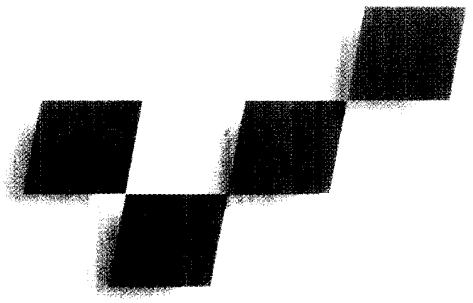
Image Warping with Scattered Data Interpolation



Detlef Ruprecht
Andersen Consulting, Germany

Heinrich Müller
University of Dortmund, Germany

A new approach to image warping based on scattered data interpolation methods provides smooth deformations with easily controllable behavior. A new, efficient deformation algorithm underlies the method.

The term "image warping" describes methods for deforming images to arbitrary shapes. Digital image warping has received a lot of interest in recent years, much of it due to the large range of applications.

Some applications are in "artistic" areas, like computer animation. Here, image warping is a basis for two-dimensional morphing, the smooth transition between keyframe images. Other "artistic" applications include facial animation and free-form deformation of images.

Other applications are in scientific image processing. In image data acquisition, the acquisition method often deforms the acquired image, for example, through lens distortion. In satellite imaging, distortions are caused by surface curvature and oblique viewing angles. In ultrasonic medical imaging, distortions are caused by the varying speed of sound in different materials. These deformations must be rectified to obtain the correct coordinates, in a process called "registering." In registration applications, correspondence points between the deformed image and a reference image might be chosen manually or automatically. An application related to registration is the normalization of samples to a standardized form to allow comparisons between the samples independent of size and slight form variations. Last, but not least, we also used image warping and morphing for interpolation between planar slices in medical data sets.[1]

The process of warping an image can be divided into two steps:

1. Compute the desired displacements of all pixels in the source image.
2. Resample the image to create the output image.

Researchers have developed a number of algorithms for the second step in recent years. Resampling can be performed efficiently by processing horizontal and vertical displacements in two separate stages. Wolberg wrote a detailed introduction to such methods,[2] and we employ one of them in our work.

This article focuses on the first step of image warping, construction of a suitable mapping function. We will call this the "deformation" step. It is commonly performed either by applying a global analytic mapping function to the image pixel positions or by using a set of control points that specify the displacements of some points in the initial image. The method developed by Smythe[3] requires these control points to be the nodes of a quadrilateral mesh. A more recent method developed by Beier and Neely[4] uses control lines. The latter method is particularly useful if we want an exact displacement of edges in the image. However, mesh- as well as line-based algorithms do not suit automatic methods for generating point correspondence.[1]

Goshtasby[5] developed a method that uses arbitrary control points to triangulate the image plane and transform each triangle independently. In the following discussion, triangulation-based deformation is just a special case of scattered data interpolation methods applied to deformation (also noted recently by Arad et al.[6]). Triangulation-based methods have several disadvantages: They have noncontinuous derivatives, and they are not defined outside the convex hull of the control points. In addition, for aesthetically pleasing results, they need a large number of triangles. We will describe deformation methods that overcome these limitations and compare their advantages and disadvantages. As an example, Figure 1 (next page) shows some subtle and some not so subtle deformations of a clown puppet image.

Image deformation and scattered data interpolation

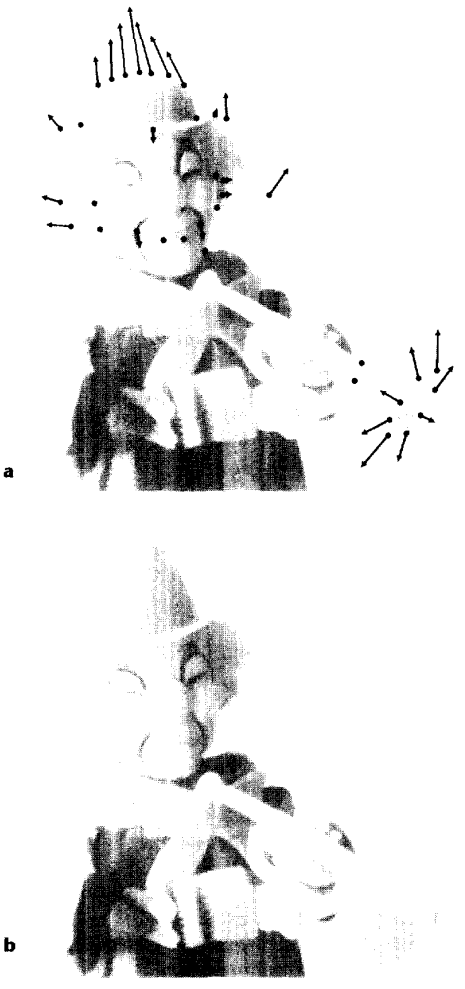
We can formulate the problem of image deformation in terms of input and output:

Input: n pairs $(\mathbf{p}_i, \mathbf{q}_i)$ of control points, $\mathbf{p}_i, \mathbf{q}_i \in \mathbb{R}^2$, $i = 1, \ldots, n$.

Output: An at-least-continuous function $\mathbf{f} : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ with $\mathbf{f}(\mathbf{p}_i) = \mathbf{q}_i$, $i = 1, \ldots, n$.

Besides continuity, other properties of the function $\mathbf{f}$ are desirable, specifically those concerning visual aes-

0272-17-16/95/$4.00 © 1995 IEEE



1 Deformation of a clown. (a) A clown puppet with control point assignments. (b) The image deformed with multiquadrics.

thetics, which are hard to describe in exact terms. These properties will become recognizable in the following discussion of concrete propositions.

The problem of scattered data interpolation is to find a real valued multivariate function interpolating a finite set of irregularly located data points. For bivariate functions, this can be formulated in terms of input and output as follows:

Input: n data points $(\mathbf{p}_i, y_i)$, $\mathbf{p}_i \in \mathbb{R}^2, y_i \in \mathbb{R}, i = 1, \ldots, n$.

Output: An at-least-continuous function $f : \mathbb{R}^2 \to \mathbb{R}$ interpolating the given data points, that is, $f(\mathbf{p}_i) = y_i$, $i = 1, \ldots, n$.

The mapping function $\mathbf{f}$ can be split into its components $f_j : \mathbb{R}^2 \to \mathbb{R}, j = 1, 2$. and each component treated separately. We can then interpolate these components with scattered data interpolation methods. The data points to use are $(\mathbf{p}_i, q_{i,j})$, with $q_{i,j}$ the jth component of $\mathbf{q}_i$.

Triangulation-based methods

Scattered data interpolation through triangulation of the data points is a classic approach to scientific visualization. This method consists of first dissecting the definition space into a suitable set of triangles with the given data points as the corners of the triangles. Then, each of the triangles is interpolated independently.

Several criteria for an "optimal" triangulation are known. One, called *Delaunay* triangulation, maximizes the minimum inner angle of all triangles to avoid thin triangles. It can be computed with a divide-and-conquer algorithm of algorithmical complexity $O(n \log n)$ where n is the number of data points.

If the pixels lie on a regular grid, as is usually the case for image warping applications, the correspondence between triangles and pixels is easily known in constant time, and thus each pixel can be mapped to its new location in constant time as well. With N the number of pixels in the images, this gives an overall complexity of $O(n \log n + N)$.

A continuous, although not smooth, interpolation can be obtained by linear interpolation within each triangle. This method has been applied to image warping.[5] The visual appearance of the result is quite acceptable if the deformations are small and if enough data points are provided so that changes of the transform coefficients between neighboring triangles remain small. A smooth deformation can be obtained by using nonlinear patches within the triangles, which can be obtained from a Clough-Tocher subtriangulation.[7]

A problem common to all triangulation-based methods for image warping is that foldover can easily occur. The term "foldover" describes the occurrence of overlapping deformations, that is, several nonadjacent pixels in the input image are mapped to the same pixel in the output image. With triangulation-based methods, this happens if the orientation of the corner points changes for any of the triangles, that is, the triangle is flipped over by the deformation.

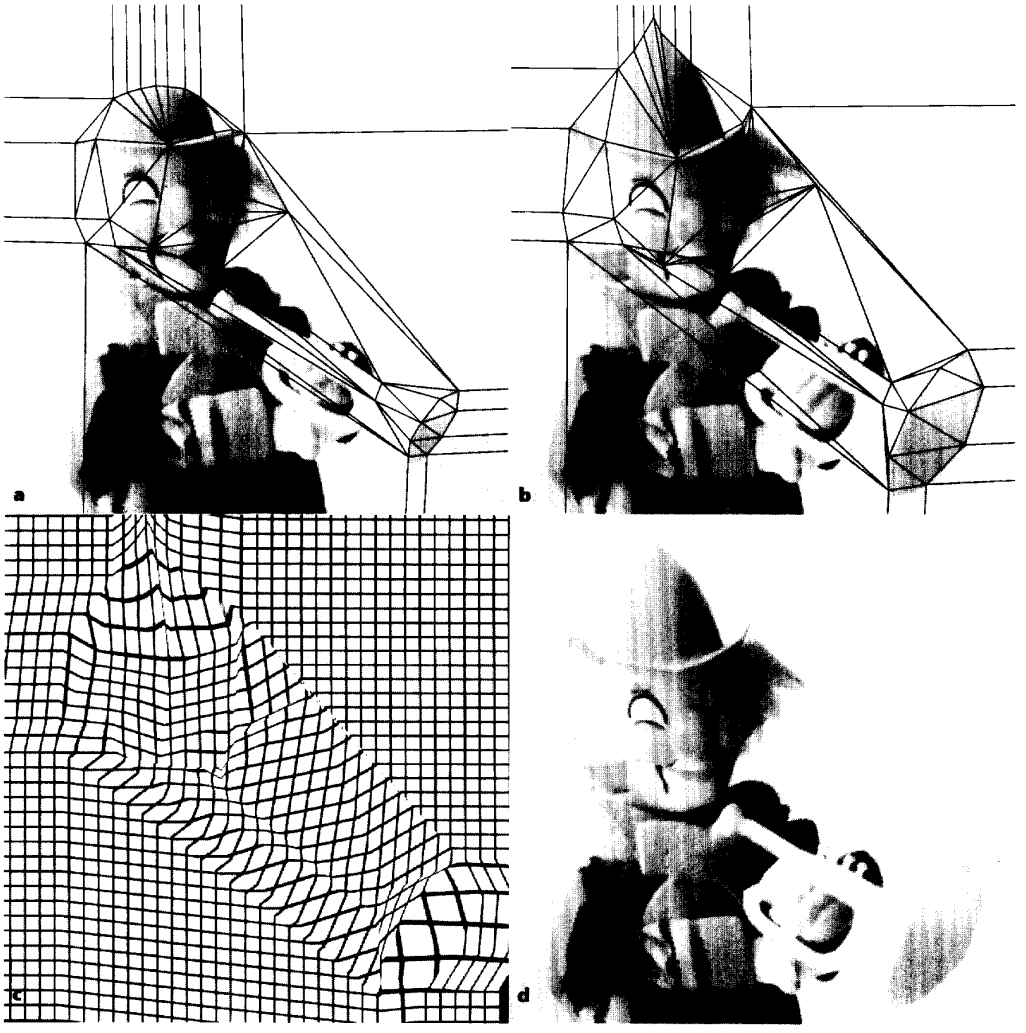
Figure 2 shows an example of deformation by triangulation with linear interpolation within the triangles. The Delaunay triangulation is shown along with the images. Note the edges in the deformed image where control points next to each other have been deformed dissimilarly. This is particularly noticeable in the deformation of a test grid with the same displacements, Figure 2c. In addition, an example of foldover is observable around the left corner of the mouth.

Inverse distance-weighted interpolation methods

Inverse distance-weighted interpolation methods were originally proposed by Shepard[8] and improved by a number of other authors, notably Franke and Nielson.[9] For each data point $\mathbf{p}_i$, a local approximation $f_i(\mathbf{p}) : \mathbb{R}^2 \to \mathbb{R}$ with $f_i(\mathbf{p}_i) = y_i$, $i = 1, \ldots, n$ is determined. The interpolation function is a weighted average of these local approximations, with weights dependent on the distance of the observed point from the given data points,

$$f(\mathbf{p}) = \sum_{i=1}^{n} w_i(\mathbf{p}) f_i(\mathbf{p}) \qquad \textbf{(1)}$$

where $f_i(\mathbf{p}_i) = y_i$, $i = 1, \ldots, n$. $w_i : \mathbb{R}^2 \to \mathbb{R}$ is the weight function, which must satisfy the conditions



2 Deformation with triangulation-based methods. (a) Delaunay triangulation of the initial positions of all control points. (b) Deformation of the triangulation. (c) Deformed test grid with linear patches within the triangles. (d) Deformed image with linear patches.

$$w_i(\mathbf{p}_i) = 1,\ \sum_{i=1}^{n} w_i(\mathbf{p}) = 1 \text{ , and}$$
$$w_i(\mathbf{p}) \geq 0,\ i = 1, \ldots, n \qquad \textbf{(2)}$$

These conditions guarantee the property of interpolation. Shepard[8] proposed the following simple weight function:

$$w_i(\mathbf{p}) = \frac{\sigma_i(\mathbf{p})}{\sum_{j=1}^{n} \sigma_j(\mathbf{p})} \text{ with } \sigma_i(\mathbf{p}) = \frac{1}{d(\mathbf{p}, \mathbf{p}_i)^{\mu}} \qquad \textbf{(3)}$$

where $d(\mathbf{p}, \mathbf{p}_i)$ is the distance between $\mathbf{p}$ and $\mathbf{p}_i$. Franke and Nielson[9] proposed a weight function with locally bounded influence of the form

$$\sigma_i(\mathbf{p}) = \left[\frac{(R_i - d(\mathbf{p}, \mathbf{p}_i))_+}{R_i\, d(\mathbf{p}, \mathbf{p}_i)} \right]^{\mu} \qquad \textbf{(4)}$$

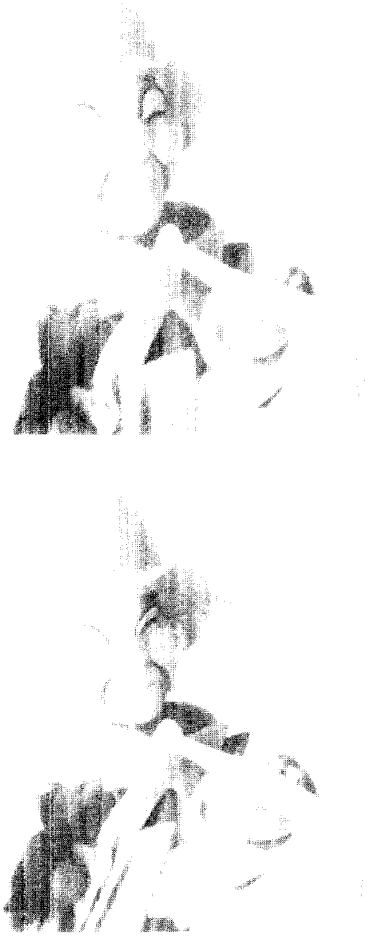
where $x_+ = \max(x, 0)$ and R_i is a user-specified range of influence. In our experiments, this weight function produced far better results, although careful choice of R_i is important.

The smoothness of the interpolation is determined by the exponent μ. $\mu > 1$ assures continuity of the derivatives, with the value of the first derivative vanishing at the data points. $\mu = 2$ is a popular choice.

For the local approximations, linear or quadratic polynomials are normally used, with coefficients derived from estimations of derivative values at the data points. Franke and Nielson[9] proposed computing the local approximation $f_i(\mathbf{p})$ by minimizing the squared error of the mapping of other nearby control points $\mathbf{p}_j$ with f_i, weighted with the $\sigma_i(\mathbf{p}_j)$ from Equation 4. The corresponding error function $E_i(f)$ is

$$E_i(f) = \sum_{j=1,\, j \neq i}^{n} \sigma_i(\mathbf{p}_j)(f(\mathbf{p}_j) - y_j)^2 \qquad \textbf{(5)}$$

The optimum local approximation is obtained by mini-



3 Deformation with inverse distance-weighted methods. (a) Linear local approximation, $\mu = 2$. (b) Locally bounded weight functions, $\mu = 3$ and $R = 400$.

mizing $E_i(f)$. We used this approach in our experiments with local approximations.

The application of inverse distance-weighted interpolation to image warping gives

$$\mathbf{f}(\mathbf{p}) = \sum_{i=1}^{n} w_i(\mathbf{p})\mathbf{f}_i(\mathbf{p}) \quad (6)$$

where $\mathbf{f}(\mathbf{p}_i) = \mathbf{q}_i$, $i = 1, \ldots, n$. $\mathbf{f}_i : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ are the local approximations. Using linear local approximations, the $\mathbf{f}_i$ can be rewritten in terms of a 2×2 matrix $\mathbf{T}_i = (t_{i,kl})$, $k, l = 1, 2$:

$$\mathbf{f}_i(\mathbf{p}) = \mathbf{q}_i + \mathbf{T}_i(\mathbf{p} - \mathbf{p}_i) \quad (7)$$

The error function corresponding to such a linear transformation $\mathbf{T}$ is

$$E_i(\mathbf{T}) = \sum_{j=1, j\neq i}^{n} \sigma_i(\mathbf{p}_j) \left\| \mathbf{q}_i + \begin{pmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{pmatrix} (\mathbf{p}_j - \mathbf{p}_i) - \mathbf{q}_j \right\|^2 \quad (8)$$

We use the partial derivatives with respect to the t_{kl} to obtain the minimum of the error function. These are linear functions of the t_{kl}. Setting the derivatives to zero, we obtain a system of four linear equations with four unknowns, easily solved for the t_{kl}.[10]

Inverse distance-weighted methods can be tailored to many specific needs. We can control the range of influence of the control points by adjusting the parameter R_i in Equation 4. We can relax the interpolation condition by adding a damping term to the weight function, or enforce strict adherence to the local approximation within a distance R around control points.[10]

Figure 3 shows examples of image deformations through inverse distance-weighted methods with linear local approximations. The results are clearly not too satisfactory, mainly because of rather uneven transitions between the control points' ranges of influence. This causes unwanted bends in the deformation, particularly along the rim of the hat in the picture. We obtained the best results with the locally bounded weight functions from Equation 4 (see Figure 3b), but even this image shows some unevenness. Finding a good choice of parameters requires considerable experimentation.

Another disadvantage of any inverse distance-weighted method is that, as a global method, it has a time complexity of $O(nN)$, which makes it rather slow. Thus, although flexible, such a method does not suit image warping very well.

Radial basis functions

Another popular approach to scattered data interpolation is to construct the interpolation function as a linear combination of basis functions, then determine the coefficients of the basis functions,

$$f(\mathbf{p}) = \sum_{i=1}^{n} \alpha_i f_i(d(\mathbf{p}, \mathbf{p}_i)) + p_m(\mathbf{p}) \quad (9)$$

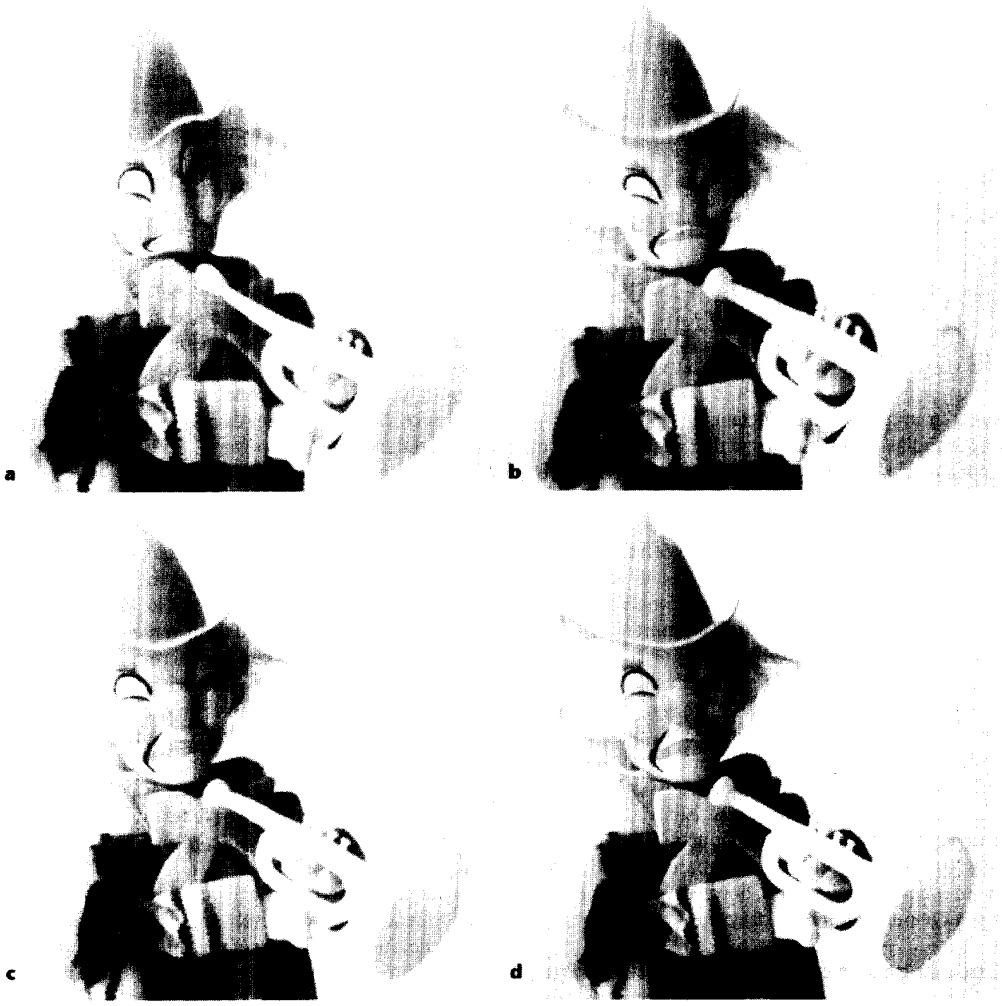
The values of the basis function f_i depend only on the distance from the data point and are thus called radial. $p_m(\mathbf{p})$ is a polynomial of degree m. It assures a certain degree m of polynomial precision. The coefficients α_i are calculated by putting the data points into Equation 9 and solving the resulting system of linear equations. The differentiability of this interpolation method depends directly on the differentiability of the basis functions f_i used.

A large number of radial basis functions are known, some of which have already been used for deformation.[9] Multiquadrics, originally proposed by Hardy,[11] are particularly effective and time-efficient radial basis functions:

$$f(d) = (d^2 + r^2)^{\mu} \quad (10)$$

with $r > 0$ and $\mu \neq 0$. For the exponent μ, Hardy proposed $\mu = \pm 0.5$. For $r \neq 0$, the basis functions are infinitely differentiable, so the resulting interpolation is in C^∞.

The characteristic radius $r > 0$ can be chosen arbitrarily. It determines the visual smoothness of interpolation at the given data points. For image deformation, the selection of r is critical for good results. Values too



4 Deformation with radial basis functions. (a) Fixed $r = 50$, $\mu = 0.5$. (b) Adaptive r_i, $\mu = 0.5$. (c) $\mu = -1$. (d) Locally bounded basis functions, $R_i = r_i$, $S = 5$.

small lead to undesirable unevenness in the deformed image; values too big can lead to foldover. With fixed values of r, both effects are likely to occur even within the same image, as shown in Figure 4a, where foldover is evident around the mouth and the horn of the trumpet, and, at the same time, the rim of the hat is unpleasantly bent. Thus, it appears necessary to choose individual values of r for each control point.

Following a suggestion by Eck,[12] we used individual values r_i for each data point $\mathbf{p}_i$, computed from the distance to the nearest neighbor:

$$r_i = \min_{i \neq j} d_i(\mathbf{p}_j) \qquad \textbf{(11)}$$

This causes the deformation to be softer where data points are widely spaced and stronger where they are closer together.

The application of radial basis functions to the problem of deformation gives

$$\mathbf{f}(\mathbf{p}) = \sum_{i=1}^{n} \alpha_i f(d(\mathbf{p}, \mathbf{p}_i)) + \mathbf{p}_m(\mathbf{p}) \qquad \textbf{(12)}$$

Now $\boldsymbol{\alpha}_i \in \mathbb{R}^2$ are coefficient vectors and $\mathbf{p}_m$: $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ is a function with polynomial components of degree m. Linear polynomials, where $m = 1$, give very good results—often better than higher degree polynomials. More easily, an identical transform is usually sufficient if no strong global rotations are involved. The resulting mapping function is then

$$\mathbf{f}(\mathbf{p}) = \sum_{i=1}^{n} \alpha_i (d(\mathbf{p}, \mathbf{p}_i)^2 + r_i^2)^{\mu} + \mathbf{p} \qquad \textbf{(13)}$$

We computed the image in Figure 4b using this method, with $\mu = 0.5$ and r_i from Equation 11.

A disadvantage of radial basis functions—like all global interpolation methods—is that for each pixel, all control points have to be taken into account. Thus, the algorithmic complexity is $O(nN)$, as opposed to triangulation-based methods with a complexity of roughly $O(N)$. In addition, the solution of the equation system in the preprocessing step requires an additional $O(n^3)$. However, radial basis functions have room for improve-

Table 1. Runtime measurements for the various algorithms.

Deformation Method	Average Time (in seconds)
Linear triangulation	0.12
Inverse distance, $\mu = 2$	38.3
Inverse distance, locally bounded	68.5
Multiquadrics, $\mu = 0.5$, scanline	40.4
Multiquadrics, $\mu = 0.5$, radial	23.9
Multiquadrics, $\mu = -1$, scanline	16.7
Multiquadrics, $\mu = -1$, radial	17.8
Multiquadrics, $R = 3r$	0.70
Multiquadrics, $R = r$, $T = 5$	0.26
Resampling	2.4

ment in computational efficiency.

For radial basis functions with $\mu = \pm 0.5$, $n \cdot N$ square roots have to be computed. This is undesirable even if we use a fast square-root approximation. Fortunately, our experiments show that $\mu = -1$ gives results that look just as good, as shown in Figure 4c. With this exponent, the square root is replaced by a division, which—even though it doesn't affect the computational complexity—more than doubles the speed of computation.

Additionally, if the pixel locations of the input image lie on a regular grid, and the initial locations of the control points also lie on that grid, we can greatly reduce the number of distances that must be evaluated by computing the pixel displacements per control point radially outward from the control point, so that the typically eight grid points with the same distance from the control point are treated simultaneously. The performance gain from this algorithm as compared to a straightforward scanline algorithm is, however, partially offset by the large number of boundary checks necessary.

Locally bounded radial basis functions

The algorithm outlined above, where evaluation of pixel displacements spreads outwards from the control points, also opens the door for another, more significant improvement.

As pointed out above, multiquadrics with a value of $\mu = -1$ give quite reasonable results for image deformation. These multiquadrics vanish for large distances. It therefore seems plausible to limit the range of influence of the basis functions so that evaluation stops at some distance from the control points. This way, we can beat the $O(nN)$ complexity limit.

We can limit the range of influence by multiplication with a damping function similar to Equation 4. In our experiments, we used a simpler and more efficient approach. We subtracted a constant offset δ_i from the basis functions, that is,

$$f_i(d) = ((d^2 + r_i^2)^\mu - \delta_i)_+ \quad \textbf{(14)}$$

where again $x_+ = \max(x, 0)$. This way, the influence of f_i ends at a distance

$$R_i = \sqrt{\delta_i^{1/\mu} - r_i^2} \quad \textbf{(15)}$$

from $\mathbf{p}_i$. The derivatives of this function are not defined for $d = R_i$, so $f_i \in C^0$, but for sufficiently large R_i, this does not matter in practice.

We tried two strategies for determining the R_i.

The first strategy just sets R_i to a multiple of r_i, the characteristic radius of the multiquadrics. This ensures some overlap of the influence ranges of near neighbors. However, it can lead to gaps where no deformation takes place if control points are spaced very irregularly.

The second strategy sets R_i to a distance where the influence of f_i lies below a certain threshold T. To achieve this, we first evaluate the coefficients $\boldsymbol{\alpha}_i$ with all $\delta_i = 0$. Using these estimates for $\boldsymbol{\alpha}_i$, we set $\delta_i = T / \max(|\alpha_{i,x}|, |\alpha_{i,y}|)$. This way, the influence of the modified radial basis function vanishes where the original multiquadric would have had an influence of at most T. The coefficients $\boldsymbol{\alpha}_i$ then have to be reevaluated for the modified multiquadrics with the offset δ_i.

The second strategy has one major flaw in that it is possible to get $R_i \leq 0$, which would cause the linear equation system for the $\boldsymbol{\alpha}_i$ to have no solution. We can solve this problem by specifying a minimum value for R_i. We use a value determined with the first strategy. Combining the two strategies also helps to obtain greater stability with regard to unwanted artifacts in the deformation, like foldover or local roughness.

Our experiments show that with locally bounded radial basis functions, we can achieve speedups on the order of more than 20 without noticeable loss of quality. In our example, with $R_i = 3r_i$ the result is indistinguishable from that shown in Figure 4c. Figure 4d shows the result with $R_i = r_i$ and $T = 5$. It is still quite satisfactory, even though the speed comes close to that of triangulation-based methods.

In fact, we can show that, since the data points' range of influence is now inversely proportional to their density, the time complexity of the interpolation is $O(N)$. The preprocessing still requires solving an equation system of size n, which needs time $O(n^3)$. However, with locally bounded basis functions this equation system becomes sparsely populated, so we can use iterative algorithms with better efficiency.

Performance comparisons

Table 1 shows the result of runtime measurements with the example picture used throughout this article. This is a 512×512-pixel, 256-gray-levels image, deformed with 37 control points. The measurements took place on a Sun Sparcstation10/40. We measured the user time spent on behalf of the deformation, as determined from calls to the system function `rusage` before and after the call to the deformation routine. This time measurement is relatively unaffected by system load. No file operations took place during the measurement.

The measurements show a high computational complexity of the global deformation methods relative to the more common triangulation-based method. The local method based on multiquadrics overcomes this problem without sacrificing quality. This method's performance can compete with that of linear triangulation-based methods, which provides lower quality, and is likely to be faster than the smooth method described by Goshtasby.[7]

Detailed profiling shows that very little of the time needed for deformation is spent on preparational operations like solving the equation system for multiquadrics. Almost all the time is used for actual evaluation of the transformed pixel locations.

For triangulation-based methods, we added four more control points outside the image to ensure that the convex hull of the control points covered the whole image.

For inverse distance weighted methods with $\mu = 2$, no exponentiation is necessary. Other exponents give lower performance. The locally bounded weight functions from Equation 4 also take longer, as they always require execution of a square root.

For multiquadrics with $\mu = 0.5$ we use the square-root function from the standard math library. For $\mu = -1$, no exponentiation is necessary. The result of the deformation with locally bounded radial basis functions with $\mu = -1$ and fixed $R_i = 3r_i$ is indistinguishable from that with unbounded basis functions, but takes only about 4 percent of the time to compute. The smaller range of influence apparent in Figure 4d results in an even greater speedup.

A number of possibilities exist to further increase the speed of computation. One approach would be to reduce the number of points mapped with a global method by skipping pixels and mapping the pixels in between with a simpler method, such as a bilinear interpolation.

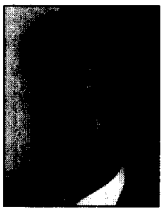
Also, if we compute displacements with a scanline algorithm, the operation to be executed is the same for all pixels. So, global interpolation methods, and radial basis functions in particular, are ideally suited for parallelization.

In an interactive environment, we could further modify locally bounded multiquadrics so that when a new control point is added, only its coefficient is evaluated to determine the desired displacement, and the complete system of equations is reevaluated only when time permits. The displacements caused by a single point can be evaluated in a small fraction of a second.

However, the time required for the deformation is already lower than the time needed for resampling, so further improvements in performance of the deformation stage do not promise a large overall improvement. The time needed for resampling the test image with a separable method appears in the last row of Table 1. ■

References

1. D. Ruprecht and H. Müller, "Deformed Cross-Dissolves for Image Interpolation in Scientific Visualization," *J. of Visualization and Computer Animation*, Vol. 5, No. 3, July 1994, pp. 167-181.
2. G. Wolberg, *Digital Image Warping*, IEEE Computer Society Press, Los Alamitos, Calif., 1990.
3. D.B. Smythe, "A Two-Pass Mesh Warping Algorithm for Object Transformation and Image Interpolation," Technical Memo 1030, Industrial Light and Magic, Computer Graphics Department, Lucasfilm Ltd., 1990.
4. T. Beier and S. Neely, "Feature-based Image Metamorphosis," *Computer Graphics* (Proc. Siggraph), Vol. 26, No. 2, July 1992, pp. 35-42.
5. A. Goshtasby, "Piecewise Linear Mapping Functions for Image Registration," *Pattern Recognition*, Vol. 19, No. 6, 1986, pp. 459-466.
6. N. Arad et al., "Image Warping by Radial Basis Functions: Application to Facial Expressions," *Computer Vision, Graphics, and Image Processing: Graphical Models and Image Processing*, Vol. 56, No. 2, March 1994, pp. 161-172.
7. A. Goshtasby, "Piecewise Cubic Mapping Functions for Image Registration," *Pattern Recognition*, Vol. 20, No. 5, 1987, pp. 525-533.
8. D. Shepard, "A Two-Dimensional Interpolation Function for Irregularly Spaced Data," in *Proc. 23rd Nat'l Conf. of the ACM*, 1968, ACM Press, New York, pp. 517-524.
9. R. Franke and G. Nielson, "Smooth Interpolation of Large Sets of Scattered Data," *Int'l J. for Numerical Methods in Engineering*, Vol. 15, 1980, pp. 1,691-1,704.
10. D. Ruprecht and H. Müller, "Free-Form Deformation with Scattered Data Interpolation Methods," in *Geometric Modelling* (Computing Suppl. 8), G. Farin, H. Hagen, and H. Noltemeier, eds., Springer Verlag, Vienna, 1993, pp. 267-281.
11. R.L. Hardy, "Multiquadric Equations of Topography and Other Irregular Surfaces," *J. of Geophysical Research*, Vol. 76, No. 8, 1971, pp. 1,905-1,915.
12. M. Eck, "Interpolation Methods for Reconstruction of 3D Surfaces from Sequences of Planar Slices," *CAD und Computergraphik*, Vol. 13, No. 5, Feb. 1991, pp. 109-120.



Detlef Ruprecht *is a senior consultant at Andersen Consulting, Germany. The research presented here was performed while he was a research assistant at the University of Dortmund. His research interests include image processing and geometric modeling as well as efficient interpolation algorithms. Ruprecht received his MS in physics from the University of Munich and his PhD in computer science from the University of Dortmund.*

Heinrich Müller *is a professor of computer science at the University of Dortmund, Germany. His research interests include scientific visualization, computer-aided geometric design, image processing, and efficient algorithms. He is the author of three books and about 60 research papers. Müller received an MS in computer science and an MS in mathematics in 1978, and a PhD in 1981, from the University of Stuttgart.*

Readers can contact Müller and Ruprecht at Universität Dortmund, Fachbereich Informatik, Lehrstuhl 7, D-44221 Dortmund, Germany, e-mail {ruprecht,mueller@ls7.informatik.uni-dortmund.de}.