensive Study and Comparison of Core Technologies for MPEG 3-D Point Cloud Compression," IEEE Transactions on Broadcasting, vol. 66, no. 3, pp. 701–717, 2020. 51

[107] Z. Que, G. Lu, and D. Xu, "Voxelcontext-net: An octree based framework for point cloud compression," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6042–6051, 2021. 51

[108] Google, "Draco 3D graphics compression," 2014. 51, 132

[109] J. Rossignac, "3D compression made simple: Edgebreaker with ZipandWrap on a corner-table," in Proceedings International Conference on Shape Modeling and Applications, pp. 278–283, 2001. 51, 133

[110] J. Duda, K. Tahboub, N. J. Gadgil, and E. J. Delp, "The use of asymmetric numeral systems as an accurate replacement for Huffman coding," in Picture Coding Symposium (PCS), pp. 65–69, 2015. 55, 132

[111] M. Van Droogenbroeck and R. Benedett, "Techniques for a selective encryption of uncompressed and compressed images," (Ghent, Belgium), pp. 90–97, Springer, 2002. 56

[112] W. Puech and J. M. Rodrigues, "Crypto-compression of medical images by selective encryption of DCT," in 13th European signal processing conference, pp. 1–4, IEEE, 2005. 57

[113] F. Gmira, S. Hraoui, A. Saaidi, A. J. Oulidi, and K. Satori, "Securing the architecture of the JPEG compression by an dynamic encryption," in Intelligent Systems and Computer Vision (ISCV), (Fez, Morocco), pp. 1–6, IEEE, 2015. 57

[114] J. Hajji, M. A. Ben Farah, M. Samet, and A. Kachouri, "Crypto-compression of images based on chaos," in 6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), pp. 344–350, 2012. 57

[115] M. Dridi, B. Bouallegue, and A. Mtibaa, "Crypto-compression of medical image based on DCT and chaotic system," in Global Summit on Computer Information Technology (GSCIT), pp. 1–6, 2014. 57

[116] F. Dufaux and T. Ebrahimi, "Scrambling for privacy protection in video surveillance systems," IEEE Transactions on Circuits and Systems for Video Technology, vol. 18, no. 8, pp. 1168–1174, 2008. 57

[117] Z. Shahid, M. Chaumont, and W. Puech, "Fast protection of H.264/AVC by selective encryption of CAVLC and CABAC for I and P frames," IEEE Transactions on Circuits and Systems for Video Technology, vol. 21, no. 5, pp. 565–576, 2011. 57

[118] W. Hamidouche, M. Farajallah, N. Sidaty, S. El Assad, and O. Deforges, "Real-time selective video encryption based on the chaos system in scalable HEVC extension," Signal Processing: Image Communication, vol. 58, pp. 73–86, 2017. 57

[119] G. Xuan, Y. Q. Shi, Z. Ni, P. Chai, X. Cui, and X. Tong, "Reversible data hiding for JPEG images based on histogram pairs," in Image Analysis and Recognition: 4th International Conference, ICIAR 2007, Montreal, Canada, August 22-24, 2007. Proceedings 4, pp. 715–727, Springer, 2007. 57

[120] F. Huang, X. Qu, H. J. Kim, and J. Huang, "Reversible data hiding in JPEG images," IEEE Transactions on Circuits and Systems for Video Technology, vol. 26, no. 9, pp. 1610–1621, 2015. 57

[121] S. Weng, Y. Zhou, T. Zhang, M. Xiao, and Y. Zhao, "Reversible data hiding for JPEG images with adaptive multiple two-dimensional histogram and mapping generation," IEEE Transactions on Multimedia, 2023. 57

[122] M. D. Swanson, B. Zhu, and A. H. Tewfik, "Data hiding for video-in-video," in Proceedings of International Conference on Image Processing, vol. 2, pp. 676–679, IEEE, 1997. 57

[123] M. Z. Konyar and S. Solak, "Efficient data hiding method for videos based on adaptive inverted LSB332 and secure frame selection with enhanced Vigenere cipher," Journal of Information Security and Applications, vol. 63, p. 103037, 2021. 57

[124] E. Abdallah, A. Hamza, and P. Bhattacharya, "Watermarking 3D models using spectral mesh compression," Signal, Image and Video Processing, vol. 3, pp. 375–389, 10 2009. 57

[125] H. Lee, Ç. Dikici, G. Lavoué, and F. Dupont, "Joint reversible watermarking and progressive compression of 3D meshes," The Visual Computer, vol. 27, pp. 781–792, 2011. 57

[126] P. Puteaux, S. Ong, K. Wong, and W. Puech, "A survey of reversible data hiding in encrypted images–the first 12 years," Journal of Visual Communication and Image Representation, vol. 77, p. 103085, 2021. 59

[127] W. Puech, M. Chaumont, and O. Strauss, "A reversible data hiding method for encrypted images," Proceedings of SPIE - The International Society for Optical Engineering, vol. 6819, 2008. 59

[128] X. Zhang, "Reversible data hiding in encrypted image," IEEE signal processing letters, vol. 18, no. 4, pp. 255–258, 2011. 59

[129] P. Puteaux and W. Puech, "An efficient MSB prediction-based method for high-capacity reversible data hiding in encrypted images," IEEE Transactions on Information Forensics and Security, vol. 13, no. 7, pp. 1670–1681, 2018. 59, 60

[130] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," Journal of the ACM (JACM), vol. 56, no. 6, pp. 1–40, 2009. 59

[131] Y.-C. Chen, C.-W. Shiu, and G. Horng, "Encrypted signal-based reversible data hiding with public key cryptosystem," Journal of Visual Communication and Image Representation, vol. 25, no. 5, pp. 1164–1170, 2014. 59

[132] S. Xiang and X. Luo, "Reversible data hiding in homomorphic encrypted domain by mirroring ciphertext group," IEEE Transactions on Circuits and Systems for Video Technology, vol. 28, no. 11, pp. 3099–3110, 2018. 59

[133] S. Zheng, Y. Wang, and D. Hu, "Lossless data hiding based on homomorphic cryptosystem," IEEE Transactions on Dependable and Secure Computing, 2019. 60

[134] P. Puteaux, M. Vialle, and W. Puech, "Homomorphic encryption-based LSB substitution for high capacity data hiding in the encrypted domain," IEEE Access, vol. 8, pp. 108655–108663, 2020. 60

[135] R. Jiang, H. Zhou, W. Zhang, and N. Yu, "Reversible data hiding in encrypted three-dimensional mesh models," IEEE Transactions on Multimedia, vol. 20, no. 1, pp. 55–67, 2018. 61, 81, 82, 83, 84

[136] N. Xu, J. Tang, B. Luo, and Z. Yin, "Separable Reversible Data Hiding Based on Integer Mapping and MSB Prediction for Encrypted 3D Mesh Models," Cognitive Computation, vol. 14, pp. 1172–1181, 2022. 61, 81, 82, 83, 84

[137] Z. Yin, N. Xu, and F. Wang, "Separable reversible data hiding based on integer mapping and multi-MSB prediction for encrypted 3D mesh models," arXiv, pp. arXiv–1908, 2019. 61, 68, 81, 82, 83, 84

[138] Y.-Y. Tsai and H.-L. Liu, "Integrating coordinate transformation and random sampling into high-capacity reversible data hiding in encrypted polygonal models," IEEE Transactions on Dependable and Secure Computing, 2022. 61

[139] W.-L. Lyu, L. Cheng, and Z. Yin, "High-capacity reversible data hiding in encrypted 3D mesh models based on multi-MSB prediction," Signal Processing, vol. 201, p. 108686, 2022. 61, 81, 82, 83, 84

[140] Y. Tang, L. Cheng, W. Lyu, and Z. Yin, "High Capacity Reversible Data Hiding for Encrypted 3D Mesh Models Based on Topology," in Digital Forensics and Watermarking: 21st International Workshop, IWDW 2022, Guilin, China, November 18-19, 2022, Revised Selected Papers, pp. 205–218, Springer, 2023. 61, 62

[141] M. Shah, W. Zhang, H. Hu, H. Zhou, and T. Mahmood, "Homomorphic encryption-based reversible data hiding for 3D mesh models," Arabian Journal for Science and Engineering, vol. 43, no. 12, pp. 8145–8157, 2018. 62, 68, 75, 81, 82, 83, 84

[142] Y.-Y. Tsai, "Separable reversible data hiding for encrypted three-dimensional models based on spatial subdivision and space encoding," IEEE transactions on multimedia, vol. 23, pp. 2286–2296, 2020. 62

[143] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser, "The Princeton shape benchmark," in Proceedings Shape Modeling Applications, 2004., pp. 167–178, IEEE, 2004. 81, 92, 118, 120, 122, 125

[144] B. Jansen van Rensburg, P. Puteaux, W. Puech, and J.-P. Pedeboy, "Homomorphic Two Tier Reversible Data Hiding In Encrypted 3D Objects," in 2021 IEEE International Conference on Image Processing (ICIP), pp. 3068–3072, 2021. 87, 169

[145] B. Jansen van Rensburg, P. Puteaux, W. Puech, and J.-P. Pedeboy, "3D Object Watermarking from Data Hiding in the Homomorphic Encrypted Domain," ACM Trans. Multimedia Comput. Commun. Appl., mar 2023. Just Accepted. 87, 169

[146] G. Lavoué, J. Vandeborre, H. Benhabiles, M. Daoudi, K. Huebner, M. Mortara, and M. Spagnuolo, "SHREC'12 Track: 3D Mesh Segmentation," in Eurographics Workshop on 3D Object Retrieval 2012, pp. 93–99, Eurographics Association, 2012. 92

[147] D. Pickup, X. Sun, P. L. Rosin, R. R. Martin, Z. Cheng, Z. Lian, M. Aono, A. B. Hamza, A. M. Bronstein, M. M. Bronstein, S. Bu, U. Castellani, S. Cheng, V. Garro, A. Giachetti, A. Godil, J. Han, H. Johan, L. Lai, B. Li, C. Li, H. Li, R. Litman, X. Liu, Z. Liu, Y. Lu, A. Tatsuma, and J. Ye, "Shape Retrieval of Non-Rigid 3D Human Models," in Eurographics Workshop on 3D Object Retrieval, pp. 101–110, Eurographics Association, 2014. 92

[148] Q. Zhou and A. Jacobson, "2018 Cover Image: Thingi10K," Computer Graphics Forum, vol. 37, no. 1, pp. 451–452, 2018. 92

[149] G. Lavoué, I. Cheng, and A. Basu, "Perceptual Quality Metrics for 3D Meshes: Towards an Optimal Multi-attribute Computational Model," in IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 3271–3276, IEEE, 2013. 94

[150] J. Guo, Contributions to objective and subjective visual quality assessment of 3D models. (Contributions à l'évaluation objective et subjective de la qualité visuelle des modèles 3D). PhD thesis, University of Lyon, France, 2016. 95

[151] B. T. Phong, "Illumination for Computer Generated Pictures," Communications of the ACM, vol. 18, no. 6, pp. 311–317, 1975. 96

[152] M. J. Dworkin et al., "SHA-3 standard: Permutation-based hash and extendable-output functions," 2015. 114, 119

[153] A. Said, "Measuring the strength of partial encryption schemes," in IEEE International Conference on Image Processing 2005, vol. 2, pp. II–1126, IEEE, 2005. 126

[154] O. Sorkine, "Laplacian Mesh Processing," in Eurographics 2005 - State of the Art Reports (Y. Chrysanthou and M. Magnor, eds.), The Eurographics Association, 2005. 126

[155] S. Beugnon, B. Jansen van Rensburg, N. Amalou, W. Puech, and J.-P. Pedeboy, "A 3D Visual Security (3DVS) score to measure the visual security level of selectively encrypted 3D objects," Signal Processing: Image Communication, vol. 108, p. 116832, 2022. 129, 169

[156] B. Jansen van Rensburg, W. Puech, and J.-P. Pedeboy, "A Hierarchical Decryption Method for an Eco-Friendly Securing of 3D Objects," in 2022 IEEE 24th International Workshop on Multimedia Signal Processing (MMSP), pp. 1–6, 2022. 130, 169

[157] B. Jansen van Rensburg, W. Puech, and J.-P. Pedeboy, "A Format Compliant Encryption Method for 3D Objects Allowing Hierarchical Decryption," IEEE Transactions on Multimedia, pp. 1–12, 2022. 130, 169

[158] "please offer an option for encryption," 2018. 133

[159] B. Jansen van Rensburg, W. Puech, and J.-P. Pedeboy, "The First Draco 3D Object Crypto-Compression Scheme," IEEE Access, vol. 10, pp. 10566–10574, 2022. 161, 169

[160] B. Jansen van Rensburg, W. Puech, and J.-P. Pedeboy, "Draco-Based Selective Crypto-Compression Method of 3D objects," in 2022 Eleventh International Conference on Image Processing Theory, Tools and Applications (IPTA), pp. 1–6, 2022. 161, 169

[161] B. Jansen van Rensburg, A. G. Bors, W. Puech, and J.-P. Pedeboy, "Simultaneous Watermarking and Draco 3D Object Compression Method," in 2023 IEEE International Conference on Image Processing (ICIP), 2023. 161, 170

[162] B. Jansen van Rensburg, A. G. Bors, W. Puech, and J.-P. Pedeboy, "Méthode Jointe de Tatouage et Compression Draco pour les Objets 3D," in COmpression et REprésentation des Signaux Audiovisuels 2023, 2023. 170

[163] E. Reinders, B. Jansen van Rensburg, P. Puteaux, and W. Puech, "Analyse d'images secrètes bruitées," in COmpression et REprésentation des Signaux Audiovisuels 2023, 2023. 170