Auto-colorization of 3D Models from Images

Juncheng Liu
Institute of Computer Science and Technology
Peking University
PR China

Zhouhui Lian*
Institute of Computer Science and Technology
Peking University
PR China

Jianguo Xiao
Institute of Computer Science and Technology
Peking University
PR China

ABSTRACT

Color is crucial to achieve more realism and better visual perception. However, majority of existing 3D model repositories are colorless. In this paper, we propose an automatic scheme for 3D model colorization taking advantage of large availability of realistic 2D images with similar appearance. Specifically, we establish a region-based correspondence between a given 3D model and its 2D image counterpart. Then we employ a PatchMatch based approach to synthesize the texture images. Subsequently, we quilt the texture seams via multi-view coverage. Finally, the texture coordinates are obtained by projecting back to the 3D model. Our method yields satisfactory results in most situations even when there exists an inconsistency between the 3D model and the corresponding image. Results obtained through a cross-over experiment validate the effectiveness and generality of our method.

CCS CONCEPTS

• **Computing methodologies → Texturing**; **Shape analysis**;

KEYWORDS

colorization, 3D model, texture synthesis

ACM Reference Format:
Juncheng Liu, Zhouhui Lian*, and Jianguo Xiao. 2017. Auto-colorization of 3D Models from Images. In *Proceedings of SIGGRAPH Asia 2017 Technical Briefs, Bangkok, Thailand, November 27–30, 2017 (SA '17 Technical Briefs)*, 4 pages.
https://doi.org/10.1145/3145749.3149432

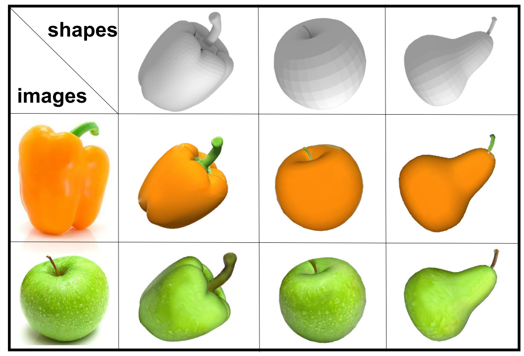


Figure 1: Our colorization results. The top row displays three shapes: pepper, apple and pear. The images on the left column are two realistic depictions of class "pepper" and "apple". Our goal is to colorize the three models with the provided images.

1 INTRODUCTION

With the development of 3D scanning and modeling techniques, more and more 3D model repositories [Chang et al. 2015] are becoming available. Researchers have focused on effective ways of shape perception and processing including 3D shape retrieval, recognition, and synthesis etc. Compared to 2D images, 3D shapes have full descriptions of geometry instead of only the visible faces. However, most of the existing models (including scanned and CAD models) are colorless. The reason for scanning is that observing color directly from camera is sometimes unreliable due to the lighting condition and reflection attributes of surfaces. When creating a CAD model by hand-craft, users focus on the mesh geometry hence models often come without the companion of valid texture. Creating a texture image for 3D model often requires parametrization and painting directly on the parametrized mesh on 2D plane. This procedure needs professional skills and a great amount of efforts which is not suitable for novice users.

Colorized 3D models not only bring stronger sense of reality for visual experience, but also helps improve cognition tasks such as classification and retrieval. With the ever-growing interest of personalized manufacture, for instance, the 3D printing, more realistic 3D models with color are desired. Since there exist large amounts of realistic images on the Internet, we are able to take advantages of them and transfer the color to the 3D objects. Early work includes TextureMontage [Zhou et al. 2005], which can colorize a 3D model seamlessly with multiple images of textures. However, it is often difficult to find several consistent images of different views for an identical object from the Internet. Image only contains the appearance of an object from a single view.

The most recent visible work of image-based 3D model colorization is the Texture Transfer [Wang et al. 2016]. Their method takes as input the 3D shape and 2D image pair. Texture patches are extracted from images and rectified using the corresponding 3D geometry. However, instead of establishing dense correspondence, their method merely overlays image and model by assuming that the two objects are nearly-identical. This assumption is very easily violated due to the topological difference and geometric variation even with same semantic components. Besides, their method extracts small patches from images and the whole model is tiled by expanding

*Corresponding author: lianzhouhui@pku.edu.cn

SA '17 Technical Briefs, November 27–30, 2017, Bangkok, Thailand

ACM ISBN 978-1-4503-5406-6/17/11…$15.00
https://doi.org/10.1145/3145749.3149432

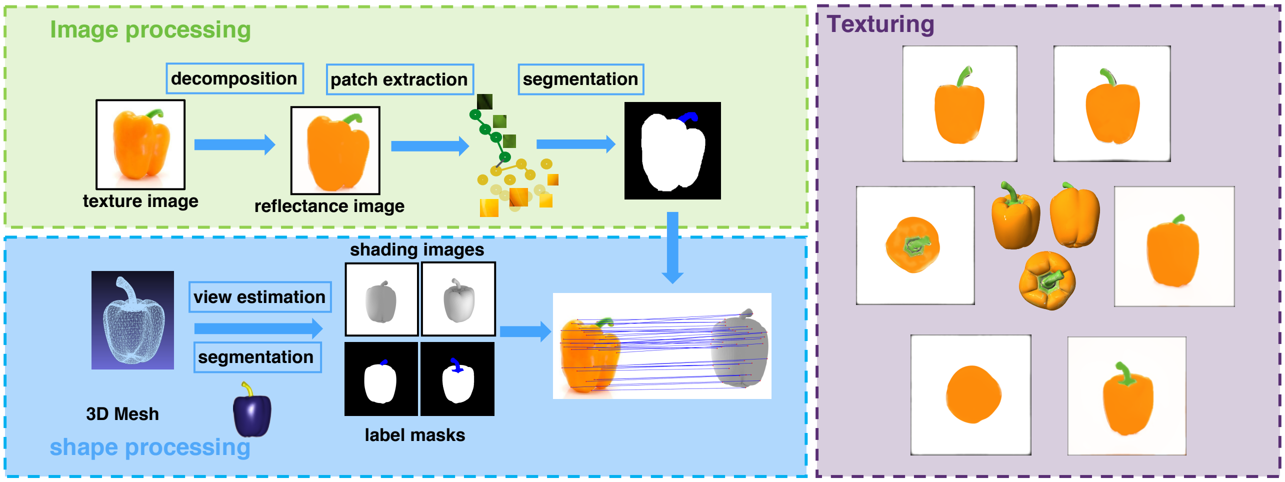


Figure 2: The pipeline of our proposed method which can be decomposed into image processing, shape processing and texture synthesis stages.

the small rectified patches, which leads to the homogeneous texture within a component. However, in real world, there exist large amounts of objects with inhomogeneous materials. [Zhou et al. 2017] could generate inhomogeneous by indicating the guidance map. However, their method is unaware of the geometry traits of the 3D models and the interaction between model and image. Another related work is [Bi et al. 2017] which also leverages the texture synthesis to produce texture mappings.

In this paper, we propose an automatic method for 3D model colorization with a relaxed required similarity between image-model pair as shown in Figure 1. Furthermore, by employing the PatchMatch [Barnes et al. 2009] method, we are able to produce inhomogeneous textures within a single part of objects.

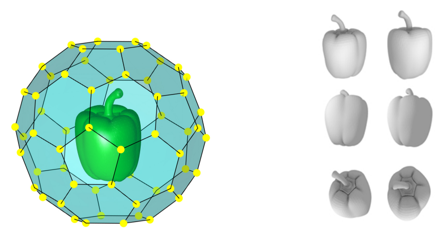
2 PREPROCESSING

Our method takes as input a model-image pair (denoted as $\mathcal{M}, \mathcal{I}$) with same semantic category. We aim to colorize the 3D model $\mathcal{M} = \{\mathcal{V}, \mathcal{F}\}$ ($\mathcal{V}$ denotes the vertex set and $\mathcal{F}$ denotes the face set, respectively) with the appearance provided by the image $\mathcal{I}$ in a plausible way. The pipeline of our method is illustrated in Figure 2.

In this section, several steps are conducted to preprocess the input model and image for the subsequent procedures. First, we bridge the 3D model and 2D image by estimating the viewing direction. Second, the input image is decomposed into several regions with consistent color. Meanwhile, we segment the 3D model into parts. The results in this step are utilized in the texture synthesis step.

2.1 View Estimation

For view estimation, we take 60 different views positioned on the vertices of a soccer ball $\{v_i | i = 1, 2, \ldots, 60\}$ as illustrated in Figure 3. For each view direction we render a 2D image of the model by assigning it a Lambert material using Blender. For the 60 rendered images and the input image, we extract their corresponding descriptors (HOG [Dalal and Triggs 2005] in our implementation). And



(a) model and sampled view points (b) sample rendered views

Figure 3: 60 sampled view points on a soccer ball surface, marked as yellow dots.

the input image is assigned a viewing direction that minimizes the descriptor difference:

$$v^* = \underset{v \in \mathcal{R}^3}{\arg\min} \ \|F(R(v, \mathcal{M})) - F(\mathcal{I})\|, \tag{1}$$

where v denotes the viewing direction, R represents the rendering operation, and F is the feature extraction operator. This step establishes a bridge that relates the model and image. However, our method does not require a high accuracy of view recovery. This is due to the fact that the 3D model and the object depicted on the image are not necessarily identical. Our following texture synthesis based strategy also compensates the inaccuracy yielded in this step.

2.2 Texture Image Analysis

In this step, we analyze the input image $\mathcal{I}$ and segment it into several regions with homogeneous color. To obtain the intrinsic color of objects, we first employ intrinsic image decomposition technique [Bi et al. 2015] to discard the shading effects. Subsequently, we extract homogeneous patches with different sizes by a quad-tree decomposition as illustrated by Figure 4. The decomposition stops when the color contained in the patch has a variance less than a given threshold σ. After that, we have patches with nearly uniform color.

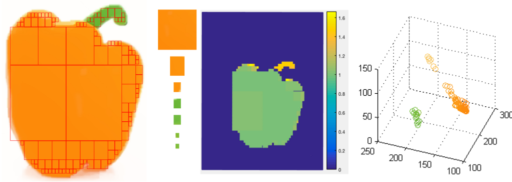


Figure 4: Source image segmentation. Left: the quad-tree decomposition and extracted patches (squares on the right). Middle: pixels are assigned with ISOMAP dimensionality reduction results. Right: patches distribution in color space.

Having the patches, we extract the color histogram descriptor for each patch, and then reduce the dimensionality of the feature vector to one using ISOMAP [Tenenbaum et al. 2000]. After this step, each patch is assigned a scalar value. This scalar value functions as an attribute that ranks these patches according to its corresponding color distribution. However, To achieve a smoother result, we employ a graph-partition [Bernstein and Wojtan 2013; Grady and Schwartz 2006] based smoothing method:

$$\min_{x} \sum_{(i,j)\in Edges} w_{ij}(x_i - x_j)^2 + \gamma \sum_{i\in Pixels} w_i(x_i - r_i)^2, \quad (2)$$

where x is the optimization variable, γ is the weighting parameter, and r denotes the computed ISOMAP values. $w_{i,j} = 1$ if adjacency pixels i and j belong to the same region, $w_{i,j} = 0$ otherwise. This problem boils down to linear equations which can be solved very efficiently by Cholesky decomposition. Having the indicators we can segment the image by clustering.

3 TEXTURE SYNTHESIS

In this section, our method produces texture images for model colorization. We adopt a view-based strategy to colorize the model. Specifically, vast majority of the visible faces are covered by 6 different views. After generating texture image for each individual view, we map these texture coordinates back to 3D mesh to obtain a seamless colorization.

3.1 Part-based Correspondence

The first step is to generate the texture image for the estimated view v^*. To accomplish this, we employ a classical texture synthesis scheme PatchMatch [Barnes et al. 2009]. However, simply using the PatchMatch is not sufficient for our task because of the mismatching. Therefore, we need to establish a "correspondence" between $\mathcal{I}$ and $R(v^*, \mathcal{M})$ to boost the original PatchMatch.

Instead of a dense pixel-based correspondence, we propose to utilize a more relaxed correspondence, namely, region-based correspondence, which is chosen for the two following reasons: 1) Correspondence is inaccurate in terms of pixel level, however, integrating the pixels to region significantly improves the results; 2) Since currently the 3D model is still colorless, establishing correspondence directly on its shading image is even more difficult than matching two color images.

Because 3D model contains more information than its shading image, we do not directly segment the shading image $R(v^*, \mathcal{M})$. Instead, we decompose 3D model $\mathcal{M}$ into components $\mathcal{M}_1, \mathcal{M}_2, ..., \mathcal{M}_n$ by its geometry property SDF [Shapira et al. 2008] and then obtain

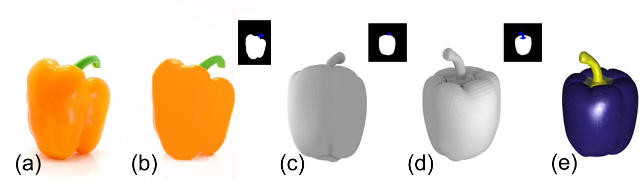


Figure 5: An illustrative figure of image and shape segmentation. (a) the original input image. (b) the decomposed reflectance image and its region indication mask (top right corner). (c) shading image of input model by estimated view. (d) the shading image of back view. (e) the segmented 3D model with different color indicating each component.

the pixel-wise labels of $R(v^*, \mathcal{M})$ by projecting the corresponding faces to view plane, that is:

$$p^i_j = \mathbf{P}v^i_j, \quad \{v^i_1, v^i_2, v^i_3\} \in \mathcal{M}_i \cap VC(\mathcal{F}), \\ i = 1, 2, ..., n. \;\; j = 1, 2, 3. \quad (3)$$

$\mathbf{P}$ denotes the camera matrix:

$$\mathbf{P} = \mathbf{K}\,[\mathbf{R} \mid \mathbf{t}],$$

where $\mathbf{K}$ is the intrinsic matrix and $\mathbf{R}$, $\mathbf{t}$ represent the rotation and translation, respectively. We adjust the corresponding value to fit the view direction v^*.

The projected pixel coordinates $\{p^i_1, p^i_2, p^i_3\}$ of the vertices triplet $\{v^i_1, v^i_2, v^i_3\}$ of the i-th component are thus obtained by left-multiplying the camera matrix. Having the three projected pixel coordinates ,we can then assign label i to the pixels within the triangular region formed by $\{p^i_1, p^i_2, p^i_3\}$. The process is iterated for all the faces in $\mathcal{M}_i$ and all the mesh components $i = 1, 2, ..., n$. The function VC is the visibility indicator implemented by visibility culling.

To establish dense correspondence between the segmented image and shading image shown in Figure 5, we first uniformly sample the valid region of shading image, leading to a point set $X = \{x_1, x_2, ..., x_m\}$. Similarly, we obtain a point set $Y = \{y_1, y_2, ..., y_m\}$ when the same operation is applied to segmented input image $\mathcal{I}$. Recall that we already have segmented the two images, we are able to mark each sample point with its corresponding region label denoted as mapping $\Phi_1 : i \mapsto \ell$ and $\Phi_2 : i \mapsto \ell'$, where $\ell \in \mathcal{L}$ and $\ell' \in \mathcal{L}'$ is the label sets. After applying a non-rigid point set registration method CPD [Myronenko and Song 2010], we have two sets of matched points. The matching result is represented by the mapping function $f : i \mapsto j$. The region correspondence is obtained by the following majority voting method:

$$\Psi(\ell) = \arg\max_{\ell'\in\mathcal{L}'} \sum_{\Phi_1(i)=\ell} \mathbb{1}\{\Phi_2(f(i)) = \ell'\}. \quad (4)$$

The formula above describes the process when each individual region in shading image $R(v^*, \mathcal{M})$ votes for the corresponding region in input image $\mathcal{I}$. The regions with the same assigned label will be merged.

3.2 View-based Colorization

After establishing the region-based correspondence, we are able to conduct PatchMatch with the region guidance. Technically, we produce the mask of each image with the each corresponding region

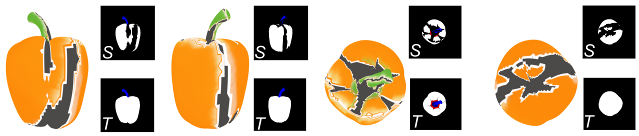


Figure 6: Four supplementary views. From left to right: side view1, side view2, top view, bottom view and their corresponding source and target masks for to be used in the PatchMatch procedure.

being filled with the same indicating color. By leveraging PatchMatch, we obtain the synthesized texture image for the estimated view v^*. The next step is to synthesize the back view side. This problem is actually not well-defined because the information of the back side of the object is missing in the image. As we know, large numbers of objects in real world are symmetric especially artificial ones. We take advantage of this observation and synthesize the back face by using the same source image. The region guidance is produced and leveraged in the same way as in the front view.

By pasting the texture images to each individual side of object, we obtain the initial results as shown in Figure 6. As we can see, it is often insufficient to cover models by their two opposite sides. There exist empty seams not being assigned texture coordinates, i.e. the triangles not covered by the two views. We solve this problem by generating 4 other views: top, bottom, and two side views. The textures for them are produced by image inpainting.

4 RESULTS

The proposed approach is implemented in MATLAB 2014a and a laptop with a Core i7 3.60Hz CPU and 8GB memory. All the data are obtained from the Internet. The input images are manually background-removed.

We evaluate the proposed method by conducting a cross-over experiment. Three classes of models are selected: pepper, pear and apple. For colorization, we choose 2 images for each class from the Internet. That is, each model shares the same 6 source images, two of which coincide with the object class and the other four belong to other 2 classes. This experiment is designed to test the robustness and adaptability of our method when handling objects that are "visually similar" yet "semantically different".

A subset of the results are collected in Figure 1. For more results please refer to the accompanied supplementary materials. The top row displays three shapes: pepper, apple and pear. The images on the left column are two realistic depiction of class "pepper" and "apple". We can observe from the results that for the visually similar image-model pair "apple" and "pepper", the colorized models are satisfactory. For the cross-over sets: "pepper-apple", "apple-pepper", "pear-apple", "pear-pepper", even there exist large variants between their layouts, the synthesis keeps plausible. Note the water drops in the source image of apple are preserved on the colorized models. This validates our method's ability to produce inhomogeneous and detailed textures.

5 CONCLUSION AND FUTURE WORK

In this paper, we proposed an automatic approach for 3D model colorization. The proposed method takes as inputs the image-model pair and yields plausible global texturing results by synthesizing texture images using PatchMatch and region guidance. Compared to existing methods, our scheme does not take the assumption of accordance between the 2D image and 3D model by leveraging the region-based correspondence. Furthermore, our method could yield both homogeneous and inhomogeneous textures. Since our method is totally automatic, the existing 3D repositories could be boosted by this auto-colorization method which facilitates the subsequent tasks such as recognition and retrieval. Additionally, the colorized models are more potential and suitable for shape modeling and training samples for machine learning techniques.

The future work will focus on generating more smooth texture images and more precise scheme of semantic segmentations. As described in the article, noncontinuous cracks are observed at the junction areas on the model. This is because we conduct the texture synthesis at 6 separate views. In the future we will try to produce textures with separate views united.

ACKNOWLEDGMENTS

This work was supported by National Natural Science Foundation of China (Grant No.: 61672043, 61472015 and 61672056), Beijing Natural Science Foundation (Grant No.: 4152022), National Language Committee of China (Grant No.: ZDI135-9), and Key Laboratory of Science, Technology and Standard in Press Industry (Key Laboratory of Intelligent Press Media Technology).

REFERENCES

Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. 2009. PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 28, 3 (Aug. 2009).

Gilbert Louis Bernstein and Chris Wojtan. 2013. Putting holes in holey geometry: topology change for arbitrary surfaces. *Acm Transactions on Graphics* 32, 4 (2013), 1–12.

Sai Bi, Xiaoguang Han, and Yizhou Yu. 2015. An L 1 image transform for edge-preserving smoothing and scene-level intrinsic decomposition. *Acm Transactions on Graphics* 34, 4 (2015), 78.

Sai Bi, Nima Khademi Kalantari, and Ravi Ramamoorthi. 2017. Patch-based optimization for image-based texture mapping. 36 (07 2017), 1–11.

Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. 2015. *ShapeNet: An Information-Rich 3D Model Repository.* Technical Report arXiv:1512.03012 [cs.GR]. Stanford University — Princeton University — Toyota Technological Institute at Chicago.

Navneet Dalal and Bill Triggs. 2005. Histograms of Oriented Gradients for Human Detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on.* 886–893.

Leo Grady and Eric L. Schwartz. 2006. Isoperimetric Graph Partitioning for Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28, 3 (2006), 469.

Andriy Myronenko and Xubo Song. 2010. Point Set Registration: Coherent Point Drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32, 12 (2010), 2262–2275.

Lior Shapira, Ariel Shamir, and Daniel Cohen-Or. 2008. Consistent mesh partitioning and skeletonisation using the shape diameter function. *Visual Computer* 24, 4 (2008), 249.

Joshua B. Tenenbaum, Vin De Silva, and John C. Langford. 2000. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science* 290, 5500 (2000), 2319.

Tuanfeng Y. Wang, Hao Su, Qixing Huang, Jingwei Huang, Leonidas Guibas, and Niloy J. Mitra. 2016. Unsupervised Texture Transfer from Images to Model Collections. *ACM Trans. Graph.* 35, 6, Article 177 (2016), 13 pages. http://doi.acm.org/10.1145/2980179.2982404

Kun Zhou, Xi Wang, Yiying Tong, Mathieu Desbrun, Baining Guo, and Heung-Yeung Shum. 2005. Texturemontage: Seamless Texturing of Arbitrary Surfaces From Multiple Images. *ACM Transactions on Graphics* 24, 3 (2005), 1148–1155.

Yang Zhou, Huajie Shi, Dani Lischinski, Minglun Gong, Johannes Kopf, and Hui Huang. 2017. Analysis and Controlled Synthesis of Inhomogeneous Textures. *Computer Graphics Forum (Proc. of Eurographics 2017)* 36, 2 (2017).