Automatic Outlier Detection from Shallow Water Multibeam Data Using Median Filtering

by

Manjinder Mann
B.Eng, Nagpur University, India, 1994

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of

MASTER OF APPLIED SCIENCE

in the Department of Electrical and Computer Engineering

We accept this thesis as conforming
to the required standard

[redacted]

Dr. P. Agathoklis, Supervisor, Dept. of Elect. & Comp. Eng.

[redacted]

Dr. A. Antoniou, Supervisor, Dept. of Elect. & Comp. Eng.

[redacted]

Dr. J. Huang, Outside Member, Dept. of Mathematics & Statistics.

[redacted]

Dr. J. Zhou, External Examiner, Dept. of Mathematics & Statistics.

Supervisors: Dr. P. Agathoklis and Dr. A. Antoniou

ABSTRACT

Accurate and fast processing of shallow water multibeam echosounder data is necessary to obtain high resolution bathymetric maps of a seafloor. The manual processing methods which are currently being used to identify the outliers in multibeam data, though accurate, are time consuming due to the large amount of data involved. In this thesis, an automatic outlier detection method is proposed to identify outliers from shallow water multibeam data fast and accurately.

Initially, two methods, robust estimation and median filtering, are described and implemented as possible candidates for an automatic method to detect outliers. Both methods are evaluated using synthetic data and based on the results obtained, median filtering is selected as the most promising candidate for automatic outlier detection. A new automatic outlier detection method is then proposed based on a two-stage median filtering algorithm. The method consists of three main parts, the preprocessing, the first stage, and the second stage of the two-stage median filtering algorithm. The preprocessing of the multibeam data is done to facilitate the implementation of a localization method used in the two-stage median filtering algorithm. The selection of parameters used in both stages of the median filtering is done based on the properties of the multibeam echosounder system and the multibeam data. Multibeam field data sets obtained from the Institute of Ocean Sciences (IOS) are used to evaluate the performance of the proposed method. The processed multibeam data sets using the proposed method are compared with the results obtained by experienced operators at the IOS manually. The evaluation of the results indicate that over 95% of the oultiers are detected and true objects on the seabed are preserved. The results are validated using visualization of the bathymetric images generated from the multibeam data sets.

Examiners:

[redacted]

Dr. P. Agathoklis, Supervisor, Dept. of Elect. & Comp. Eng.

[redacted]

Dr. A. Antoniou, Supervisor, Dept. of Elect. & Comp. Eng.

[redacted]

Dr. J. Huang, Outside Member, Dept. of Mathematics & Statistics.

[redacted]

Dr. J. Zhou, External Examiner, Dept. of Mathematics & Statistics.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

CHS	Canadian Hydrographic Services
DTM	Digital terrain model
IRLS	Iterative re-weighted least squares
IOS	Institute of Ocean Sciences
MBES	Multibeam echosounder system
SBES	Singlebeam echosounder system
SONAR	Sound navigation and ranging

Acknowledgement

I would like to express my gratitude to my supervisors, Dr. Panajotis Agathoklis and Dr. Andreas Antoniou, for their valuable guidance and comments throughout my graduate studies. I would also like to acknowledge the financial support of my supervisors, without which I would not have been able to undertake this work. I also greatly appreciate the help and support given by the staff at the Institute of Ocean Science specially Terry Curan, Rob Hare and Doug Collins. I would also like to thank Dr. John Hughes-Clarke at the University of New Brunswick for giving me helpful suggestions during my research.

I would like to thank my supervisory committee, Dr. Jing Huang for the guidance and consultation they provided me in the research. I would also like to thank the computer and secretarial staff of the Department of Electrical and Computer Engineering, in particular Vicky Smith and Catherine Chang, for the support that was extended. I would like to thank all my friends at University of Victoria who became integral part of my student life and extended their help both academically and otherwise.

Finally, I would like to thank my family who encouraged me and gave me the will to continue.

Dedication

Dedicated to my family

Chapter 1

Introduction

1.1 Multibeam Echosounder Systems

Bathymetry is the area of science that deals with the measurement of ocean depth in seas and lakes as well as the processing of the information derived from such measurements [1]. An image map of ocean depths is known as a bathymetric map. Over the past sixty years, acoustic echosounding has dominated the field of bathymetry. The use of sound to measure water depth can be traced back to World War I [2]. One of the earlier instruments used for ocean topography was the echosounder [3]. The technique of echosounding, first used by German scientists in the early 20th century, uses sound waves bounced off the ocean bottom. The technology of echosounding has improved through the years with the introduction of more accurate and reliable equipment. There are two primary systems for the acquisition of acoustic bathymetric data: the singlebeam echosounder system (SBES) and the multibeam echosounder system (MBES).

A conventional singlebeam echosounder system consists of a single hull-mounted transducer that acts both as an acoustic transmitter as well as a receiver (transceiver). The echosounder system produces a vertically transmitted acoustic pulse with a single frequency, typically within the $100 - 300$ kHz range. The transducer produces an acoustic pulse with a cone angle of $3 - 8°$, oriented vertically downwards as shown in Fig. 1.1, thereby concentrating the energy of the transmitted pulse in a circular area on the seabed. The radius of this circular area is primarily dependent upon the water depth, i.e. the deeper

the water, the larger is the radius of the circular area ensonified by the echosounder. The returned echo from the ensonified area on the seabed is received by the transducer, amplified electronically, and recorded. The time taken for the sound to travel through the sea and back is then used to calculate water depth. The sooner the sound waves return, the smaller the water depth and the higher the elevation of the seabed. The echosounder repeatedly *pings* the seabed as the ship moves along the water surface, producing a continuous line showing ocean depths directly beneath the ship.

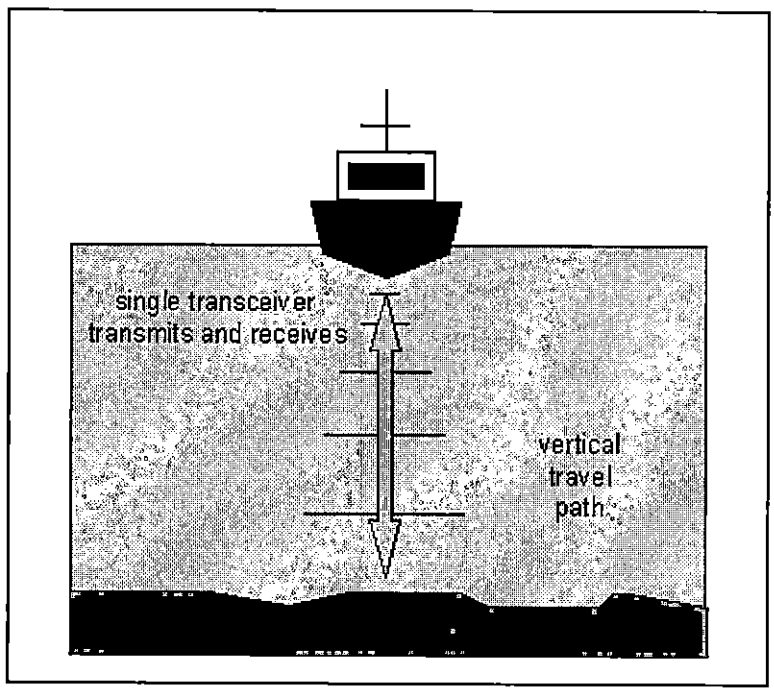

Figure 1.1. *A Singlebeam Echosounder System.*

Since the 1950s, this technology has evolved in two different ways. One way is to mount a boom across the ship and install a number of transducers along the boom. At one specified instant of time, all the transducers transmit pulses in sequence to the bottom and thus a number of depths are measured. This kind of system is in effect a *multitransducer* system. It is used in surveys in inland lakes, rivers, and open harbors with calm water. The other way is to design a transducer so that at one specified instant of time, acoustic energy

is transmitted in a sector and a number of discrete beams are received within the sector. The sector is wide in an athwartship plane, i.e. perpendicular to the ship track, but narrow in the fore-aft plane, i.e. parallel to the the ship track. The beams in the sector are received either to have a uniform angular spacing within the sector or to have a uniform distance spacing in the intersection of the sector with the sea floor. Such a system is called a *multibeam* system. It was first used for offshore deep water surveys. Recently, it has started to be used for medium and shallow water bathymetric mapping. With the extensive application of multitransducer and multibeam systems, their names and the meaning of these names have been changed. Some authors call these two systems *sweep* and *swath* systems, respectively. Other authors refer to both of these systems as swath systems but the second type as *fan-type* swath systems. The term swath is related to the fact that the coverage of a survey line in both systems is a strip instead of a line in the traditional singlebeam mapping. In this thesis, the term multibeam systems refers mainly to fan-type swath systems.

Over the last decade, medium and shallow water multibeam bathymetric mapping systems have been produced. Compared to the singlebeam mapping systems, these systems have the advantages of 100% coverage of the seafloor, high rates of data collection, and wide swath coverage. A multibeam echosounder system is based on the fact that more beams are better than one. About 30 years ago, the US Navy developed a system that sends out many acoustic beams simultaneously to get a series of water depth readings along the line of a moving vessel. With recent advancements in acoustic transducer design and digital signal processing, the use of multibeam echosounders in bathymetry has grown rapidly. Multibeam echosounder systems are highly efficient and offer the only way of obtaining *full sea bottom coverage.*

A typical multibeam echosounder system consists of a single transducer array. The transducer array consists of several elements called *staves*, which transmit and receive acoustic energy independently, and a unit to connect staves and to control the transmission and reception. Different types of transducer arrays, such as circular or flat arrays, are used depending on the system and the application. The transducer array transmits a fan

of acoustic energy into the seawater as shown in Fig 1.2. The available frequencies range from 100 kHz to 450 kHz. Acoustic energy is transmitted in a wide sector in an athwartship plane, typically 120°. When the transmitted acoustic energy hits the seabed or some underwater structure, reflection or backscatter acoustic energy is generated which returns to the transducer array as shown in Fig. 1.2. The backscatter energy is received in a number of narrow channels which have different angles with the normal of the transducer array. The received acoustic energy in each narrow channel is called a beam. The angle between the center of the beam and the normal of the transducer array is called a beam angle. Usually, the outer parts of the swath are ensonified later than the inner parts. Hence, the echo signals from the outer parts of the swath arrive later and form a larger angle as shown in Fig. 1.2.

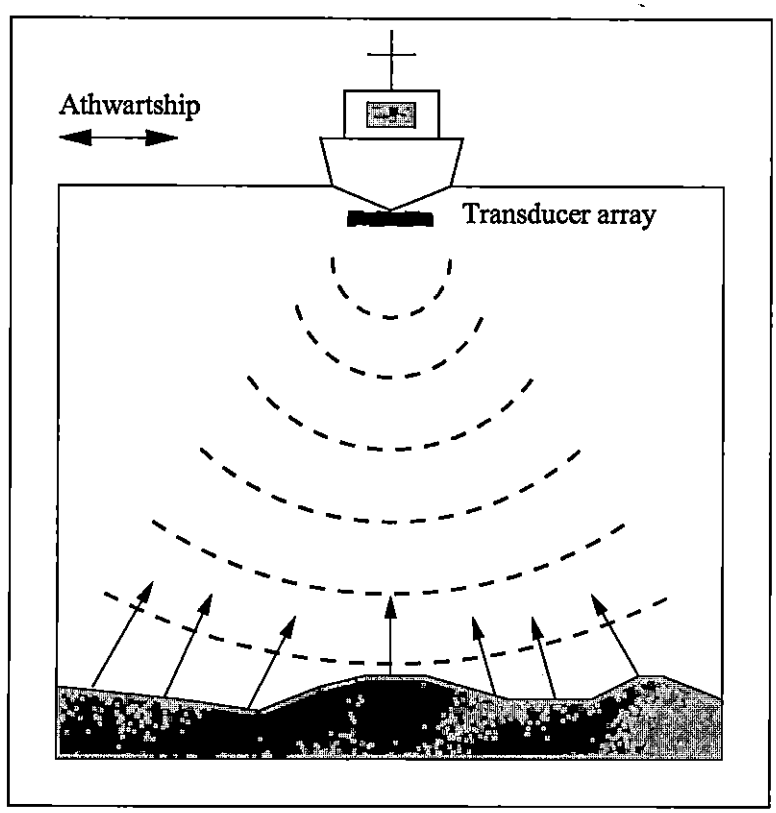

Figure 1.2. *Operation of multibeam echosounder system.*

From the angle and the travel time (or rather the range, which is the product of the

two-way travel time and the speed of sound), the echo location and the depth at each echo location is computed. As shown in Fig. 1.2, a multi-element transducer array provides many (30-150) individual soundings of the water depth and echo strength for each ping. Automatic seafloor tracking programs determine depths and echo strengths for each transducer element, correct for transducer motion, and calculate a geographic coordinate for each individual sounding. A wide swath (up to 7 times the water depth) can be surveyed in a single pass through an area. Survey lines are spaced to provide overlapping coverage of the seafloor. The data acquired are used to generate high resolution images which contain information about the morphology of the seafloor. Multibeam echosounding can detect features which are not always detected by singlebeam echosounding.

1.2 Concept and Operation

The thesis is based on a project sponsored by Institute of Ocean sciences (IOS), Sidney, BC, Canada. Canadian Hydrographic Services (CHS) operates Simrad EM3000 which is a 300 kHz multibeam echosounder capable of mapping the seafloor at depths between 3 and 70 meters below the transducer. The post-processing of EM3000 multibeam echosounder data acquired by CHS is performed at the IOS. The objective is to generate high-quality bathymetric maps of the seafloor from the multibeam echosounder data. The multibeam data contains sparse erroneous soundings which do not correspond to the soundings in their neighborhood. These soundings are termed *outliers*. The occurrence of outliers can be explained by

- surface reflection
- reflection from fish shoals
- low signal-to-noise ratio in bad weather conditions
- turbulent flows with bubbles in front of the transducers

Outliers are random in nature and they can be considered as impulsive error in the multibeam echosounder data. Although outliers are sparse in numbers, it is very time consuming to identify them from the huge amount of data acquired by multibeam echosounder systems as the processing methods in use are mainly manual. The latest generation shallow-water multibeam sonar can acquire more than $1,000,000$ depth measurements per hour [4]. For example, the density of data for a Simrad EM3000 running at 10 knots at a depth of 10 meters is around $25,000,000$ points/km^2 and it can acquire more than $10,000,000$ data points in one hour [5]. With this amount of data, even a low percentage of erroneous data (outliers) can take a long time to be detected and removed. An automatic outlier detection method is necessary to speed up the post-processing of multibeam echosounder data. An automatic method should be capable of detecting most of the outliers in a very short time and it should not miss any objects on the seafloor.

1.3 Literature Survey

With the advancement in multibeam echosounder system technology, interest in automating the outlier detection process has grown considerably. Different approaches are found in the literature, such as:

- The simplest algorithms filter outliers using slope based criteria, which only flag the most severe outliers and produce poor results on low noise shallow water data [6].
- Mitchell [7], Eeg [8], and Claussen and Kruse [9] select an "environment" around each beam (data point) and the local standard deviation is used as a criterion for erasing single measurements. The standard deviation for the data in the environment is calculated from a fitting plane or surface. These methods work well but are unrealistically time consuming for large data files.
- Ware et al. [10] divide the data set into cells and some statistical values are estimated for each data set. These statistical values are then used to classify data points as valid or as outliers. These authors concluded that automatic detection by their algorithm

requires three times the normal coverage of the seafloor. Such an algorithm can therefore only be used as an aid to interactive outlier editing.

- Dijkstra et al. [11] developed an automatic algorithm based on filtering. Frequencies that are larger than that of the seafloor topography are rejected and the remaining frequency spectrum is used to construct the clean data. This technique requires knowledge of the seafloor topography in the frequency domain and is not robust with respect to outliers.
- Bourillet et al. [12] produced a software module that can be used for outlier elimination, which is included in a commercial package called TRISMUS. It uses more than one method to select the outliers: a quartile method and a slope method. The method is not fully automatic and some operator editing is strongly suggested.
- Du et al. [13] proposed a method based on a statistical analysis of the data in a working window, the dimension of which starts from the whole data set and decreases or increases during data analysis. The outlier elimination on the working windows is based on data clustering based on the assumption that the noise in the data follows a uniform distribution. This method is feasible for automatic outlier elimination because the data analysis can be carried out in realistic time even though the algorithm elaborated by these authors "may be far away from practical applications" [11].
- Bisquay and Debese [14] used a robust estimator to detect outliers from the the multibeam data. The algorithm is based on a local model of the seabed. The fitting of a quadratic surface over the raw data is carried out using a *Tukey* robust estimator [15]. Detected outliers are soundings with high residual values between the measured depths and the depths estimated from the local model. The method has been shown to give promising results for deep water multibeam data.

Some of the cited methods were tested on data sets of limited dimensions (10,000 data points) and would be unrealistically time consuming on large data sets ($> 100,000$ data points).

1.4 Outline of Thesis

This thesis presents a method that can automate the process of outlier detection in the multibeam echosounder data. The thesis is organized in five chapters.

In Chap. 1 the thesis topic is introduced. Some background information about multibeam echosounder systems is provided and relevant details about the working of the MBES are discussed. The motivation behind the problem considered in this thesis is presented and it was observed that an automatic method is required to speed up the post processing of the shallow water multibeam data. Some automatic methods available in the literature are discussed and the organization of the thesis is presented.

In Chap. 2 two methods, robust estimation and median filtering, are presented as possible candidates for automatic outlier detection. The two methods are evaluated using synthetic data and results obtained are presented. Based on the results, median filtering is found to be very fast and computationally less extensive and is selected as a method to detect outliers. The results using the median filtering method are further investigated and some suggestions to improve the performance of the method are discussed. Conclusions based on the results are presented.

In Chap. 3 a new automatic outlier detection method based on a two-stage median filtering algorithm is presented. The method consists of three main parts, the preprocessing, the first stage, and the second stage of the two-stage median filtering algorithm. The preprocessing is done to facilitate the implementation of a localization method used in the two-stage median filtering algorithm. The two-stage median filtering algorithm is devised based on the suggestions made in Chap. 2. The first stage of the algorithm is used to detect potential outliers and the second stage of the algorithm is used to revalidate some of the potential outliers that are located on the boundaries of the true objects. The selection of the parameters used in the algorithm is discussed and the conclusions drawn are presented.

In Chap. 4 the multibeam field data obtained from the IOS is used to evaluate the performance of the automatic outlier detection method presented in Chap. 3. The selection of

parameters used for the multibeam field data is discussed. The results obtained are evaluated using direct comparison of the locations of the outliers detected by the proposed automatic method with the locations of the outliers detected by the experienced operators at the IOS. The results indicate that the proposed method is very fast and is capable of detecting most of the outliers in the multibeam data without removing the true objects on the seafloor. The results are further validated using visualization of the bathymetric images generated from the multibeam data sets.

In Chap. 5 the conclusions drawn from the work are presented. The selection of median filtering as an outlier detection method and the proposed automatic outlier detection method are summarized. The results obtained using the proposed method indicate that the proposed method is able to detect, in a very short time, over 95% of the outliers in the multibeam field data without removing true objects. Some suggestions for future research are presented at the end of the chapter.

Chapter 2

Analysis of Two Outlier Detection Methods

2.1 Introduction

In the field of multibeam survey, post-processing of the multibeam data takes much more time as compared to the time taken for data acquisition. A crucial step in the post-processing is to remove outliers from the data set. Two approaches are generally considered for the identification of outliers [14]:

- The first approach entails visualizing all the soundings of a survey and then removing the doubtful soundings manually. This procedure requires powerful and versatile graphic tools and is very time consuming.
- The second approach uses automatic tools which remove the soundings identified as doubtful according to some rules.

Both approaches have their pros and cons. The manual approach is painfully slow but, on the other hand, the automatic approaches presented so far in the literature are prone to corrupt the data leading to incorrect results. The ideal approach would be to have an automatic detection algorithm that

- correctly identifies most of the outliers present in the data,
- keeps data corruption to absolute minimum, and

- is time efficient.

In this chapter, the effectiveness of two different methods in identifying potential outliers in multibeam data is evaluated. The first method uses a robust estimator to identify outliers from the data. This method is appropriately termed as *robust estimator* [14] method and is described in detail in Sec. 2.2. The second method uses a median filter to detect the outliers from the data. This method is called *median filtering* [16] method and is described in detail in Sec. 2.3. In Sec. 2.4, the performance of each method is evaluated using synthetic data. The synthetic data used is a model of the seabed and hence represents the multibeam data. A detailed discussion of the results is presented in Sec. 2.5. Based on the results and discussion, median filtering is selected for further investigation. Some suggestions for improving the performance of median filtering as a method to detect outliers are presented in Sec. 2.6 and will be used in chapter 3. In Sec. 2.7, conclusions are drawn.

2.2 Outlier Detection Using a Robust Estimator

In the robust estimator method, the detection of outliers relies on a local model of the seabed using a quadratic surface which is generated using a robust estimator. The *Tukey* robust estimator [15] is used because of its adaptive capabilities.

The method is based on the assumption that the topography of the seabed can be approximated using patches of quadratic surfaces. The geographic area is divided into square cells each of size $L \times L$ and a quadratic surface is fitted to the data in the cell. Consider a quadratic surface given by

$$z = x^2 a_5 + y^2 a_4 + xy\, a_3 + x\, a_2 + y\, a_1 + a_0 = \mathbf{x}^T \mathbf{a} \tag{2.1}$$

where

$$\mathbf{x} = [x^2 \;\; y^2 \;\; xy \;\; x \;\; y \;\; 1]^T$$

and

$$\mathbf{a} = [a_5 \quad a_4 \quad a_3 \quad a_2 \quad a_1 \quad a_0]^T$$

is a column vector of real coefficients. In order to fit a surface to the data in each cell, a standard estimation procedure such as a *least squares* method without any weighting of the data points cannot be used. The reason is that in the *least squares* method all the data points are taken into account in the same way to estimate the surface and as the data contains outliers, they would distort the surface and lead to an erroneous seabed representation [17]. In a robust estimator, the weights for each data point are adaptively modified and, as a result, not all the data points are taken into account for the surface fitting [14].

The robust estimator fits the majority of the data and usually can be computed using *iterative re-weighted least squares* (IRLS). The robust estimator is not sensitive to small proportion of outliers in the data. The iterative construction scheme is based on the generalized least-squares scheme consisting of several iterations. In the first iteration, a quadratic surface patch is fitted to the data in the selected cell of size $L \times L$ using the same weight for all data points. The residual values, r_i, between the data points on the fitted surface and the data points on the original surface provide the information to compute the weights, w_i, that will be used in the next iteration. In each iteration, coefficients a of quadratic surface are calculated as follows:

$$\hat{\mathbf{a}}^{(j)} = \arg\left\{\min_{\mathbf{a}} \sum_i w_i^{(j-1)} \left| z_i - \mathbf{x}_i^T \cdot \mathbf{a} \right|^2 \right\} \tag{2.2}$$

with

$$r_i^{(j-1)} = \left| z_i - \mathbf{x}_i^T \cdot \hat{\mathbf{a}}^{(j-1)} \right| \tag{2.3}$$

where $\mathbf{x}_i$ and z_i represent the location and the depth value for sounding i, and

$$w_i^{(j-1)} = \begin{cases} \left[1 - \left(\frac{r_i^{(j-1)}}{\alpha \cdot r_{median}^{(j-1)}}\right)^2\right]^2 & \text{if} \quad r_i^{(j-1)} < r_{median}^{(j-1)} \\ 0 & \text{otherwise} \end{cases} \tag{2.4}$$

Eq. (2.4) allocates a weight between 0 and 1 to each point which depends on its residual value r_i. The two extremes 0 and 1 indicate possible outliers and data points very close

to the seabed, respectively. The above robust estimator is called a *Tukey estimator* [18]. Soundings having a weight equal to zero at the last step are detected as possible erroneous soundings. The Tukey estimator has adaptive capabilities since the threshold of rejection of the soundings changes from one step to the next one (Eq. 2.4). It depends on the median value $r_{median}^{(j-i)}$ computed over all the residuals in the cell and on α which is known as the *inverse sensitivity* of the estimator [14].

The method requires the setting of two parameters, namely the inverse sensitivity α, and the size of the cells, $L \times L$. The inverse sensitivity is a parameter used to compute the weights for each data point. To detect outliers, the difference between the original data and the surface generated by the robust estimator is computed for all points with zero weight. A histogram of these error values is generated. A threshold is selected and all data points corresponding to error values above the threshold are identified as outliers.

2.3 Outlier Detection Using a Median Filter

Median filtering is a nonlinear signal processing technique which is useful in reducing impulsive or salt-and-pepper noise. It is also useful in reducing random noise while preserving edges in an image [19]. This edge-preserving property of median filters can be very useful in processing multibeam data as the edges in the multibeam data may represent rocks or crevices in the otherwise flat seabed.

The median of n numbers $x_1, x_2, \ldots, x_n$, for n odd, is the middle number in size. For example, if all n numbers are ordered into an ascending array X_i i.e. $X_i = x_1, x_2, \ldots, x_n$ so that $x_1 < x_2 < x_3$ and so on, the median of this array would be

$$y_i = \text{Median}\, X_i = x_v \tag{2.5}$$

where $v = (n+1)/2$. The median is denoted by

$$\text{Median}(x_1, x_2, \ldots, x_n) \tag{2.6}$$

where $x_1 \leq x_2 \leq x_3 \leq \ldots \leq x_n$.

A *one-dimensional median filter of size n*, for odd n, whose input is a sequence $\{X_i, i \in \mathcal{Z}\}$ i.e. $X_i = x_1, x_2, \ldots, x_n$, is an algorithm that will perform the operation [20]

$$y_i = \text{Median } X_i \triangleq \text{Median}(x_{i-v} \ldots, x_i, \ldots, x_{i+v}), \quad i \in \mathcal{Z} \tag{2.7}$$

where $v = (n+1)/2$, and $\mathcal{Z}$ denotes the set of all natural numbers. However, as we are dealing with a surface (seabed), 2-D filtering is required. A *two-dimensional median filter with window size* $L \times L$ on a surface $\{X_{ij}, (i,j) \in \mathcal{Z}^2\}$, is an algorithm that will perform the operation [20]

$$y_{ij} = \text{Median } X_{ij} \triangleq \text{Median}(x_{i-r,j-s}, \ldots, x_{i,j}, \ldots, x_{i+r,j+s}) \tag{2.8}$$

where $(r,s) \in A, (i,j) \in \mathcal{Z}^2$.

In a two-dimensional median filter, a window slides over the surface, and the median value of the sample points within the window becomes the output value corresponding to the sample point being processed. An important parameter in using a median filter is the size of the filter window. Median filters are well suited for suppressing *impulsive noise* provided that the size of the window is chosen to be at least twice the width of the impulses [19]. If this is the case, noise impulses which are sufficiently separated will be completely eliminated by the median filter. However, impulses lying close to each other may not be removed, depending on the window size. The window size is an important parameter and has to be chosen carefully for the median filtering method to work properly.

Median filtering can be used to detect outliers in the multibeam data. The data is filtered using a median filter of suitable window size, and the filter output is subtracted from the original multibeam data and a histogram of the error values is generated. Based on the histogram, a threshold is then selected and if the error value for a data point is larger than the threshold, this point is identified as an outlier.

2.4 Results

The two detection methods discussed in the previous section have been applied to synthetic data. The use of synthetic data helps in evaluating the performance and the properties of these methods.

The criteria used to evaluate the performance of each method are

$$\text{Detection rate} = \frac{\text{number of actual outliers detected}}{\text{total number of actual outliers}} \tag{2.9}$$

$$\text{False detection rate} = \frac{\text{number of false outliers detected}}{\text{total number of actual outliers}} \tag{2.10}$$

and computation complexity.

Since the number and location of outliers are known for the synthetic data, the detection rate and false detection rate can be easily calculated according to Eqs. 2.8 and 2.9. The computation complexity is also readily measured by calculating the number of flops (floating point operations) required by each method to process the same synthetic data.

2.4.1 Synthetic data

Seabed can be of any shape and the modeling of a typical seabed can be complicated. The synthetic data is used for testing the ability of the two methods to detect outliers as well as to discriminate between true objects and outliers. For this reason, a rather simple seabed is used with the addition of some bumps representing rocks as well as a large number of outliers of various magnitudes.

Fig. 2.1 shows two different models of simple seabed, flat and rising. The x-axis of the plot is the direction of travel of the ship, the y-axis marks the individual beams, and the z-axis represents the depth in meters. For the purpose of illustration, only 35 beams are shown in the figure as it makes it easier to observe the grid and individual beams. In the synthetic data used for testing the methods, 127 beams are used representing one ping of an actual MBES with 127 beams.

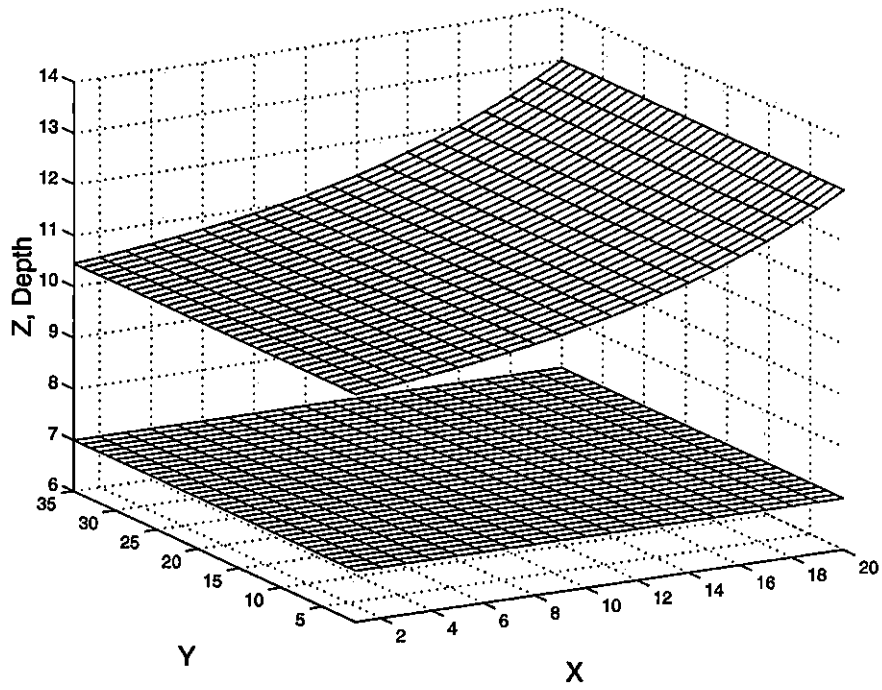

Figure 2.1. *Two different models of seabed.*

To a flat model of seabed, a few bumps representing rocks or other objects on the seabed are added. Impulsive noise is then added to the synthetic data using uniformly distributed random numbers. Fig. 2.2 shows a segment of the seabed including a bump representing a rock. In Fig. 2.3 the same segment of data is shown when impulsive noise is added. The rest of the synthetic data look similar to the data shown in Fig. 2.3.

2.4.2 Robust estimator

The robust estimator iteratively generates surface patches considering only those samples with residual values less than the median residual value of all samples within the specified window. The samples which do not satisfy this criterion are given zero weight and are considered as potential outliers. Fig. 2.4(a) shows a segment of the original synthetic data, which is the same as that shown in Fig. 2.3. The inverse-sensitivity parameter α and the

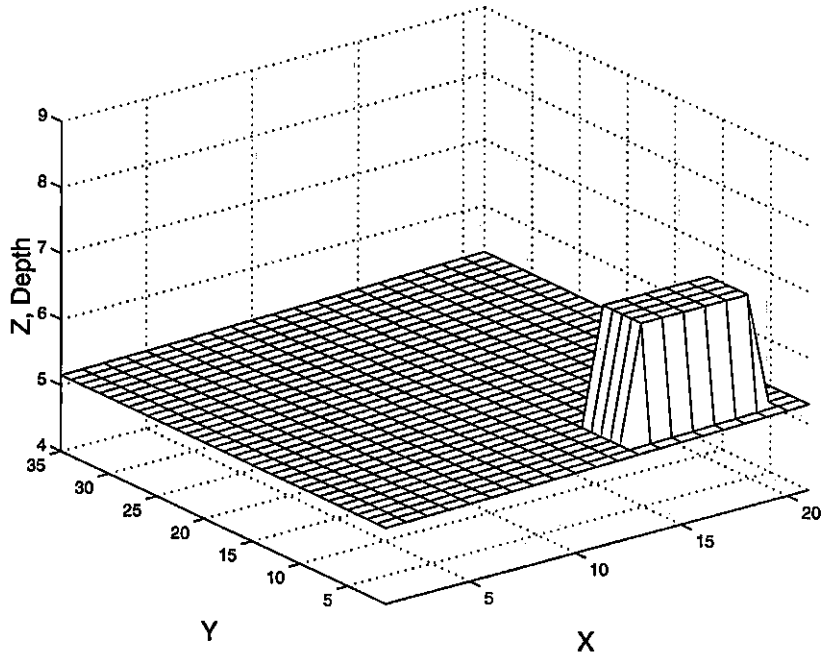

Figure 2.2. *Segment of original synthetic data with true object.*

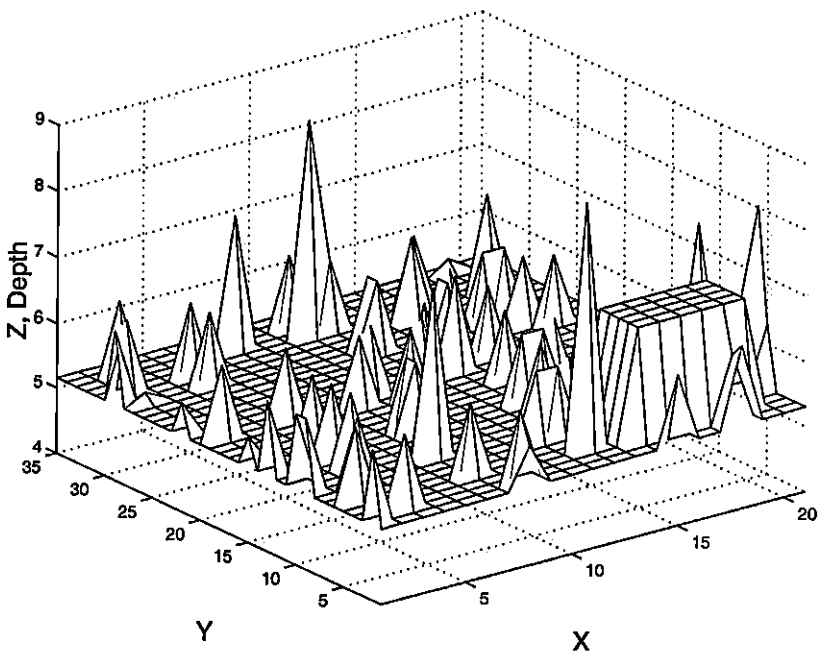

Figure 2.3. *Segment of original synthetic data with true object and outliers.*

window size $L \times L$ were chosen as 4 and 7×7, respectively.

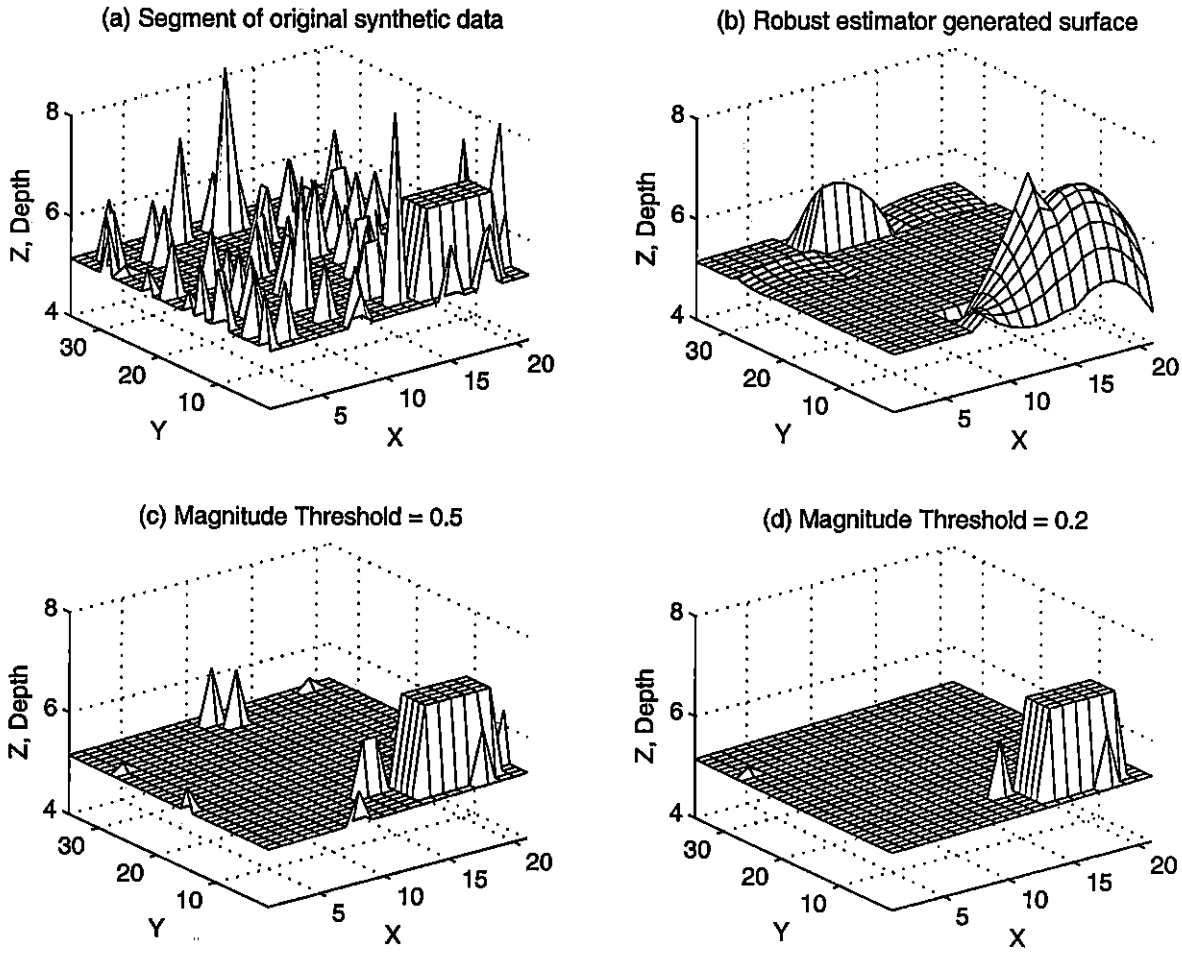

Figure 2.4. *(a) segment of original synthetic data, (b) Robust estimator generated surface, (c, d) Seabed obtained using the robust estimator method with a window of* [7 7] *and various values of magnitude threshold.*

Fig. 2.4(b) shows the surface generated by the robust estimator. As discussed earlier in Sec. 2.3, the residual values between the data points on this generated surface and the data points on the original surface provide the information to compute the weights assigned to each data point according to Eqs. 2.3 and 2.4. In Fig. 2.4(c), a magnitude threshold of 0.5 is used and the resulting seabed is shown. A magnitude threshold of 0.5 detects as the outliers all those data points that have absolute magnitude difference of 0.5 or more

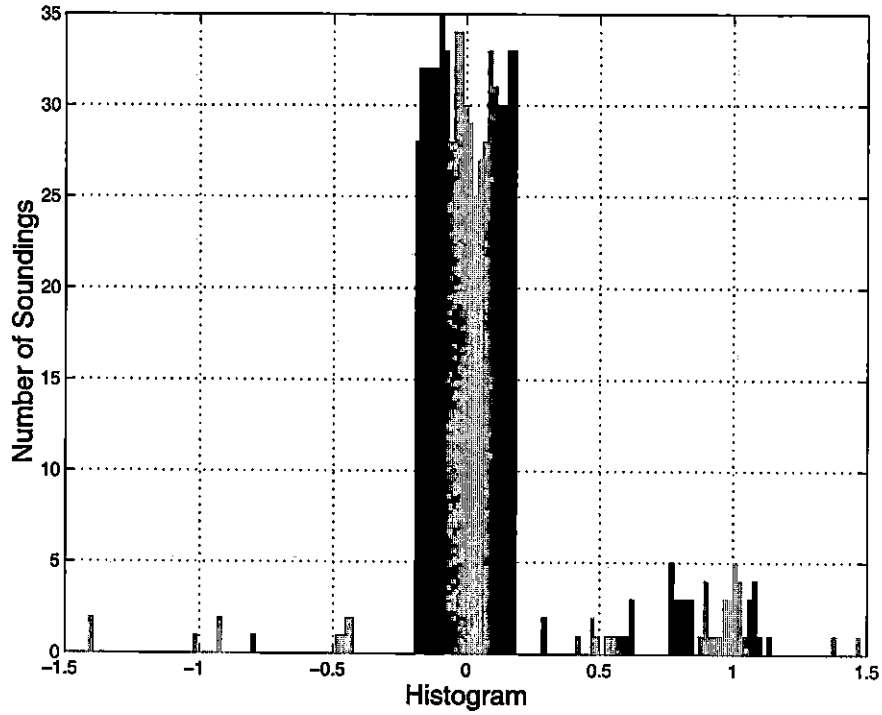

Figure 2.5. *Histogram of the error values obtained using the robust estimator method.*

than the corresponding data points on the robust estimator generated surface. Similarly, Fig. 2.4(d) shows the resulting seabed after using a magnitude threshold of 0.2. The detected outlier value was replaced by the corresponding data point value from the robust estimator generated. In Fig. 2.5, a histogram of the error values of the detected soundings (potential outliers having zero weight) is generated. The large rectangular shaped area near the zero on the horizontal axis represents the number of data points with error values less than ± 0.25. It shows that the magnitude differences between data points on the original surface and corresponding data points on the robust estimator generated surface are very small for most of the data points. The detected outliers are the points corresponding to a magnitude difference of more that ± 0.5.

To evaluate the performance of the robust estimator, various window sizes, values of α, and threshold values have been used. The results are shown in Tables 2.1 and 2.2.

Table 2.1. *Results using robust estimator with* $\alpha = 4$ *and with synthetic data containing* 73 *outliers.*

Window Size	Magnitude Threshold	Detection Rate, %	False Detection Rate, %	Number of Flops
[3 3]	0.5	75.34	17.81	50,524,678
[3 3]	0.3	76.71	30.14	50,524,678
[3 3]	0.1	83.56	52.05	50,524,678
[5 5]	0.5	79.45	21.92	41,461,551
[5 5]	0.3	84.93	34.25	41,461,551
[5 5]	0.1	89.04	64.38	41,461,551
[7 7]	0.5	83.56	12.33	42,160,901
[7 7]	0.3	84.93	19.18	42,160,901
[7 7]	0.1	89.04	42.47	42,160,901
[9 9]	0.5	89.04	24.66	42,520,320
[9 9]	0.3	95.89	45.21	42,520,320
[9 9]	0.1	95.89	75.34	42,520,320

2.4.3 Median filter

Figs. 2.6(a), 2.6(b), and 2.6(c) show the results obtained by applying 2-D median filtering to the data of Fig. 2.3 using window sizes of [3 3], [5 5], and [9 9], respectively. In Fig. 2.6(d), a median filter with window size of [1 3] is used, which corresponds to 1-D median filtering to each individual beam with a window length of 3. With this filter length, spikes of length of at least two samples are not detected as outliers. Fig. 2.6(d) shows the spikes which were present on a particular beam for at least two consecutive pings.

For the detection of outliers, the error values between the original data and the output of the median filter were computed. A histogram of these error values is presented in Fig. 2.7.

Table 2.2. *Results using robust estimator with window size* [7 7] *and with synthetic data containing* 73 *outliers.*

Alpha	Magnitude Threshold	Detection Rate, %	False Detection Rate, %	Number of Flops
1	0.5	78.08	34.25	45,173,419
1	0.3	84.93	58.90	45,173,419
1	0.1	86.30	80.82	45,173,419
2	0.5	82.19	13.70	42,817,343
2	0.3	86.30	24.66	42,817,343
2	0.1	87.67	42.47	42,817,343
4	0.5	83.56	12.33	42,160,901
4	0.3	86.30	26.03	42,160,901
4	0.1	89.04	42.47	42,160,901
8	0.5	84.93	10.96	42,520,320
8	0.3	87.67	28.77	42,520,320
8	0.1	90.41	43.84	42,520,320

There is a good separation of outliers corresponding to the clusters above 0.5 and below 0.5. For the elimination of the outliers, a magnitude threshold is selected. Based on the histogram of Fig. 2.7, a threshold of 0.5 is a good choice and the elimination of outliers resulted in the seabed segment shown in Fig. 2.8. By comparing the processed data with the original synthetic data in Fig. 2.1 it can be seen that most of the outliers have been removed while the true objects i.e. rocks etc. have been retained.

To observe the effect of the window size on the detection of outliers, various window sizes were used and the results obtained are presented in Table 2.3. These results indicate that median filtering requires less computations but is sensitive to the window size. As can be seen from Table 2.3, when the window size is increased, the false detection rate is

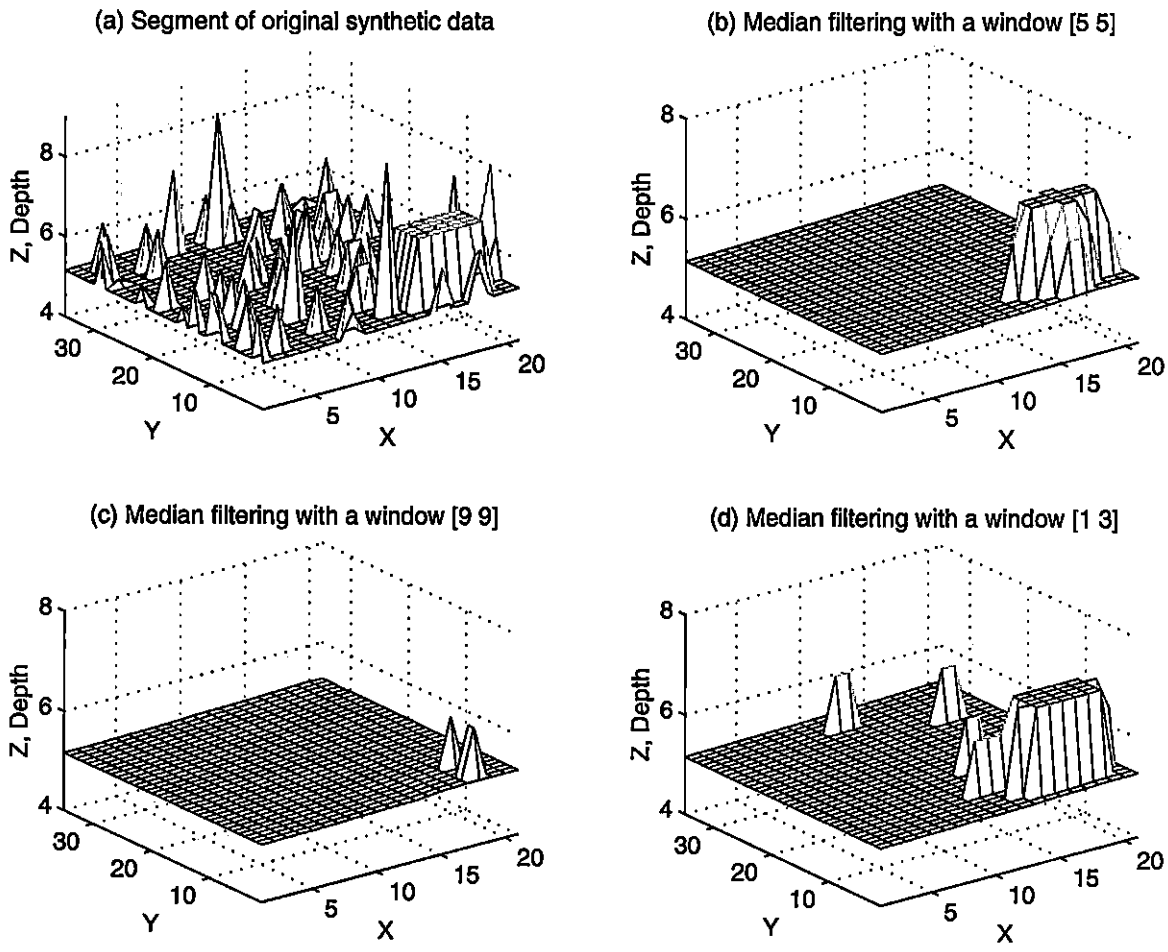

Figure 2.6. *Output of the median filter.*

increased.

The results in Tables 2.1 and 2.2 indicate that the robust estimator requires significantly more computation than the median filter (see Table 3.2) but tends to be slightly less sensitive to the window size.

2.5 Discussion of the Results

From the results, both methods, i.e., the robust estimator as well as the median filtering were able to detect most of the outliers. The median filtering was more sensitive to the window

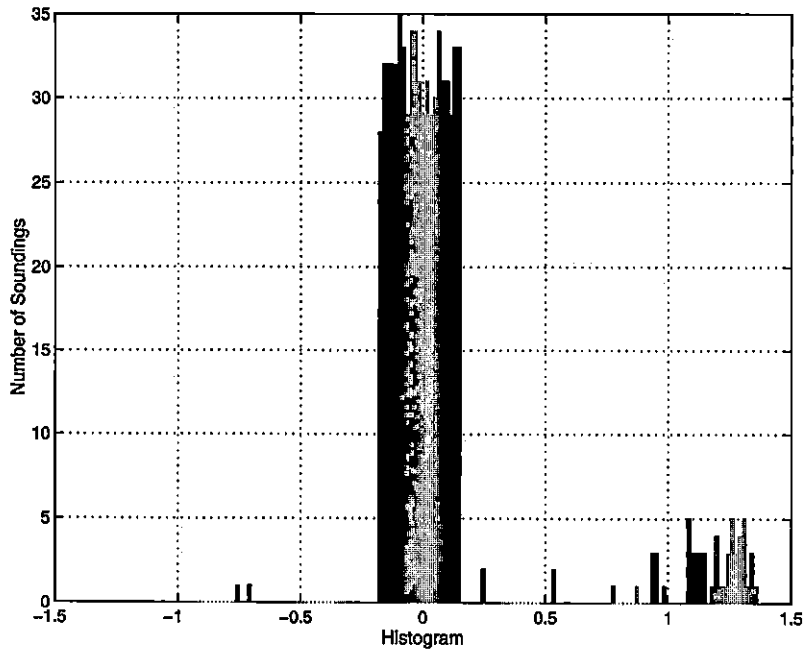

Figure 2.7. *Histogram of the error values obtained using the median filter method.*

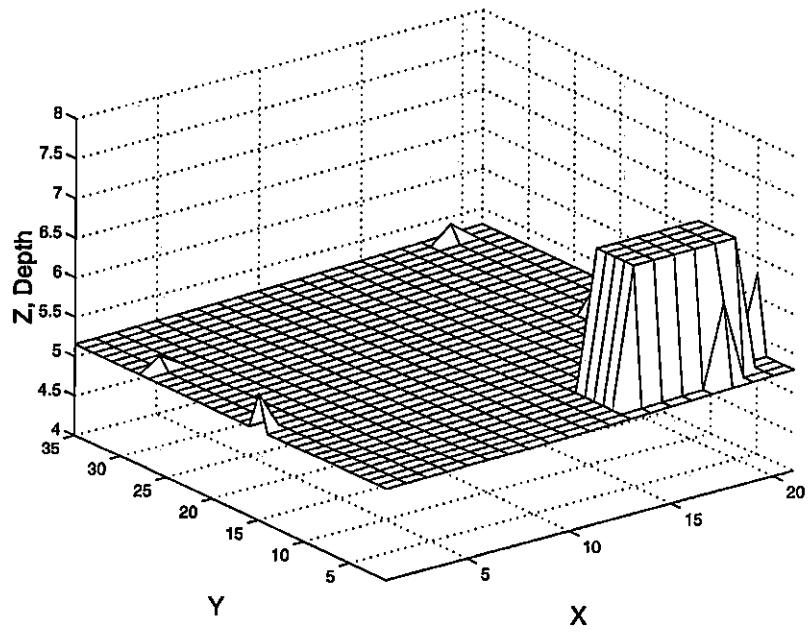

Figure 2.8. *Final result after median filtering with a window of* $[3\ 3]$ *and elimination of outliers using a threshold* $= 0.5$.

Table 2.3. *Results using median filter with synthetic data containing 73 outliers.*

Window Size	Threshold	Detection Rate, %	False Detection Rate, %	Number of Flops
[3 3]	0.5	89.04	13.70	20,225
[3 3]	0.3	91.78	13.70	20,225
[3 3]	0.1	94.52	13.70	20,225
[5 5]	0.5	91.78	31.51	20,392
[5 5]	0.3	94.52	31.51	20,392
[5 5]	0.1	97.26	31.51	20,392
[7 7]	0.5	93.15	47.95	20,536
[7 7]	0.3	95.89	47.95	20,536
[7 7]	0.1	98.63	47.95	20,536
[9 9]	0.5	94.52	79.45	20,801
[9 9]	0.3	97.26	79.45	20,801
[9 9]	0.1	100.00	79.45	20,801

size than the robust estimator and an inappropriate selection of window size resulted in erroneous outlier detection or elimination of data points corresponding to true objects on the seabed. As shown in Fig. 2.6 with a window size [9 9] the true object on the seabed model is completely eliminated. However, with an appropriate window size, the median filter was able to detect nearly all the outliers while preserving the data points representing true objects on the seabed.

Both median filtering and robust estimator methods have been compared according to the number of flops taken by each method for processing the same synthetic data set. In general, the number of flops required by the median filtering method is less than 0.1% of that required by robust estimator method which indicates that median filtering is signifi-

cantly faster. The number of flops depended on the number of iterations used for the robust estimator method, as may be expected.

In both outlier detection methods, as the detection rate increases, a significant increase in false detection rate was observed. For example, in the case of the robust estimator (Table 2.1), at a detection rate of 95.9%, a false detection rate of 75.34% is observed. In the case of median filtering (Table 2.3), a detection ratio of 100% corresponds to a false detection rate of 79.45%. False detection rates of these magnitudes are unacceptable as they imply that perfectly valid data points data are identified as outliers.

As median filtering is much faster than robust estimation, it will be useful to use median filtering for automatic detection process. To reduce the false detection rate, improvements should be made to the median filtering method. Some suggestions for improvement are discussed in the following section.

2.6 Suggestions for Improvements

Two improvements are required for the median filtering method to make it useful for reliable outlier detection. First an approach for finding the appropriate window sizes is needed and second a method is needed that can reduce false detection rate. As we have used only synthetic data so far, the appropriate window size can be selected as 3×3 which is the smallest window size for true median filter. Selection of the smallest window size avoids the problem of losing small data groups representing 3 objects on the seabed. However, to increase the detection rate, a larger window size is required, so that outliers lying close to each other can be detected. The appropriate window size should be chosen based on the information about the size of the smallest object observable on the seabed and this is the approach that will be used in the following chapters.

To reduce the false detection rate, the effect of median filtering on the synthetic data was carefully studied. The location of each valid data point that was falsely detected as an outlier was investigated and it was found that most of these such data points were lying on

the boundaries of data groups representing objects on the seafloor. It means that although median filtering was able to detect outliers and retain the true objects, it falsely detected the data points lying on the boundaries of these objects as outliers. This process can be seen in Fig. 2.6(b) and Fig. 2.6(d) where a data group representing a rock on the seabed is reduced in size as data points on the boundary of this data group are falsely detected as outliers and are eliminated. The false detection rate increases as the window size is increased as more and more data points lying on the boundary of the true objects are being falsely detected as outliers.

The problem of false detections can be greatly reduced by identifying the locations of the falsely detected data points and if these data points are on the boundaries of objects, then it can be safely assumed that they were falsely detected as outliers and can be categorized as valid data points. For example, in the case shown in Fig. 2.8 where median filtering with a window size of [3 3] and a magnitude threshold of 0.5% were used, approximately 98% of the false detections are located on the boundary of the true object.

2.7 Conclusions

Two methods for automatic outlier identification of outliers have been presented and their performance has been evaluated using synthetic data. Both methods were successful in detecting most of the outliers present in the synthetic data. However, it was observed that improper window size leads to a high false detection rate for both median filtering and robust estimator.

The results indicate that median filtering is significantly faster than the robust estimator. However, median filtering is more sensitive to the window size and an inappropriate choice of the window size may lead to a high number of false detections.

As median filtering is much faster than the robust estimation method, the former has been selected as a method to be investigated further. Two modifications to improve the performance of median filter have been suggested. The first modification deals with the proper

choice of the window size based on available information about the multibeam echosounder data. The second modification discussed is to re-investigate the boundaries of the objects on the seabed in order to reduce the high false detection rate to acceptable levels.

Chapter 3

Automatic Outlier Detection Using Median Filtering

3.1 Introduction

In the preceding chapter, median filtering has been shown to be able to eliminate outliers in the regularly spaced data due to its ability to eliminate impulsive noise. However, the multi-beam echosounder data is always irregularly spaced due to the nature of data acquisition process. This fact leads to two important observations:

- The median filtering method employed in Chap. 2 whereby multibeam data points have been assumed to be regularly spaced cannot be used for multibeam data.
- An advantage of regularly spaced data is that access to the data is very fast. As multibeam data is irregularly spaced, a localization method is required that is able to locate the required neighboring data points fast and efficiently.

In this chapter, an automatic outlier detection method using a two-stage median filtering algorithm is presented for multibeam echosounder data. The algorithm is based on the median filtering method described in Chap. 2. In Sec. 3.2 an overview of the automatic outlier detection method is presented. Preprocessing of the multibeam data is discussed in Sec. 3.3. In Sec. 3.4 the first stage of the two-stage median filtering algorithm is described. A localization method used to locate the nearest neighborhood data points in the algorithm

and its efficient implementation are presented in Sec. 3.4.1. The second stage of the two-stage median filtering algorithm is presented in Sec. 3.5. The selection of the parameters used in the algorithm is discussed in Sec. 3.6. Conclusions are drawn in Sec. 3.8.

Due to the irregular spacing of the multibeam data, two new terms are introduced in this chapter to facilitate the description of the two-stage median filtering algorithm. The first term is *neighborhood size*, which denotes the number of nearest data points in the neighborhood of a particular data point. This term defines the nearest neighborhood of a data point and it is the equivalent of the term window size in the median filtering method discussed in the preceding chapter. The second term is *vertical threshold*, which refers to a threshold value used to detect outliers. This term is the same as the magnitude threshold in the median filtering method discussed in the previous chapter.

3.2 Automatic Outlier Detection Method

The proposed automatic outlier detection method uses a two-stage median filtering algorithm as its core unit to identify outliers from the multibeam data. Fig. 3.1 shows the block diagram of the proposed method. The block diagram is divided into three main parts. The first part describes the preprocessing of the raw multibeam data. The next two parts describe the first and the second stage of the two-stage median filtering algorithm respectively. The input data in this block diagram are the raw field multibeam data.

In the preprocessing part, the multibeam data are first geometrically transformed and normalized to facilitate the subsequent division of the data into smaller cells. Then the multibeam data are divided into a number of square cells. Once the division of the data into cells is done, the preprocessing is completed and the first stage of the two-stage median filtering algorithm is employed.

The first stage of the algorithm uses preselected values for neighborhood size and vertical threshold to detect and flag the potential outliers in the data. It can be summarized

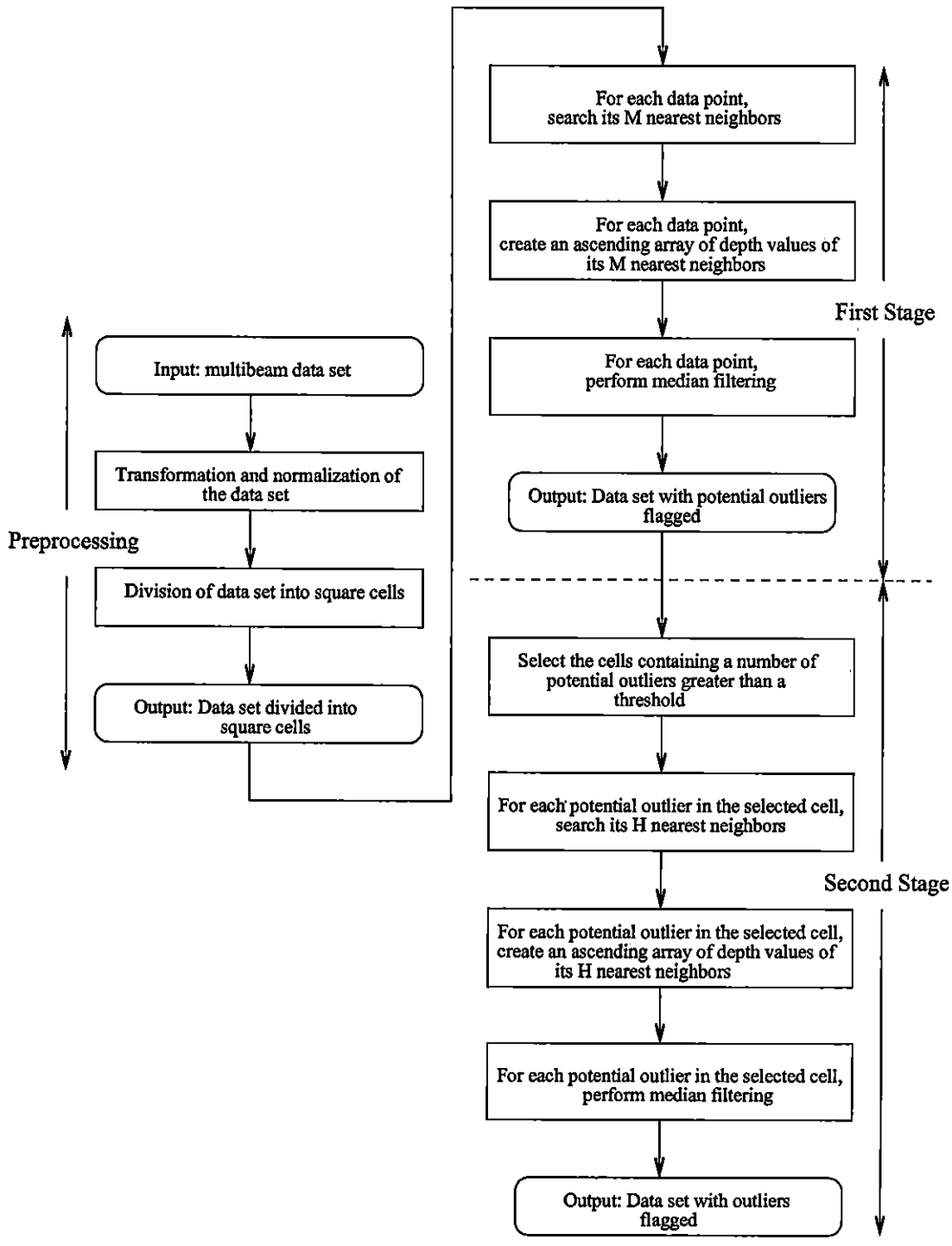

Figure 3.1. *Block diagram of the automatic outlier detection method.*

as follows. Suppose that we have a set of depth estimates $D = \{d_i,\ i = 1,\ 2,\ \ldots,\ N\}$ at the associated locations $c_i = (x_i,\ y_i)$. In order to determine whether d_i is a potential outlier, its M nearest neighbors, where M is the neighborhood size, are located using a localization method. The localization method uses the positioning of each data point within its corresponding square cell and its Euclidean distance from its neighboring data points to generate the list of its M nearest neighbors. All M nearest neighbors are ranked according to their depth estimates and the median of the ranked list is selected as the predicted depth estimate at the location c_i where c_i is the location of d_i. If d_i is different from the prediction by a significant amount (i.e. greater than the vertical threshold), then this indicates that d_i is an isolated value which is inconsistent with its neighbors. Hence, d_i is considered as a potential outlier. The output of this stage is the multibeam data set with potential outliers flagged accordingly.

In the second stage of the two-stage median filtering algorithm, a selected set of cells undergo reprocessing to identify and recategorize data points that have been detected as potential outliers in the first stage of the algorithm. Many of the data points identified as potential outliers in the first stage are located on the boundaries of true objects on the seabed as discussed in Chap. 2. By reprocessing, data points that are true outliers can be distinguished from the data points that are located on the boundaries of true objects on the seabed. Suppose that we have a cell where the number of potential outlier, say Q, exceeds a certain number of outliers say N_{out}. The potential outliers in this cell are reprocessed using a smaller preselected neighborhood size. As observed in Chap. 2, the use of a smaller neighborhood size results in a reduced number of boundary points of true objects being flagged as outliers. Therefore, some of the data points that were detected as potential outliers in the first stage of the algorithm are recategorized as valid data points. This reprocessing significantly reduces the number of falsely detected outliers. The output of the second stage of the algorithm is the multibeam data set with the detected outliers flagged.

In the following sections, preprocessing and the two-stage median filtering algorithm

are described in detail.

3.3 Preprocessing of the Multibeam Data

Real field multibeam data consist of location $c_i = (x_i, y_i)$ and corresponding depth value d_i of soundings obtained from acoustic reflections. The direction of the track of a surveying ship is mainly dependent upon the geological location of the surveyed area and it varies in different survey missions. When the ship is moving along the track in a particular direction, the sounding pattern generated on the floor of the ocean has an orientation that is related to the ship direction. Fig. 3.2 shows the sounding pattern of a particular ship track in one of the EM-3000 survey missions. As shown in the figure, the position of each sounding is represented by x and y coordinates.

As discussed briefly in the preceding section, the multibeam data is divided into a number of square cells. This division of the data into cells is very important as these cells are used in the localization method to efficiently search for a number of nearest neighbors to a particular data point. As shown in Fig 3.2, the multibeam data has an orientation that makes the division of the data into cells difficult. To facilitate the division into cells, a 2-D geometric transformation is applied to the multibeam data. In this way, data sets acquired with different ship track directions are transformed to one fixed orientation of the Cartesian-coordinate system.

Two types of 2-D geometric transformations namely translation and rotation are applied to the multibeam data. In Sec. 3.3.1 the transformations applied to the multibeam data are discussed. The transformed data set is then normalized to further facilitate the implementation of the localization method to search the neighborhood soundings. Then using a procedure described in Sec. 3.3.3, the cell size is calculated. The normalized data set is then divided into a number of square cells.

3.3.1 2-D geometric transformations

Two types of 2-D geometric transformations are applied to the multibeam data. First the data are translated to the origin of the x-y coordinate system and then are rotated about the origin by an angle [21]. To illustrate this, we refer to Fig 3.2 again. For the purpose of illustration, only eight swaths of multibeam echosounder soundings projected along a single track are shown in the figure. Each swath consist of 9 beams which are numbered from beam 0 to beam 8. In order to determine the direction of the ship track, a straight line is fitted through all soundings of a particular beam number in the least-squares sense (beam 4 was chosen in this figure). The slope of the least-squares line is then used to determine the angle θ as shown in the Fig. 3.2. With the angle θ known, we can define a new Cartesian-coordinate system, say the x'-y' system shown in Fig. 3.3.

The data shown in Fig. 3.3 are first translated to the origin of the x-y coordinate system.

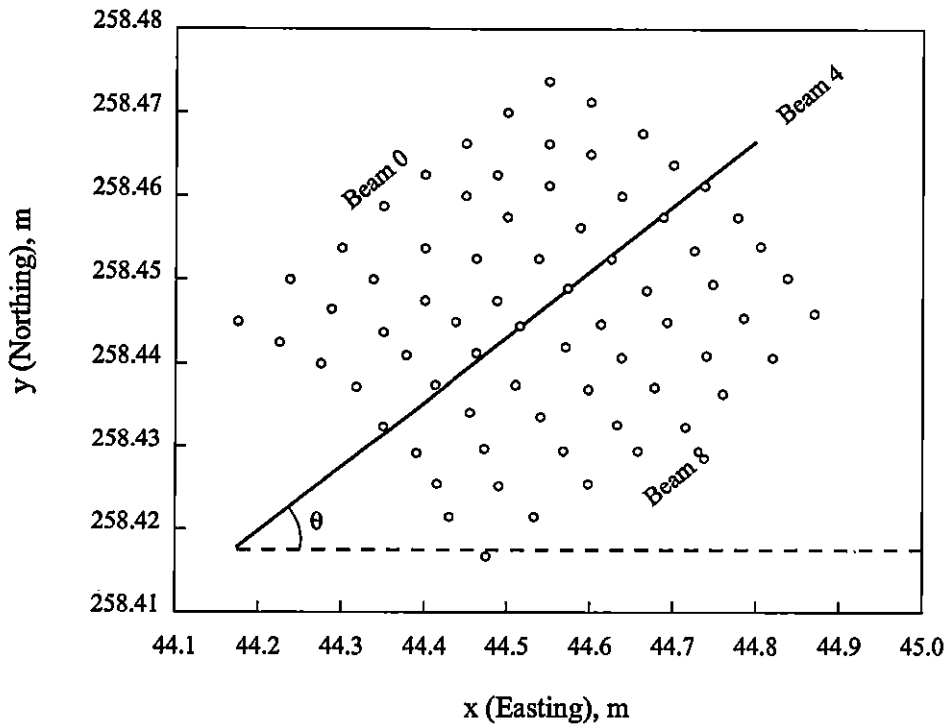

Figure 3.2. *A segment of the multibeam echosounder data.*

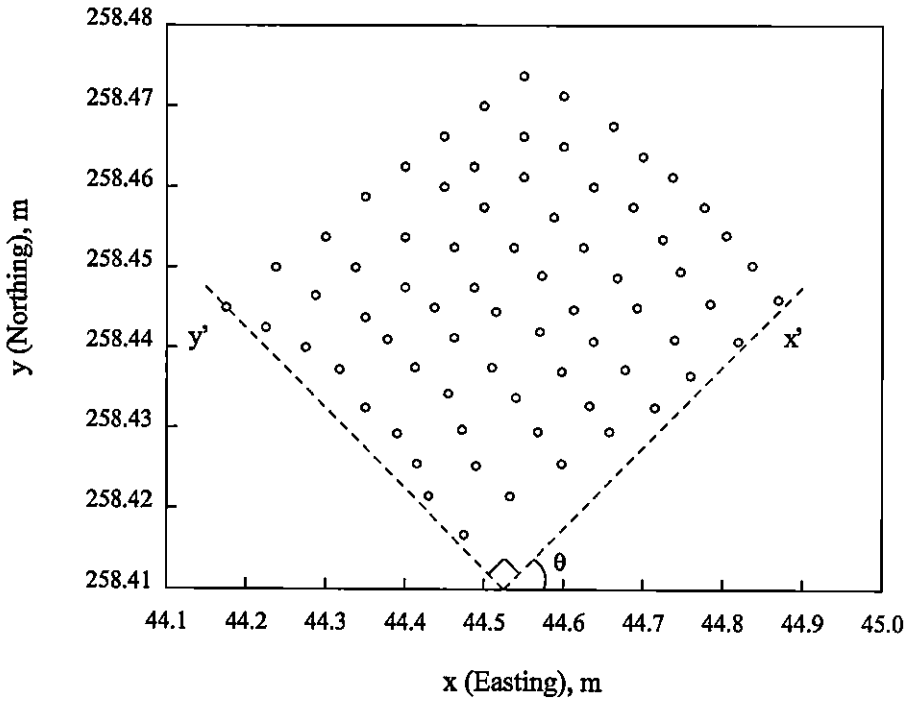

Figure 3.3. *Translation and rotation of the x-y coordinate system to form the x'-y' coordinate system.*

The translation matrix used for this transformation is [22]

$$[T(T_x,\, T_y)] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -T_x & -T_y & 1 \end{bmatrix} \tag{3.1}$$

where T_x and T_y are the translation distances in the x and y directions, respectively.

After the translation, the new coordinate system x'-y' is rotated clockwise by an angle θ. The rotation matrix used for this translation is [22]

$$[R(\alpha)] = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

where α is angle of rotation measured *counterclockwise*. As in this case, the angle of

rotation θ is clockwise, we have $\alpha = -\theta$. Using these two transformations, a data point (x_i, y_i) in x-y coordinate system is transformed to (x'_i, y'_j) in the new x'-y' coordinate system. The 2-D geometric transformation equation is [22]

$$[p'] = [p][T(T_x, T_y)][R(\alpha)] \tag{3.3}$$

where

$$[p'] = [x' \ y' \ 1] \tag{3.4}$$

represents the transformed sounding location expressed in the homogeneous-coordinate form [22], and

$$[p] = [x \ y \ 1] \tag{3.5}$$

represents the original sounding location expressed in the same form.

3.3.2 Transformation and normalization of the multibeam data

Fig 3.3 shows the multibeam data obtained in one of the EM-3000 survey. Using the 2-D transformation given in the Eq. (3.3), a data point (x_i, y_i) in x-y coordinate system is transformed to (x'_i, y'_j) in the new x'-y' coordinate system. Hence, the data point

$$\begin{aligned} x'_i &= (x_i - T_x)\cos\theta + (y_i - T_y)\sin\theta \\ y'_i &= (y_i - T_x)\cos\theta - (x_i - T_y)\sin\theta \end{aligned} \tag{3.6}$$

where θ is the angle by which the x-y coordinate system is rotated to form the x'-y' coordinate system and $T_x = 48.52$ and $T_y = 258.41$ for the data shown in Fig. 3.3. The data points are then normalized to the values $(\hat{x}_i, \hat{y}_i)$ according to [23]

$$\begin{aligned} \hat{x}_i &= (x'_i - x'_{min})/(\text{maxmin}) \\ \hat{y}_i &= (y'_i - y'_{min})/(\text{maxmin}) \end{aligned} \tag{3.7}$$

where

$$\text{maxmin} = \text{maximum}(x'_{max} - x'_{min},\ y'_{max} - y'_{min})$$

and

$$x'_{max} = \text{maximum}\{x'_i\}$$
$$x'_{min} = \text{minimum}\{x'_i\}$$
$$y'_{max} = \text{maximum}\{y'_i\}$$
$$y'_{min} = \text{minimum}\{y'_i\}$$

This ensures that the values of $\hat{x}$ and $\hat{y}$ lie between 0 and 1. The normalization eliminates different measurement scales used in x and y directions which makes it easy to manage large data sets. Fig. 3.4 shows the multibeam data after the transformation and normalization.

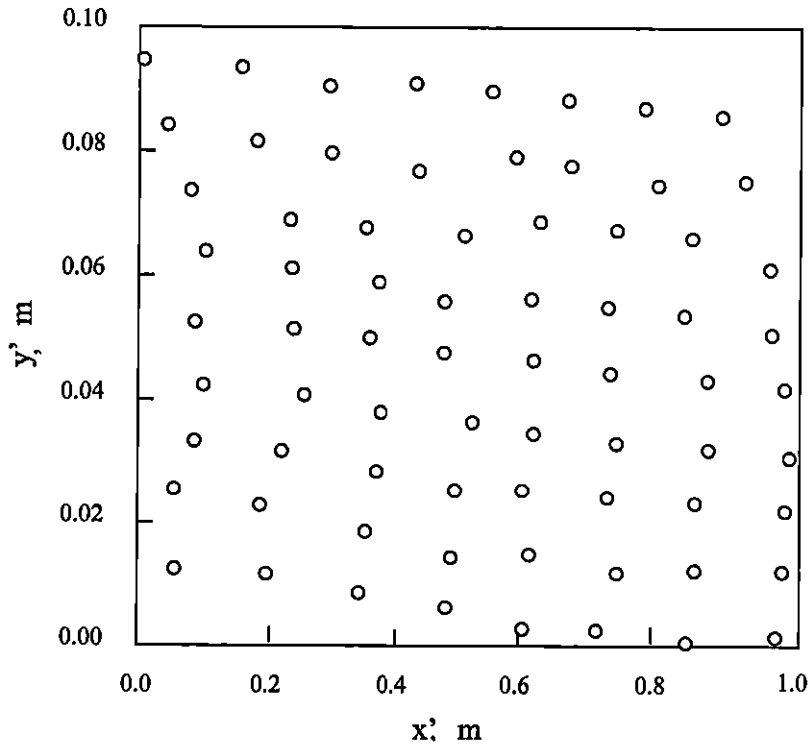

Figure 3.4. *Multibeam sounding data after transformation and normalization.*

3.3.3 Division of the data into cells

The two-stage algorithm given in Sec. 3.2 involves a lot of searching to find a fixed number of nearest sounding locations to a particular sounding location. In order to make the search process more efficient, the multibeam data set is divided into smaller cells or blocks. Thus, to search for data points in the neighborhood of a particular data point located in a specific cell it is sufficient to search for data points in that specific cell and, if need be, in adjacent cells instead of searching the whole data set. This division of data set into smaller cells significantly reduces the search time. A simple procedure to search for data points in the neighborhood of a particular data point is described in the next section.

The size of the cells has to be appropriately chosen depending on the density of the data points. The greater the number of data points in a cell, the greater is the number of data points which have to be searched in order to determine the nearest neighbors. On the other hand if the cell size is small, there will not be enough data points in the cell. So the size of the cell is chosen in such a way that each cell contains an appropriate number of data points in connection with the neighborhood size M. In order to chose a suitable cell size a procedure based on the standard Euclidean distance between two data points is employed. In order to automate the computation of the search radius r for a given neighborhood size M, the following procedure is used:

1. Select an arbitrary data point from the profile and search for its M nearest neighbors to form a set of $M+1$ data points.
2. Compute the mean center [24] $m_c = (x_c, y_c)$ from these $M+1$ data as

$$x_c = \frac{1}{M+1}\sum_{i=1}^{M+1}\hat{x}_i \text{ and } y_c = \frac{1}{M+1}\sum_{i=1}^{M+1}\hat{y}_i$$

   where $(\hat{x}_i,\ \hat{y}_i)$ represents the sounding location of the ith data in the normalized coordinate system.
3. Compute the standard distance [24] l_O as

$$l_O = \sqrt{\frac{1}{M+1}\sum_{i=1}^{M+1} l_i^2}$$

where l_i is the Euclidean distance between ith data and m_c.

4. Repeat Steps 1 to 3 in different areas of the profile in order to compute an average of the standard distance l_O, denoted as $\bar{l}_O$.
5. Compute $r = \lambda \bar{l}_O$ where λ is a scaling constant typically in the range $1 < \lambda < 4$.

Since standard distance is a concise statistical measure of the spatial dispersion of data points, the use of this measure for the computation of r provides a convenient and reliable way to define r. The cell size is then chosen as two times the search radius r. As a circle of radius r centered at a particular location is expected to contain the required number of nearest points, a square cell of length $2r$ centered around the same location can also be expected to contain the required number of nearest points.

Once the sizes of the cells have been finalized, the data set is divided into a number of square cells. Suppose we have a set of depth values, $D = \{d_i,\ i = 1,\ 2,\ \ldots,\ N\}$ at the associated locations $\hat{c}_i = (\hat{x}_i,\ \hat{y}_i)$ in the normalized coordinate system. The data set is divided into square cells $C_{k,l}$ for $k = 1,\ 2,\ \ldots,\ P$ and $l = 1,\ 2,\ \ldots,\ S$ each of size $2r \times 2r$. The values of P and S are determined depending upon the lengths of data set in the x and y directions. A linked list is then maintained for each cell, the nodes of the linked list being the data points included in the cell. Each node in the list consists of x and y coordinates of the data point it represents and also its depth value d_i. Each node also contains a variable f which is the flag value representing the status of each data point. A value of $f = 1$ means that the particular data point is flagged as a potential outlier and a value $f = 0$ declares the data point as a valid sounding. A separate field in each cell keeps track of the number of data points which are flagged as outliers. Memory is allocated dynamically depending upon the number of data points in the cell.

The first stage of the two-stage median filtering algorithm can now be presented.

3.4 The Algorithm - First Stage

Suppose we have a set of depth values, $D = \{d_i,\ i = 1,\ 2,\ \ldots,\ N\}$ at the associated sounding locations $\hat{c}_i = (\hat{x}_i,\ \hat{y}_i)$ in the normalized coordinate system. The data set is divided into square cells $C_{k,l}$ for $k = 1,\ 2,\ \ldots,\ P$ and $l = 1,\ 2,\ \ldots,\ S$. A potential outlier counter $Q_{k,l}$ is associated with each cell that keeps track of the number of data points that are flagged as potential outliers in that cell.
Given M_1, the neighborhood size, and T, the vertical threshold, do:

F1: Initialize the counter for potential outliers for the whole data set, i.e. $U = 0$.

F2: Initialize the counters for potential outliers for each cell, i.e. $Q_{k,l} = 0$ for $k = 1,\ 2,\ \ldots,\ P$ and $l = 1,\ 2,\ \ldots,\ S$.

F3: For each sounding location $\hat{c}_i$, $i = 1, 2, \ldots, N$ in the data set, do:

(a) Determine the corresponding square cell $C_{k,l}$ for the sounding location $\hat{c}_i$, so $\hat{c}_i \in C_{k,l}$.

(b) Search for the M_1 sounding locations s_j, $j = 1,\ 2,\ \ldots,\ M_1$ closest to $\hat{c}_i$ where $\hat{c}_i$ is the sounding location of d_i.

(c) Sort $Z = \{d_j,\ j = 1,\ 2,\ \ldots,\ M_1\}$ where d_j is the depth value at s_j into ascending order to form an ordered array Y, i.e. $Y = \{d_1, d_2, \ldots, d_{M_1}\}$.

(d) Assign $p = \text{med}(Y)$, where $\text{med}(Y)$ is the median value of the array Y.

(e) If $|p - d_i| > T$, then:

i. Flag d_i as a potential outlier.

ii. Record the location $\hat{c}_i$ as $\hat{c}_i^*$.

iii. Increment counter U.

iv. Increment counter $Q_{k,l}$.

F4: Record the number of total potential outliers detected, i.e. U, and the number of potential outliers detected in each cell, $Q_{k,l}$.

Based on the results and discussion of Chap. 2, it is observed that a bigger neighborhood size would lead to a higher detection rate of potential outliers. However, a number of data points declared as potential outliers are located on the boundaries of the true objects on the seabed. The measured depth in the neighborhood of some of these data points can vary significantly due to their locations. This variation is the reason why some of the data points are declared as potential outliers even though they are actually part of true objects. The second stage of the two-stage median filtering algorithm presented in Sec. 3.5 is used to identify and declare such data points as valid data points.

3.4.1 Search for neighboring points

In step F3:(b) a simple yet efficient localization method is used to search for the M nearest data points to a particular data point where M is the neighborhood size. This method selects the nearest points based on the distance between a particular data point and its neighboring data points. As discussed in Sec. 3.3.3, Fig. 3.5 shows the multibeam data set divided into a number of square cells $C_{k,l}$. The procedure used for the localization is based on some discussion for a similar search in [23] and on the following observations:

- The search to find M neighbors close to point p_0 starts by finding the number of data points in cell $C_{k,j}$. If $C_{k,l}$ contains more than M data points, there may not be a need to search for data points located outside $C_{k,l}$. If the number of data points contained in $C_{k,l}$ is less than M, all of the cells adjacent to $C_{k,l}$ are also searched. If the total number of data points included in $C_{k,l}$ and all its adjacent cells is still less than M, cells adjacent to these cells are also searched and so on. However, due to the selection of the cell size $2r$, as discussed in Sec. 3.3.3, the probability of not finding the required number of neighbors to a particular data point in a cell and in the cells immediately adjacent to it, is very small.

- In the case where p_0 is located near the boundary of $C_{k,l}$ and $C_{k,l}$ contains more than M data points, the nearest neighbors of p_0 could be selected among the data points included in $C_{k,l}$ itself although other data points included in cells adjacent to $C_{k,l}$ may be nearer to p_0 as illustrated in Fig. 3.5. This can lead to false detections of nearest neighbors. In order to avoid this problem, data points located in adjacent cells are also included in the search to find the M nearest neighbors of p_0.

Based on these observations the procedure to search for the M nearest data points to a particular data point p_0 located in cell $C_{k,l}$ can be summarized as follows:

Cell $C_{k,l}$ is broken into five parts: *center (ct), top-left (tl), top-right (tr), bottom-left (bl), and bottom-right (br)* as illustrated in Fig. 3.6. The area of the shaded parts can be chosen appropriately. If p_0 is located inside the *top-right* part of the cell $C_{k,l}$, all data points contained in $C_{k,l}$ as well as in its immediately adjacent cells $C_{k-1,l}$, $C_{k-1,l+1}$, and $C_{k,l+1}$ are used. Similarly, if p_0 is located in any of the other three parts of the cell $C_{k,l}$ denoted by the shaded regions in Fig. 3.6, then all points in $C_{k,l}$ and the cells adjacent to the shaded

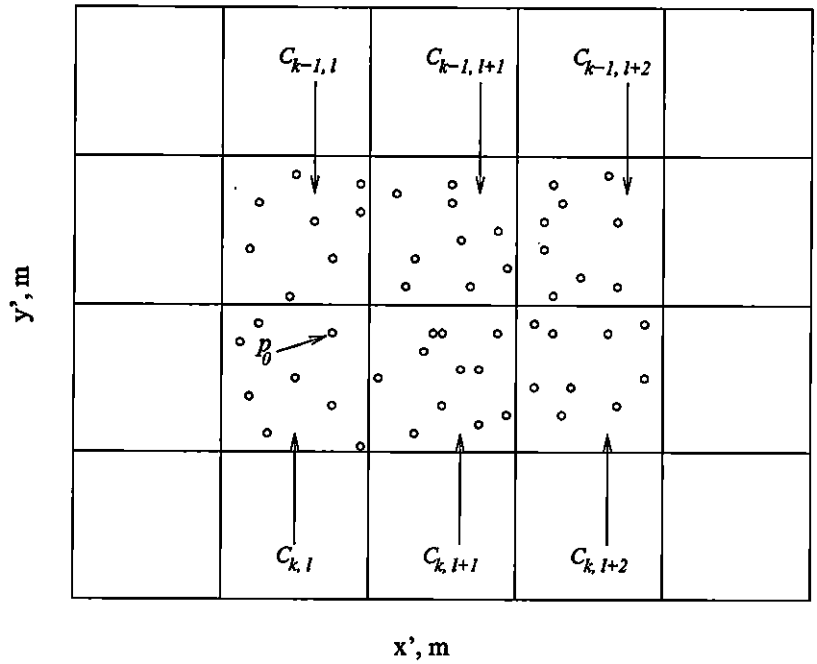

Figure 3.5. *Multibeam data set after division into smaller blocks.*

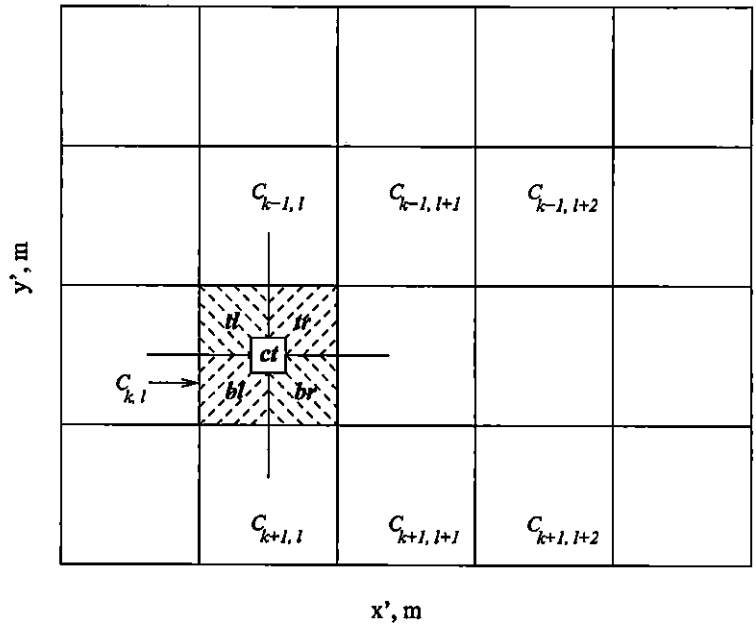

Figure 3.6. *Block $C_{k,l}$ divided into five parts.*

portion are also used. If p_0 is located inside the *center* part of the cell $C_{k,l}$ then no adjacent cells are considered unless $C_{k,l}$ has fewer than M points. In each of these five cases, the distance between p_0 and all the data points used is computed and the M nearest data points are chosen.

The procedure described above works very reliably in most of the cases. However in some cases it can happen that when p_0 is located in the *center* of the cell $C_{k,l}$, some of the data points in adjacent cells are nearer to p_0 than some of the data points selected as its nearest neighbors from within cell $C_{k,l}$ itself. We see in Fig. 3.7 that p_3 is considered as one of the 7 nearest neighbors of p_0 although p_1 and p_2 located in adjacent cells are nearer to it. Thus, it is advisable to search for m data points, where $m > M$, and then select the M points nearest to p_i as its nearest neighbors. Although this will eliminate the problem of false identification of the nearest neighbors to a great extent, for some cases when most of the data points in $C_{k,l}$ are concentrated at the corners, this may still not work. However, this is very unlikely to happen since data points in general are evenly distributed and it is not likely that data points in any cell will be concentrated only at the corners.

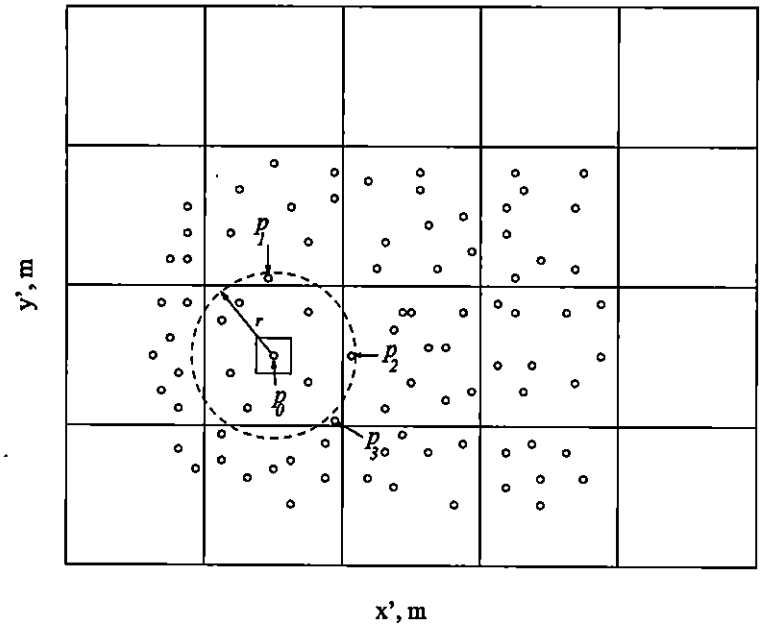

Figure 3.7. *Point p_0 nearer to points p_1 and p_2 than point p_3.*

3.5 The Algorithm - Second Stage

In the second stage of the algorithm, we have the same data set that is used in the first stage of the algorithm. However, the locations of the data points which have been identified as potential outliers are flagged.

Given M_2, the neighborhood size, and T, the vertical threshold, do:

S1: Initialize the counter for the rejected potential outliers, i.e. $U' = 0$.

S2: For each cell $C_{k,l}$ for $k = 1, 2, \ldots, P$ and $l = 1, 2, \ldots, S$, do:

A1: If $Q_{k,l} > N_{out}$, where N_{out} is the threshold for number of outliers in a cell, then for each sounding location $\hat{c}_i^* \in C_{k,l}$, do:

(a) Search for the H sounding locations s_j, $j = 1, 2, \ldots, H$ closest to $\hat{c}_i^*$ where $\hat{c}_i^*$ is the sounding location of d_i. $H = M_2 + L$ where $L \geq 10$.

(b) Sort $Z = \{d_j, j = 1, 2, \ldots, H\}$ where d_j is the depth value at s_j into ascending order to form an ordered array Y, i.e. $Y = \{d_1, d_2, \ldots, d_H\}$.

(c) Eliminate those depth values from the ordered array Y that have been flagged as outliers and make a new ordered array of X, i.e. form $X = \{d_1, d_2, \ldots, d_{M_2}\}$

(d) Assign $p = \text{med}(X)$ where $\text{med}(X)$ is the median value of the ordered array X.

(e) If $|p - d_i| \leq T$, then:

i. Clear flag on d_i.

ii. Record the location $\hat{c}_i^*$ as $\hat{c}_i$.

iii. Increment counter U'.

iv. Decrement counter $Q_{k,l}$.

S3 Record the number of total outliers detected, i.e. $U - U'$, and the number of outliers detected in each cell, $Q_{k,l}$.

In the second stage of the algorithm, some cells which contain more than N_{out} potential outliers are selected. The potential outliers in these cells are then reprocessed using a smaller neighborhood size. In Chap. 2, a smaller window size was shown to lead to a smaller number of points on the boundaries of the true objects being detected as outliers. This motivates the use of a smaller neighborhood in the second stage of the algorithm to identify data points that are located on the boundaries of the true objects which have been flagged as potential outliers in the first stage. In this way, the false detection rate of the two-stage median filtering algorithm is decreased significantly.

One important aspect of step S2:A1:(c) in the second stage is that for a particular data point, data points that are flagged as potential outliers in the first stage of the algorithm are removed from the list of closest neighbors for this data point. Therefore, if a data point is identified as a potential outlier in the first stage and it is not located on the boundary of a true object, a smaller neighborhood size in the second stage will not change its status to a valid data point and it will remain identified as an outlier. For example, if a neighborhood size of 13 is used in the first stage of the algorithm, then groups of data points of 7 or less, all of them with similar depth but different than that of their neighbors, will be identified as

potential outliers. In the second stage, a smaller neighborhood size of 5 or 7 will be chosen to reprocess the selected cells. As most of the data points in these groups are identified as potential outliers, they will be removed from the ordered array of nearest neighbors and hence the data points belonging to these small groups will remain correctly identified as outliers. This aspect of step S2:A1:(c) ensures that the high true detection rate achieved using a bigger neighborhood size remains unaffected in the second stage of the algorithm.

There are several parameters in both stages of the algorithm, the vertical threshold T, neighborhood sizes M_1, M_2 and N_{out} that need to be selected. Guidelines for selecting these parameters are described in the following section.

3.6 Parameter Selection

The choice of neighborhood sizes depends on the resolution of the multibeam echosounder system used for the data acquisition. The resolution of a particular type of multibeam system gives a suitable measure in selecting the neighborhood size for both the stages of the algorithm. This thesis deals with multibeam data acquired from shallow water bathymetric surveys. If the resolution of the multibeam system is given as 7, for example, it implies that the system can ensonify an object which is detected by at least 7 beams. Thus, data groups of 7 or more should be treated as true objects on the seabed. In this case, a neighborhood size of 13 should be used in the first stage of the algorithm to obtain a high outlier detection rate. On the other hand many valid data points lying on the boundaries of the objects on the seabed will be detected as potential outliers with this neighborhood size. In the second stage of the algorithm, a neighborhood size of 5 or 7 will recategorize some of the data points which are lying on the boundaries of the objects on the seabed and have been identified as potential outliers in the first stage of the algorithm. This helps in reducing the number of data points that were detected as potential outliers in the first stage of the algorithm.

Another important parameter in the algorithm is the vertical threshold, T. In shallow waters, a vertical threshold of 1 to 3 meters, depending upon the depth range of the multi-

beam data, has been found to be very effective.

In the second stage of the algorithm, N_{out} is used as a threshold to select a particular cell for reprocessing. This threshold represents the number of potential outliers found in the cell by the first stage of the algorithm. The choice of N_{out} can be made based on the accuracy estimates of the multibeam echosounder system. For example, given that approximately 10% of the EM-3000 multibeam echosounder data are erroneous, N_{out} can be chosen as an integer approximation of $N \times 0.1$ where N is the average data points in a cell.

The selections of neighborhood size and vertical threshold are further discussed in Chap. 4 where the performance of the algorithm is evaluated using field multibeam data.

3.7 Conclusions

An automatic outlier detection method that uses a two-stage median filtering algorithm to detect outliers in the multibeam data has been proposed and described in detail.

The raw field multibeam data are first preprocessed to facilitate the division of the data into square cells. Preprocessing of the data involves geometrical transformation, normalization, and eventually, the division of the data set into square cells of predetermined size.

After preprocessing the first stage of the two-stage median filtering algorithm is employed which uses preselected values of neighborhood size and vertical threshold to identify potential outliers. This stage is used to achieve a high detection rate. The second stage of the algorithm uses another preselected value of the neighborhood size to reprocess some of the data points that have been identified as potential outliers and to identify those data points that were flagged as potential outliers in the first stage but are located at the boundaries of true objects.

The selection of parameters used in the two stages of the algorithm has been discussed in detail.

Chapter 4

Validation and Presentation of Results

4.1 Introduction

Multibeam field data supplied by IOS is used in this chapter to evaluate the performance of the automatic outlier detection method presented in Chap. 3. Results obtained for two multibeam data sets are presented. Two different approaches are used to evaluate and validate the results. The first approach uses direct comparison of the locations of the outliers and the second approach uses visualization of bathymetric images generated from the data sets. The results indicate that the proposed method is very efficient, fast, and accurate.

4.2 Application of Two-Stage Median Filtering Algorithm

The data used to test the two-stage median filtering algorithm was collected by a Simrad EM3000 multibeam echosounder system. This is a 300 kHz multibeam echosounder capable of mapping the seafloor at depths between 3 and 70 meters below the transducer. This system employs 127 beams to measure a 120° swath transverse to the vessel heading with an overlapping angular resolution of 1.5° × 1.5° [25]. The data used in this chapter was provided by the IOS and comes from a shallow water area off Newfoundland. The first data set consists of eight track lines: six East-West and two North-South lines. The second data set contains 7 track lines: five East-West and two North-South lines. The first data set is named data set D1 and the second data set is called data set D2. The depth variation

in both data sets is from 30 meters to 60 meters approximately. Data set D1 consist of 586, 410 data points and data set D2 contains 641, 421 data points. Figs. 4.1 and 4.2 show the bathymetric images of raw data sets D1 and D2, respectively. These images show a section of the actual seafloor as ensonified by the EM3000 multibeam echosounder.

The automatic outlier detection method presented in the Chap. 3 was used to identify the outliers in the two multibeam data sets. The results obtained are compared with processed data sets also provided by the IOS. These data sets were produced by experienced operators at the IOS by manually eliminating all the outliers from the multibeam data and are used here to evaluate the detection rates and the false detection rates of the proposed two-stage median filtering algorithm. The criteria used are

$$\text{Detection rate} = \frac{\text{number of true detected outliers}}{\text{total number of actual outliers}}$$

and

$$\text{False detection rate} = \frac{\text{number of false detected outliers}}{\text{total number of actual outliers}}$$

where the "actual outliers" are the data points that are declared as outliers in the manually cleaned data sets and the "true detected outliers" are those of the data points that are declared as outliers by the proposed automatic outlier detection method which coincide with the actual outliers. Therefore, the true detection rate is the measure of the accuracy of the automatic outlier detection method. A false detected outlier is a data point that is declared as an outlier using the proposed automatic outlier detection method but this data point is considered a valid data point in the manually cleaned data set. The false detection rate is the measure of error rate in the automatic outlier detection method.

Visualization of bathymetric images generated from the raw data sets and the clean data sets is also used for validation of the results obtained using the proposed automatic outlier detection method.

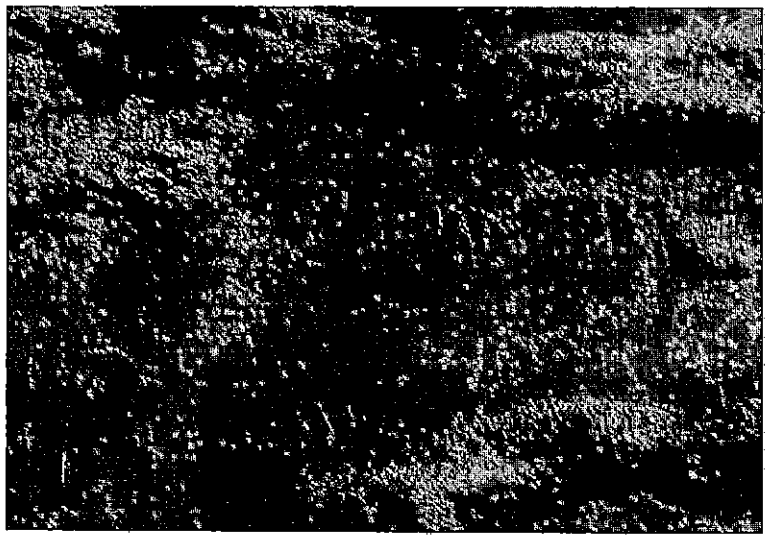

Figure 4.1. *Bathymetric image of raw multibeam data set D1.*

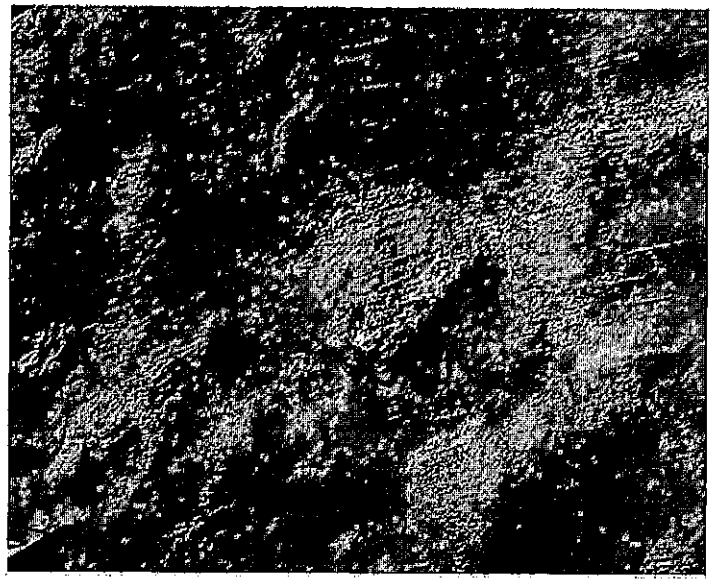

Figure 4.2. *Bathymetric image of raw multibeam data set D2.*

4.3 Parameter Selection

The selection of neighborhood sizes to be used in the automatic outlier detection method to identify outliers from a multibeam data set is based on the resolution of the multibeam echosounder system used to acquire that data. The resolution of this echosounder system can be interpreted as the smallest object on the seafloor that it can ensonify. Ensonification of an object means that the object is detected or illuminated by the echosounder beams. For the object to be ensonified and validated by an operator using graphical methods, it has to be detected by a certain minimum number of beams. For data sets D1 and D2, an object that is illuminated by 7 beams is considered as a valid object. As each beam corresponds to a particular data point, any groups of data points of 7 or more with similar depth values should be considered as a true object on the seafloor. Groups of data points smaller than 7 should be considered as outliers. This measure provides a suitable way to select the neighborhood size for the first stage of the two-stage median filtering algorithm. A neighborhood size of 13 will in most cases detect groups of data points of less than 7 as outliers and it will preserve groups of 7 or more data points as valid data points representing objects on the seafloor. Therefore, in the first stage of the two-stage median filtering algorithm, the neighborhood size M_1 is selected to be 13. In the second stage of the two-stage median filtering algorithm, 7 is chosen as the neighborhood size M_2.

The choice of vertical threshold depends on the data density and the depth range of the multibeam data set. In the data sets used in this chapter, the data density is very high and the depth range variation is 30 to 60 meters. High data density and low depth range, as in this case, mean that the depth of the data points is gradually increasing and decreasing representing a smooth seafloor representation. A vertical threshold of 1 m can be chosen to detect a data point that has a depth value that does not conform to its nearest neighbors.

In the second stage of the algorithm, parameter N_{out} needs to be selected. As described in Sec. 3.6 N_{out} is selected as 6 based on the accuracy estimates of EM3000.

4.4 Results

In this section, results obtained by using the proposed automatic outlier detection method to detect outliers from the field multibeam data set D1 and D2 are presented. The results are evaluated using the criteria described in Sec. 4.1.

4.4.1 Data set D1

Data set D1 contains 586,410 data points. The data set was first preprocessed using the procedure described in Sec. 3.3. Using Eq. (3.6), a data point (x_i, y_i) in the x-y coordinate system is transformed to (x_i', y_i') in the new coordinate system, the x'-y' system. For data set D1, we have

$$\begin{aligned} x_i' &= (x_i + 52.144)\cos 36 + (y_i + 64.520)\sin 36 \\ y_i' &= (y_i + 52.144)\cos 36 - (x_i + 64.520)\sin 36 \end{aligned} \tag{4.1}$$

The value of r was chosen according to the procedure in Sec. 3.3.3 and the whole data set D1 is divided into 87×129 square cells. The two-stage median filtering algorithm was then used to detect the outliers from data set D1. In the first stage of the algorithm, a neighborhood size of 13 was selected and a vertical threshold of 1 m was chosen as discussed in Sec. 4.2. After the first stage, 50,718 data points were flagged as potential outliers. A neighborhood size of 7 was used in the second stage of the algorithm and 47,141 data points were detected as outliers.

Table 4.1 presents the results for data set D1 using the direct comparison of the locations of the outliers. The first column of the table shows the neighborhood sizes M1 and M2 used in the first and second stage of the automatic outlier detection method respectively. The vertical threshold is the depth value used as a threshold to detect outliers. N_{out} is the cell outlier threshold. CPU time is the time taken (in minutes) by the algorithm to process data set D1. The CPU used in this case was SUN-SOLARIS workstation with 1024MB RAM.

Table 4.1 shows the best results obtained for two sets of neighborhood sizes for data

Table 4.1. *Results for data set D1 using the direct comparison of the locations of outliers detected by the automatic outlier detection method to the locations of outliers detected by the manual methods.*

Neighborhood Sizes (M_1, M_2)	Vertical Thresh (T)	Cell Outlier Thresh (N_{out})	True Det. Rate, %	False Det. Rate, %	CPU Time (Minutes)
[11], [7]	1.0	6	94.37	4.43	12 : 35
[13], [7]	1.0	6	95.56	4.92	12 : 58

set D1. As can be seen, with the proper selection of the parameters the automatic outlier detection method detected over 95% of the true outliers from data set D1 while keeping the false detections to below 5%. Also, the results were obtained in a very short time as shown by the CPU time taken by the automatic method.

4.4.2 Data set D2

Data set D2 contains 641, 421 data points. The data set was first preprocessed using the procedure described in Sec. 3.3. Using Eq. (3.6), a data point (x_i, y_i) in x-y coordinate system is transformed to (x_i', y_i') in the new coordinate system, the x'-y' system. For data set D2, we have

$$\begin{aligned} x_i' &= (x_i - 60.394)\cos 15 + (y_i + 58.825)\sin 15 \\ y_i' &= (y_i - 60.394)\cos 15 - (x_i + 58.825)\sin 15 \end{aligned} \tag{4.2}$$

The value of r was chosen according to the procedure in Sec. 3.3.3 and the whole data set D1 was divided into 101 × 159 square cells. The two-stage median filtering algorithm was then used to detect the outliers from data set D2. In the first stage of the algorithm, a neighborhood size of 13 was selected and a vertical threshold of 1 m was chosen. After the first stage, 64, 835 data points were flagged as potential outliers. A neighborhood size of 7 was used in the second stage of the algorithm and 58, 773 data points were detected as the

outliers.

Table 4.2 presents the results for data set D2 using the direct comparison of the locations of the outliers. With the proper selection of the parameters, a detection rate of over 96% was obtained while the false detection rate was about 5% . Again the processing time take by the automatic method is very small as indicated by the CPU time taken.

Table 4.2. *Results for data set D2 using the direct comparison of the locations of outliers detected by the automatic outlier detection method to the locations of outliers detected by the manual methods.*

Neighborhood Sizes (M_1, M_2)	Vertical Thresh (T)	Cell Outlier Thresh (N_{out})	True Det. Rate, %	False Det. Rate, %	CPU Time (Minutes)
[11], [7]	1.0	6	93.70	4.80	13 : 40
[13], [7]	1.0	6	96.71	5.10	14 : 05

4.5 Discussion of the Results

It was shown in the previous section that with the proper selection of parameters, the automatic outlier detection method can correctly identify most of the outliers and preserve the valid data points in the multibeam data. There are two main parameters in the automatic outlier detection method, the neighborhood size and the vertical threshold. In order to show the effect of parameter selection on the results, the automatic outlier detection method was used to detect outliers in data set D1 with different values of M_1, M_2, and T. Also to illustrate the function of each stage of the two-stage median filtering algorithm, the results were examined after each stage.

Table 4.3 shows the results after the first stage of the two-stage median filtering algorithm for data set D1 with different values of neighborhood size M_1 and vertical threshold T. The values of other parameters in the algorithm are the same as in Sec. 4.4.1. In Table

4.3, the potential outliers flagged in the first stage were used to calculate the detection rate and false detection rate to show the effect of this stage on the results. The table shows that

Table 4.3. *Results after the first stage of the two-stage median filtering on multibeam data set D1 for several different neighborhood sizes and vertical threshold values.*

Neighborhood Size(M_1)	Vertical Threshold(T)	Detection Rate, %	False Detection Rate, %	CPU Time (Minutes)
[9]	3.0	11.34	0.00	9 : 00
[9]	2.0	23.20	0.00	9 : 12
[9]	1.0	90.60	7.31	9 : 40
[17]	3.0	11.34	0.00	12 : 30
[17]	2.0	23.20	0.00	13 : 04
[17]	1.0	96.24	16.43	13 : 20

with the larger neighborhood size the detection rate is increased, but the false detection rate is increased as well. The reason is that more data points lying on the boundaries of true objects are being detected as potential outliers. With the smaller neighborhood size, the detection rate is comparatively smaller and the false detection rate is low as well. Using a larger vertical threshold yields a smaller detection rate because there are few outliers that have significant depth difference relative to the surrounding data points. The purpose of this stage is to achieve high detection rate while keeping the false detection to a reasonable level so that it can be reduced further in the second stage down to an acceptable level. The CPU time shown is the time taken by the prepocessing and first stage of the two-stage algorithm to process data set D1. The time taken increases as the neighborhood size increases because the search for locating the larger number of neighborhood data points takes more time as may be expected.

Table 4.4 shows the results after the second stage of the two-stage median filtering algorithm for data set D1. By comparing Tables 4.4 and 4.3, it can be seen that the false

Table 4.4. *Results after the second stage of the two-stage median filtering on multibeam data set D1 for several different neighborhood sizes and vertical threshold values.*

Neighborhood Sizes(M_1,M_2)	Vertical Threshold(T)	Detection Rate, %	False Detection Rate, %	CPU Time (Minutes)
[9],[7]	3.0	11.34	0.00	11 : 00
[9],[7]	2.0	23.20	0.00	11 : 32
[9],[7]	1.0	90.60	4.84	12 : 40
[17],[7]	3.0	11.34	0.00	15 : 50
[17],[7]	2.0	23.20	0.00	16 : 44
[17],[7]	1.0	96.24	9.26	18 : 50

detection rate for the neighborhood size $M_1 = 9$ is reduced from 7.31 to 4.84 in the case of a vertical threshold of 1.0. This illustrates that the second stage of the algorithm reduces the false detection rate. However, if the choice of the neighborhood size in the first stage of the algorithm is inappropriate, then the effect of the second stage on false detection rate is limited. This can be seen from the fact that the false detection rate for the neighborhood size M_1=17 is also reduced, however it is still very high. The reason it remains high is that the neighborhood size M_1=17 eliminates some groups of data points of more than 7 that represent true objects on the seabed. Thus, for data set D1 generated with the echosounder resolution of 7, a neighborhood size M_1 of 17 is an improper choice as it is eliminating true objects on the seafloor. The CPU time shown in the Table 4.4 is the total time taken by the two-stage median filtering method.

By examining the CPU time taken by the automatic outlier detection method further, it was found that 20% of the CPU time is used in the preprocessing of the multibeam data. Approximately 60% of the CPU time is spent by the localization method to locate the required number of nearest neighbors. The rest of the CPU time, about 20%, is utilized in the median filtering and the flagging of the outliers. The second stage takes much less time

compared to the first stage because only a relatively small number of data points in a few selected cells are reprocessed in this stage.

4.6 Visualization of Results

In this section visualization of bathymetric images generated from the raw and cleaned multibeam data sets is used to validate the results. To generate the images the depth values at the locations of the outliers are replaced with the predicted depth values. The predicted depth value of an outlier in this case is the median value of its larger neighborhood computed in the first stage of the two-stage median filtering algorithm. The bathymetric images of the data sets processed using the manual methods are also presented here. All the bathymetric images are generated using a software package named *Surfer*. A regular grid whose size depends on the density of the data set is first created using this software package. Then this is used to generate a bathymetric image of the data set using an interpolation method known as the Delaunay triangulation method [26].

Fig. 4.3 shows the bathymetric image of the original data set D1. There are many outliers visible in this image. Fig. 4.4 shows the bathymetric image of the same data set D1 with the outliers identified and replaced using the automatic outlier detection method. In Fig. 4.5 the bathymetric image of data set D1 that was manually cleaned at the IOS is shown. The visual comparison of Fig. 4.4 and Fig. 4.5 shows that the seafloor topographies in these two images are almost identical. This indicates that using the proposed automatic outlier detection method data points that were outliers have been identified and replaced and data points that represent true objects have been preserved. Figs. 4.6, 4.7 and 4.8 show similar results for data set D2.

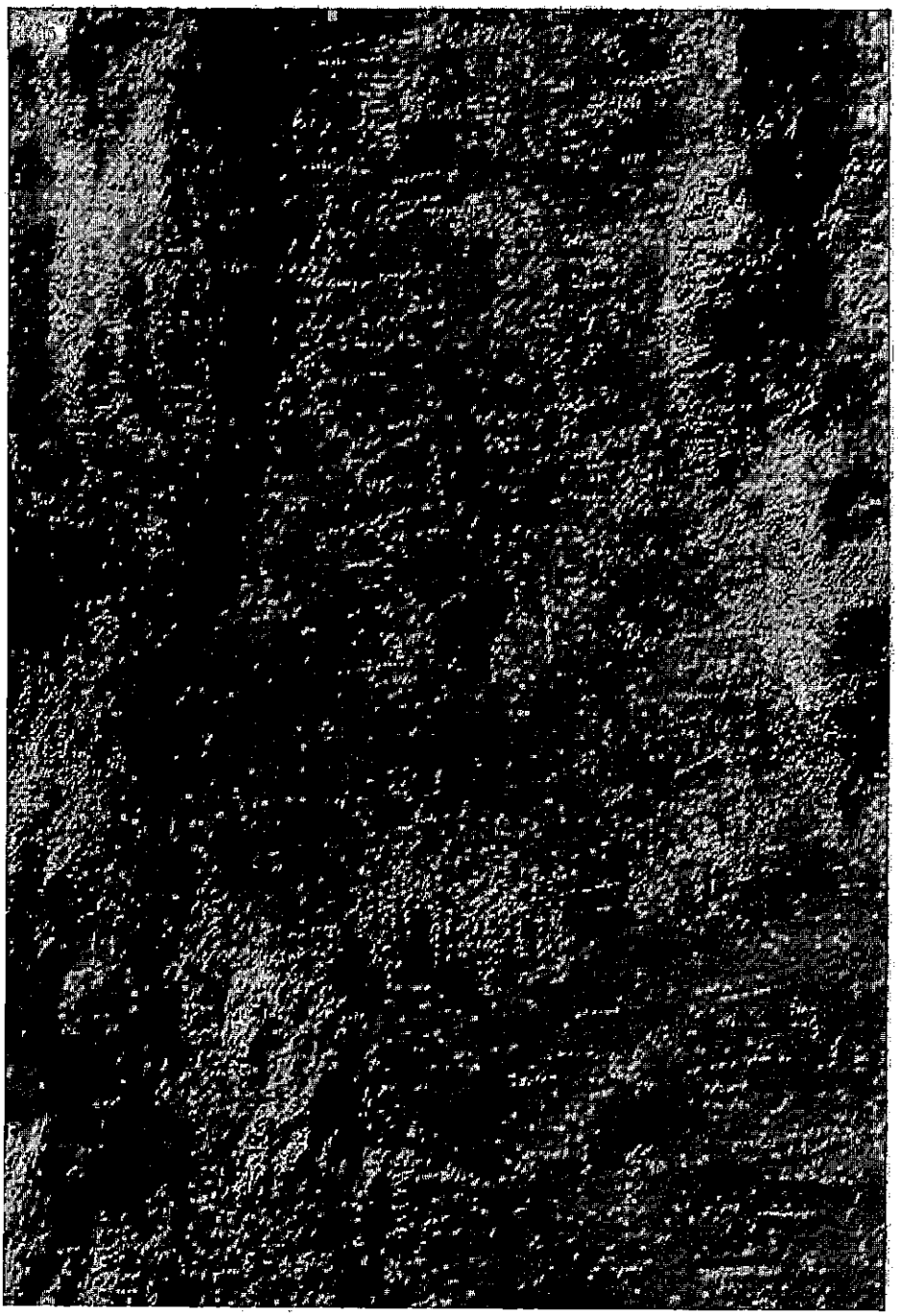

Figure 4.3. *Bathymetric image of raw multibeam data set D1.*

Figure 4.4. *Bathymetric image of multibeam data set D1 that was cleaned using the automatic outlier detection method.*

Figure 4.5. *Bathymetric image of multibeam data set D1 that was cleaned using manual cleaning method.*

Figure 4.6. *Bathymetric image of raw multibeam data set D2.*

Figure 4.7. *Bathymetric image of multibeam data set D2 that was cleaned using the automatic outlier detection method.*

Figure 4.8. *Bathymetric image of multibeam data set D2 that was cleaned using manual cleaning method.*

4.7 Conclusions

Results obtained by using the proposed automatic outlier detection method to detect outliers from the multibeam field data provided by the IOS have been presented. These results were evaluated using direct comparison of the locations of outliers detected by the proposed automatic outlier detection method to the locations of outliers detected by the manual cleaning methods used at the IOS. The evaluation of results shows that the automatic outlier detection method was able to detect 95% of the outliers in the shallow water multibeam data. The false detection rate of the outliers were found to be below 5%. The results also show that the automatic outlier detection method is very time efficient.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

The thesis is based on a project sponsored by the Institute of Ocean Sciences for the development of efficient data processing algorithms to remove outliers from the shallow water multibeam echosounder data. Multibeam data collected using echosounder systems, such as the EM3000, contain outliers that need to be identified and removed during the post-processing before the data can be further used to create bathymetric charts. Current manual methods to remove outliers are very time consuming and an automatic method to detect outliers is required that will detect the outliers and do so in a very reasonable time.

Several methods available in the literature were investigated as potential automatic outlier detection methods. Based on this investigation, a method based on robust estimation was chosen as one of the possible automatic methods to detect outliers. As the outliers in the multibeam data can be considered as impulsive noise, median filtering was also investigated as a method to detect outliers. In Chap. 2, these two methods, namely, robust estimation and median filtering, were evaluated as possible candidates for an automatic outlier detection method. Both methods were implemented using MATLAB and tested using synthetic data. With the proper selection of parameters used, both methods were successful in detecting the outliers and preserving the true objects in the synthetic data. The median filtering method was found to be very time efficient and hence it was chosen as the candidate for the automatic outlier detection method.

In Chap. 3, a new automatic outlier detection method was proposed. This method is based on a two-stage median filtering algorithm. The method has three main parts, the preprocessing, the first and the second stages of the two-stage median filtering algorithm. The preprocessing of the multibeam data is done to facilitate the search for required data points and is an essential step in the two-stage median filtering. In this part, the multibeam data is geometrically transformed, normalized, and divided into a number of cells. Then the first stage of the two-stage median filtering algorithm is used to detect potential outliers using a preselected neighborhood size and vertical threshold. A localization method based on the Euclidean distance between locations of the data points is used to detect the required number of neighborhood points. The outliers detected at this stage are flagged as potential outliers. This stage was designed to achieve a high outlier detection rate. Some selected cells that have a higher number of potential outliers than what is statistically expected are then reprocessed in the second stage of the two-stage algorithm. This stage is designed to validate the data points located on the boundaries of true objects that have been declared as potential outliers to reduce the false detection rate. The selection of parameters used in the two-stage median filtering was also discussed.

In Chap. 4 real field multibeam data sets were used to evaluate the performance of the automatic outlier detection method. The results obtained show that most (over 95%) of the outliers present in the multibeam data sets can be identified and the objects on the seabed are preserved as the false detection rate is low (below 5%). The automatic outlier detection method can locate the outliers very time efficiently. This means that the method can be implemented in a quasi-real time multibeam post-processing system to detect outliers in a timely and efficient manner.

5.2 Future Work

The work done so far provides a strong case for using the automatic outlier detection method to detect outliers in the field multibeam data. The following steps should be con-

sidered for future research to improve the proposed algorithm further:

- A very effective but simple localization method is used in the proposed algorithm. Still most of the CPU time is spent on localization and thus to make the method more efficient, new localization methods could be developed or the one presented in the thesis can be improved. This will further reduce the processing time taken to clean the large multibeam data sets.
- The method presented was tested with shallow-water multibeam data as its performance is sensitive to the parameters used in the two-stage median filtering algorithm. For shallow-water multibeam data, the selection of neighborhood sizes and vertical threshold is easier due to the limited range of depth variation involved. More testing should be done to evaluate the use of this method for deep water multibeam systems.
- The proposed algorithm uses two stages to detect the outliers. The algorithm can be further improved by changing the neighborhood size depending upon the location of the data point that is being processed. If the data point is near the edges of true objects then a smaller neighborhood size can be used; otherwise a bigger neighborhood size can be used. For this method to work, the edges of true objects in the multibeam data will need to be identified. This method needs further research and if successful will eliminate the need for the second stage of the proposed algorithm.

Bibliography

[1] D. Bhattacharya, *Neural Networks for Signal Processing*, Ph.D. dissertation, The University of Victoria, Department of Electrical and Computer Engineering, Victoria, BC, Canada.

[2] R. J. Urick, *Principles of Underwater Sound*, 1st ed. New York: McGraw-Hill, 1967.

[3] G. Haines, *Sound Underwater*, 1st ed. New York: Crane Russak, 1974.

[4] G. Burke, S. Forbes, and K. White, "Processing large data sets from 100% bottom coverage shallow water sweep surveys," *International Hydrographic Review*, vol. LXV, no. 2, July 1988, pp. 75–89.

[5] D. N. Chayes, "Hydrosweep-ds on the r/v ewing," *Proceedings of the MTS/IEEE Conference OCEANS'91*, vol. 2, 1991, pp. 737–742.

[6] D. W. Caress and D. N. Chayes, "Improved processing of hydrosweep-ds multibeam data on the r/v maurice ewing," *Marine Geophysical Researches*, vol. 18, 1996, pp. 487–506.

[7] N. C. Mitchell, "Processing and analysis of simrad multibeam sonar data," *Marine Geophysical Researches*, vol. 18, 1996, pp. 729–739.

[8] J. Eeg, "Image coding using vector quantization: A review," *International Hydrographic Review*, vol. LXXII, no. 1, March 1995, pp. 33–41.

[9] H. Claussen and I. Kruse, "Application of the dtm-program tash for bathymetric mapping," *International Hydrographic Review*, vol. LXV, no. 2, July 1988, pp. 117–125.

[10] C. Ware, L. Slipp, K. W. Wong, D. Wells, Y. C. Lee, and D. Dodd, "A system for cleaning of high volume bathymetry," *International Hydrographic Review*, vol. LXIX, no. 2, Sept. 1992, pp. 77–94.

[11] S. Dijkstra, L. A. Mayer, J. E. Hughes-Clarke, and C. Ware, "Interactive tools for the exploration and analysis of multibeam and other seafloor acoustic data," *NATO SACALANT Conference Proceedings*, July 1997, pp. 355–362.

[12] J. F. Bourillet, C. Edy, F. Rambert, C. Satra, and B. Loubrieu, "Swath mapping system processing: Bathymetry and cartography," *Marine Geophysical Researches*, vol. 18, 1996, pp. 487–506.

[13] Z. Du, D. Wells, and L. Mayer, "An approach to automatic detection of outliers in

mutltibeam echo sounding data," *The Hydrographic Journal*, vol. 79, Jan. 1996, pp. 137–154.

[14] H. Bisquay and N. Debese, "Automatic detection of punctual errors in multibeam data using a robust estimator," *International Hydrographic Review*, vol. LXXVI, no. 1, March 1999, pp. 49–63.

[15] D. C. Hogalin, F. Mosteller, and J. W. Tukey, Eds., *Understanding Robust and Exploratory Data Analysis*, 2nd ed. New York: Wiley, 1983.

[16] J. C. Russ, *The Image Processing Handbook*, 2nd ed. Florida: CRC Press, 1995.

[17] P. J. Huber, *Robust Statistics*, 2nd ed. New York: Wiley, 1981.

[18] F. Hampel, P. Rousseeuw, and W. Stahel, *Robust Statistics: The Approach based on Influence Functions*, 2nd ed. New York: Wiley, 1986.

[19] S. K. Mitra and J. G. Kaiser, *Handbook for Digital Signal Processing*, 2nd ed. New York: Wiley, 1993.

[20] T. S. Huang, Ed., *Two-Dimensional Digital Signal Processing System II: Transforms and Median Filters, Topics in Applied Physics*, 2nd ed. New York: Springer-Verlag, Berlin, Heidelberg, 1981.

[21] J. D. Foley and A. Van Dam, *Fundamentals of Interactive Computer Graphics*, 2nd ed. Massachusetts: Addison-Wesley, 1984.

[22] D. Hearn and M. P. Baker, *Computer Graphics*, 2nd ed. N.J.: Prentice-Hall, 1986.

[23] S. W. Sloan, "A fast algorithm for constructing delaunay triangulations in the plane," *Advances in Engineering Software*, vol. 9, no. 2, Jun. 1987, pp. 34–55.

[24] B. L. Berry and D. F. Marble, *Spatial Analysis: A reader in Statistical Geographer*, 2nd ed. N.J.: Prentice-Hall, 1984.

[25] W. T. Collins and J. L. Galloway, "Seabed classification with multibeam bethymetry," *Sea Technology*, Sept. 1998, pp. 45–49.

[26] R. Sibson, "Locally equiangular triangulations," *Computing Journal*, vol. 21, no. 3, Sept. 1978, pp. 243–245.

VITA

Surname: Mann ***Given Names:*** Manjinder

Place of Birth: Victoria, British Columbia, Canada

Educational Institutions Attended

University of Victoria	1998 to 2003
Nagpur University, India	1990 to 1994

Degrees Awarded

B.Eng.	Nagpur University, India	1994
PGDCA	Punjabi University, India	1996

Honors and Awards

University of Victoria Research Assistantship	1998-2000
Punjab State Merit Scholarship	1989-1994

Conference Publications

1. M. Mann, P. Agathoklis and A. Antoniou, "Automatic Outlier Detection in multi-beam data using median filtering", IEEE Pacific Rim Conference on comm., comp. and signal processing, Victoria, pp. 690-693, August 2001.

UNIVERSITY OF VICTORIA PARTIAL COPYRIGHT LICENSE

Title of Thesis:

Automatic Outlier Detection from Shallow Water Multibeam Data Using Median Filtering

Author: [redacted]
MANJINDER MANN
August 1, 2003