MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY

A.I. Memo No. 1666 August 12, 1999

Organizing a Global Coordinate System from Local Information on an Amorphous Computer

Radhika Nagpal

Abstract

This paper demonstrates that it is possible to generate a reasonably accurate coordinate system on randomly distributed processors, using only local information and local communication. By coordinate system we imply that each element assigns itself a logical coordinate that maps to its global physical location, starting with no apriori knowledge of position or orientation. The algorithm presented is inspired by biological systems that use chemical gradients to determine the position of cells [12]. Extensive analysis and simulation results are presented. Two key results are: there is a critical minimum average neighborhood size of 15 for good accuracy and there is a fundamental limit on the resolution of any coordinate system determined strictly from local communication. We also demonstrate that using this algorithm, random distributions of processors produce significantly better accuracy than regular processor grids - such as those used by cellular automata. This has implications for discrete models of biology as well as for building smart sensor arrays.

This report describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Nagpal is advised by Professors G.J. Sussman and H. Abelson. The author is supported in part by DARPA, under contract number N00014-96-1-1228 administered by ONR, and in part by the Lucent GRPW Fellowship Program. Address: radhi@ai.mit.edu, Room 432, 545 Technology Square, Cambridge MA 02139.

1 Introduction

Recent developments in micro-fabrication will enable the inexpensive manufacturing of massive numbers of tiny computing elements with integrated sensors and actuators. It will become possible to cheaply assemble systems that incorporate large numbers of elements into the environment, provided that the elements are manufactured in bulk, without individual programming or precise interconnects. Smart sensing agents can be randomly distributed on surfaces or in structures to create intelligent environments, such as bridges with active surfaces that monitor traffic load and structural integrity, improved materials such as beams that resist buckling or airplane wings that actively reduce drag, or sensitive walls that locate shadows and sounds [6, 2].

Amorphous computing is the study of programming paradigms for such environments, where specified global behavior must be achieved from local information and interactions, without depending on precise arrangement or individually programmed parts. An Amorphous Computer [1] consists of massive numbers of randomly distributed, identical, programmable elements that have only local information and communicate with a small neighborhood of physically nearby elements within a fixed radius.

Since the elements are identically programmed, they have no apriori knowledge of global location. The question arises whether it is possible to organize a coordinate system, such that each element assigns itself a logical coordinate that approximately maps to its global physical location, strictly using local communication. For many of the applications mentioned, it is important for elements to know their physical location relative to other elements in order to interpret sensor information. Coordinate systems are also useful for pattern generation, shape detection, naming and routing.

In this paper we present an algorithm for organizing a coordinate system on an amorphous computer. The algorithm is inspired by biological systems that use chemical gradients to determine the position of cells [12]. We show, via analysis and simulation, that it is possible to generate a reasonably accurate coordinate system on randomly distributed processors using only local information and local communication. Two key results are: there is a critical minimum average neighborhood size of 15 for good accuracy and there is a fundamental limit on the resolution of any coordinate system determined strictly from local communication. We also demonstrate that, using this algorithm, random distributions of processors produce significantly better accuracy than regular processor grids, such as those used by cellular automata. This has implications for biological models as well as building smart materials.

The outline of this paper is as follows: Section 2 presents the amorphous computing model. Section 3 presents a biologically inspired algorithm for generating a coordinate system from local information. An extensive analysis of the accuracy of the coordinate system generated by this algorithm is presented in section 4 with experimental results in section 5. Section 6 compares the results to coordinate systems generated on regular processor grids. The remaining paper discusses future work, such as generating coordinate systems from a manifold of local coordinate patches.

2 An Amorphous Computer

The amorphous computing model is a massively parallel computing model [1], like cellular automata, but with some significant differences. In an amorphous computer, myriad identical processors are randomly distributed on a surface or in a volume, in this case on a two dimensional plane. Processors do not have global knowledge of the topology or their physical location. Nor do they have global ids; instead they have random number generators. Each processor communicates with physically nearby processors within a fixed distance r, where r is much smaller than the dimensions of the plane. All processors within the distance r of a processor are called its communication neighborhood and, unlike cellular automata, there is no knowledge of the relative orientation of any of the neighbors.

3 A Biologically Inspired Coordinate System

Developmental biology is an important source of inspiration for amorphous computing paradigms because of its many similarities to the amorphous computing model. Cells with identical programs (DNA) use many different techniques to robustly determine their position relative to other cells. Positional information is key for pattern formation [15]. One commonly observed technique involves the use of *chemical gradients* - a chemical is released from a cell such that the concentration of the chemical decreases as one moves further away from that cell, giving an indication of distance. For example, in the drosophila embryo there are three chemical gradients originating from three points at the anterior, posterior and dorsal side of the embryo respectively [12]. These three chemical gradients effectively create a coordinate system which is used to segment the embryo into head thorax and abdomen regions, as well as dorsal and ventral regions. On the amorphous computer, we use a algorithm inspired by the drosophila embryo.

3.1 Coordinate System Algorithm

The algorithm is based on the fact that the position of a point on a two dimensional plane can be uniquely described by its distance from three non-colinear reference points. The basic algorithm consists of three main steps. First three non-colinear *anchor processors* are chosen, either by an external stimulus or by a leader election algorithm. Second, each anchor produces a *gradient* that allows other processors to determine their distance from the three anchors. Finally a *triangulation* formula is used to convert from distances to cartesian coordinates relative to the three anchor processors. The following subsections describe each step of the algorithm in more detail.

3.1.1 Gradient Algorithm

An anchor processor initiates a gradient by sending its neighbors a message with a count set to one. Each recipient remembers the value of the count and forwards the message to its neighbors with the count incremented by one. Hence a wave of messages propagates outwards from the anchor. Each processor maintains the minimum counter value received and ignores messages containing larger values, which prevents the wave from traveling backwards. If two processors can communicate with each other directly (i.e. without forwarding the message through other processors) then they are considered to be within one communication hop of each other. The minimum count value, h_i, that a processor i maintains will eventually be the length of the shortest path to the anchor in communication hops. Hence a gradient is essentially a breadth-first-search tree [9].

In an amorphous computer, a communication hop has a maximum physical distance of r associated with it. Therefore we see that processors with the same count tend to form concentric circular rings, of width approximately r, around the anchor processor. Figure 1 shows gradients originating from two corner anchors. The circular shape and the uniformity of the width of the ring are better when the average neighborhood size is high, because then the shortest communication path between any two processors is likely to lie close to the straight-line path between them.

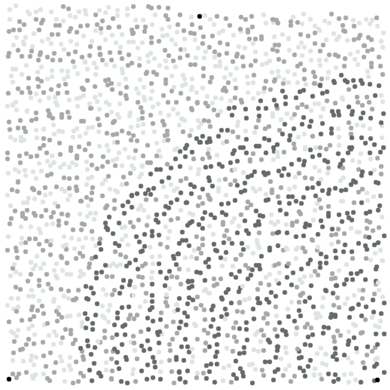


Figure 1: Gradients propagating from two corner anchors. Each dot represents a processor. Processors with even valued distances are colored darker.

3.1.2 Smoothing Algorithm

As described, the distance estimates obtained from the gradients are integral multiples of r. To obtain a better resolution, all processors perform a *smoothing* step, where they average their coarse distance values with their neighbors' values to compute a new distance of much higher resolution.

$$s_i = \frac{\sum_{j \in nbrs(i)} h_j + h_i}{|nbrs(i)| + 1} - 0.5 \qquad (1)$$

Figures 2(a) and (b) show the effect of smoothing. Before smoothing processors 'think' they lie on concentric circles around the anchor. After smoothing the resolution improves significantly.

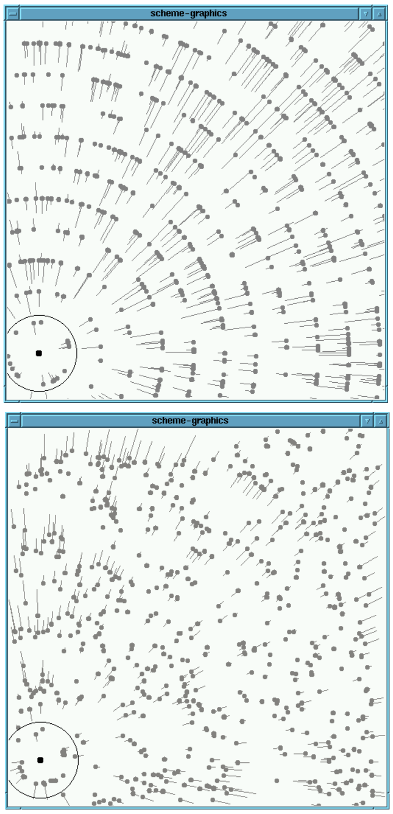
3.1.3 Triangulation Algorithm

After the smoothing phase, processors combine the distances from the three anchors to estimate their position relative to the triangle created by the anchor processors. There are several possible formulas for obtaining cartesian coordinates from the distances. For example one side of the triangle can be made the x-axis or the actual coordinates of the anchors can be used, if they are available. Figure 3 shows the error in the final position estimate.

3.1.4 Algorithm for Choosing Anchors

The three anchors can be chosen by an external stimulus, for example by using a probe to select three processors. Or processors can select the three anchors in a distributed manner. Processors use their random number generators to select a single leader

(a)



(b)

Figure 2: Error in distance estimates (a) before and (b) after smoothing. The line represents the difference between the actual distance of the processor from the anchor and the estimated distance (represented by a dot). Before smoothing, processors in concentric rings around the anchor share the same distance estimate. After smoothing the overall error in distance estimates decreases.

(several distributed algorithms for leader election are presented in [9, 11]). The leader can then use a gradient to find all processors at a given distance and select one to be the second anchor. Together the two anchors together choose a third point at given distances from them, again using gradients. This is similar to constructing a triangle using a compass. Other heuristics can also be used, such a giving a higher weight to processors that are likely to be on the boundary of the plane.

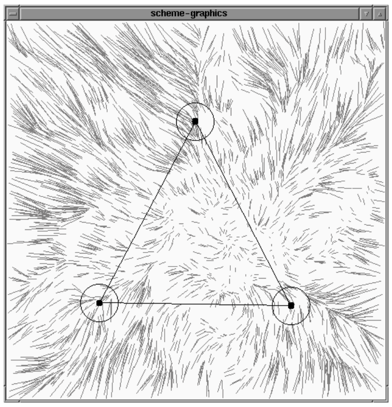


Figure 3: Error in position after triangulation. The line connects the actual position to the logical position and represents the error in the position estimates. The error is less inside the anchor triangle and least in the triangle center.

4 Theoretical Analysis

In this section we analyze the quality of the coordinate system produced by the algorithm. In particular we look at the effect of the random distribution of processors and the average neighborhood size on the accuracy of the position estimates. The next section (section 5) presents simulation results that confirm the analysis presented here. This section also demonstrates that there is a fundamental limit on the resolution of a coordinate system generated by any algorithm that depends only on local communication information.

The quality of the coordinate system is measured by computing the average absolute error (distance) between the actual physical location and the logical position. We found three main sources of error in the position estimates.

1. Errors in the distance estimates produced by gradients due to the discrete distribution of processors.

2. Errors in smoothing due to variations in the density of processors.

3. Region specific errors introduced by the triangulation formula used to combine the distances into a position estimate.

The remaining section presents an analysis of each of these three sources of error. For the purpose of analysis we introduce some notation regarding the processor distribution. Processors are distributed independently and randomly on a two dimensional plane. The probability of a certain number of processors in a given area can be described by a Poisson distribution [1] [14]. The expected neighborhood size, n_{avg}, is equal to $\rho\pi r^2$, where ρ is the density of processors per unit area and r is the communication radius. The processor density ρ is equal to $\frac{N}{S}$ where N is the total number of processors and S is the total surface area.

4.1 Accuracy of Gradients Before Smoothing

4.1.1 Error due to Discrete Distribution

Given any two processors, there may not be enough intermediate nodes for the shortest communication path to lie along the straight-line path between the source and destination. In that case, the shortest communication path will overestimate the actual distance between processors. Intuitively this is more likely if the density of processors is low.

This phenomena has been extensively studied in the context of random plane graphs and packet radio networks, which share a similar model to an amorphous computer. The average distance covered per communication hop, d_{1hop}, can be determined by dividing the physical distance between a pair of processors by the number of hops in the shortest communication path. Kleinrock and Silvester [8] show that the expected distance covered in a single hop, depends only on the average neighborhood size, n_{avg}.

$$d_{1hop} = 1 + e^{-n_{avg}} - \int_{-1}^{1} e^{-\frac{n_{avg}}{\pi}(\arccos t - t\sqrt{1-t^2})} dt \tag{2}$$

In figure 4, d_{1hop} is plotted for different n_{avg}. As n_{avg} increases, d_{1hop} gets closer to one and hence the distance estimate improves as the density of processors increases. Once n_{avg} reaches 15, d_{1hop} begins to level off and increasing the neighborhood size has diminishing returns.

Hence we expect 15 to be a critical average neighborhood size for achieving low errors in the distance estimates. Diminishing improvements in the gradient distance estimates are expected as the density is increased beyond the critical value. In addition, d_{1hop} represents the average width of a ring in the gradient.

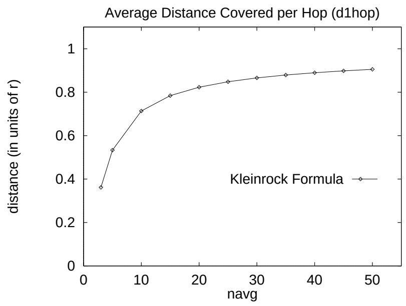


Figure 4: Kleinrock and Silvester's formula for the expected distance covered in one communication hop, d_{1hop}, plotted for different neighborhood sizes, n_{avg}. At $n_{avg} =$ 15 the improvement in d_{1hop} levels off.

The distance of a processor i from the anchor can be estimated as $h_i d_{1hop} r$.

4.1.2 Error due to Coarse Resolution

Even with infinite density, the gradients produce distance estimates that are integral multiples of the communication distance, r. This low resolution adds an error of approximately 0.5 r on average to the distance estimates.

4.2 Accuracy of Gradients After Smoothing

A gradient propagated on a linear array of regularly spaced processors looks like a staircase, where the width of each step is r (figure 5(a)). Averaging self and neighbor values produces a smooth line through the staircase, which when shifted gives the correct distance from the anchor, in units of r. This is the basis for the smoothing formula.

Claim: *For processors evenly spaced along a line, the distance of a processor i from the anchor is given by the smoothing formula (1).*

Proof: Let processor i be distance d_i from the anchor. $d_i = (h_i - 1)r + x_i$, where h_i is the distance estimate from the gradient and x_i is less than r. ρ is the density of processors. After smoothing the distance estimate is:

$$\begin{aligned} s_i &= \frac{\sum_{j \in nbrs(i)} h_j + h_i}{|nbrs(i)| + 1} - 0.5 \\ &= \frac{(h_i - 1)(r - x_i)\rho + h_i r \rho + (h_i + 1) x_i \rho}{2r\rho} - 0.5 \\ &= h_i - 1 + \frac{x_i}{r} = \frac{d_i}{r} \quad \text{Q.E.D.} \end{aligned}$$

[1]
$$Pr(\text{k processors in area a}) = \frac{(\rho a)^k}{k!} e^{-\rho a}$$

However in an amorphous computer the processors are not regularly spaced, and there are variations in density even within a neighborhood. In the absence of any positional information about the neighbors, a processor is forced to weigh all its neighbors equally, which introduces error into the smoothed distance estimate. The expected error in smoothing is related to the variance in the density of processors.

Theorem 1: *For processors distributed unevenly along a line, the error in the distance estimate (smoothed estimate - actual distance) is inversely proportional to the square root of the density of processors.*

Proof: Let the neighborhood size of processor i be $2r\rho$. Suppose the processors are redistributed such that the neighborhood size remains the same, but the processors are placed unevenly within the neighborhood. Then, using formula (1), the smoothed estimate is:

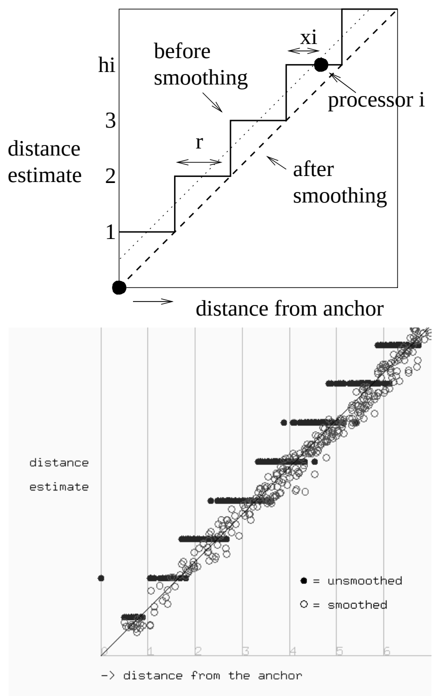
$$s_i = \frac{(h_i - 1)(r - x_i)(\rho - \epsilon_0) + h_i r\rho + (h_i + 1)x_i(\rho + \epsilon_1)}{2r\rho} - 0.5$$

ϵ_0 and ϵ_1 represent the variation from the density when the processors were evenly distributed (ρ). The neighborhood size remains the same, therefore $2r\rho = (r - x_i)(\rho - \epsilon_0) + r\rho + x_i(\rho + \epsilon_1)$ and $(r - x_i)\epsilon_0 = x_i\epsilon_1$. The error in the smoothed estimate is:

$$\begin{aligned} \text{error}_i &= s_i - \frac{d_i}{r} \\ &= \frac{-(h_i - 1)(r - x_i)\epsilon_0 + (h_i + 1)x_i\epsilon_1}{2r\rho} \\ &= (\frac{x_i}{r})\frac{\epsilon_1}{\rho} \end{aligned}$$

The ratio x_i/r depends on the distance of the processor from the anchor and on average is constant. ϵ_1 represents the expected variation in processor density. Therefore the error is proportional to the ratio of the variation in density to the density. As explained before, the processor distribution on an amorphous computer can be described as a Poisson distribution and the standard deviation in density is $\sqrt{\rho}$. Hence the error is proportional to $\sqrt{\rho}/\rho$. Q.E.D.

Intuitively, the higher variation in density is counteracted by the larger number of processor values that get averaged. Figure 5(b) plots the distance estimates for processors before and after smoothing, for an amorphous computer simulation on a 2D plane with $n_{avg} = 20$. As can be seen, smoothing improves the distance estimate as expected, however it is not perfect. Section 5 experimentally determines the improvement due to smoothing.



(a)

(b)

Figure 5: (a) Smoothing on a 1D array of regularly spaced processors. (b) Smoothing on a 2D plane of randomly placed processors, from simulation ($n_{avg} = 20$). Before smoothing the distance estimates are integral multiples of the communication radius. Smoothing improves the resolution significantly, even for uneven distributions

4.3 Accuracy of Triangulation

The error in the distance estimate from a single anchor does not depend on the orientation of the processor about the anchor (radially symmetric). However, when the distances from the three anchors are combined, the error varies depending on the position of the processor relative to the three anchors. Figure 3 indicates that the error in position is largest outside the triangle, particularly behind the vertices, and smallest at the triangle center.

This can be understood analytically by looking at the position of a processor relative to two anchors on a two dimensional plane (figure 6(a)).

Claim: *Let the distance estimates for a processor p be $d_A + \epsilon$ and $d_B + \epsilon$ from anchors A and B respectively, where d_A, d_B are the actual distances from the anchors and ϵ is the error in the distance estimate. Let x be the position of p along the along the A B axis. Then the error Δx in the position estimate, derived from the distance estimates, is proportional to $\epsilon(d_A - d_B)$.*

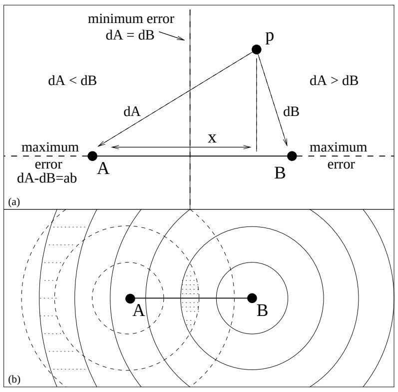


Figure 6: (a) The error in the x position of p relative to two anchors, A and B. (b) Gradient overlap regions are smallest in the center and largest behind the anchors.

Proof: From the formulas for d_A and d_B, one can derive that $x = ({d_A}^2 - {d_B}^2 + ab^2)/2ab$. The error in position is obtained by simply substituting the distances with the distance estimates and is given by $\Delta x = \epsilon \frac{d_A - d_B}{ab}$ where ab is the distance between the anchors. Q.E.D.

The result is that the error in position does not only depend on the error in the distance estimates, but is weighted by the difference in distances from the anchors. Hence, for the same error in distance estimates, processors along the bisector of line AB have very accurate position estimates because the errors cancel each other out. On the other hand, areas behind points A and B have a large error because the difference between d_A and d_B is large[2].

The spatial distribution of error can also be understood qualitatively, without using cartesian coordinates. Along the bisector of the anchor line, the concentric rings of the gradients from both anchors intersect creating small overlapping regions where processors have similar distance estimates. Behind the anchors the concentric rings look almost parallel and intersect to create large overlap regions with many processors that share similar values. Higher resolution distance estimates are needed to distinguish between so many processors and the sensitivity to error is much higher.

When using three anchor points, the high and low sensitivity areas of the three sides of the anchor triangle overlap, resulting in larger errors outside the triangle and behind the vertices and small errors near the centroid of the triangle where the bisectors of the triangle sides meet. Therefore it is advantageous to create the largest possible triangle within the given area.

[2] $\frac{d_A - d_B}{ab} < 1$ by the triangle formula

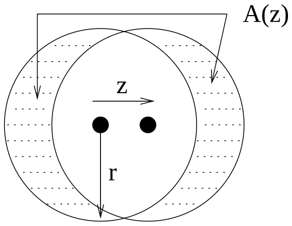


Figure 7: Processor p can move distance z without changing the connectivity if there are no processors in the shaded area.

4.4 Resolution Limit

Here we demonstrate that there is a fundamental limit to the accuracy of any coordinate system developed strictly from the topology of the processor graph. An amorphous computer can be represented by a processor graph where processors are nodes and the nodes are connected by an edge iff the processors can communicate in one hop, i.e. they are less than r distance apart. However, a processor can be physically moved a non-zero distance without changing the set of processors it communicates with, and thus without changing any position estimate that is based strictly on communication between processors. The average distance a processor can move without changing the connectivity of the processor graph gives a lower bound on the expected resolution of any such coordinate system, because the old and new locations of the processor are indistinguishable.

Theorem 2: *The expected distance a processor can move without changing the connectivity of the processor graph on an amorphous computer is* $(\frac{\pi}{4 n_{avg}})r$.

Proof:[3]. Let Z be a continuous random variable representing the maximum distance a processor p can be moved without changing the neighborhood. The probability that Z is less than some real value z is:

$$F(z) = Pr(Z \leq z) = 1 - e^{-\rho A(z)}$$

which is the probability that there is at least one processor in the shaded area $A(z)$ (figure 7). The area $A(z)$ can be approximated as $4rz$ when z is small

[3] proof courtesy of Chris Lass

compared to r and we expect z to be small for reasonable densities of processors. The expected value of Z is:

$$
\begin{aligned}
E(Z) &= \int_0^\infty z\dot{F}(z)dz \\
&= \int_0^\infty \rho 4rze^{-\rho 4rz}dz \\
&\text{by the product rule} \\
&= -ze^{-\rho 4rz}\Big|_0^\infty + (-\frac{1}{\rho 4r})e^{-\rho 4rz}\Big|_0^\infty \\
&= r(\frac{\pi}{4n_{avg}}) \text{ Q.E.D}
\end{aligned}
$$

Hence, we do not expect to achieve resolutions smaller than $\frac{\pi}{4n_{avg}}$ of the local communication radius, r, on an amorphous computer.

4.5 Analysis Conclusions

The analysis in this section suggests that there is a critical neighborhood size of 15 up to which we expect to see significant improvements in the position estimates due to improvements in distance estimates. Beyond 15 most improvements are likely to be due to smoothing alone. In addition we expect that, for a given area, larger anchor triangles will produce more accurate coordinate systems. The resolution limit provides a lower bound on the expected accuracy of the coordinate system.

5 Simulation Results

In this section we present experimental results that support the analysis from the previous section. The experiments were performed on an amorphous computer simulator, that simulates a random layout of processors executing identical programs in parallel and only communicating with other processors within a distance of r. In each experiment, the processors are randomly placed on a unit square, and r is constant. The average neighborhood size, n_{avg}, is varied by varying the total number of processors, N, thus keeping the diameter of the processor graph roughly the same over all experiments. All data points are averaged over ten or more simulation runs.

5.1 Experimental Values for d_{1hop}

Our simulation experiments (figure 8) confirm that the average distance covered per hop, d_{1hop}, is significantly less than the communication radius r and is

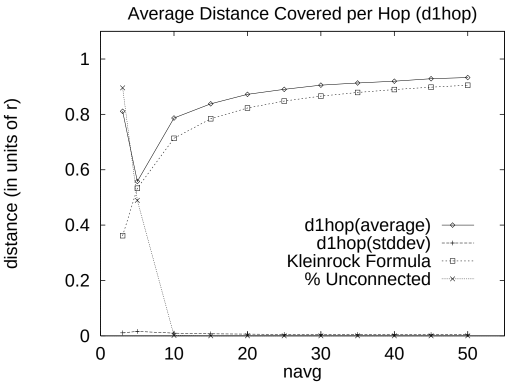


Figure 8: Experimental values for the average distance covered in one communication hop, d_{1hop}, closely follow Kleinrock and Silvester's formula for different neighborhood sizes, n_{avg}.

much closer to the value predicted by Kleinrock and Silvester. The experimental values for d_{1hop} follow the same trend as formula (2) predicts, with diminishing improvements once n_{avg} is greater than 15. The formula slightly under-predicts d_{1hop} due to an approximation made when the source and destination are close.

In addition we plot the percentage of processors not connected to the anchor. Our simulations suggest that an average neighborhood size of above 10 is required to ensure high probability of connectedness. In [13] it is shown that theoretically the average neighborhood size to ensure connectedness is between 2.195 and 10.526 and simulations in [8] suggest that n_{avg} should be at least 5.

5.2 Error in Distance Estimates

These experiments measure the average absolute error in distance estimates from an anchor placed in the lower left corner. The absolute error in the distance estimate (units of r) for processor i is calculated as:

$$\text{error}_i = m_i d_{1hop} - \frac{d_i}{r}$$

where m_i is the distance estimate in communication hops (before or after smoothing) and d_i is the physical distance between processor i and the anchor. We use Kleinrock and Silvester's formula (2) for d_{1hop} (again in units of r) to estimate the distance covered in one communication hop. Figure 9 plots the average absolute error in distance estimates before and after smoothing for different average neighborhood sizes.

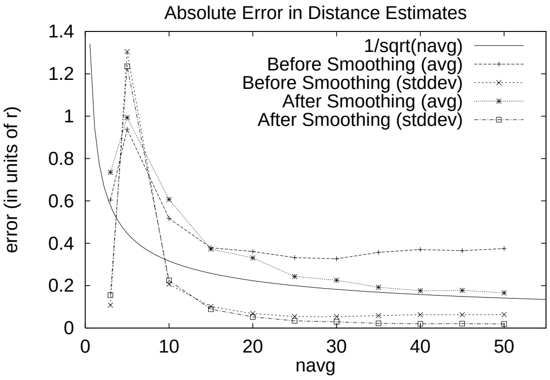


Figure 9: Average absolute error in distance estimates before and after smoothing for different n_{avg}. Significant improvements in distance estimates occur until $n_{avg} = 15$. Beyond that smoothed estimates continue to improve slowly.

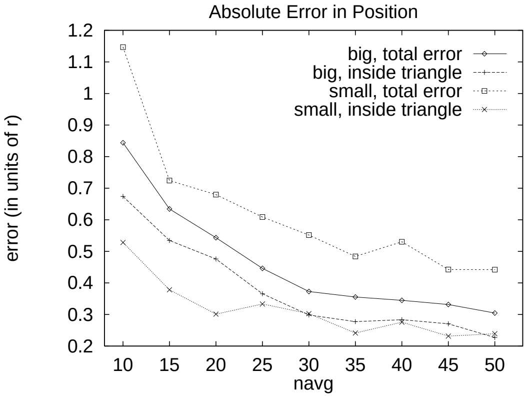


Figure 10: The average error in position after triangulation, for different n_{avg}. The error is plotted for two anchor triangles, *big* and *small*, placed on the same unit square, where *small* is 1/10 the area of *big*. The overall error is less for larger triangles because the average error is smaller inside the triangle.

5.2.1 Error Before Smoothing

As n_{avg} increases, the accuracy of the distance estimate before smoothing improves due to the increased likelihood that the shortest communication path lies along a straight line from the anchor to the processor. As expected, the improvement peaks around a n_{avg} of 15 and the error becomes practically constant at 0.4 r due to the limited resolution before smoothing.

5.2.2 Error After Smoothing

As shown in figure 9, the average error is reduced after smoothing due to increased resolution. We observe that the improvement from smoothing continues to increase as n_{avg} increases beyond 15, as predicted by theorem 1 and the trend is similar to $1/\sqrt{n}_{avg}$ for higher densities. At $n_{avg} = 40$ the average error is 0.2 r. The distance estimate is not perfect even for high n_{avg} because of variations in density.

5.3 Error in Triangulation

Figure 10 plots the average absolute error in position after triangulation for different n_{avg} for two different triangle sizes. The first triangle (*big*) is an isoceles with unit base and height (placed on a unit square). The second smaller triangle (*small*) has a base and height of .3, or approximately one tenth the area. We measure the overall error as well as the difference in errors inside and outside the triangles.

For both anchor triangles we observe that the error in position estimates decreases as n_{avg} increases because of improvements in distance estimates, becoming as low as 0.3 r for the larger triangle. Significant improvements in position estimates occur before the critical neighborhood size of 15, however improvements continue beyond 15 due to the increased accuracy of smoothing.

Our simulation results also confirm that the overall error is larger for smaller triangles and that this is a result of higher errors outside the triangle. However even with a large triangle and high density, the errors are not close to the resolution limit.

5.4 Simulation Conclusions

The simulation results confirm our analysis and show it is possible to obtain a reasonably accurate coordinate system, provided the processors are reasonably dense. The resolution obtained is smaller than the local communication radius. An average neighborhood size of 15 to 20, with a large triangle, produces reasonably accurate coordinates and higher densities produce average errors as low as 0.3 r.

6 Simulation Results on Hexagonal Grids, or A Case for Randomness

We also performed simulation experiments on hexagonal grids to understand the effect of the random distribution of processors on the accuracy of the position estimates. Neighborhood sizes of 16, 36, 48 and 64 were used, such that the radius of communication is a

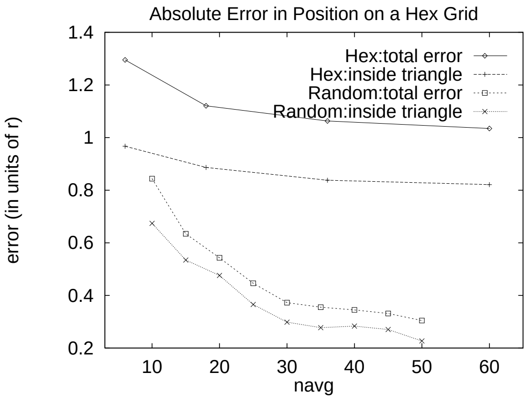


Figure 11: Average absolute error in position estimates on hexagonal grids with different n_{avg}. The errors are significantly higher than for random grids with the same average neighborhood size.

integral multiple of the unit grid distance. An isoceles anchor triangle of unit base and height was used. The results were surprising.

- The average error in position is significantly larger than for a random distribution with the same average neighborhood size (figure 11).
- This error is a direct result of the large direction-based errors in the distance estimates before smoothing (figure 12). Smoothing is unable to compensate for these errors.
- Other discrete regular grids show similar results.

Regular grids tend to prefer certain directions - directions that fall along grid lines. In the non preferred directions there are errors in the distance estimates because of the lack of intermediate processors along the straight-line path. These errors quickly accumulate the further away a processor moves from the anchor in the same direction. As a result some processors have no error whereas others have a very high error that is proportional to their distance from the anchor. This produces a high average error and high standard deviation in error. Figure 12 shows an example simulation on a hexagonal grid with a neighborhood size of 18. Processors on straight grid lines from the anchor experience no error at all whereas processors thirty degrees off experience a small constant error for every communication hop and this accumulates to produce very large errors. The processors 'think' they are on a warped space and this can not be corrected by local smoothing.

Random distributions on the other hand tend not to favor any particular direction and the error experienced by any single processor is small. Using d_{1hop} to estimate the distance covered in one communication hop removes any per hop error that might accumulate. Experiments (not presented here) confirm that the error in distance estimates for most processors is similar, irrespective of the distance from the anchor or the direction relative to the anchor. Thus random distributions are able to avoid the many of the artifacts generated by regular grids.

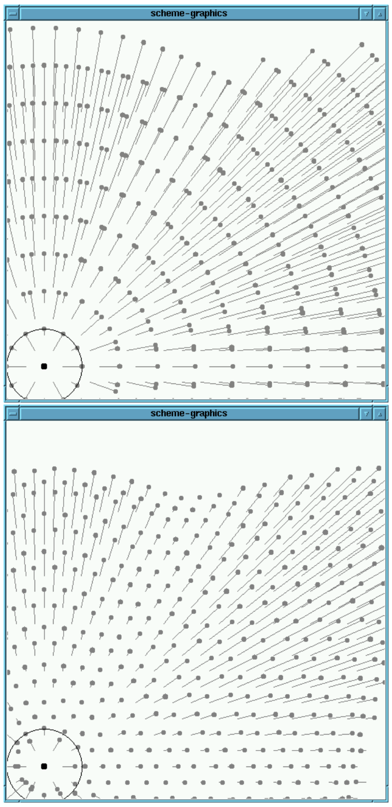


Figure 12: Error in position before and after smoothing on a hexagonal grid with a neighborhood size of 18. The line connects the actual position to the logical position (represented by a dot). As we can see processors 'think' they are on a warped space.

This has important implications for modeling biology. The amorphous computing model may be a more appropriate model for biological systems, since cells are unlikely to be organized in perfect grids. Furthermore, pattern formation is likely to be sensitive to directional biases in the grid. The algorithm presented in this paper is based on a commonly observed technique from developmental biology and has significantly different behavior on regular grids. Different levels of randomness and reliability would more accurately model biological systems and help avoid the pitfalls caused by regularity.

This also has implications for designing applications such as distributed smart sensor arrays. Our simulations suggest that the smart sensors need not be placed on a regular grid to be able to determine their position accurately. There are two main advantages to using randomly distributed sensors. First, the sensors are significantly easier to deploy and one can dispense with wires all together. Second, algorithms designed for random distributions have the advantage of being more robust to the occasional failure of some processors (the resulting graph is still random).

One can design algorithms that take advantage of the exact structure of the regular grid to produce more accurate results - for example, on a rectangular grid, if processors have local orientation (which neighbors are opposite one another) one can determine the position perfectly. However such algorithms quickly become complicated and untenable when random processors and wires may fail. Algorithms, like gradients, are more robust because they do not depend on the existence of perfectly reliable parts and can tolerate a large degree of variation. This paradigm shift makes it possible to conceive of large systems of cheap easily deployable sensors that can robustly self organize global information by communicating local pieces of information, which is at the heart of making smart materials a reality.

7 Related Work

Several other efforts have been made to create and analyze coordinate systems on amorphous computers. In [3], Coore presents a technique for developing a coordinate system on an amorphous computer by solving Laplace's equations in a circle. Preliminary measurements suggest that the algorithm produces similar average error. However, because it is based on iteratively averaging neighbor information, it is very sensitive to variations in density and termination is difficult to detect locally. Katzenelson [7] and Abelson have been investigating algorithms that use multiple anchor processors that know their exact global position.

The algorithm for generating gradients presented in this paper has also been used to create patterns in a local manner. In his thesis [4], Coore presents a pattern formation language based on following local pheromone gradients. The analysis in this paper furthers the understanding of the effects of density and random distribution on his results.

Our group is in the process of implementing example applications of active controlled structures, such as randomly distributed microphones that localize sound, that will make use of these techniques to discover their position.

8 Future Work

Future work will consist of extending these techniques to more complex surfaces. For very large surfaces, creating a coordinate system using a single anchor triangle, may not be feasible because of the time taken and the accumulation of errors over distance. Instead one can construct a large coordinate system by creating a manifold from multiple small overlapping patches of coordinates [5, 10]. A similar approach may be possible on more complex surfaces, such as a sphere, torus or a surface with holes, by segmenting the surface into overlapping plane coordinate patches. Since the processors have no apriori positional information, algorithms for detecting surface characteristics, such as curvature and holes, from local information will aid in segmentation.

9 Conclusions

In this paper we present an algorithm for organizing a coordinate system on an amorphous computer. The algorithm is inspired by biological systems that use chemical gradients to determine the position of cells [12]. We show, via analysis and simulation, that it is possible to generate a reasonably accurate coordinate system on randomly distributed processors using only local information and local communication. Two key results are: there is a critical minimum average neighborhood size of 15 for good accuracy and there is a fundamental limit on the resolution of any coordinate system determined strictly from local communication. We also demonstrate that random distributions of processors produce significantly better accuracy than regular processor grids, such as those used by cellular automata. This has implications for biological models as well as building smart materials.

Finally, we would like to acknowledge the help of Hal Abelson, Daniel Coore, Kevin Lin, Ron Weiss, and Chris Lass in this work.

References

[1] H. Abelson, D. Allen, D. Coore, C. Hanson, G. Homsy, T. Knight, R. Nagpal, E. Rauch, G. Sussman, and R. Weiss. Amorphous com-

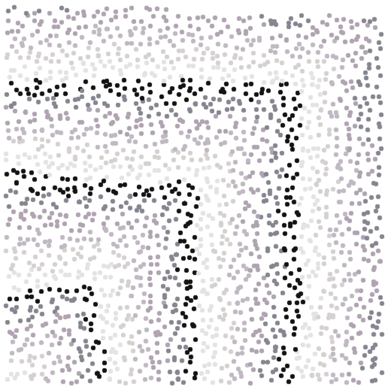


Figure 13: Patterns generated on a triangulation based coordinate system. The picture plots the level curves of $f(x, y) = \max(\lfloor|x|\rfloor, \lfloor|y|\rfloor)$.

puting. *White paper*, 1999. http://www-swiss.ai.mit.edu/~switz/amorphous/.

[2] A. Berlin. *Towards Intelligent Structures: Active Control of Buckling*. PhD thesis, MIT, Department of Electrical Engineering and Computer Science, May 1994.

[3] D. Coore. Establishing a coordinate system on an amorphous computer. In *1998 MIT Student Workshop on High Performance Computing in Science and Engineering, MIT/LCS/TR-737*, 1998.

[4] D. Coore. *Botanical Computing: A Developmental Approach to Generating Interconnect Topologies on an Amorphous Computer*. PhD thesis, MIT, Department of Electrical Engineering and Computer Science, February 1999.

[5] D. Coore, R. Nagpal, and R. Weiss. Paradigms for structure in an amorphous computer. AI Memo 1614, MIT, 1997.

[6] S. Hall, E. Crawley, J. Howe, and B. Ward. A hierarchic control architecture for intelligent structures. *Journal of Guidance, Control and Dynamics*, 14(3):503–512, 1991.

[7] J. Katzenelson. notes on amorphous computing. *(in progress)*, 1999.

[8] L. Kleinrock and J. Silvester. Optimum tranmission radii for packet radio networks or why six is a magic number. *Proc. Natnl. Telecomm. Conf.*, pages 4.3.1–4.3.5, 1978.

[9] N. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers, Wonderland, 1996.

[10] J. Munkres. *Analysis of Manifolds*. Addison-Wesley Publishing Company, Redwood City, CA, 1995.

[11] R. Nagpal and D. Coore. An algorithm for group formation in an amorphous computer. In *Proceedings of the 10th International Conference on Parallel and Distributed Computing Systems (PDCS'98)*, October 1998.

[12] C. Nusslein-Volhard. Gradients that organize embryo development. *Scientific American*, August 1996.

[13] Philips, Shivendra, Panwar, and Tatami. Connectivity properties of a packet radio network model. *IEEE Transactions on Information Theory*, 35(5), September 1998.

[14] R. Scheaffer W. Mendenhall, D. Wackerly. *Mathematical Statistics with Applications*. PWS-Kent Publishing Company, Boston, 1989.

[15] L. Wolpert. Positional information and the spatial pattern of cellular differentiation. *Journal of Theoretical Biology*, 25:1–47, 1969.