Techniques for Non-Linear Magnification Transformations [*]

T. Alan Keahey[†] and Edward L. Robertson
Department of Computer Science
Indiana University
215 Lindley Hall
Bloomington, IN 47405
{*tkeahey, edrbtsn*}*@cs.indiana.edu*

Abstract

This paper presents efficient methods for implementing general non-linear magnification transformations. Techniques are provided for: combining linear and non-linear magnifications, constraining the domain of magnifications, combining multiple transformations, and smoothly interpolating between magnified and normal views. In addition piecewise linear methods are introduced which allow greater efficiency and expressiveness than their continuous counterparts.

1. Introduction

Since the introduction of the video display terminal, computer users have had to deal with the problem of limited screen space. This problem is becoming increasingly significant as modern computer systems and applications become more graphics oriented. Although screen sizes of 1280×1024 pixels are common, they are still not adequate to display many complex visual scenes.

Conversely, an argument can also be made that screen sizes have become too large. As resolution increases, more graphical information can be displayed on a single frame, oftentimes beyond an user's ability to perceive. A general purpose mechanism for allowing the user to explicitly enhance visualization of some area(s) of the screen could facilitate better focus on items of interest.

Magnification is one tool that can be brought to bear on both of these problems. This paper will discuss general, efficient techniques for performing magnification. All of the examples discussed in this paper were obtained from the FAD (*Future Acronym in Development*) toolkit being developed by the author which allows highly interactive construction and application of magnifications, as well as a programmer's interface to a library of transformation routines. All transformations provide interactive frame rates as the user moves the magnification over a domain and adjusts the magnification parameters.

2. Magnification Transformations

The available literature on distortion-based magnification abounds with a full range of terminologies to describe the techniques and characteristics of magnification/zoom/distortion/warp/etc. We borrow from Leung and Apperley [10], who make the distinction between *transformation* functions and *magnification* functions, the magnification function is the derivative of the transformation function. Figure 1 shows a simple linear *transformation* function on the left, and the corresponding *magnification* function on the right. It will be shown in this paper that it is useful to be able to translate between these two functions. Related work illustrating a technique for visualizing these transformations has been done by Furnas and Bederson [4].

2.1. Linear Transformations

The most familiar magnification technique involves the application of linear transformation functions as shown in Figure 1. The result of these transformations is a constant level of magnification across the domain, similar to what one would perceive through an ordinary magnifying glass. This has the advantage of a close analogy to familiar experiences for the user, however certain limitations are inherent:

- The user is forced to create a mapping between disjoint levels of resolution in the image. While it can be argued that the user will already be somewhat familiar with this task, it can also be argued that very rarely in the real-world does our perception of objects follow this type of behaviour: when viewing a physical object, our levels of perceived detail tend to follow

[*]Copyright 1996 IEEE. Published in the Proceedings of IEEE Visualisation '96, Information Visualization Symposium, October 28-29 1996, San Francisco, CA Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works, must be obtained from the IEEE.

[†]This work was supported by US Dept. of Education award number P200A502367.

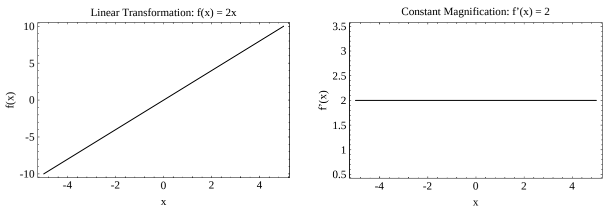

Figure 1. Transformation and Magnification

a smooth continuum, rather than jump in and out to different levels of resolution.

- If the magnified representation of some area of the screen is to appear in some other part of the screen (i.e. a separate window), then the user will be forced to make abrupt transitions on *two* cognitive/perceptual levels to tie the representations together: spatial (attention will have to shift from the normal resolution view to the higher resolution view), and resolution (the user will be forced to create a mapping between the displayed levels of resolution).
- If we use *in situ* linear magnification techniques, where the magnified image sits on top of the normal resolution image, then we introduce the problem of *occlusion*. Because of the linear nature of this magnification, the magnified representation of an image must necessarily be larger than the non-magnified representation, with the result that the magnified image will block neighboring areas of the non-magnified image.

2.2. Non-Linear Transformations

In order to overcome some of the problems with linear magnification, several different *non-linear* (or *distortion-based*) magnifications have been developed. These systems all share the properties of enhanced local resolution with some preservation of global context.

2.2.1. Fisheye Zoom

The *fisheye lens* is one mechanism for dealing with the problems incurred by linear magnification. In this schema, the user is provided with a view which is distorted in a manner similar to that produced by a very wide-angle camera lens. Furnas popularized this idea in 1986 [3], and it has been used in other systems such as Sarkar and Brown [15]. Many different methods can be used to implement the fisheye lens effect through simple geometric transformations.

2.2.2. Hyperbolic

Hyperbolic geometries provide a natural means for producing local resolution enhancement and global context. This approach allows infinite Euclidean space to be mapped into a finite disk in a continuous, non-occluding manner so that the space is "bigger" near the center of the disk, and "smaller" near the periphery. Examples of work using such geometries can be found in [9] and [12].

2.2.3. 3D Pliable Surfaces

Perspective projections of curved 3D surfaces can be used to create non-linear magnification effects [2] which allow for complex magnifications with multiple foci, and provides a potentially useful analogy for describing the magnification to the user. However, additional computational overhead is incurred in the 3D contouring and projection of essentially 2D data with this system, and special care must be taken to ensure that the projection of the surfaces does not present problems with occlusion. The techniques we present guarantee non-occlusion with simple 2D mathematics, and do not require special graphics hardware for interactive speeds. Furthermore, these methods scale up directly to viewing 3D data, although issues of occlusion may arise.

2.2.4. General Non-Linear

The above examples are special cases of the class of *non-linear transformations*. There have been a number of papers describing different approaches to this, a survey of many of these can be found in [10]. The remainder of this subsection will describe some techniques for non-linear transformations.

- **One Dimensional:** A desirable property for a 1D base function for non-linear transformations is a smoothly varying slope over some domain such that some segment(s) of the curve will have slope > 1 (areas of magnification) and some other segment(s) of the curve have slope < 1 (areas of *demagnification* or *minification*). In addition, the function chosen should be such that a parameter exists for increasing the slope of the function in some areas, and will also cause a corresponding decrease of the slope in other areas. These requirements on the function reflect the common sense notion that "you can't get something for nothing". Every extra level of magnification must incur a corresponding decrease in magnification at some other location if the range boundaries are not to exceed the domain boundaries (which is a requirement for non-occluding magnification).

  Several different choices exist for a 1D base transformation function. Sarkar and Brown [15] use the function $G(x) = \frac{(d+1)x}{dx+1}$ to perform this transformation. $G(x)$ is well behaved and has desirable non-linear characteristics over the domain $[0, 1]$, so that $G(x) : [0, 1] \rightarrow [0, 1]$. Although this function is by itself relatively inexpensive computationally, all coordinates to be transformed by this function must first be normalized or constrained to the $[0, 1]$ domain.

  Alternatively, it is possible to use a function which is well behaved over an infinite domain $h(x) : \mathcal{R} \rightarrow (-C, C)$. There are many possible functions which can be used to accomplish this, such as the hyperbolic tangent function $h(x) = tanh(x)$. To this function we can add a parameter β which controls the degree of magnification, so that $h(x, \beta) = \tanh(x\beta)$. $h(x)$ is well behaved across all domains, and there is no need to normalize the domain coordinates to some specific range before applying the magnification. Note that several functions exist which can produce a similar effect to that of $\tanh(x)$, such as a modified logistic function ($h(x, \beta) = \frac{2.0}{(1.0+e^{-2\beta x})} - 1.0$) which is computationally less expensive on many machines. It will also be shown in section 5.1 that piecewise linear functions can be used to provide a reasonable approximation. Figure 2 shows a comparison of the fisheye and tanh transformation functions. Figure 3 shows the $\tanh$ transformation function and its associated magnification function.

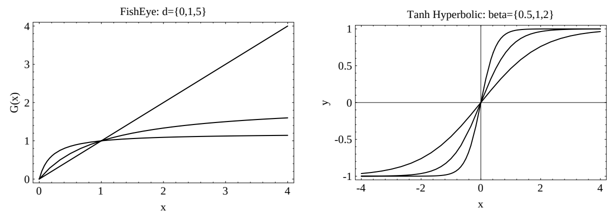

Figure 2. Fisheye and Tanh Basis Functions

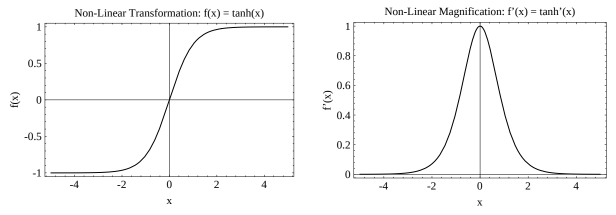

Figure 3. Tanh Transformation/Magnification

What is still required is a mechanism for moving the center of magnification across the domain. This is easily achieved: if we want x_0 to be the center of magnification, we simply replace $h(x)$ with $(h(x - x_0) + x_0)$, and the center of maximal magnification will translate to the desired location.

As will be shown in the following section, a 1D function can be used as the base function for transformations in two or more dimensions. It can also be used for magnification in one dimension only; Keahey and Marley [7] have shown that using such a "magnification bar" can be superior to scrolling for viewing highly structured text documents.

- **Two Dimensional:** By applying these simple 1D transformation functions to 2D coordinate spaces in various ways, many different effects can be produced. Figure 4 shows a few of the possible effects of applying such transformations to a regular grid of two dimensional points (as described below).

Orthogonal: this is the result of applying the 1D transformation to the x and y coordinates of a point separately, thus making the magnification in one dimension *orthogonal* to magnification in other dimensions. This transformation preserves horizontal and vertical lines within the domain, and allows for independent control of the magnification parameters for the x and y axes. It does not preserve angles relative to the center of magnification, however.

Radial (Fisheye): this effect can be produced by transforming each point in the domain as follows:

1. Given a center point of magnification H and a point to transform P, let $\hat{P} = P - H$
2. Find the radius component of the polar coordinates of $\hat{P}$ so that $r = \sqrt{\hat{P}_x^2 + \hat{P}_y^2}$
3. The new coordinates are then $H + \frac{h(r)}{r}\hat{P}$

This transformation preserves angles relative to the center of the magnification. Additionally, the fisheye effect is familiar to most users, and provides a ready analogy for the user to relate to. There is only a single magnification parameter, which controls degree of magnification in all directions. The result is the same as the polar transformation described in [6], except that no computationally expensive trig functions are needed for explicit polar-rectangular coordinate conversion.

Bi-Radial: this is a combination of the radial and orthogonal transformations. The transformation is achieved by making the *direction* of transformation the same as in the radial case, however the *magnitude* is weighted by the x and y components of the distance from the center of magnification as in the orthogonal transformation. This transformation preserves angles relative to the center of magnification, and provides some degree of independence for x and y magnification parameters.

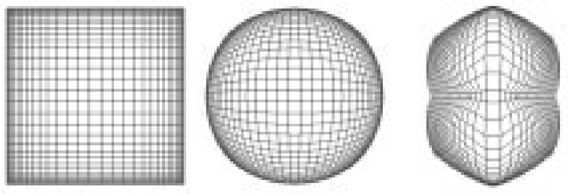

Figure 4. Orthogonal, Radial and Bi-radial

2.3. Hybrid Transformations

This section introduces techniques for combining desirable properties from linear and non-linear magnifications within a single transformation. Related work on combining these properties can be found in [2] and [16], although the methods used in these works are very different from those presented here. In particular we note the efficiency and simplicity of our methods, and that no special graphics hardware is required for interactive speeds.

2.3.1. Combined Linear/Non-Linear

An advantage of linear transformations is that they produce distortion-free zooming (when the aspect ratio is maintained). For example, a user viewing textual data would prefer to see the letters at the area of maximal magnification without the distortions presented by non-linear transformations. This is relevant with a technique used by the author [8] where arbitrary images (often containing text) are texture-mapped to a grid which is transformed via the techniques presented in this paper. As the grid points are transformed, the image is transformed along with them.

We can combine the advantages of linear and non-linear transformations by linearly transforming all points within a certain area A, and all points not falling within A are interpolated into the area surrounding the linearly magnified area of A and subsequently transformed with a suitable non-linear transformation. Note that the maximum degree of magnification that can be used in the linearly magnified area is constrained by: 1) the size of the domain to be magnified (inversely proportional), and 2) the size of the target domain (directly proportional).

Figure 11 shows what a combined linear/non-linear transformation might look like on a regular grid of points, and Figure 5 shows two examples of this type of transformation. In the left image, a round area is used for the linear magnification, with the surrounding points being treated by a radial non-linear magnification. In the right image, a rectangular area is used for the linear magnification and the surrounding points are transformed by an orthogonal non-linear magnification.

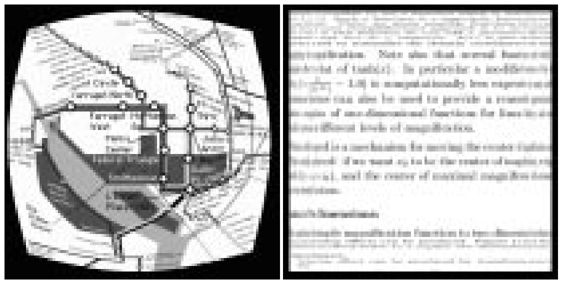

Figure 5. Combined Linear and Non-Linear

2.3.2. Constraining Transformations

Often it is the case that we do not want the transformation to apply to the entire source domain, but would rather perform non-occluding magnification on some sub-area of the domain, constraining the transformed points from that sub-area to the same sub-area. This allows for a localized, non-occluding magnification which can be moved over the source domain. This localization offers the benefits that the global context now remains more static, and the boundaries of that context remain fixed even as the center of the magnification is changed. Figure 6 shows examples of applying these constrained domains to the texture mapping example described above (note that the linear/non-linear transformation is used *within* the constrained domain), compare these with Figure 5.

The mechanism for performing these constrained domain transformations is similar to that used for the combined linear/non-linear transformations described in section 2.3.1, except that we perform the non-linear transformation *inside* the sub-area of the domain, and all points outside the domain remain untransformed. Note that it is often desirable to map the points inside the sub-area to some regular domain (e.g. $[-1, 1] \times [-1, 1]$) before performing the actual transformation on them, and then map the transformed points back into the original sub-area. These mappings are often easier to perform if we use a transformation function with a fixed range (such as $\mathcal{R} \rightarrow (-1, 1)$).

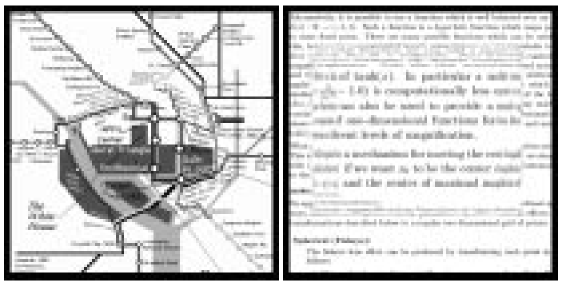

Figure 6. Constrained Domains

2.3.3. Continuous/Discrete Domains

For tasks such as graph visualization, the domain consists of discrete objects in a continuous coordinate space, and (in general) adjacencies between nodes can be maintained through rendering of edges directly between then, regardless of the type or level of distortion introduced by the transformation. For this reason, graph visualization tasks are very amenable to non-linear transformation techniques, and there are many graph visualization systems which take advantage of this [13] [15].

For discrete domains (such as the texture mapping example described in section 2.3.1), adjacency requirements are more of a concern. This is particularly the case when linear/non-linear combinations or constrained boundaries are introduced. In these cases, boundaries between regions of different transformations must be carefully thought out to preserve adjacency information. The basic task is to ensure that all mappings between regions generate boundary conditions which are consistent with the other transformation functions that are used. Referring to Figure 7, we can say that the boundary conditions at the perimeter of region A should be the same for both $f(A)$ and $g(\sim A)$.

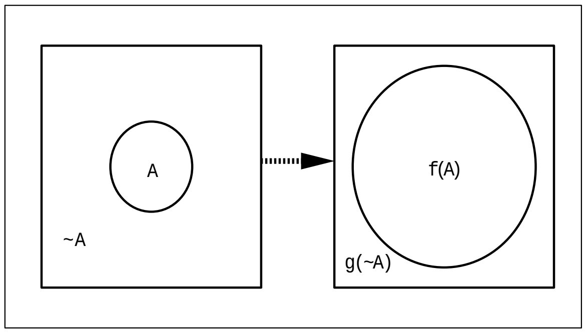

Figure 7. Boundary Conditions

3. Compound Transformations

Early work on static transformations with multiple areas of magnification began with [5]. This section will consider dynamic methods for combining transformations from multiple individual transformations. These techniques fall into three general categories, each with their own distinct questions and characteristics: 1) restricting the transformations so they do not overlap, 2) globally combining the transformations, and 3) applying the transformations in succession. The images in Figure 8 are the result of applying instances of these three approaches to a regular two dimensional grid.

Maximal Ray Clipping: In this schema, each point in the domain is transformed only by the transformation whose center of magnification is closest to that point. After the transformation has been performed, the point is clipped back along the ray between itself and the transformation center (if necessary) until the clipped point is again closer to the original transformation center than to any other transformation center. This produces a partitioning of the domain space similar to Voronoi diagrams from computational geometry, so that each transformation has its own "domain of influence". The visual result of this method is similar to that produced by the vector blending method in [2], except that with this system the boundaries between transformations are independent of the degree of magnification, and each region can be magnified or demagnified without altering the partitioning of the space.

Weighted Averaging: This is achieved by independently applying each transformation to a point in the original domain, and then making the final transformed point the weighted average of the independent transformations. The weights are inversely proportional to the distance between the center of magnification and the point in the original domain. This is similar to the arithmetic mean used in [6] and [11], except that here the weights compensate for the reduction in magnification caused by the averaging, thus enhancing both degree and localization of the individual transformations.

Composition: Here we apply each transformation in sequence to the points in the domain. This raises issues of what order the transformations should be applied in, since the magnifications will in general be non-commutative. The system used to produce the images below allowed for this through an interface to a stack of transformations, and all transformations were applied to the domain in order from the bottom to the top of the stack. The effect of this (when magnifications with constrained domains are used) is that of a stack of lenses layered on top of the display, each of which can be controlled independently.

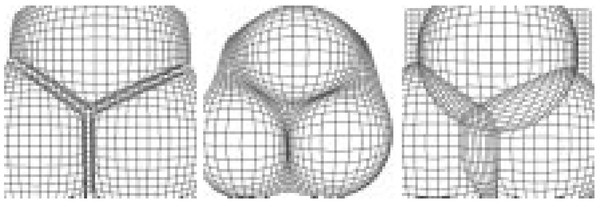

Figure 8. Clipped, Average, and Composition

4. Filtering Transformations

Independent of the complexity of the transformations employed, it is often useful to have a *single* mechanism for controlling the degree to which *all* warping transformations take effect. This can allow the user to smoothly shift between the warped and unwarped views of the space. Such functionality can be provided through a simple filter applied to the transformed and untransformed points, which provides a variable weighting of the points. Let s be the weight attached to the source point P, and d be the weight attached to the transformed point Q. We make the conditions that $0 \leq s \leq 1$ and $d = 1 - s$. The filtered point is then $sP + dQ$. Figure 9 shows an example of changing the single filter parameter to alter the effect of several transformations simultaneously. This simple method provides the user with an effective means of realizing the relationship between normal and distorted views. In addition, this technique can greatly facilitate construction of the mappings described in sections 2.3.1 and 2.3.2, points can be mapped to the extreme boundaries of a region, and the filter will smooth out the spacing between them in a non-linear fashion.

5. Piecewise Transformations

In addition to the continuous transformations described above, it is also possible to approximate these transforma-

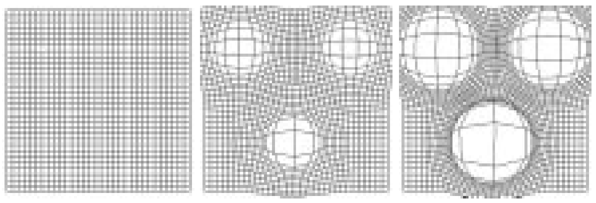

Figure 9. Filtering for s = 1.0, 0.65, 0.35

tions using piecewise linear functions. Such functions offer the potential for performance gains, and also allow for a more general class of transformations.

5.1. 1D Piecewise Transformations

These transformations involve piecewise linear approximations of the single-variable transformation basis functions which are used to drive the higher-order (2D and 3D) transformations (as described in section 2.2.4). As an example of this, a piecewise approximation of $\tanh(x)$ can be used to replace that computationally expensive function call with a simple table lookup and a linear interpolation. Since we can easily parameterize the resolution of the piecewise approximation, it is possible to produce a step function that is arbitrarily close to the continuous one, at a reduced computational cost. Table 1 shows some timing results [1] for the simple radial transformation using the 1D basis functions shown in Figure 10.

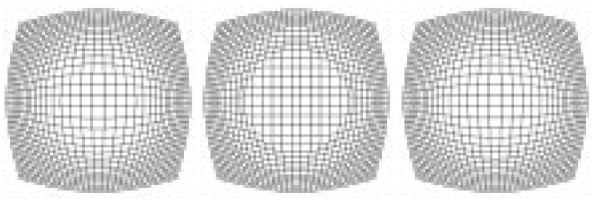

Figure 10. 1D Radial: tanh, piecewise n = 8, 16

Note that Table 1 shows the time for the piecewise approximations to be slightly slower than the fisheye transformation time, and that there is no time increase for the piecewise function as the number of components in the approximation is increased. However, the expressive power of the fisheye basis transformation is limited to altering the degree of curvature of the function based on a single parameter. With piecewise functions on the other hand, it is possible to transform points with 1D transformations of arbitrary complexity at no extra computational cost.

[1]These results were obtained on a R4400 processor with the unix times() command. Times shown are average times (in seconds) for computing a transformation on a single point. Normalized times reflect the difference between the transformation time for the distortion function and a "Null" transformation which returns the original point, thus eliminating overhead common to all transformations.

Function	Time	Normalized
Null	2.15e-06	0.0
$\tanh$	7.23e-06	5.08e-06
logistic	6.64e-06	4.49e-06
fisheye	5.47e-06	3.32e-06
1D Piecewise 8 Steps	6.05e-06	3.90e-06
1D Piecewise 16 Steps	6.05e-06	3.90e-06
1D Piecewise 32 Steps	6.05e-06	3.90e-06

Table 1. Timings for Radial Transformation

We can see a simple example of the additional expressiveness of the piecewise transformation, by considering the methods used for producing the linear/non-linear combined transformations discussed in section 2.3.1. Using a continuous warping function such as $tanh$ or the fisheye transformation, it is necessary to treat points inside the area of linear magnification as a special case. However piecewise linear transformations can automatically provide the desired area of flat magnification. Figure 11 shows a comparison of the results of these transformations (the piecewise version was obtained by using the piecewise radial transformation with 6 segments). Table 2 shows some timing results for the different methods (the final entries in this figure and table will be described in the following section). Note that the times for the piecewise function are identical to the times for the simpler step radial function in Table 1. These data clearly show a performance potential for the piecewise transformation.

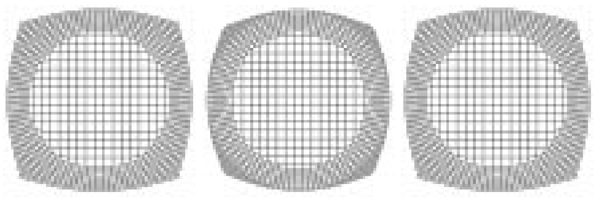

Figure 11. Flat/Radial: tanh, 1D, 2D Piecewise

Function	Time	Normalized
$\tanh$	1.30e-05	1.09e-05
fisheye	1.19e-05	9.75e-06
1D Piecewise	6.05e-06	3.90e-06
2D Piecewise	7.42e-06	5.27e-06

Table 2. Timings for Flat/Radial Transform

In addition to the performance gains offered by such functions however, piecewise functions generalize much more readily to arbitrary shapes. In general we can say that a suitable piecewise transformation function is one composed of linear segments having zero-order parametric continuity C^0 (i.e. the endpoints of adjacent segments meet). The degree of magnification provided by the function is directly proportional to the slope of the segment at that point. There are several ways in which such functions can be constructed. The simple piecewise functions used in this paper were all generated through routines which sample a continuous function at a parameterized resolution to obtain the resulting piecewise function.

It is also possible to provide an interface which allows a user to drag control points on the piecewise transformation function and thus design customized functions with multiple areas of magnification. Not all functions constructed in this manner will be useful, the loose restriction of C^0 continuity will allow the user to construct wildly distorting transformations with sharp transitions rather than smooth curves. This problem can be alleviated by using spline functions with higher degrees of continuity to construct a "transformation curve", and then sampling the spline function to obtain an approximating piecewise linear function.

Piecewise transformations can also be constructed through manipulation of the values of the magnification function (which is the derivative of the transformation function). By placing a grid over some domain and assigning suitable magnification values (perhaps corresponding to a degree of interest) for each cell in grid, we can integrate over the magnification values to obtain the corresponding transformation function. This allows the user to construct a transformation function without having to be aware of the slope/magnification relation, instead the user is able to construct such a function by simply expressing interest in certain areas of the domain. Since we are integrating the magnification values to obtain the transformation function, we automatically obtain C^0 continuity in the resulting transformation function. As with the transformation functions, this process is also amenable to the use of spline functions for direct construction.

5.2. 2D Piecewise Transformations

In the previous subsection, we explored methods for using piecewise linear functions to approximate the 1D basis functions described in section 2.2.4. In this subsection, we will examine a more powerful set of piecewise transformations, where an approximation is made of a complete two dimensional transformation.

The most straightforward method to construct a 2D piecewise linear transformation of this type is to sample an existing transformation with a regular grid of two dimensional points to construct a grid of the transformed points. After this grid has been computed, any number of points can be transformed through table lookup and linear interpolation on the x and y coordinates independently. The accuracy of the piecewise approximation is controlled by parameterizing the resolution of the "sampling grid" that is used.

Table 2 shows that even for *slightly* complicated transformations such as the Flat/Radial combination (Figure 11), construction of a 2D piecewise transformation can result in significantly faster transformation times than for the procedural fisheye technique. As with the 1D piecewise transform, the time required for the complex piecewise transformation remains constant with increasing complexity of the function, and increasing grid size (within memory and cache constraints). Note however that the 1D piecewise approach is the fastest of the three, requires much less memory than the 2D, and would be a better choice for this particular transformation.

When we moved from continuous basis transformation functions to their 1D piecewise transformation counterparts, we found that a new level of expressiveness became possible. Similarly, we can express an even richer set of transformations with a single 2D piecewise transformation function than with a continuous 2D transformation. There are several ways in which this potential increase in expressiveness can be realized. One method that offers large performance benefits is the ability to combine multiple transformations (as described in section 3) and express them in terms of a single 2D piecewise transformation. This can be achieved in a manner similar to that described above for construction from a single existing transformation, the difference being that the entire set of transformations is applied to the sampling grid before it is used to construct the single 2D piecewise transformation grid. The timing results shown in Table 3 indicate better than a magnitude of order increase in performance when comparing the procedural (fisheye) and 2D piecewise approaches to producing the transformation shown in Figure 12.

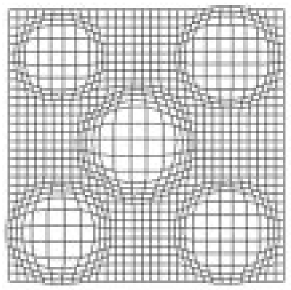

Figure 12. Multiple Transform for Table 3

Function	Time	Normalized
Null	4.59e-06	0.0
Fisheye	7.10e-05	6.64e-05
2D Piecewise	1.02e-05	5.61e-06

Table 3. Times for Multiple and 2D Piecewise

Because of the overhead involved in construction of 2D piecewise transformations, they are best suited to tasks which are dynamic in time and placement rather than in shape. Once computed, these 2D transforms can be translated over a domain (or the domain "underneath" the transform can change through time) efficiently without recomputing the transformation grid. Whenever the *shape* of the transformation changes however, it becomes necessary to recompute the transformation grid, thus 1D piecewise or continuous functions might be better suited to those applications where the shape of the transformation changes frequently.

6. Conclusions

We have shown several techniques for constructing and using non-linear magnification, and that it is possible to construct sophisticated transformations by building up from simpler ones. A hierarchy of transformations emerges from these constructions which allows for categorization of new classes of transformations. Some of the construction tools that we have described include: combination with linear magnifications, constrained transformation domains, combining multiple transformations, and enhanced control of the overall degree to which transformations should take effect. All of these methods use simple and efficient two dimensional mathematics to produce occlusion-free magnification.

We have also shown how piecewise linear transformations can be used to approximate continuous non-linear transformations of one or two dimensions. Timing data clearly indicate a potential for increased efficiency with these piecewise transformations. In addition, a greater degree of expressiveness becomes possible with these functions, which provides greater flexibility for the designer of more sophisticated magnification systems.

7. Further Work

General methods for enforcing the consistency of domain boundary conditions in aggregate transformations should be addressed for domain constraints and combinations of transformations. More work is currently being done on effective techniques for constructing the piecewise linear transformations. We are investigating interactive methods for construction of these transforms which will be more natural for an user to deal with. Also, several issues arise in the conversion between the transformation and magnification grids for the 2D piecewise functions. Different conversion techniques exist, each giving different results. A significant area of work centers on the question of how to best utilize the additional expressiveness possible with these piecewise transformations.

References

[1] B. B. Bederson and J. D. Hollan. Pad++: A zooming graphical interface for exploring alternate interface physics. In *UIST '94 Proceedings*, 1994.

[2] M. Carpendale, D. Cowperthwaite, and D. Fracchia. 3D pliable surfaces: For the effective presentation of visual information. In *UIST '95 Proceedings*, pages 217–226, 1995.

[3] G. W. Furnas. Generalized fisheye views. *Human Factors in Computing Systems, CHI '86*, pages 16–23, April 1986.

[4] G. W. Furnas and B. B. Bederson. Space-scale diagrams: Understanding multiscale interfaces. In *Proceedings of CHI '95 Human Factors in Computing Systems*, 1995.

[5] N. Kadmon and E. Shlomi. A polyfocal projection for statistical surfaces. *Cartograph. J.*, 15(1):36–41, 1978.

[6] K. Kaugars, J. Reinfelds, and A. Brazma. A simple algorithm for drawing large graphs of small screens. In *Proc. of Graph Drawing '94; Lecture Notes in Computer Science 894*, pages 278 – 282, 1994.

[7] T. A. Keahey and J. Marley. Viewing text with non-linear magnification: An experimental study. Technical Report 459, Department of Computer Science, Indiana University, April 1996.

[8] T. A. Keahey and E. L. Robertson. Non-linear image magnification. Technical Report 460, Department of Computer Science, Indiana University, April 1996.

[9] J. Lamping, R. Rao, and P. Pirolli. A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In *CHI '95 Proceedings*, 1995.

[10] Y. Leung and M. Apperley. A review and taxonomy of distortion-oriented presentation techniques. *ACM Transactions on Computer-Human Interaction*, 1(2):126–160, 1994.

[11] K. Misue and K. Sugiyama. Multi-viewpoint perspective display methods: Formulation and application to compound graphs. In *Human Aspects in Computing: Design and Use of Interactive Systems and Information Management*, pages 834 – 838. Elsevier Science Publishers, 1991.

[12] T. Munzner and P. Burchard. Visualizing the structure of the world wide web in 3D hyperbolic space. In *VRML '95 Proceedings*, 1995.

[13] E. G. Noik. Exploring large hyperdocuments: Fisheye views of nested networks. In *Hypertext '93: ACM Conference on Hypertext and Hypermedia*, 1993.

[14] G. Robertson and J. D. Mackinlay. The document lens. In *UIST '93: Proceedings of the ACM Stmposium on User Interface Software and Technology*, pages 101–108, 1993.

[15] M. Sarkar and M. H. Brown. Graphical fisheye views. *Communications of the ACM*, 37(12):73–84, 1994.

[16] M. Sarkar, S. S. Snibbe, O. Tversky, and S. P. Reiss. Stretching the rubber sheet: A metaphor for visualizing large layouts on small screens. In *UIST '93, Proceedings of the ACM User Interface Software and Technology*, 1993.

[17] R. Spence and M. D. Apperley. Data-base navigation: An office environment for the professional. *Behaviour and Information Technology*, 1(1):43–54, 1982.