Developing and Deploying Sonar and Echosounder Data Analysis Software

By Jean-Marie Augustin, Ifremer

While satellite photographs and radar provide high-resolution images of virtually every square meter of the earth’s surface, views of the ocean floor are spottier and less detailed. Yet accurate seabed maps are vital to scientific research and to many industrial applications. Beyond its interest for oceanography, the geosciences, and biology, precise knowledge of the contours and composition of the seabed helps power companies place wind farms and drilling platforms, communications companies plan where to lay fiber-optic cable, and environmental specialists evaluate the effects of climate change on oceans and seas.

The relatively obscure view of the seafloor is not due to a lack of data; ships equipped with multibeam echosounders (MBESs) and sidescan sonar systems survey wide areas of the sea, gathering terabytes of raw oceanographic data. Before scientists and engineers can apply this information in their work, however, they must convert it into meaningful data.

Although I am not an expert programmer, MATLAB® enabled me to apply my expertise in signal processing and sonar to develop and deploy SonarScope®, a high-performance software product for processing, analyzing, and visualizing raw MBES data. SonarScope is highly customizable and provides an original approach to MBES data processing, making it an invaluable tool for researchers and engineers who need to test and calibrate MBES systems.

MATLAB proved to be an ideal environment for developing SonarScope because it enabled me to develop algorithms, visualize results, and then refine the algorithms in an iterative cycle. Design iterations take much longer with a language like C++, which requires additional compiling and linking steps, as well as a significant amount of additional programming to visualize results.

From Basic Algorithms to Standalone Software

When I began developing MBES and sonar data analysis algorithms, I had no intention of creating a complete software package; I simply needed reliable analysis tools for my research. I developed the original algorithms in C. Finding C inefficient because of its slow compile, link, and execute cycle and its inability to check data in the debugger, I turned to MATLAB.

In MATLAB I develop algorithms interactively and can instantly plot the results to see the effect of any changes. Functions from Signal Processing Toolbox®, Image Processing Toolbox®, Optimization Toolbox®, and Statistics Toolbox™ further speed development because I don’t have to write and debug them myself.

Soon after developing the algorithms that would ultimately form the basis of SonarScope I realized that my work could have a much broader reach, and that more of my colleagues could make use of my expertise, if I created a graphical interface to drive the analysis of MBES data. That motivated me to build the SonarScope interface using customized labels and buttons with terminology that my colleagues would readily understand (Figure 1). The interface was well received by my fellow researchers. It also raised awareness of my work among researchers at other institutions.
As interest in SonarScope expanded beyond Ifremer, I used MATLAB Compiler™ to build a standalone software package that can be used by researchers even if they do not have access to MATLAB. Today, SonarScope can be used by any researcher or engineer wishing to process seafloor mapping sonar data.

**Gathering the Data**

To survey the seafloor, a ship equipped with an MBES covers the area to be mapped, following a series of parallel lines. The swaths covered on both sides of the survey lines can be up to 20 kilometers wide. They overlap to ensure that no area is left unmapped. As the ship travels along each swath, the MBES transmits a series of high-intensity, short-duration *pings*. Each ping is received and processed inside several hundred beams, which are steered in a span from directly below the ship to the edges of the swath (Figure 2).

For each beam, the MBES records the time lapse between transmitting the signal and receiving the echo. This metric is used to compute the oblique range from the ship to the ocean floor spot "seen" by the beam and, ultimately, to build a digital terrain model depicting the seafloor. The intensity of the echo is recorded because it correlates with the reflectivity of the seabed and hence with its physical characteristics. This reflectivity can be used to distinguish rock, sand, vegetation, and other features of the seabed.

To produce a map of the seafloor that is accurate to within centimeters, algorithms must take into account the echo delay, the transmit angle of the beam, and the motion (roll, pitch, and heave) of the ship. All this raw data is recorded, together with the ship’s latitude and
longitude for each ping transmitted by the MBES. Gigabytes—sometimes terabytes—of data are collected on a single survey, which usually lasts from a few days to several weeks.

**Performing Complex Transformations**

Converting the raw data acquired from the MBES equipment into seafloor maps requires complex transformations involving multiple geometries. Using MATLAB and Mapping Toolbox™, I developed algorithms that process the data for a single ping and calculate the depth of the seafloor covered by the ping (Figure 3). In addition to identifying traits of the seafloor, the algorithms also detect features of the water. They can, for example, detect bubble plumes caused by gas released from the seafloor.

![Figure 3. A 3D compilation of bathymetry data processed by SonarScope.](image)

Beyond bathymetry, SonarScope performs many different kinds of data processing, all supported by MATLAB and related toolboxes. I used MATLAB, Statistics Toolbox, Image Processing Toolbox, and Signal Processing Toolbox to implement noise reduction, speckle filtering, segmentation, and bottom detection techniques. I used Optimization Toolbox for curve fitting throughout SonarScope. I plan to replace my own cartographic projections with the ones provided in Mapping Toolbox.

The segmentation algorithm is based on texture analysis using co-occurrence matrices and Gabor filters (Figure 4). The frontier lines were drawn automatically by the MATLAB based segmentation algorithm.

![Figure 4. Before-and-after images showing the results of four image processing techniques. Top to bottom: mosaicing, speckle filtering, segmentation, and bottom detection. The speckle filter was designed by Pr. Alexandru Isar, Politehnica University, Timisoara, Roumania. The segmentation algorithm was developed by Imen Karoui, ENST Brest.](image)

**Applying Object-Oriented Programming Techniques to Handle Extremely Large Data Sets**

SonarScope comprises about 270,000 lines of MATLAB code. Similar software packages written in C commonly contain a million or more lines of code. Nevertheless, developing and maintaining any project with hundreds of thousands of lines of code can be a challenge without a way to organize and reuse the code efficiently. I used the object-oriented (OO) programming capabilities of the MATLAB language to create classes for components that are reused frequently throughout the application. For example, I defined a class for images, which makes it easy to manage and manipulate instances of images by changing parameter values.
Handling the extremely large data sets that SonarScope processes was also made easier by applying OO principles. When performing operations on large matrices holding gigabytes of data, it's easy to run out of computer memory. To solve this problem, I created a class that uses a MATLAB `memmapfile` object to map dynamic memory to files on a hard disk. Using this approach, I can easily work with 10,000 x 10,000 matrices and yet use only 120 bytes in memory. With this class, access to the values of a variable is the same whether the variable is a MATLAB matrix or a `cl_memmapfile` object.

The SonarScope interface takes advantage of reusable class objects and OO design patterns enabled by MATLAB to provide drag-and-drop capabilities, a property editor, keyboard shortcuts for panning and zooming, and custom menus.

**Building a Standalone Application**

Many of the researchers, scientists, and engineers who use SonarScope are familiar with MATLAB. However, not every user is a MATLAB expert, nor do all users have MATLAB installed on their workstations. To deliver a solution to these users, I used MATLAB Compiler to create standalone 32-bit and 64-bit versions of SonarScope for Windows® and Linux® operating systems.

With its detailed and customizable data processing capabilities, SonarScope excels at helping engineers verify, calibrate, and troubleshoot their MBES hardware. When MBES systems produce extraneous artifacts that show up in the data, engineers use SonarScope to examine the data dynamically and identify the source of the artifacts. In one case, to minimize the number of spikes in their bathymetry data, an MBES manufacturer replaced the bottom detection algorithm they were using with the one from SonarScope.

Development of SonarScope is active and ongoing. I continue to update the code to take advantage of new features as they become available in MATLAB, and to enable me to collaborate effectively with more research colleagues worldwide.

---

**SonarScope Projects at the University of New Hampshire**

By Mashkoor Malik, University of New Hampshire

At the University of New Hampshire (UNH) Center for Coastal and Ocean Mapping, we use SonarScope to assess backscatter characteristics from multibeam sonar systems. Backscatter is the amount of acoustic energy returned to the sonar. Multibeam sonars are typically used for measuring the depth of the seafloor and for acquiring a continuous 3D map of the seafloor. Multibeam backscatter research is an emerging field with many promising applications. For example, it enables classification of seafloor sediments, characterization of fishery habitats, assessment of suitable routes for transoceanic cables, and identification of mineral and natural resources.

SonarScope accelerates seafloor backscatter analysis by enabling us to quickly evaluate the quality of our data and apply a range of data acquisition and processing parameters.

On a recent research project, we needed to analyze two weeks of continuous MBES sampling that produced more than 600 GB of data. We saved months of code development time on the project by using SonarScope to preprocess the data, which we then integrated into MATLAB for further analysis. The work we do would be much slower—and in some cases impossible—without SonarScope.

*Mashkoor Malik is a PhD candidate at UNH, where he works with Professor Larry Mayer, director of UNH’s Center for Coastal and Ocean Mapping.*
References


Products Used
- MATLAB
- Image Processing Toolbox
- Mapping Toolbox
- MATLAB Compiler
- Optimization Toolbox
- Signal Processing Toolbox
- Statistics Toolbox

Learn More
- SonarScope
- Video: Deploying Applications with MATLAB 2:00
- Introduction to Object-Oriented Programming in MATLAB

See more articles and subscribe at mathworks.com/newsletters.