Empirical Characterization of Discretization Error in Gradient-based Algorithms

Jonathan Bachrach, Jacob Beal, Joshua Horowitz, Dany Qumsiyeh
jrb@csail.mit.edu, jakebeal@mit.edu, joshuah@mit.edu, danyq@mit.edu
MIT Computer Science and Artificial Intelligence Laboratory
77 Massachusetts Ave, Cambridge, MA, USA

Abstract

Many self-organizing and self-adaptive systems use the biologically inspired "gradient" primitive, in which each device in a network estimates its distance to the closest device designated as a source of the gradient. Distance through the network is often used as a proxy for geometric distance, but the accuracy of this approximation has not previously been quantified well enough to allow predictions of the behavior of gradient-based algorithms. We address this need with an empirical characterization of the discretization error of gradient on random unit disc graphs. This characterization has uncovered two troublesome phenomena: an unsurprising dependence of error on source shape and an unexpected transient that becomes a major source of error at high device densities. Despite these obstacles, we are able to produce a quantitative model of discretization error for planar sources at moderate densities, which we validate by using it to predict error of gradient-based algorithms for finding bisectors and communication channels. Refinement and extension of the gradient discretization error model thus offers the prospect of greatly improving the engineerability of self-organizing systems on spatial networks.

1 Introduction

A common building block for self-organizing and self-adaptive systems is a *gradient*—a biologically inspired primitive in which each device in a network estimates the distance through the network to the closest device designated as a source of the gradient. This measure of distance through the network is often used as a proxy for geometric distance. For example, gradients have been used in algorithms that guide people (e.g. [10]) and robots (e.g. [13], [11]) through space, establish coordinate systems (e.g. [1], [4]), and create self-scaling geometric patterns (e.g. [12], [6]).

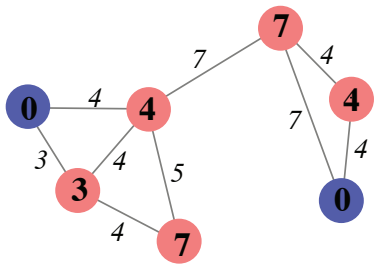


Figure 1. A gradient is a distributed calculation of the shortest distance from each device to a source region (blue). For a spatially embedded network, distance along edges in the network approximates geometric distance in the space containing the network.

Despite its usefulness and popularity, however, the use of gradients to approximate geometric distance has remained a matter of craft rather than engineering—it has not previously been possible to quantitatively predict the behavior of new gradient-based algorithms. If we are able to build a model of gradient that allows such predictions, it will be an important step toward realizing the vision of *self-managing systems engineering* enunciated in [2].

A key component of any model of gradient behavior is the *discretization error*—error due to the difference between geometric distance and distance along network edges. The lower the density of devices, the less relationship there is between geometric distance and the estimates produced by a gradient. The relationship is nonlinear, though, and impacts different algorithms in different ways. While the impact of network structure has been studied for specific gradient-based algorithms (e.g. routing[9] and localization[1]), these models do not generalize.

We have therefore performed a series of experiments characterizing gradient's discretization error on unit disc

978-0-7695-3404-6/08 $25.00 © 2008 IEEE
DOI 10.1109/SASO.2008.53

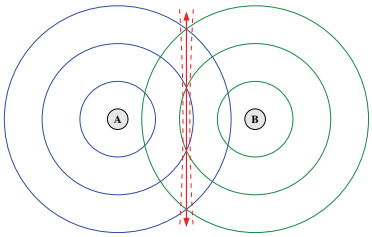


Figure 2. A bisector between regions A and B can be found using two gradients, one with A as its source (blue), the other with B (green). Any point where the distances are equal is part of the bisector (red line), but since there may be no devices precisely on the dividing line, a bisecting region is selected instead (red dashes), containing all the devices where the difference between the two gradients is at most one hop.

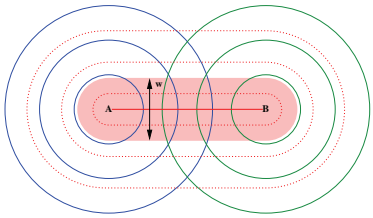


Figure 3. A redundant communication channel w units wide between points A and B can be found using three gradients. Two measure distance, as for bisector, and A broadcasts its value in B's gradient. Devices where the two distances sum to this value are on the shortest path from A to B (red line), which emits its own gradient (red dots). The channel is then the set of devices within $w/2$ units (pink).

networks:

- Section 3 presents a set of characterization experiments. These reveal two troublesome phenomena: a significant dependence of error on source shape and a previously unnoticed transient that becomes a major source of error at high device densities.
- Section 4 explores the causes and structure of the newly discovered transient, suggesting possibilities for how it may be compensated for or eliminated.
- Section 5 extracts a discretization error model for moderate densities. This model can be used to make quantitative predictions about the behavior of gradient-based algorithms, which we demonstrate on algorithms for finding bisectors and communications channels.

Together, these represent a significant advance in our ability to predict the behavior of self-adaptive and self-organizing algorithms that make use of the gradient primitive, and point out possibilities for further advances.

2 The Gradient Primitive

We begin with a review of the gradient primitive. Gradient[1] is a distributed algorithm in which each device estimates its distance from the nearest device designated as a source of the gradient (Figure 1). Gradients are generally calculated through iterative application of a triangle inequality constraint: devices in the source set their distance estimate to zero and others minimize over the set of distances through neighbors. The simplest implementations of gradient just start every device at infinity and relax. More sophisticated versions adjust to changes in the network or source[3, 5] or smooth the final estimates to reduce the impact of measurement noise[1].

For a spatially embedded network, distance along edges in the network approximates geometric distance in the space containing the network. Algorithm designers have frequently taken advantage of this, using gradient in geometric or topological constructions. The two gradient-based algorithms—"bisector" and "channel"—that we use for testing error predictions are typical, if relatively simple, examples.

A bisector between two regions, **A** and **B**, can be found using two gradients, one with **A** as its source, the other with **B** (Figure 2). Any point where the distances are equal is part of the bisector. There may be no devices precisely on the bisector, though, so instead a bisecting region is selected, consisting of all devices difference between the two gradients is at most one hop. Note that this region widens as it moves away from the direct path between **A** and **B**. One example of its use is in Origami Shape Language[12], which uses bisectors to implement certain types of fold.

A redundant communications channel w units wide between points **A** and **B** can be found using three gradients (Figure 3) in a variant of the algorithm from [4]. Two measure distance, as for bisector. Next, **A** broadcasts to every device the distance estimate it calculated for **B**'s gradient. Devices where the two distances sum to this value are on the shortest path from **A** to **B**. The shortest path then emits its own gradient, and the channel is then the set of devices within $w/2$ units of the shortest path.

[1]The sometimes-confusing name comes from its biological inspiration—the chemical gradients that organize structure in a developing embryo—and not the mathematical operator.

These and many other gradient-based algorithms have been found to work well and produce qualitatively predictable results. The vital question, however, is whether we can predict the behavior quantitatively as well.

3 Characterization Experiments

This section presents our characterization experiments, executed in simulation on random unit disc graphs. These establish a baseline of behavior for gradient with a planar source, then vary conditions from that base to establish the impact of communication range and source shape.

Two troublesome phenomena are revealed by these experiments. First, source shape has a significant effect on error—disappointing but unsurprising. More serious, however, is the discovery that a previously unnoticed transient becomes a major source of error at high device densities.

3.1 Assumptions

We use the following network model:

- The network consists of n devices, distributed uniformly randomly in a region of A units area.
- Devices are connected using the unit disc model: each device is connected to all others within a communication radius of r units distance. We will call the expected number of neighbors per device the device density ρ.
- Each device knows the range to each of its neighbors.
- Each device executes its program in rounds. Execution is not synchronized, but the time between rounds on any particular device is in the range $[\tau - \epsilon, \tau + \epsilon]$. Devices broadcast to their neighbors halfway between executions.

Because we wish to study discretization error, we will eliminate other error sources for purposes of this characterization. We assume that devices are never added or removed, that messages are never lost, that there is no error in the range measurement, and that $\epsilon = 0$ (though devices still execute at arbitrary phase to one another). Note that the four bulk network parameters actually have only three degrees of freedom, since $\rho = \frac{n\pi r^2}{A}$ (neglecting edge effects).

3.2 Gradient from a Planar Source

We select a planar source as our base condition for characterization, both for simplicity of analysis and because far enough from any source, curves of equal value should be nearly planar.

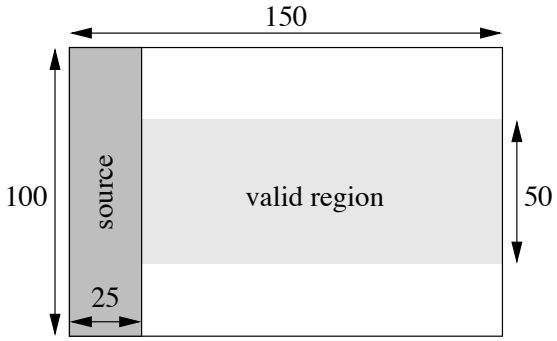


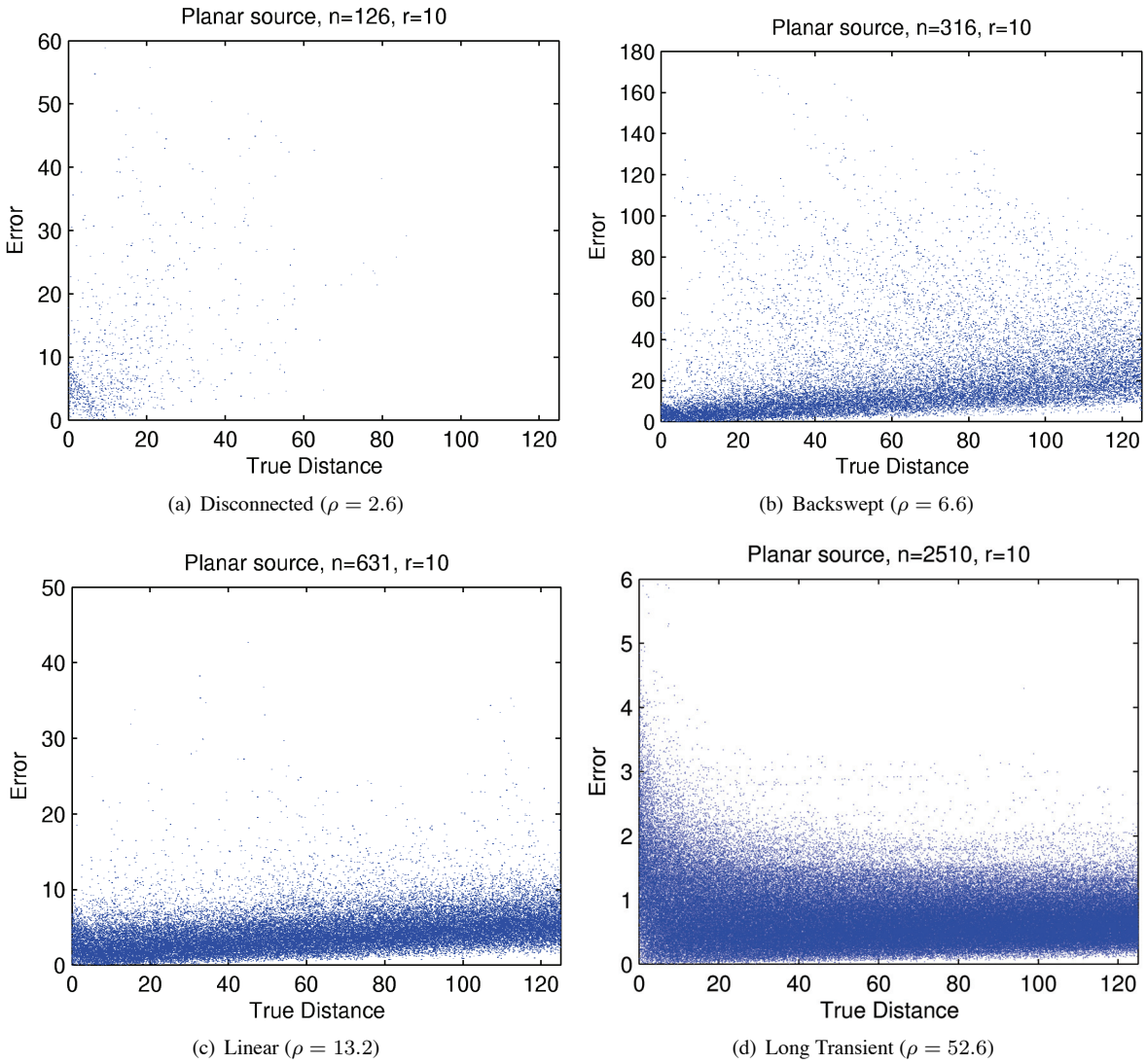
Figure 4. A planar source in a rectangular region. Only the middle portion of the non-source region is used for building the model, to minimize any possible edge effects.

Initial Survey We begin with a survey that was intended to verify general predictions about how gradient error is expected to behave, but actually discovered a previously unreported phenomenon. In a rectangular region 100 units high and 150 units wide ($A = 15000$), the left-most 25 units of the region are designated as the source (Figure 4). We then run experiments with parameters $r = 5$, 10, and 15, and $n = 10^x$, with x ranging from 2.0 to 4.0 in steps of 0.1. For each combination of parameters, we run gradient on 100 randomly generated networks for 100 rounds of computation (long enough for the estimates at all devices to settle), then record the position and gradient value of each device.

It was predicted, based on experience working with gradients and previous studies of distance measurement in unit disc graphs[9, 8], that discretization error would depend primarily on the density ρ. Furthermore, it was predicted that there would be three distinct domains of behavior.

- Below the percolation threshold (approximately $\rho = 4.5$ for two-dimensional unit disc networks[7]) the network is disconnected, most devices will stay at infinity, and a few will receive values with large and highly variable error.
- Well above the percolation threshold, error should climb linearly with distance to the source, as each step adds a random sideways element to the forward progress. This behavior is expected to take hold by the time ρ is around 10.
- In the transitional range between, the network is connected but not approximating space very well, and error is expected to have a linearly climbing core with many wild high-error excursions.

Finally, although the transition from disconnected to connected is a phase change, the transition from a poor spatial



(a) Disconnected ($\rho = 2.6$)

(b) Backswept ($\rho = 6.6$)

(c) Linear ($\rho = 13.2$)

(d) Long Transient ($\rho = 52.6$)

Figure 5. Gradient exhibits four domains of behavior associated with various device densities. Below the percolation threshold ($\rho \leq 4.5$), the network is disconnected (a) and there is little relationship between distance and gradient value. At slightly higher densities, there is a "backswept" domain (b) where error grows linearly for some devices but spikes in regions which "double back" due to large voids in the network. By around $\rho = 10$, there are no large voids and error grows linearly with distance except for occasional local "speckles" of higher error caused by small voids (c). At very high density, an initial transient emerges as a major component of error (d).

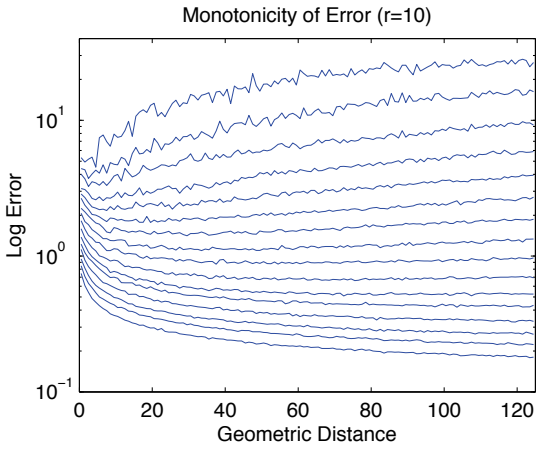


Figure 6. Although the relationship of error to distance changes as ρ rises, the amount of error always decreases. Each line above shows a mean error vs. distance for exponentially increasing values of ρ, from 6.6 at the top to 209.4 at the bottom. Notice that lines never cross.

approximation to a good one was expected to be a smooth progression as ρ rises.

Plotting error (estimated distance - geometric distance) against geometric distance from source shows that these three domains of behavior appeared and transitioned as predicted. Figure 5(a) shows all of the devices with finite nonzero distance estimates in 100 runs of $r = 10$, $\rho = 2.6$, scattering out from the source with effectively no relationship between distance and error. In the transitional range, error grows linearly with distance, but large voids lead to "backswept" formations in the error plot where there are large diversions from the underlying geometry, as in Figure 5(b), which shows 100 runs of $r = 10$, $\rho = 6.6$. By around $\rho = 10$ the network has become a good approximation of the space and there is only a small speckling of higher error points caused by voids with local impact, as in Figure 5(c), which shows 100 runs of $r = 10$, $\rho = 13.2$.

As the density continues to increase, however, an unexpected factor begins to dominate the behavior of the gradient. Near the source there is an initial transient region in which error behaves differently, starting high and declining, as in Figure 5(d), which shows 100 runs of $r = 10$, $\rho = 52.6$. This transient dies out slowly, and becomes the dominant contributor to error over a greater distance at higher densities.

Once the network is approximating space well, however, the mean error is always decreasing as density increases. Figure 6 shows mean error when devices are clustered into bands 1 unit of geometric distance wide. What appears to be happening is that the error has two components, one responsible for the linear rise with distance and another responsible for the transient. Both decrease as ρ increases, but the transient component decreases more slowly. Thus, even though at moderate densities the transient component is so small that it has not previously been noticed, at high densities it comes to dominate the linear component for many hops.

Should we concern ourselves with this transient? Even though it only dominates at densities which are not frequently achieved in current application areas where gradients are used, there is no reason to expect that this must remain the case. Moreover, once one knows to look for it, its effect, though relatively small, is noticeable even at moderate densities (e.g. $\rho = 20$).

Detailed Survey The discovery of the transient means that the initial survey has too few data points to fit a model. This is remedied with a second, more detailed survey, fixing $r = 10$, and letting n range from $10^{2.70}$ to $10^{4.00}$ ($\rho = 10.5$ to $\rho = 209.4$), varying the exponent in steps of 0.01 and rounding to the nearest integer. As before, we run 100 networks for each value of n, recording values after 100 rounds of computation.

To build an error model from this data, we first cluster devices by their geometric distance d from the source. Taking all 100 runs for a given n, all devices with $d \in (0, 1]$ go in a cluster with center $d = 0.5$, all devices with $d \in (1, 2]$ go in a cluster with center $d = 1.5$, and so on. For each cluster, mean and standard deviation of error are computed, yielding a total of 125 data points for each of 131 values of ρ from which to build the model.

Our error model assumes a normal distribution of error, characterized by a mean and standard deviation and parameterized by distance d and density ρ. For the mean, the transient appears to be a power law relation, so we fit each the points for each ρ to the equation

$$\bar{\varepsilon_G} = \alpha d + \beta d^{-\gamma}$$

where $\bar{\varepsilon_G}$ is the mean error, d is the distance, and α, β, and γ are fit constants. The first term models the linear components, the second the transient. We than plot the fit constants against ρ and discover that each also appears to be a power law relation. The empirical equation for the mean error is thus:

$$\bar{\varepsilon_G} = \alpha_1 \rho^{\alpha_2} d + \beta_1 \rho^{\beta_2} d^{(\gamma_1 + \gamma_2 \rho^{\gamma_3})}$$

where the coefficients are:

Name	Value	95% confidence bounds
α_1	7.8	(6.8, 8.7)
α_2	-2.14	(-2.19, -2.10)
β_1	11.2	(10.8, 11.5)
β_2	-0.516	(-0.526, -0.505)
γ_1	-0.292	(-0.303, -0.282)
γ_2	1.6	(1.3, 1.9)
γ_3	-0.77	(-0.86, -0.69)

For the standard deviation σ_{ε_G}, we want to fit it to the equation

$$\sigma_{\varepsilon_G} = \kappa + \lambda d^{-\mu}$$

where κ, λ, and μ are fit constants, because the amount of variation in the linear domain appears to be constant. Unfortunately, the small percentage of high error points still to be found at moderate densities has the effect of introducing a large amount of noise into the standard deviation measure, and it is thus necessary to fit the equation in two stages, first getting a model for λ and μ from high-density trials ($\rho > 20.5$) where the transient dominates, then using that partial model to find κ. The final model for standard deviation is:

$$\sigma_{\varepsilon_G} = \kappa_1 \rho^{\kappa_2} + \lambda_1 \rho^{\lambda_2} d^{(\mu_1 + \mu_2 \rho^{\mu_3})}$$

where the coefficients are:

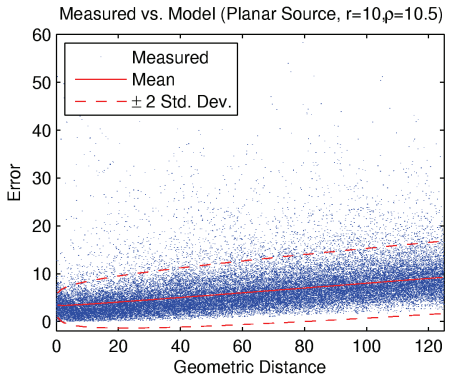
Name	Value	95% confidence bounds
κ_1	-25000	(-52000, 2000)
κ_2	-4.5	(-4.9, -4.0)
λ_1	7.40	(7.07, 7.73)
λ_2	-0.529	(-0.541, -0.517)
μ_1	-0.278	(-0.283, -0.272)
μ_2	11	(5, 16)
μ_3	-1.38	(-1.54, -1.21)

Comparing the model against the data it was derived from shows an overall good fit: $R^2 > 0.99$ for mean and $R^2 > 0.95$ for standard deviation. Figure 8 shows examples of the model compared against raw error.

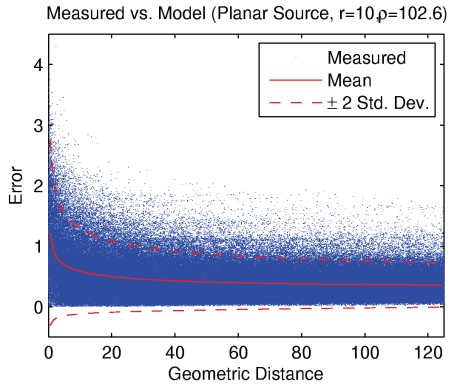
Unfortunately, at moderate density (ρ up to about 20) it is hard to evaluate the fit because there is so much noise in the standard deviation measure. Given the large number of experiments and data points going into each cluster (e.g. an average of 517 for $\rho = 16.6$), we interpret this to mean that a normal distribution is not ultimately an appropriate model for the error distribution at moderate densities. Even so, an alternate model for moderate-ρ (introduced in Section 5) will prove good enough to predict error in the gradient-based algorithms we consider in this paper.

3.3 Variation of Communication Radius

Thus far, we have produced an empirical model of error with respect to two parameters, distance from source d and device density ρ. The total area A and number of devices n cannot affect the algorithm, though behavior might be affected by proximity to an edge. The only remaining network parameter is r, the communication radius.



(a)

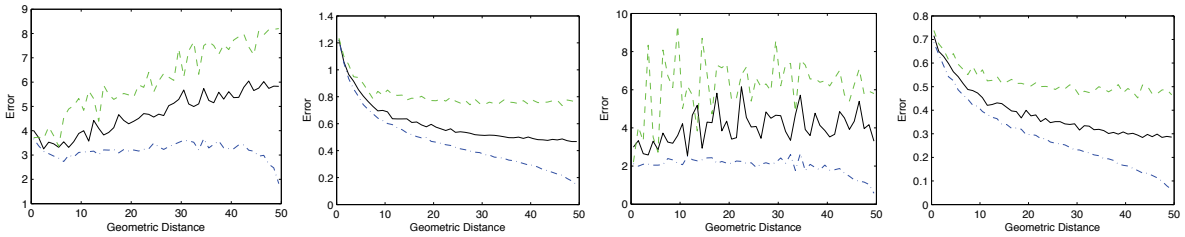


(b)

Figure 8. Examples of comparison of gradient error model (red) against measured error (blue). Note that the model does not capture the asymmetry in variation of error.

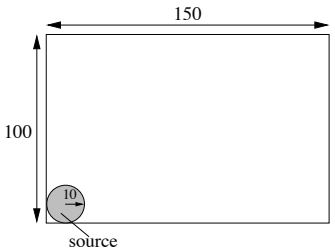
It is certain that there will be no qualitative changes in behavior associated with changes in r, since any two values of r can be mapped onto one another by changing units. A change in units means a linear rescaling of distance, so we expect to see error change proportional to r.

We investigate this with three "crosscutting" surveys, holding ρ constant as r varies from 5 to 20 in steps of 1. Using the same network layout, n is thus determined by r, and data is gathered as before: 100 runs of 100 rounds each. The first survey ($\rho = 10$) examines a point where the linear component dominates, the second ($\rho = 100$) examines a point where the transient dominates, and the third ($\rho = 50$) examines a point with some clear mixing of behavior.

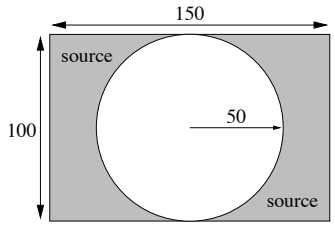


(a) Mean, $\rho = 10.4$ (b) Mean, $\rho = 83.4$ (c) Std.Dev. $\rho = 10.4$ (d) Std.Dev. $\rho = 83.4$

Figure 7. Source shape has a small but significant effect on gradient error. Shown here are typical distance/error curves for planar (black line), convex (green dashes), and concave (blue dash/dot) sources.



(a) Circular Source



(b) Annular Source

Figure 9. Convex and concave sources: circular source in a corner (a), and annular source around outside (b).

In all cases, there is clear variation with radius of communication, and the variation appears to be roughly linear, as predicted. The number of data points is far too small, and noisy, however, to make a serious attempt at fitting, particularly for the $\rho = 10$ survey, where half of the runs involve less than 300 data points. In future work, a better survey would be to hold n constant and vary area instead.

3.4 Variation of Source Shape

To evaluate the whether source shape has a significant effect, we survey the behavior of two alternate source shapes, one convex and one concave. For the convex source, we use a circular source in one corner of the area (Figure 9(a)), and for the concave source we set the source as everything except a circular region in the middle (Figure 9(b)). The surveys are otherwise exactly like the first planar source survey.

The convex source generally produces more error than the planar source, while the concave source produces less (and dramatically less right at the center). This is presumably due to a difference in the number of potential paths with near-optimal lengths from any given point to the source. Worse for predictability, however, a significant amount of the difference between convex and planar source behavior appears to be tied up in the transient. An important question for future investigation is whether there is a simple relationship between the radius of curvature at the source and the error curve.

4 Structure and Causes of the Transient

Where does this transient come from, and how can we compensate for it? If we are to predict the behavior of gradient-based algorithms at high device density, we must have answers to these questions.

We can begin to gain insight on this question by inspecting the structure of error along lines tangent to the surface of the source. Although there is no coherent structure when 100 runs are viewed together, as is to be expected, looking at single runs reveals a startlingly different picture.

Figure 10 shows the error for a single run of gradient at $\rho = 209.4$, plotted against tangent coordinate value and colored to indicate bands of distance from the source. While the relatively high initial error does smooth out farther from the source, it appears that those initial perturbations cast a long "shadow" that sets the general error landscape far into the distance.

It is not surprising that a significant error casts a long shadow, since the error cannot be corrected until a shorter path to the source penetrates the "shadowed region" at a

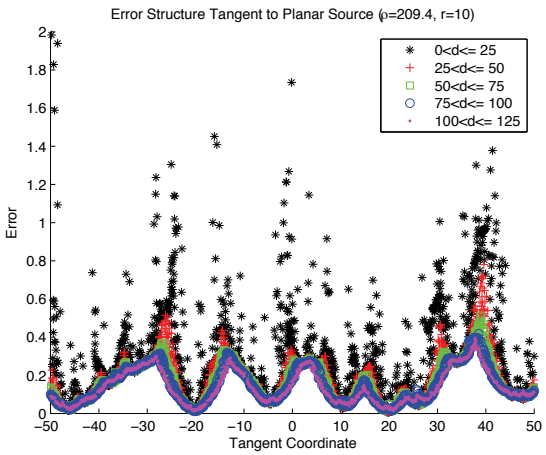


Figure 10. At high neighbor density, error is strongly clustered in space, with the clustering determined by the behavior very close to the source.

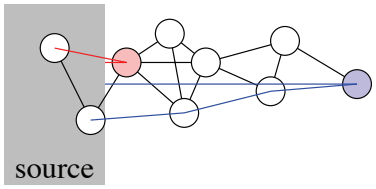


Figure 11. Gradient error behaves differently close to the source: the closest device (red) may have a high relative error because source devices all have zero gradient value and small perpendicular displacements have a large impact on distance close to the source. A device several hops away (blue) has many more possible small-angle paths to a device near the surface of the source.

shallow angle. What is surprising is that there should be so much more error in the very first hop than is added later.

What is different about the first hop? One thing is that there is a non-linearity in the values held by the source: every device in the source has a distance estimate of zero, no matter its depth within the apparent "surface" of the source. Another thing is that small displacements perpendicular to the surface of the source have a larger impact on distance, since their effect on angle is greater. The combination of these two differences would seem to be a good candidate for causing the distribution of error to scale differently in the first hop (Figure 11), so that as density increases the first-hop error becomes increasingly prominent.

If this is, in fact, the case, then it should be possible to eliminate the the transient by removing the difference between the ideal and discrete surface of the source. One way of doing this is to have the source be a single device rather than a planar region; another is to have source devices give their depth within the source rather than zero.

We survey the behavior of gradient under both of these conditions, using surveys that are otherwise exactly like the first planar source survey. As expected, the transient is completely eliminated: examples are shown in Figure 12.

Many gradient-based algorithms use region sources, so using only point sources is not an acceptable solution. It may be possible, however, to locally estimate a node's depth within the source by examining the distribution of its neighbors which are, themselves, sources.

5 Predictions using a Linear-Range Discretization Error Model

Finally, we return to the problem which caused us to build such models in the first place: predicting the error of algorithms that make use of the gradient primitive. We consider only the range where space is well approximated and, since source shape appears to have a particularly significant effect on transient error, where the linear component is dominant (approximately $\rho = 10$ to $\rho = 20$). The boundaries of this range are soft, however, and predictions are still reasonable for some distance outside the range.

For these predictions in the linear range, we will use a model that includes only the linear component. For the mean, this just means dropping the second term of the model from Section 3.2:

$$\bar{\varepsilon_G}' = \alpha_1 \rho^{\alpha_2} d$$

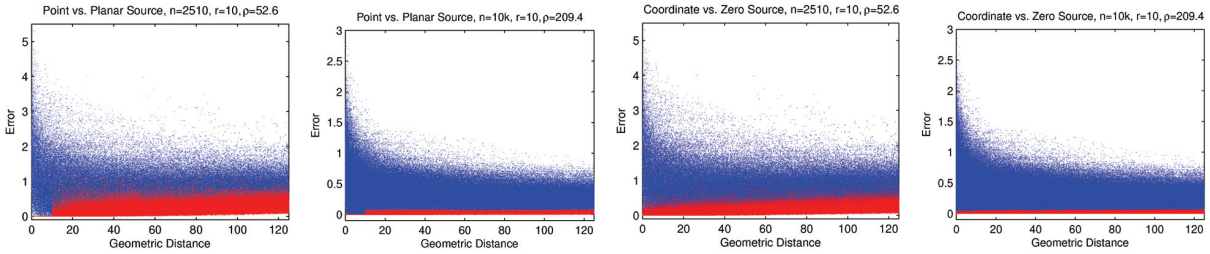
The standard deviation model, however, needs to be rederived. Taking the means of the standard deviation curves $\rho = 10.5$ to $\rho = 16.25$ from the detailed survey, we have 20 data points that appear to form a power law relation. Fitting to the equation

$$\sigma'_{\varepsilon_G} = \kappa'_1 \rho^{\kappa'_2}$$

produces a good fit ($R^2 > 0.95$) where the coefficients are:

Name	Value	95% confidence bounds
κ'_1	350	(180, 510)
κ'_2	-2.0	(-2.2, -1.8)

We now apply this linear-range model of discretization error to predicting the behavior of the two gradient-based algorithms described in Section 2. Because our aim is to show the predictive power of the gradient error model, we will do only a first-order analysis of the behavior of the algorithms.



(a) Point source, $\rho = 52.6$ (b) Point source, $\rho = 209.4$ (c) Coordinate source, $\rho = 52.6$ (d) Coordinate source, $\rho = 209.4$

Figure 12. A gradient from a point source (red, (a) and (b)) or a source where the values indicate depth within the "ideal source" (red, (c) and (d)) does not exhibit the initial error transient shown by a normal planar source (blue) at high densities. Note that for the point source, devices within one hop have zero error since they can measure their distance to the "surface" of the source directly.

Bisector We choose to compute the error of a bisector from the set of devices claiming to be in the bisector region when they should not or vice versa—i.e. those with values different than they would be given if they knew their geometric distance perfectly. The error ε_B will be the sum of the square of the distances of such points from the surface of the bisector region, divided by the number of devices that should be within the region.

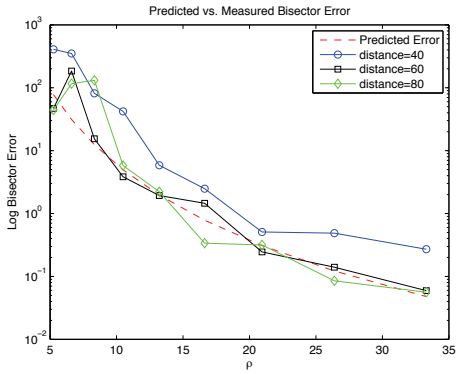
Error comes from two sources: thinning of the bisector region due to distance overestimates and displacement of its center due to differences in the two gradients' distance measures. Because the algorithm looks for two gradients to be equal, their mean errors should cancel and displacement error should come from the combination of the two standard deviations. Thinning, on the other hand, is determined by the rate at which mean error accumulates. These may add or subtract from one another and act on both sides of the bisector region. If we neglect the widening of the region, this yields the approximate error equation:

$$\hat{\varepsilon_B} = 2(\frac{\bar{\varepsilon_G}' r}{d + \bar{\varepsilon_G}'} + \sqrt{2}\sigma'_{\varepsilon_G})^2 / r$$

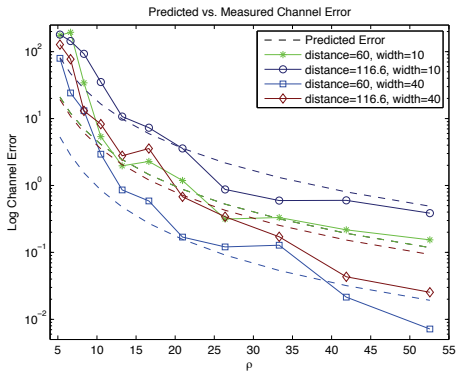
We survey the bisector algorithm using source regions that are circles 20 units in diameter, set 40, 60, or 80 units apart, with $r = 10$ and $n = 10^x$ where x ranges from to 2.4 to 3.2 in steps of 0.1 ($\rho = 5.2$ to 33.3). Figure 13(a) shows that the predicted and actual errors match well.

Channel We compute error for channel the same way as for bisector: the sum of squares of distances of mislabelled devices from the surface of the ideal channel region, divided by the number of devices in the ideal region.

Error comes from the same two sources, thinning and displacement, but displacement now comes from the sideways drift in the shortest path between the two endpoints of the channel. We make a worst-case assumption that the



(a) Bisector Error



(b) Channel Error

Figure 13. A rough analysis using the linear-range model of gradient error produces error predictions close to the actual measured error for both the bisector and channel gradient-based algorithms.

direction of drift is correlated, producing an isosceles triangle whose base is the geometric shortest path length l and whose sides are each 1/2 the expected shortest path plus expected error. The mean displacement is thus:

$$D_C = \frac{1}{4}\sqrt{2\bar{\varepsilon_G}'l + \bar{\varepsilon_G}'^2}$$

and expected thinning (like for bisector) is:

$$T_C = \frac{\bar{\varepsilon_G}'w}{l + \bar{\varepsilon_G}'}$$

The displacement operates only on the "bar" of the channel, while the thinning acts on both round ends as well, yielding the approximate error equation:

$$\hat{\varepsilon_C} = \frac{(D_C + T_C)^2 l + T_C^2 w\pi}{w(l + w\pi)}$$

We survey the channel algorithm using a width w of 19 or 40 and source points that are displaced 60 units from one another vertically and either 0 or 100 units horizontally, for a total length 60 or 116.6. The network is set up with $r = 10$ and $n = 10^x$ where x ranges from to 2.4 to 3.4 in steps of 0.1 ($\rho = 5.2$ to 52.6). Figure 13(b) shows that the predicted and actual errors match well.

6 Contributions

We have shown that discretization error for the gradient primitive is more complex than previously believed, incorporating a transient component that becomes important at high device densities. The transient component of discretization error appears to be caused by gradient's change in behavior at the surface of the source, and we argue that this may allow it to be largely eliminated by locally estimating the depth of devices within the source. We have also modelled the linear component of discretization error, with the aid of intensive empirical characterization, and this model successfully predicts the error exhibited by gradient-based bisector and channel algorithms.

Looking toward the future, it would be vastly preferable to have an analytic model than an empirical one. This is especially true since many of the most tightly-determined constants, like the exponents on the power series, are close to round numbers, but clearly not round numbers (e.g. $\alpha_2 = -2.14 \pm 0.045$). Likewise, there are many areas that need more thorough investigation, such as the effect of source shape.

This investigation has shown, however, that it is reasonable to hope for a relatively simple predictive model of error for gradient within a broad range of plausible device densities. This capability for prediction both grounds existing work on gradient-based algorithms and makes the engineering of new algorithms a more routine process with simpler debugging. Finally, this is an important step forward in demonstrating the "language engineering" approach to self-organizing and self-adaptive systems, where we gain the ability to build complex self-organizing and self-adaptive systems by capturing pre-existing ones as engineering components with well understood compositional behavior.

References

[1] J. Bachrach, R. Nagpal, M. Salib, and H. Shrobe. Experimental results and theoretical analysis of a self-organizing global coordinate system for ad hoc sensor networks. *Telecommunications Systems Journal, Special Issue on Wireless System Networks*, 2003.

[2] J. Beal and J. Bachrach. Infrastructure for engineered emergence in sensor/actuator networks. *IEEE Intelligent Systems*, pages 10–19, March/April 2006.

[3] J. Beal, J. Bachrach, D. Vickery, and M. Tobenkin. Fast self-healing gradients. In *23rd ACM Symposium on Applied Computing*, 2008.

[4] W. Butera. *Programming a Paintable Computer*. PhD thesis, MIT, 2002.

[5] L. Clement and R. Nagpal. Self-assembly and self-repairing topologies. In *Workshop on Adaptability in Multi-Agent Systems, RoboCup Australian Open*, Jan. 2003.

[6] D. Coore. Establishing a coordinate system on an amorphous computer. In *MIT Student Workshop on High Performance Computing*, 1998.

[7] J. Dall and M. Christensen. Random geometric graphs. *Physical Review E*, 66(1):016121, Jul 2002.

[8] J. Katzenelson. Notes on amorphous computing. Draft paper, 1999.

[9] L. Kleinrock and J. Silvester. Optimum transmission radii for packet radio networks or why six is a magic number. In *Natl. Telecomm. Conf.*, pages 4.3.1–4.3.5, 1978.

[10] M. Mamei, F. Zambonelli, and L. Leonardi. Co-fields: an adaptive approach for motion coordination. Technical Report 5-2002, University of Modena and Reggio Emilia, 2002.

[11] J. McLurkin. Stupid robot tricks: A behavior-based distributed algorithm library for programming swarms of robots. Master's thesis, MIT, 2004.

[12] R. Nagpal. *Programmable Self-Assembly: Constructing Global Shape using Biologically-inspired Local Interactions and Origami Mathematics*. PhD thesis, MIT, 2001.

[13] K. Stoy. Controlling self-reconfiguration using cellular automata and gradients. In *8th int. conf. on intelligent autonomous systems (IAS-8)*, 2004.