Cubic Stylization

YIFAN REN* and YANG HENG, Shanghai Jiao Tong University, China

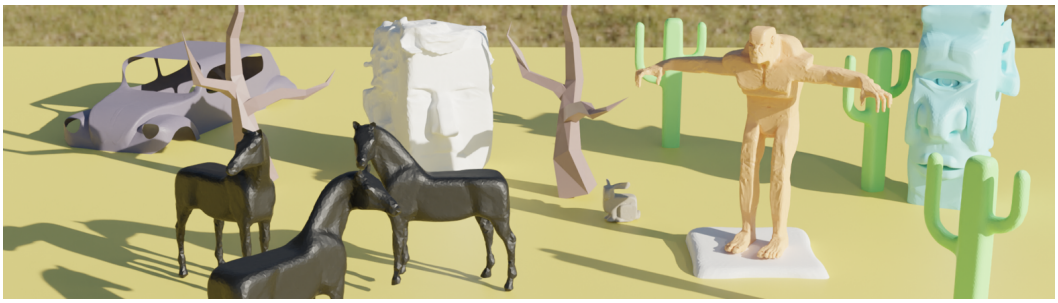

Fig. 1. Seattle Mariners at Spring Training, 2010.

We implement a state of the art 3D stylization algorithm[6] that can turn an input shape into the style of a cube while maintaining the content of the original shape. The key idea is to capture cubic style sculptures by the *as-rigidas-possible* energy with an l^1-regularization on rotated surface normals.Minimizing this energy naturally leads to a detail-preserving cubic geometry. This method can solve the problem efficiently without any mesh surgery.

Additional Key Words and Phrases: ADMM formulation, ARAP energy, cubic stylization

ACM Reference Format:
Yifan Ren and Yang Heng. 2022. Cubic Stylization. 1, 1 (June 2022), 5 pages.

1 INTRODUCTION

1.1 Background

Cubic style have been attracting artists' attention over centuries. However, apart from those image stylization filters and non-realistic rendering methods, direct stylization and modeling algorithm has received less attention. Our group aims to generate 3D cubic stylized shape and create cubic stylized scene. It is always hard to preserve geometry details while stylization because cubifying a shape leads to loss of details. We are going to use HSUEH-TI DEREK LIU's state of the art algorithm[6] as our primary resource.

1.2 Overview

During this project, firstly we figure out how to preserve geometry details while cubifying a shape and get fully understanding about cubifying. Then we implement LIU's algorithm[6] and write the core code by ourselves. For this project, we deliver a series of stylized shapes and scenes consisting of those shapes.

*Both authors contributed equally to this research.

Authors' address: Yifan Ren, ivan-ren@sjtu.edu.cn; Yang Heng, wnmMrrd@sjtu.edu.cn, Shanghai Jiao Tong University, Shanghai, China.

XXXX-XXXX/2022/6-ART $15.00

Besides, we add another feature, we can decide for which axis we do the cubifying operation, and for which axis we do not operate.

1.3 Related works

Our work has the similar motivations to a large body of work on image stylization[5], non-photorealistic rendering[3], and motion stylization[4].

Direct 3D stylization has been an important topic in computer graphics. Many other generative models also have been proposed for producing specific styles, without relying on identifying and transferring style components from other shapes. Such as collage art[2], voxel/lego art[7] and so on.

Many works deal with the question of how to deform shapes given modeling constraints. One of the most popular choices is the ARAP energy[8], which measures local rigidity of the surface and leads to detail-preserving deformations. We take advantage of the ARAP energy for detail preservation.

2 METHODOLOGY

The input of our method is a .obj file with all meshes triangulated, and it output a .obj file containing the cubified object of the input. Meanwhile, the geometric details will not change during the process.

We first denote an energy formula below:

$$\sum_{i \in V} \sum_{j \in \mathcal{N}(i)} \frac{w_{ij}}{2} ||R_i d_{ij} - \widetilde{d}_{ij}||_F^2 + \lambda a_i ||R_i \widehat{n}_i||_1$$

Now we explain the symbols in the formula:

- V is the set containing the 3D positions of all vertex of the input object and can be regarded as a $|V| \times 3$ matrix, and $\widetilde{V}$ is used to describe the vertexes' positions after deformation.
- $\mathcal{S}$ denotes the entire triangle mesh.
- $C(i)$ denotes the cell corresponding to the ith vertex.
- $\mathcal{N}(i)$ denotes the "spokes and rims" edges of the ith vertex.
- $w_i j$ is the cotangent weight.
- R_i is a 3×3 rotation matrix.
- $d_{ij} = [v_j - v_i]^T$ where v_i and v_j corresponds to the ith and jth row of V.
- $\widetilde{d}_{ij} = [\widetilde{v}_j - \widetilde{v}_i]^T$ where v_i and v_j corresponds to the ith and jth row of $\widetilde{V}$.
- a_i is the the barycentric area of vertex i.
- λ describes level of cubeness of the object.
- $\widehat{n}_i$ is the unit area-weighted normal vector of a vertex i.

The first term of the formula is ARAP energy, and the second term uses 1-norm to help maintain the cubified result.

Our main goal is to maintain a $\widetilde{V}$ and a set of $\{R_i\}$ such that:

$$\underset{\widetilde{V}, \{R_i\}}{minimize} \sum_{i \in V} \sum_{j \in \mathcal{N}(i)} \frac{w_{ij}}{2} ||R_i d_{ij} - \widetilde{d}_{ij}||_F^2 + \lambda a_i ||R_i \widehat{n}_i||_1$$

We adapt the standard local-global update strategy to achive our goal and the pseudo code is like this:

Input : A triangle mesh V, F
Output: Deformed vertex positions $\widetilde{\mathsf{V}}$

1. $\widetilde{\mathsf{V}} \leftarrow \mathsf{V}$
2. **while** *not converge* **do**
3. $\quad \mathsf{R} \leftarrow \textit{local-step}\,(\mathsf{V}, \widetilde{\mathsf{V}}, \lambda)$
4. $\quad \widetilde{\mathsf{V}} \leftarrow \textit{global-step}\,(\mathsf{R})$

Consider the maximum position change of the $\widetilde{V}$ as $v1$, the maximum position change from V to the newest $\widetilde{V}$ as $v2$. If $\frac{v1}{v2}$ is smaller than a specific value, the "while" process comes to the convergence.

2.1 Local-step

After $\widetilde{V}$ fixed, we can separately calculate the energy contributed by each vertex i and the goal turn out to find the optimal $R_i^\star$ for vertex i. So by simplification our local-step of i can be written as:

$$R_i^\star = \arg\min_{R_i} \frac{1}{2}||R_i D_i - \widetilde{D}_i||_{W_i}^2 + \lambda a_i ||R_i \widehat{n}_i||_1$$

where W is a diagonal matrix of cotangent weights and $W_{k,k} = w_{ij}$, and D_i and $\widetilde{D}_i$ is a $3 \times |\mathcal{N}(i)|$ and the kth cow of them respectively corresponds to d_{ij} and $\widetilde{d}_{ij}$(vertex j is the kth vertex in $\mathcal{N}(i)$). By adding a new variable z and set $z = R_i \widehat{n}_i$ we can rewrite the formula above as:

$$\underset{R_i, z}{minimize}\ \frac{1}{2}||R_i D_i - \widetilde{D}_i||_{W_i}^2 + \lambda a_i ||R_i \widehat{n}_i||_1$$

$$\text{subject to} R_i \widehat{n}_i - z = 0$$

After transposing the subjection the formula above can be regarded as the standard ADMM formulation[1]. We solve this local step using the scaled-form ADMM updates where y is the dual variable and u is the scaled variable with $u = \frac{1}{\rho} y$:

$$\mathsf{R}_i^{k+1} \leftarrow \arg\min_{\mathsf{R}_i \in SO(3)} \frac{1}{2}\|\mathsf{R}_i \mathsf{D}_i - \widetilde{\mathsf{D}}_i\|_{\mathsf{W}_i}^2 + \frac{\rho^k}{2}\|\mathsf{R}_i \hat{\mathsf{n}}_i - \mathsf{z}^k + \mathsf{u}^k\|_2^2$$

$$\mathsf{z}^{k+1} \leftarrow \arg\min_{\mathsf{z}} \lambda a_i \|\mathsf{z}\|_1 + \frac{\rho^k}{2}\|\mathsf{R}_i^{k+1} \hat{\mathsf{n}}_i - \mathsf{z} + \mathsf{u}^k\|_2^2$$

$$\tilde{\mathsf{u}}^{k+1} \leftarrow \mathsf{u}^k + \mathsf{R}_i^{k+1} \hat{\mathsf{n}}_i - \mathsf{z}^{k+1}$$

$$\rho^{k+1}, \mathsf{u}^{k+1} \leftarrow \textit{update}\,(\rho^k)$$

At the beginning of this local-step, we set u and z completely random value and the initial $\rho = 10^{-4}$ in our implementation.

2.1.1 Updating R_i^{k+1} and z^{k+1}. R_i is the rotation matrix, which is also a unitary matrix. Then the updating method of R_i is an instance of the orthogonal Procrustes:

$$\mathsf{R}_i^{k+1} \leftarrow \arg\max_{\mathsf{R}_i \in SO(3)} \mathrm{Tr}(\mathsf{R}_i \mathsf{M}_i)$$

$$\mathsf{M}_i = \begin{bmatrix} \mathsf{D}_i & \hat{\mathsf{n}}_i \end{bmatrix} \begin{bmatrix} \mathsf{W}_i & \\ & \rho^k \end{bmatrix} \begin{bmatrix} \widetilde{\mathsf{D}}_i^\top \\ (\mathsf{z}^k - \mathsf{u}^k)^\top \end{bmatrix}.$$

We can apply SVD on M_i that $M_i = \mathcal{U}_i \Sigma_i \mathcal{V}_i^T$ and the optimal R_i can be computed as: $R_i^{k+1} \leftarrow \mathcal{V}_i \mathcal{U}_i^T$. If $\det(R_i) \leq 0$, we can change the the sign of the column of $\mathcal{U}_i$ until $\det(R_i) > 0$(in the implementation we simply change the column 2).

The updating method of z is an instance of lasso problem and we solve it with a shrinkage step:

$$\mathbf{z}^{k+1} \leftarrow \mathcal{S}_{\lambda a_i/\rho^k}(\mathbf{R}_i^{k+1}\hat{\mathbf{n}}_i + \mathbf{u}^k)$$
$$\mathcal{S}_\kappa(\mathbf{x})_j = (1 - \kappa/|x_j|)_+ \, x_j$$

Here $x_+ = \max(0, x)$.

2.1.2 updating u, ρ and the stopping creteria. First simply follow the guide $\widetilde{u}^{k+1} \leftarrow u^k + R_i^{k+1}\widehat{n}_i - z^{k+1}$. Then with the equation $y = \rho u^{k+1}$ and y fixed, we only need to find out how the value of ρ changes and u^{k+1} changes from $\widetilde{u}^{k+1}$ in the opposite way.

First we introduce some symbols and values:

- $\epsilon^{abs} = 10^{-5}$, $\epsilon^{rel} = 10^{-3}$, $\mu = 10$, and $\tau^{incr} = \tau^{decr} = 2$.
- $s^k = -\rho(z^{k+1} - z^k)$.
- $r^k = -R_i * \widehat{n} + z$.
- $\epsilon^{pri} = \sqrt{3}\epsilon^{abs} + \epsilon^{rel} \max\{||R_i\widehat{n}_i||_2, ||z||_2\}$.
- $\epsilon^{dual} = \sqrt{3}\epsilon^{abs} + \epsilon^{rel}\rho||\widetilde{n}_i\widetilde{u}^{k+1T}||_\infty$.

And the value of ρ^{k+1} is below:

$$\rho^{k+1} := \begin{cases} \tau^{\text{incr}}\rho^k & \text{if } \|r^k\|_2 > \mu\|s^k\|_2 \\ \rho^k/\tau^{\text{decr}} & \text{if } \|s^k\|_2 > \mu\|r^k\|_2 \\ \rho^k & \text{otherwise,} \end{cases}$$

The interation stops when $||r||_2^k \leq \epsilon^{pri}$ and $||s||_2^k \leq \epsilon^{dual}$.

2.2 Global-step

In global step, we are going to find optimal vertex positions minimizing ARAP energy from given rotations with specific constraints.

The ARAP energy is defined as:

$$E(C_i, C_i') = \sum_{j \in \mathcal{N}(i)} w_{ij}(p_i' - p_j') - R_i(p_i - p_j)^2$$

.

In order to compute optimal vertex positions from given rotations, we compute the gradient of $E(S')$ with respect to the position p'.

$$\begin{aligned}\frac{\partial E(\mathcal{S}')}{\partial p_i'} &= \frac{\partial}{\partial p_i'}\Big(\sum_{j \in \mathcal{N}(i)} w_{ij}(p_i' - p_j') - R_i(p_i - p_j)^2 + \sum_{j \in \mathcal{N}(i)} w_{ji}(p_j' - p_i') - R_j(p_j - p_i)^2\Big) \\ &= \sum_{j \in \mathcal{N}(i)} 2w_{ij}((p_i' - p_j') - R_i(p_i - p_j)) - \sum_{j \in \mathcal{N}(i)} 2w_{ji}((p_j' - p_i') - R_j(p_j - p_i)) \\ &= \sum_{j \in \mathcal{N}(i)} 4w_{ij}((p_i' - p_j') - \frac{1}{2}(R_i + R_j)(p_i - p_j))\end{aligned}$$

Setting the partial derivatives to zero w.r.t each p_i' we arrive at the following sparse linear system of equations.

$$\sum_{j \in \mathcal{N}(i)} w_{ij}(p_i' - p_j') = \sum_{j \in \mathcal{N}(i)} \frac{1}{2}(R_i + R_j)(p_i - p_j)$$

We also need to incorporate the modeling constraints into this system, which fix some points to specific positions.

$$p_i' = c_i$$

3 PERFROMANCE

file name	size	$\lambda = 0.2$	$\lambda = 0.5$	$\lambda = 0.8$
bunny.obj	933.4kB	5.1s	2s	1.3s
human.obj	57.9kB	2.5s	0.8ss	0.3s
ogre.obj	3.4MB	12.7s	7.4s	4.5s

We can perform better if we can use implement the section 3.2 in Liu's paper and optimize the process to calculate R_i^{k+1}.

REFERENCES

[1] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. 2011. .

[2] Ran Gal, Olga Sorkine, Tiberiu Popa, Alla Sheffer, and Daniel Cohen-Or. 2007. 3D Collage: Expressive Non-Realistic Modeling. In *Proceedings of the 5th International Symposium on Non-Photorealistic Animation and Rendering* (San Diego, California) *(NPAR '07)*. Association for Computing Machinery, New York, NY, USA, 7–14. https://doi.org/10.1145/1274871.1274873

[3] Bruce Gooch and Amy Gooch. 2001. *Non-photorealistic rendering*. AK Peters/CRC Press.

[4] Aaron Hertzmann, Carol O'Sullivan, and Ken Perlin. 2009. Realistic Human Body Movement for Emotional Expressiveness. In *ACM SIGGRAPH 2009 Courses* (New Orleans, Louisiana) *(SIGGRAPH '09)*. Association for Computing Machinery, New York, NY, USA, Article 20, 27 pages. https://doi.org/10.1145/1667239.1667259

[5] Jan Eric Kyprianidis, John Collomosse, Tinghuai Wang, and Tobias Isenberg. 2013. State of the "Art": A Taxonomy of Artistic Stylization Techniques for Images and Video. *IEEE Transactions on Visualization and Computer Graphics* 19, 5 (2013), 866–885. https://doi.org/10.1109/TVCG.2012.160

[6] Hsueh-Ti Derek Liu and Alec Jacobson. 2019. Cubic Stylization. *ACM Transactions on Graphics* (2019).

[7] Sheng-Jie Luo, Yonghao Yue, Chun-Kai Huang, Yu-Huan Chung, Sei Imai, Tomoyuki Nishita, and Bing-Yu Chen. 2015. Legolization: Optimizing LEGO Designs. *ACM Trans. Graph.* 34, 6, Article 222 (oct 2015), 12 pages. https://doi.org/10.1145/2816795.2818091

[8] Olga Sorkine and Marc Alexa. 2007. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, Vol. 4. 109–116.