A New Constrained Texture Mapping Method

Yan-Wen Guo[1,2,⋆], Jin Wang[1], Xiu-Fen Cui[1], and Qun-Sheng Peng[1,2]

[1] State Key Lab of CAD&CG, Zhejiang University, Hangzhou 310027, China
[2] Department of Mathematics, Zhejiang University, Hangzhou 310027, China
{ywguo, jwang, xfcui, peng}@cad.zju.edu.cn

Abstract. The validity of texture mapping is an important issue for point or mesh based surfaces. This paper provides a new constrained texture mapping method which is capable of ensuring the validity of texture mapping. The method employs the "divided-and-ruled" strategy to construct a direct correspondence between the respective patches of the texture image and 3D mesh model with feature matching. The mesh model is segmented based on the "approximate shortest path". Further, a "virtual image" relaxation scheme is performed to refine the rendering effect. We show how mesh morphing can be conducted efficiently with our constrained texture mapping method. Experiment results demonstrate the satisfactory effects for both texture mapping and mesh morphing.

Keywords: Texture mapping; Approximate shortest path; Morphing.

1 Introduction

Texture mapping plays an important role in computer graphics and virtual reality. Without this technique, it would be a time-consuming task to render the details of complex models and the nature scene with realistic appearance. To accomplish texture mapping from a texture image onto a 3D face mesh model, the fundamental issue is to construct a one-to-one correspondence between the model and texture plane. Conventionally, this is achieved by parametrization of the mesh surface. Furthermore, some applications demand rigid correspondence between the features of model and those of the texture image during the parametrization process, for example, when mapping an image of a face to a 3D face model, the details of the eyes, nose, etc must be mapped precisely to the correspondent parts on the face model. This problem is commonly referred to as constrained texture mapping. The constrained texture mapping of a 3D mesh is a tedious work since no intrinsic parametrization of the 3D mesh satisfying these constraints is available. Although several constrained texture mapping approaches were proposed in recent years, most of them make no guarantee of the validity of the parametrization.

1.1 Related Work

Continuous efforts were made concerning constrained texture mapping in the past several years [3] [4] [6] [14] [15]. Most of the proposed methods mainly deal

⋆ Corresponding author.

F. Kishino et al. (Eds.): ICEC 2005, LNCS 3711, pp. 48–58, 2005.

with the mesh model with disc topology, since a complicated mesh model can always be decomposed into several "meaningful" disc topological patches and all current texture mapping methods can be applied to a patch with disc topology.

Levy proposed a method capable of dealing with iso-parametric curves [3]. Subsequently Levy solved the problem of feature mapping by respecting an arbitrary set of constrained features [4] and enabled the method to satisfy the constraints. Levy's method works well for a small number of constraints but can lead to invalid parametrization when dealing with a large set of constraints. The main drawback of the above approaches is that both of them cannot always ensure the validity of the texture mapping. Validity is an essential and important issue for constrained texture mapping which can be regarded as a problem of special parametrization of the mesh surface.

In a recent work, Kraevoy *et al* brought forward a method [6] called Matchmaker. This method adopts the "divided-and-ruled" method to parameterize the 3D meshes onto the planar region. Although this method can ensure the validity of texture mapping, it needs to parameterize the mesh model onto the planar region as a pre-processing, which is in fact a difficult work. On the other hand, it incurs incorporating with steiner vertices and the number of steiner vertices may vary from several tens to thousands according to the complexity of the mesh model. The process is therefore tedious. Furthermore, if the 3D mesh model is of low density or the mesh quality is poor, the decomposed triangular patches are usually jagged, and the matching between the irregular patches and the triangles on the texture plane must undergo a refinement operation as a post-process.

1.2 Our Work

This paper provides a new constrained texture mapping method ensuring the validity of parametrization. The method employs also the "divided-and-ruled" strategy; however it does not need to parameterize the 3D mesh model as a preprocess, rather it applies directly a decomposition scheme to the 3D mesh model and parameterize each divided 3D patch. The texture image is segmented based on the same feature set. Texture mapping is then preformed between each pair of corresponding mesh region and texture triangle.

Besides the validity of the texture mapping, our method has two advantages over the previous methods. Firstly, our method does not inquire a global parametrization of the entire mesh as a preprocess, instead, it performs a local parametrization over each segmented region on the mesh, thus both the complexity and computation of the parametrization are greatly reduced; Secondly, our method supports direct mapping between each pair of corresponding parts on both the 3D mesh and the texture image, user can intuitively adjust the locations of the feature points to get the desired mapping result.

The rest of this paper is organized as follows: In Section 2, we describe our constrained texture mapping method in detail. Section 3 shows some experimental results. In Section 4 we show how the proposed method can be extended to solve mesh morphing. The last section summarizes our method and discusses the future work.

2 A Divided-and-Ruled Algorithm

The main steps of our algorithm can be summarized as follows (see Fig. 1):

Step 1 Interaction: Specify the same set of feature points on both the mesh model and the texture image. This is the only interaction involved in our method.

Step 2 Decomposition and Matching: Triangulate the texture plane based on the feature points by Delaunay triangulation, decompose the mesh model accordingly using "approximate shortest path" on meshes.

Step 3 Embedding: Embed each patch on the mesh to the corresponding triangle on the texture plane to generate the texture coordinates.

Step 4 Refinement: Refine the texture mapping effect using a virtual image relaxation approach for vertices along the boundaries.

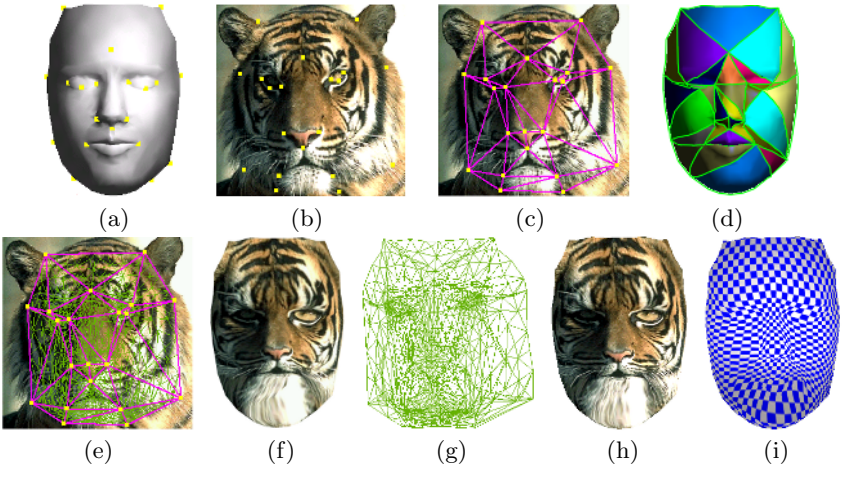

Fig. 1. The stages of the method. (a) feature vertices on the mesh model; (b) feature points on the texture image; (c) triangulation of the feature points; (d) decomposed different patches; (e) the embedding result; (f) the direct mapping result without a further refinement; (g) texture mesh after refinement; (h) the final rendering effect; (i) the effect of mapping grids with the same texture coordinates as (g).

2.1 Notations

For description convenience, we introduce some notations:

- $V = \{v_i | i = 1, ..., n\}$: the specified feature vertices set on the mesh model.
- $P = \{p_i | i = 1, ..., n\}$: the corresponding feature points set on the texture image, v_i corresponds to p_i.
- T_P: the triangle set obtained by triangulating the texture plane with feature points set P.
- E_P: the edge set of T_P.

2.2 Interaction

For a disc topological model, our method first specifies some feature vertices on its surface, including some selected vertices along the boundary of 3D mesh denoted by $V_b = \{v_{bi} | i = 1, ..., m\}$. It is obvious that V_b is a subset of V. Accordingly, some feature points are specified on the texture image, some of which are denoted by $P_b = \{p_{bi} | i = 1, ..., m\}$, p_{bi} corresponds to v_{bi} in V_b.

2.3 Decomposition and Matching

The purpose of this step is to construct the correspondences between the patches of mesh model and those of the texture image. So the mesh model and texture image must firstly be segmented.

The texture image is triangulated based on the feature points of P with boundary constraints, which means that two ordinal points p_{bi}, $p_{b(i+1)}$ of P_b compose a boundary edge of the triangulation so as to preserve the same connectivity as the vertices in V_b. Suppose that $E_b = \{e_{bi} = (p_{bi}, p_{b(i+1)}) | i = 1, ..., m\}$ stands for the edge set consisting of the vertices in P_b, E_b is defined as boundary constraint for the Delaunay triangulation of the feature points on the image (see Fig. 2). It is sure that E_b is included in E_P.

After triangulating the texture plane with the feature points, the mesh model is decomposed accordingly by a kind of precise shortest path called "approximate shortest path" on meshes.

Unlike the Dijsktra shortest path which connects two specified vertices consisting of the edges on mesh, *approximate shortest path* [7] [8] is a more precise shortest path passing through the surface of the mesh. We employ an algorithm similar to the one proposed by Zhang *et al* to compute the *approximate shortest path* [8]. Their method represents the triangle mesh by a weighted graph, each node of the graph denotes a vertex of the mesh and each edge represents an edge of the triangular mesh. Traditional Dijsktra algorithm is then applied to calculate the shortest path between two feature points on the graph, by iteratively subdividing the related triangle edges and constructing new weighted graph, the

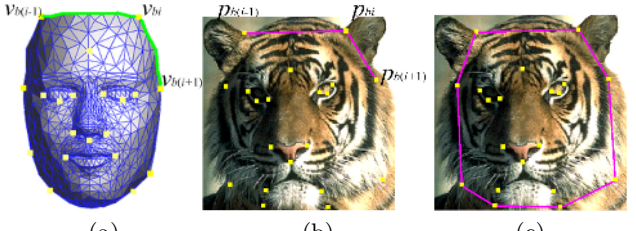

Fig. 2. Boundary constraint for triangulation. Since v_{bi} has direct topology relation with $v_{b(i-1)}$ and $v_{b(i+1)}$ in some sense (a), there should be two edges linking p_{bi} with $p_{b(i-1)}$ and $p_{b(i+1)}$ respectively (b). The edge set in order linking $p_{b(i-1)}$ with p_{bi} is the boundary constraint for Delauny triangulation (c).

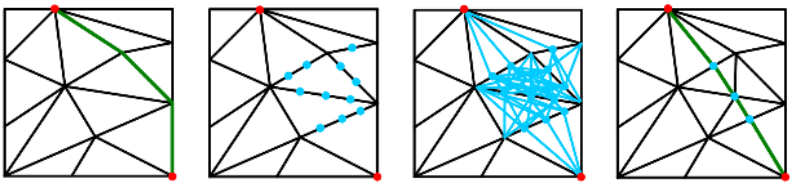

Fig. 3. The process for computing *approximate shortest path*. (a) Dijsktra shortest path on original mesh connecting two points (red dots); (b) subdivision on the related edge; (c) the constructed new graph for subdivision; (d) the final shortest path by applying Dijsktra algorithm to the new graph and the remeshed meshes.

shortest path between points on the graph finally approaches the shortest path on the mesh surface. Rather than subdividing the related edges iteratively, we adopt a more straightforward method and subdivide the related edge on the graph only once according to the length of the edge. Given a constant length, each edge is inserted with several intermediate points and the number of inserted points amounts to the ratio of the edge length to the given length. Thus a long edge is divided into more fragments compared with a short edge. This method of computing the shortest path is highly efficient and quite suitable for our demand. Note that remeshing is needed on the correlative region of the shortest path. Fig. 3 gives an example of computing the shortest path.

For each edge $e_i = (p_j, p_k)$ in E_p, if e_i is in E_b, then v_j, v_k are connected by the boundary, otherwise, we connect v_j, v_k by the *approximate shortest path*. The pathes connecting adjacent feature vertices are generated one by one so as to maintain the same topology as the triangulation result of the texture image. When generating one new shortest path, a number of validity conditions must be taken into account as suggested by Kraevoy *et al* in [9], such as, the new path must not intersect with existing paths except at the ends of the path;

Table 1. Algorithm: Matching process

```
while E_p is not empty
begin
  Pop edge e=(p_j, p_k) from E_p;
  if e ∈ E_b
    Connect v_j and v_k with boundary vertices lying between them;
  else
    Compute approximate shortest path S_Path between v_j and v_k;
    if S_Path satisfies validity conditions
      Connect v_j and v_k with S_Path;
    else
      Utilize the method propose by Kraevoy et al [9] to generate the path
    between v_j and v_k;
end
```

the new path must not block necessary future paths, etc. In case that one of these conditions is violated, additional Steiner vertices must be introduced for generating new path between two feature vertices.

Nevertheless, the invalidity situations seldom occur in our experiments. This is because our approach adopts the *approximate shortest path* which is a kind of relatively precise shortest path independent of the density of meshes. Note that, the Dijsktra shortest pathes advocated by Kraevoy's paper may incur unforeseen situation since the Dijsktra shortest path may be highly irregular if the concerned mesh is of poor quality and low sampling density.

The above algorithm can be described by the codes presented in **Table 1**.

2.4 Embedding and Refinement

The correspondence between the texture plane and mesh model is constructed by decomposing them into consistent triangular patches, and then the texture coordinates of each point on the 3D mesh can be determined by embedding its triangular patch of the mesh onto the corresponding triangle on the texture plane. We use the convex combination parametrization method [1] [2] with mean value coordinates [5] to compute the embedding.

Up to now, we are ready to map the texture image onto the mesh model precisely. As shown in Fig. 1, (f) is an initial mapping result. Although this effect is remarkably well in some sense, the distortion of the initial embedding is relatively high especially near the boundary of each patch due to boundary constraints of the local parametrization. To handle this problem, we can optimize the texture meshes (the embedding mesh in the Fig. 1 (e)) to generate more ideal rendering effect.

We conduct the mesh optimization by applying the similar method proposed in [16] to every unstrained point on the texture mesh. Each planar texture coordinates are recomputed by averaging its neighboring coordinates with some weighted value, such as the mean value [5], while the new coordinates should guarantee the triangles incident on this vertex without foldovers.

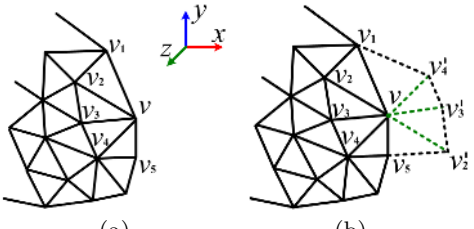

Fig. 4. The "virtual image" scheme. (a) $v_1, ..., v_5$: the neighbors of a boundary vertex v. (b) v_2', v_3' and v_4': the virtual reflection. With this scheme, we in fact obtain an weighted form of v about v_2, v_3 and v_4 that will facilitate the refinement operation.

Nevertheless the mean value coordinates of the boundary vertex is difficult to compute, since its neighboring vertices cannot form a circuit. We propose here a "virtual image" scheme to deal with this situation.

Suppose that a boundary vertex v has a number of incident vertices $v_1, ..., v_n$, where v_1 and v_n are also boundary vertices. For the interior vertices v_2,...,v_{n-1}, we calculate their "virtual images" $v'_2, ..., v'_{n-1}$ which are the symmetrical vertices of $v'_2, ..., v'_{n-1}$ about v. Connecting them with v_1, v and v_n as show in Fig. 4 (b) will create a circuit around v and we can thus compute the mean value coordinates of v with its neighbors and their "virtual images". Note that since $v'_2, ..., v'_{n-1}$ are the linear combination of v with v_2,...,v_{n-1} respectively, we can obtain a weighted value form of v based on $v_1, ..., v_n$, which facilitates the optimization operation. (Fig. 4).

3 Experimental Results

We adopted several examples to test the validity of our texture mapping algorithm on an Intel Pentium IV 2.0GHz PC with 512MB main memory under the Windows XP operating system (In Figs. 5, 6, 7 the yellow dots are the specified feature points). Table 2 gives the performance results, among which the 2th, 3rd and 4th columns present the time for computing the *approximate shortest path*, embedding and refinement. It can be seen that the total time varies from several seconds to less than 20 seconds, among which the computation of shortest path and embedding consume much more time.

Fig. 5 demonstrates the process of seamlessly mapping two images of a girl from different viewpoints onto the Igea model which is composed of 4442 triangles and 2223 vertices. Different from traditional methods, we do not need to segment the Igea model into two parts and parameterize each path as pre-processing. The sole interaction is to specify 58 feature points on the texture image Fig. 5 (b) and 45 feature vertices on Igea Fig. 5 (a), among which 13 are located on the imaginary bisection line of the mesh model. Fig. 5 (f) shows the rendering results of the model after applying our constrained texture mapping approach.

Fig. 6 shows the results of mapping the texture of a rabbit and its mirror image onto a 3D rabbit model (850 triangles and 462 vertices), while Fig. 7 concerns the mapping of a tiger texture and a boy image onto a cow head model (1891 triangles and 968 vertices) and a human head model (2923 triangles and 1532 vertices) respectively.

4 Extension to Mesh Morphing

Mesh morphing is an interesting research topic for mesh based models. To achieve a natural smooth morphing between the source model and the target model, it is essential to set up a correspondence between the feature points on both models. This is accomplished in current approaches [10] [11] [13] by conducting a global parametrization of the two mesh surfaces. The parametrization of the source model is then embedded into the parametric domain of the target mesh and

Table 2. Performance results

Model	Shortest path (s)	Embedding (s)	Refinement (s)	Total time (s)
Rachel	1.71125	0.755625	0.023625	2.490500
Igea	7.011625	9.179875	0.773250	16.96475
Rabbit	1.232200	0.562523	0.013320	1.808043
Cow	2.959125	2.508000	0.029250	5.496375
Face	2.947500	3.621000	0.036750	6.605250

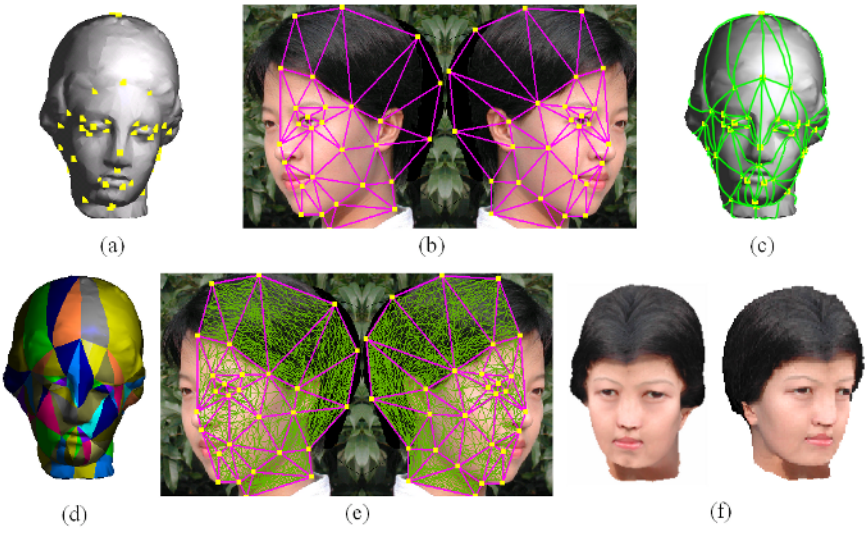

Fig. 5. The rendering effect of mapping two charts of mirrored image onto the Igea model

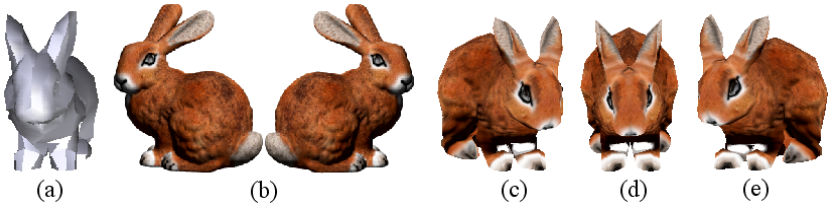

Fig. 6. The rendering effect of mapping two charts of mirrored rabbit onto the Rabbit model

warped to yield a new version which is consistent with that of the target model with feature alignment. No wonder that the above process is complicated and time-consuming.

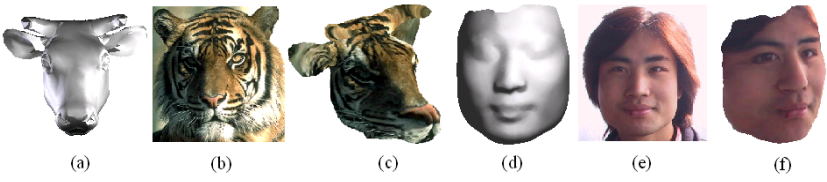

Fig. 7. Other texture mapping results for mesh model with disc topology

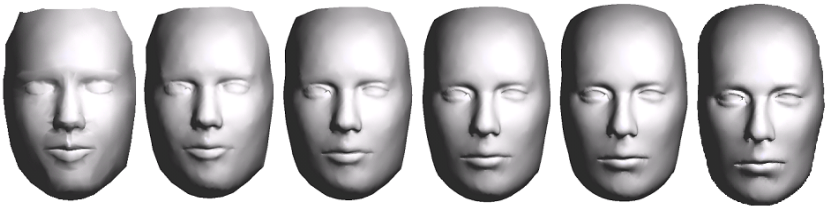

Fig. 8. The morphing process from Rachel to Head

Fig. 9. The morphing process from Isis to Beethoven

Since the fundamental issue of mesh morphing is also feature alignment, we can easily extend our method for constrained texture mapping to mesh morphing. Instead of parameterizing both the source mesh and the target mesh simultaneously, our method need only parameterize the target mesh. This 2D global parametrization result is regarded as a texture image and triangulated based on specified feature points. The souce mesh is then decomposed into segments accordingly. After a process similar to constrained texture mapping applying to both the source mesh and target texture, each segmented patch of the source mesh is embedded into the corresponding triangle in the parametric domain of the target mesh. Finally, resampling and remeshing operation is performed to generate the same topology between the source mesh and target mesh. Note that our morphing method also employs the "divided-and-rules" ideology, the correspondence is constructed precisely and efficiently.

Fig. 8 and Fig. 9 show two morphing results generated by our method. Note that the feature on the intermediate model is well preserved since the feature region on the source model is precisely matched with that on the target model by our method.

5 Conclusions and Future Work

In this paper a new constrained texture mapping method is presented. The method maps precisely the texture image onto the mesh model, discarding the parametrization of the model as a pre-processing adopted by most of the previous methods, at the same time guarantees the validity of constrained texture mapping which is often a problem of constrained parametrization. Instead of Dijsktra shortest path, *Approximate shortest path* is employed to generate the path between feature vertices, which can indeed alleviate the distortions incurred by mapping and runs in interactive time. Since only the feature points on the texture image and the mesh model are needed to be specified, the interaction involved in this method is also simple, especially for the close model. Experiment results show the satisfactory effect of this method.

The provided algorithm can be easily extended to process other feature matching problem, such as mesh morphing, we treat it as an application of our proposed approach. Part of the processing in our algorithm, such as computing the shortest path, can be ported to the Graphics Processing Unit (GPU) that may realize real-time texture mapping. The problem of how to determine a limited set of feature points with the best morphing effect remains as a future work to be concerned. Some other applications such as expression colon for meshes and image driven meshes etc., are also the works in the future.

Acknowledgement

Some of the models for experiments are fetched from [12].

This paper is supported by National 973 program (No. 2002CB312101) and NSFC grant (No. 60033010)&(No.60403038).

References

1. Floater, M.S.: Parametrization and Smooth Approximation of Surface Triangulations. Computer Aided Geometric Design. Vol. 14(3), (1997) 231-250
2. Eck, M., DeRose, T., Duchamp, T., Hoppe, H.: Mutiresolution Analysis of Arbitrary Meshes. In: Proceedings of ACM SIGGRAPH 1995. Los Angeles, CA, (1995) 173-183
3. Levy, B., Mallet, J.L.: Non-Distortion Texture Mapping for Sheared Triangulated Meshes. In: Proceedings of ACM SIGGRAPH 1998. Orlando FL, (1998) 343-352
4. Levy, B.: Constrained Texture Mapping for Polygonal Meshes. In: Proceedings of ACM SIGGRAPH 2001. Los Angeles CA, (2001) 417-424
5. Floater, M.S.: Mean Value Coordinates. Computer Aided Geometric Design. Vol. 20(1), (2003) 19-27
6. Kraevoy, V., Sheffer, A., Gotsman, C.: Matchmaker: Constructing Constrained Texture Maps. In: Proceedings of ACM SIGGRAPH 2003, ACM Transactions on Graphics. Vol. 22(3), (2003) 326-333
7. Kanai, T., Suzuki, H.: Approximate Shortest Path on a Polyhedral Surface and its Applications. Computer-Aided Design. Vol. 33, (2001) 801-811

8. Zhang, L.Y., Wu, X.: Approximate Shortest Path on Triangular Mesh Surface. Journal of CAD&CG. Vol. 15(5), (2003) 592-597
9. Kraevoy, V., Sheffer, A.: Cross-Parameterization and Compatible Remeshing of 3D Models. In: Proceedings of ACM SIGGRAPH 2004, ACM Transactions on Graphics. Vol. 23(3), (2004) 861-869
10. Alexa, M.: Merging Polyhedral Shapes with Scattered Features. The Visual Computer. Vol. 16(1), (2000) 26-37
11. Praun, E., Hoppe, H.: Spherical Parameterization and Remeshing. In Proceedings of ACM SIGGRAPH 2003, ACM Transactions on Graphics. San Diego. Vol. 22(3), (2003) 340-349
12. CYBERWARE INC., http://www.cyberware.com
13. Shapiro, A., Tal, A.: Polyhedron Realization for Shape Transformation. The Visual Computer. Vol. 14(8), (1998) 429-444
14. Matsushita, K., Kaneko T.: Efficient and Handy Texture Mapping on 3D Surfaces. In Proceedings of EUROGRAPHICS 1999, Computer Graphics Forum. Vol. 18(3). (1999) 349-358
15. Eckstеinl, L., Surazhsky, V., Gotsman, C.: Texture Mapping with Hard Constraints. Computer Graphics Forum. Vol. 20(3), (2001) 95-104
16. Sander, P., Gortler, S., Snyder, J., Hoppe, H.: Signal Specialized Parametrization. In Proceedings of Eurographics Workshop on Rendering 2002. (2002)