Watermark Recovery from 2D Views of a 3D Video Object [*]

Emmanuel Garcia[†], Jean-Luc Dugelay[‡]
Institut Eurécom
2229 route des Crêtes, B.P. 193
06904 Sophia-Antipolis, France

ABSTRACT

In this paper, we describe a novel framework for watermarking 3D video objects via their texture information. Instead of classical existing algorithms dealing with 3D objects and that operate on meshes in order to protect the object itself, the main goal of our work is to retrieve information originally hidden in the texture image of the object, from *resulting images or videos* having used the 3D synthetic object.

After developing the theory and practical details of our 3D object watermarking scheme, we present preliminary results of several experiments carried out in various conditions, ranging from ideal conditions (e.g. a priori known rendering parameters) to more realistic conditions (e.g. rendering projection estimated from 2D view) or within the context of possible attacks (e.g. mesh reduction).

Keywords: Watermarking, 3D video objects, texture.

1. INTRODUCTION

Ever more synthetic objects can be used in videos or images. Watermarking can then be useful for several purposes. In particular, viewers would like to check in videos if an object is synthetic or natural, to check if the use of a given object is legal or not, to access additional information concerning that object (e.g. copyright, date of creation, and so on). This range of preoccupations will emerge with the increasing realism of synthetic objects, in particular realistic clones, and some upcoming standards such as MPEG-4 that will include possibilities to combine natural and synthetic information and allow users to easily manipulate data and make up virtual objects that look real.

Image watermarking is an emerging technique that allows to hide, in an invisible and robust manner, a message inside a digital audio-visual document. According to the targeted service, the message can contain some information about the owner (copyright), the picture itself (content authentication or indexing) or the buyer (non repudiation). It is then possible to recover the message at any time, even if the picture has been modified following one or several non destructive attacks (malicious or not).[1]

Originally mainly designed for the owner protection of still images, the range of both possible applications of watermarking technologies and possible covers (especially among new objects) have recently significantly increased.[2]

To the best of our knowledge,[4] all previous works dealing with 3D video objects are based on slight modifications performed on meshes via geometric and/or topological data of 3D objects. Typically, authors propose to modify either the 3D coordinates of some points or the connectivity of triangles within a mesh.[5–7]

2. TEXTURE WATERMARKING OF A 3D OBJECT

2.1. Principle of our approach

Figure 1 shows the principle of our 3D object watermarking scheme. Given a known 3D object consisting of a geometric definition, a texture image and a texture mapping function, we hide information in the object by watermarking its texture (step 1) using a robust watermarking algorithm (originally designed for still images). This watermarked object can then be released for further representations in virtual scenes (step 2). We can then check that the represented object is protected by extracting the watermarked texture (step 3) from the

[*]This research was supported in part by the EU Certimark project.[3]

[†]garciae@eurecom.fr
[‡]dugelay@eurecom.fr

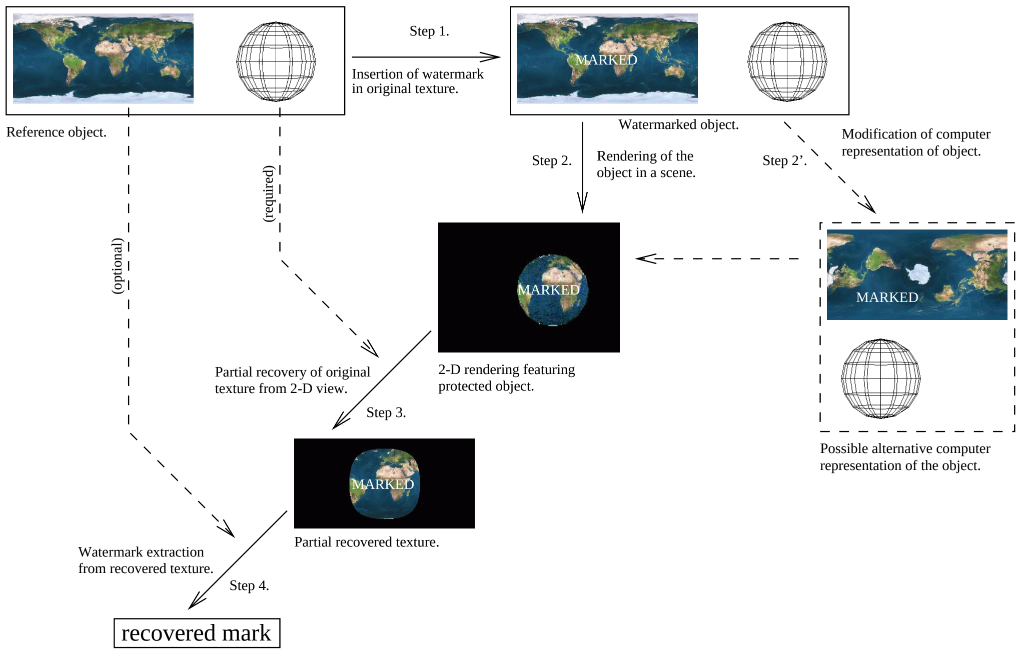
Step 1.
Insertion of watermark in original texture.
Reference object.
Watermarked object.
MARKED
Step 2.
Rendering of the object in a scene.
Step 2'.
Modification of computer representation of object.
(required)
(optional)
2-D rendering featuring protected object.
Possible alternative computer representation of the object.
Partial recovery of original texture from 2-D view.
Step 3.
Partial recovered texture.
Watermark extraction from recovered texture.
Step 4.
recovered mark

Figure 1. General overview.

represented views of the object, and by finally extracting the inserted watermark (step 4) from the recovered texture image.

Let us point out that this process does not depend on any modification of the internal representation of the released 3D object (either geometry or texture) that may have been performed to try to erase the mark or to simplify the object or for any other purpose, as long as the appearance of the rendered 3D object remains the same (step 2'), except if such modifications are destructive (e.g. severe subsampling of the texture image or severe mesh reduction).

One view of the 3D object generally provides only a partial knowledge of the whole texture. So it is better, if possible, to recover partial texture images from several 2D views of the 3D object (e.g. face and profiles of the model of a human face in a movie) and then to merge them into a more complete texture image.

In order to reverse the tranformation undergone by the watermarked reference texture image until it is viewed, we need to know the projection matrix of the virtual camera and the reference 3D object (geometry, texture, and texture mapping function)

Under these assumptions, the proposed framework is valid and can be implemented, and is also found to be resilient to any modification of the internal representation of the protected 3D object as underlined earlier.

2.2. Practical implementation

2.2.1. Partial texture reconstruction

As shown in figure 2,

- let S be the set of 3D points of the surface of the 3D object,
- let I be a rendered view of the 3D object,
- let T be the texture image to recover,
- let $\mathcal{F} : S \to T$ be the reference texture mapping function,

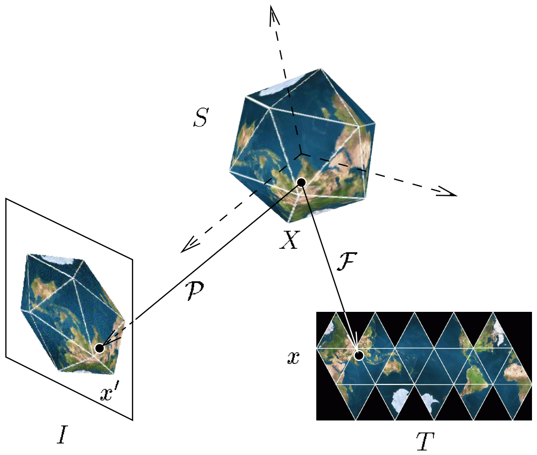

Figure 2. Texture mapping and viewing of a 3D object.

- let $\mathcal{P}$ be the projection from the system of coordinates of the reference 3D object onto the image of the virtual camera.

The main problem is to accurately compute the projection matrix $\mathcal{P}$ of the virtual camera (detailed in section 2.2.2). Assuming we know this projection matrix, to recover the part of texture T visible in image I we

1. consider each pixel x of T
2. compute $X = \mathcal{F}^{-1}(x)$ (if it exists)
3. compute the pixel $x' = \mathcal{P}(X)$ of the image I where X is projected
4. check that X is actually visible at pixel x' of the image I
5. set the color of x to be that of x' in case X is visible in I at pixel x'

The fourth step is required because several points of the 3D object could project on the same pixel in the image but (at least if we assume the object is opaque) only one would be seen. For this we used a Z-buffer technique associated with the observed image I.

Since x' generally has non-integer coordinates, its color is generally not directly specified and must be interpolated. What we currently did is rounding its coordinates to the closest integers and taking the color of the resulting pixel: this is a nearest neighbour interpolation. It is not obvious that using other interpolation schemes would improve the recovered texture, at least for the specific purpose of extracting the watermark. We only changed the interpolation scheme to bilinear in some experiments and observed no significant differences in the results.

2.2.2. Projection estimation

Even though it is possible to perfectly know the projection $\mathcal{P}$ that was used to render the 3D object in the case of controlled experiments carried out to empirically show the limits of our approach, this projection would not be known in a more realistic context but would have to be estimated between the 3D object itself and its 2D view, thereby performing a 3D/2D registration.

This projection must be estimated as accurately as possible in order to recover the original texture with as little distortion as possible (the theoretical limit being when this projection is known a priori in controlled experiments). It is possible, but still to be assessed, that the still image watermarking algorithm used could cope to some extent with the local deformations induced in the recovered texture by the imprecision in the projection's estimate.

In the first stage of this work we dealt with the case where the projection is known, in order to estimate the limit performances of our algorithm. This is the purpose of the first experiments that are presented in section 4. We also developed a 3D/2D registration algorithm in order to carry out other experiments, closer to what would be the final application of our algorithm. This is not the purpose of this article to present this registration algorithm. Let us only say that we found no suitable 3D/2D projective registration algorithm in the literature. Let us also point out that such an algorithm should estimate the whole 11 independant parameters of the projection and should ultimately be able to cope with the colorimetric differences between texture and view induced by synthetic lighting. In section 4, we present results obtained with our registration algorithm.

3. IMPEDIMENTS TO WATERMARK RECOVERY

Between the insertion of a watermark in the texture of a 3D object and its extraction, the texture may have been altered by various processes and manipulations in either of the following three stages: releasing of the object, rendering of the object or/and reconstruction of the texture, which will now be detailed.

3.1. Releasing of the object

After the releasing of the object for use, one might change the format of the object's geometrical description, perform mesh reduction and such, without modifying the visual aspect of the object's shape. In case the geometrical description is changed after releasing of the object, the geometry used for rendering (the modified geometry) and for texture reconstruction (the original geometry) would be different and this would induce geometrical distortions in the recovered texture with respect to the original one. However, in the specific case where the geometry is subject to an affine or even projective transform, without change of topology, this factor could be completely integrated in the estimation of the viewing projection so that this would not induce any change in the process of recovering the texture.

One might also modify the texture mapping function and the texture image accordingly so that the object would still look the same (e.g. the texture of the Earth can be represented using various mappings but the Earth globe still looks the same). These manipulations only add a noise in the final reconstructed texture image, especially when it is the texture mapping that is modified, because then the original texture image undergoes an additional warping before rendering which introduces resampling/interpolation noise due to the discrete nature of the image.

3.2. Rendering of the object

Due to the rendering of the object, the texture information is reduced and altered. First, the whole surface of the object is generally not visible and thus the texture information is cropped. Then, the perspective projection induces an alteration similar to that induced by an image warping and thus adds some noise to the texture information. In particular, when a part of the image is seen from a large distance, the corresponding texture information is visible only in its roughest details and thus suffers an alteration similar to a sub-sampling. And lastly, the texture information is modified, most likely in the low spatial frequencies, by the synthetic lighting conditions which may change the resulting colors of the object in the 2D view.

3.3. Reconstruction of the texture

In the last stage, i.e. reconstruction of the texture, the texture information is again affected by a noise due to the warping (reverse projection and reverse texture mapping) of a discrete image but also by another kind of alteration which is much more severe. This alteration consists in local deformations of the recovered texture image with respect to the original texture image and it comes from the imprecision with which the perpective projection is known. In the case of supervised experiments where this projection may be perfectly known, there would not be such local deformations, but only noise, cropping, and colorimetric alterations. However, we emphasize that in more realistic experiments, where the projection is not known a priori, the accuracy with which this projection is estimated is critical for the quality of the reconstructed texture image, and for limiting local deformations to a minimum.

3.4. Comparison with still image attacks

All those alterations are inherent to the process of recovering the watermarked texture from a 2D view. They can be compared to what is commonly called attacks (malicious or not) in the now well-known field of still image watermarking. In fact such attacks include in particular the alterations we are concerned with, that is, cropping, low-pass filtering, local geometric deformations and colorimetric attacks. Since our approach requires the use of a (possibly adapted) still image watermarking algorithm, we should choose one that is robust to the mentionned manipulations, if possible. In order to cope with local deformations, we can either estimate the viewing projection more accurately, or design a still image watermarking algorithm more robust to local deformations, whichever is less difficult.

4. NUMERICAL RESULTS

Let us report here the results of our preliminary experiments, aimed at probing the promises and limitations of our approach, as well as placing a few landmarks for further experiments. Since our scheme works on top of an underlying still image watermarking algorithm, the choice thereof is important. For that purpose we used Eurécom's still image watermarking algorithm[13] in all our experiments.

In the presented experiments, the watermarked 3D object was alone in a scene, with no background, and with no lighting, i.e. the texture was mapped as-is on the object and was projected as-is on the 2D image plane. Therefore the recovered texture does not suffer from any color alteration. We hid a varying number of bits in the texture of the 3D object to assess the optimal capacity under given conditions. In fact watermarking is commonly seen as a tradeoff between capacity (amount of data that can be hidden), robustness (resistance to attacks) and visibility (perceptibility of watermark) as illustrated by figure 3. It is impossible to improve one of the three characteristics without degrading one of the other two. For our experiments we explored the tradeoff between capacity and robustness with a fixed visibility of 38dB (signal-to-noise ratio between original and watermarked texture).

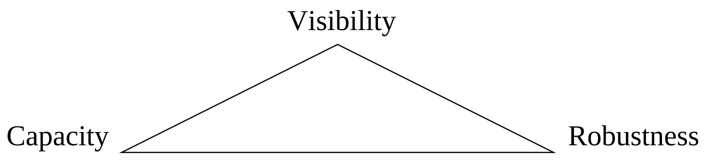

Figure 3. General watermarking tradeoff.

In each experiment we have also computed the number of independant recovered texture pixels, which, we believe, is a relevant information that is related to the quantity of information present in the recovered texture. For instance, if an object is seen from a large distance, it would be viewed as a small set of pixels. These would be magnified (duplicated) in the recovered texture but each would count only once as an independant pixel in the recovered texture. We now present the results obtained from three kinds of experiments.

4.1. Reference experiment

In this first set of experiments, the 3D/2D projection and the computer representation of the object used for the rendering are known during the texture reconstruction process. There is no need to turn on the feature of the still watermarking algorithm that can deal with local deformations since there are no such local deformations when the viewing projection is perfectly known.

Figure 4 (resp. 5) shows one (resp. the other) of the two objects we used for our experiments.

Figure 6 (resp. 7) shows a view of this object along with the texture image that was reconstructed from this view by performing the operations described in section 2.

Figure 8 (resp. 9) shows a view of the object from the same point of view but with a different scaling factor, along with the texture image that was reconstructed from it. We observe that the reconstructed texture image

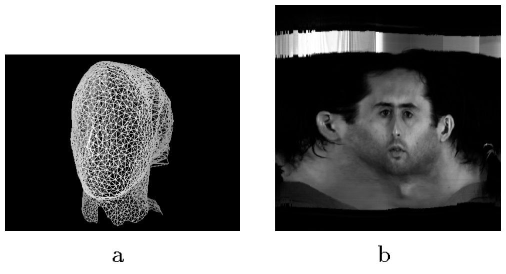
Figure 4. Original object #1, mesh (left) and texture (right).

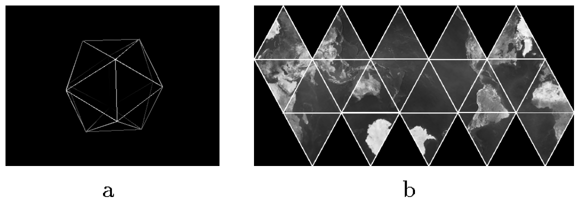
Figure 5. Original object #2, mesh (left) and texture (right).

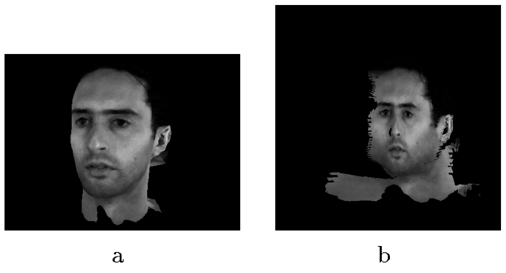
Figure 6. A view of object #1 (left) and the corresponding recovered texture image (right).

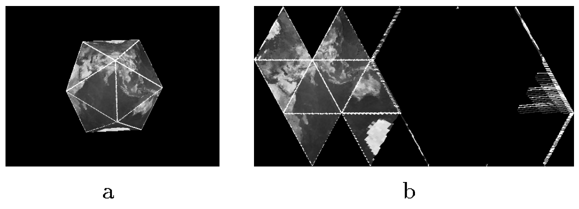
Figure 7. A view of object #2 (left) and the corresponding recovered texture image (right).

has the same shape than the previous reconstruted texture image since the point of view is the same (the same part of the object is visible) and only the scaling factor changes. However it has a lower definition than the previous reconstructed texture image because the object appears smaller and less details are discernable.

Figure 10 (resp. 11) shows the results of our first experiment. In this experiment, we inserted a 64-bit watermark in the texture image of the object presented in figure 4. We then viewed this object under the same point of view as in figure 6 (resp. figure 7) with a varying scaling factor. Finally, for each view (function of the scaling factor) we reconstructed the texture image as in figure 6 and ran the still image watermark extraction algorithm on it. We show the number of erroneous bits in the extracted watermark as a function of the number of independant observable texture pixels in the view used for texture recovery. The solid line represents a fit of the data points using a minimum mean squared error criterion and a fitting model of the form

$$\frac{y}{64} = \frac{1}{2} erfc(\alpha x^{\beta})$$

where $erfc$ denotes the "complimentary error function". This choice, however arbitrary, is justified by the fact that x is somewhat related to a quantity of information and that $\frac{y}{64}$ represents the bit error rate, and in that case we can expect a formula close to the formulas obtained in information theory for noisy channels. When x is zero, the bit error rate would be $\frac{1}{2}$ which sounds fair since the bits of the watermark are chosen completely at random in the absence of information. And at the opposite the bit error rate drops very quickly and stays to zero when there is enough information and more. In the case of figure 10, we can observe that with about 10000 distinct visible pixels we can expect between 0 and 1 erroneous bit in the recovered 64-bit mark. A more secure result could be enforced using error correcting codes.

Figure 12 (resp. figure 13) shows the results obtained when hiding a 256-bit mark in the same view as figure 6 (resp. figure 7) and with varying scaling factor. We observe that the bit error rate (number of erroneous bits over size of the mark) is higher and converges slower to zero than when using a 64-bit mark. For curve fitting we used the same formula involving the bit error rate (we replaced $\frac{y}{64}$ with $\frac{y}{256}$).

Finally, figure 14 (resp. figure 15) shows the number of erroneous bits when trying to recover a mark of varying size and when using a magnified (high resolution) view of the object. After a certain threshold, the number of

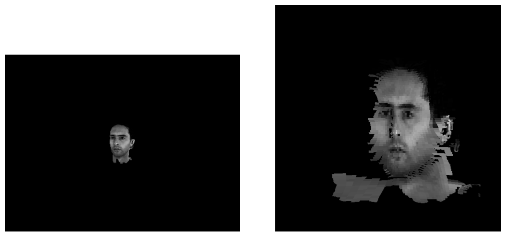

Figure 8. Same view of object #1 as in figure 6 but with a different scale (left) and corresponding recovered texture image (right).

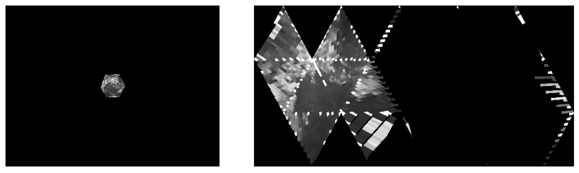

Figure 9. Same view of object #2 as in figure 7 but with a different scale (left) and corresponding recovered texture image (right).

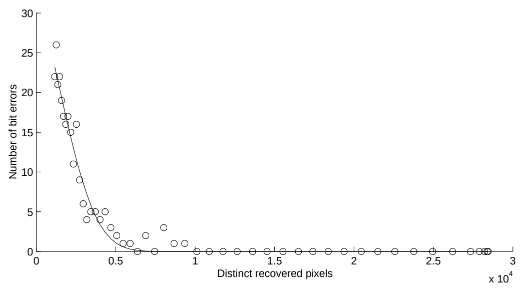

Figure 10. 64-bit watermark. View of figures 6 with varying scaling factor.

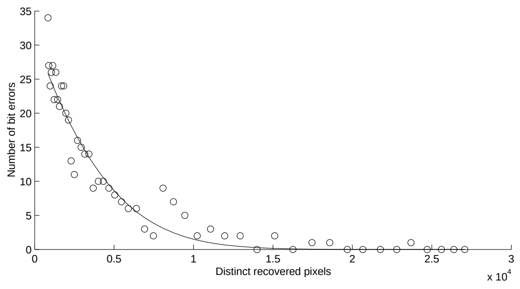

Figure 11. 64-bit watermark. View of figure 7 with varying scaling factor.

errors seems to increases dramatically with the attempted size of the watermark. It is likely that this threshold (around 200 bits here) depends linearly on the amount of observable data (around 28000 distinct pixels here) but we did not yet experimentally verify this point.

4.2. Estimated projection

In this second set of experiments, we have not used the knowledge of the viewing projection but instead we have used an estimation provided by a 3D/2D projective registration algorithm that we developed for that purpose. For the sake of comparing the performance of this registration algorithm with other current or future algorithms, let us just say that when applied to the 3D object and the view of figure 16a, the projection of the 3D object using the estimated projection was on average 1 pixel away from its real location.

The performances are expected to degrade unless, maybe, if we used a still image watermarking algorithm robust to slight local deformations. Actually, the used software (Eurémark) is robust to (at least Stirmark's) local deformations at the cost of reducing the capacity by half. While we did not yet experiment with this feature, it would be interesting to see how it could compensate for the local deformations induced by the inaccuracy in the projection's estimate.

Figure 16a shows the same view as in figure 6 but the projection that was used to reconstruct the texture image is not the actual viewing projection, but an imprecise estimated one. Thus, the reconstructed texture image (figure 16b) should be a slightly warped version of the original texture image which could impair the watermark extraction process. In fact if we compare the results of figure 17 with those of figure 14, we see that a 200-bit mark can no longer be hidden and recovered without errors. Only a 64-bit mark can be somewhat reliably used.

Now it is a matter of designing good 3D/2D registration algorithms that could accurately estimate a viewing projection and/or of designing a still image watermaking algorithm that can cope with slight local geometrical deformations.

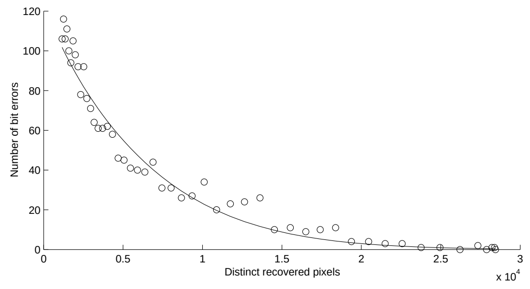

Figure 12. 256-bit watermark. View of figure 6 with varying scaling factor.

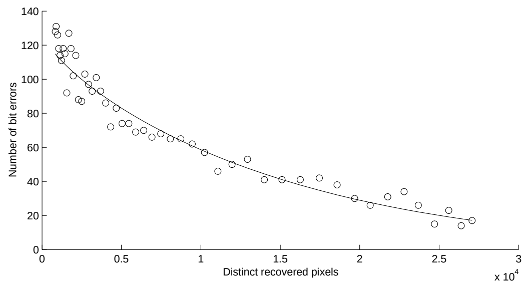

Figure 13. 256-bit watermark. View of figure 7 with varying scaling factor.

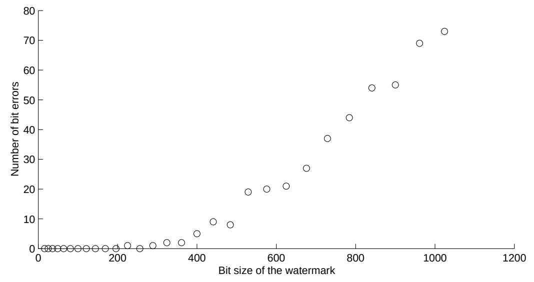

Figure 14. Varying size of watermark. View of figure 6 with 28445 observable texture pixels.

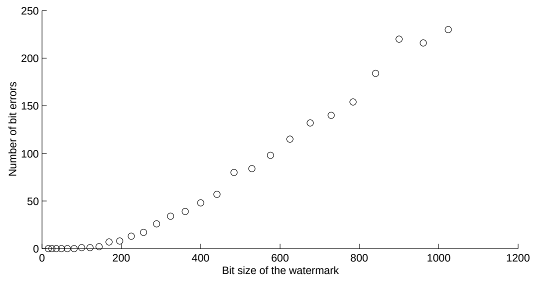

Figure 15. Varying size of watermark. View of figure 7 with 27073 observable texture pixels.

4.3. Mesh simplification

In this third set of experiments, we used the knowledge of the viewing projection, like in the reference experiment, but we modified the computer representation of the 3D object just before rendering by performing a mesh reduction[§] (we use triangle meshes to describe the geometry of our objects, but this is not a requirement for the applicability of our algorithm).

Figure 18a shows the same view as figure 6 but using the object simplified down to 100 triangles and figure 18b shows the reconstructed texture image that was built from this view and from the *original geometry* (10000 triangles) which is the only geometric information at the disposal of the retriever.

Since there is a mismatch between the geometrical descriptions used for rendering and for texture reconstruction we should expect that the reconstructed texture be a distorted version of the original one, thus introducing errors in the recovered watermark.

Figure 19 shows the number of bit errors when trying to recover a 256-bit from the same view as shown in figure 6 but using a description of varying complexity of the object. We can note no significant degradation from 10000 to 1000 triangles, but below 1000 triangles, the object's geometrical alteration induce too severe geometrical distortions in the reconstructed texture image and the bit error rate increases dramatically. It is still to assess whether a still watermarking algorithm that is robust to local geometrical deformations could help cope with 3D geometrical deformations.

5. CONCLUDING REMARKS

The idea developed in this paper is to protect the use of a 3D object, instead of a particular computer representation of the object itself, by watermarking its texture in a given reference representation.

[§]We used Michael Garland's QSlim mesh simplification software[12] to perform mesh simplifications.

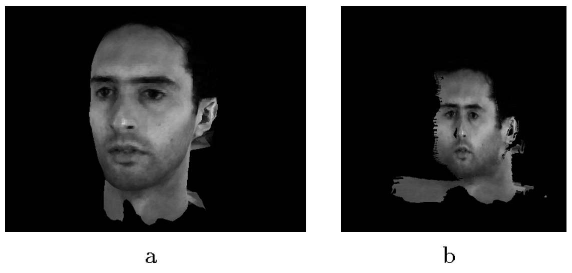

Figure 16. Same view as in figure 4 (left) and corresponding reconstructed texture image using estimated projection (right).

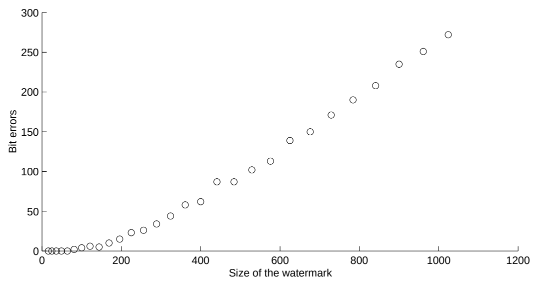

Figure 17. View of figure 16. Varying watermark size. 27975 observable texture pixels.

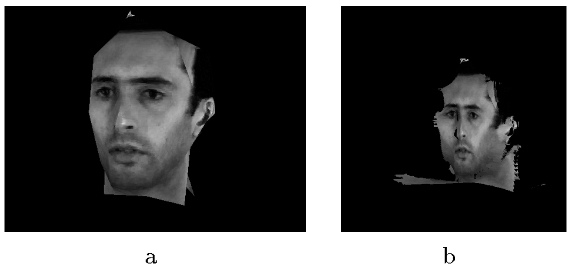

Figure 18. View of the object of figure 4 simplified down to 100 triangles (left), and corresponding reconstructed texture image (right).

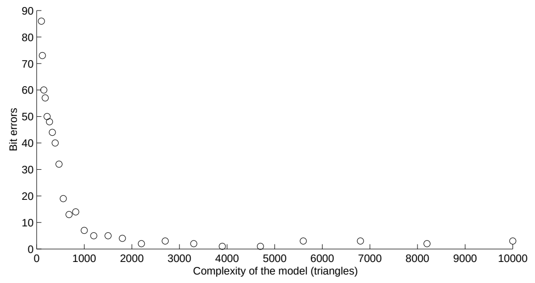

Figure 19. View of figure 18. 256-bit mark. Varying object complexity. Constant scale. 27975 observable texture pixels when using the 100-triangle object.

As reported, this idea is realistic under certain conditions: we must know the reference 3D object (geometry, texture and texture mapping function), which means that we recover the watermark in a non-blind mode, we must know the position and orientation of the object with respect to the virtual camera that was used to render it, and we must know the intrinsic parameters of this camera.

We carried out a wide range of experiments on the model of a human face and on a 2D view of it. For our reference experiment we observed the 3D object under a known point of view and with known rendering parameters. We varied the scale of the view and the size of the watermark, thus outlining the tradeoff between the resolution of the view, the size of the watermark, and the correctness of the watermark recovery. A point of reference may be the successful retrieval of a 64-bit mark from a view where around 10000 distinct texture pixels of the object were seen (figure 10). This is meant only to provide a rough idea of the upper limit of the performances that can be expected from our algorithm but they are subject to variations according to the many parameters involved. Results were not as good when using another object for instance (figure 11), and given the large number of factors, we can only suggest, among many other possible explanations, that it is because the shape of the involved subpart of the picture was very different in both cases, which may have a significant influence on the still image watermarking algorithm used.

In another experiment we did not assume to know the point of view from which the object was seen but we estimated it using an ad-hoc 3D/2D projective registration algorithm. The results (figure 17) were obviously worse than in the ideal case of the reference experiment (figure 14) but nevertherless show that our framework is valid.

In yet another experiment (figure 19), we simplified the triangulated mesh of the model before rendering (between 100 and 10000 triangles) but for the watermark recovery we used the original 10000-triangle model in which the watermark was hidden. The recovery of the mark was possible even from the view of a model simplified in an unknown way, which backs up the claim that our algorithm is inherently resilient to modifications of the computer representation of the 3D object as long as they do not significantly modify its aspect.

Those are only the first steps towards texture-based 3D watermarking and many hurdles lie ahead. In fact we should ultimately be able to work from a view of an object with unknown lighting conditions. While the still image watermarking algorithm currently used in our experiment could cope with color alterations caused by synthetic lighting, this is not the case of our projective registration algorithm. Since what will determine the performance of our 3D watermarking algorithm in the end is the ability to register a 3D model with a 2D view in the presence of synthetic lighting, future work should definitely be dedicated to finding good projective registration algorithms for this specific purpose.

Finally, let us point out the possibility of a possible future path, combining the complementary features of our texture-based approach, and of "traditional" 3D watermarking algorithms that aim at hiding information in the geometry data.

REFERENCES

1. Katzenbeisser (S.), Petitcolas (F. A.P.), *Information Hiding - Techniques for Steganography and Digital Watermarking*, Artech House, Boston-London, 2000.
2. *Special Session on Watermarking for Industrial Applications*, 2001 IEEE Fourth Workshop on Multimedia Signal Processing, October 3-5, Cannes.
3. European Project - IST-1999-10987, *CERTIMARK - Certification for watermarking technique*, `http://www.certimark.org`.
4. C. Mallauran, *Internship Report on 3-D Video Objects Watermarking*, Eurécom/ESSI-UNSA, September 2001.
5. Olivier Benedens, *Watermarking of 3D polygon based models with robustness against mesh simplification*, Proceedings of SPIE: Security and Watermarking of Multimedia Contents, SPIE, pp. 329-340, 1999.
6. Yutarou Ohbuchi, Hiroshi Masuda, Masaki Aono, *Geometrical and Non-Geometrical Targets for Data Embedding in Three-Dimensional Polygonal Models*, Computer communications, Elsevier, August 1998.
7. E.Praun, H.Hoppe, A.Finkelstein, *Robust Mesh Watermarking*, ACM Siggraph 99 Conference Proceedings, Los Angeles, California, August 1999.
8. J.-L. Dugelay, *Method for hiding binary data in a digital image*, Pending Patent PCT/FR99/00485 (EURECOM 09-PCT), March 1999.
9. J.-L. Dugelay & S. Roche, *Process for marking a multimedia document, such as an image, by generating a mark*, Pending Patent EP 99480075.3 (EURECOM 11/12 EP), July 1999.
10. J.-L. Dugelay & C. Rey, *Method of marking a multimedia document having improved robustness*, Pending Patent EUP 99480075.3 (EURECOM 14 EP), May 2001.
11. J.-L. Dugelay & C. Rey, *Image Watermarking for Owner and Content Authentication*, ACM Multimedia, Los Angeles, California, US, November, 2000.
12. M. Garland, *QSlim mesh simplication software*, `http://graphics.cs.uiuc.edu/~garland/software/qslim.html`.
13. J.-L. Dugelay, F. A. Petitcolas, *Image watermarking: possible counterattacks against random geometric distortions*, Proc. of SPIE Security and Watermarking of Multimedia Contents II, San Jose, California, Jan. 2000.