SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •
Special Focus

May 2012 Vol. 55 No. 5: 983–993
doi: 10.1007/s11432-012-4574-y

L_p shape deformation

GAO Lin[1]*, ZHANG GuoXin[1] & LAI YuKun[2]

[1]*Department of Computer Science and Technology, Tsinghua University, Beijing* 100084, *China;*
[2]*School of Computer Science and Informatics, Cardiff University, Wales* CF24 3AA, *UK*

Received October 24, 2011; accepted January 6, 2012; published online March 16, 2012

Abstract Shape deformation is a fundamental tool in geometric modeling. Existing methods consider preserving local details by minimizing some energy functional measuring local distortions in the L_2 norm. This strategy distributes distortions quite uniformly to all the vertices and penalizes outliers. However, there is no unique answer for a natural deformation as it depends on the nature of the objects. Inspired by recent sparse signal reconstruction work with non L_2 norm, we introduce general L_p norms to shape deformation; the positive parameter p provides the user with a flexible control over the distribution of unavoidable distortions. Compared with the traditional L_2 norm, using smaller p, distortions tend to be distributed to a sparse set of vertices, typically in feature regions, thus making most areas less distorted and structures better preserved. On the other hand, using larger p tends to distribute distortions more evenly across the whole model. This flexibility is often desirable as it mimics objects made up with different materials. By specifying varying p over the shape, more flexible control can be achieved. We demonstrate the effectiveness of the proposed algorithm with various examples.

Keywords shape deformation, L_p norm, geometric modeling

Citation Gao L, Zhang G X, Lai Y K. L_p shape deformation. Sci China Inf Sci, 2012, 55: 983–993, doi: 10.1007/s11432-012-4574-y

1 Introduction

The proliferation of digital geometry models nowadays makes digitized 3D objects, often represented as triangulated meshes, widely available. In addition to direct capture 3D shapes from real objects, a variety of geometric modeling tools have been developed. Surface deformation is a fundamental tool that produces altered shapes effectively. This has various applications including editing shapes to suit the needs, producing sequences of objects for animation and simulating the deformation of objects in VR systems [1] such as virtual surgery simulation systems [2]. Shape deformation has received a lot of attention in recent years. Since physically modeling geometric objects undergoing deformation is both difficult and computationally expensive, most algorithms focus on using geometric shape information alone. There is no single "correct" answer for surface deformation, due to the potential variable material natures. Many existing methods produce deformed objects by minimizing some energy functional measuring the usually unavoidable distortions incurred in the deformation process. To support general, large-scale deformation, energies need to be defined locally, as these properties tend to be well preserved

*Corresponding author (email: gaolinorange@gmail.com) after deformation. The L_2 norm is often used to combine local energies, often defined at some element (e.g. vertex) level. This strategy tends to spread out unavoidable distortions to all the vertices quite uniformly and penalize outliers. Although working reasonably well for a large set of models, this may not lead to the most "natural" deformation in practice. Distributions of unavoidable distortions reflect the material nature and the design intentions of the user, thus giving the user intuitive control is often desirable. In this work, inspired by recent advances in sparse signal reconstruction [3,4] and surface reconstruction [5], we introduce L_p norms for surface deformation instead of the traditional L_2 norm, in the energy functional combining local energies. More precisely, assuming the local energy associated with the ith element (e.g. vertex) is E_i, the overall energy E is defined in a more general manner as $E = \sum_i w_i \|E_i\|^p$, where p is a positive parameter to flexibly control the distributions of unavoidable distortions after the deformation. For practical use, we assume $p \geqslant 1$. $p = 2$ leads to the traditional L_2 norm. Using smaller p tends to distribute unavoidable distortions to a sparse set of vertices, often in feature regions. A typical case is $p = 1$ which reduces to the L_1 norm used in sparse shape reconstruction [5]. A brief theoretical explanation of the sparsity when $p = 1$ is given in Appendix. This has some nice features such as making most areas less distorted and structures better preserved. Using larger p tends to distribute distortions more evenly as outliers are more significantly penalized. Using our approach, a wide range of deformation results can be obtained, which gives the user effective and intuitive control over the deformation process. Results obtained with different p's may all be natural, mimicking objects made up with different materials. Our implementation is based on [6], which optimizes an energy functional for locally as-rigid-as-possible deformation. The idea could be used in other surface deformation framework as well. We show that using our L_p norm, the energy functional is convex and thus can be effectively optimized to the global minimum with iterative backtracking line search [7]. In Section 2, we review the most relevant work. The algorithm is described in detail in Section 3. Experimental results and discussions are presented in Section 4 and finally concluding remarks and future work are given in Section 5.

2 Related work

Shape deformation is an active research direction in recent decades. A large amount of research work has been carried out in the field. A complete survey is beyond the scope of this paper. Please refer to [8–10] for recent surveys of surface deformation techniques. Here we only review the most relevant work to this research.

The key principle of surface deformation algorithms is to produce visually plausible deformation and follow the deformation of actual physical objects. Surface based deformation can be modeled as an energy minimization problem with boundary constraints from user input (such as handle movements or specified local frames at certain positions). A natural approach is to deform the model according to the physical rules [11–13]. These physically based methods involve solving partial differential equations which are complicated and time consuming to achieve accurate deformation results. To improve the computational efficiency, Barbic et al. [14] proposed a large-scale deformation method by decomposing the deformable object into components.

Skeletons are used to deform the images/videos (e.g. [15]) and surface models [16–19]. These skeleton driven deformation methods need extra efforts to construct skeletons. Some works deform the shape by building the coarse cages encompassing the shape [20–24]. Such deformations may also be achieved cage-free, as demonstrated in [25] by using umbrella shaped cells constructed automatically. A new approach [26] has been proposed recently which allows the user to freely combine different types of handles to deform the object.

Another strategy to make intuitive deformation results is to preserve details and/or volumes after deformation. Local differential coordinate are used to encode local details and recover them after deformation. These methods include rotation invariant coordinate for better handling rotations [27], Laplacian coordinates [28], Poisson-based gradient field [29], and iterative dual Laplacian approach for improved results [30]. Volumetric Laplacian constructed in the interior of the shape is proposed to better preserve the volume [31,32]. A subspace technique is used for efficiently optimizing the nonlinear energy first at the coarser mesh and uses 3D mean value coordinates [20] for interpolation on the original mesh.

Rigidity is an important principle in deformation that is well studied. Terzopoulos et al. [11] formulate a shell energy to measure the distortions between the input and the deformed models.

$$E(S, S') = \int_{\Omega} (k_s \|I - I'\|^2 + k_b \|II - II'\|^2) \mathrm{d}u\mathrm{d}v, \tag{1}$$

where S and S' are the surfaces before and after deformation, I, II are the first and second fundamental forms before deformation, and I', II' are corresponding fundamental forms after deformation; they are used to measure the shearing and bending incurred by the deformation. k_s and k_b are two coefficients to balance the terms. The energy $E(S, S')$ reaches zero only for completely rigid transformations. Practical solutions often involve unavoidable distortions, and thus only achieves the minimizer of E. Preserving local as-rigid-as-possible is practically useful as this preserves geometric shape and features while allowing for sufficient flexibility for deformation. Sorkine et al. [6] estimate the rigid transformations of local cells and collect the transformations to deform the whole model. The principle of as rigid as possible deformation has also been applied to shape interpolation [33] and shape manipulation [34]. Such works based on the as-rigid-as-possible principle have a similar framework. They all estimate local rigid transforms of geometric elements (e.g. triangle faces), and then build a global energy formulation based on L_2 norm. L_1 norm was recently used to reconstruct point set surfaces, and has achieved sparse optimization with improved feature and structure preservation [5]. Bougleux et al. [35] recently used similar formulation in p-Laplacian, and applied this for applications such as mesh denoising.

Differential domain methods and local as-rigid-as-possible methods mentioned above are mainly concerned with surface deformation with uniform material properties. Some previous works consider non-uniform materials. Popa et al. [36] use a painting-like interface to specify the material properties, which are then used to guide the propagation of transformations. Our method does not need user interactions to specify material properties. Instead we use a single parameter to control the distribution of deformation distortions. Thus our approach provides more flexibility than most traditional deformation methods while having less burden on user efforts. Some research work explicitly considers man-made models. They are often self-similar and made up of piecewise quasi-rigid components. Gal et al. [37] extract feature curves according to the analysis of the models and manipulate the models through editing these curves. In this paper, we introduce general L_p norm in the energy formulation for surface deformation, leading to flexible and intuitive control of residual error distributions.

3 Algorithm

In this section, we first describe our L_p surface deformation formulation in the as-rigid-as-possible framework. We then show that the resulting energy functional is convex, leading to an effective optimization algorithm.

3.1 L_p surface deformation formulation

We denote the input triangle mesh by S which contains n vertices. For each vertex v_i, one-ring neighbors form a set, denoted by N_i. We use $p_i \in \mathbb{R}^3$ to represent the position of v_i. The surface is deformed into S' with the same connectivity and positions changed to p'_i. To define local rigidity, similar to [6], for each vertex v_i, a cell C_i is formed which covers 1-ring neighbors N_i. This definition is sufficiently local and involves overlaps between cells which are essential for smoothness of transforms between cells. The local energy between the cell C_i and its deformation C'_i is defined similarly to [6]

$$E(C_i, C'_i) = \sqrt{\sum_{j \in N_i} w_{ij} \|(p'_i - p'_j) - R_i(p_i - p_j)\|^2}. \tag{2}$$

Here, R_i represents a 3×3 rotation matrix that transforms locally from C_i to C'_i. The weight w_{ij} can be chosen as the cotangent weight $w_{ij} = \frac{1}{2}(\cot\alpha_{ij} + \cot\beta_{ij})$ [38] to take mesh discretization into account, where α_{ij} and β_{ij} are angles opposite to the edge v_iv_j. To form the overall energy, we propose to use L_p norm instead of the traditional L_2 norm:

$$E(S,S') = \sum_{i=1}^{n} E(C_i,C_i')^p = \sum_{i=1}^{n}\left\{\sum_{j\in N_i} w_{ij}\|(p_i'-p_j') - R_i(p_i-p_j)\|^2\right\}^{\frac{p}{2}}. \tag{3}$$

To minimize the nonlinear energy $E(S,S')$, an iterative algorithm is used. From an initial guess which can take either the input mesh or the mesh obtained from relatively simple deformation algorithms, rotation matrices R_i and positions p_i' are optimized in turn. This process guarantees convergence as the energy is monotonically decreasing. The optimal rotation matrix R_i for fixed positions p_i' can be solved independently for each vertex v_i [5]. Denote by S_i the covariance matrix of C_i. $S_i = \sum_{j\in N_i} w_{ij}(p_i - p_j)(p_i' - p_j')^{\mathrm{T}}$. Singular decomposition of S_i satisfies $S_i = U_i\Sigma_iV_i^{\mathrm{T}}$, and then R_i can be obtained as $R_i = V_iU_i^{\mathrm{T}}$, subject to changing the sign of the column of U_i corresponding to the minimal singular value, to make $\det(R_i) > 0$. We will give the details of finding the optimal positions p_i' for given R_i in the next subsection.

3.2 An effective convex optimization

To find the optimal positions p_i', we first prove that the energy E w.r.t. p_i' is convex. The position p_i at each vertex v_i is three dimensional, and denoted by p_{ix}, p_{iy}, p_{iz}. We take P to represent a vector collecting all of these coordinates, i.e. $P = [p_{1x}, p_{1y}, p_{1z}, \ldots, p_{nx}, p_{ny}, p_{nz}]^{\mathrm{T}}$. Since R_i's are fixed. $R_i(p_i - p_j)$ is constant vector which can be expressed as the difference of two vectors.

Eq. (2) can be rewritten as

$$E(C_i,C_i') = \sqrt{\sum_{j\in N_i} w_{ij}\|a_{ij}^{\mathrm{T}}(P-d)\|^2}, \tag{4}$$

$$E(C_i,C_i') = \sqrt{(P-d)^{\mathrm{T}}\bigg(\sum_{j\in N_i} w_{ij}^2 a_{ij}^{\mathrm{T}} a_{ij}\bigg)(P-d)}, \tag{5}$$

where a_{ij} is a vector of length $3n$. $a_{ij}(k) = 1$, for $k = 3i-2, 3i-1, 3i$ and $a_{ij}(k) = -1$, for $k = 3j-2, 3j-1, 3j$. For any other k, $a_{ij}(k) = 0$. $a_{ij}^{\mathrm{T}}d = R_i(p_i - p_j)$. To show that the energy defined in Eq. (3) is convex, since $\sum_{j\in N_i} w_{ij}^2 a_{ij}^{\mathrm{T}} a_{ij}$ is symmetric semi-positive definite, it suffices to prove that this holds for a more generalized function, namely for a symmetric semi-positive definite matrix A, and $p \geqslant 1$, $c \geqslant 0$, the form $f = (x^{\mathrm{T}}Ax + c)^{\frac{p}{2}}$ is convex. The gradient ∇f and Hessian matrix $H(f)$ of f can be calculated as

$$\nabla f = p(Ax)(x^{\mathrm{T}}Ax + c)^{\frac{p-2}{2}}, \tag{6}$$

$$H(f) = p(x^{\mathrm{T}}Ax + c)^{\frac{p-4}{2}} \cdot [A(x^{\mathrm{T}}Ax + c) + (p-2)Axx^{\mathrm{T}}A]. \tag{7}$$

As $p \geqslant 1$, it suffices to show that for a general vector y,

$$(y^{\mathrm{T}}Ay)(x^{\mathrm{T}}Ax) \geqslant (y^{\mathrm{T}}Axx^{\mathrm{T}}Ay) = (y^{\mathrm{T}}Ax)^2. \tag{8}$$

Since A is symmetric semi-positive definite, there exists a unique symmetric semi-positive definite matrix B, such that $B \cdot B = A$. We use $\sqrt{A}$ to represent the matrix B, which can be calculated using eigen decomposition. In this case, Eq. (8) actually holds due to the following inequality

$$[(\sqrt{A}y)^{\mathrm{T}}(\sqrt{A}y)][(\sqrt{A}x)^{\mathrm{T}}(\sqrt{A}x)] \geqslant [(\sqrt{A}y)^{\mathrm{T}}(\sqrt{A}x)]^2. \tag{9}$$

If $c = 0$, the Hessian matrix $H(f)$ may not exist when $Ax = 0$. This situation can be verified by showing that f is convex when restricted to any line. More specifically, assume $f = g(t)$, $t \in \mathbb{R}$, then $\exists t, g(t) = 0$ if and only if $g(t) \equiv 0$, so either g'' exists for every t, or $g(t) \equiv 0$. In both cases, our conclusion is proved.

Since the energy is convex, we can find the global minimum effectively by an iterative backtracking line search [7]. Starting from $\tilde{p}'_i$, we compute the descent direction

$$\Delta\tilde{p}'_i = -\frac{\partial E}{\partial p'_i}\bigg|p'_i = \tilde{p}'_i, \tag{10}$$

as the negative gradient direction. Assume all the $\tilde{p}'_i$ form $\tilde{p}'$ and all the $\Delta\tilde{p}'_i$ form $\Delta\tilde{p}'$. The step size t is initialized to 1 and can be obtained by repeatedly multiplying t by a constant β, until $E(\tilde{p}' + t\Delta\tilde{p}') \leqslant E(\tilde{p}') + \alpha t \nabla E(p')^{\mathrm{T}}\Delta\tilde{p}'$. We choose $\alpha = 0.3$ and $\beta = 0.5$ for our experiments. The updated position can then be obtained as $\tilde{p}' + t\Delta\tilde{p}'$. This process repeats until convergence happens.

3.3 More efficient solution of the L_1 problem

Using L_1 norm (the special case with $p = 1$) has a few advantages. The distortions tend to concentrate on a sparse set of vertices, leading to generally well preserved shapes after deformation. The problem can be converted to the dual form of conic programming as follows. Based on Eq. (3), we introduce variables $T_1, T_2, \ldots, T_n$, and these variables along with optimized positions p'_i are unknown variables to optimize. We need to maximize $\sum_{i=1}^{n}(-T_i)$, subject to the constraints

$$T_i \geqslant \sqrt{\sum_{j\in N_i} \|\sqrt{w_{ij}}((p'_i - p'_j) - R_i(p_i - p_j))\|^2}. \tag{11}$$

This can be effectively solved using SDPT3 (an open source conic programming solver) [39].

4 Experimental results

We carried out our experiments on a desktop computer with 2× Quad 2.27 GHz CPUs. Our current implementation has not been optimized for multi-core CPUs. In all the examples, we use blue dots to indicate the handles used to control the deformation, and use color coding to show the distortion distributions incurred by the deformation ($E(C_i, C'_i)$ for each vertex v_i), where increasing distortions are represented using colors from blue to red. We use conic programming described in Subsection 3.3 to solve L_1 problem, the method in [6] to solve L_2 problem and backtracking line search described in 3.2 to solve other L_p problems. For all the examples, we initialize the deformed mesh with naive Laplacian deformation (as in [28] without rotation estimation). Assuming for each vertex v_i the position change after each iteration is Δp_i, the average displacement is defined as $\bar{d} = \sqrt{\frac{\sum_{i=1}^{n}\|\Delta p_i\|^2}{n}}$. The terminating condition of convergence is indicated by $\bar{d} < \varepsilon_C$ (for conic programming) and $\bar{d} < \varepsilon_L$ (for line search) respectively. Different thresholds are used because these two optimization methods tend to update the positions differently. We have found $\varepsilon_C = 0.01$ and $\varepsilon_L = 0.00005$ work well in practice and these same parameters are used for all the examples in the paper. Detailed statistics of iteration numbers and running times are given in Table 1.

As shown in Figure 1, the energy consistently decreases with more iterations. Although line search involves much more iterations, each iteration takes less time. From the experiments, using larger p, the convergence is likely to be much faster. For this example, the L_6 norm actually takes less time than the traditional L_2 norm. The typical L_2 norm is only a special case. The histograms of the cell energies using L_1 norm and L_2 norm are shown in Figure 2. This verifies the deformation results in Figure 3. The energy of most cells is small for in L_1 norm since distortions in most regions are small. On the contrary the energy of cells distributes much more uniformly when L_2 norm is used. Figure 3 shows the results of translating the handles to fold the sheet, using L_1, $L_{1.5}$, L_2 and L_6 norms, respectively. All of these results can be natural, depending on the material of the sheet. It is clear that when smaller p (e.g. $p = 1$) is used, the unavoidable distortions are highly concentrated on certain regions. Another extremity is when large p (e.g. $p = 6$) is used, the distribution of distortions are rather uniform. With this more

Table 1 Statistics of running times[a)]

Example	N_V	N_F	N_1	T_1	N_2	T_2	$N_{1.5}$	$T_{1.5}$	N_6	T_6
Sheet (Figure 3)	3962	7920	19	187.141	50	28.002	1496	658.848	32	15.974
Bar (Figure 4)	370	736	6	3.770	9	1.133	92	5.739	12	0.760
Block (Figure 5)	2132	4272	21	144.153	17	6.109	–	–	–	–
Dinopet (Figure 6)	2251	4498	3	15.539	2	2.005	–	–	–	–
Rocker arm (Figure 7)	2108	4216	3	18.454	3	2.000	–	–	–	–

a) N_V: the number of vertices; N_F: the number of faces; N_1, N_2, $N_{1.5}$, N_6: the number of iterations used before convergence for L_1, L_2, $L_{1.5}$, L_6 norms respectively; T_1, T_2, $T_{1.5}$, T_6: corresponding running times in seconds.

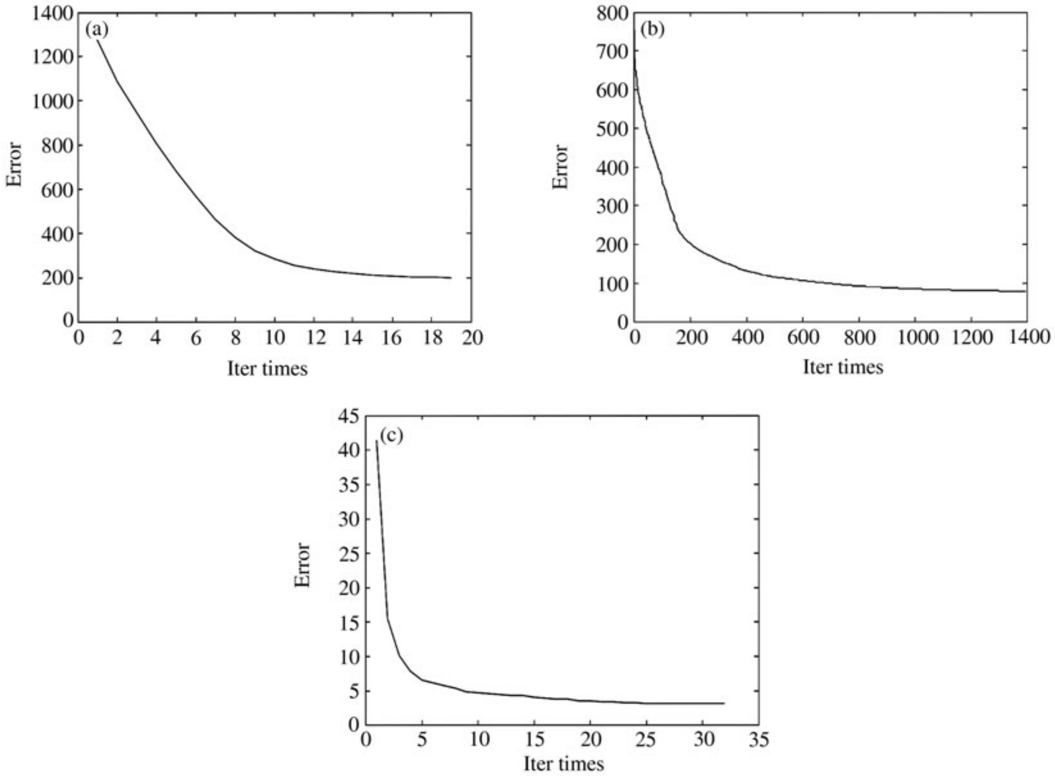

Figure 1 Energy decreases with iterations, using (a) L_1 norm; (b) $L_{1.5}$ norm and (c) L_6 norm.

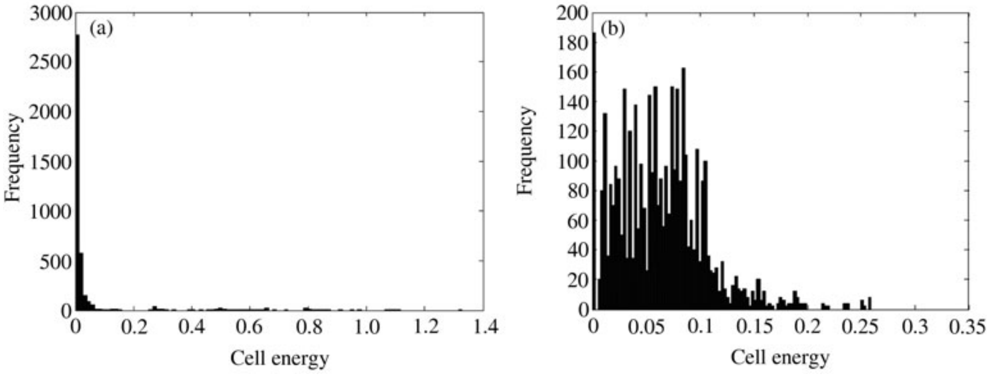

Figure 2 Histograms of cell energies obtained with L_1 norm (a) and L_2 norm (b).

generalized L_p surface deformation, the user can easily obtain these results by changing only a single parameter. Figure 4 shows another example involving rotations of handles with different L_p deformations. The bar is twisted 90 degrees in the middle while keeping both ends fixed. With increasing p, the distortions distribute more evenly over the whole deformed model.

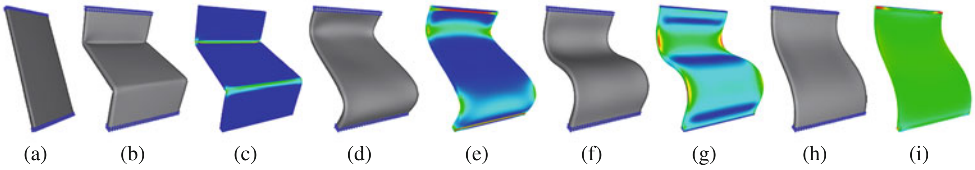

Figure 3 Results of L_p surface deformations. (a) The input surface; (b)(d)(f)(h) the results with L_1, $L_{1.5}$, L_2 and L_6 norms respectively; (c)(e)(g)(i) color coded distortion distributions of (b)(d)(f)(h). Distortions increase from blue to red.

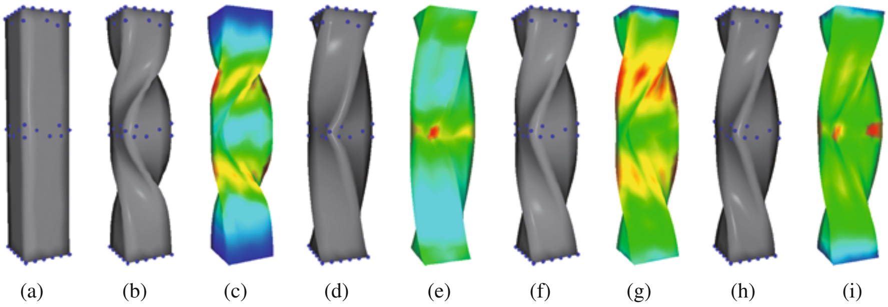

Figure 4 Results of L_p surface deformations. (a) The input surface; (b)(d)(f)(h) the results with L_1, $L_{1.5}$, L_2 and L_6 norms respectively; (c)(e)(g)(i) color coded distortion distributions of (b)(d)(f)(h). Distortions increase from blue to red.

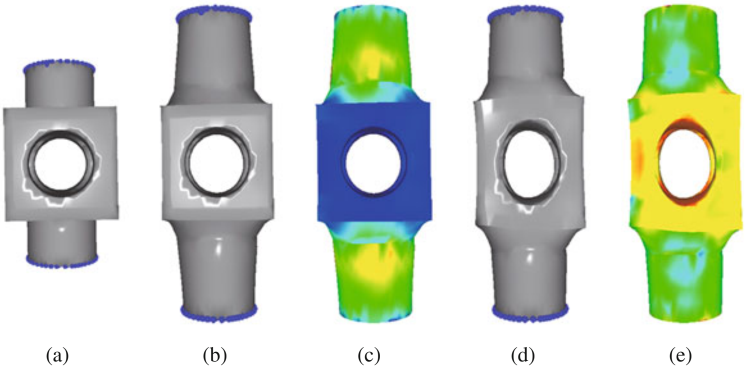

Figure 5 Comparison of deformation results of the input model (a) with L_1 (b)(c) and L_2 (d)(e) norms. Color coded results represent increasing distortions from blue to red.

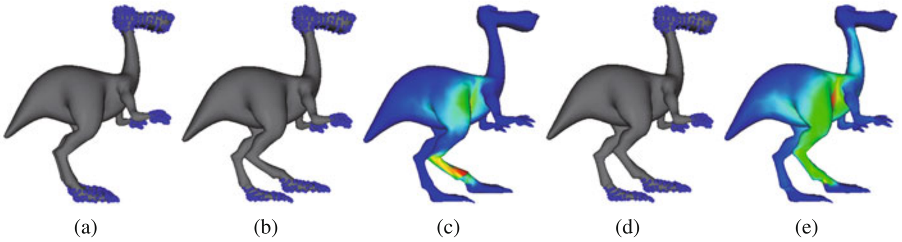

Figure 6 Comparison of deformation results obtained with L_1 and L_2 norms. (a) Input model; (b)(d) results obtained with L_1 and L_2 norms respectively; (c)(e) corresponding color coded distortions of (b)(d). Distortions increase from blue to red.

L_1 norm is a special case of our method which is particularly useful for concentrating distortions to a sparse set of areas. In Figures 5, 6 and 7, we compare the results with L_1 norm and the traditional L_2 norm. Since the distortions are concentrated, the deformation will be less likely to affect regions far away from the handles (as shown e.g. in Figure 6). When objects are stretched in Figures 5 and 7, since most areas have less distortions, the structures and features can be much better preserved (e.g. circular and

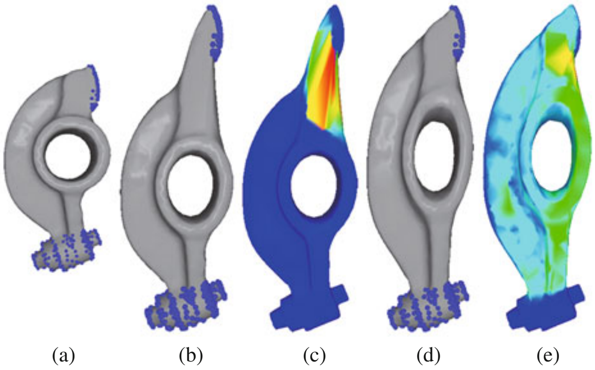

Figure 7 Comparison of deformation results obtained with L_1 and L_2 norms. (a) Input model; (b)(d) results obtained with L_1 and L_2 norms respectively; (c)(e) corresponding color coded distortions of (b)(d). Distortions increase from blue to red.

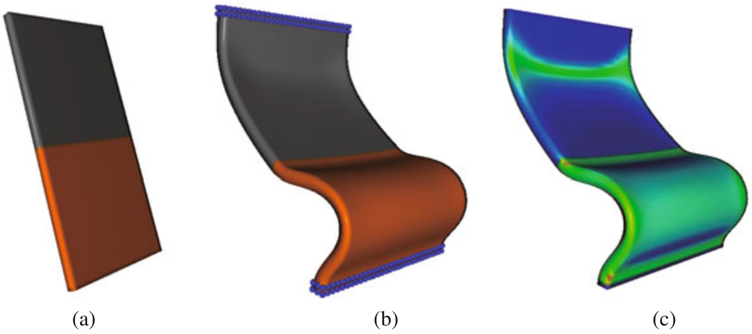

Figure 8 Deformation result with mixed L_1 and L_2 norms. (a) The distribution of L_1 norm and L_2 norm on the surface model; (b) deformation result with L_1 and L_2 norms; (c) corresponding color coded distortions of (b).

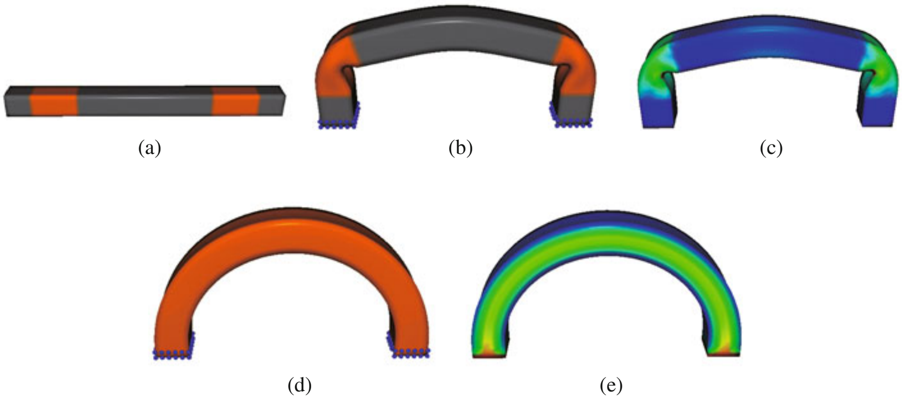

Figure 9 Deformation results with mixed L_1 and L_2 norms. (a) The distribution of L_1 norm and L_2 norm on the surface model; (b) deformation result with L_1 and L_2 norms; (d) deformation result with L_2 norm; (c)(e) corresponding color coded distortions of (b)(d).

cubic shapes). Our method can be generalized to distribute different norms over the surface model. This simulates objects composed of multiple materials. Examples are shown in Figures 8 and 9, where L_2 norm is used in the orange region and L_1 norm elsewhere. These models deform differently due to the variation of norms. The deformation results and the energy distribution within the same region are consistent with those using L_1 norm and L_2 norm respectively (e.g. see Figure 3). If the changes in the surface deformation process are relatively simple, only a small number of iterations are needed before convergence. This shows the potential application of our approach in interactive editing. Current unoptimized implementation takes under 20 seconds (for 'dinopet' for example). We expect to explore potential speed-up in the future, as detailed in the next section.

5 Conclusions and future work

In this paper, we propose a novel surface deformation approach that optimizes energy functional based on general L_p norms. The extra parameter p provides the user with intuitive and flexible control over the deformation process. Continuous variations of results can be obtained by simply changing a single parameter. We have demonstrated that the effects of different p's can be well anticipated. Using smaller p (e.g. L_1 norm) makes distortions well concentrated on a sparse set of vertices, producing results with most areas less distorted and structures better preserved. Larger p on the other hand promotes even distribution of distortions. This flexibility mimics deforming objects with different material natures. A major limitation is that our method is relatively slow, due to its nonlinear nature. We would like to explore potential techniques to speed up the computation, including subspace technique and parallelism. The optimization currently used such as iterative line search can be well parallelized using either multi-core CPUs or the GPU, which can potentially improve the performance quite significantly. With such further development, interactive performance is possible to achieve as fewer iterations are often needed for relatively small changes in interactive editing. In our current approach, we assign different norms manually to simulate different material properties. In certain cases, where material stiffness is related to geometric properties (for example, joints are more flexible than rigid components), it is possible to develop an automatic algorithm to distribute norms over the surface based on the geometry. This approach can also be extended to other techniques such as content-aware model resizing [40]. Using the L_p norm in shape deformation is general to be incorporated in other shape deformation frameworks; we expect to explore this in the future.

Acknowledgements

This work was supported by National Basic Research Project of China (Grant No. 2011CB302203), Natural Science Foundation of China (Grant No. 61120106007). The third author is supported by Engineering and Physical Sciences Research Council (Grant No. EP/I000100/1).

References

1 Böttcher G, Allerkamp D, Wolter F E. Multi-rate coupling of physical simulations for haptic interaction with deformable objects. Vis Comput, 2010, 26: 903–914

2 Chang J, Yang X S, Pan J J, et al. A fast hybrid computation model for rectum deformation. Vis Comput, 2011, 27: 97–107

3 Candes E J, Romberg J, Tao T. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. IEEE Trans Inf Theory, 2006, 52: 489–509

4 Donoho D L, Elad M, Temlyakov V N. Stable recovery of sparse overcomplete representations in the presence of noise. IEEE Trans Inf Theory, 2006, 52: 6–18

5 Avron H, Sharf A, Greif C, et al. ℓ_1-Sparse reconstruction of sharp point set surfaces. ACM Trans Graph, 2010, 29: 135

6 Sorkine O, Alexa M. As-rigid-as-possible surface modeling. In: Proceedings of 5th Eurographics Symposium on Geometry Processing (SGP '07), Barcelona, 2007. 109–116

7 Boyd S, Vandenberghe L. Convex Optimization. Cambridge: Cambridge University Press, 2004

8 Botsch M, Sorkine O. On linear variational surface deformation methods. IEEE Trans Vis Comput Graph, 2008, 14: 213–230

9 Cohen-Or D. Space deformations, surface deformations and the opportunities in-between. J Comput Sci Tech, 2009, 24: 2–5

10 Gain J, Bechmann D. A survey of spatial deformation from a user-centered perspective. ACM Trans Graph, 2008, 27: 107

11 Terzopoulos D, Platt J, Barr A, et al. Elastically deformable models. In: Proceedings of the 14th Annual Conference

on Computer Graphics and Interactive Techniques(SIGGRAPH '87), Anaheim, 1987. 205–214

12 James D L, Pai D K. Artdefo: accurate real time deformable objects. In: Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques(SIGGRAPH '99), Los Angeles, 1999. 65–72

13 Nealen A, Mueller M, Keiser R, et al. Physically based deformable models in computer graphics. Comput Graph Forum, 2006, 25: 809–836

14 Barbic J, Zhao Y L. Real-time large-deformation substructuring. ACM Trans Graph, 2011, 30: 91

15 Shen Y, Ma L Z, Liu H. An MLS-based cartoon deformation. Visual Comput, 2010, 26: 1229–1239

16 Lewis J P, Cordner M, Fong N. Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation. In: Proceedings of 27th Annual Conference on Computer Graphics and Interactive Techniques(SIGGRAPH '00), New Orleans, 2000. 165–172

17 Yan H B, Hu S M, Martin R R, et al. Shape deformation using a skeleton to drive simplex transformations. IEEE Trans Vis Comput Graph, 2008, 14: 693–706

18 Jacobson A, Sorkine O. Stretchable and Twistable bones for skeletal shape deformation. ACM Trans Graph, 2011, 30: 165

19 Kim B U, Feng W W, Yu Y Z. Real-time data driven deformation with affine bones. Vis Comput, 2010, 26: 487–495

20 Ju T, Schaefer S, Warren J. Mean value coordinates for closed triangular meshes. ACM Trans Graph, 2005, 24: 561–566

21 Joshi P, Meyer M, DeRose T, et al. Harmonic coordinates for character articulation. ACM Trans Graph, 2007, 26: 71

22 Lipman Y, Kopf J, Cohen-Or D, et al. GPU-assisted positive mean value coordinates for mesh deformations. In: Proceedings of Eurographics Symposium on Geometry Processing (SGP' 07), Barcelona, 2007. 117–123

23 Ju T, Zhou Q Y, Panne M V D, et al. Reusable skinning templates using cage-based deformations. ACM Trans Graph, 2008, 27: 122

24 Lipman Y, Levin D, Cohen-Or D. Green coordinates. ACM Trans Graph, 2008, 27: 78

25 Li Z, Levin D, Deng Z J. Cage-free local deformations using green coordinates. Vis Comput, 2010, 26: 1027–1036

26 Jacobson A, Baran I, Popovis J, et al. Bounded biharmonic weights for real-time deformation. ACM Trans Graph, 2011, 30: 78

27 Lipman Y, Sorkine O, Levin D, et al. Linear rotation-invariant coordinates for meshes. ACM Trans Graph, 2005, 24: 479–487

28 Sorkine O, Cohen-Or D, Lipman Y, et al. Laplacian surface editing. In: Proceedings of Eurographics Symposium on Geometry Processing (SGP '04), Nice, 2004. 175–184

29 Yu Y Z, Zhou K, Xu D, et al. Mesh editing with poisson-based gradient field manipulation. ACM Trans Graph, 2004, 23: 644–651

30 Au O K C, Tai C L, Liu L, et al. Dual laplacian editing for meshes. IEEE Trans Vis Comput Graph, 2006, 12: 386–395

31 Liao S H, Tong R F, Dong J X, et al. Gradient field based inhomogeneous volumetric mesh deformation for maxillofacial surgery simulation. Comput Graph, 2009, 33: 424–432

32 Liao S H, Tong R F, Geng J P , et al. Inhomogeneous volumetric laplacian deformation for rhinoplasty planning and simulation system. Comput Anim Virt Worlds, 2010, 21: 331–341

33 Alexa M, Cohen-Or D, Levin D. As-rigid-as-possible shape interpolation. In: Proceedings of 27th Annual Conference on Computer Graphics and Interactive Techniques(SIGGRAPH '00), New Orleans, 2000. 157–164

34 Igarashi T, Moscovich T, Hughes J F. As-rigid-as-possible shape manipulation. ACM Trans Graph, 2005, 24: 1134–1141

35 Bougleux S. Local and nonlocal discrete regularization on weighted graphs for image and mesh processing. Int J Comput Vis, 2009, 84: 220–236

36 Popa T, Julius D, Sheffer A. Material-aware mesh deformations. In: Proceedings of the IEEE International Conference on Shape Modeling and Applications 2006(SMI '06), Matsushima, 2006. 22–30

37 Gal R, Sorkine O, Mitra N J, et al. iWIRES: An analyze-and-edit approach to shape manipulation. ACM Trans Graph, 2009, 28: 33

38 Meyer M, Desbrun M, Schröder P, et al. Discrete differential-geometry operators for triangulated 2-manifolds. Vis Math, 2002, 3: 34–57

39 Toh K C, Todd M J, Tutuncu R H. SDPT3 Version 4.0 - A MATLAB Software for Semidefinite-Quadratic-Linear Programming, 2009

40 Chen L, Meng X X. Anisotropic resizing and deformation preserving geometric textures. Sci China Inf Sci, 2010, 53: 2441–2451

Appendix

Some discussion about the sparsity of L_p optimization when $p=1$ is given here. The optimization problem can be formulated as

$$\min \|Ax+b\|_p, \tag{A1}$$

where A is a matrix of size $m\times n$, x is a vector of length n to be optimized, b is a vector of length m, $\|\cdot\|_p$ is L_p norm with $p=1$. Without loss of generality, we assume $\|b\|=1$. We define $u=Ax+b$, so $u_i=\sum_{j=1}^{n}a_{ij}x_j+b_j$, where u_i, x_j and b_j are elements of u, x and b respectively and a_{ij} is an element of matrix A. The optimization problem (Eq. (A1)) is equivalent to $\min\sum_{i=1}^{m}|u_i|$. Assume S is the image space of A, so rank(S)=rank(A) $=r_s$. Given these definitions, we can reformulate the optimization problem in Eq. (A1) as:

$$\min \|u\|_p, \tag{A2}$$

$$\text{s.t.} \quad u\in S+b. \tag{A3}$$

This is equivalent to

$$\max t, \tag{A4}$$

$$\text{s.t.} \quad u\in S+tb, \quad \|u\|_p=1. \tag{A5}$$

Suppose the optimal solution of Eq. (A4) is $\bar{u}$ and $\bar{t}$, with $\bar{t}>0$. It can be easily verified that $\tilde{u}=\bar{u}/\bar{t}$ is also the optimal solution of Eq. (A2). We further define C_k as the set of length m vectors with k non-zero elements. We have the following lemmas:

Lemma 1. When $k<m-1-r_s$, we have $m_B\{b|\exists t\geqslant 0, s\in S, b\in B, s+tb\in C_k\}=0$. B is the $(m-1)$-dimension unit hyper sphere and m_B is the measure on B.

Lemma 2. When $p=1$, $C_k\subseteq$ convex(C_{k-1}), where convex$(\cdot)$ is the convex set operator.

We give a constructive proof of Lemma 2. For any $u\in C_k$, without loss of generality we assume $u_1,u_2,\ldots,u_k\neq 0$. We take k elements $\mathbf{s}_i$ $(i=1,2,\ldots,k)$ from C_{k-1}, defined as:

$$\begin{aligned} s_i=\{&u_1(1-|u_i|^p)^{-1/p}, u_2(1-|u_i|^p)^{-1/p}\ldots u_{i-1}(1-|u_i|^p)^{-1/p}, 0,\\ &u_{i+1}(1-|u_i|^p)^{-1/p}\ldots u_k(1-|u_i|^p)^{-1/p}, 0\ldots 0\}.\end{aligned} \tag{A6}$$

The ith element of s_i is zero. $(k-1)$ non-zero elements of s_i are given as $u_j(1-|u_i|^p)^{-1/p}$, for any $1\leqslant j\leqslant k, j\neq i$. Given weights $w_i=(1-|u_i|^p)^{1/p}/(k-1)$, when $p=1$, we have $\sum_{i=1}^{k}w_i=1$ and $\sum_{i=1}^{k}w_is_i=u$.

Based on these two lemmas, we can obtain the following conclusion: the probability of the solution to the optimization problem in Eq. (A1) with $(m-1-r_s)$ non-zero elements is 1.