Cartographic Manipulation Panel

Ronald Bolton, Presiding
National Ocean Survey

Nicholas Chrisman
Harvard University

Michael McCullagh
University of Nottingham, U.K.

David W. Rhind
University of Durham

F. Robert Niedermair
National Ocean Survey

P. Frederick Stepler
IBM Corporation

Bolton: Since I am a specialist in data manipulation, you would expect me to say that cartographic manipulation proceses are so far advanced that we need more and better hardware. But right now we have more sophisticated hardware than we can use. We have computers that stay operational 98 percent of the time, cathode ray tubes that are almost 100 percent reliable, plotters that can run 1 to 2 consecutive days without any significant problems, and digitizers that can reliably digitize 1 to 2 mil day after day. We have the hardware to do almost any manipulation of data, but we are not keeping up with the hardware. One of our problems could be an asset: the diversity of interests.

A key problem in manipulating digitized data is inaccuracy in the maps to be digitized. Much of the data base of World Data Bank I, used by by the CAM system, had to be revised. The existing maps were distorted and had compilation errors. In many cases you will have to recompile, go to photogrammetric sources, or make corrections on the sheets to be digitized.

Another problem is organizing the program requirements for the programer. You cannot just write a program by studying someone's operations for a few hours. Writing manipulation programs requires cartographic and mathematical knowledge. Most systems grow; they are not adequately defined in the beginning. In requirements development it is essential that purpose and product be accurately defined. Then a programer, mathematician, cartographer, and a data base specialist working together can help you. The resources needed for manipulation of cartographic data are not usually contained within a single orgranization. Probably no one person in the audience can do everything necessary to complete a cartographic manipulation process.

Before manipulation begins, materials must also be considered. We study everything--Mylar, scribecoat, deformations of paper, output devices, plotters, CRT's, and COM devices. We just see if these devices can actually produce the desired product. Many people do not test their equipment, and they fail.

We have also failed to keep up with hardware because we have not gathered together the resources in the average installation before automating. I have just spoken of what skills are needed. What do you need in dollars and manpower? Even a small system is expensive. For example, the Tektronix tube and interactive plotter on display represent 25-man years, and this system is small. When considering the cost of automated cartography, you must also think of the future: how many people and how much money will be needed to maintain the equipment? In successful systems a great percentage of the initial investment is used to keep manpower in-house or on contract for the first few years in order to maintain and update the system. Maintenance can account for 60 percent of the manpower used to develop the system.

Cost-effectiveness is also an important consideration in automated cartography. Manpower can be exchanged for machines at an even level. I used a modified version of the spaghetti file to generate world shorelines and saved a lot of money. Instead of spending 5 or 6 mo making the map, we spent $120 for computer time. In the Navy alone, we used that data bank enough to pay for its cost. Though to some people it might be a spaghetti data bank, to me it was cost-effective. But there never is a good reason for automating something if you don't save money, if you don't increase speed, or if you don't acquire new capability. Many people, in striving for 100 percent accuracy, spend more money than they would spend getting 98 percent with the automated system and then finishing the job by hand.

My final point is that our data manipulations are not easy enough for the average cartogarpher to use. As a result you may find that your manipulation system isn't used. While we think that is is easy to type in 15 commands on a CRT command program, this task is hard for a technician who has been working on paper. We must combine human engineering with our software and hardware manipulations.

Although I have talked about some of our failures, we are advancing. As long as we keep our feet on the ground, define our requirements, and get the people in our organizations to do the job, automation will be a success.

Chrisman: We spend most of our time processing coordinate points. Therefore, the quickest, cheapest way of reducing the cost of operating the system is to find a way to reduce the number of coordinate points processed and still get the same result. Every existing data base is good down to a certain accuracy, but it is not always used at that level. We have done some work at the Harvard lab with a point reduction algorithm developed by Douglas and Peucker at Vancouver (also independently discovered by someone else). This algorithm was explained in The Canadian Cartographer (1972).

There is some garbage in a file like the New England county DIME file, which has 2163 coordinate points. Using a detailed filter based on the plotter increment of this device, we can get the same picture reduced from 2163 points to 1700 points, a substantial reduction. The internal points of the chain can be dropped out at will because with a topological chain the nodes will remain. We can reduce to 962 points and still get a fairly good representation, although it will begin to look "liney." We can go down further and almost get a characterization, which is still fairly good. If you are using SYMAP, there is no problem.

For every output device there will be an effective level of generalization. That level will allow you to cut costs either substantially or inconsequentially depending on your requirements for accuracy. Consider for example the coarsest piece of World Data Bank II--Australia. In this first picture Australia is plotted at a somewhat smaller scale than it is capable of being plotted at (although not an abnormal plotting scale). (Editor's note: Pictures not available for publication.) It has 18,218 points, a lot to a plotting routine. Plotting Australia at this scale would not be feasible if we were not working on an off-line plotter. The next picture has 1,000 points--a reduction by a factor of 18. There is no visible difference because I used the plotter increment as my step size. This one interaction with the data structure can produce terrific reductions in cost. We should not be dealing in relative coordinates, which require us to keep all the points, but in absolute coordinates because the amount of processing, not the external storage, is the real problem. Using absolute coordinates can really reduce the processing requirements.

McCullagh: I want to consider three main problems with data manipulation. But first, however, I want to say that I agree that, for production, cost effectiveness is a good yardstick. When you are simply automating a present cartographic technique, cost effectiveness can be calculated. As soon as you introduce new methods, however, cost effectiveness is impossible to measure because you can't do much by hand. For example, the color, multivariable mapping and the various possible relief representations are almost impssible to do by hand. You can't expect them to be cost effective because you have no way of determining that effectiveness.

The first problem has to do with access to data. Most people are very worried about the difficulties of data storage and representation of that data set. However, a long-neglected aspect is access to core data. If you have no effective means of directly accessing this data, you will waste valuable CPU time just trying to find what you need. For instance, if you happen to be storing data matrices and you want particular statistics--items, values, or shapes--you have to go through the matrix sequentially, usually using FORTRAN to make everything compatible. A formula translation language doesn't seem very appropriate to cartographic needs because its operations are based on engineering concepts and designs rather than on geographical categories. Working with image processing at the University of Maryland, Foltz and Rosenfeld have tried to develop products like the PACKS language to overcome these accessing problems. The orientated language of such a system allows a user to discover features and boundaries with little difficulty. However, these orientated languages are probably not too fast. Maybe what we ought to be considering as well as cartographic storage and representation is the development of a high-quality language to represent cartographic locational problems.

The second problem is storing large amounts of data. Word packing can economize on the core store but only at the sacrifice of the accuracy of the data. However, in many cases you may need to store a number to 5- or 6-digit precision, and in such cases you can't make any great savings in store by word packing. Various people, again at the University of Maryland, have suggested "bit-plane encoding," a parallel processing capability for either software or hardware. The PACKS language is a software-oriented, parallel processing language that enables you to do operations on different bit planes; a bit plane is simply a binary map of part of the data set. Without even using a special language maybe this approach could speed up cartographic data operations. In many cases you can't store all the data anyway, and you must use row-by-row, column-by-column, or tree-by-tree structure. Then speed depends not so much on the core access system as on the back-up structure. Random-access disk has been around for a long time, and everybody tends to use it at great cost. Even with exchangeable disk packs in many machines, algorithm cost is based on the number of transfers performed. For a research organization this system can quickly become prohibitively expensive.

The third point involves working with a data set. It is not always possible to tell whether your data are clean. For example, in Kansas there is an API file with a tremendous amount of valuable well-log data, but the errors in this file are many. They were discovered during the development of a graphics package called Surface II by the Kansas Geological Survey. The errors weren't noticeable until we started plotting block diagrams or contour maps and suddenly discovered elevation discrepancies of about 1,000 ft higher at one point than at another. This example shows how graphics could be used in an apparently nongraphic field. A well-log system is not usually thought of in a graphic form, but you could discover errors using image processing techniques on these data sets. The error checking sections of the Kansas oil exploration project have produced significant improvements in the API well-log system.

Rhind: Before I discuss generalization, I will follow my U.K. colleague's lead and speak to the chairman because another critical aspect of automation has been omitted--we must think of automation for the future as well as the present. Of course, the manpower situation is critical. Already in some parts of the U.K. we have problems getting draftsmen. On the basis of methods suggested earlier, I have just worked out the way to save colossal, indeed infinite, amounts of money: draw as many maps as possible, the faster the better. The fact that I wouldn't have drawn them before is perhaps irrelevant.

Recently a number of papers have been published on generalization in automated cartography, the most recent of these by Peucker and Douglas in The Canadian Cartographer and by Stewart in The Cartographica Monograph. I don't mean to imply that such recent papers are the best, but that these papers are important and easily obtained. The most important characteristic of generalization in an automated environment is that it permits you to rely on the graphic depiction. This reliance is possible because the rules must be explicit in automation, whereas in manual cartography there often aren't many written rules at all. Even where rules do exist, they are variously interpreted by the individual cartographer.

Even more important though, we can't sensibly involve ourselves in generalization by machine until we understand what we can actually see on the maps. We can go through elegant line simplification routines, but this process may be quite unnecessary; instead, it may be completely satisfactory and much simpler to throw away every second point, or every third point, or *n* points every second. Simpler algorithms are adequate if we can't see any differences between the final products. The recent work in cartographic perception is improving, but we still know very little about what we can actually distinguish on map forms.

Most talks proceed from a definition--I am following the reverse process. Another problem is defining generalization. I don't refer only to line sinosity reduction, the most normal application, but to ways of thinning the available data to reduce the plotting time. When you have 6 different areas and wish to map them as 3, the problem is not only in the representation but also in the data. We have to consider the data characteristics as well as the cartographics.

Let me conclude with two points about line smoothing as a sub-category of generalization. First, there are many different algorithms for line unwinding. Making some assumptions, all algorithms are equally correct unless we have some theory of what is desirable. We must have something better than the present idiosyncratic intuition before we can get very far with automated generalization--unless we define generalization merely as money-saving. Second, line-unwinding routines so far have performed sequential operations. As a result, some algorithms have neatly produced crossing lines. Very few algorithms for line unwinding treat the whole map or even part of it as a map, except for some work by Hans Gottschalk.

Generalization capabilities--when we actually find out what they should be--are an important constituent of a geographic or geometric processing language. Indeed some computer languages already enable us to talk of graphic elements. While we still have a long way to go, generalization should be in such a language.

Niedermair: I'm here to talk with you about a system originally designed to support FAA's National Airspace System. Perhaps the system is rather mundane, compared to some of the other systems that we've talked about. It only produces a product every 28 days, but it is a working system. The system produced a book intended to maintain over 20 data bases at air-traffic control centers throughout the U.S. Because picking the latitudes and longitudes of airways off the charts proved tedious and inaccurate, we were asked to produce a book of these airways to within 0.1 sec. While producing this book, we got requests for data other than the book--chart plots, magnetic tapes, and air route change lists. Among the magnetic tapes produced by the system is a Linetron tape, an automatic typeset tape. The controls for the Linetron, such as the typeset, the font, and the centering of the columns, are put on the tape, and the tape is sent to the Government Printing Office for printing. I have the book here if anyone is interested in the printing quality of this system.

At first we kept the product data separate from the base or fixed data so that the fixed data would not become product oriented. One major advantage of this approach was that a fix (a latitude and longitude in this particular book) was only in the master data set once and could be used in as many products as need be. Another advantage was that the product information could be kept separate, thereby enabling us to change certain products without changing the master data set. When we received requests for products other than the book, separation of product data and fixed data proved especially advantageous.

To identify the charts affected by a particular change in the data base, we have implemented a polygon search routine that determines where a part of the polygon falls within each quadrant of the Cartesian coordinate system, the origin of which is at the search fix. The test runs very quickly--we can tell by testing as few as two sides whether the change is internal or external to the polygon. The system identifies the affected charts by using a cross reference by geographic area between two charts. The geographic area is an arbitrary consideration. For example, 1-degree squared is an entry in the data base. The address of the entry, its latitude and longitude, is in the lower right corner of the grid square. There is an entry for each section of the geographic area covered by the data base.

Each entry in the cross reference has a pointer to a chart record which contains such information as corner postions of the chart and the type of data on the chart. If a fix has a change, longitude is concatenated to the latitude at the fix and used as an entry point to the cross reference. For the latitude and longitude the pointers to the chart record are picked up one at a time. From the chart records a preliminary determination is made if the data are charted at all. If the data are supposed to be on the chart, the corner coordinates of the chart are applied to the polygon search, and the polygon search determines whether the fix is actually within the chart. In general, the software has made a gross determination of the area of the fix and applies to the polygon search only the particular charts that fall or overlap that geographic grid square. With this system we do not test charts in Maine for a fix change in southern California. Thus this search technique is very quick and very efficient.

The cartographer can use the search himself by using the symbolic address of the record in the chart data set. This process greatly reduces the amount of input needed to compile or recompile a new chart. He simply puts in one parameter. The rest of the parameters are taken from the chart record, and the search is applied to the data base. If we get a special request chart, the parameters to the search can be put into the data base by the cartographer. In the system we use a free-form, key-word parameter control. We found early that by using positional parameters to control the system, with a certain punch in a particular column, we had many wrong runs. Using a batch system, our turnaround wasn't too good. Sometimes we had to wait as much as 24 hr to get another run back. I then decided to use a key-word parameter--the system recognizes the word and sets up the controls itself.

Using some slides I will quickly go over the basic data structure of the system. (Editor's note: Slides not available for publication.)

(Slide) The master data set is a type of index sequential file. These dotted lines representing links are the auxiliary set, a strict direct access file on a mass storage device. The master data set is entered using symbolic parameters in the index sequential mode. The auxiliary storage is used in stacking, chaining, and indexing. For instance, an airway gets longer, more area is needed on the disk, and cells in the auxiliary storage are linked together to accommodate the extra area. We wrote the change in a modular fashion using high-level languages for easy transferability and maintenance. The two languages used are COBOL and FORTRAN. The drivers are written in COBOL, the computation subroutines and some of the search routines are written in FORTRAN, and the rest of the search routines are written in COBOL.

(Slide) This plot is a Boston area Air Traffic Control Center (ARTCC) region done by this system. We had a special request from the Boston ARTCC area to plot NAVAIDS, the circles and air routes which are the lines and the boundary of the ARTCC. The boundaries don't show. The plot is rather crude, but as Bolton said, to travel that last mile can cost double what it cost you to go 98 percent of the way. With this plot we're 98 percent there. The cartographer will touch it up, but the software has put the points and the graticule down accurately.

Stepler: The fourth version of CAM is now available. The documentation came off the press Monday. Those of you who attended the CAM workshop, have already received copies. There are some new features in this version: a polyconic projection, UTM grids and ticks, latitude and longitude ticks, legend block blackout, and 12 new line symbols including 4 varieties of dash lines, 6 varieties of railroads, and canal and reef symbols. These features are independent of the plotter.

In our system and in both World Data Bank I and II we decided to use output generalization. (Slide) The first slide shows generalization at various scales. (Editor's note: Slides not available for publication.) We decided to use output generalizations because changing projections distorts the output. A line on one projection takes a different shape on a different projection. Also our scheme eliminates commands to the plotter. (All the projections were produced on the Gerber flatbed plotter.)

(Slide) The algorithm is very simple. We just compare the point of interest with the previous point plotted, and if the delta-x and -y are less than some input value of 0.2, it is not plotted. I have several examples of data from World Data Bank II plotted at the same scale but with different dropout factors. In this example we had a dropout factor of 0.01 in.

(Slide) This slide shows the same area but with a dropout factor of 0.02 in--there is little change. You begin to see the change with the next dropout of 0.04 in. The straight lines lengthen and begin to lose their character.

(Slide) In this last example, with a 0.06-in dropout, the lines really begin to lose character. For most of the work we use the 0.01-in dropout and find it quite satisfactory.

Finally I want to discuss the data structure for World Data Bank II Since we knew that we would have a number of points in the Bank, we needed a different structure than that of Data Bank I. As a result, we came up with two data sets for each feature. One data set is simply a table data set with one record for each line. That record contains a line identifier, the rank value of the line used to determine the map scale on which the line should be plotted, the number of points in the line, and four values for the latitude-longitude limits for the line which serve to window it. We also have a pointer to another data base giving the longitude-latitude values for that line start. The second data base includes simply the latitudes and longitudes set up by about 100 values in each record, and again, retrieval is rather simple.

(Slide) We read a record from the table data set and compare the latitude-longitude limits against the map that is being generated. If the line is outside the limits of the map, it is not read or transformed. Next, we examine the line identifier. The user has the option of selecting all the line identifiers for a feature, selecting a range of line identifiers, or just pulling out one line. Here the line identifiers are checked. The user can also select by rank, either all ranks or individual ranks.

Robe (Central Intelligence Agency): In Bolton's introduction he said that World Data Bank I was compiled, but that is not exactly so--it was taken primarily from one source. World Data Bank II, however, was compiled from numerous sources.

Moritz (PRC Information Sciences Co.): Everyone here should consider the relative cost of hardware and software. When considering the cost of software, it is misleading to consider just the cost of running an item through a machine or just the cost of developing the program. Many different variables should be considered to estimate the cost of software. We must also recognize the fantastic impact of hardware on the cost--the cost to rent IBM, CDC, or UNIVAC equipment, to keep a disk file, and to use the equipment. On the other hand, a minicomputer will cost from $4,000 for stripped-down models to $120,000 for full-blown systems with hundreds of megawords of storage. In the latter case, once the purchase has been made, the operation costs are largely labor. In some laboratories the same people are doing the programing and the research and running the machine.

<u>Chrisman</u>: You can move software from place to place, but hardware is difficult to move. Software people can exchange ideas and give the tools to everybody. The hardware people, who have a physical device, can't be so generous.