Shape Deformation via Interior RBF

Zohar Levi and David Levin

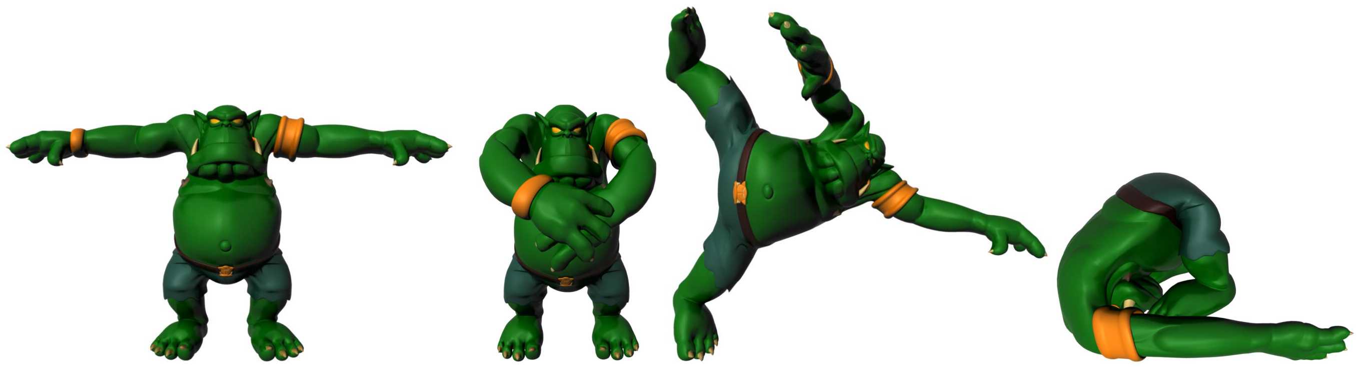


Figure 1: Troll. (b) IRBF deformation with interior distances.

Abstract—We present a new framework for real-time shape deformation with local shape preservation and volume control. Given a 3D object, in any form, one would like to manipulate the object using convenient handles, so that the resulting shape is a natural variation of the given object. It is also important that the deformation is controlled, thereby enabling localized changes that do not influence nearby branches. For example, given a horse model, a movement of one of its hooves should not affect the other hooves. Another goal is the minimization of local shape distortion throughout the object. The first ingredient of our method is the use of Interior Radial Basis Functions (IRBF), where the functions are radial with respect to interior distances within the object. The second important ingredient is the reduction of local distortions by minimizing the distortion of a set of spheres placed within the object. Our method achieves the goals of convenient shape manipulation and local influence property, and improves the latest state-of-the-art cage-based methods by replacing the cage with the more flexible IRBF centers. The latter enables extra flexibility and fully automated construction, as well as simpler formulation. We also suggest the IRBF interpolation method that can extend any surface mapping to the whole subspace in a shape-aware manner.

Index Terms—Space deformation, IRBF

1 Introduction

Shape deformation techniques are the cornerstones of computer animation used in computer games and films. One popular family of shape deformation techniques are the space deformation methods, which define a mapping from a source region to a target region within the Euclidean space, and provide the user a control structure that enables intuitive control over the deformation. Since the mapping is defined for the subspace, any type of object embedded in the ambient space can be implicitly deformed. The most common type of models used in the industry are multi-component models, where each part of a character is modeled independently from a mesh or a NURBS surface. For example, the troll in Fig. 1 consists of a body, a head, tusks, bracelets, an earring, and a belt. This makes space deformation methods attractive for this type of models, as well as for volumetric data, a polygon soup, or even a point cloud. In this aspect, space deformation methods have a clear advantage over direct deformation methods [4, 5, 2], which require a single connected mesh to manipulate directly. Another advantage is that space deformation methods usually optimize a set of controls, which is usually smaller and independent from the deformed object, and thus exhibit better performance than direct deformation methods.

Sederberg and Perry [6] introduced space deformation using a control lattice. However, the lattice proved to be too rigid for controlling an articulated object, and often several overlapping lattices are required, which makes the control cumbersome. Other control structures have been suggested, but the most notable one is the cage-based method, which was introduced in [7]. A cage is a polyhedron that encloses the object in a tight fashion, and the deformation is controlled by maneuvering the cage. The cage needs to be relatively simple (coarse), so that the user is able to edit the cage vertex-by-vertex in order to control the deformation. Although an improvement over previous methods, cage editing is still a tedious task. Moreover, for the user to have the desired flexibility, the cage is required to enclose the embedded shape nicely. A well-constructed cage usually involves a considerable amount of manual work. A recent work by Ben-Chen et al. [8] introduced the cage-based Variational Harmonic Map (VHM) method, which makes the manual editing of the cage redundant, and instead enables control with intuitive positional and rotational constraints. This allows for a more refined cage to bound the deformed object, and enables the user a more convenient control method that consists of a few handles, similar to the direct deformation methods. The positional and rotational constraints are enforced through energy minimization, which optimizes the deformation rigidity and smoothness. All of the above make VHM at the least on par with the state-of-the-art direct deformation methods in terms of deformation quality and intuitive user control; see Fig. 4. However, the cage still has its limits. For example, the complexity of the cage (the number of vertices) still needs to be restricted, because it directly influences the algorithm complexity. Therefore, the user might need to plan and use different degrees of resolution in different parts of the cage; for example, use a coarse shell to envelop a character's head to move as a single unit, and a refined shell to enclose the character's hand to enable finger movement. This makes the task of fully-automating cage construction challenging. Another limitation of the cage is that the deformation on the cage is

- *Zohar Levi is with the Technion, Israel*
- *David Levin is with Tel-Aviv University, Israel*

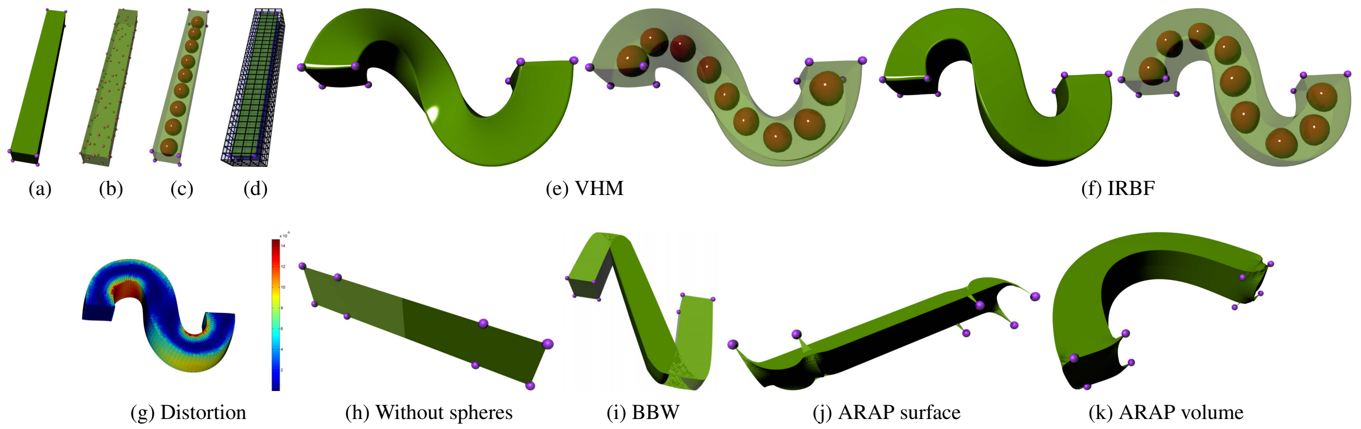


Figure 2: Bar. (a) source model and 8 anchors, (b) 100 IRBF centers, (c) 9 spheres, (d) VHM Cage, (e) VHM, 96% volume preservation, total distortion 28.3, max distortion 0.025, (f) IRBF, 92.7% volume, total distortion 32.5, max distortion 0.014, (g) distortion visualization of IRBF, (h) IRBF without spheres, such that only the smoothness term comes into play, (i) BBW [1], (j) ARAP surface [2], (k) ARAP volume [3], 96.6% volume, total distortion 27.6, max distortion 2.55.

undefined, and in general, it is not obvious how to extend the deformation to the cage exterior. One implication of the mapping singularity on the cage is that there must be a certain distance between the cage and the object. This limits the cage from fitting into tight places and, as a result, makes the deformation of models such as the clam and the chameleon's mouth in Fig. 3 impossible.

We present the IRBF method that consists of two major modifications to the VHM method: First, the use of a different set of basis functions, which are based upon interior distances within the object. The basis functions are defined with respect to centers on the surface of the object, which replace the cage. The user can then easily manipulate the density of the centers to vary over the mesh. Second, our energy functional explicitly penalizes distortions of the object and enables local control over the volume, by preserving the shapes of spheres placed within the object. This allows for a simpler formulation that eliminates the need to integrate the Green functions and differentiate them twice, as was done in VHM. In addition, IRBF mapping is regular on the whole space, and in particular on the object surface; this makes the deformation of the clam and the chameleon models as feasible as in the direct deformation methods.

Fig. 2 shows another advantage of IRBF (and VHM) over direct deformation methods such as the As-Rigid-As-Possible (ARAP) deformation methods [2, 3]. The optimization of a direct deformation method, such as the ARAP deformation, usually involves minimization of a non-linear least squares energy that measures the sum of distortions of the mesh elements. Since usually fine meshes are deformed, the number of elements is large, and the phenomenon in Fig. 2 occurs. Energy based on the total distortion (Section 5) prefers to sacrifice a few elements for the greater good of reducing the distortion in the rest of the elements by little, which is unacceptable. It is preferable to lower the maximum distortion. Unfortunately optimizing an infinity norm or even bounding the aspect-ratio distortion [9] is hard. To alleviate the problem, direct deformation methods usually use regions of anchor points (see Fig. 4) instead of a handful. Fig. 2 also shows a weakness of the popular Linear Blend Skinning (LBS). Four out of the eight anchor points undergo translation. A linear combination of translations cannot produce the necessary rotations for the deformed vertices. Even choosing the skin weights carefully with Bounded Biharmonic Weights (BBW) [1] cannot help. Since the anchor points undergo translation only, using dual quaternion blending [10] produces no improvement as well.

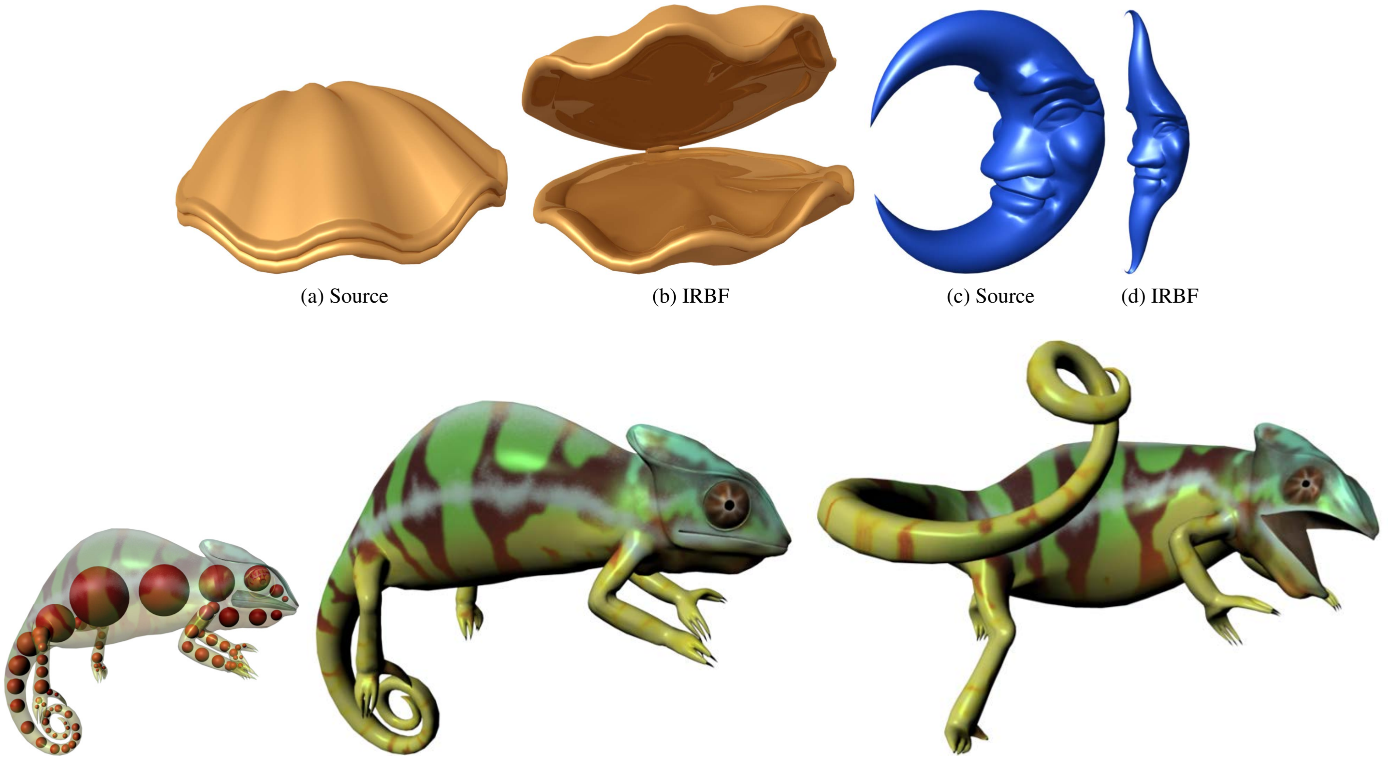
1.1 Contribution

We offer a space deformation technique that replaces the cage with control points placed over the object surface, without sacrificing the *interior locality* property that was suggested in [11]. For example, in Fig. 5, moving the horse's front left hoof does not influence the front right hoof, despite their proximity to each other in the Euclidean space. Our technique allows for a simpler formulation than the state-of-the-art VHM method, while producing comparable results in the same order of complexity; see Fig. 2, 4, 5, and 25. In addition, our technique allows for better performance, and in general deforms complex shapes at interactive rates. In Section 3.2 we suggest an IRBF-based interpolation method that enables the user to impose exact preservation of structures such as spheres within the object, or rigid parts which should not be distorted. Alternatively, the IRBF interpolation approach can be applied on top of any surface (or any other sparse data type for that matter) deformation method in order to upgrade it to a shape-aware space deformation method.

1.2 Related Work

IRBF is not a cage-based method; however, since it evolved as an improvement of cage-based methods, we shall concentrate our review on this group of methods. The first cage-based method used a generalization of the Mean Value Coordinates (MVC) to 3D [7]. One drawback of the MVC cage coordinates is that they can be negative for non-convex cages. In this case, the desired interior locality property, which was discussed previously, is lost, which leads to severe artifacts. Joshi et al. [11] offered a method that is based on the solution of the Dirichlet problem, which is guaranteed to be positive inside the domain. Lipman et al. [12] offered the Positive Mean Value Coordinates, which are computed numerically with graphical hardware. As noted in [13], all of the aforementioned methods are affine invariant, which may result in a deformation that includes shearing and anisotropic scale, which violate the shape-preserving property. Instead they offered the Green Coordinates (GC), which have closed-form expressions, and result in a detail-preserving quasi-conformal mapping. Recently, Crane et al. [14] introduced a new method for computing conformal transformations of triangle meshes in $\mathbb{R}^3$. Their method improves GC by finding a nearby spin transformation for a given mapping. The VHM method [8], which we elaborated on earlier, uses the same basis functions as GC, but with a simpler formulation. The method in [15] is also based on GC, but instead of a full cage they offer a simple umbrella-shaped cage to control the deformation locally.

We will mention a number of space deformation methods that are not cage-based. Botsch et al. [16] use triharmonic radial basis functions for real-time freeform shape editing. Zhu et al. [17] presented a deformation method based on Moving Least Squares, in which they derived a closed form solution that is based on Singular Value Decomposition (SVD). Their method includes a weighting scheme that determines the influence of point handles on vertices, based on approximate mesh geodesics, in order to address the interior locality property. In [18] the deformation is defined using a *deformation graph*. An affine transformation is associated with each graph node, which is optimized


(a) Source (b) IRBF (c) Source (d) IRBF

(e) Spheres (f) Source (g) IRBF

Figure 3: Clam, moon, and chameleon. The clam and the chameleon's mouth cannot be opened using cage-based methods.

for a smooth and rigid behavior. The deformation function is based on Euclidean distances, and therefore, does not address the interior locality property. Botsch et al. [19] define a deformation on a voxelization of the input region. They use a volumetric elastic energy that is very similar to the prism-based shell energy proposed in PriMo [4]; however, while PriMo integrates the energy over the shared faces, the new formulation integrates over the volume. This method suffers from some aliasing effects due to the discretization, and in addition, its implementation is somewhat involved. Adams et al. [20] offer to control the motion of a meshless shape in a physical simulation. The method is based on a deformation scheme that employs material distances to uphold the locality property. The method is limited to differentiable distances such as the Fast Marching Method (FMM), and thus, for example, cannot handle touching surfaces. The energy used in [20] consists of a six degree polynomial, and is optimized using L-BFGS solver, which is an order of magnitude slower than a second order energy optimized with the *local/global* scheme as in IRBF. Since the focus of the [20] is on dynamics, it supplies only two examples of a simple deformation, and it is unclear how the method would handle more challenging cases, such as the benchmark models in [21]. Linear Blend Skinning (LBS) methods are very popular space deformation methods due to their simplicity of combining joints transformations linearly, which admits real-time performance. However they have two major drawbacks: One is the setup which involves the tedious task of painting skin weights, and the other is the requirement to bind a skeleton (a disjoint set of joints is possible, but does not perform well), which limits the deformation to articulated movements, and thus cannot be used for intuitive freeform deformation. Bounded Biharmonic Weights (BBW) [1] alleviate the first problem of LBS, and offer an automatic way to compute smooth skin weights. BBW minimizes the Laplacian energy subject to bound constraints, and spreads the influence of the controls in a shape-aware and localized manner, even for objects with complex and concave boundaries. Jacobson et al. [22] alleviates the second problem of LBS using a non-linear optimization of clusters of vertices. The clustering of vertices is based on their distance in *weight space*, and the cluster transformations are optimized using the ARAP energy. However, introducing a non-linear optimization compromises LBS simplicity and performance; incurring smaller performance penalty is traded off with the deformation quality.

For the purpose of completeness, we will go over a few notable direct deformation techniques. These direct approaches achieve high quality shape-preserving deformation, but are often non-linear and cannot be used interactively. PriMo [4] achieves intuitive and robust deformations by emulating physically plausible surface behavior inspired by thin shells and plates. The surface mesh is embedded in a layer of volumetric prisms, which are coupled through non-linear elastic forces. The method in [5] is one of the few Laplacian-based deformation methods that can handle large rotations and run at interactive rates. However it does not cope well with situations where the handles undergo translation only. The reconstruction of the deformed mesh requires solving only two sparse linear systems that arise from discrete forms, but the method possibly requires more iterations to compete with the quality of state-of-the-art methods. Botsch et al. [21] survey these two methods among others, and provide benchmark models for comparison, which we used in Fig. 4. Sorkine et al. [2] suggested a non-linear, yet conceptually simple energy formulation, the so called As-Rigid-As-Possible (ARAP), which results in a detail-preserving deformation. A similar method was proposed in [23].

1.3 Method Overview

We will review the steps of IRBF, which are illustrated in Fig. 6. For simplicity, we will refer to a triangle mesh, which bounds an object Ω; however, any type of data that fulfills the requirements detailed in Section 4 is applicable. Precomputation steps:

1. The vertices of the mesh are automatically sampled (Section 4.3), and these samples are chosen as the IRBF centers (Section 2.1); see Fig. 6(a). The user is given a choice to change the sampling density in various parts of the mesh for better local control.

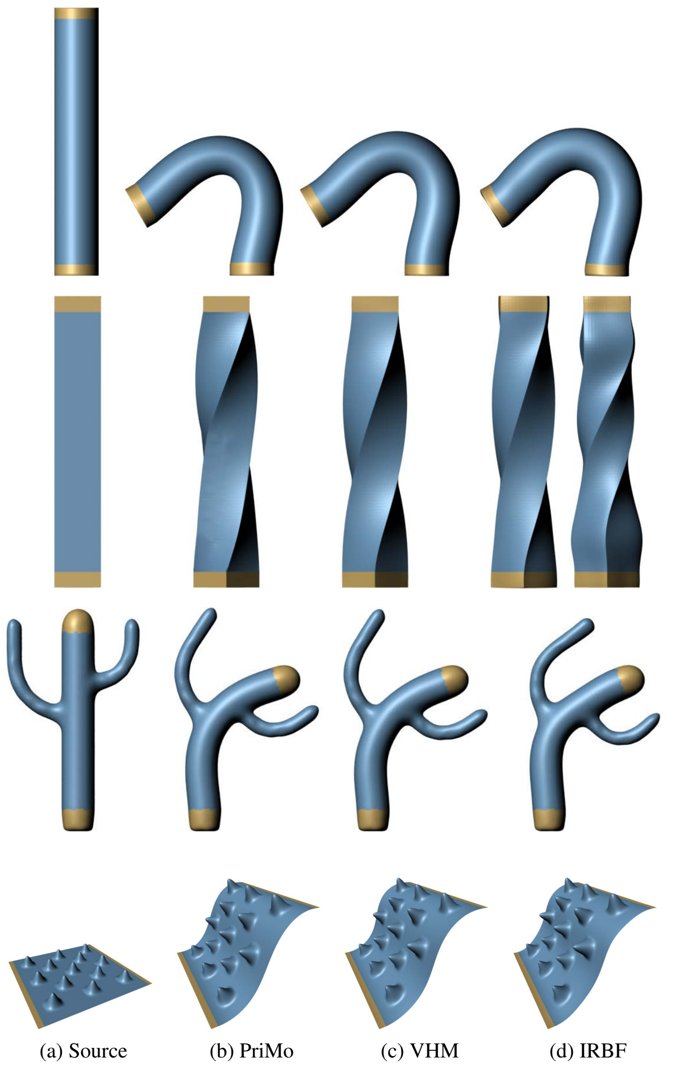

(a) Source (b) PriMo (c) VHM (d) IRBF

Figure 4: Survey models. From top to bottom: Cylinder, bar twist, cactus, and plane lift. The two IRBF models of the bar-twist model differ in the value of the shift parameter h. In the first $h = 2$, which we used for most of our models, and in the second $h = 1$ for a more local effect.

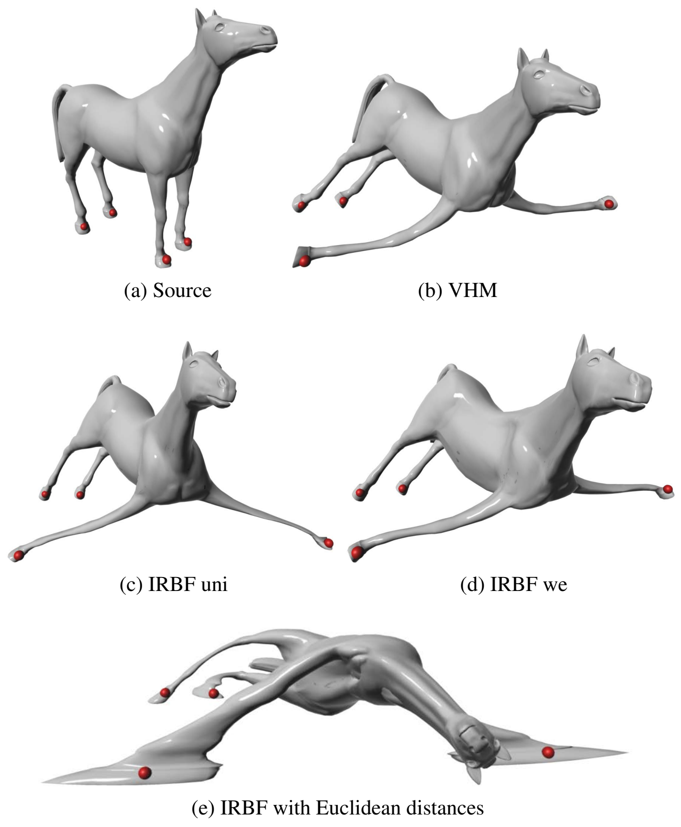

(a) Source (b) VHM (c) IRBF uni (d) IRBF we (e) IRBF with Euclidean distances

Figure 5: Horse. (a) Source mesh, (b) VHM result, (c) IRBF with uniform weights (and basis functions based on interior distances), (d) IRBF with inverse edge length weights (and basis functions based on interior distances) designates equal influence to the smaller spheres in the legs as the larger ones in the torso, (e) IRBF with basis functions based on Euclidean distances instead of interior distances.

For example, moving the hand as a whole requires fewer IRBF centers than when a movement of individual fingers is required.

2. The mesh is automatically filled with spheres (Section 4.2); see Fig. 6(b).

3. Interior distances are calculated from the IRBF centers to the local rigidity structures that represent the spheres (Section 2.3), to the deformed points, and to the anchor points (Section 4.1).

4. A linear least-squares system is constructed and pre-factored.

Deformation steps:

1. The user manipulates the positional, rotational, and scaling handles.

2. Using *local/global* optimization, the coefficients of the IRBF are calculated (Section 4.6) and the shape is deformed.

2 IRBF INGREDIENTS

This section describes the ingredients that constitute the IRBF method for manipulating an object Ω, namely the special choice of the basis functions for representing the space deformation, the utilization of spheres within the object for local 3D shape preservation, and the energy functional.

2.1 Interior RBF

One of the most successful methods for scattered data approximation to functions on $\mathbb{R}^d$ is through Radial Basis Functions (RBF) [24, 25], where an approximation at p is obtained by

$$F(p) = \frac{1}{D(p)} \sum_{c \in C} a_c \Phi(\|p-c\|), \;\; a_c \in \mathbb{R}\,, \tag{1}$$

where $C \subset \mathbb{R}^d$ is a set of centers and Φ is a real-valued function. The additional factor $\frac{1}{D(p)}$, with

$$D(p) = \sum_{c \in C} \Phi(||p-c||) \;\; ,$$

is a slight variation of the standard method, such that the form (1) can reproduce the constant function, and thus produces better approximations [24].

Similar to the RBF methodology, we suggest approximations within an object Ω, termed as IRBF approximations. Let us denote the interior distance between two points $p, c \in \Omega$ by $d_I(p,c)$, the length of the shortest path within Ω between p and c. The suggested approximation is of the form

$$\bar{F}(p) = \sum_{c \in C} a_c \phi(p,c)\,, \tag{2}$$

where

$$\phi(p,c) = \frac{1}{D_I(p)} \Phi(d_I(p,c))$$

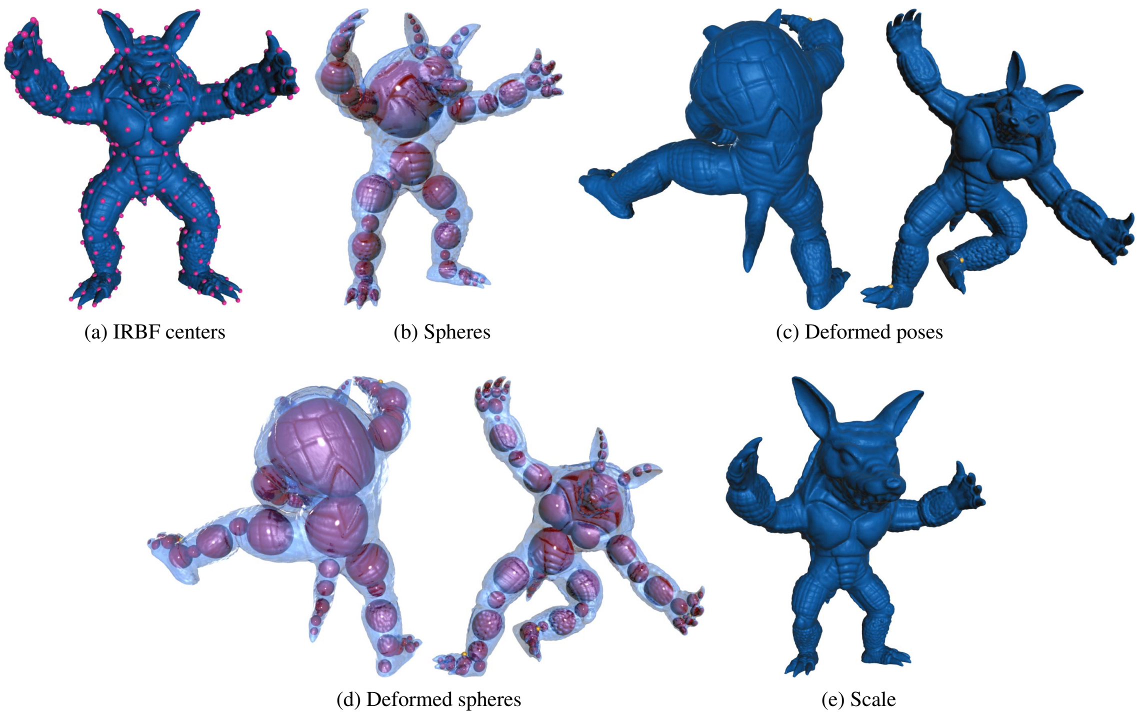


Figure 6: Armadillo. (a-b) Illustration of the IRBF ingredients. (c) The deformed poses with four marked positional constraints. Since IRBF is a space deformation, we used the same function to deform the spheres for illustration purposes. (d) Local scaling.

$$D_I(p) = \sum_{c \in C} \Phi(d_I(p,c)) \ ,$$

and we assume that we can compute $d_I(p,q)$ for any points $p,q \in \Omega$.

In applications, $\Phi(d)$ is usually chosen as an increasing function, such as d^3 [16], for approximation in $\mathbb{R}^3$. We remark that this is a natural generalization of the univariate cubic splines. On the other hand, IRBF uses a decreasing function. This is important for our application, and together with the use of interior distances, it yields the desired interior locality property. In particular we use *Hardy inverse multiquadric* basis functions that would be described shortly. Furthermore, we would like the class of mappings to include exact affine mappings. Therefore, we consider mappings of the form

$$T_\Pi(p) = Ap + t + \sum_{c \in C} a_c \phi(p,c), \ \ a_c \in \mathbb{R}^3 \ , \qquad (3)$$

where A is a $3x3$ affine matrix and t is a translation vector. Since we are looking for a mapping $T : \Omega \to \mathbb{R}^3$, the free coefficients $\{a_c\}$ are in $\mathbb{R}^3$. To denote the dependence of the mapping upon the parameters, we denote it by T_Π, where $\Pi \equiv \{A, t, \mathbf{a}\}$ is the set of free parameters, and $\mathbf{a} = \{a_c\}_{c \in C}$.

The idea behind using interior distances is as follows: As explained above, a good deformation tool should be able to move one branch of the object without affecting other branches of the object, even if they are close to each other in the Euclidean space. If we use Euclidean distances, then a source point c on one branch may have a significant influence on close-by branches of the object. However, if Φ is a decreasing function, and the distance function is the interior distance d_I, the influence of a source point c upon a point p, which is far from c within Ω, will be small.

For the deformation function to be harmonic, Φ may be chosen as the Green function in $\mathbb{R}^3$, $\Phi(d) = \frac{1}{d}$. To avoid singularity within the object or on its surface, one may choose the centers to be outside Ω, as suggested in [26]. Instead, we have chosen to add a shift parameter h, and use a function of the form,

$$\Phi(d) = \frac{1}{\sqrt{d^2 + h^2}} \ ,$$

known as Hardy inverse multiquadric [24]. The main advantage of this choice is that the functions are regular everywhere, and thus we can choose the centers inside the object as well as on its surface. We choose centers on the surface, as detailed in Section 4.3. Micchelli [27] shows that such radial basis functions are good for RBF interpolation in any dimension, i.e., that the corresponding linear system for interpolation at the centers is always regular. In our application we use interior distances instead of Euclidean distances. Our results show that this choice of basis functions, together with the use of interior distances, yield good deformation quality, and allow good local control even in cases of adjacent branches of the object. Note that the smoothness of the deformation depends on the smoothness of the interior distance function $d_I(p,q)$, which depends on the underlying algorithm (which might involve a smoothing step; see Section 4.1), and is usually continuous on the boundary and smooth otherwise.

2.2 Shape and Volume Control

Equipped with the above form (3) for the mapping, we would now like to impose some constraints on the mapping, both hard and soft constraints, so that the resulting deformation satisfies some desired properties. The hard constraints include positional requirements specifying the mapping from a set of anchor points (source points), $\{s_k\}_{k \in K} \subset \Omega$, to a set of target points, $\{t_k\}_{k \in K} \subset \Omega$,

$$T_\Pi(s_k) = t_k \ , \ k \in K \, . \qquad (4)$$

Our main goal is to obtain a deformation of a given object Ω, such that it is locally as rigid as possible, i.e. spheres B within Ω are mapped as close as possible to spheres of the same size. To achieve this goal we define a cost function U, which reflects the deviation from rigid maps (with no reflection) of the spheres.

$$U(T_\Pi) \equiv U(\Pi) = \sum_{b \in B} \rho(T_\Pi(b), M_b(b)) \ , \qquad (5)$$

where $\rho(T_\Pi(b), M_b(b))$ measures the deviation of the image of the sphere b, $T_\Pi(b)$, from the closest rigid map of b, $M_b(b) \in SE(3)$. Therefore, the unknown parameters of the mapping T_Π , namely

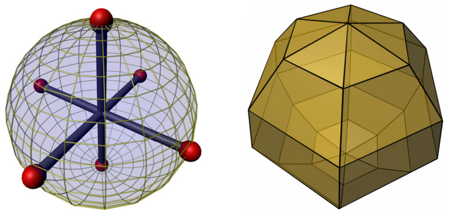

(a) 6 points (b) 32 points

Figure 7: Sphere discrete structures.

$\Pi = \{A, t, \mathbf{a}\}$, will be defined by minimizing $U(\Pi)$, subject to the constraints in (4). The computational aspects and the algorithms involved are described next.

2.3 The Energy Functional

In a practical application, the number of centers C is usually larger than the number of anchor points K and the number of reference spheres B combined (see Section 4.3). Therefore, we have more unknown coefficients in (3) than constraints. To avoid a situation of an under determined system, we add a regularization term to our energy functional

$$V(\mathbf{a}) = \sum_{c \in C} \|a_c\|^2 , \tag{6}$$

which penalizes possible solutions with large coefficients. The regularization term is also necessary for controlling the deformation smoothness. Smaller coefficients imply smaller derivatives of the transformation, and hence a smoother transformation.

Therefore, we would like to minimize an energy functional of the form

$$E(\Pi) = U(\Pi) + \lambda V(\mathbf{a}) , \tag{7}$$

subject to the positional constraints in (4). Here λ is a balancing parameter, which determines the amount of smoothness. Fig. 9 shows the influence of λ on the deformation. Large enough value of λ does not permit the jump near the farthest sphere, while a too large value suppresses the rigidity term. The first term $U(\Pi)$ in (7) has been defined above in a non-explicit way. In order to make the term explicit, we replace a sphere b with a discrete structure. To enable fast computation of the first term, we utilize a minimal structure of six points $\{p_{j,b}\}$ on b; see Fig. 7(a). Our experiments show that this structure is good enough to preserve the local shape. Denoting the three perpendicular edges by $\{e_{i,b} = p_{2i,b} - p_{2i-1,b}\}$, and their images by $\{e'_i = T_\Pi(p_{2i,b}) - T_\Pi(p_{2i-1,b})\}$, we look for the rigid map $M_b \in SO(3)$, which takes the set $\{e_{i,b}\}$ the closest, in the least-squares sense, to $\{e'_{i,b}\}$, and we replace the distortion measure $\rho(T_\Pi(b), M_b(b))$ above by a discrete variation

$$\bar{\rho}(b) = \sum_{i=1}^{3} w_{i,b} \|M_b e_{i,b} - e'_{i,b}\|^2. \tag{8}$$

where $w_{i,b} = \|e_{i,b}\|^\alpha$, $\alpha \in \mathbb{R}$. In most of our experiments we used $\alpha = -1$, which allows smaller spheres equal expression as bigger ones; see Fig. 5 for a comparison with $\alpha = 0$. The computation of M_b is discussed in Section 4.5. The total distortion is evaluated as

$$\bar{U}(\Pi) = \sum_{b \in B} \bar{\rho}(b) . \tag{9}$$

To summarize, we minimize the discrete functional

$$E(\Pi) = \bar{U}(\Pi) + \lambda V(\mathbf{a}) , \tag{10}$$

subject to the linear positional constraints (4).

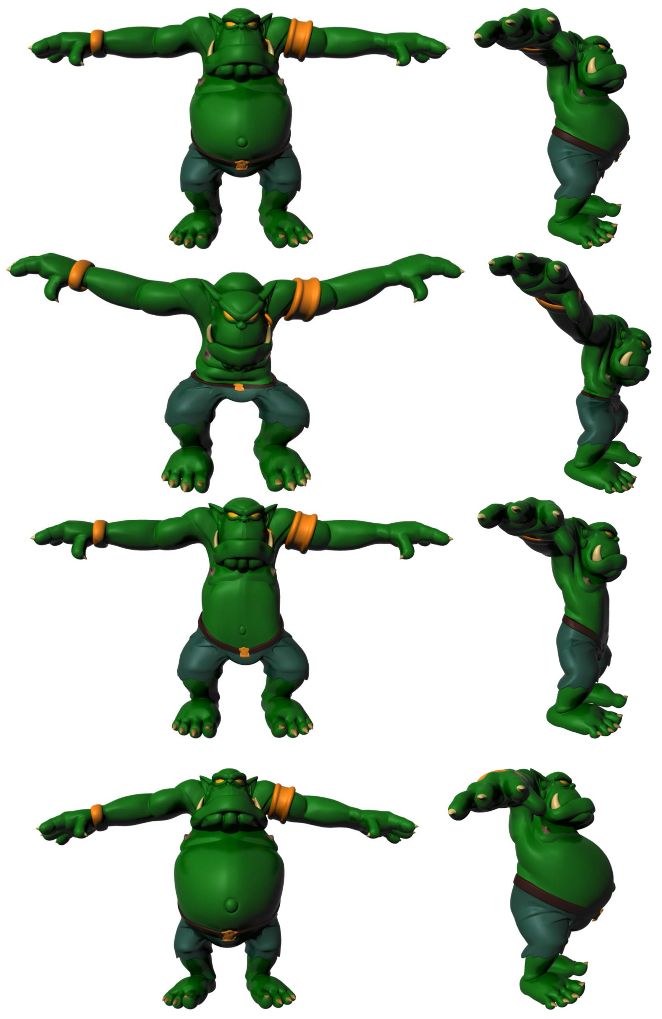

(a) Front view (b) Side view

Figure 8: Anisotropic scaling. The troll's belly is scaled. The first row contains the source model. The anchor points are the same in all the images

2.4 Local Shape and Volume Control

To better preserve the shape of large spheres, a larger set of points $\{p_{j,b}\}$ and edges $\{e_{i,b}\}$ may be chosen on the sphere. Furthermore, the set $\{p_{j,b}\}$ may be chosen as any structure that better matches the inner object shape. For example, instead of filling an elongated part with small spheres, one may fill it with one long ellipsoid to emulate a bone in an articulated object. This framework enables further flexibility of shape control: Instead of aiming at preserving a local volumetric shape, say b, one can aim at transforming it into a desired shape. For example, one may want $T_\Pi(b)$ to be as close as possible, up to a rigid map, to $S(b)$, which is a prescribed (linear or nonlinear) transformation S applied to b, such as inflation or stretching. This amounts to computing the rigid map M_b, which takes the set $\{S(e_{i,b})\}$ the closest to the corresponding set of the images $\{e'_{i,b}\}$, and defining $\bar{\rho}(b)$ instead as

$$\bar{\rho}(b) = \sum_{i} \|M_b S(e_{i,b}) - e'_{i,b}\|^2. \tag{11}$$

This operation is demonstrated in Fig. 6(e), 8.

3 IRBF Interpolation

In this section we discuss the interpolation problem, which is a related to the shape deformation method. Through this problem, we will investigate further the attributes of the basis functions, and suggest improvements in terms of memory usage and performance, which are

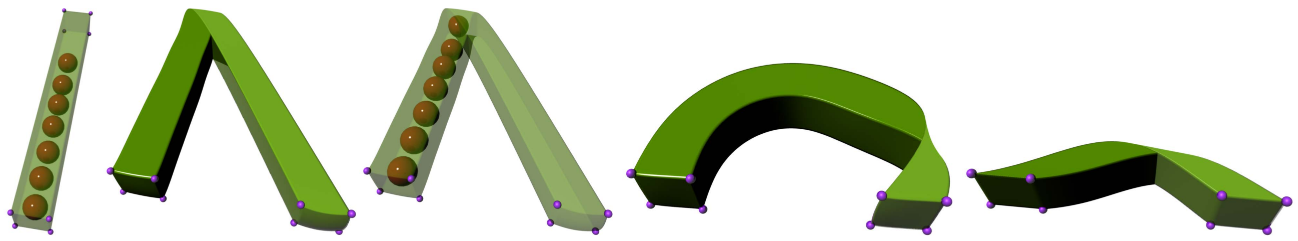

(a) Source (b) $\lambda = 10^{-10}$ (c) $\lambda = 10^{-5}$ (d) $\lambda = 10^{-2}$

Figure 9: Balancing parameter λ. Bar with only 7 spheres.

relevant to both problems. Later, we propose an IRBF-based interpolation method, which can be used to augment and extend the capabilities of the IRBF deformation method.

3.1 Interpolation and Compact Support

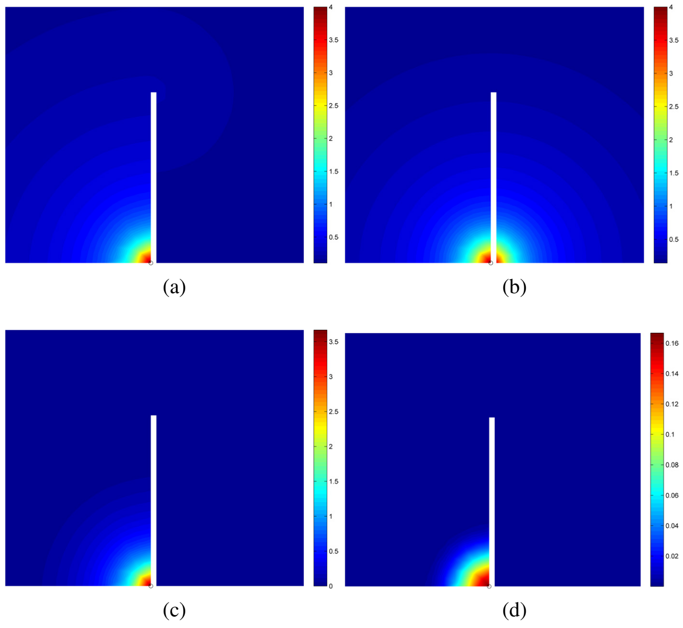

(a) (b) (c) (d)

Figure 10: An illustration of one basis function. (a) inverse multiquadric, (b) inverse multiquadric using Euclidean distances, (c) compact inverse multiquadric, (d) Buhmann [28],

We will discuss some more aspects of IRBF, and we will illustrate them on the interpolation problem, which goes as follows. Given a set of samples in the domain, we would like to construct a smooth function which interpolates them. The RBF methodology suggests to linearly combine a set of basis functions, each associated with a sample point and centered around it. As an example, we will interpolate $n = 55$ samples from the following function $f : \mathbb{R}^2 \to \mathbb{R}$. Let o be the average of the two points located at the bottom left most corner and the bottom right most corner of the shape in Fig. 11 (o is on the bottom of the narrow vertical space outside the shape). f is defined on the interior of the shape:

$$f(p(x,y)) = \begin{cases} e^{-y^2} & arccos(\frac{[1,0]\cdot(p-o)}{\|p-o\|}) > \frac{\Pi}{2} \\ -e^{-y^2} & else \end{cases} .$$

To get the mapping, we solve the linear system for the vector of coefficients a

$$Ga = f \ ,$$

where G is a $n \times n$ matrix, $G_{ij} = \phi(c_i, c_j)$, and a, f are vectors of length n. As explained in Section 2.1, we use for the basis function Φ the inverse multiquadric function.

One improvement that can be considered is using a basis function with compact support. This leads to a sparse matrix G that would improve memory usage and solution performance in an order of magnitude. However, there is a typical tradeoff between locality and quality of the approximation [28]. First, we suggest our compact inverse multiquadric basis function

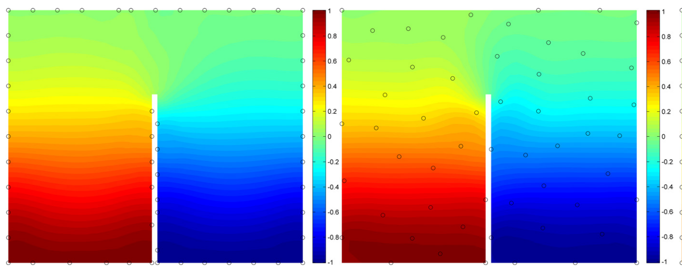
$$\Phi_{CIMQ}(d) = \begin{cases} \frac{1}{\sqrt{d^2+h^2}} - \frac{1}{\sqrt{d_{max}^2+h^2}} & d \le d_{max} \\ 0 & else \end{cases} ,$$

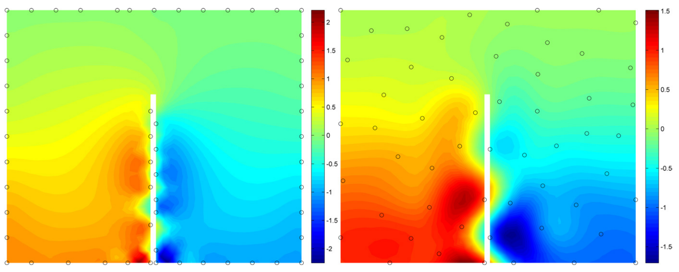
where h is the shift parameter (Section 2.1), and d_{max} is the radius of influence. Φ_{CIMQ} is continuous only, but sufficient for IRBF deformation, which doe not require derivatives. Another basis function that behaved well in our experiments is offered by Buhmann [28]

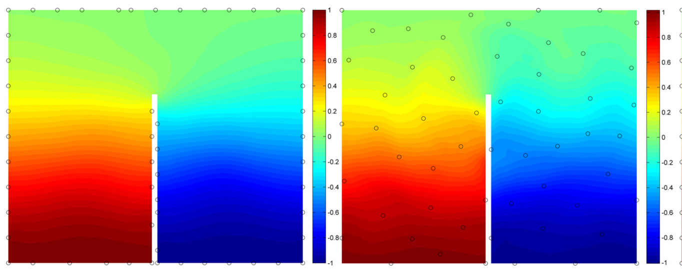
$$\Phi_{Buhmann}(r) = \begin{cases} \frac{1}{6} & r = 0 \\ 2r^4 ln(r) - \frac{7}{2}r^4 + \frac{16}{3}r^3 - 2r^2 + \frac{1}{6} & 0 < r \le 1 \\ 0 & else \end{cases} ,$$

where $r = \frac{d}{h}$. $\Phi_{Buhmann}$ is a three times continuously differentiable function. See Fig. 10 for an illustration of the basis functions, and Fig. 11 for the interpolation results using Euclidean and interior distances. The problem with using the standard RBF functions with Euclidean distances is immediately apparent. The Euclidean distances are not shape-aware, and influence nearby branches, which degrades the interpolation significantly. The results show both uniform boundary sampling and farthest point sampling in the interior. For the compact support functions, the average of non-zeros in a row of G using boundary sampling is 14.5, and the average of of using farthest point sampling is 9.8. But while the interior sampling admits better sparsity, the boundary sampling produces smoother results. Φ_{CIMQ} has a clear separation between the radius of influence d_{max}, which determines the number of non-zeros, and the shift parameter h. Normally, h is dependent on the distance between the centers in the domain. We suggest the following heuristic to choose adaptive h and d_{max} for a basis function at a vertex automatically. Based on the the previous experiments, we choose d_{max} to include 14 closest neighbors when using boundary sampling, and 9 closest neighbors when using farthest point sampling. h is chosen as the average of the distance between a point in the neighborhood and its closest neighbor. Besides using the heuristic, the user can tweak h to get a more local effect; see Fig. 12 and 4.

The basis functions of VHM are harmonic, which have nice properties such as the maximum principle and the mean value property. Since our motivation for IRBF deformation arose from VHM, we opted to use inverse multiquadric with centers sampled on the boundary to approximate the behavior of a Green function. The farther away from the boundary, the closer $\Delta\Phi_{CIMQ}$ approaches to zero, which implies smoothness.


(a) Inverse multiquadric


(b) Inverse multiquadric with Euclidean distances


(c) Compact inverse multiquadric

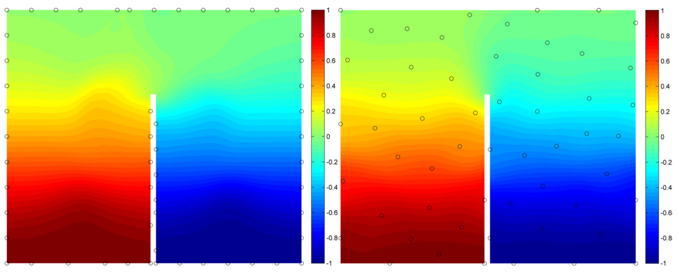
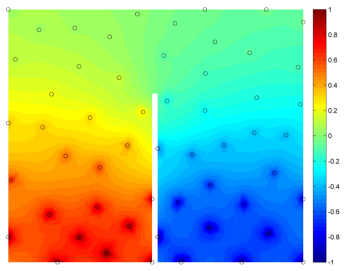

(d) Buhmann

Figure 11: Interpolation. In each sub-figure: (left) 55 centers on boundary, (right) farthest point sampling of 55 centers.


(a)

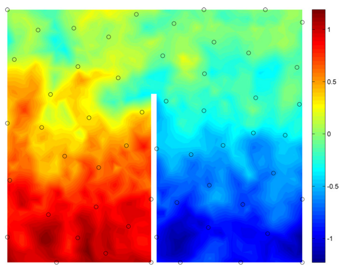

(b)

Figure 12: Interpolation. (a) inverse multiquadric with a small h results in strong local influence around the centers, (b) inverse multiquadric with a large h results in global influence; centers which are farther apart influence each other's neighborhood, which leads to color mixing.

3.2 Spheres and Surface Interpolation

Based on the mapping that was generated with IRBF deformation, we propose an IRBF-based interpolation method that we term IRBF interpolation, which interpolates the spheres, such that the mapping preserves their discrete structure exactly. After optimizing (10), we have the mapping T_Π, and we know where each sphere is mapped to

$$b' = T_\Pi(b) \ .$$

Knowing where the barycenter of b is mapped to, along with the optimal rotation M_b that was found in the optimization process, we can map b rigidly to an optimal (closest rigid transformation) position:

$$\hat{b} = M_b(b - t) + t'$$

where t, t' are the barycenters of b, b' respectively. Now we are ready to construct a new space deformation that rigidly maps the spheres. We put IRBF centers at the anchor points s_k and at the vertices of a discrete structure that represents each of the spheres. We found that interpolating the discrete structure in Fig. 7(b) with IRBF interpolation preserves spheres (of any resolution) almost exactly. We note that the initial IRBF optimization, which determines the rigid transformations of the spheres, is performed quickly, as before, with 7(a). Having defined a basis function at each point that we would like to interpolate, we solve a linear system for the new mapping parameters $\hat{\Pi}$. Spheres interpolation results can be seen in Fig. 13.

When constructing the IRBF interpolation, spheres that do not overlap in the domain, should not overlap in the range as well; else it is a prescription for foldovers. Moreover, even if the spheres do not overlap, but still close to each other, since they are usually mapped with different optimal rotations (i.e. there are at least two adjacent points with a rotation jump), this causes an unsmooth deformation as can be seen in the bends in Fig. 13(a). To remedy such situations we use smaller spheres. We also note from the results that by preserving the spheres, we introduced distortion into the surface (that was mapped with $\hat{\Pi}$).

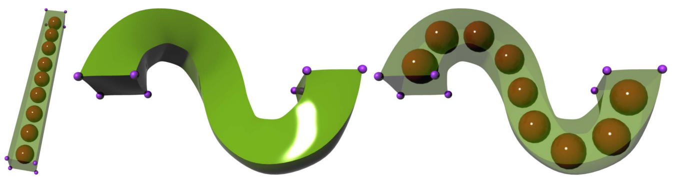
In a similar fashion, the spheres can be deformed into any shape, and using the IRBF interpolation the mapping can be extended to the whole subspace. Moreover, such a procedure can applied to an arbitrary shape deformed by an arbitrary method. For example, by employing IRBF interpolation, any surface (or any other sparse data type for that matter) deformation can be extended to the whole subspace, where only the source and deformed surface are needed.

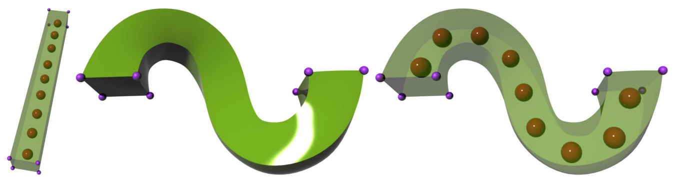
We will demonstrate the IRBF interpolation in two practical examples:

- In Fig. 14 we used IRBF interpolation to extend the surface mapping, which was generated by PriMo [4] for the cylinder in Fig. 4, to the whole subspace. The input for the IRBF interpolation is a mapping from the n vertices of a source triangle mesh P to the vertices of a deformed triangle mesh Q. We set an IRBF center at each vertex of P; for fine meshes, we sample P arbitrarily or use compact basis functions for better performance. We construct the matrix $G \in \mathbb{R}^{n\times n}$ of the basis functions in a similar way that was described in the previous section. Next we solve for the matrix of coefficients $a \in \mathbb{R}^{n\times 3}$

$$Ga = f \ ,$$

where $f \in \mathbb{R}^{n\times 3}$ are the deformed coordinates of the vertices where the IRBF centers were placed at. For simplicity, we omitted from the system the variables A, t of the affine mapping (Section 2.1). Now, each point in the subspace can be deformed using (3). For illustration we deformed some spheres using the extended mapping (which were not used in the construction of the IRBF interpolation).


(a)


(b)

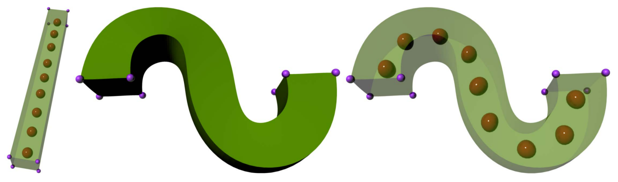
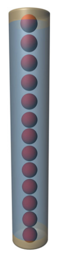

(c)

Figure 13: Spheres interpolation. (a) interpolating the spheres in Fig. 2, 85% volume, total distortion 31.4, max distortion 0.03, (b) smaller spheres interpolation, 81% volume, total distortion 28.5, max distortion 0.02, (c) smaller spheres without interpolation, 89.1% volume, total distortion 29.9, max distortion 0.014.

- In a typical user interface, the user paint-selects areas on the object for rigid preservation, e.g. the troll's arm bracelets. In Fig. 15, we first deformed the troll using IRBF deformation with Euclidean distances. This as opposed to Fig. 1, where interior distances (FMM) were used. We can see that the face and the upper arms got thinner. Next, we placed centers at each of the discretized spheres, and on the two arm bracelets. We found for each of these structures the closest rigid transform to the IRBF deformation, and we set it as the RHS of the linear system that solves for the interpolation coefficients. Meaning, the mapping (rigid for each object) was extended from the spheres and the bracelets to the whole subspace, which was then used to deform the rest of the troll. As can be seen, the bracelets shape is preserved exactly, and the exact preservation of the spheres fixed the thinning (e.g. of the face and upper arms) that the Euclidean distances caused (for the interpolation we used Euclidean distances as well).


(a) Source

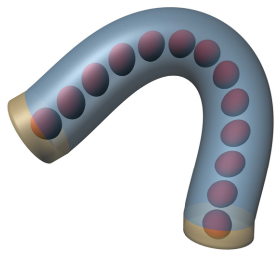

(b) PriMo+IRBF

Figure 14: Cylinder interpolation. The surface mapping generated by PriMo [4] in Fig. 4 is interpolated to the whole subspace using IRBF interpolation. For illustration we deformed some spheres using the extended mapping.

4 Implementation Details

In this section we provide the details for the algorithms that are used in IRBF.

4.1 Interior Distances

IRBF requires the interior distances within the deformed object, from the centers C to the points on the spheres B, to the anchor points, and to all the transformed points. In our experiments we tested four different methods for calculating interior distances, depending upon the data type and the object shape:

Euclidean If the deformed object is convex, or if it does not have moving parts that are close to each other, then Euclidean distances may be used as interior distances. This method is applicable to any type of data.

FMM The fast marching method on a regular grid. The deformed object is voxelized using a regular 3D grid, and holes in the interior are flood-filled [29]. The result is a 3D binary image, where values of voxels contained within the object are set to one, and values of voxels outside the object are set to zero. We calculate geodesic distances in the image using [30], where the voxel values prescribe the speed for the Eikonal equation. We should note that since the speed in the voxels outside the image is zero, the distance to these voxels is infinity. Thus, if we would like to define the mapping on the outside space as well, some form of distance diffusion should be performed (e.g., see [31]). The discrete values at the voxel centers are smoothly interpolated using trilinear interpolation. The grid size should be fine enough to separate between two surfaces such as a character's legs. This method is suitable for triangle meshes, tetrahedral meshes, polygon soups, and any other type of object that can be voxelized. In the case of a sparse point cloud, a snake [32] or the level set method [33] should be employed in the initial step, to extract a watertight boundary that can be filled.

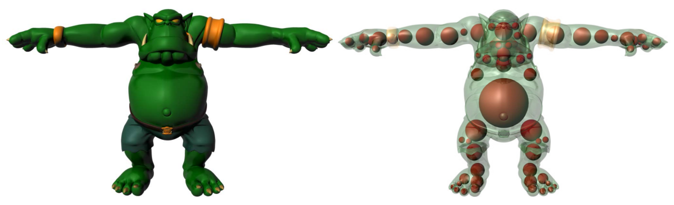
MVC We use the method in [34], to calculate geodesic distances on a triangle mesh, and interpolate them to the entire space using MVC. This method is suited for closed triangle meshes that consist of one connected component.

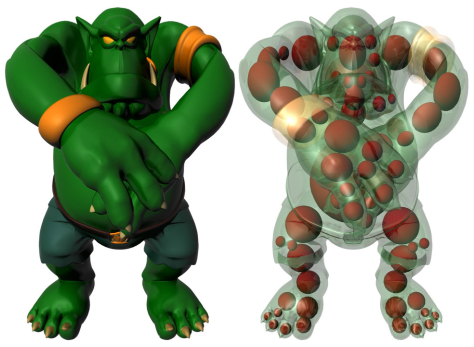
Geodesics in heat Crane et al. [31] offer an algorithm to compute the shortest geodesic distances in a domain, which involves a solution of a pair of standard linear elliptic problems. The only requirement is a definition of a Laplace-Beltrami operator, and thus the method supports a wide variety of data types including meshes with non-planar faces or scattered point clouds. Where IRBF is concerned, this method is suitable for tetrahedral meshes. For other types of data some sort of interpolation is required, such as diffusion over a regular grid.

4.2 Setting The Controlling Spheres

We have adopted two methods to fill or approximate an object with spheres.

Curve skeleton A curve skeleton is extracted using [35, 36] for meshes, and [37] for point clouds (we used the default parameters). At sampled points along the skeleton, we place spheres with a radius of 90% of the distance from the sampled point to the surface. This method is suitable for a model with limbs.


(a) Source


(b) IRBF with Euclidean distances

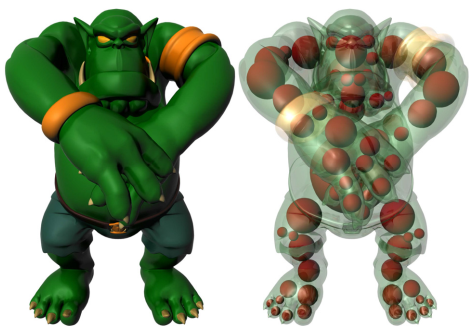

(c) IRBF interpolation with Euclidean distances

Figure 15: Troll interpolation. (b) IRBF deformation with Euclidean distances, (c) IRBF interpolation with Euclidean distances that preserves exactly the spheres and the two arm bracelets; see Fig. 16.

Sphere tree Approximating a shape using spheres is usually used for collision detection. We use the method by Bradshaw et al. [38] (we changed only the octree level parameter) to approximate with spheres models, such as the clam in Fig. 3 and the plane in Fig. 4; see Fig. 19.

The set of spheres that is generated with the methods above is pruned according to a threshold of maximum overlap between the spheres (can be negative), while giving priority to larger spheres. Spheres that are too small are pruned as well.

Comparing Fig. 2(f) with Fig. 13(c) we can see that the spheres size influences the quality of the deformation, and in this case larger spheres are better.

4.3 Choosing IRBF Centers

The number of centers depends on the deformation resolution. For example, moving the fingers of a hand requires more centers than moving the entire hand as a unit. The user can supply hints for adaptive sampling. A practical rule of thumb is covering each sphere with four IRBF centers. It is better to have more centers than necessary, where the excess of DOFs would be taken care of by the smoothing term (Section 2.3), than less centers than necessary. In our experiments we used between 100-800 centers; see Table 1. We use farthest point sampling to select N vertices on a triangle mesh. For other types of objects, any other sampling method, such as random sampling, is applicable. See Fig. 20 for an illustration of the influence of the number of centers on the deformation.

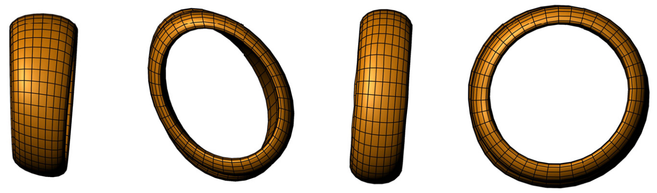

(a) IRBF Euclidean (b) IRBF interpolation

Figure 16: Troll's arm bracelets.

4.4 Proxy Mesh

For some of the methods in the initialization step, which were described in the last three subsections, we noted limitations regarding the type of the data being deformed. Nevertheless, in order to test these methods and compare their performance to others, we applied them to a proxy mesh, which needs not be an accurate approximation of the object. We created the proxy by voxelizing the object [29], and extracting an isosurface as a mesh. The rest of the algorithm steps run as usual on the original object. For example, the MVC implementation requires a single component mesh. In order to test IRBF with MVC interior distances on a multi-component object, we fed a proxy mesh to MVC. In another situation, we fed a coarse proxy mesh to the curve skeleton algorithm, which reduced the extraction time significantly. Thus, for extreme cases, or where a method does not support the data type natively, the proxy mesh can function as the cage in VHM, but without the restrictions. The proxy is used in the initialization step if necessary, and unlike the cage, its complexity does not influence the real-time deformation.

4.5 Closest Rigid Map

Here we refer to the problem of finding a rigid map $M \in SO(3)$, which takes a set of vectors $\{e_i\}$ the closest, in the least-squares sense, to a set $\{e'_i\}$, minimizing

$$\sum_i w_i \|Me_i - e'_i\|^2 \ , \qquad (12)$$

for some scalar weights w_i. This is known as the *Procrustes problem* or the *absolute orientation* problem, and a solution can be computed using SVD or polar decomposition of the covariance matrix [2, 37, 17, 39]. We should also note that instead of finding the closest rigid map, we can find the closest similarity map, as explained in [17], with an additional minor calculation.

4.6 The Optimization Procedure

The optimization problem of minimizing (10) subject to the linear constraints (4) is a non-linear problem. Similar to [2, 8, 23], we use the so called *local/global* approach. In the local step, the unknown parameters $\Pi = \{A, t, \mathbf{a}\}$ are frozen, and the local optimal rigid maps $\{M_b\}_{b \in B}$ are calculated using the approach from the previous section. In the global step, the maps $\{M_b\}_{b \in B}$ are frozen, and a linear system is solved for the parameters Π. The linear system is derived as a constrained least-squares system by differentiating (10), and adding the linear constraints in (4) as an additional equal number of rows and columns of Lagrange multipliers. The result is a square system of dimension $(4+|C|+|K|) \times (4+|C|+|K|)$. The linear system is prefactored, since only the RHS is changed during the iterations. We initialize the iterations with the parameters $\Pi_0 = \{I, 0, \mathbf{0}\}$, which represent the identity map. The stopping condition tests the amount of change in the RHS. See the Appendix for a faster formulation using soft positional constraints.

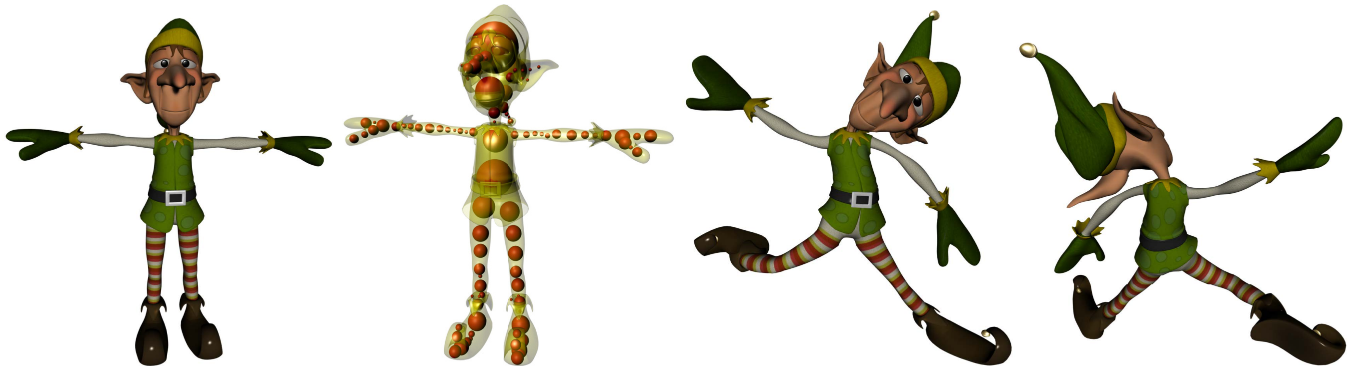

(a) Source (b) Spheres (c) IRBF

Figure 17: Elf.

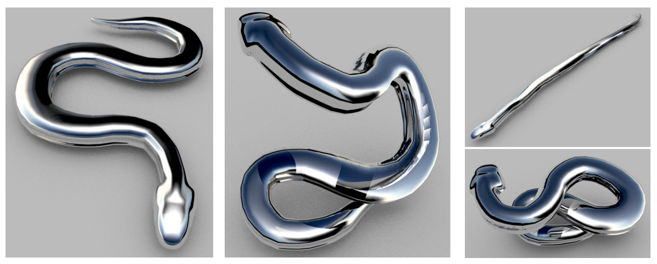

(a) Source (b) IRBF

Figure 18: Snake.

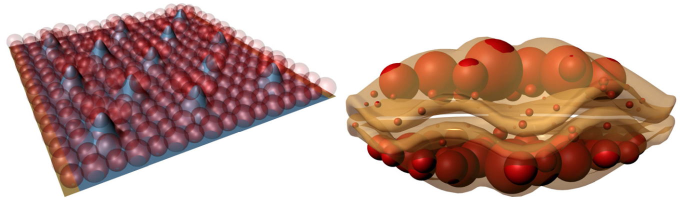

(a) Plane (b) Clam

Figure 19: Sphere tree.

Model	#Vertices	#Spheres	#Centers	#Iterations	Solve (ms)	Deform (ms)	Total (ms)
Armadillo	173k	79	400	28	.82	57	80
Bar	7k	9	100	24	.2	8	14
Chameleon	8k	73	800	51	.8	5	46
Clam	8k	88	800	41	1	8	50
Cyclops	53k•	66	800	41	.85	14	50
Elf	30k•	107	600	33	1	12	49
Horse	8k	77	400	22	0.7	5	20
Hydra	67k•	160	800	25	1	43	77
Moon	5k	29	300	34	.55	5	24
Octopus	51k•	182	800	43	1.1	36	84
Rat	45k•	281	600	15	2.9	45	89
Snake	15k	41	400	39	.71	8	36
Sin bar	32k	44	100	38	.84	9	41
Troll	40k•	73	400	24	.87	18	39

Table 1: Performance measured in milliseconds. *#Vertices* is the number of the model vertices and a '•' was added to multi-component objects; *#Spheres* is the number of spheres that were used; *#Centers* is the number of IRBF centers; and *#Iterations* is the number of iterations in the optimization. *Solve* is the time for one local/global iteration; *Deform* is the time for computing (13) in the Appendix, namely deforming the model points after finding the IRBF coefficients; and *Total* is the total deformation time.

5 Results and Discussion

We ran the experiments on a laptop with Intel Core i7-2720QM CPU 2.2Ghz, 8GB RAM, and Nvidia Quadro 3000M. We summarized the deformation performance statistics of IRBF with soft constraints in Table 1. In our implementation, we used the CUBLAS library for the linear algebra. Note that a major part (96%) of the *Deform* step in Table 1 is dominated by memory transfers between GPU and CPU, and a better implementation can skip this step and render the new coordinates directly from the GPU memory. The results in Table 1 were generated using the soft constraints approach in the Appendix, with infinite support basis functions. In terms of memory usage, the dominant part was taken by the precomputed matrix $\Psi Z_{trunc}^{\dagger}$ of size $N \times (3|B|+|K|)$, where N is the number of points, $|B|$ is the number of spheres, and $|K|$ is the number of positional constraints. For example, the largest precomputed matrix was 173000×241 for the Armadillo, and it occupied 0.31GB RAM for double precision calculation.

We compared IRBF to the surveyed methods in [21], VHM, BBW, ARAP surface, and ARAP volume; see Fig. 2, 4, 5, and 25. IRBF results on the survey models not only implies a comparison to the surveyed direct deformation methods in [21], but also a comparison to subsequent methods that show results on the survey models as well (e.g. [22]). A comparison to VHM implies a comparison to [19, 18], which the VHM paper compares with (e.g. the sinus bar model in Fig. 25), while pointing out some limitations of these methods.

In all our comparisons with VHM, we positioned VHM anchors, where the Jacobian is optimized to be rigid, in the sphere centers that were chosen for IRBF. We also added Hessian points in the middle of each cage face. The cages that we used for VHM consist of 400-550 vertices, with roughly twice as many faces. For the same models we used 100-400 IRBF centers. In the linear system constructed in VHM the number of variables that represent the coefficients of the basis functions depends on the cage complexity and equals to $\#vertices + \#faces$ of the cage. In IRBF, we solve for the coefficient of each IRBF center, thus the number of variables is equal to the number of IRBF centers. In other words, if we allot for the IRBF centers the number of VHM cage vertices, the VHM method would solve for thrice as many variables as the IRBF, and the matrix of the linear system in Section 4.6 would be nine times larger. In terms of local steps, the number of SVDs that are calculated in each iteration is the same for both methods. We also used the same number of iterations for both methods, 30 on average, depending on the model and how large the deformation is. On these

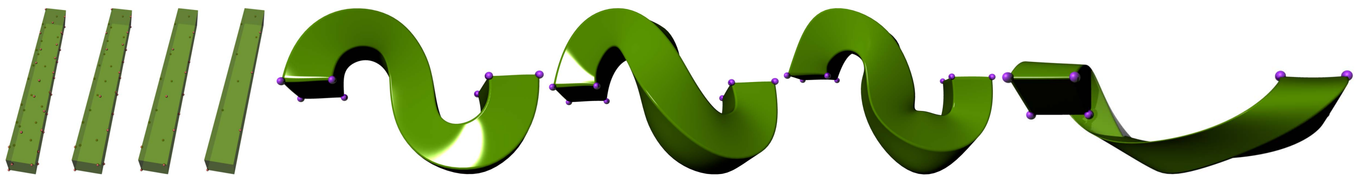

(a) Source (b) IRBF

Figure 20: Varying number of IRBF centers sampled with farthest point sampling: 50, 35, 20, 10.

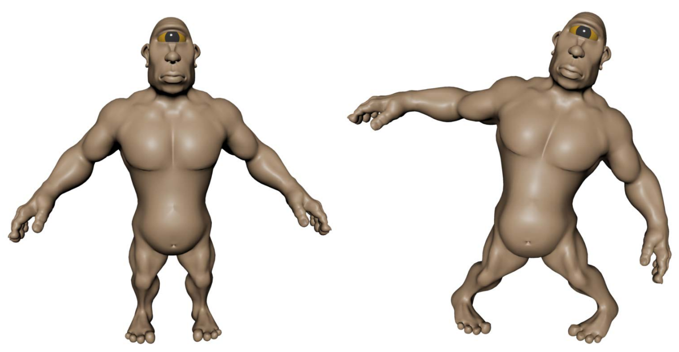

(a) Source (b) IRBF

Figure 21: Cyclops.

terms IRBF runs faster than VHM. Also, as discussed in Section 3.1, IRBF performance can be farther improved using compact support basis functions.

We already noted that cage-based deformations cannot handle deformations such as the clam and the chameleon's mouth in Fig. 3. Likewise, it would be quite challenging to build high quality cages for models such as the hydra and the octopus in Fig. 22, 23.

We tested the four methods for calculating interior distances that were detailed in Section 4.1. The Euclidean method takes a few seconds to run on a moderate mesh. MVC takes two minutes for a mesh of 10k vertices; bigger meshes should be handled by the other methods, or a coarse proxy (Section 4.4) should be employed. The Geodesics In Heat method takes a few seconds to compute the distances of a moderate size tetrahedral mesh. FMM complexity depends mainly upon the grid size, which in turn depends on the distance between the branches in the object. In our experiments FMM took three minutes on average, and up to ten minutes for the hydra model in Fig. 22.

Distortion measurment In Fig. 2 and 13 we measure the deformation quality. The bar has 7202 vertices and we tetrahedralized it with 21568 tetrahedra. First we measured the volume of the deformed mesh w.r.t. to the volume of the source measure. Then we measured the total (sum on all tetrahedra) and max ARAP distortion [3], which is defined for a tetrahedron as

$$E = A \min_{R \in SO(3)} \|df - R\|_F^2 \ ,$$

where A is the tetrahedron volume, df is the differential of the map, and R is the closest rotation to df in the Frobenius norm. The ARAP distortion measures how far the Jacobian matrix (constant over a tetrahedron) is from a rotation, or alternatively measures the amount of stretching a tetrahedron has gone through.

A vivid demonstration of IRBF can be seen in the accompanied video.

6 Conclusion

We proposed a new detail-preserving space deformation method for deforming a 3D object at interactive rates. Our method uses Internal Radial Basis Functions (IRBF), where the functions are radial with respect to interior distances within the object. The basis functions are defined at centers on the object surface, which replace the controlling cage that was used in previous methods. This allows for a deformation of touching surfaces and an easier, fully-automated setup. The algorithm functional represents a measure of distortion of a set of spheres placed within the object. It is quite surprising that the local shape of an object can be preserved or controlled by controlling the shape of just a few spheres in it. We have shown by several examples that our results are on par with the state-of-the-art deformation methods.

One limitation of our method is the lack of a theoretical justification of the choice of Φ. We cannot guarantee, for example, that the maximum principle holds, and therefore we cannot estimate the error bound, unlike harmonic mapping methods such as GC and VHM. This can be an avenue for future research, which should test for other choices of basis functions. Another limitation is the need to tweak the λ parameter in order to get reasonable results. Determining λ automatically is another goal.

References

[1] A. Jacobson, I. Baran, J. Popović, and O. Sorkine, "Bounded biharmonic weights for real-time deformation," in *SIGGRAPH*, 2011, pp. 78:1–78:8.

[2] O. Sorkine and M. Alexa, "As-Rigid-As-Possible surface modeling," in *SGP*, 2007, pp. 109–116.

[3] I. Chao, U. Pinkall, P. Sanan, and P. Schröder, "A simple geometric model for elastic deformations," in *SIGGRAPH*, 2010, pp. 38:1–38:6.

[4] M. Botsch, M. Pauly, M. Gross, and L. Kobbelt, "Primo: coupled prisms for intuitive surface modeling," in *SGP*, 2006.

[5] Y. Lipman, O. Sorkine, D. Levin, and D. Cohen-Or, "Linear rotation-invariant coordinates for meshes," in *SIGGRAPH*, 2005, pp. 479–487.

[6] T. W. Sederberg and S. R. Parry, "Free-form deformation of solid geometric models," in *SIGGRAPH*, 1986, pp. 151–160.

[7] T. Ju, S. Schaefer, and J. Warren, "Mean value coordinates for closed triangular meshes," in *SIGGRAPH*, 2005.

[8] M. Ben-Chen, O. Weber, and C. Gotsman, "Variational harmonic maps for space deformation," in *SIGGRAPH*, 2009.

[9] N. Aigerman and Y. Lipman, "Injective and bounded distortion mappings in 3D," *TOG*, vol. 32, no. 4, pp. 106:1–106:14, 2013.

[10] L. Kavan, S. Collins, J. Žára, and C. O'Sullivan, "Geometric skinning with approximate dual quaternion blending," *TOG*, vol. 27, no. 4, pp. 105:1–105:23, 2008.

[11] P. Joshi, M. Meyer, T. DeRose, B. Green, and T. Sanocki, "Harmonic coordinates for character articulation," *TOG*, 2007.

[12] Y. Lipman, J. Kopf, D. Cohen-Or, and D. Levin, "GPU-assisted positive mean value coordinates for mesh deformations," in *SGP*, 2007, pp. 117–123.

[13] Y. Lipman, D. Levin, and D. Cohen-Or, "Green coordinates," *TOG*, vol. 27, no. 3, pp. 78:1–78:10, 2008.

[14] K. Crane, U. Pinkall, and P. Schröder, "Spin transformations of discrete surfaces," *TOG*, vol. 40, 2011.

[15] Z. Li, D. Levin, Z. Deng, D. Liu, and X. Luo, "Cage-free local deformations using green coordinates," *Vis. Comp.*, 2010.

[16] M. Botsch and L. Kobbelt, "Real-time shape editing using radial basis functions," in *CGF*, vol. 24, no. 3, 2005, pp. 611–622.

[17] Y. Zhu and S. Gortler, "3D deformation using moving least squares." Harvard Univ.: TR-10-07, Tech. Rep., 2007.

[18] R. W. Sumner, J. Schmid, and M. Pauly, "Embedded deformation for shape manipulation," in *SIGGRAPH*, 2007.

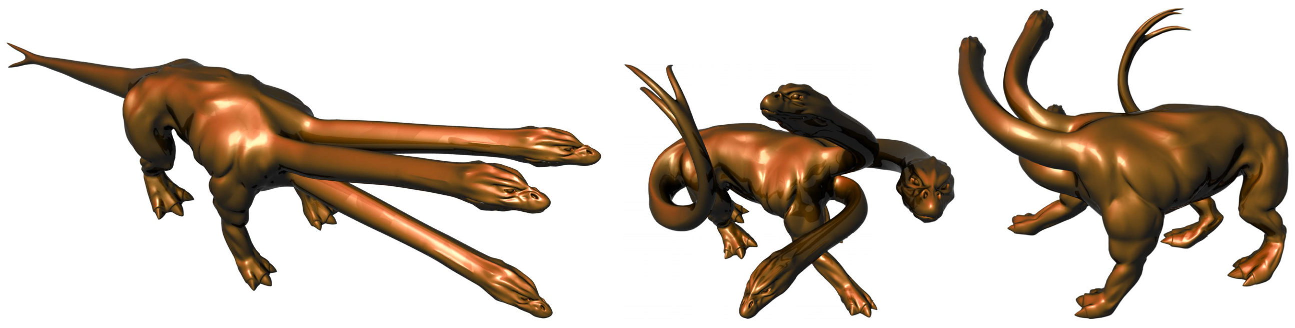

(a) Source (b) IRBF

Figure 22: Hydra.

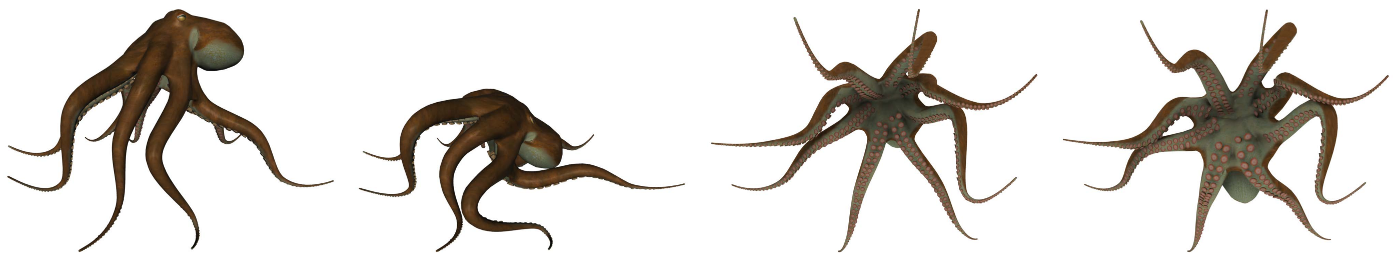

(a) Source (b) IRBF (c) Source (d) IRBF

Figure 23: Octopus.

[19] M. Botsch, M. Pauly, M. Wicke, and M. Gross, "Adaptive space deformations based on rigid cells," *CGF*, 2007.

[20] B. Adams, M. Wicke, M. Ovsjanikov, M. Wand, H.-P. Seidel, and L. J. Guibas, "Meshless shape and motion design for multiple deformable objects," *CGF*, vol. 29, no. 1, pp. 43–59, 2010.

[21] M. Botsch and O. Sorkine, "On linear variational surface deformation methods," *TVCG*, vol. 14, no. 1, pp. 213–230, 2008.

[22] A. Jacobson, I. Baran, L. Kavan, J. Popović, and O. Sorkine, "Fast automatic skinning transformations," *SIGGRAPH*, vol. 30, no. 4, 2012.

[23] W. Xu, K. Zhou, Y. Yu, Q. Tan, Q. Peng, and B. Guo, "Gradient domain editing of deforming mesh sequences," in *SIGGRAPH*, 2007.

[24] R. Franke, "Scattered data interpolation: Tests of some method," *Math. Comp.*, vol. 38, pp. 181–200, 1982.

[25] M. D. Buhmann, "Radial basis functions," *Acta Numerica*, vol. 9, pp. 1–38, 1 2000.

[26] D. Levin and A. Tal, "A boundary collocation method for the solution of a flow problem in a complex three-dimensional porous medium," *Num. Meth. Fluids*, vol. 6, no. 9, pp. 611–622, 1986.

[27] C. A. Micchelli, "Interpolation of scattered data: Distance matrices and conditionally positive definite functions," *Constructive Approximation*, vol. 2, pp. 11–22, 1986.

[28] M. D. Buhmann, "A new class of radial basis functions with compact support," *Math. Comp.*, vol. 70, pp. 307–318, 2001.

[29] D. Morris and K. Salisbury, "Automatic preparation, calibration, and simulation of deformable objects," Tech. Rep., 2007.

[30] M. S. Hassouna and A. A. Farag, "Multistencils fast marching methods: A highly accurate solution to the eikonal equation on cartesian domains," *PAMI*, vol. 29, no. 9, pp. 1563–1574, 2007.

[31] K. Crane, C. Weischedel, and M. Wardetzky, "Geodesics in heat: A new approach to computing distance based on heat flow," *TOG*, 2013.

[32] C. Xu and J. L. Prince, "Snakes, shapes, and gradient vector flow," *IEEE Trans. on Image Processing*, vol. 7, no. 3, pp. 359–369, 1998.

[33] J. A. Sethian, "A fast marching level set method for monotonically advancing fronts," in *Proc. Nat. Acad. Sci*, 1995, pp. 1591–1595.

[34] R. M. Rustamov, Y. Lipman, and T. Funkhouser, "Interior distance using barycentric coordinates," in *SGP*, 2009.

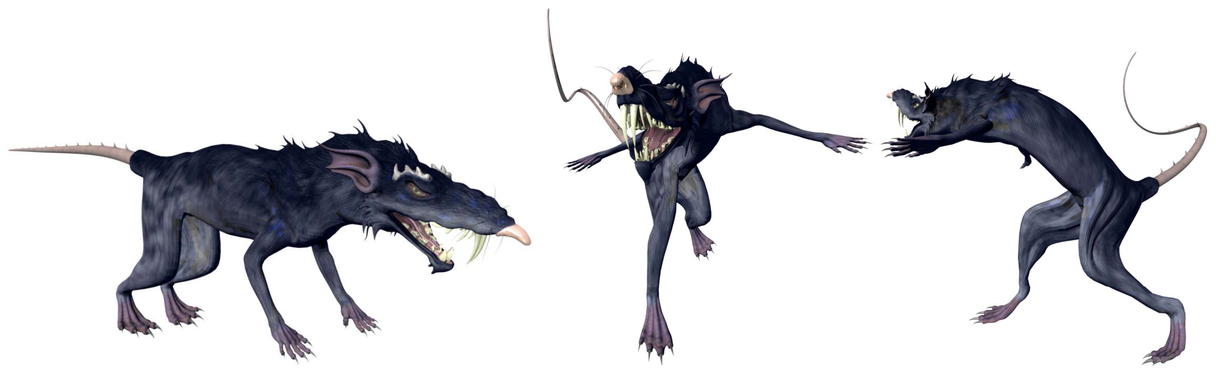
[35] O. K. C. Au, C. L. Tai, H. K. Chu, D. Cohen-Or, and T.-Y. Lee, "Skeleton extraction by mesh contraction," in *SIGGRAPH*, 2008, pp. 44:1–44:10.

[36] T. K. Dey and J. Sun, "Defining and computing curve-skeletons with medial geodesic function," in *SGP*, 2006, pp. 143–152.

[37] J. Cao, A. Tagliasacchi, M. Olson, H. Zhang, and Z. Su, "Point cloud skeletons via laplacian based contraction," in *SMI*, 2010, pp. 187–197.

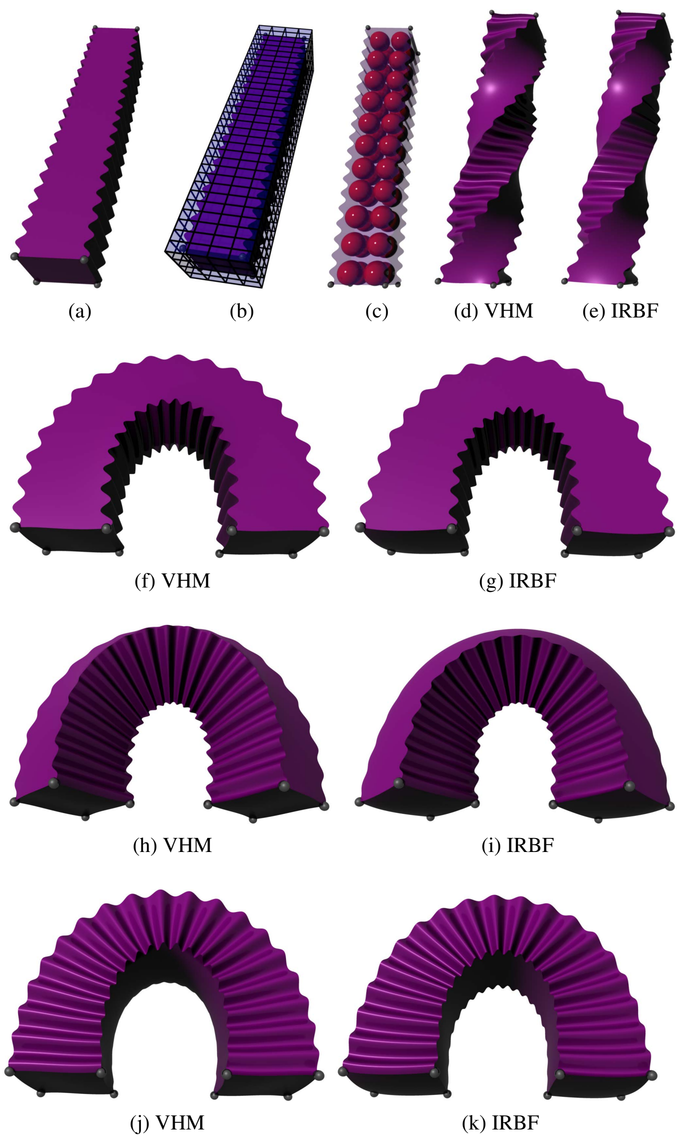
[38] G. Bradshaw and C. O'Sullivan, "Adaptive Medial-Axis Approximation for Sphere-tree Construction," *TOG*, 2004.

[39] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-D point sets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, no. 5, pp. 698–700, May 1987.



(a) Source (b) IRBF

Figure 24: Rat.



(a) (b) (c) (d) VHM (e) IRBF

(f) VHM (g) IRBF

(h) VHM (i) IRBF

(j) VHM (k) IRBF

Figure 25: Sinus bar. (a) Source mesh, (b) Cage for VHM, (c) Spheres for IRBF.

Zohar Levi received his PhD in computer science from the Technion, Israel, in 2013. His research interests include computer graphics and geometry processing.

David Levin is a professor of applied mathematics at the School of Mathematical Sciences, Tel-Aviv University, Israel. His research interests include multivariate approximation, convergence acceleration and computer graphics. Professor Levin received his BSc, MSc, and PhD from Tel-Aviv University (in 1970, 1972 and 1975), and he joined the School of Mathematical Sciences in 1978.

APPENDIX: SOFT POSITIONAL CONSTRAINTS

For most practical applications, the hard constraints in (4) can be relaxed into soft constraints, and instead of the linear system in Section 4.6, similar to [8], we construct a thinner matrix, Z, which results in better performance. To that end, instead of the energy in (10), we define a new energy $E_{soft}(\Pi)$ as a block stack matrix of a concatenation of the energy terms and constraints:

$$\left\| \underbrace{\begin{bmatrix} P & W_{3|B|\times 1} & G \\ \mu S & \mu \mathbf{1}_{|K|\times 1} & \mu H \\ 0 & 0 & \lambda I_{|C|\times|C|} \end{bmatrix}}_{Z} \underbrace{\begin{bmatrix} A^T \\ t \\ a \end{bmatrix}}_{\Pi} - \underbrace{\begin{bmatrix} \hat{P} \\ \mu T \\ 0 \end{bmatrix}}_{\Gamma} \right\|_F^2 .$$

The first row of block matrices results from differentiating the spheres distortion term in (9), the second row from the positional constraints in (4), and the third row from differentiating the regularization term in (6). Let us index the spheres in B, $B = \{b_\beta\}_{\beta=1}^{|B|}$. Z is a $(3|B| + |K| + |C|) \times (4 + |C|)$ block matrix, Π is a $(4 + |C|) \times 3$ matrix, Γ is a $(3|B| + |K| + |C|) \times 3$ matrix, $P = (e_{1,b_1}, ..., e_{3,b_{|B|}})^T$, and $W = (w_{1,b_1}, ..., w_{3,b_{|B|}})$. The block G is of size $3|B| \times |C|$ with entries $G_{3(\beta-1)+i,j} = w_{i,\beta}(\phi_j(p_{2i,b_\beta}, c_j) - \phi_j(p_{2i-1,b_\beta}, c_j))$, $i \in \{1,2,3\}$. The entries of $H_{K\times|C|}$ are $H_{k,j} = \phi_j(s_k, c_j)$, $S_{|K|\times 3} = (s_1, ..., s_{|K|})^T$, $T_{|K|\times 3} = (t_1, ..., t_{|K|})^T$, $\hat{P} = (w_{1,b_1} M_{b_1} e_{1,b_1}, ..., w_{3,b_{|B|}} M_{b_{|B|}} e_{3,b_{|B|}})^T$, and μ, λ are weighting scalars. In most of our experiments we used $\mu = 100$, $\lambda = 0.001$.

When the rotations are frozen, as explained in Section 4.6, minimizing E_{soft} can be done using normal equations:

$$\Pi = Z^\dagger \Gamma, \;\; Z^\dagger = (Z^T Z)^{-1} Z^T$$

Since the last lines of Γ are zero, we can truncate them from the system along with the corresponding columns from $Z^\dagger$. Given N points in a matrix form, $X = (x_1 ... x_N)^T$, and their basis functions in a matrix Ψ, $\Psi_{i,j} = \phi_j(x_i, c_j)$, the resulting deformation is

$$F(X) = XA^T + t + \Psi Z^\dagger_{trunc} \Gamma_{trunc} \, . \tag{13}$$

$\Psi Z^\dagger_{trunc}$ is a precomputed matrix of size $N \times (3|B| + |K|)$, compared to a similar precomputed matrix in Section 4.6 of size $N \times |C|$.