A Mesh Watermarking Approach for Appearance Attributes

Liangjun Zhang Ruofeng Tong Feiqi Su Jinxiang Dong
State Key Laboratory of CAD & CG, AI Institute, Zhejiang University, Hangzhou, 310027

Abstract

This paper describes an algorithm to watermark appearance attributes, as well as shape of mesh. Appearance attributes are potential watermarking primitives and watermarking approach for them can be generalized from that for the shape. The major challenge of generalization is that the watermarking for appearance attributes has more constraints. Our paper focuses on this challenge. Especially for normal vector, we embed watermark by modifying its orientation, not magnitude. Results show our scheme effectively improves the capacity and enhances the robustness of mesh watermarking.

Keywords: Mesh watermarking, Appearance attributes.

1. Introduction

Digital watermarking provides a mechanism for copyright protection of digital media. Most of mesh watermarking approaches [1,2,5] target shape of mesh, and few focus on appearance attributes such as color, normal vector, and texture coordinate [4]. Embedding watermark into them can improve watermarking capacity. Moreover, we can generalize the shape approach to work for them. The major challenge of generalization is that the watermarking for appearance attributes has more constraints. For instance, mesh's color components should be in a range [0, 1], and the watermark embedded into normal vector must resist the normalization operation. In addition, since the modification on them will affect the mesh appearance more greatly, the change induced may be more perceptual.

In our paper, we extend *Praun*'s watermarking scheme [5] to appearance attributes, and address this challenge.

2. Embedding

2.1. Overview of the embedding process

Construct PM. The original mesh is decomposed into the *progress meshes* (PM) representation by applying *edge collapse* transformations [3].

Identify m refinements. We identify the *m* refinement operations [3] that cause the greatest change to the mesh. Here we emphasize that when measuring the change, not only geometry change but also appearance change should be considered [3,5]. In each refinement, the magnitude of mesh change *H* is defined below.

$$H = We_0 |Hg| + We_1 |Hc| + We_2 |Hn| + We_3 |Ht|$$

Where *Hg*, *Hc*, *Hn*, and *Ht* are geometry, color, normal vector and texture coordinate magnitudes of mesh in this refinement. We_0, We_1, We_2, We_3 are their weights. $\sum We_i = 1$ ($We_i \in (0,1)$)

Calculate basis functions. In each refinement *i* of the selected *m* refinements, we calculate scalar basis function $\Phi_{i,j}$ over the vertices v_j [5]. Though each watermarking primitive can share the basis function, we define one set for every primitive. The reason is that the basis function may be dynamically adjusted in the embedding process (Section 2.2), and to improve watermarking robustness, we can deliberately select one set for each primitive.

Modify attributes. In each identified refinement, we embed watermark coefficients into shape and appearance by letting each coefficient induce small modification on the attributes according to the basis functions.

We denote the modification process in one refinement as a *loop*. The detail modification rules are shown below.

2.2. Color and texture coordinate attributes

For each *loop i*, we modify color of every vertex *j*.

$$C'_j = C_j + \varepsilon_1 Hc_i \phi^1_{ij} w_{4i+1}$$

Where C_j , C'_j are the old and modified color of vertex j,

ε_1 is a user-provided global parameter,

w_{4i+1} is watermark coefficient. w_{4i}, w_{4i+2} and w_{4i+3} are embedded into vertex position, normal vector and texture coordinate of mesh.

Color components (*RGB-Model*) are in a range [0, 1], and we should prevent them from overflow or underflow. The solution by dynamically adjusting ε_1 would lead to the watermark extraction failure. An available method is that if the modification result is greater than 1, we clip it to 1 (if it is less than 0, clip it to 0) and adjust the basis function $\Phi^1_{i,j}$ correspondingly.

Proceedings of the 10 th Pacific Conference on Computer Graphics and Applications (PG'02)
0-7695-1784-6/02 $17.00 © 2002 IEEE

Texture coordinate is a good watermarking primitive, since it is crucial for proper rendering of texture-mapped objects, and it is difficult to regenerate once lost [4]. Our embedding method for texture coordinate is as same as that for color.

2.3. Normal vector attribute

The color's magnitude modification rule isn't applicable to the normal vector, as the normalization operation would destroy the embedded information. Here we can modify its orientation by rotating it around an axis. Since our watermark extraction (section 3) requires the deviation of the normal vector should be added linearly, the axis should be identical in each *loop*.

$$N'_j = \text{rotate}(N_{axis}, N_j, \varepsilon_2 Hn_i \varphi_{ij}^2 w_{4i+2})$$

Where N_j and N'_j are *the old and modified normal vector of vertex j, Function rotate rotates the normal vector around the axis N_{axis}.*

Because the normal vector is a significant factor that effects mesh appearance, the arbitrary perturbation on it will badly destroy the appearance. In our approach, since the normal vector magnitude isn't changed, and its orientation deviation is regulated by the basis function, the appearance is well preserved.

3. Extracting

Register and resample. Because the similarity transforms and attacks may change the mesh's geometry and topology, we need to register and resample the suspect mesh before the watermark extraction. Here we employ Praun's *registration* and *resampling* method [5].

Calculate difference of attributes and solve square systems. By taking the difference of attributes between the original and sampled mesh, we extract the watermark by solving the sparse linear least squares system.

$$B_0 W_0^* = (V^* - V) \qquad B_1 W_1^* = (C^* - C)$$
$$B_2 W_2^* = \text{PlaneAngle}(N_{axis}, N^*, N)$$
$$B_3 W_3^* = (T^* - T)$$

Where W^*_i is the extracted watermark,
B_0, B_1, B_2 and B_3 are the modification matrixes associating with attributes induced by watermarking.
Function PlaneAngle(N_{axis}, N_1, N_2) calculates angle between the plane defined by N_{axis} and N_1, and the plane by N_{axis} and N_2.

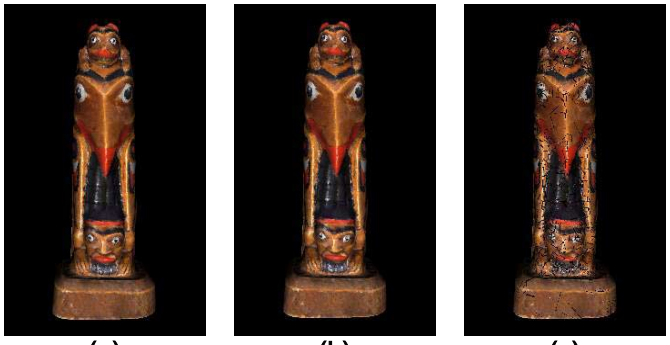
Finally, we combine the extracted sub-watermark W^*_i to obtain the entire watermark W^*.

Analyze the extracted watermark. We compare the inserted and extracted watermark by calculating their linear correlation [5]. If the result is greater than the deliberately chosen threshold, we conclude that the watermark is present in the suspect mesh.

4. Results

Fig 1 illustrates the effect of watermarking into texture coordinate. Fig 1(a) is an original 'Totem' model; Fig 1(b) is the watermarked model with ε_3 = 0.001. The modification is perceptual (Fig 1(c)) when ε_3 is 0.003.

Results show that our method can embed watermark into appearance attributes, while keeping the appearance almost unchanged. Furthermore, the robustness of our watermarking algorithm is comparable to *Praun*'s [5].



(a) (b) (c)

Figure 1: The original model and the results when we embed watermark into its texture coordinate (a) Totem (5,017 faces, 15,016 vertices) (b) Watermarking with ε_3 = 0.001 (c) Watermarking with ε_3 = 0.003

5. Conclusion and future work

Our method for mesh appearance attributes effectively improves the watermark capacity and enhances its robustness. This project suggests some possible future work: we would like to extend our method for other appearance attributes of mesh, such as *BRDF* coefficients.

References

[1] Benedens, O. Geometry-based watermarking of 3d models. *IEEE Computer Graphics and Applications*. 1999. 46-55.

[2] Date,H. Kanai,S. Kishinami,T. Digital Watermarking for 3d Polygonal Model Based on Wavelet Transform. *Proceedings of DETC'99*. 1999.

[3] Hoppe,H. Progressive meshes. *ACM SIGGRAPH 96 Conference Proceedings*. 1996. 99-108.

[4] Ohbuchi,R., Masuda,H., Aono,M. Data Embedding Algorithms for Geometrical and Non-Geometrical Targets in Three-Dimensional Polygonal Models. Computer Communications [in press].

[5] Praun E,Hoppe H,Finkelstein A. Robust mesh watermarking. *SIGGRAPH'99 Proceedings*. 1999. 49-56.