with many bumps and valleys of just the right size) could have yield a minimal path on a neighborhood size of 10 which was slightly longer than the (true) minimum path on a larger neighborhood. In practice, however, this limitation has not been a problem. Our experience with distance measurements in monkey visual cortex, which is a highly complicated, folded surface, suggest that any errors associated with this algorithm and order of iteration are no more than a few percent.

We use this algorithm as an heuristic approximation to a minimal geodesic-distance algorithm. In our application, running this algorithm to relatively low orders of iteration (i.e., 10) does seem to yield very good approximations to the geodesic distances that we need in order to obtain flattening of the cortical surfaces of the brain.

It is worth pointing out that we have implemented two versions of this algorithm. The first version is the one described above, where at the ith iteration, n-chains up to length 2^{i-1} can possibly be obtained. The other version, at the ith iteration, is limited to n-chains of length i. We have used both runs and merged them for later flattening, thus obtaining a sampling of very long-range distances and also spanning a large neighborhood around each node. Limitations of machine time and storage space make this procedure worthwhile.

IV. Performance

For a polyhedron consisting of about 2500 triangles (about 1200 nodes) representing the surface of monkey visual cortex, we calculated all of the distances over 10-chains on each node. On a Sun-3 workstation, this run took about 2 h and used 12 μbytes of memory; it provided sufficient data for successful flattening of the surface of monkey visual cortex [5].

V. Other Applications

This algorithm finds local shortest-distance patches. Used together with our flattening algorithm, it lets us measure overall shortest distances. This may in fact be the most efficient way to compute long-range shortest distances on the class of polyhedra whose global curvature allows flattening with a relatively small error.[5]

References

[1] J. S. B. Mitchell, D. M. Mount, and C. H. Papadimitriou, "The discrete geodesic problem," *SIAM J. Comput.*, vol. 16, pp. 647-668, Aug. 1987.

[2] D. M. Mount, "Voronoi diagrams on the surface of a polyhedron," Carnegie Mellon Univ., Pittsburgh, PA, Center for Automation Res. Tech. Rep. CAR-TR-121, vol. CS-TR-1496, Dep. Comput. Sci., May 1985.

[3] J. O'Rourke, S. Suri, and H. Booth, "Shortest paths on polyhedral surfaces," in *Proc. Second Symp. Theoretical Aspects of Computer Science*, Aug. 1984.

[4] E. L. Schwartz and B. Merker, "Flattening cortex: An optimal computer algorithm and comparisons with physical flattening of the opercular surface of striate cortex," *Soc. Neurosci. Abstracts*, vol. 15, 1985.

[5] E. L. Schwartz, A. Shaw, and E. Wolfson, "A numerical solution to the generalized mapmaker's problem: Flattening nonconvex polyhedral surfaces," *IEEE Trans. Pattern Anal. Machine Intell.*, this issue, pp. 1005-1008.

[6] M. Sharir and A. Schorr, "On shortest paths in polyhedral surfaces," *SIAM J. Comput.*, vol. 15, no. 1, pp. 193-215, 1986.

[7] R. B. Tootell, M. Silverman, E. Switkes, and R. deValois, "Deoxyglucose analysis of retinotopic organization in primate striate cortex," *Science*, vol. 218, pp. 902-904, 1982.

[8] W. K. Kaplow and E. L. Schwartz, "Measuring mean and Gaussian curvature on triangulated brain surfaces: The differential geometry of macaque striate cortex," Computat. Neurosci. Lab., NYU Med. Center, Tech. Rep. CNS-TR-1-86, 1986.

[9] F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*. New York: Springer-Verlag, 1985.

[5]In other work we describe ways to measure the mean and Gaussian curvature of polyhedra [8].

A Numerical Solution to the Generalized Mapmaker's Problem: Flattening Nonconvex Polyhedral Surfaces

ERIC L. SCHWARTZ, ALAN SHAW, AND ESTAROSE WOLFSON

***Abstract*—We describe methods to "unfold" and flatten the curved, convoluted surfaces of the brain in order to study the functional architectures and neural maps embedded in them. In order to do this, it is necessary to solve the general mapmaker's problem for representing curved surfaces by planar models. This algorithm has applications in areas other than computer-aided neuroanatomy, such as robotics motion planning and geophysics.**

Our algorithm maximizes the goodness of fit of distances in these surfaces to distances in a planar configuration of points. We illustrate this algorithm with a flattening of monkey visual cortex, which is an extremely complex folded surface. We find distance errors in the range of several percent, with isolated regions of larger error, for the class of cortical surfaces which we have so far studied.

***Index Terms*—Cortex, flattened surface, geodesic distance, map.**

Introduction

The mapmaker's problem is to find a flat representation of a curved surface, for example, the surface of the earth. Classical mapmaking has been restricted to the relatively simple spherical surface of the earth. In the case that the surface of interest is complex, and possibly nonconvex, there are no known methods of finding quasi-isometric planar representations, that is, those that distort distance relationships as little as possible.[1] (An *isometric* map would be one in which the distance between any two points was identical to the corresponding distance in the original surface.)

The solution to this problem is of importance to computer-aided neuroanatomy, since it is often desired to view the surface of various cortical areas in a planar model. Primate cortex is highly convoluted, and provides one of the more complex surfaces encountered in practical applications.

Motion planning in robotics is another area of application for which the finding of shortest distances on polyhedral surfaces is of importance [10], [11]. Other areas of biophysics and geophysics would seem to provide possible areas of application of a generalized mapmaker's algorithm.

Our interest is in obtaining a flat representation of the cortical surfaces of the brain, because the detailed maps of sensory and other neural data embedded in these surfaces are easiest to study, measure, and model when they are presented in planar form.

This correspondence describes the method we use to find an optimal quasi-isometry that maps an arbitrary curved surface into a plane. This mapping is optimal in the sense that it is derived from a variational principle that optimizes the overall fit between the curved and planar surfaces. The mapping is a quasi-isometry because it optimizes the fit of distances over multiple scales, rather than, for example, local angles (in which case it would be quasi-conformal).

Manuscript received February 23, 1987; revised December 9, 1988. Recommended for acceptance by J. O'Rourke. This work was supported by the Air Force Office of Scientific Research under Contract 85-0341, System Development Foundation, and the Nathan S. Kline Psychiatric Research Center.

E. L. Schwartz is with the Computational Neuroscience Laboratories, Department of Psychiatry, New York University School of Medicine, 550 First Avenue, New York, NY 10016, and Robotics Research, Department of Computer Science, Courant Institute of Mathematical Sciences, New York University, New York, NY 10003.

A. Shaw and E. Wolfson are with the Computational Neuroscience Laboratories, Department of Psychiatry, New York University School of Medicine, 550 First Avenue, New York, NY 10016.

IEEE Log Number 8928499.

[1]An early version of this work was described by [4]. An alternative approach has been described by [1].

We use a Newton–Raphson minimization, applying it to a matrix of distances between nodes in a polyhedral representation of the surface up to some neighborhood of distances from each node. Minimal geodesic distances are calculated in the surface, and these form the initial distance matrix. The planar distance matrix is initially generated from random numbers. Planar points are moved along the direction of the gradient of a "stress," which measures the quality of the isometry.

We have found that different random starting configurations converge to virtually identical final states. This finding indicates lack of "trapping" in local minima, and also indicates the robustness of our method. We also show that in general, local distances (i.e., edge lengths between adjacent nodes in the polyhedral graph; we call these nearest-neighbor distances) are insufficient for this problem. We base this conclusion on general arguments and on our own experience with local distance problems.

Description of the Algorithm

Given an arbitrary curved surface, we wish to "flatten" it. Because surfaces generally have nonzero Gaussian curvature, there is usually no isometry between the surface and the corresponding plane [13]. In other work [2], we describe ways to measure the mean and Gaussian curvature of a polyhedral surface.

We state the mapmaker's problem in the following way. Given a polyhedral surface, determine the entire set of interpoint distances.[2] This lower triangular distance matrix is of size $N(N-1)/2$ for N nodes. We then determine a set of N points in the plane such that the corresponding distance matrix in the plane provides a best fit, in the least-squares sense, to the distance matrix of the original surface. This is done by a Newton–Raphson gradient descent on the difference between the three-dimensional (geodesic) distance matrix and that of a corresponding set of initially random two-dimensional points [3], [7]. This procedure should yield an optimal quasi-isometric mapping of the surface into the plane.

Distances in Polyhedral Surfaces

One difficulty with this algorithm is that until recently, no algorithms were known that could find distances on arbitrary (i.e., nonconvex) polyhedral surfaces. Sharir and Schorr [11] describe an algorithm that finds minimal distances in a convex polyhedral space. Other workers [5], [6] have described algorithms to find distances on nonconvex polyhedrons. While these algorithms have attractive complexity (that of Mitchell *et al.* [5] is $O(N^2 \log N)$ where N is the number of nodes), they appear very difficult to implement, and we know of no actual implementations of them. Elsewhere we describe an algorithm we have developed that finds minimal geodesic distances in arbitrary polyhedral surfaces [12]. Our algorithm has exponential complexity, but we have implemented it, and it is capable of providing adequate distance information for surfaces as complex as monkey cortex. In the present correspondence, we assume that we have obtained the distance matrix of the original surface by applying that algorithm to compute the discrete minimal distances.

Variational Algorithm

Our mapping problem is very similar to the conventional "multidimensional" scaling problem, which is used to optimally map point sets from N to M dimensions, in which M is usually 1, 2, or 3, and N is usually large (e.g., 10–50). This is a common cluster analysis procedure that is often used to visualize multivariate data sets [3], [7], [8].

In our case, we are mapping between a two-dimensional surface in three-space and one in the plane. We would expect good performance, because our surfaces are not closed, that is, they do not usually represent "crumpled" spheres but rather crumpled *sections* of spheres; and because, although they are highly convoluted or folded, their integral mean curvature, as indicated by separate measurements, is not very large. Typically radii of mean curvature of visual cortex were in the range of 20 mm, while the typical size of cortex was also in the range of 20 × 20 mm. Thus, our surfaces span a fairly small fraction of the solid angle of a sphere, and can, in principle as well as in fact, be flattened with little error [2].

[2]For large problems (our N is typically more than 1000), it is not practical to determine the entire distance matrix. Later in this paper, we describe methods for using subsets of the distance matrix which correspond to overlapping patches of the surface to be flattened.

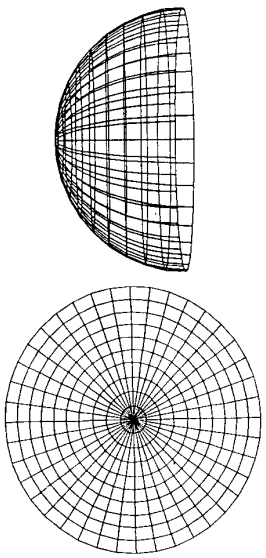

Fig. 1. A hemispherical shell flattened by our algorithm. This figure shows the 3-D wire-frame model of the original surface and also the flattened version of the surface.

Description of the Algorithm

Let the minimal geodesic distances in the surface be organized as a lower triangular matrix d_{ij}. Let the (unknown) distances in the plane be organized as a lower triangular matrix $\bar{d}_{ij}$. We pose the mapmaker's problem as obtaining the best least-squares goodness of fit between these two distance matrices. That is, we propose to minimize the "stress," defined by the quantity L below [7]:

$$L = \frac{1}{c} \sum_{i<j}^{i=N} \frac{(d_{ij} - \bar{d}_{ij})^2}{\bar{d}_{ij}} \tag{1.0}$$

where

$$c = \sum_{i<j}^{i=N} \bar{d}_{ij}. \tag{1.1}$$

Because the distance matrix contains all possible interpoint distances in this discrete problem, an optimal preservation of the distance matrix in a planar representation is equivalent to an optimal preservation of the metric structure of the original surface.

We calculate the initial planar distance matrix from a random set of points. Then we analytically calculate the gradient of (1.0) above, and move each point in the planar layout in the direction of the negative gradient by an amount scaled according to the components of the Hessian of (1.0) [Newton–Raphson method]. Our approach minimizes the stress. We have tested the algorithm with several different random starting configurations, producing solutions that differ by random rotations and reflections but with identical interpoint distance relations, as shown in Fig. 2, which supports the validity of the solution. The quantity L of (1) is a function only of interpoint distances, and is normalized to preserve size. Thus, the solution is determined only up to a rotation and reflection. Plots of stress as a function of the number of iterations (Fig. 2, lower right) give some indication of how well this algorithm performs. For the example illustrated in Fig. 3, there were large oscillations in the stress early in the run; but after about 100 iterations, all three runs converged to the same stress and to the same geometric solution.

Experiments with Nearest-Neighbor and Second-Neighbor Distances

Our early experiments involved only nearest-neighbor distances and second-neighbor distances (we obtained these easily by apply-

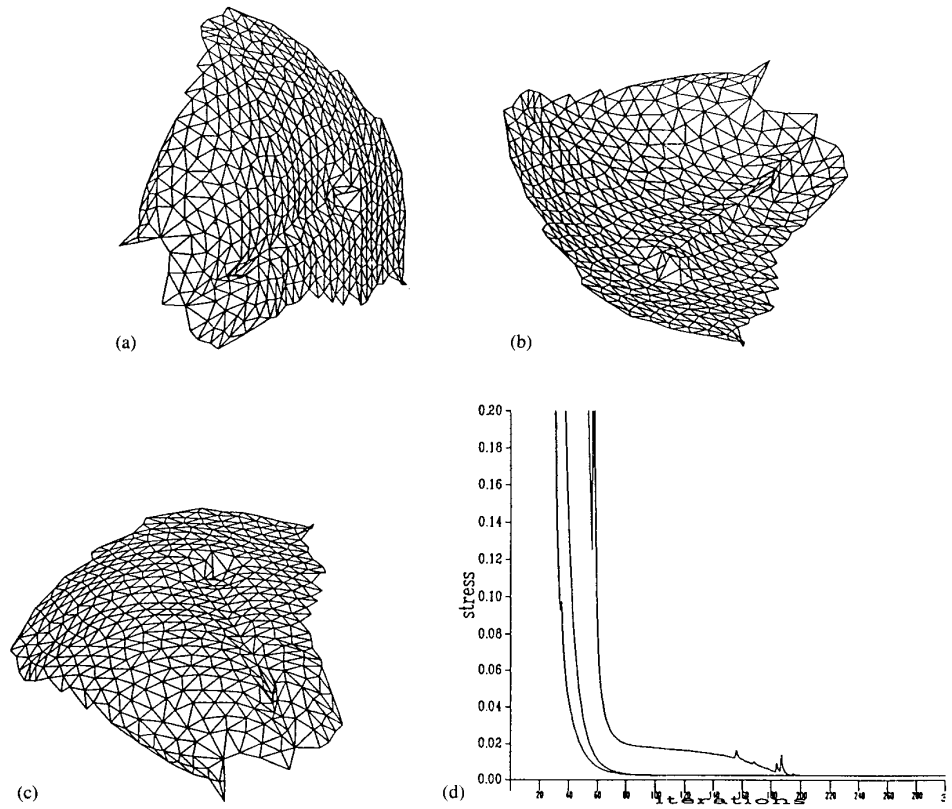

Fig. 2. A section of the operculum of visual cortex, reconstructed from serial sections of monkey brain, flattened by our algorithm. The curved appearance of these figures is illusory; they are of course planar representations. The three different versions of the planar surface were obtained in successive runs, with different random starting configurations. The plot of the stress as a function of iterations is shown at the lower right. For this model, minimal distances were calculated using the algorithm of Wolfson and Schwartz [12]. Use of an eight-node neighborhood (that is to say, a neighborhood extending eight nodes away from a given node) of the 600-node model limited the size of the distance matrices.

ing the law of cosines across two neighboring triangles). (Nearest-neighbor distances are the edge lengths of the original polyhedron, and second neighbors are the neighbors of nearest neighbors; in general, two nodes in a graph are called nth neighbors if there is a path from one to the other consisting of n edges.) The results of these experiments were disappointing. When the algorithm was initialized with a random starting configuration, there generally was no convergence to the expected final state. Instead, the configurations we obtained seemed to be "folded." In other words, large regions had the right structure but were in the wrong position in the final planar configuration.

This result was easy to understand. When only short-range distances are available, little penalty is imposed on misplaced patches whose internal structural details are correct. This point is best illustrated with a simple example. Consider a correctly flattened surface, in which only nearest-neighbor distances were used in the flattening. Now fold this solution along any line. This "folded" solution does not differ appreciably in its fit to the original 3-D surface. Since the distance matrix is composed only of nearest-neighbor distances, only pairs of nodes whose connecting edge lies across the fold line make an erroneous contribution to the goodness of fit of the planar configuration. The relative "weight" of incorrect distances is thus proportional to the length of the fold, while the number of correct distances in the folded solution is proportional to the area of the entire solution. Thus, there is little penalty attached to "folded" solutions for flattenings based on short-range distances.

The only condition under which we were able to obtain successful flattenings with short-range distances was by supplying the algorithm with an initial state that had been relatively well flattened beforehand (i.e., through human intervention). Obviously, such a procedure cannot be called a flattening algorithm. However, by adding longer-range distance terms, computed using an implementation of the algorithm of [12], we successfully flattened polyhedra from random starting configurations, as described below.

Heuristic Simplification of the Algorithm

The size of the distance matrix (and hence the complexity of this algorithm) scales as N^2, where N is the number of nodes. The computer memory and CPU time required for large N are therefore prohibitive. We restricted the size of the neighborhood of distances represented for each node to a "patch" of surface.

The extent of these patches is an empirical problem. In our experience, a patch whose size is roughly 10 percent of the surface area of the entire problem yields good results, while smaller patch sizes lead to solutions which are "folded," which was clearly evident in graphic displays of the flattened solution. Thus, for example, as shown in Fig. 3(d), with some 1200 nodes, a patch of "diameter" of about 10 nodes gave good results.

Through the use of patches, our distance matrix became relatively sparse. By storing this data in a forest of binary search trees,[3] rather than a matrix, we achieved reasonable performance on a Sun-2 microprocessor system (which is roughly comparable to a VAX 750). CPU time required to run these algorithms for the ex-

[3]Each binary search tree corresponds to a node on the polyhedron. We call the collection of these binary search trees a "forest."

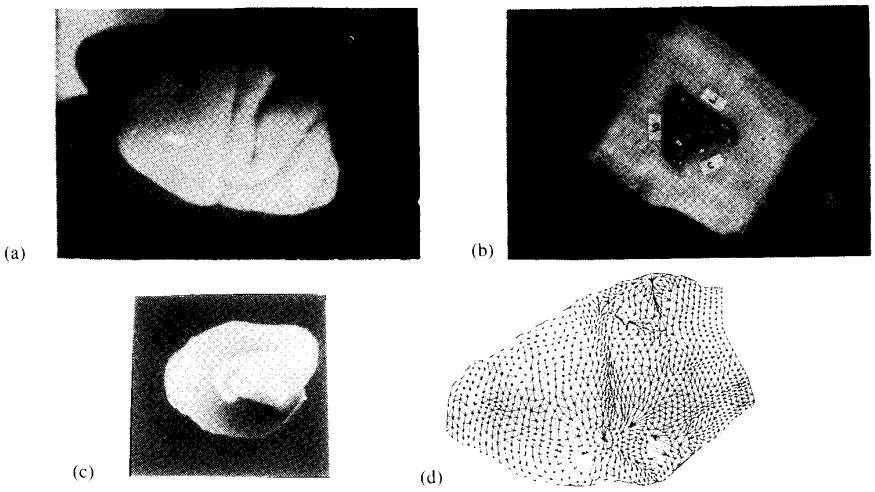

Fig. 3. (a) A block of monkey brain. The outer surface (operculum) of visual cortex is the bullet-shaped region at the left. (b) The opercular visual cortex physically flattened by being pressed between glass plates while being filmed with an 8 mm movie camera. The India-ink dots visible on the cortex were used to measure the distortion caused by physical flattening. (c) A computer-generated reconstruction of the entire striate cortex, viewed from the bottom. The opercular surface is away from the viewer, while the convoluted "calcarine" cortex appears as a flower-petal shape closer to the viewer. (d) The entire striate cortex from (c), as flattened by our algorithm. Greater densities of triangles in the model indicate the regions of higher curvature on the cortical surface.

amples of monkey cortex consisting of 1200 nodes, using a patch diameter of ten nodes, was about 10 hours for the geodesic distance calculation and about 10 hours for the flattening on a Sun-2 workstation.

Examples of Surface Flattening

Example 1: Hemisphere.

We generated a hemispherical shell, calculated the distance matrix (of nearest neighbors only) analytically, and "flattened" it with our algorithm. The result is shown in Fig. 1.

Example 2: Visual cortex.

The surface of the visual cortex of a macaque monkey was digitized, triangulated, and flattened. Fig. 2 shows the "opercular" surface of visual cortex. The operculum is the easily accessible outer surface of primary visual cortex. In fact, as shown in Fig. 3(b), the opercular cortex can be flattened physically with relatively little distortion. We studied the error component of physical flattening of the opercular surface of cortex by making India-ink dots on cortex which we then flattened and filmed with a movie camera. This allowed us to observe an animation of the distortions caused by physical flattening. Fig. 3(b) is a picture from this series [9].

The usefulness of digital flattening is clearer when we consider the entire striate cortex instead of just the operculum. Fig. 3(c) is a computer-generated reconstruction of the entire cortex, viewed from below. Although the cortex is quite convoluted, this view shows that its surfaces are generally rather flat. In fact, it is easy to fold a sheet of paper into the petal-like shape shown in Fig. 3(c) without tearing or stretching the paper.

Fig. 3(d) shows the flattening of the entire cortex. The model in this example was a polyhedron consisting of about 2500 triangles. We flattened the model by using a 10-neighbor deep distance around each node. Mean error (defining error as the average local difference between edge lengths in three dimensions and the corresponding lengths in two dimensions) was 5 percent.

Other Applications

As noted earlier, this algorithm is a solution to the general mapmaker's problem. The visual cortex is arguably the most convoluted of the cortical regions, and our cortical data seems to be rather more complex than the data in other common instances of the problem. The good results we have obtained with this data encourage us to think that our methods can deal competently with a variety of problems related to the mapping of surfaces. In particular, we believe that problems in robot motion planning and autonomous vehicle navigation might yield to an application of this algorithm.

References

[1] G. J. Carman and D. Van Essen, "Computational algorithms for the production of two-dimensional maps of cerebral cortex," *Neurosci. Abstracts*, vol. 11, p. 1243, 1985.

[2] W. K. Kaplow and E. L. Schwartz, "Measuring mean and Gaussian curvature on triangulated brain surfaces: The differential geometry of macaque striate cortex," Computat. Neurosci. Lab., NYU Med. Center, Tech. Rep. CNS-TR-1-86, 1986.

[3] J. B. Kruskal, "Nonmetric multidimensional scaling: A numerical method," *Psychometrika*, vol. 29, pp. 1-27, 115-129, 1964.

[4] B. Merker and E. L. Schwartz, "Computer aided neuro-anatomy: Reconstruction and characterization of the opercular surface of Macaque striate cortex," *Investigative Ophthal. and Vis. Res. (Supplement)*, vol. 26, no. 3, p. 164, 1985.

[5] J. S. B. Mitchell, D. M. Mount, and C. H. Papadimitriou, "The discrete geodesic problem," *SIAM J. Comput.*, vol. 16, pp. 647-668, Aug. 1987.

[6] J. O'Rourke, S. Suri, and H. Booth, "Shortest paths on polyhedral surfaces," in *Proc. Second Symp. Theoretical Aspects of Computer Science*, Aug. 1984.

[7] J. W. Sammon, "A nonlinear mapping for data-structure analysis," *IEEE Trans. Comput.*, vol. C-18, pp. 401-409, 1969.

[8] S. S. Schiffman, M. L. Reynolds, and F. W. Young, *Introduction to Multidimensional Scaling*. New York: Academic, 1981.

[9] E. L. Schwartz and B. Merker, "Flattening cortex: An optimal computer algorithm and comparisons with physical flattening of the opercular surface of striate cortex," *Soc. Neurosci. Abstracts*, vol. 15, 1985.

[10] M. Sharir, "On shortest paths amidst convex polyhedra," *SIAM J. Comput.*, vol. 16, pp. 561-572, 1987.

[11] M. Sharir and A. Schorr, "On shortest paths in polyhedral surfaces," *SIAM J. Comput.*, vol. 15, no. 1, pp. 193-215, 1986.

[12] E. Wolfson and E. L. Schwartz, "Computing minimal distances on arbitrary two-dimensional polyhedral surfaces," *IEEE Pattern Anal. Machine Intell.*, see this issue, pp. 1001-1005.

[13] M. P. do Carmo, *Differential Geometry of Curves and Surfaces*. Englewood Cliffs, NJ: Prentice-Hall, 1975.