Real time 3D display of hydrographic data for QA and QC.

Roland Arsenault
Center for Coastal and Ocean Mapping & Joint Hydrographic Center
University of New Hampshire
roland@ccom.unh.edu

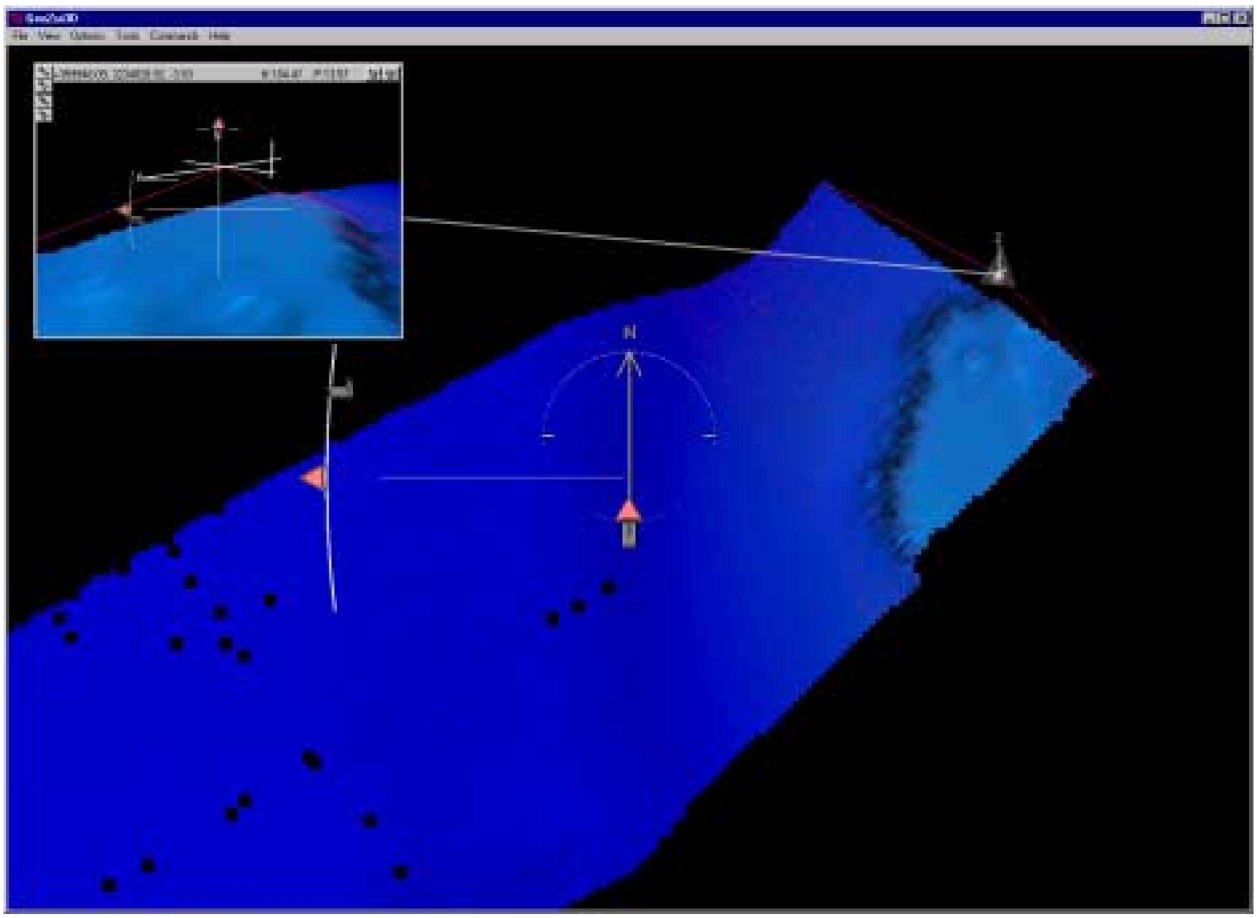

Figure 1 – Data collected from a Simrad EM1002 in the Gulf of Mexico. The main window shows an overview of recent data collected, while the subwindow shows a close-up view of a diagram representing the ship and its attitude.

Abstract

A real time gridding system (RTG) is being developed to enable real time quality control during multibeam sonar surveys. RTG displays georeferenced, gridded data as it is collected from the multi-beam sonar. The system is capable of integrating historic data, such as charts, DTM's and sidescan sonar images, while building a surface that grows as the new data is being collected, all in a georeferenced 3D environment. Other data, such as position and attitude of the ship or towed vehicle can also be displayed in real time in order to give the user a full picture of the complete data acquisition process as it is happening. The real time display of geo-referenced sounding data is an invaluable tool for identifying both systematic artifacts and outliers. The technology that

enables RTG consists of a dynamically growing surface that is gridded on-the-fly and realtime display using variable resolution to maintain interactivity with the system. A number of display modes have been implemented, including displaying of all soundings, displaying the median surface or shoal-biased surface. Both the derived DTM and the underlying soundings can be displayed simultaneously revealing the relationship between the soundings and the DTM.

Introduction

In a complex system such as a multibeam sonar survey system, it is very difficult to eliminate all sources of errors. Some of these errors can be corrected if discovered early enough. It is therefore crucial to pinpoint and correct as many errors as possible, as early as possible to increase data quality, and reduce the amount of error correction necessary in post processing.

This problem came to my attention on my first multibeam survey cruise, a multibeam survey of the Pinnacles area in the Gulf of Mexico [Gardner et al]. During that survey, valuable time was spent debugging the installation of the survey equipment. Time was also spent reprocessing data to correct errors discovered later in the survey. With my limited knowledge of multibeam survey systems, I developed a prototype system to visualize the geo-referenced data as it was collected, along with the position and attitude of ship, so I could better understand the problems encountered. It quickly became obvious that such a system could be very useful to multibeam survey operators.

On a more recent survey of Portsmouth harbor, in New Hampshire by NOAA, it was observed that determining if the proper coverage was achieved sometimes took as much 24 hours as data from the day's survey was processed at night. Immediate feedback of area effectively covered was not available. It was possible to know which areas for which soundings were available, but without post processing, the quality of the data was not known, and the effective coverage could not be easily determined.

To help in identifying possible errors, and help in coverage assessment, a real time gridding system is being developed. The real time grid allows a surface to grow as soundings are added. The system is also capable of displaying the ship's position and attitude. This is useful in finding configuration errors, such as incorrect pitch or roll correction, by matching attitude with artifacts in the soundings and resulting surface.

The system can help in coverage assessment not only by determining what areas are covered by pings, but by immediately displaying a surface, low quality pings might be better identified, and areas of interest can also be identified and extra passes can be accomplished right away, if so desired.

Various methods of integrating new soundings in the surface may be used. The system can use simple averaging or median filters, or could incorporate more sophisticated algorithms to produce a surface.

The following sections will describe the technology developed to enable such a system.

Technology

Today's multibeam sonar collects more data that can be display in a 3d interactive manner. One method of showing a picture of dense hydrographic data is to bin the soundings, then using statistical techniques select a single value to represent that bin. With this collection of representative values, laid out in a regular grid, it is possible to display interactive 3d surfaces of the area covered by those soundings. The following method was developed to allow the data to be assigned to bins, and surfaces to be created in an interactive manner while the data is being collected.

Requirements

The collection of data and real time display of a surface introduces many requirements on a data structure to hold this data. The data structure must allow efficient access to a random bin in which to deposit the incoming geo-referenced data. The data structure must also allow orderly access to the bin values to construct the displayed surface. It is also important, to achieve acceptable rendering speed, to keep the bin values in memory, rather than swap to disk. Another useful feature is to have a structure that can grow without prior knowledge of the bounds of the incoming data. This eliminates the need to establish bounds before data is collected, and allows the flexibility to collect data in areas that might not have previously been defined to be within the bounds of the current survey.

Design

A data structure has been designed with those requirements in mind. The structure is based on a quadtree, with a linked list connecting its node into rows and columns. A quadtree is a structure with two indices, one representing the x value of an x,y pair, and the second representing the y value. The top level of a quadtree represents the whole bounds of the tree. The 4 children of that node represent the northeast, southeast, southwest, and northwest quadrants of that node. Each child can have its own set of children down to the level where the nodes represent an area the size of a single bin. This structure allows for a bin to be found in just a few hops through the tree. The depth of the tree is $\log_2 n$ where n is the number of bins in one side of a square containing all the data positions.

A quadtree is efficient to access bins in a random order, such as at insertion time, but to get efficient rendering, it is important that the values used for rendering are organized in a method that allows fast sequential access. Drilling down the tree to reach each bin is definitely not the most efficient way of getting to those values. The most efficient method would be to have all the values organized into a two-dimensional array. The problem with 2d arrays is that it must all be allocated at its creation time. This makes it hard for the area to expand, and it also uses memory that might not be needed if the area surveyed is not the exact dimension of the array. The next best solution to allow timely access to sequential bins is to have a link from each bin to its neighbors. This allows the quadtree to grow as needed, without the need for a fully allocated array. This also saves on memory requirements, as memory is only allocated for a bin if a sounding is added to that bin.

Implementation

The design was implemented as a hybrid quadtree/linked list. The quadtree is designed to grow from the leaf nodes, where the bins are located. The quadtree structure is initiated when the first sounding is added. A bin is created to hold the sounding, which becomes the quadtree. As more soundings are added, they either fall into an existing bin, or a new bin must be created. When a new bin needs to be created, it may fall within the bounds of the current quadtree, or outside the bounds of the quadtree. For the case where the new bin is to be created within the bounds of the current quadtree, we start from the top, and we go down the nodes until the leaf level is reached. When we encounter a node where the desired child node does not yet exist, that child is created. In the case of adding a node outside the current bounds of the quadtree, a level is added above the top level, which doubles the dimensions of the quad tree in both the x and y directions, effectively quadrupling the area covered by the quadtree. This is repeated until then new bin would fall within the bounds of the new expanded quadtree. We then add the bin using the same method for adding a bin within the bounds of the quadtree. (Figure 2)

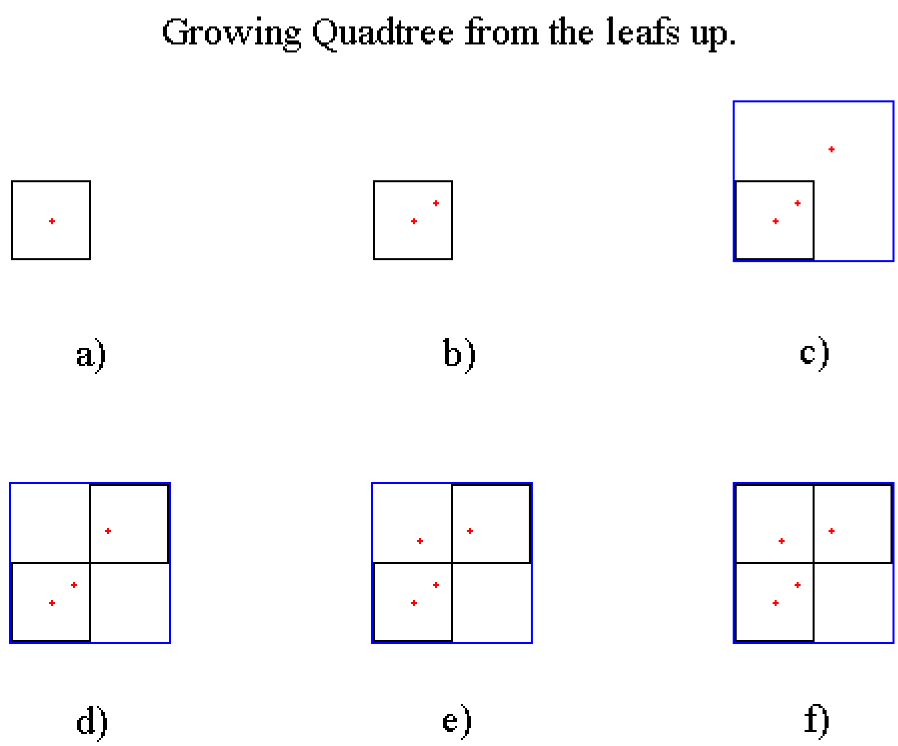

Figure 2 - Growing a quadtree as soundings are added. a) The first sounding is inserted, creating the first bin. b) A second sounding is added which falls in the same bin as the first sounding, therefore it is added to the existing bin. c) A third sounding is added, but falls outside the existing bin. The quadtree grows by adding a level above the existing bin. d) A new bin is created for the third sounding. e) A fourth sounding is added, which falls within the bounds of the quadtree, but not in an existing bin. f) A bin is created for the fourth sounding.

Integrated with the quadtree is a linked list. The linked list links each node with its neighbors. As a node is added to the quadtree, links are added to its neighbors, and the neighbors' links are updated to reflect the added node. The linked list is useful not only for efficient rendering, but also to quickly find the neighbors necessary to calculate the normal vectors necessary for shaded rendering.

Accessing the quadtree structure is done using a pointer to the top-level node. Reaching the linked list for a particular line is done using supporting linked lists. The first list determines which resolution we want the list to traverse. The length of this list is equal to the height of the quadtree. This list allows for the rendering to occur at lower resolutions for faster rendering. Each element of this list contains a list of all the lines running from west to east at the particular resolution. Rendering at a given resolution is done by going down the initial list to the desired resolution. The list contained at that level is then traversed, and each line is rendered from west to east.

The quadtree/linked list structure was designed to support various methods for binning data. This was accomplished by storing an abstract data type at the leaf nodes. The abstract data type supports methods for adding a data item to the node, and to produce a value that reflects the data collected at that node. This value may be an average, a median or any other value implemented in a derived data type. This allows different data types to be used with different space requirements. If a relatively low amount of soundings is being collected, they may be stored using a data type that keeps all the soundings. In a case where more soundings than available memory are being collected, a data type that can compute a reasonable value without having to keep all the soundings can be used.

Future work

The use of a linked list in the rendering of the surface is still a possible source of inefficiencies as the data for the next vertex is not stored contiguously in memory. A pointer must be dereferenced to find each vertex. To solve this problem, an array can be used where all the vertices follow each other in memory. The problems of using an array and the reasons to use a quadtree/linked list hybrid to contain the bins have been discussed. Another possible solution that would reduce some of the wasted space of an array, yet still have some of the benefits of efficient access to subsequent vertices in the rendering process, is to use tiles that are themselves small arrays. These tiles would allow groups of vertices to be contained in the same memory area, which should improve rendering efficiency. The tiles data structure is being implemented, and will be evaluated in comparison with the quadtree/linked list hybrid.

The option to drape collected backscatter imagery or side scan imagery over the derived surface is also being considered. Effective means for storing the extra data and rendering the data efficiently will need to be developed. This could be useful in determining the validity of possible targets, thus determining the quality of the incoming data.

Another enhancement being considered is adding a diagram of the survey vehicle, along with all its sensors, including transducers, motion sensors, GPS antennas, etc. The equipment would be displayed in their actual position on the vehicle, along with all the measured offsets used to calculate proper sounding positions. This might be helpful in finding errors with offsets used in the system

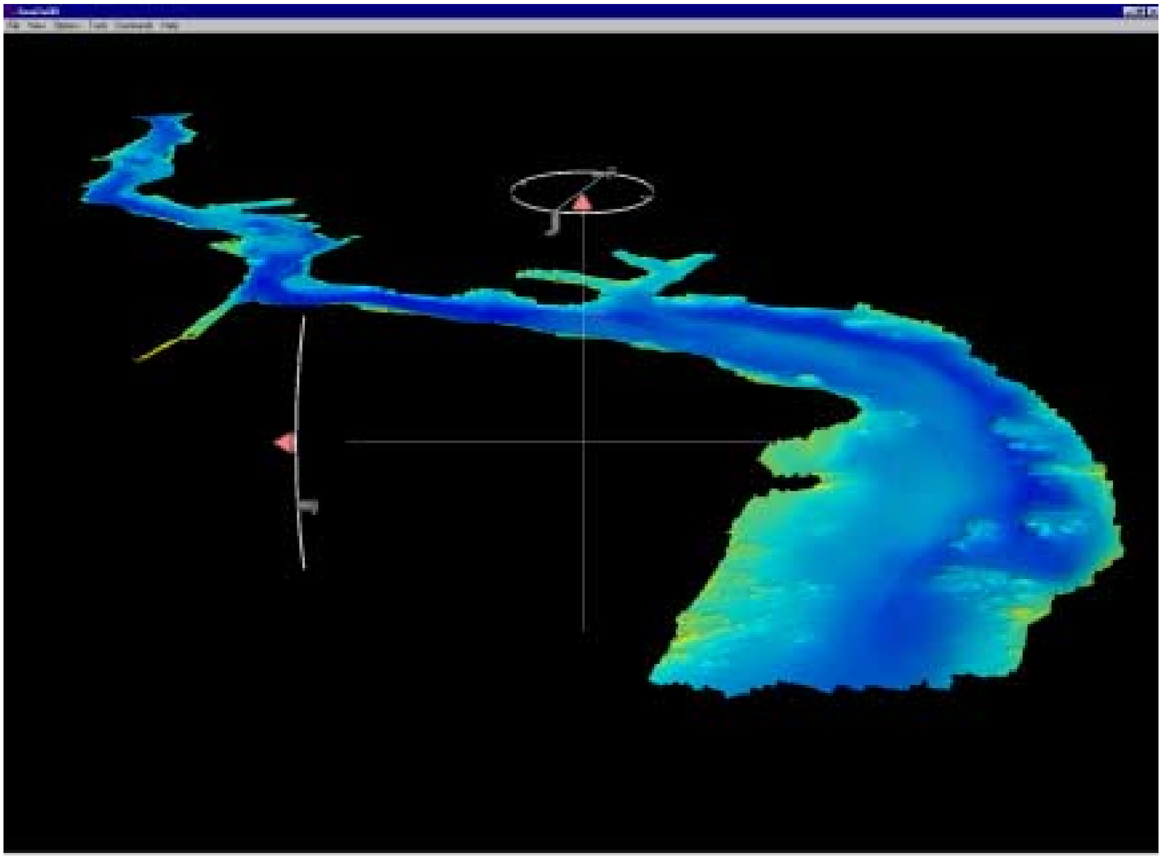

Figure 3- Over 40 million soundings from a Reson SeaBat 8101 collected by NOAA in Portsmouth, NH. The data has already been hand cleaned by the crew of the NOAA Ship Whiting.

Conclusion

A real time gridding system has been developed to help give a clear picture in real time of the data being collected, along with existing data. The system also displays the position and attitude of the vehicle collecting the data. The system has been tested by gridding over 40 million soundings in about half an hour in tests where data was fed to the system at higher rates than realtime. (Figure 3) These test were run on off the shelf PC's with added 3D video cards running Linux or Windows NT. The next step is to test the system in real working environments to find deficiencies in the system, and areas of possible enhancements.

It must be noted that the system is a work in progress, and that many new features need to be evaluated, and added if deemed beneficial. More work is also needed to better display ship attitude data, and also the relationship between the soundings and the derived surface. (Figure 1) The addition of a surface confidence metric could also be useful.

Bibliography

Gardner, J.V., Sulak, J.K., Dartnell, P., Hellequin, L., Calder, B., and Mayer, L.A., 2000, Cruise Report RV Ocean Surveyer Cruise O-1-00-GM, The bathymetry and acoustic backscater of the Pinnacles area, Northern Gulf of Mexico, US Geological Survey Open-File Rept. 00-350.