SPATIAL COGNITION AND COMPUTATION, *3*(2&3), 207–222

Buffering Fuzzy Maps in GIS

Hans W. Guesgen
University of Auckland

Joachim Hertzberg
Fraunhofer Institute for Autonomous Intelligent Systems, Germany

Richard Lobb
University of Auckland

Andrea Mantler
University of North Carolina

In this paper, we show how standard GIS operations like the complement, union, intersection, and buffering of maps can be made more flexible by using fuzzy set theory. In particular, we present a variety of algorithms for operations on fuzzy raster maps, focusing on buffer operations for such maps. Furthermore, we show how widely-available special-purpose hardware (in particular, z-buffering in graphics hardware) can be used for supporting buffer operations in fuzzy geographic information systems (GIS).

Keywords: Geographic information systems, raster maps, buffering, fuzzy logic

1 Introduction

Geographic information systems (GIS) have been in use for quite a while now (Coppock and Rhind, 1991), but their functionality has changed only little over

Correspondence concerning this article should be addressed to Hans W. Guesgen, Department of Computer Science, University of Auckland, PMB 92019, Auckland, New Zealand; email hans@cs.auckland.ac.nz

the years. In spite of being called *geographic*, GIS have so far been mostly geometric in nature, ignoring the temporal, thematic, and qualitative dimensions of geographic features (Molenaar, 1996; Sinton, 1978; Usery, 1996). There are, however, several attempts to overcome these limitations and to incorporate qualitative (Egenhofer and Golledge, 1997; Frank, 1994; Frank, 1996; Peuquet, 1994) or fuzzy aspects (Altmann, 1994; Brimicombe, 1997; Molenaar, 1996; Plewe, 1997) in GIS. This paper focuses on the latter aspect, extending work that has been published previously (Guesgen and Hertzberg, 2000a; Guesgen and Hertzberg, 2000b; Guesgen and Hertzberg, 2001; Guesgen et al., 2001).

An essential operation in GIS is map overlay, where new maps are computed from existing ones by applying either:[1]

- Buffer operations, which increase the size of an object by extending its boundary, or
- Set operations, such as complement, union, and intersection.

In traditional GIS, these operations are exact quantitative operations. Humans, however, often prefer a qualitative operation over an exact quantitative one, which can be achieved by extending the standard map overlay operations to fuzzy maps and using fuzzy logic rather than crisp logic.

Consider, for example the simple raster maps in Figure 1. The first row shows raster maps for the location of roads (a-i), the location of water (a-ii), locations of residences (a-iii), and the location of native forest (a-iv) in a fictional region. The second row is an illustration of crisp buffering operations. It contains raster maps illustrating roads buffered by 200m (b-i), rivers buffered by 400m (b-ii), residences buffered by 200m (b-iii), and unbuffered native forest areas (b-iv). The third row shows the result of fuzzy buffering operations: areas close to roads, obtained with a cone buffer function (Guesgen et al., 2001) of radius 400m (c-i), areas close to rivers, obtained with a cone buffer function of radius 800m (c-ii), areas close to residences, obtained with an inverse-distance buffer function (c-iii), and a probability density distribution (Scott, 1992) based on the recorded location of native forest, obtained with a 50m-radius gaussian distribution (c-iv). Dark areas are areas of high membership, while light areas are those of low membership.

The last row illustrates how the fuzzy maps can be combined to find areas close to roads, not close to rivers or residences, and not on native forest. Such areas might be required for an industrial development, for example. The boolean classification of the crisp map (d-i) would generally be much less useful for decision making than the fuzzy map (d-ii), in which darkness (i.e., the membership grade) increases with suitability. The 3D membership surface illustration (d-iii) provides an alternative view of the fuzzy map (d-ii). Here, the membership grade

[1] A more detailed introduction to these operations can be found elsewhere (Guesgen and Histed, 1996).

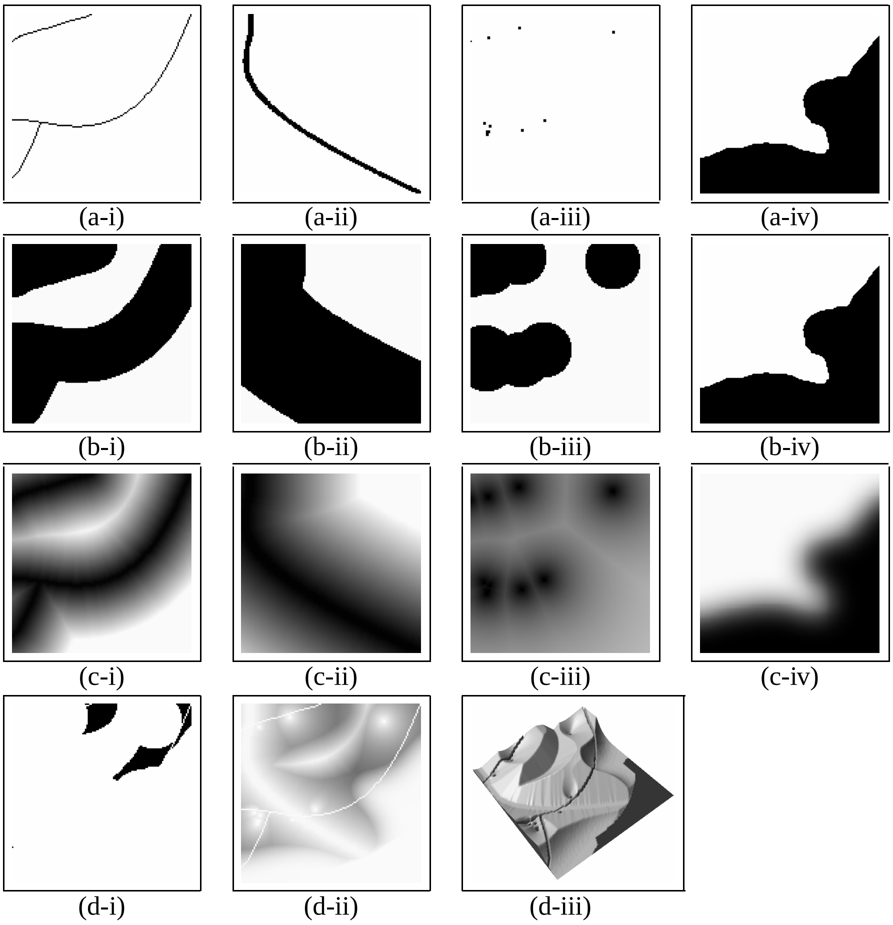

Figure 1. An illustration of buffering in a fuzzy GIS (Duff and Guesgen, 2002).

is represented by terrain height rather than greyscale (which is irrelevant in this illustration).

The rest of this paper is organized as follows. We start with a brief review of fuzzy maps and the definition of set operations for fuzzy maps; a more detailed introduction can be found elsewhere (Guesgen and Albrecht, 1998). We then continue with the introduction of various algorithms for buffering fuzzy maps, which is an extension of previously published work (Guesgen and Hertzberg, 2000a). Finally, we show how graphics hardware can be used to implement buffering algorithms more efficiently.

2 Fuzzifying Maps

In the following, we restrict ourselves to raster-based maps. Such a map consists of a grid of cells whose values specify certain attributes of the locations represented by the map. In the simplest case, the cell values are restricted to 0 and 1, where 0 signals the absence and 1 the presence of a certain attribute, like the attribute of a location being part of a road, waterway, residential area, commercial area, rural area, etc.

In some cases there is a crisp boundary between locations that have a certain attribute and those that do not have that attribute, but often this is not the case. For example, it is not always clear where a rural area stops and a residential area starts, or where a forest is not a forest any more. To cater for this fact, we extend the range of cell values from the set $\{0, 1\}$ to the interval $[0, 1]$, and thereby convert a regular raster map into a fuzzy raster map. Given a cell l in the fuzzy raster map, $\mu(l) \in [0, 1]$ indicates the degree to which l has the attribute represented by the map. The function $\mu(l)$ is called the membership function of the fuzzy raster map.

Performing a set operation (complement, union, and intersection) on fuzzy raster maps is straightforward. There are several ways of defining the complement, union, and intersection of membership functions (Driankov et al., 1996), but they all have in common that they are defined cell-wise for all cells L in the fuzzy raster map. In the case of the original max/min scheme (Zadeh, 1965), the membership functions for the complement, union, and intersection are defined as follows, where μ_1 and μ_2 denote the membership functions of the underlying maps and μ_3 the one of the resulting map:

Complement:	$\forall l \in L : \mu_3(l) = 1 - \mu_1(l)$
Union:	$\forall l \in L : \mu_3(l) = \max\{\mu_1(l), \mu_2(l)\}$
Intersection:	$\forall l \in L : \mu_3(l) = \min\{\mu_1(l), \mu_2(l)\}$

Since the membership functions for the complement, union, and intersection are defined cell-wise, an algorithm for performing a set operation on fuzzy raster maps can just iterate through the set of cells and compute a new value for each cell based on the given value(s) for that cell, which means the algorithm is linear in the number of cells, i.e, its complexity is $O(|L|)$.

3 Iterative Buffering of Fuzzy Maps

Unlike the set operations, buffer operations cannot be defined cell-wise. To determine the new value of a cell l in a crisp raster map, the values of all cells in the neighborhood of l are considered. If at least one of these values is 1, then the value of l is changed to 1; otherwise it remains unchanged. In other words, the new value of l is the maximum of the original value of l and the values of all cells in the neighborhood of l. A fuzzy raster map can be buffered crisply in a similar way: the value of l is changed to the maximum fuzzy value in the neighborhood of l, which might be any value from the interval $[0, 1]$ (rather than the set $\{0, 1\}$).

Although buffering a fuzzy raster map as indicated above might be of use for many applications, we do not want to restrict ourselves to crisp buffer operations for fuzzy raster maps. Rather, we want the buffer operation to depend on the proximity of the cells under consideration. For example, if there is an area on the map with very high membership grades, then the buffer operation should assign high membership grades to cells that are very close to that area, medium high membership grades to cells close to the areas, and low membership grades to cells further away.

One way to achieve this behavior is to determine the direct neighbors of a cell and to apply a buffer function to determine the new membership grade of these neighbors. There are two types of direct neighbors:

- Edge-adjacent (4-adjacent) neighbors, or edge neighbors for short. Two cells of the grid are edge neighbors if and only if they have an edge in common.
- Vertex-adjacent (8-adjacent) neighbors, or vertex neighbors for short. Two cells of the grid are vertex neighbors, if and only if they have a vertex in common.

A buffer function is a monotonically increasing function $\beta : [0, 1] \to [0, 1]$ whose value never exceeds its input:

$$\forall m \in [0, 1] : \beta(m) \leq m$$

An example of a simple buffer function is $\beta(m) = \max\{0, m - 0.1\}$.

If l_0 is a neighbor of l_1, then the new membership grade of l_1 is determined by the maximum of the old membership grade of l_1 and the value of the buffer function applied to the membership grade of l_0:

$$\mu(l_1) \leftarrow \max\{\mu(l_1), \beta(\mu(l_0))\}$$

Brute-Force β-Buffering

Let μ be the membership function of the map.
Let β be a buffer function.
Let L be the set of all cells in the map to be buffered.
Repeat until μ is stable:
  For each $l_0 \in L$ do:
    For all neighbors l_i of l_0 do:
      $\mu(l_i) \leftarrow \max\{\mu(l_i), \beta(\mu(l_0))\}$

Figure 2. A brute-force algorithm for β-buffering raster fuzzy maps.

Since updating the membership grade of l_1 can have an impact on the membership grades of the neighbors of l_1, the update process has to be repeated for all cells of the map over and over again until a stable situation is obtained. In the following, we refer to the process of buffering a fuzzy raster map, using a buffer function β as defined above, as *iterative buffering*, or *β-buffering*.

A brute-force algorithm for β-buffering is shown in Figure 2. The algorithm visits each cell of the map and updates its membership grade based on the membership grades of the neighboring cells.[2] If any of the membership grades is changed, the algorithm repeats the updating process until all membership grades become stable. More precisely, the algorithm applies the buffer function β to the membership grade $\mu(l_0)$ of a cell l_0 and uses the result to update the membership grades of the four edge neighbors, or the eight vertex neighbors, of l_0, respectively. Since the maximum operator is commutative and associative, the order in which the cells are updated does not have an impact on the final result of the updating process.

Since the algorithm revisits each cell when repeating the updating process, even the ones whose neighbors have not been changed in the previous iteration, it performs many unnecessary checks. An improved approach is to keep track of the changed cells and to revisit a cell only if at least one of its neighbors has been changed. The algorithm in Figure 3 achieves this by applying the principle of local propagation: the membership grade of a cell is propagated to the neighbors of the cell, which are then put on to the list of cells to be visited in the future.

The local propagation algorithm is guaranteed to terminate: since the value of the buffer function never exceeds its input, we cannot get a cyclic set of updates.

[2] The membership grades of the neighboring cells are lower bounds for the new membership grade.

β-Buffering by Local Propagation

Let μ, β, and L be defined as before (Figure 2).
While $L \neq \emptyset$ do:
  Select $l_0 \in L$.
  $L \leftarrow L - \{l_0\}$
  For all neighbors l_i of l_0 do:
    $\mu(l_i) \leftarrow \max\{\mu(l_i), \beta(\mu(l_0))\}$
    If $\mu(l_i)$ has changed, then $L \leftarrow L \cup \{l_i\}$

Figure 3. A local propagation algorithm for β-buffering fuzzy raster maps.

At worst, a single cell can receive $|L| - 1$ updates, which correspond to paths of updates originating at each of the other cells in the map. (Also note that a path of updates cannot be longer than $|L| - 1$ cells.)

Although the propagation algorithm is guaranteed to terminate, it can be rather inefficient: many cells may be revisited repeatedly as their membership grades are overwritten by successively larger values. To prevent this from happening, we can select a cell l_0 from L with a maximum membership grade. The grade for l_0 cannot be increased by any buffer operation $\beta(m)$, since $\beta(m) \leq m$ for all $m \in [0, 1]$, which means that the current grade of l_0 is the final membership grade for that cell. Since the membership grade of l_0 is both final and maximal, buffering the neighbors of l_0 results in assigning a final membership grade to the neighbors of l_0 as well. This means that none of the neighbors have to be revisited. The improved algorithm is shown in Figure 4.

4 From Iterative Buffering to Global Buffering

So far, our discussion of algorithms revolved about a buffer function β. Although propagating the result of β locally through a fuzzy raster map is a reasonable way to buffer such a map, it is not ideal for global effects, since the membership grade of a cell is determined by its original membership grade and the grade of its immediate neighbors, but not by the membership grade of cells further away from the cell under consideration. To achieve a more global effect, we replace β with a global buffer (or proximity) function ψ that is applied not only to the membership grades of the neighbors of a given cell l_0 but potentially to any cell l in the map. The function ψ has two arguments, one of which is $\mu(l_0)$, the membership grade

β-Buffering with Ordered Cells

Let μ, β, and L be defined as before (Figure 2).
While $L \neq \emptyset$ do:
    Select $l_0 \in L$ such that $\mu(l_0)$ is maximal in L.
    $L \leftarrow L - \{l_0\}$
    For all neighbors l_i of l_0 do:
        $\mu(l_i) \leftarrow \max\{\mu(l_i), \beta(\mu(l_0))\}$

Figure 4. An algorithm for β-buffering fuzzy maps using ordered cells.

of l_0, and the other is $\delta(l, l_0)$, the distance between l and l_0, which can be defined as follows:

1. $\delta(l_0, l_0) = 0$
2. $\forall l \neq l_0$:
   $\delta(l, l_0) = \min\{\delta(l', l_0) \mid l'$ neighbor of $l)\} + 1$

We require ψ to be monotonically increasing in the first argument, i.e., the larger the membership grade of l_0, the larger the value of ψ, and monotonically decreasing in the second argument, i.e., the further away l is from l_0, the smaller the value of ψ. We further require that the value of ψ never exceeds the value of the first argument:

$$\forall m \in [0, 1] \text{ and } \forall d \in [0, \infty) : \psi(m, d) \leq m \qquad (1)$$

The update of a membership grade is computed in a similar way as before:

$$\mu(l) \leftarrow \max\{\mu(l), \psi(\mu(l_0), \delta(l, l_0))\}$$

In addition, we have to ensure that the resulting membership grades are intuitively plausible. In particular, we want to avoid having a local effect override a more global one if they originate in the same cell. For example, if a cell l_0 has a distance of 1 to a cell l_1 and a distance of 2 to a cell l_2, then $\psi(\psi(\mu(l_2), 1), 1)$ should not exceed $\psi(\mu(l_2), 2)$, i.e., the new membership grade of l_0 is influenced by the membership grade of l_2 directly rather than the propagation of that membership from l_2 through l_1 to l_0. We can enforce this property by requiring:

$$\begin{aligned} &\forall m \in [0, 1] \text{ and } \forall d_0, d_1, d_2 \in [0, \infty) : \\ &d_2 = d_1 + d_0 \Longrightarrow \psi(m, d_2) \geq \psi(\psi(m, d_1), d_0) \end{aligned} \qquad (2)$$

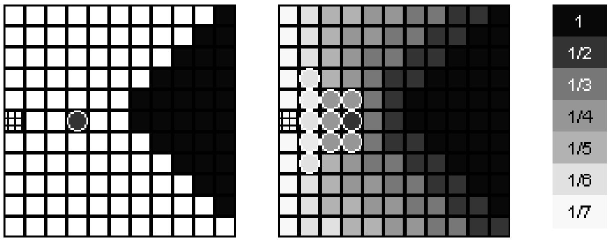

Figure 5. A raster map with its original fuzzy membership grades and its buffered version. Greyscale shades indicate membership grades. A white circle in the buffered map denotes a cell that received its membership grade directly or indirectly from the circled cell in the original map. Where two membership grades overlap, a larger value has precedence over a smaller one.

The buffer function $\psi(m,d) = \frac{m}{1+d}$, for example, satisfies this criterion, whereas $\psi(m,d) = \frac{m}{1+d^2}$ does not.

If we require equality instead of inequality in Formula (2), we achieve the same effect as with the function β as introduced in Section 3. If $\psi(m,d_2) = \psi(\psi(m,d_1),d_0)$, then the new membership grade of a cell l with distance d from cell l_0 can be computed by applying ψ successively to the membership grade of l_0, i.e., by defining $\beta(m) \equiv \psi(m,1)$:

$$\mu(l) \leftarrow \max\{\mu(l), \underbrace{\psi(\psi(\ldots\psi}_{d}(\mu(l_0),1)\ldots),1)\}$$

Figure 5 shows a fuzzy map being buffered, using $\psi(m,d) = \frac{m}{1+d}$ as the proximity function and using a distance measure based on vertex adjacency. The original map has only membership grades of 0 (unfilled white cells), except for: (A) the filled black cells on the right of the map, which have a membership grade of 1 and (B) the single dark grey cell with a circle, which has a membership grade of $\frac{1}{2}$. An interesting effect occurs at the cell with the grid in Figure 5. This cell is closer to the single cell of Object B than to any cell in Object A. However, the effect of buffering Object A overtakes the effect of buffering Object B due to the larger membership grade of Object A:

$$\psi(\tfrac{1}{2},3) = \tfrac{1}{8} \quad < \quad \psi(1,6) = \tfrac{1}{7}$$

A brute-force algorithm for global buffering (also referred to as *ψ-buffering*) a fuzzy map using a global buffer function ψ can be obtained by extending the update operations in the algorithm of Figure 2 to all cells in the map. The resulting algorithm is shown in Figure 6. The algorithm repeatedly iterates through the set of cells, using the membership grades of a cell to update the membership grades of

Brute-Force ψ-Buffering

Let μ be the membership function of the map.
Let ψ be a global buffer function.
Let L be the set of all cells in the map to be buffered.
Repeat until μ is stable:
  For each $l_0 \in L$ do:
    For all $l \in L - \{l_0\}$ do:
      $\mu(l) \leftarrow \max\{\mu(l), \psi(\mu(l_0), \delta(l, l_0))\}$

Figure 6. A brute-force algorithm for ψ-buffering fuzzy raster maps.

the other cells. This is done regardless of whether the membership grade of a cell can possibly have an effect on other cells or not. An improvement can be achieved by using only those cells that have the potential to influence other cells. This is the case if the current membership grade of the cell is not minimal and was not derived from the membership grade of another cell through buffering. Cells with minimal membership grade cannot increase the membership grade of another cell during buffering, because the buffer operation always returns a value smaller than or equal to the membership grade of the cell that is used as argument of the buffer operation (cf. Formula (1)). A cell whose membership grade was derived from the membership grade of another cell through buffering cannot make any contribution because the other cell has spread its influence to all cells of the map already, and since global effects dominate local ones (cf. Formula (2)), the current membership grade of the cell under consideration does not have any additional effect.

Figure 7 shows an improved algorithm, which restricts the outer loop to the set of cells that might have an influence on other cells. Initially, this set contains all cells of the map. However, when a cell is detected whose membership grade is updated through a buffer operation, the cell that was updated is removed from the set of influential cells, because it won't have any effect on the membership grades of other cells in a future iteration. In addition to that, the cells to be buffered are selected according to their membership grades. Cells with large membership grades are more likely to cause a cutoff than those with smaller grades. It therefore makes sense to consider cells with large membership grades first.

ψ-Buffering with Ordered Cells and Cutoffs

Let μ, ψ, and L be defined as before (Figure 6).
$L' \leftarrow L - \{l \mid \mu(l)$ is minimal in $L\}$
While $L' \neq \emptyset$ do:
  Select $l_0 \in L'$ such that $\mu(l_0)$ is maximal in L'.
  $L' \leftarrow L' - \{l_0\}$
  For all $l \in L - \{l_0\}$ do:
    $\mu(l) \leftarrow \max\{\mu(l), \psi(\mu(l_0), \delta(l, l_0))\}$
    If $\mu(l)$ has changed, then $L' \leftarrow L' - \{l\}$

Figure 7. An algorithm for ψ-buffering fuzzy raster maps using ordered cells and cutoffs.

5 Using Graphics Hardware for Buffering

So far, we have focussed on improving buffering by applying various software techniques and heuristics. In the rest of this paper, we adopt a different course: we describe an implementation of the brute-force algorithm that is efficient because we use special-purpose hardware. Since a fuzzy map can be viewed as a two-dimensional pixel image in which the colors represent the different membership grades of the fuzzy map, it is not surprising that techniques from the area of computer graphics can be used to buffer fuzzy maps more efficiently. (Hoff et al., 1999) suggest using graphics hardware to compute generalized Voronoi diagrams. (Mustafa et al., 2001) use hardware generated Voronoi diagrams as the basis for map simplification. We also adapt the idea of (Hoff et al., 1999) and will show how such hardware (in particular, the z-buffer of the hardware) can be used to buffer fuzzy maps.

The z-buffer (or depth buffer) is similar to the frame buffer in that it stores information for each pixel of the image. The value stored in the z-buffer is the depth of the closest object found so far that covers that pixel. Before a pixel is given the color of a new object, the depth of the object at that particular pixel is computed and compared with the depth stored in the z-buffer. If the new object is closer, its color will be stored in the frame buffer and its depth value in the z-buffer. Figure 8 shows this approach in algorithmic form.

In the context of buffering fuzzy maps, we use the z-buffer to mimic the current fuzzy map, i.e., we take the depth $\zeta(l_0)$ of a given pixel l_0 to uniquely represent

Z-Buffer Algorithm

Let L be the set of pixels of the image.
Let $\zeta(l)$ be the value of the z-buffer for the pixel $l \in L$.
Let $\pi(l)$ be the value of the frame buffer for the pixel $l \in L$.
Let O be the set of objects to be rendered.
For all $o \in O$ do:
  For each $l \in L$ covered by o do:
    Let z be the depth of o at l.
    Let p be the color of o at l.
    If $z < \zeta(l)$, then do:
      $\zeta(l) \leftarrow z$
      $\pi(l) \leftarrow p$

Figure 8. An algorithm showing the use of the z-buffer in graphics hardware.

the membership grade $\mu(l_0)$. A close depth indicates a high membership grade, whereas a far depth stands for a low membership grade.

To buffer the membership grade represented by the depth of the pixel, we render an object that approximates the buffer function ψ applied to $\mu(l)$. If we restrict ourselves to buffer functions of the type $\psi(m, d) = \max\{0, m - kd\}$, where k determines how fast the original membership m diminishes with the distance d, the object to be rendered is a right circular cone. Cones expand away from the camera, and thus the depth of the cone is determined by $\zeta(l) = (1 - \psi(\mu(l_0), \delta(l, l_0)))$. A membership grade of 1 is mapped to a depth of zero, and a membership grade of 0 is mapped to a depth of 1. Note that it is possible to use different values of k within the same map to obtain different buffering effects for the objects represented in the map. Beyond that, cones of a different shape (not necessarily with constant slopes) can be used to model other buffer functions.

To speed up the rendering process (i.e., to make the buffering process more efficient), we approximate each buffering cone as a triangle fan, as shown in Figure 9. To further speed up the process, we suppress the rendering process for certain pixels, if there are regions with equivalued pixels. In this case, we only need cones from the boundary pixels.

As mentioned in Section 3, distance is sometimes defined in a grid structure through neighborhood relationship. In the case of vertex neighbors, this means that our buffer function ψ can be represented by a cone with four triangles (i.e., a

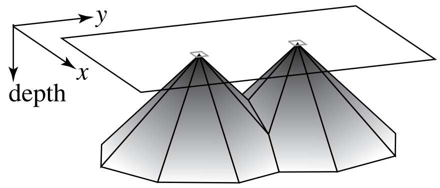

Figure 9. $\zeta(l_i)$ approximated by triangle fans.

pyramid) that is aligned parallel to the grid structure. In the case of edge neighbors, this is also possible, but the pyramid is rotated by 45 degrees.

In general, we cannot guarantee the soundness of hardware buffering, as the rendered objects are only approximations of the buffer function ψ.[3] On the other hand, there are experiments showing that fuzzy membership grades are quite robust, which means that it is not necessary to have exact membership grades (Bloch, 2000). The explanation given for this observation is twofold: first, fuzzy membership grades are used to describe imprecise information and therefore do not have to be exact, and second, each individual fuzzy membership grade plays only a minor role in the whole reasoning process, as it is usually combined with several other membership grades.

6 Evaluation of the Algorithms

The brute-force algorithm of Figure 6 iterates through the set of cells L, using the membership grades of a cell to update the membership grades of the other cells. Since this is done regardless of whether the membership grade of a cell can have an effect on other cells or not, the algorithm has an average complexity of $O(|L|^2)$.

The improved algorithm of Figure 7 still has the worst-case time complexity of $O(|L|^2)$, since it can happen that the algorithm does not change any membership grade and therefore has to iterate through all cells of the map. The time complexity is bound from below by the time complexity of selecting cells in the order of their membership grades. If the membership grades are discrete, bucket sorting can be used to sort the list beforehand, which makes sorting and selecting the cells linear; otherwise, sorting and selecting is $O(|L| \cdot \log |L|)$. In practice, the sorting time is negligible and the quadratic properties of the algorithm dominate on most data.

The worst-case time complexity for buffering a fuzzy map using graphics hardware is $O(|L|^2)$ in general and $O(|L|)$ for certain special cases (Duff and Guesgen,

[3]There are special cases where the result is identical with the one obtained through the software buffering algorithms (like the buffer function $\psi(m, d) = \max\{0, m - kd\}$, which corresponds to a right circular cone).

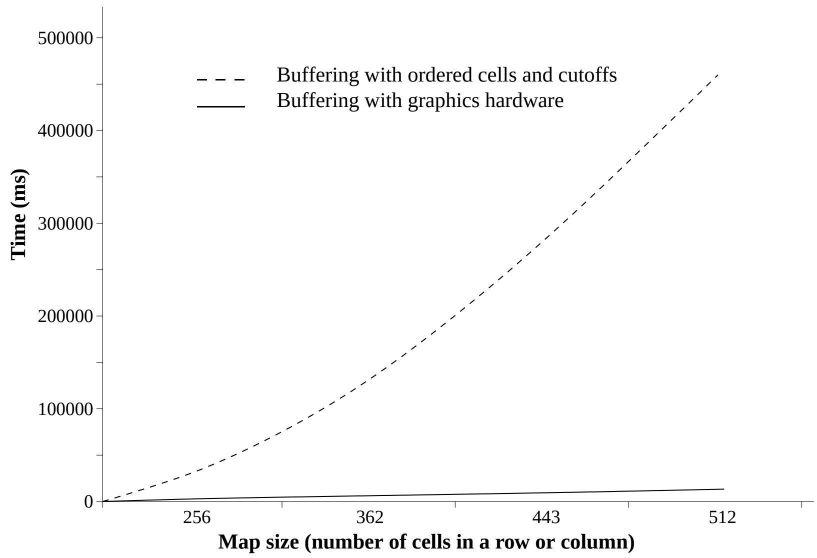

Figure 10. Processing time of buffering with ordered cells and cutoffs plotted against processing time of buffering using graphics hardware.

2002). In these cases, the graphics hardware method performs significantly better than any of the software algorithms. Figure 10 shows a plot of processing time for buffering with ordered cells and cutoffs and for buffering with graphics hardware, applied to a typical fuzzy map. The horizontal axis shows the number of cells in each row or column of the map, rather than the total number of cells of the map, and therefore a linear curve indicates quadratic processing time.

7 Conclusion

The idea of using fuzzy set theory to handle imprecision in spatial reasoning is not new (Altmann, 1994), and compared to other approaches our way of defining buffer operations in GIS might look like a step backwards. However, our more rigid way of looking at buffering of fuzzy maps has two advantages. Firstly, it allows us to apply algorithms that are practically more efficient than brute-force buffering, owing to restricting re-calculations of cell membership grades to candidates for potential value changes. And secondly, it enables us to implement a particular brute-force variant of buffering on widely-available special hardware.

Acknowledgement

Many thanks to Damien Duff, who performed the evaluation of the algorithms and provided some of the illustrations used in this paper.

References

Altmann, D. (1994). Fuzzy set-theoretic approaches for handling imprecision in spatial analysis. *International Journal of Geographical Information Systems, 8*,271–289.

Bloch, I. (2000). Spatial representation of spatial relationship knowledge. In *Proceedings of KR-00* (pp. 247–258). Colorado: Breckenridge.

Brimicombe, A. (1997). A universal translator of linguistic hedges for the handling of uncertainty and fitness-of-use in GIS. In Kemp, Z. (Ed.), *Innovations in GIS 4*. (pp 115–126). London, England: Taylor and Francis.

Coppock, J. and Rhind, D. (1991). The history of GIS. In Maguire, D., Goodchild, M., and Rhind, D., (Eds.), *Geographical information systems: Principles and applications* (pp. 21–43). Essex, England: Longman Scientific & Technical.

Driankov, D., Hellendoorn, H., and Reinfrank, M. (1996). *An Introduction to Fuzzy Control 2nd edition*. Berlin, Germany: Springer.

Duff, D. and Guesgen, H. (2002). An evaluation of buffering algorithms in fuzzy GISs. In *Proceedings of The International Conference on Geographic Information Science (GIScience-02)*. (pp 80–92). Boulder, Colarado: Springer.

Egenhofer, M. and Golledge, R. (1997). *Spatial and temporal reasoning in geographic information systems*. New York: Oxford University Press.

Frank, A. (1994). Qualitative temporal reasoning in GIS-ordered time scales. In *Proceedings of SDH-94* (pp. 410–431). Edinburgh, Scotland.

Frank, A. (1996). Qualitative spatial reasoning: Cardinal directions as an example. *International Journal of Geographical Information Systems, 10*,269–290.

Guesgen, H. and Albrecht, J. (1998). Qualitative spatial reasoning with a fuzzy distance operator. In *Proceedings of SDH-98*. Vancouver, Canada.

Guesgen, H. and Hertzberg, J. (2000a). Buffering in fuzzy geographic information systems. In *Proceedings of International Conference on Geographic Information Science (GIScience-00)* (pp. 189–192). Savannah, Georgia.

Guesgen, H. and Hertzberg, J. (2000b). Buffering in fuzzy geographic information systems. GMD Report 105, GMD. St. Augustin, Germany.

Guesgen, H. and Hertzberg, J. (2001). Algorithms for buffering fuzzy raster maps. In *Proceedings of FLAIRS-01* (pp. 542–546). Key West, Florida.

Guesgen, H., Hertzberg, J., Lobb, R., and Mantler, A. (2001). First steps towards buffering fuzzy maps with graphics hardware. In *Proceedings of FOIS-01 Workshop on Spatial Vagueness, Uncertainty and Granularity (SVUG-01)*. Ogunquit, Maine.

Guesgen, H. and Histed, J. (1996). Towards qualitative spatial reasoning in geographic information systems. In *Proceedings of AAAI-96 Workshop on Spatial and Temporal Reasoning* (pp. 39–46). Portland, Oregon.

Hoff, K., Culver, T., Keyser, J., Lin, M., and Manocha, D. (1999). Fast computation of generalized Voronoi diagrams using graphics hardware. In *Proceedings of SIGGRAPH-99* (pp. 277–286). Los Angeles, California.

Molenaar, M. (1996). The extensional uncertainty of spatial objects. In *Proceedings of SDH-96* (pp. 9B.1–13). Delft, The Netherlands.

Mustafa, N., Koutsofios, E., Krishnan, S., and Venkatasubramanian, S. (2001). Hardware-assisted view-dependent map simplification. In *Proceedings of The 17th ACM Symposium on Computational Geometry*. (pp 50–59). Medford, Massachusetts.

Peuquet, D. (1994). It's about time: A conceptual framework for the representation of temporal dynamics in geographic information systems. *Annals of the Association of American Geographers, 84*,441–461.

Plewe, B. (1997). A representation-oriented taxonomy of gradation. In Hirtle, S. and Frank, A. (Eds.), *Spatial information theory: A theoretical basis for GIS*, COSIT'97, Vol. 1329 Lecture Notes in Computer Science (pp. 121–136). Berlin, Germany: Springer.

Scott, D. (1992). *Multivariate density estimation: Theory, practice, and visualization*. Chichester, England: John Wiley & Sons.

Sinton, E. (1978). The inherent structure of information as a constraint to analysis: Mapped thematic data as a case study. In Dutton, G. (Ed.), *Harvard papers on geographic information systems* Vol. 6. Reading, Massachusetts: Addison-Wesley.

Usery, L. (1996). A conceptual framework and fuzzy set implementation for geographic features. In Burrough, P. and Frank, A. (Eds.), *Geographical objects with indeterminate boundaries*, GISDATA Series No. 2 (pp. 71–85). London, England: Taylor and Francis.

Zadeh, L. (1965). Fuzzy sets. *Information and Control, 8*,338–353.