INTERACTIVE SPACE-VARIANT IMAGE FILTERING

Kevin W. Moore

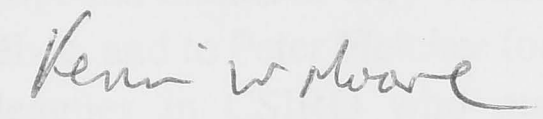
July 1997

A thesis submitted for the degree of Doctor of Philosophy at

The Australian National University

Declaration

I hereby declare that except where otherwise explicitly stated, the work presented in this thesis is my own original work.



Kevin W. Moore

Acknowledgments

My time as a postgraduate student has been one of the most interesting and busy times of my life to date. I have been exposed to many new ideas and concepts both in the field of my academic pursuit, and about the world in general. My postgraduate student life-style and work environment has helped shape who I am and the direction of my life.

There have been many people who have contributed directly or indirectly to my postgraduate work, and it is appropriate that I try to acknowledge their contribution here.

First and most importantly I'd like to thank my supervisory panel: Phil Robertson, Don Fraser, Ken Tsui, Richard Brent, and Iain MacLeod. They have provided me with technical help, financial support, and moral support.

Special thanks to Guy Vézina for providing a role model and the opportunity to visit Québec, and to Peter Fletcher for the juggling practice and inspiration. Also thanks to my colleagues in CSIRO who made my work environment enjoyable and technically stimulating: Don Bone, Neale Fulton, Lifang Gu, David Keightley, Danielle Landy, John Lilleyman, John McLaughlin, Rochelle O'Hagan, Graham Reynolds, Mike Sharrott, Dione Smith, and Duncan Stevenson for their motivation and help; Paul Veldkamp for the exercise.

Thanks to the support staff at CSIRO for helping me organize trips, sort out pay, and the myriad of other things that needed doing. My situation never made it easy for them.

I am pleased to mention my fellow postgraduate students, for the discussions on the trials and tribulations on student life: Stephen Barass, Matthew Hutchins, Andrew Lui, Rongxin Li, and Ernest Wan.

I'd like to thank those at the Image Vision and Digital Systems laboratory at the Université Laval in Québec, Canada, for making my three month stay as a visiting student enjoyable, and providing me with a motivating work environment: Prof. Denis Laurendeau, Prof. Andre Zaccarin, Daniel Germain, and Robert Wolfe.

I need to thank CSIRO for providing me with funding to pursue my studies, and for allowing me to take leave without pay from my position. The ACSys Cooperative Research Center for taking over my scholarship, for organizing the student symposiums and for funding travel.

Without my Rover friends, and my friends from the ANU Mountaineering club, I would never have survived. With them I renewed myself almost every weekend.

Finally I need to thank my family, especially my father John, without whom I probably would never have dreamed of doing postgraduate study. He has given me encouragement and support during my entire education. Also my mother Jan, and my brother Alan for listening to me all the time.

Abstract

Image filtering is a complex and often difficult operation, more so when the requirements of the problem dictate that the filter must vary spatially over the image. For many problems, the extra difficulty and cost caused by this spatial variation puts the best available forms of such filtering beyond the reach of most users.

We develop a general workbench for space-variant image filtering that simplifies the tasks of implementing and applying spatially variant filters. The workbench is based on a classification scheme that classifies such filters according to the source of their parameter values. It defines four major source classes: data, system, geometric, and user dependent filters. The definitions of the filter classes characterize the shared behaviour between different filters, which in turn is exploited by the workbench for the development of tools to be used with filters in that class. The workbench is designed to maximize the degree of code reuse between applications.

The workbench supports interactive visualization as a tool for understanding the effects of, and improving the results of image filtering. These techniques allow the filter variation to be directly or indirectly controlled. This helps steer the computations towards optimal solutions by using the user's expert domain knowledge. A side benefit is that it increases the user's conceptual understanding of a filter's action. These techniques are developed initially for removing quantization artifacts from shaded terrain images, and for improving the signal-to-noise ratio of seismic data. The workbench then generalizes the concepts to a wider range of problems.

Evaluating the results of filtering is often difficult, so here we develop a method for comparing filters based on the human visual system's sensitivity to motion. This and other methods are incorporated into the workbench to aid filter control.

The research area of interactive space-variant image filtering is explored in this thesis, and its territory mapped out through work on the general conceptual framework, software workbench, and supporting tools.

CONTENTS

List of Figures

1 INTRODUCTION

Image filtering is a complex and often difficult operation, more so when the requirements of a problem dictate that the filter must vary spatially over an image. For many problems, the extra difficulty and cost caused by such spatial variation puts any attempt at optimal filtering beyond the reach of most users. Where solutions to such problems are found, they are usually strongly custom or application specific. General solutions have yet to be developed that can be used to solve a variety of different space-variant filtering problems.

A key problem with existing methods is the lack of information they provide to the user about the action of a filter. Without proper understanding, a user is less likely to achieve the desired filtering result, and interpretation of the data may be subject to error.

Interactive visualization or computational steering is being increasingly used as a tool for providing a user with a means for understanding the effects of, and improving the results of calculations [Chen et al., 1996]. There has been a small number of instances where interactive filters have been developed for specific space-invariant filters (e.g [Miller et al., 1983], [Miller et al., 1988], [King et al., 1987], and [Huck et al., 1991]), but no generalized work or work on developing these techniques for space-variant image filters has been reported.

This thesis opens up the problem area of interactive visualization for steering and evaluating space-variant image filters, and seeks to lay its foundations. It solves a number of specific space-variant filtering problems which involve interactive techniques, and then proposes a software workbench for space-variant filtering. It demonstrates how the workbench can be used for each problem.

1.1 Importance of filtering

Filtering

Filtering is an essential part of many image processing and graphics operations. In image restoration, filters are used to invert degradations caused by the imaging system (e.g [Trussell and Fogel, 1992]). The more accurately degradation can be inverted by a filter, the higher the fidelity of the image to the original. In image enhancement, filters are used to accentuate specific image features to make further processing by visual interpretation or machine processing easier. More precisely controlled and targeted filters will better accentuate desired features without introducing degrading artifacts. In coordinate transforms, filters are used to resample images at non-pixel positions. Poor reconstruction, and insufficient low-pass filtering before sampling can lead to aliasing and other artifacts such as ringing and excessive blurring (e.g [Mitchell and Netravali, 1988]).

Space-variant filtering

Most real world applications require or would benefit from the use of space-variant filters. Imaging systems suffer from aberrations, diffraction, motion blur, atmospheric turbulence, recording medium noise, sensor sensitivity variation, and other space-variant system imperfections. A restoration filter must vary its operation over the data to account for the spatially varying point spread function of the real system. For example, in seismic data processing space-variant band-pass filters are used to improve the signal-to-noise ratio of the reflected signal. As the source signal's higher frequencies are attenuated more rapidly by the geology than the lower frequencies, the filter's bandwidth must vary with depth to achieve the most optimal results [Yilmaz, 1987].

In image enhancement, space-variant filters are required to enhance different areas of an image based on data properties or spatial position. For example, in digitized street maps, roads and textual annotations may be selected for enhancement over textured regions [Moore and Vezina, 1995].

Coordinate resampling requires space-variant filtering if the sample spacing is irregular with respect to the original image grid. The reconstruction low-pass filter's bandwidth needs to vary over the image to retain sampling integrity. For example, to prevent aliasing when resampling from a polar to a Cartesian grid, low-pass filtering must be performed when super-sampling regions of the original data [Nickerson and Haykin, 1989].

Difficulties with space-variant filtering

Space-variant filters are generally more computationally expensive than space-invariant filters due to the extra complexity introduced by the spatial variation. Their calculations are less regular, and thus it is harder to optimize their computation. Modeling the spatial variation required by a problem can be difficult due to incomplete information. Depending on the application, calculation of filter variation can take hours or days. In some cases it may be impossible to correctly compute the required spatial variation, and it can only be approximated. In many applications, space-variant filters are approximated by space-invariant filters for these reasons [Greene and Heckbert, 1986].

Often underestimated but still important is the ability of a user to understand the operation of a filter. If the action of different filters is well understood, it is more likely that they will be correctly applied. Different filters add different types of artifacts to the results. The introduction of these artifacts is unavoidable for many problems but their degrading effect can be reduced if, through understanding, the user can "see past" the artifacts when interpreting the data. The action of, and the artifacts introduced by space-variant filters are more difficult to understand and interpret than for space-invariant filters, and thus require extra attention.

1.2 Power of the human visual system

The human visual system is a powerful image processing tool which is largely under-exploited for aiding and controlling image processing operations. A task such as recognizing a tree is incredibly difficult and time consuming for advanced computer vision systems (and not always possible), yet is done almost instantaneously by any two year old child. The speed and reaction of players in any number of games where balls and other projectiles require tracking and fast response is an example of the powerful combination of the human vision system with physical motor responses. Often in these fast game scenarios, action is accomplished almost without thinking, as the visual processing and physical response are performed at a pre-conscious or pre-attentive level. The ability for humans to quickly recognize and interpret objects, regardless of spatial variation, orientation, and incomplete details is far beyond the capabilities of current and immediate future image processing and vision systems.

Using the power of the human visual system in many image processing problems can help to reduce their computational cost, improve results, and allow better understanding of the process by the user [Bracewell, 1995]. Unfortunately the human visual system is not being exploited in this fashion for many tasks which would benefit from it.

The human visual system is usually applied to problems in the real world through physical interaction [Robertson, 1991]. New objects are best examined by holding and rotating them to gain views at different angles. The human hand-eye coordination skills are well developed through application to physical problems. Thus, the human visual system can be exploited for image processing problems through hand-eye-brain interaction. For interactive computing the control, computation, and feedback cycle has to be completed in a time that appears natural. Typically this loop should be computed 20 times or more a second for smooth interaction, but it is possible to achieve acceptable results with interaction as low as once a second. Until recently, it has not been possible to achieve such rates for most filtering problems, and even today only a small number of computers are so capable. With increases in computing power, computers capable of interactive speeds for such problems will become more widespread, and the need for interactive control of filters will grow.

1.3 Interaction paradigms

In addition to a lack of computer power, the lack of suitable paradigms for interaction is another reason why these solutions are not more widely adopted. For interactive filtering, the most obvious control is through the filter parameters. For some filtering problems this, combined with an appropriate visualization is sufficient. For others, such interaction is difficult and counter-intuitive. In these cases a higher level

interaction paradigm may need to be employed; an intelligent method whereby the user can confidently steer the filter's operation.

1.4 Visualization problem

Necessary for control of a filter's operation is the means for judging the results of interaction. Viewing image statistics is one method that has a place, but a suitable visualization can go much further. The difficulty lies in "what to visualize" to allow a user to make a well informed decision. The filtered image is an immediate option which is already available. If computation is fast enough, interaction by swapping between filters, or by changing filter parameters, can produce animated change in the image which is easily seen. If computation is too slow, then results can be pre-computed and displayed back as if in real-time to gain a similar effect.

Visualization for interaction in any form must provide sufficient information to allow a user to converge to a desired solution. Combination of several visualizations may increase the likelihood that a user is able to achieve such convergence.

1.5 Interactive space-variant image filtering

This thesis develops a software workbench that simplifies the task of implementing, controlling, and visualizing space-variant image filters.

First we consider several filtering problems in terms of their computational requirements, scope for computational steering through user interaction, and evaluation through suitable visualization techniques. Unique solutions to these problems are presented in each case.

Generalizing the solutions to the specific problems and incorporating them into a single coherent model, together with some additional ideas, leads to the design of a workbench that achieves the above mentioned goals. Further examples are used to demonstrate the use of this framework.

1.6 Organization of thesis

Chapter 1 introduces the major requirements underlying this work, and outlines the approach taken to meeting these requirements.

Chapter 2 examines recent work on interactive control of space-invariant filters, space-variant filtering, filter comparison and evaluation, and image restoration.

Chapter 3 formulates a space-variant smoothing filter whose action can be tuned to the data through three types of user interaction. This filter is developed for removing quantization artifacts from shaded digital terrain images. These images are commonly smoothed by uniform smoothing filters to improve their subjective quality, but such smoothing reduces the total information content of the images. The filter developed minimizes this information loss by smoothing most where noise is most visible, and has the added benefit of enhancing user interpretation of the data through its unique methods of interaction.

Chapter 4 develops an interactive space-variant band-pass filter for removing noise and increasing the signal-to-noise ratio of reflection seismic data. Traditional methods merge the results of uniform band-pass filters, that are selected in a static way by the geophysicist, to effect a variable band-pass filter. The filter method developed enables interactive visualization of the frequency content of the data, and allows direct control over the continuous variation of the cutoff frequencies of the filter.

Chapter 5 examines visualization of digital raster maps (DRM) through geometric scaling. It describes an experiment to determine the most perceptually pleasing filter for scaling [Moore and Vezina, 1995], and a software clustering scheme to speed up parallel scaling algorithms.

Down-scaling DRM requires a low-pass filter to prevent aliasing. Due to the complex nature of the human visual system, the most numerically accurate filter may not provide the best filtering for applications where the resulting images are viewed by humans. The experiment is designed for the subjective selection of a suitable perceptually pleasing filter for down-scaling DRM

Image filtering is a costly operation, and difficult to compute at interactive rates for large images. Serial algorithms scale poorly with data size, so parallel implementation is necessary to support a more general framework for interactively filtering large data sets. To improve the efficiency and speed of parallel filtering algorithms, a software clustering scheme, which works in the reverse of traditional virtualization scheme, is developed. The clustering uses several real processors to perform the task of one virtual processor.

Chapter 6 shows how space-variant filters can be generated, modified and applied to real filtering problems interactively using visualization of filter kernel images and the effects of their application [Moore and Robertson, 1995]. Massively parallel processing is exploited to provide scalable realizations of the filtering, in which space-variant filters of varying type and bandwidth are embedded within parallel tool-kits. Control of filter characteristics is achieved using reference image masks derived from interaction, data properties, modeling parameters, and data format information.

Chapter 7 brings together the ideas developed in the proceeding chapters into a uniform framework [Moore and Tsui, 1997]. It first develops the ideas of filter dependencies, then derives a more general form of the reference masks developed in Chapter 6. These generic filter parameter control maps form the base of a software workbench that simplifies the tasks of implementing, controlling, and visualizing space-variant image filters.

Chapter 8 summarizes the results of the thesis, the major achievements and their limitations. Areas for future work are discussed.

Appendix A. presents an overview of the software design of the filter workbench described in Chapter 7. The general model of the workbench is described in terms of its major components, and class diagrams given for each of its major classes.

2 RELATED WORK AND BACKGROUND

Interactive space-invariant and space-variant filtering for image processing are new areas of research, with little direct contributions from existing work. To approach this problem, contributions have been taken from a number of related areas. These areas include interactive space-invariant filter control, subjective filter evaluation, human perception and visualization, filter evaluation and comparison, restoration, and space-variant filtering. This chapter reviews relevant work in these areas.

2.1 Interactive space-invariant filtering

Interactive filtering can be defined as the process of controlling the action of a filter, based on feedback in real time. It allows users to apply their real world expertise and hand-eye coordination to the problem of tuning or steering the operation of a filter for a particular application or data set. As the user interacts, changes in an image caused by different filtering properties can be quickly processed by the human visual system's pre-attentive processes, allowing rapid assimilation of filter characteristics, and tuning to a desired result. Interactive filtering is a dynamic form of subjective filter evaluation where the process is performed continuously. Understanding the action of a filter and its effect on data are a key benefit of interaction.

Figure 1 shows the four key elements of an interactive filtering loop and their relationships: user, interactive steering method, filtering, and visual feedback.

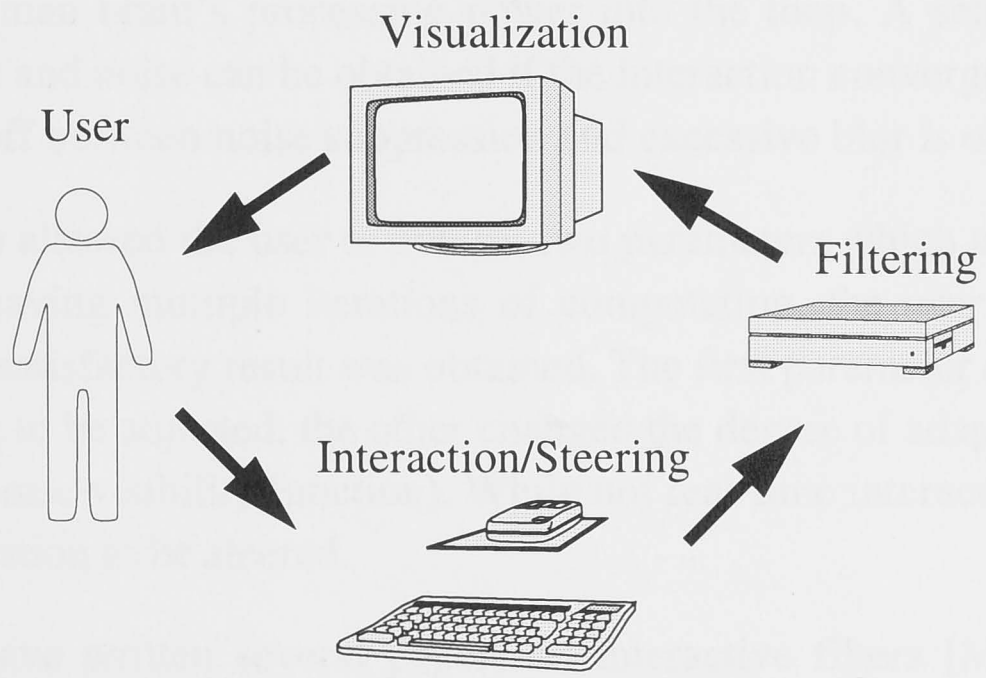


Figure 1 Interactive feedback loop

To allow a user to converge to a satisfactory solution, interactive filtering loops must provide suitable interaction and visualization feedback methods. Interactive adjustment of a filter's parameters does not necessarily allow successful filter control. This is

analogous to the problem of a colour matching. Interaction can be provided directly by allowing the user to separately adjust the colour's RGB components. There is a direct link between action and result, however, this does not make the task intuitive or quick. With filtering, having a fast response time is not sufficient, it is important that the interaction paradigm allow users to quickly understand a filter's operation, and to converge upon a desired result. For example, one intuitive paradigm for filter control which many restoration and enhancement filters can be mapped to is that of sharpening and smoothing.

The visualization feedback loop must provide sufficient information for the user to judge the results of their interaction. The more detailed the information provided, the finer can be the control of filtering. The feedback needs to be tied to the interaction method and goals of the filter for best performance. The most common form of feedback provided is a view of the filtered image. While sufficient in many cases, additional information can speed up the process. Visualization of some filter properties such as frequency response, mean square error, other metrics may also be needed.

The concept of interactive filtering has been around for at least the last 20 years, but practical experimentation and application has been limited due to the availability of sufficiently powerful computing hardware. Early work focused on iterative filtering where the user could modify parameters between steps of the computation.

[Anderson and Netravali, 1976] applied iterative filtering to the problem of noise removal. Their goal was restoration for human viewing where the information most discernible to the human eye should be presented, and the most discernible noise removed. Since understanding of the human visual system is incomplete, their approach was to add the human brain's processing power into the loop. A satisfactory tradeoff between resolution and noise can be obtained if the interaction converges, which is when an optimum tradeoff between noise suppression and excessive blur is obtained.

Their approach allowed the user to control two parameters which adapted the action of the filter. By having multiple iterations of computation, the user could adjust the parameters until a satisfactory result was obtained. The first parameter allows the overall amount of filtering to be adjusted, the other changed the degree of adaptivity of the filter to the data (via a noise visibility function). While not real-time interaction, it does allow the filter's computation to be steered.

Miller et al. have written several papers on interactive filters [Miller et al., 1983, Miller and Rollins, 1985, and Miller et al., 1988]. This work has focused on the restoration of nuclear medicine images such as those obtained from thyroid, bone, and renal scans. Good restoration is achieved through matching filter characteristics to the distortion of each image. In nuclear medicine, a series of images is normally taken of a given patient with a given equipment setup. In these cases, a filter tuned for one image can be applied to other images in the set.

In the first instance [Miller and Rollins, 1985], they use some properties of nuclear medicine images to simplify the functional form of the Weiner filter. The new form is controlled by two parameters. The first parameter controls the shape of the function used to model the modulation transfer function (MTF) of the imaging system; this allows subjective adjustment of the sharpening properties of the filter. The second parameter controls the cutoff frequency of the filter and thus its smoothing properties. These two parameters allow the generation of a family of filters with continuously varying sharpening and smoothing properties. By producing different filtered versions of the data, the image can be seen under different levels of sharpening and smoothing for a more comprehensive analysis.

User interaction with the filter is via a joystick. The joystick moves a cursor within a box on the screen. The sharpening and smoothing parameters varies along the horizontal and vertical axis of the box. Computation of a new filtered image occurred in less then a second, and thus the user is able to successfully experiment with different amounts of sharpening and smoothing for image viewing. Feedback for interaction is provided by the filtered image, with navigation cues provided by the cursor's position in the screen box.

In [Miller et al., 1988], they extend this work for control of filtering in a looped time sequence of images for gated cardiac studies. A sequence of images is taken and displayed in an animation loop (some temporal filtering), with the user controlling the filtering applied to each image frame. For cardiac studies, several of these animation sequences are shown together and compared simultaneously. The same interactive Weiner filter and method for user interaction is applied as per the previous work. Real-time interactive rates are obtained for filtering these sequences using a new "imaging computer" with fast display and tightly coupled processor with large local memory.

[King et al., 1987] worked on Single Photon Emission Computed Tomography (SPECT) images. His interaction allows selection from a family of Metz filters (related to Weiner), which have been tailored to the imaging conditions. Feedback is provided via the filtered image, combined with a guideline based on knowledge of the probability distribution of the noise power spectrum. A plot of the image power spectrum overlaid by a plot of the filter's power spectrum is also shown. This additional information provides an excellent guide in the situation where feedback provided by the image may not be sufficient to allow the user to converge to a good solution.

[Huck et al., 1991] examines some problems with the Weiner filter for the application in hand, and improves upon it. This improved Weiner filter, which has greater sensitivity to noise, is combined with interactive gaussian smoothing, synthetic high edge enhancement, and non-linear tone scale adjustment. These interactions allow the user to tune the filtering to obtain a balance between noise suppression and resolution for each image.

None of the works examined address the problem of controlling the spatial variance of a filter. Filter interaction consisted solely of global interactions which control the action of a single filter over the image, or adaptivity of the filter to the data.

2.2 Filter comparison and evaluation

There is no standard method or generic technique for comparing or evaluating image filters. The variety of applications and filter requirements precludes the setting of the one set of standards for all such tasks. Custom methods are typically developed for each application or application area depending on its requirements.

Some tools or methods have become widespread in their application for filter evaluation, though how they are used can often vary between applications. Analysis of a filter's frequency response is one such method. When evaluating a low-pass filter for interpolation, the frequency response is usually examined for frequency leakage in the stop-band, and frequency attenuation in the pass-band (e.g. [Marschner and Lobb, 1994], [Park and Schowengerdt, 1982], [Parker et al., 1983], [Maeland, 1988], and [Fraser, 1989a]). How this information is used to make a comparison varies from problem to problem, and from researcher to researcher.

The process of evaluating or selecting a filter is one of trading off benefits with defects to determine the most suitable filter. Usually several metrics or methods are applied to a filter, with the results weighted to enable selection. Metrics and methods can be divided into two categories: objective methods and subjective methods.

2.2.1 Objective evaluation

Objective methods which are in essence numerical methods allow a filter to be selected in an impersonal way. Numerical methods produce numbers that can be compared in an objective way. A filter has either a higher or lower value for the metric. For reconstruction filters, there exist a number of different numerical methods. Three of them will be discussed: L_2 norm, the smoothing and postaliasing metrics proposed by [Marschner and Lobb, 1994], and the reconstruction error bound method of [Machiraju and Yagel, 1996].

The L_2 norm finds the distance between two images. It is used to compare filters when there is a reference image that each filtered image can be compared against. The distance it computes is the sum of the squared difference between each corresponding pixel value in the two images. While it produces a number that is easily compared, it can be misleading. An image can be very close to another as defined by the L_2 norm, but contain undesirable artifacts that indicate poor filtering. The reverse condition is also true. For example, if an image is shifted by one pixel and compared to itself, the images

are almost perceptually identical, but the L_2 norm distance between them is large. In general the L_2 norm is a poor metric with which to evaluate filters.

Marschner and Lobb define three metrics, the smoothing metric ***S***, the postaliasing metric ***P***, and the overshoot metric ***O***, to allow them to evaluate these filter qualities for reconstruction filters. The smoothing metric is defined as

$$\boldsymbol{S}(h) = 1 - \frac{1}{|R_N|}\int_{R_N} |H|^2 dV$$

where R_N is the Nyquist region, $|R_N|$ is the frequency volume of R_N, dV an infinitesimal volume element of R_N, and H the Fourier transform of the spatial filter kernel h. This metric measures the difference between a particular filter and its ideal filter inside the Nyquist region. The postaliasing filter metric measures the difference of the particular filter and the ideal filter outside the Nyquist region. This is defined as

$$\boldsymbol{P}(h) = \frac{1}{|R_N|}\int_{\overline{R_N}} |H|^2 dV$$

The overshoot metric ***O*** measures how much overshoot occurs if the filter is used to bandlimit a unit step function ρ.

$$\boldsymbol{O}(h) = max(\rho_s \otimes h) - 1$$

where ρ_s is 1 if $x > 0$ and 0 otherwise ($\otimes$ is the convolution operator).

Each filter can be plotted on a three dimensional graph, with a different metric along each axis. Marshner and Lobb use a two dimensional graph to plot the smoothing and postaliasing metric values, and examine the overshoot separately. Using this graph, a filter can be selected with the right balance of these two properties.

The metrics perform well for evaluating interpolation filters. Marshner and Lobb note that the postaliasing metric does not adequately address the problem of sample frequency ripple, however, the results of the metrics correlate well with the observed behaviour of the filters.

While these metrics measure the overall difference in regions of the frequency response, no account is made of the shape of the response. This may result in better than expected values for some filters, and worse in others. Having more than one metric allows the user to choose a filter with the right balance to suit their reconstruction requirements.

[Machiraju and Yagel, 1996] develop a spatial domain method for determining the point-wise error bound of reconstruction filters. The error bound is defined as the maximum difference between the value reconstructed by a filter, and the ideal reconstructed value. Using the infinite sinc function as the ideal reconstruction filter,

they define the *truncation error* (e_t) as the difference between the ideal sinc filter and any truncated approximation of it. The *truncation error* can be calculated by adding together those filter terms dropped during truncation. Since calculation of this can be expensive, Marchiraju and Yagel derive two approximations which give the upper bound of the error. One of the approximations is given by

$$e_t \leq \frac{Max_f \left| \sin \frac{\pi x}{h} \right|}{\pi M \cos \frac{r\pi}{2}}$$

Where Max_f is the maximum value of the function, h is the sample spacing, $2M+1$ is size of the filter kernel, and r is the frequency guard (the frequency guard measures the size of the significant spectrum of the signal, and is calculated by taking the ratio of the maximum significant frequency with the cutoff frequency).

To give an error for non-sinc functions, they define a non-sinc error component which they add to the *truncation error* of the equivalently sized truncated sinc. This sum gives them the error bound for non-sinc filters. The non-sinc error component is calculated by taking the difference between the non-sinc function and its equivalently sized truncated sinc in the L_2 domain, and then multiplying it by the maximum function value under the filter.

One of the strengths of this method is its point-wise determination of error bounds. These error bounds can be used to drive visualizations of a reconstruction filter's fidelity, and for comparing different filters. It can also be used to control an adaptive reconstruction filter which limits reconstruction error. However, like the L_2 norm, it can be misleading because it doesn't take into account the perceptual quality of the errors.

2.2.2 Subjective filter evaluation

Subjective filter evaluation is the evaluation of the worth, quality, and suitability of a filter for an operation using subjective assessment. It is assessment based on personal judgement rather then objective numerical measurement. Most papers which evaluate and compare filters do so using mathematical criterion such as L_2 norm, or by a numerical evaluation of some property of the filter such as the frequency domain response. However, once a selection has been made, a vast percentage of these papers then present results visually so that the reader may judge or confirm for themselves the filter's worth. This visual inspection can be thought of as a subjective approval to validate the theoretical evaluation.

In a number of recent papers, subjective evaluation has been the primary method for the selection and evaluation of some filters, with numerical evaluation as a possible secondary evaluation. While not suitable for many applications, subjective evaluation

should play an increasingly larger role for this purpose, especially in areas where the results are intended for human viewing. Two papers which use subjective methods are reviewed.

In [Schreiber and Troxel, 1985], subjective experiments are reported which evaluate several filters for reconstruction and sampling. Their goal was to find one with a perceptually good balance between the common filter properties of sharpening, smoothing, and the effects of aliasing. In their experiments, a single image was filtered with 25 different filters, with the results arranged in a 5x5 matrix. Subsets were taken along the horizontal, vertical, and 45 degree diagonals so that each image appeared in four sets. Observers had to rank images from best to worst in each set, with each ranking receiving a value. The sum of rankings for each image gave it its final score. Ranking involved comparing five images using side-by-side comparisons, with no specific definition of "goodness". Side-by-side comparison is difficult as it forces the observer to visually correlate features and compare them (much like the "find the 10 differences" game), which is time consuming and difficult when differences are small. The observer has to move their eyes back and forth between the images to identify differences.

In [Mitchell and Netravali, 1988], the experimental aim was to classify the subjective qualities of the two-parameter cubic convolution filter. They displayed four images typifying the filtering behaviour of ringing, anisotropy, blurring, and most satisfactory. In the middle of these was displayed a sample image filtered with a random two-parameter cubic filter. Each observer had to classify the middle image according to the four categories demonstrated by the example images. A set of 500 images were classified using this method to build up a subjective picture of variation of filter behaviour over the parameter space. By using example images they provided a good evaluation criteria, and by evaluating 500 sample images they obtained good statistical results. However, subjective evaluation of images with multiple distortion effects can be difficult, and the experiment was only able to produce a rough map of subjective filter properties.

In both papers, experiments were set up to allow users to make subjective evaluation or comparison of images filtered with different filters. While producing quite successful results, their experiments provided poor methods for comparison and evaluation of images. Methods which better exploit the properties of the human visual system are developed in Chapter 5 of this thesis.

2.3 Space-variant filtering

Space-variant filters are required in a large number of problems, covering a variety of applications. Their irregular nature, caused by their spatial variation means their computation is harder to optimize. On parallel machines, balancing computational load for filter calculations is difficult. Some problem areas where space-variant filtering is

required are described, then details of a few specific examples are presented. Methods for efficient space-variant filtering are then discussed.

2.3.1 Space-variant filtering problems

A space-variant filter is one whose operation varies over the data. Variation of the filter can be dependent on the user, or due to properties of the data, properties of the system that capture it, properties of a geometric transformation applied to it. Space-variant filtering problems are examined within the broad categories of image restoration, image enhancement, graphics, and geometrical transformations.

Restoration is the process of recovering the original object from an image distorted by its imaging system. They are typically data or system dependent filters. Fidelity to the original object is the key concern. Most real world applications require space-variant filters [Ozkan et al., 1994], as the distortions are rarely uniform over the image. Many restoration algorithms approximate by assuming a uniform point spread function, which is satisfactory for some applications, but better results can be obtained through proper space-variant modeling. Example problems include correction for motion blur (e.g. [Ozkan et al., 1994]), noise (e.g. [Pratt, 1991]), atmospheric turbulence (e.g. [Thorpe and Fraser, 1996]), Single Photon Emission Computed Tomography (SPECT) (e.g. [King et al., 1987]), correction for the geometry of forward looking radar (e.g. [Jain, 1989]), and seismic data processing (e.g. [Yilmaz, 1987]).

Image enhancement can involve both user dependant and data dependent filtering. Its goal is to improve the appearance of an image so that specific information within it is more discernible. Space-variant filters can be used to enhance particular regions of interest. Examples include: edge sharpening [Marr, 1982], [Algari, Ford, and Potharlanka 1991], contrast enhancement, image segmentation, three component image model [Ran and Farvardin, 1995], and image touch up.

In graphics, space-variant filters are used for such problems as adding motion blur to simulate motion [Potmesil and Chakravarty, 1983], and interpolation between animation keyframes.

With geometrical transformation, images are resampled from one coordinate system to another. If the new samples are irregular with respect to the original image grid, then reconstruction and sampling filters must vary their low-pass filtering over the image to prevent aliasing. Examples include texture mapping (e.g. [Heckbert, 1986b]), arbitrary warps (e.g [Wolberg and Boult, 1989]), Onimax projection from projective views (e.g. [Greene and Heckbert, 1986]), resampling radar images from polar to Cartesian coordinates (e.g. [Nickerson and Haykin, 1989]), and perspective projection (e.g. [Vezina and Robertson, 1992b], [Kaba and Peters, 1993]).

2.3.2 Space-variant filtering methods

As discussed previously, space-variant filtering is usually more costly than space-invariant filtering because of the additional complexity associated with the spatial variation. Filters can vary in many ways across the image depending on the application. The cost of computing the filter to use at each pixel location can be expensive. Space-invariant filters can often be precomputed and their weights stored in a look-up table (LUT) for efficiency [Feibush et al., 1980]. If the number of variations of a space-variant filter is finite and small, then it can be precomputed and stored in LUTs (at the cost of extra memory).

In most restoration problems, the filter is designed to invert the action of the modulation transfer function (MTF) of the imaging system that captured the image. This filter's size and shape can vary; often approximations are used which simplify this variation. For some systems, the MTF can be modeled as a simple parametric function such as the Gaussian function. This increases the possibilities of optimizing filter computation. Alternatively the filter has to be calculated for each location.

For geometrical transformations, the low-pass filters used during the resampling stage are usually modeled after the ideal low-pass filter (with some approximation), where frequencies are removed above a certain frequency and preserved below it. Filter variation is in terms of its cutoff frequency. Where a finite impulse response (FIR) filter kernel is used, variation of the bandwidth can be achieved by scaling its width and height. Where sampling is sparse, a wide filter results in decreased pass-band of the filter. Where sampling is dense, a narrow filter results in increased pass-band. Where subsampling is sparse, the filter becomes very wide, covering a large number of the original samples and thus is time consuming to compute.

In image enhancement, the filter variation depends on the type of enhancement. Different amounts of low-pass filtering may be applied, so the width of the filter varies. Shape may be varied to sharpen or preserve edge features over other structures.

The most straightforward method for computing the space-variant filtering is by direct convolution of the filter function h, with the data f, in the spatial domain. Computational cost is determined by the size of the filter function (number of pixels it covers), and the cost of computing the filter weight at each point (if not pre-computed) [Heckbert, 1986b], [Feibush et al., 1980], [Greene and Heckbert, 1986].

$$g(u, v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x, y)h((u-x), (v-y))dxdy$$

The filter computation is irregular and difficult to optimize. The methods used for a SIMD massively parallel implementation have assumed the total number of filter-weighted samples per scanline are about equivalent, and each processor computes one

weight per step. Most of the filters are treated as being separable, so two orthogonal 1D filter passes are applied.

For coordinate resampling, when the filter acts as a low-pass filter, a number of methods exist for performing filtering in constant time or near constant time through the use of pre-filtered images. Pre-filtering of images has been exploited by a number of different people in a number of different ways. A few of them are summarized here.

Mip maps

[Williams, 1983] uses a pyramid structure of pre-filtered images he calls a mip map to perform constant time filtering of images for texture mapping. Constant time filtering is especially important for applications requiring smoothly flowing animation or interaction, but is useful for a broad range of applications.

Mip maps consist of a series of scaled down versions of the original image. Each level is created by performing bilinear interpolation to create an image half the size of the level above. For a colour RGB image, the image pyramid structure is shown in Figure 2. As can seen, the resulting structure only takes up 1/3 more space than the original image.

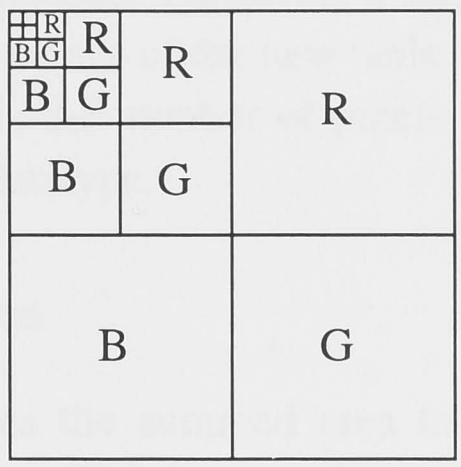


Figure 2 Mip map structure

To interpolate/filter an image at a point at a particular sampling density, the two prefiltered versions above and below that sampling density are point sampled using bilinear interpolation. The final value is obtained by linearly interpolating these values.

The footprint of the filter implemented by the mip map is limited to being square, and the filter itself is a simple box filter.

Summed area tables

An approach which is similar in many ways to the mip map is that of the summed area table [Crow, 1984]. In a summed area table, each pixel value stores the sum of all

the original pixel values in the rectangle formed by the pixel's position and the lower left hand corner.

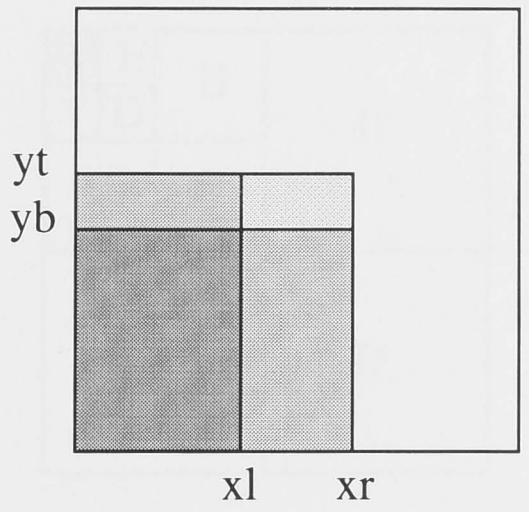


Figure 3 Summed area table

To find the sum of intensities over an arbitrary rectangle, the sum and difference of two pixel sums is taken. In the example, the average of the rectangle formed by (xr, yt) (xl, yb), is computed by the sum of table values (xr, yt) - (xl, yt) - (xr, yb) + (xl, yb).

The same averaging filter is used as per the mip map, but has the added flexibility of covering a rectangular footprint. The table size is the same as the original image unlike the mip map whose area is increased. However, a larger datatype with more bits is required to store the larger pixel values of the new table. The maximum number the new datatype must be able to store is the number of pixels of the image multiplied by the maximum value of the original datatype.

Filtering by repeated integration

[Heckbert, 1986b] generalizes the summed area table approach by what he calls filtering by repeated integration. This generalizes the summed area table so that piecewise polynomial filters can be applied as well as simple averaging box filters. Filtering is based on the property that convolution of a signal with any piecewise polynomial kernel of degree (n-1) can be computed by integrating the signal n times and point sampling it several times for each output sample. As for the summed area table, extra memory is required to store the pre-computed tables.

Wavelets

Wavelet based filtering approaches are similar to the mip map approach in that a prefiltered multiresolution structure is used. The wavelet structure [Mallat, 1989] however is based on working with frequency content in a localized sense which leads to a form of band-pass filtering of the data. The filtered versions are usually created using quadrature mirror filters. The structure often used to carry the wavelet transformed data, shown in Figure 4, is similar to that used by mip maps. Instead of RGB versions, each

structure contains only one channel of the data, but carries the horizontal (H), diagonal (D), and vertical (V) frequency information of the data in each layer.

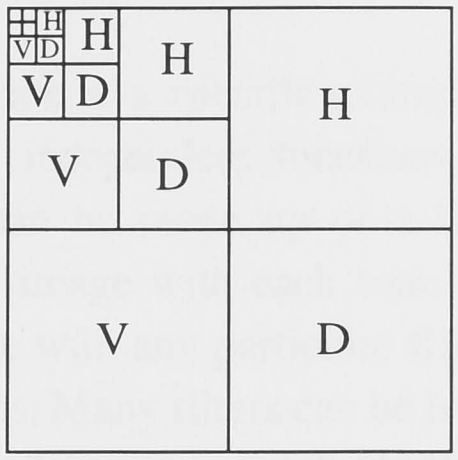


Figure 4 Multiresolution wavelet structure

One method for space-variant filtering using wavelets which lends itself to interaction consists of reconstructing the original image by combining weighted versions of each resolution layer. The weight applied to each layer can be controlled by the user. This type of filtering allows different frequencies to be removed at each image location by zeroing values in different resolution layers.

Mip maps, summed area tables, filtering by repeated integration, and wavelet based techniques all perform space-variant filtering with startup costs associated with prefiltering and table setup. As such, they are ideal for situations where a single image is filtered many times. This occurs in texture mapping when a single texture is mapped to multiple objects, or when the texture has to be continually mapped to an object – such as for animation. In these cases, the setup costs are not important. If the image is only filtered once, then more straight forward methods may be more cost effective.

Nil maps

Nil maps [Fournier and Fiume, 1988], are another method with bounded computation time, with time nearly independent of size of filter. Nil maps approximate the filter surface by a weighted sum of some suitably chosen basis functions. These basis functions are precomputed with the image, with the correct weights and the sum computed at runtime. To achieve constant time, these basis functions are computed at several different resolutions and stored in a pyramid structure. Cost of the filtering depends on the order, and the number of basis functions chosen to represent the filter surface.

PCC

Filters do not always vary their size over the image. In restoration and enhanced reconstruction, the shape of the filter may vary to better enhance different features (a locally optimal filter), or to more accurately restore a space-variant degradation. In these

cases, pyramid structures are less important, but efficient computation still is vital since the time for filtering can be expensive. Real-time filtering is important for many applications.

Parametric cubic convolution is a specific example of decomposing a filter kernel function into several linearly independent functions [Park and Schowengerdt, 1983]. Any filter from the family can be made up of a linear combination of these basis functions. Convolution of the image with each basis function is performed as a setup step. Convolution of the image with any particular filter is then replaced by a weighted sum of the precomputed images. Many filters can be broken down in this fashion. Others can be approximated by Chebyshev polynomials [Miller et al., 1983].

Coordinate transformation

The method of space-variant restoration by coordinate transforms [Sawchuk, 1974] can be applied to a variety of restoration problems. It involves applying a geometric distortion to the image so that the space-variant restoration becomes a space-invariant problem. The space-variant PSF of the imaging system is mapped to a space-invariant one. Once the mapping is achieved, a whole host of space-invariant restoration methods can be applied. Once restored, the inverse of the original geometric transform is applied.

The method can be thought of in terms of mapping a four dimensional problem to a two dimensional problem. Example problems include rectification for motion blur and turbulence.

The geometric distortion first applied to the image to turn the problem into a space-invariant one is itself a space-variant filter. However this spatial variation may be more tractable then the restoration filter. Unfortunately this method is sensitive to noise, and is limited in the scope of problems it can be used to solve.

Sectional methods

Restoration by sectional methods, [Trussell and Hunt, 1978b], is a method for applying statistical, space-invariant restoration techniques to space-variant problems. It consists of sectioning the image into parts and applying different filters to each part. The authors use localized deconvolutions to overlapping sections, iterative filters, and other global filters.

This method only supports a very coarse grained filter variation, which limits the class of problems to which it can be applied successfully.

2.4 Human perception and visualization

Visualization is the process of turning the symbolic into the geometric, turning raw data which are numerical or in symbol form, into a visual picture which allows the information to be interpreted better and faster [McCormick et al., 1987], [Kelly and Keller, 1992], [Gershon, 1994].

In order to design fast and effective visualizations to allow users to control their filtering, the human visual system, and the characteristics of perception must be understood. In [Gershon, 1992] and [Gershon, 1994] several methods for visualization based on the sensitivity of the human visual system to movement have been examined. Movement can be detected fast and efficiently by the human visual systems pre-attentive processes, allowing the brain's natural mechanisms to process them. Movement can be introduced into a static image in many ways. To evaluate the effects of changing a filter parameter, flipping between the before and after images will introduce movement into those parts of the image with the most significant differences.

2.5 Summary

To provide a context for the rest of this thesis, this chapter has reviewed past and present research areas which are relevant to interactive space-variant image filtering. It lists key results, and describes their particular relevance.

The chapter first defines the area of interactive space-invariant filtering. The key elements of an interactive filtering loop are listed: user, interactive steering, filtering, and visualization feedback. As many facets of the loop are shared with interactive space-variant filtering loops, the key elements of these are discussed in detail. Several interactive filters developed for processing medical images are reviewed and discussed in regards to the elements of the loop.

As the results of filter interaction need to be evaluated, objective and subjective methods for filter evaluation have been reviewed. For subjective filter evaluation, several experiments based on subjective filter evaluation are discussed.

Space-variant filtering comes in many guises in the real world. The types of problems areas for such filtering have been classified into three broad categories: restoration, enhancement, and graphics. The categories are described and several specific problems in each mentioned. A number of methods for efficient space-variant filtering are then reviewed.

The key points of human perception and visualization have also been described.

3 INTERACTIVE SPACE-VARIANT SMOOTHING FOR BETTER SHADING

Relief shading of digital terrain height data is commonly used for improving the visualization and interpretation of its geographic features. However, shading also highlights noise and quantization artifacts that are often invisible in an intensity plot.

In this chapter an interactive space-variant smoothing filter is developed for removing quantization artifacts. The filter variation is derived from data properties and tuned through user interaction. The interaction allows the space-variant smoothing to be optimized for visual quality for a particular data set and smoothing filter, and increases the user's understanding of the data.

3.1 Shading terrain data

Terrain data are usually obtained through direct measurement and can be degraded in several ways. The most common type of degradation is noise, which can be introduced into the data during the acquisition, transmission, and storage stages. Less common but important are quantization errors. Quantization is the process of mapping a continuous variable onto a corresponding discrete value which can take on any one of a finite set of values [Jain, 1989]. If there are insufficient values, then an artifact known as terracing can occur.

When viewing intensity plots of terrain data, noise and quantization artifacts may not be visible because of the human visual system's poor sensitivity to intensity differences; perceived contrast decreases exponentially with the sharpness of transition, and increases somewhat as a function of distance from the transition [Anderson and Netravali, 1976].

Shading data adds a depth cue [Wanger et al., 1992], which increases the visibility of features within it. It also increases the visibility of noise and quantization artifacts. In some cases these degrading features can make an image almost impossible to interpret. Figure 5 shows an 8-bit height field that has been shaded. The noise and quantization artifacts reduce the quality of the image and make its features difficult to understand.

The application of a uniform smoothing operator to a shaded image[1] is a common technique for reducing the visibility of noise and quantization artifacts. This method works because degrading artifacts are usually contained within the high frequency image bands. One problem with this method is that high frequencies in the image are lost.

---

1. Smoothing is often performed on the original data or on the gradient image used to calculate the shading

When visualizing terrain data, this loss of information can lead to false interpretation of features.

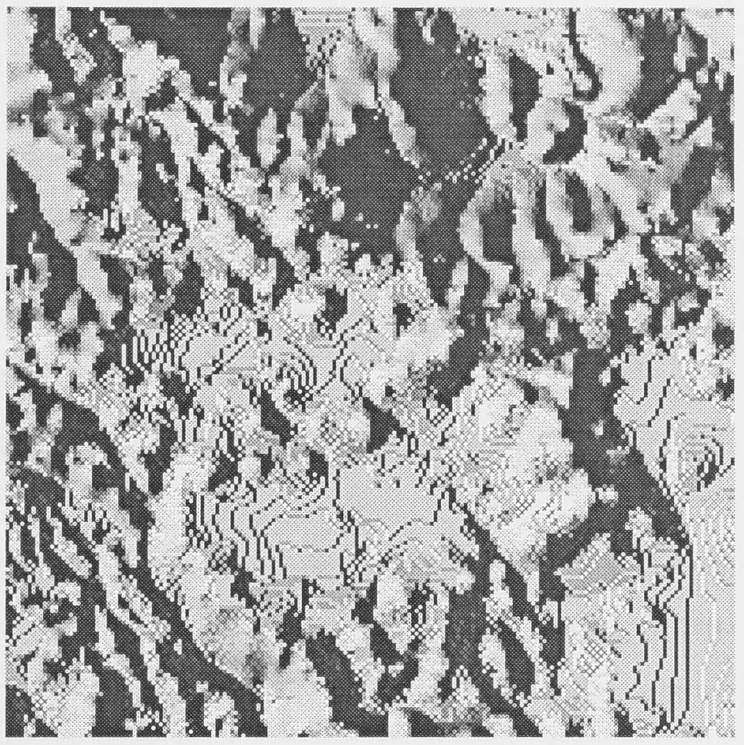


Figure 5 Shaded 8-bit height image showing quantization artifacts

A number of restoration algorithms smooth data in an adaptive way in an attempt to reduce the effects of noise while preserving the data's high frequency information, [Scher et al., 1980], [Wang and Vagnucci, 1981], [Abramson and Schowengerdt, 1993], and [Mastin, 1985]. These methods reduce the effects of the noise by attempting to smooth regions which are relatively homogeneous more than regions with high spatial variation. Noise is most visible in homogeneous regions since they don't contain much of the data's high frequency information. In regions containing high spatial variation, noise is less visible and harder to remove without removing image detail. Typically these algorithms use a form of local averaging with an adaptive component. While all these algorithms improve image quality, none gives the user any idea as to how much the image has been modified, and there are no methods for adjusting the filters for different types of data.

To address the problems of uniform smoothing filters and adaptive filters, a new interactive space-variant smoothing filter is developed. The filter is aimed specifically at reducing the degradation effects of quantization artifacts in shaded images. The effects of general noise are also reduced. The filter allows interactive adjustment of its action to allow the user to gain a better understanding of its properties, and an overall idea of how much and where the image has been smoothed. By interacting with the filter, the user should be able to achieve a suitable balance between noise removal and signal preservation.

Before describing the filter in detail, this chapter first examines the cause of quantization artifacts and the process of shading. Based on the properties of both, the

interactive space-variant smoothing filter is developed. The characteristics of this filter are revealed through several examples.

3.2 Quantization

In a quantized image, the number of different levels used to represent the image, together with the distance between adjacent levels greatly affects the fidelity of the quantized image with the original. If a grayscale image is quantized with an insufficient number of levels for its range of real values, then the artifact known as terracing occurs. More specifically, terracing occurs when groups of neighboring pixels with similar values are quantized to the same value, creating regions of a constant gray level. The boundaries between these regions form terraces when shaded, which are visually similar to contour lines found on topological maps; dark lines at constant height. Figure 6 shows where and how terraces are formed. When shaded, regions which are almost flat can appear rich in terraces. Steeper regions, or regions of high gradient can appear void of terraces.

For standard monochrome intensity images, about 64 levels or 6 bits per pixel is usually sufficient to prevent terracing. Non-uniform quantization methods (for example dithering) can reduce the number of bits required for images, but these methods are not examined in this work. For confident display of images, 256 levels or 8-bits per pixel is most commonly used. When images are to be shaded, a much greater number of levels is required.

3.3 Shading

When shading terrain images, each pixel is assigned an intensity value based on how much illumination it receives, or more precisely how much illumination it reflects. The Lambertian reflectance model gives the reflected intensity I_r as $I_r = h\cos\theta$, where h is some constant, and θ is the angle between the surface normal vector and the direction of illumination. It is commonly calculated by taking the dot product of the unit surface normal vector and the unit illumination vector.

The surface normal vector is often approximated by taking the cross product of the unit gradient vectors in the x and y directions. For one dimensional problems, the gradient direction is taken as pointing from the left to the right neighbouring data points. Single neighbour gradient functions tend to smooth less than two neighbour gradients. The method used to compute gradient vectors is one factor contributing to the visibility of quantization terraces. In most cases the greater the number of elements used for calculating the gradient, the smoother the gradient image.

The Lambertian reflectance model treats each pixel as a small flat surface upon which illumination falls. The closer to perpendicular the surface is to the direction of illumination, the greater the intensity value it receives. Therefore pixels which face similar directions will receive similar values. If adjacent pixels have quite different gradients, their shaded values will be quite different. If a pixel forms an angle greater than 90 degrees from the angle of illumination, then the pixel is self occluding and no direct illumination from the light source falls upon its surface (the pixel can still be lit by ambient and reflected illumination).

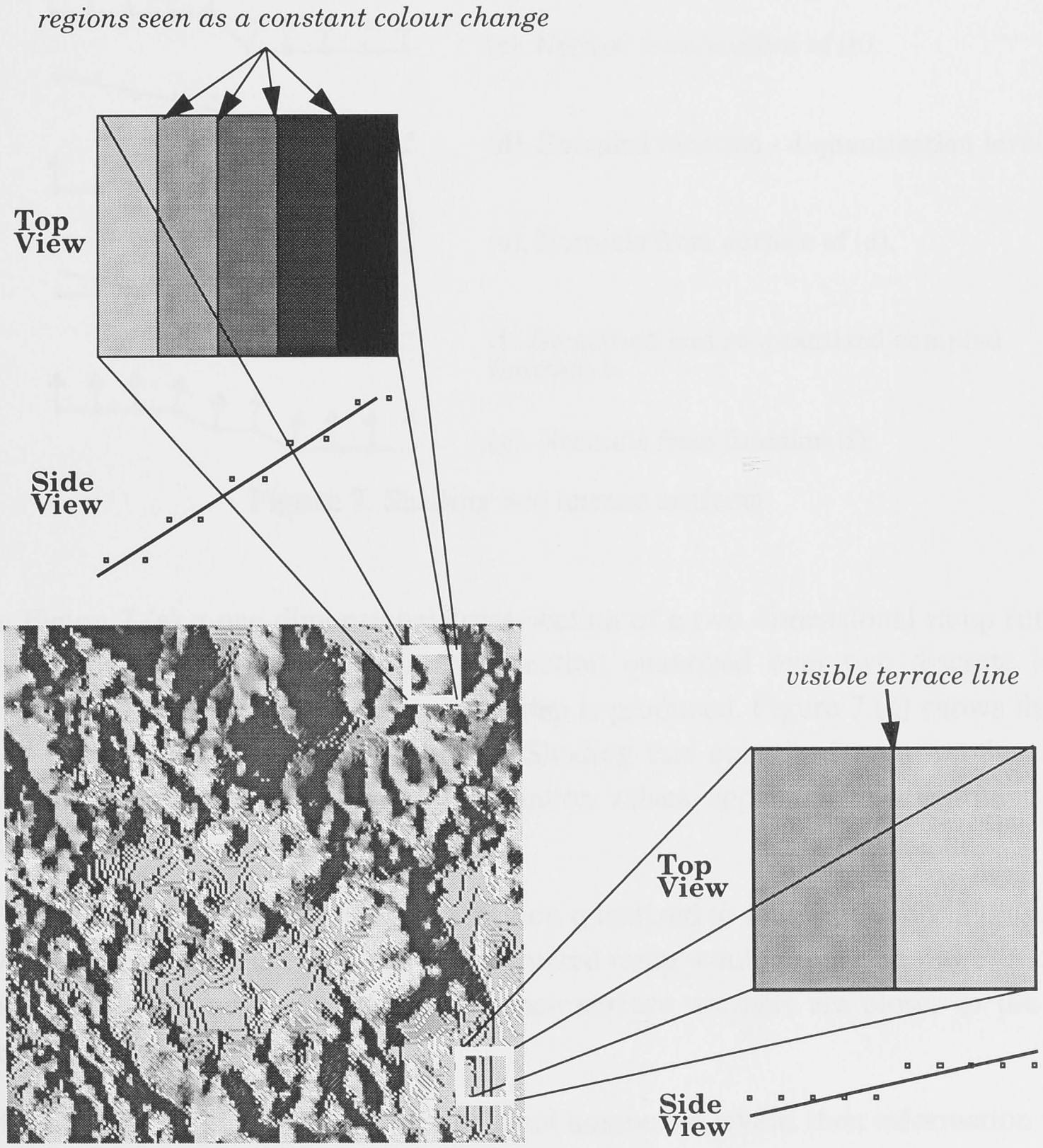
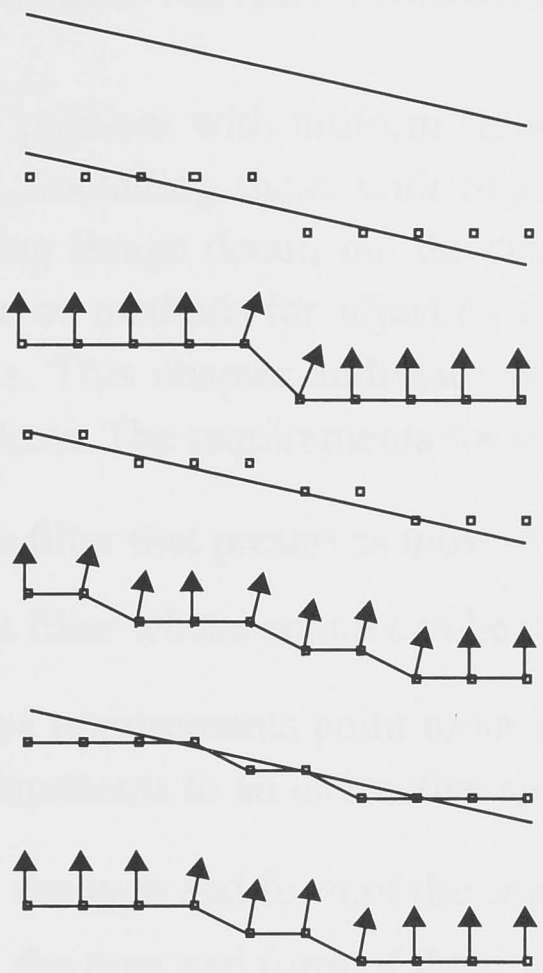


Figure 6 How and where terraces are caused in quantized images.

3.3.1 Example

To illustrate how quantization causes terracing and why shading highlights terracing, a simple example is given in Figure 7.



(a). Original Function

(b). Sampled function - 2 quantization levels

(c). Normal from surface of (b).

(d). Sampled function - 4 quantization levels

(e). Normals from surface of (d).

(f). Smoothed and re-quantized sampled function b.

(g). Normals from function (f).

Figure 7 Shading and terrace artifacts

In Figure 7.(a) a one dimensional cross-section of a two dimensional ramp function is shown. Figure 7.(b) shows this cross-section quantized over two discrete levels. Because only two levels are used, a single step is produced. Figure 7.(c) shows the unit surface normal vectors at the data points. Shading this quantized ramp produces two regions of similar smoothly varying illumination values, separated by a different shade terrace line.

Figure 7.(d) shows the same ramp function quantized to four levels, and Figure 7.(e) shows its surface normals. Shading this quantized ramp would produce a more smoothly varying ramp with no real terrace line, since surface normals are closer to the same angle.

If a function is quantized to an insufficient number of levels, then information is lost in the process and the operation is irreversible. A function quantized to two levels cannot necessarily be re-quantized to four levels with the same result as if it had been originally quantized at that level. We approximate a re-quantization process by converting the data to real values, low-pass filtering (smoothing) it, then quantizing to a new number of levels.

Figure 7.(f) shows the original two level quantized ramp function after it has been low-pass filtered and then re-quantized to four levels. Shading these samples produces

almost as good a shading as the original four level quantized ramp. More low-pass filtering would reduce the differences even further.

3.4 Space-variant smoothing

The problem with uniform smoothing filters is that they equally smooth all image features, including those with high spatial frequencies. Adaptive filters are better at preserving image detail, but do not show the user how much information is lost, and there are no methods for adjusting the level or variation of filtering for specific filters or data sets. This chapter addresses these problems and develops a filtering method that solves them. The requirements for such a filtering method are

- a filter that preserves most of the image's high frequency information
- a filter whose action can be understood and adjusted by the user.

These requirements point to an interactive space-variant filtering method. There are four components to an interactive space-variant filter that need to be addressed.

1. the type and form of the smoothing filter
2. the type and form of the filter variation
3. the filter interaction methods
4. the visualization of filter properties

3.4.1 Smoothing filter

The requirements for the smoothing filter are

- the smoothing should be continuously variable over an image
- the filter must be able to be computed fast enough for interaction.

A convolutional low-pass (smoothing) filter is chosen whose filter kernel is stored in a lookup table. To achieve variation of low-pass filtering, the lookup tables indices are scaled, [Feibush et al., 1980] [Ward and Cok, 1989]. Figure 8 shows how this is performed.

The lookup table approach allows any type of low-pass filter kernel to be used. The B-spline, cubic, sinc, windowed sinc, and Gaussian filter kernels were tested. Variation of the filter can be controlled via a simple parameter which controls the scaling of lookup table indices and hence the amount of low-pass filtering.

The speed and quality of the filter can be balanced by selecting wider or narrower filters. Wider kernels can be made to produce better quality results, but are more

expensive to compute because they cover more data points. Narrower kernels are faster but sacrifice quality.

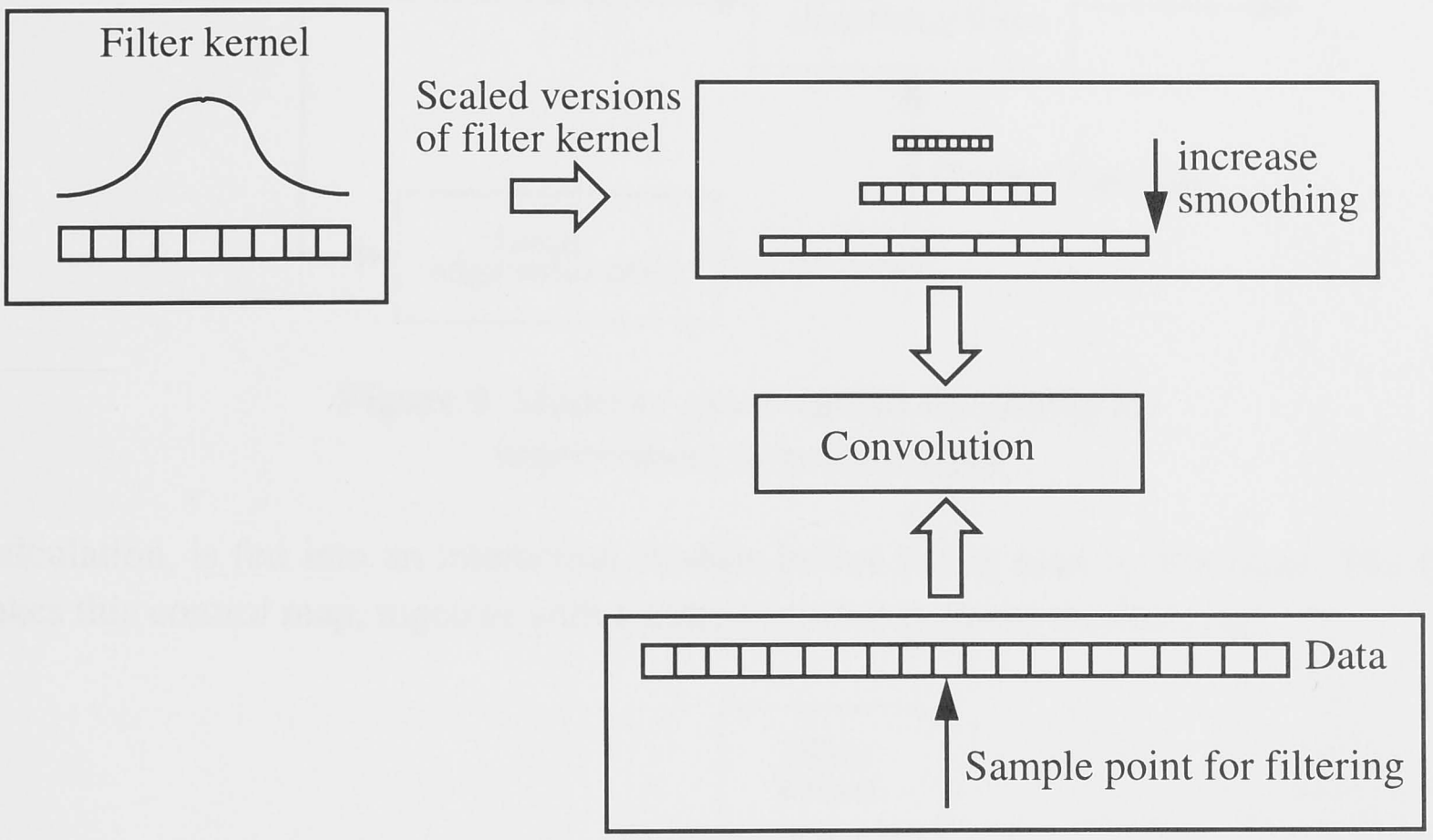


Figure 8 Convolutional scaling using lookup tables

3.4.2 Filter variation

The goal of filtering is to improve the visual quality of a shaded image. To do this based on the properties of quantization artifacts, we propose to smooth regions of low gradient more than regions of high gradient.

In [Wahl, 1987], a signal dependent space-variant smoothing operation is described for improving noisy images. It uses a simple gradient operator with thresholding to detect light-dark transitions. It applies one of three filters based on whether a vertical, horizontal, or non-transition is detected. Test images consisted of overlapping squares of uniform intensity onto which speckle noise has been added. The resulting filtered images maintained clear edges with most of the noise effects removed. The model for the operation for [Wahl, 1987] is shown in Figure 9.

We also use a gradient operator to determine smoothing. The amount of smoothing is calculated as being inversely proportional to the gradient of the image. The higher the gradient in a region, the less the region is smoothed, and vice-versa.

The smoothing parameter values calculated from the gradient are stored in an image of the same size and dimensions as the image to be filtered. This image is called a control map. Figure 10 shows a system diagram for the filter. It shows how the gradient

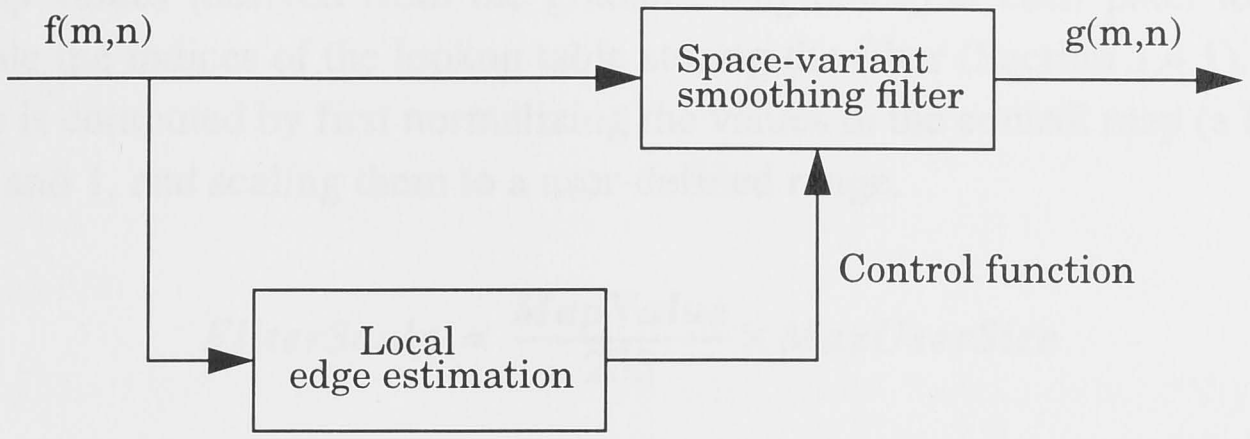


Figure 9 Model of space-variant smoothing for improvement of noisy images

calculation, is fed into an interaction module before being used by the filter. The filter takes this control map, together with a filter kernel to perform the filtering.

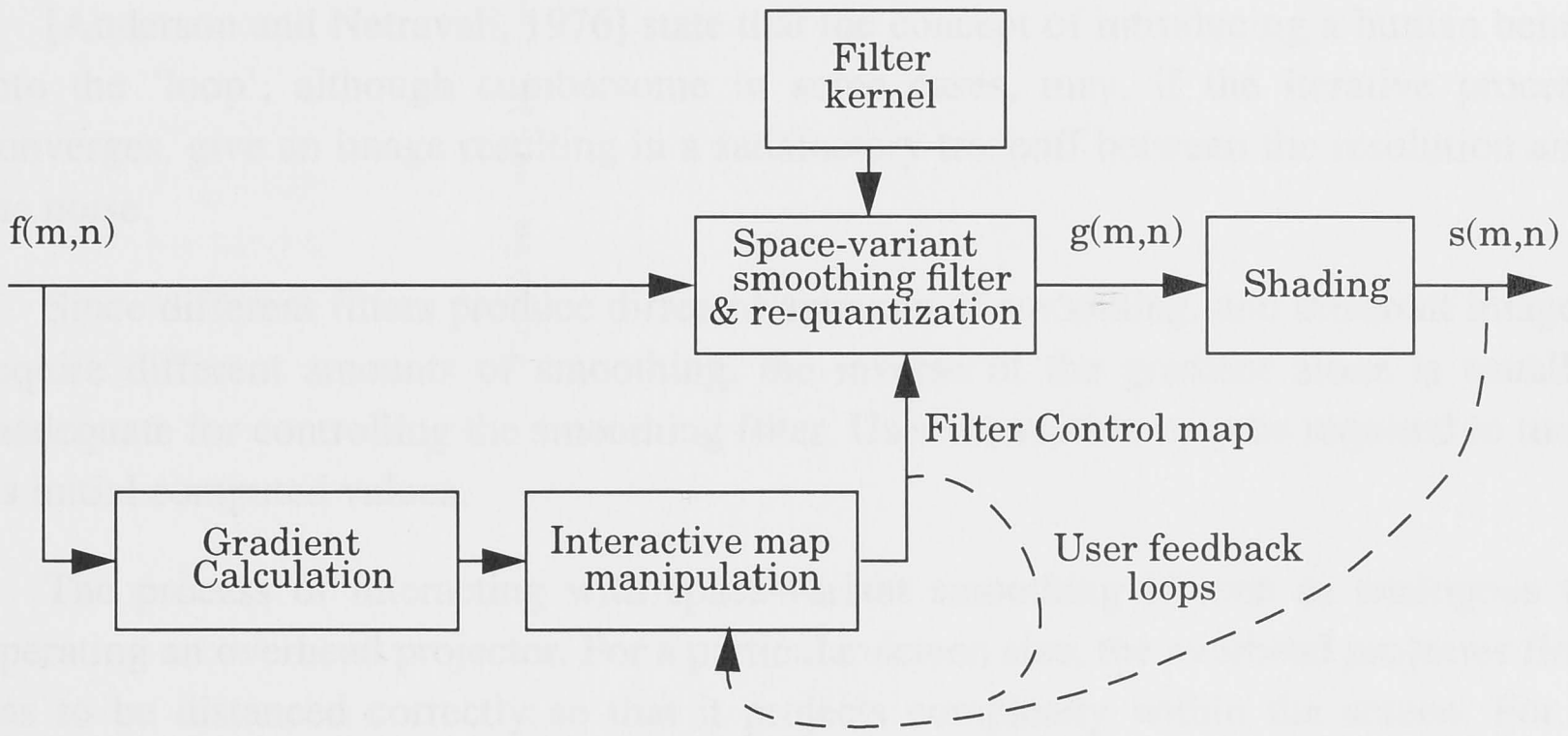


Figure 10 Model of space-variant interactive smoothing and re-quantization for reduction of noise and quantization artifacts for shading

To compute the gradient, the pixel difference method is used. This computes the row and column gradients by convolving two 3x3 gradient operators with the image. These operators are

$$columngradient = \begin{bmatrix} 0 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, rowgradient = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$$

The magnitude of the gradient vector is computed from the row and column components. Other gradient operators such as Roberts, Sobel, and Prewitt could also have been used to compute the gradient.

To achieve space-variant low-pass filtering, a scale value is computed from the control map values (derived from the gradient magnitude) at each pixel location, and used to scale the indices of the lookup table storing the filter (Section 3.4.1). This filter-scale value is computed by first normalizing the values in the control map (a byte image) between 0 and 1, and scaling them to a user-defined range.

$$FilterScale = \frac{MapValue}{255} \times MaxUserSize$$

The width of the scaled filter is *FilterScale*×*OriginalWidth*, and its values are *OriginalFilterValue* / *FilterScale*.

3.4.3 Filter interaction

[Anderson and Netravali, 1976] state that the concept of introducing a human being into the 'loop', although cumbersome in some cases, may, if the iterative process converges, give an image resulting in a satisfactory tradeoff between the resolution and the noise.

Since different filters produce different amounts of smoothing, and different images require different amounts of smoothing, the inverse of the gradient alone is usually inadequate for controlling the smoothing filter. User interaction may be required to tune its initial computed values.

The process of interacting with space-variant smoothing is seen as analogous to operating an overhead projector. For a particular screen size, the overhead projector first has to be distanced correctly so that it projects completely within the screen. For a particular transparency slide, the projector has to be focused so that the image of the slide is clear on the screen. The process of focusing usually involves moving the focus either side of the ideal by successively smaller amounts until the user is satisfied that the best focus has been obtained. The projector setup is then used for subsequent slides. If a different type of slide is used, further small adjustments may need to be made to tune up the setup.

Adjusting the space-variant smoothing for shading works in a similar fashion. For a particular filter kernel and a particular image, the smoothing is adjusted (distanced and focused) to produce a shaded image with the best subjective quality. Adjusting the smoothing by oscillating between over-smoothing and under-smoothing, the user will eventually find an appropriate level of compromise. For subsequent images with the same filter kernel, the same space-variant smoothing may be adequate; different images have different levels of quantization artifacts and thus may need more or less filtering.

Since different filters have different smoothing properties, when changing the filter, the smoothing parameters may require adjusting.

[Anderson and Netravali, 1976] also state that while our understanding of signal processing capabilities and limitations of the human visual system has increased, its complexity has made optimum restoration under a complete visual criterion impractical, if not impossible.

Creating an algorithm to determine the best space-variant smoothing for visual quality is hard because of the complexity of the human visual system. Providing the user with a means for controlling the smoothing, even if it is only to tune up an initial computed guess, is a valid solution. Important to the success of this is the method of interaction. It must provide enough feedback and a suitable manipulation paradigm to allow convergence to a good solution.

Three main forms of interaction are provided: global shifting, global scaling, and lookup table manipulation of the values in the filter control map.

The values for the filter control map are initially calculated as the inverse of the gradient magnitude of the image. These values provide a good initial guess, and provide the basis for the variation of smoothing over the image. Different filters provide different amounts and types of blurring. A B-spline filter blurs more than a piece-wise continuous cubic filter for example. For different filters, a different amount of filter scaling for each data point maybe required. Different shading angles and different data sets may also require different amounts of smoothing.

The global shifting and scaling interaction allows the smoothing to be adjusted in a simple manner. This type of manipulation is at the same level as the focusing adjustment for an overhead projector. The prime feedback for this type of manipulation is the final shaded image, though visualization of the filter map is useful.

The third form of manipulation, lookup table manipulation, allows more detailed control of the smoothing variation over the image. The gradient magnitude determines the initial amount of blurring per data point, but lookup table manipulation allows these individual values to be adjusted. If the user observes that only a few gradient values are responsible for most of the noise and quantization artifacts, then lookup table interaction will allow those values to be blurred, leaving other values untouched. The best method for using the lookup table manipulation is to interactively 'play' with it and observe the results in the shaded image, to interact with the smoothing on an intuitive level. Visualization of the histograms of both the image gradient and the filter map allowed more directed manipulations to be carried out. Figure 11 shows how lookup table manipulation is performed.

A prototype interface for controlling image smoothing using the three methods of control is shown in Figure 12. In the example a 2D sine function image is being space-

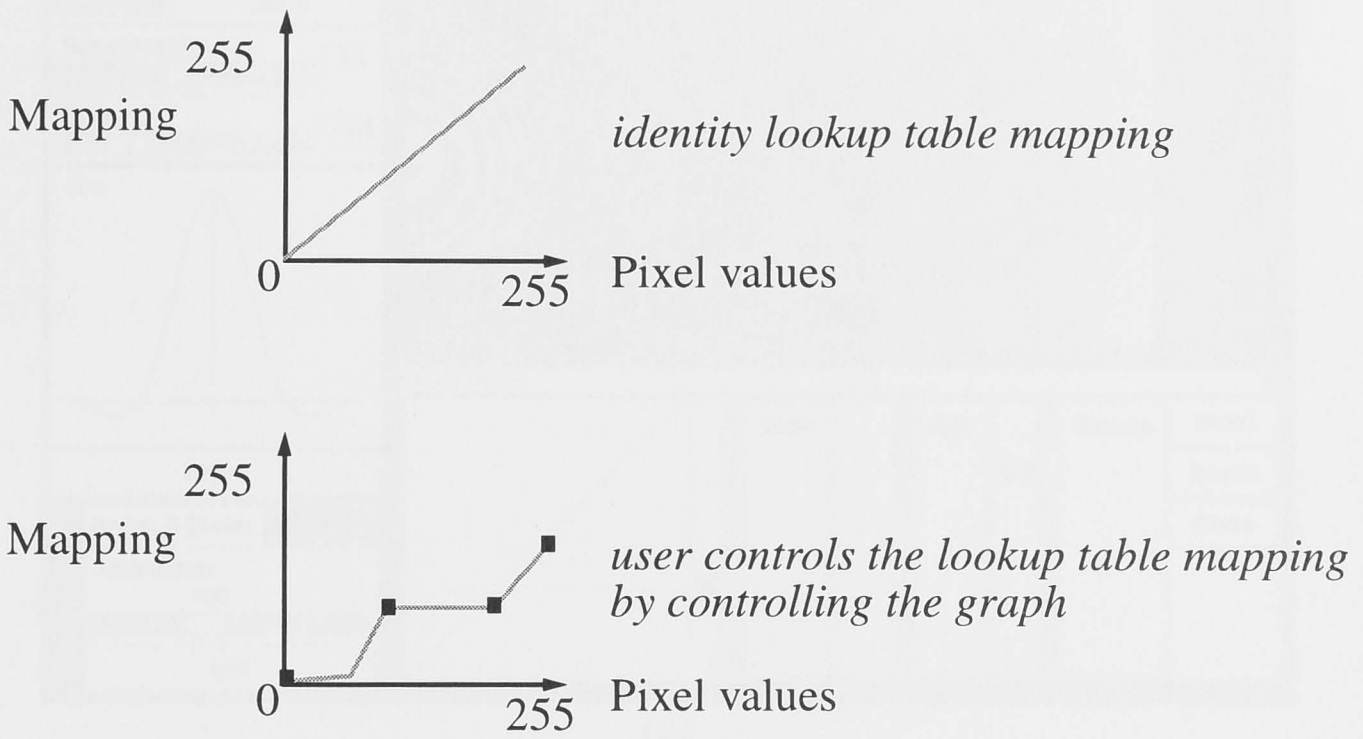
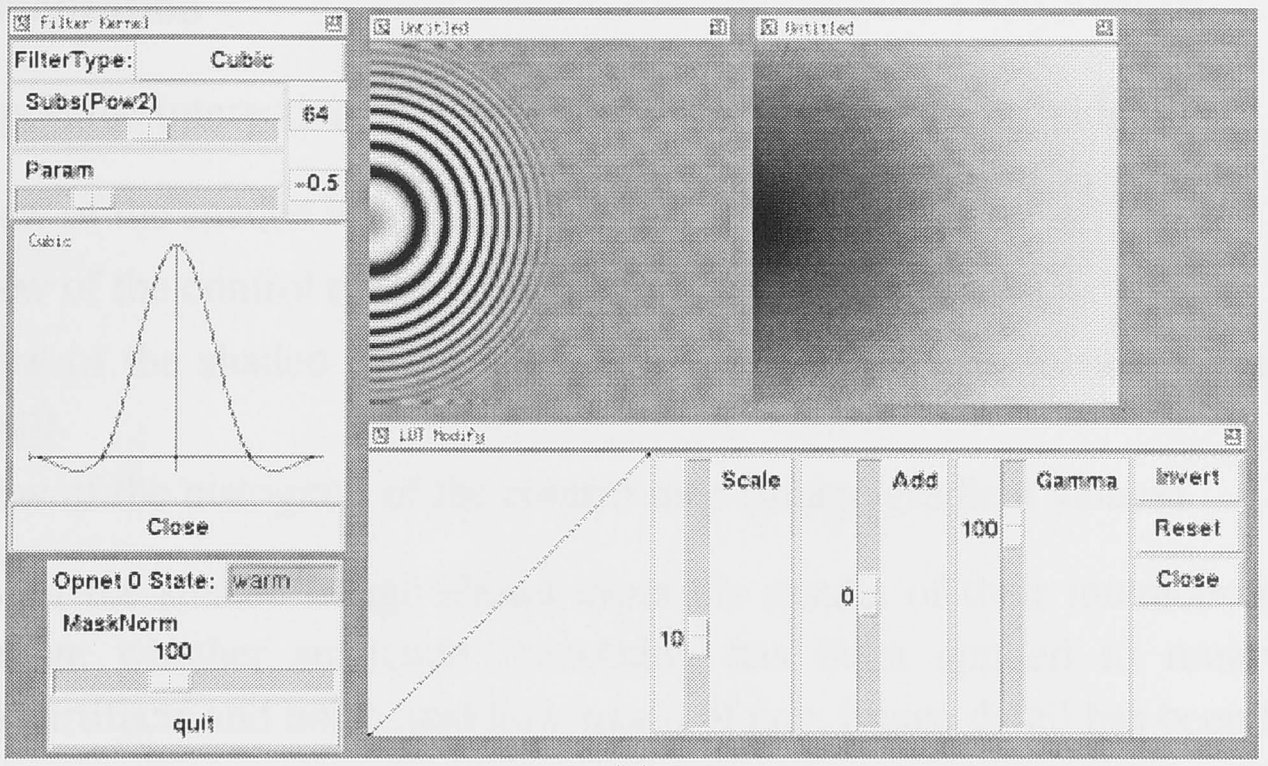


Figure 11 Lookup table manipulation - for 8-bit pixel values.

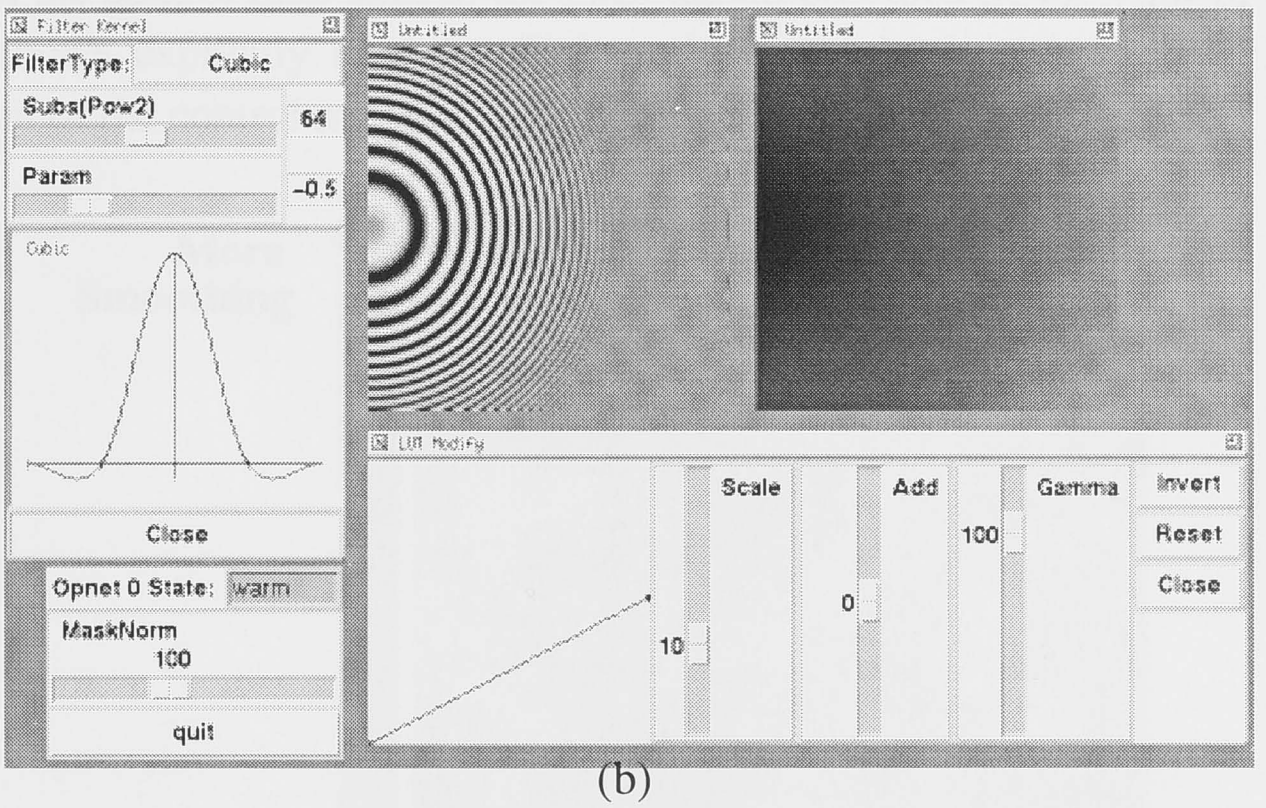
variantly smoothed, with the smoothing proportional to the frequency content; the amount of smoothing increases from left to right as the period of the sine function decreases. The filter kernel is first selected, then the amount of smoothing applied across the image is adjusted using the LUT interface. In Figure 12.(a) the sine image is heavily smoothed based on the pre-calculated filter variation (right image). In Figures 12.(b) and 12.(c) the amount of smoothing applied to the higher frequencies is reduced until quite strong moire patterns appear in the image due to aliasing. In this example the user is able to produce an image with good perceptual quality by finding the minimum amount of smoothing required to remove aliasing.

Interaction with the space-variant smoothing by itself provides a unique method for data exploration and understanding the effects of different filters. It allows the limitations of the space-variant smoothing and the fidelity of the final solution to be seen. Being able to see the raw shaded image gradually transform to the final image shows the user how different parts of the data are affected, which allows the quality to be understood and the extent to which choice of parameters can influence the perceived end result. This helps the user recognize the potential for introducing artifacts unknowingly.

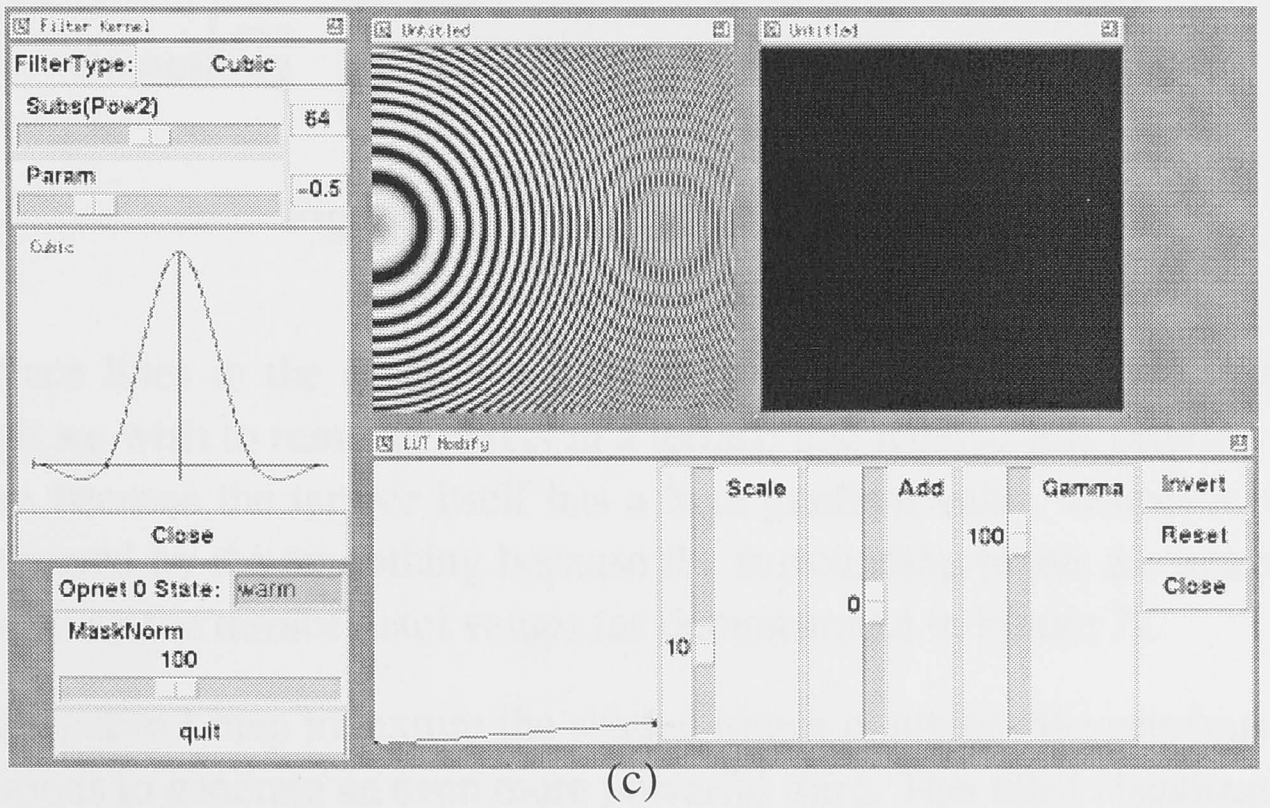
[Robertson, 1991] states that interaction is clearly a key aspect of the interpretation process, just as in the real world, where an observer continually interacts through eye, head, hand, and other movements.



(a)



(b)



(c)

Figure 12 Interface for controlling the image smoothing. The amount of smoothing applied is gradually being reduced for the higher frequencies (a) - (c)

3.4.4 Visualization

To support the interaction, four different visualizations are provided.

1. view of the shaded image
2. view of the control map; either false coloured or grayscale
3. view of the shaded image with the control map as its surface texture (texture map).
4. view of the histogram of the control map (image gradient values)

A view of the shaded image shows users the results of their interaction. It allows them to judge whether sufficient smoothing has been applied to remove all the quantization artifacts and noise, and how much, if any, image detail has been lost.

View of the control map adds to the information provided by the view of the shaded image. It shows explicitly how the filter varies over the image. Figure 13 shows a visualization of the control map used to filter the terrain image of Figure 5.

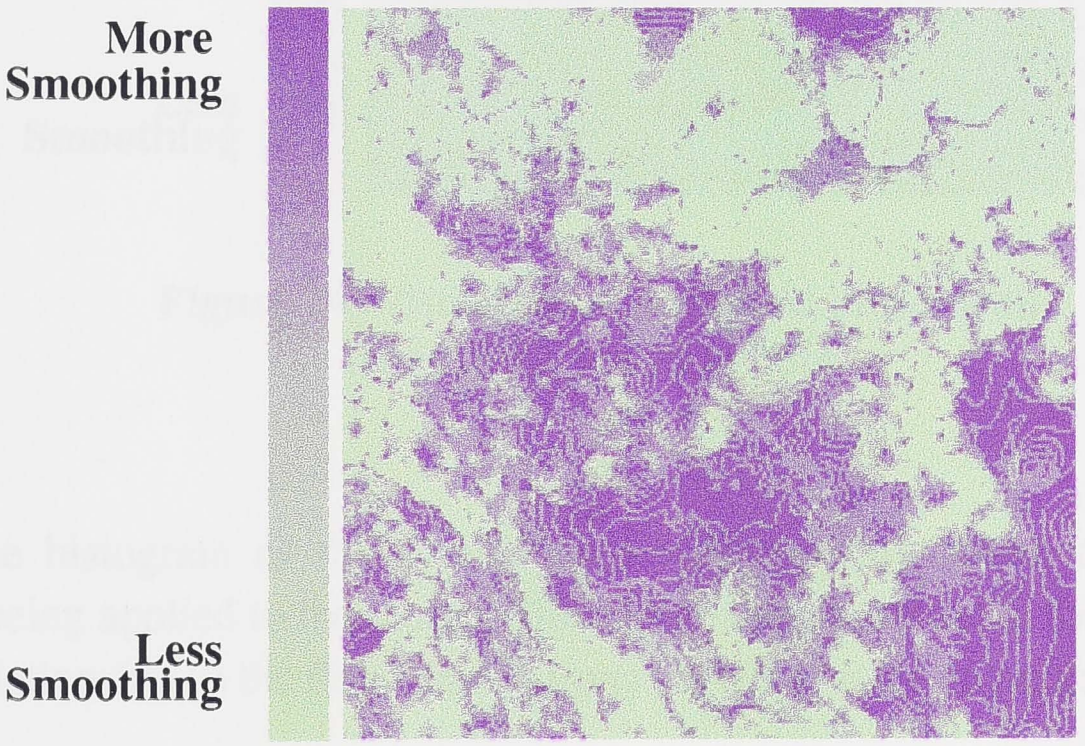


Figure 13 Visualization of a filter control map

The terrace lines in the control map correspond to the terrace lines in the shaded image which we wish to remove. Pixels in a terrace line receive less filtering than those around them because the terrace itself has a high gradient value. However, the terrace lines are removed by the smoothing because the surrounding pixels are heavily filtered and influenced by the terrace pixel values (as demonstrated in Figure 7).

Using the control map to texture the shaded image combines the previous two types of visualizations to generate an even more powerful third. This third visualization allows the user to view the filter variation directly over the shaded image, which simplifies the task of determining the level of filtering applied to each region in the image. When

interactively playing with the filter variation, this visualization explicitly shows how each type of interaction affects the variation over the image. This is especially important during the initial filter interaction for each image. However, because texturing distorts the shaded image, the final variation tuning appears best achieved using the plain view of the shaded image. Figure 14 shows an example of a shaded image textured with its filter control map. It shows how the least filtering is performed on the hills, and most in the smoother valleys.

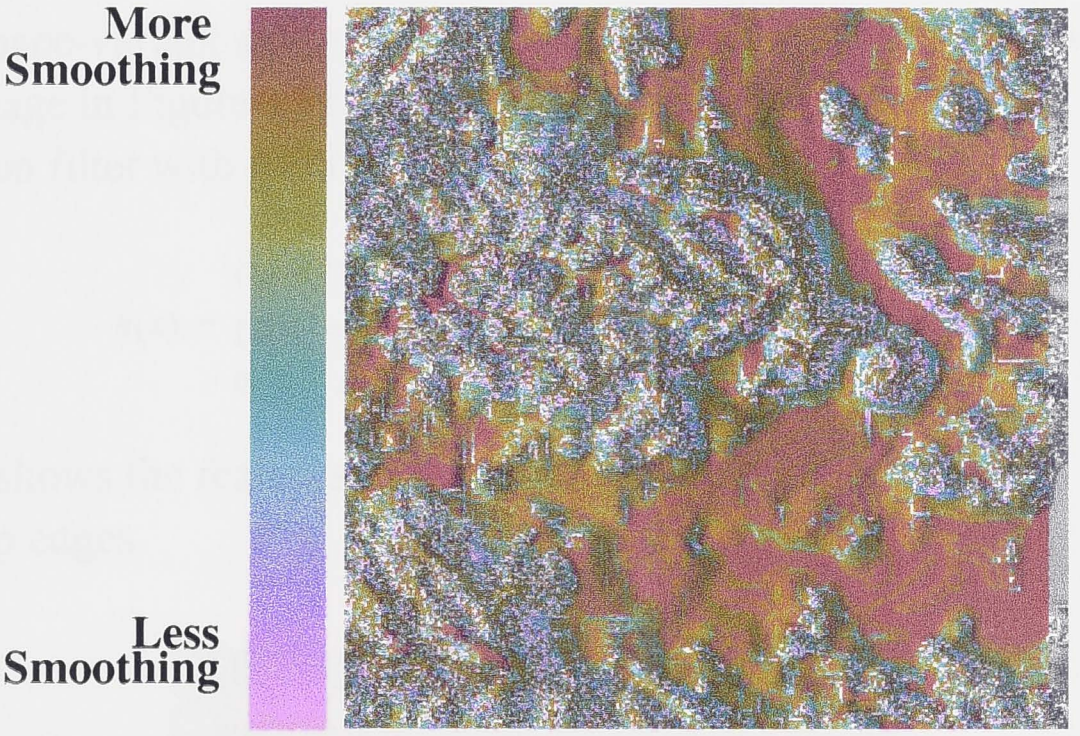


Figure 14 Visualization of shaded image with control map as its texture

Finally, the histogram of the control map indicates how much of each level of smoothing is being applied to the image. This type of visualization is for more advanced users, as its relationship to the final shaded image is not always immediately apparent.

Placing the user in a feedback loop to control the filter map, based on visualizations of the final shaded image and the filter control map, is seen as an essential part of the system.

3.5 Results

To achieve as close to interaction rates as possible, a massively data-parallel machine, the MasPar MP1 with framebuffer is used. For a 512x512 image, filtering using a base filter kernel of width four is performed in just under one second. The exact time depends on the amount of image smoothing. This is just sufficient for slow interaction. Faster filtering should improve interaction and increase understanding of the data.

Interactive smoothing allows the filtering operation to be better understood because it lets the user see exactly what is happening to the original data set. When interacting, visual feedback from the animated shaded image is found to be the best source of information for control of the interaction, though visualization of the filter control map helps show the exact variation.

Using the space-variant smoothing technique for shading, for the image shaded in Figure 5, the image in Figure 15 has been interactively produced using a one-parameter cubic convolution filter with parameter value of -0.5.

$$h(x) = \begin{cases} ((a+2)|x|^3 + (a+2)|x|^2 + 1) & \text{if}(|x| < 1) \\ (a|x|^3 + -5a|x|^2 + 8a|x| - 4a) & \text{if}(1 \le |x| < 2) \\ 0 \quad otherwise & \end{cases}$$

This image shows the features of the terrain much more clearly, and yet maintains reasonably sharp edges.

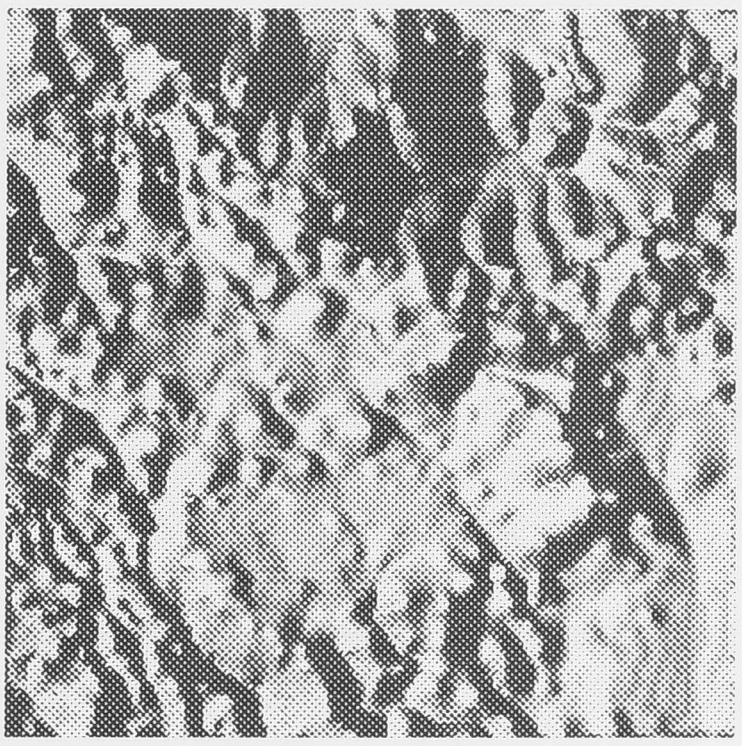


Figure 15 Shaded 8 bit height image pre-smoothed with an interactive space-variant filter

A uniformly smoothed image, shown in Figure 16 allows a comparison with existing techniques. It also has noise and quantization artifacts removed, but there is noticeably less detail in mountainous regions.

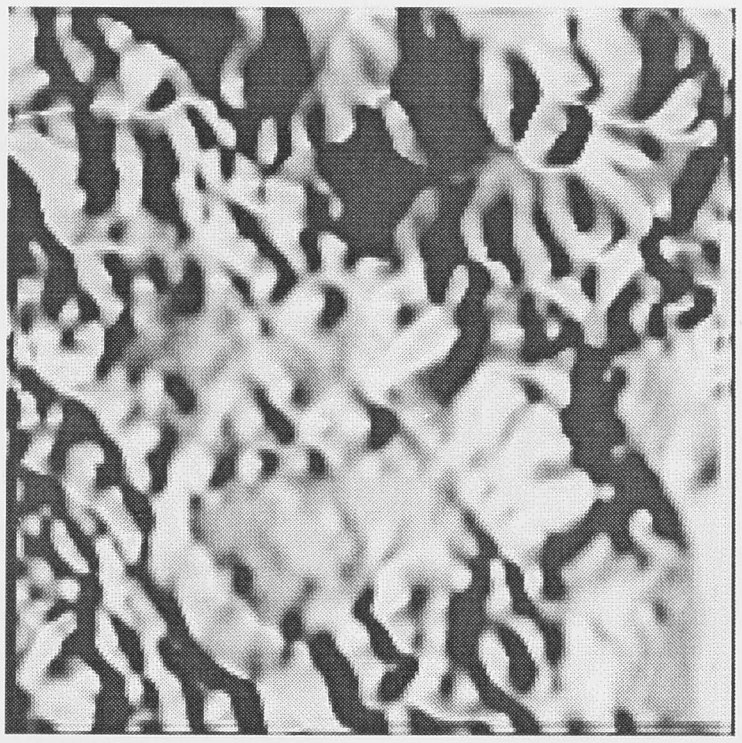
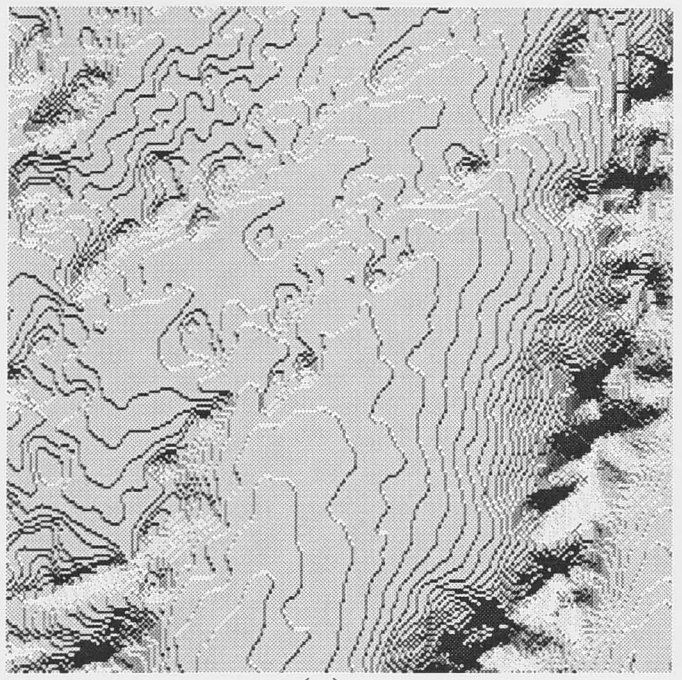


Figure 16 Shaded 8 bit height image pre-smoothed with a space-invariant filter

Figures 17 and 18 show two more examples of shaded images that have been interactively pre-smoothed by a space-variant filter. Both give a comparison with a uniformly smoothed image to highlight the improvement.

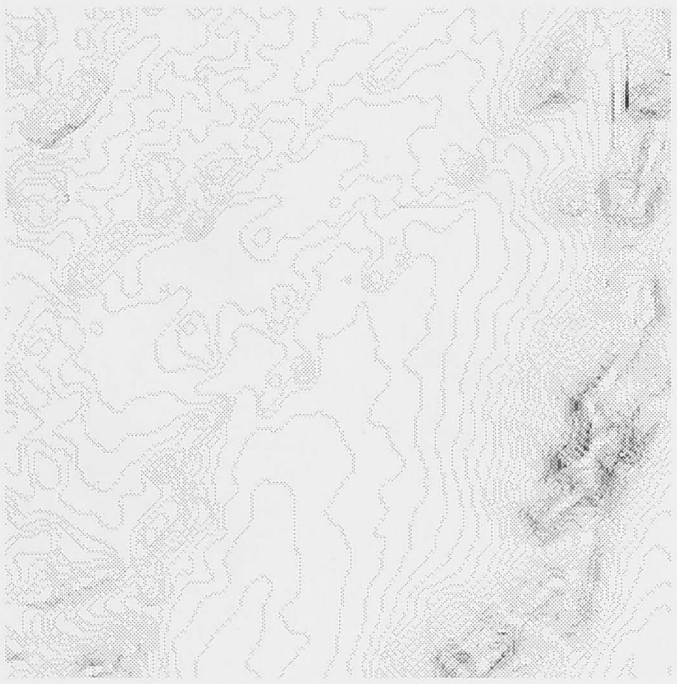
From the examples, it can be seen that the degree to which an image is improved by interactive space-variant filtering depends on the image's frequency content. The first example shows the greatest subjective improvement because it contains the most area of mountainous regions containing high frequencies. The second example has fewer mountainous regions, but still has a definite improvement of quality over the uniformly smoothed image because it preserves a visible amount of high frequencies. The last example has the least improvement. While the image does contain mountains, these have only a small number of high frequencies (they are smooth to begin with) so the uniform smoothing filter does a comparable job for improving image quality.

When interactively adjusting the image smoothing, we find that most of the interaction is directed at reducing the smoothing applied to mountainous regions. Global adjustment is first used to increase the level of filtering so that the terracing artifacts are smoothed out, then the LUT manipulator used to reduce the levels of filtering applied to mountainous regions. With the LUT interface, one or two extra control points are usually adequate for control.



(a)

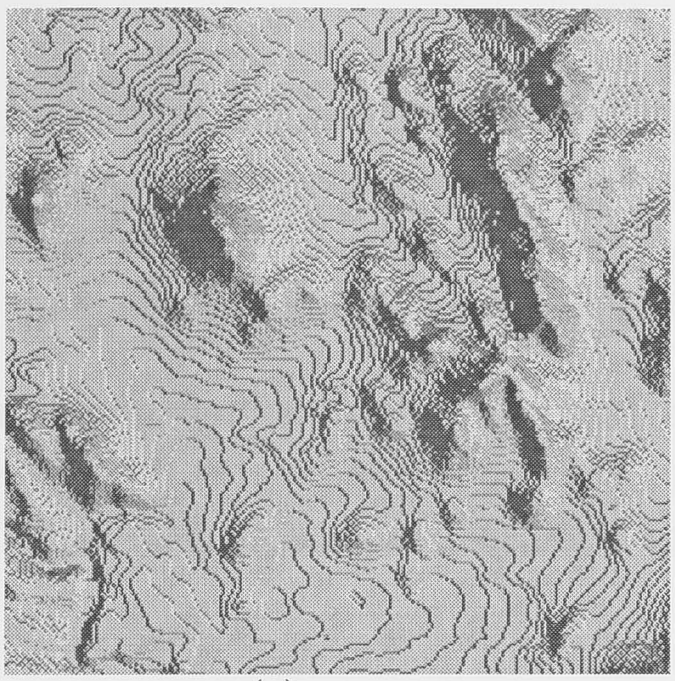
(b)



(c)

(d)

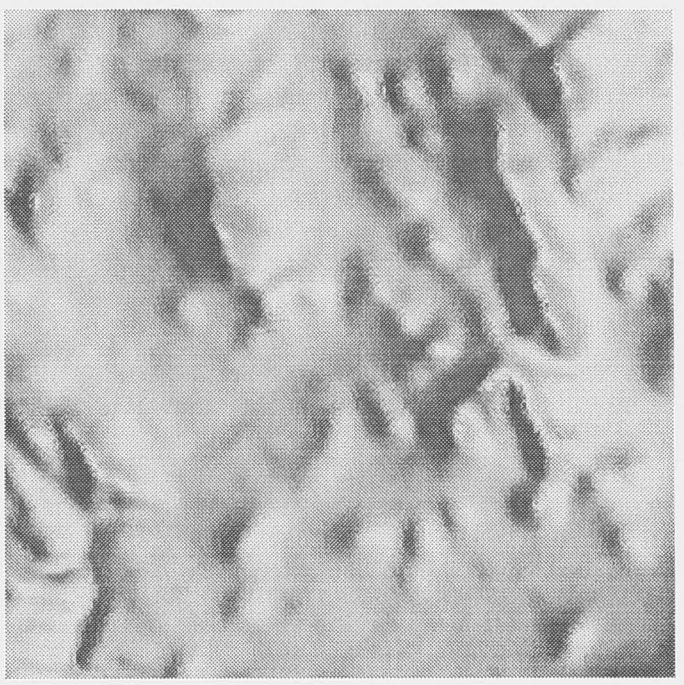
Figure 17 Shade image example 2. (a) Shade without smoothing, (b) space-invariant smoothing, (c) space-variant smoothing mask, (d) shaded image with space-variant smoothing



(a)

(b)

(c)



(d)

Figure 18 Shade image example 3. (a) Shade without smoothing, (b) space-invariant smoothing, (c) space-variant smoothing mask, (d) shaded image with space-variant smoothing

3.6 Summary

A space-variant filtering technique has been presented for smoothing noise and quantization artifacts in relief shaded digital terrain images. It produces shaded images with good visual quality by balancing noise removal with image smoothing through user interaction. The user interaction allows the user to converge onto a good visual solution and to obtain an intuitive understanding of both the data and the filter.

The interaction methods developed for filter control are general enough to be applied to a variety of other problems; problems where the spatial variation of the filter can be pre-calculated, but whose level of filtering needs to be balanced by the user.

4 INTERACTIVE SPACE-VARIANT FILTERING FOR IMPROVING SEISMIC DATA

Seismic data processing is a highly subjective task because of the lack of verifiable and certifiable assumptions. Traditionally human experts have been used to determine processing parameters, with the subsequent problem that interpretations can vary significantly between different experts.

Seismic exploration has three major stages: data acquisition, data processing, and data interpretation. During acquisition the data can be distorted or become noisy. The data processing stage attempts to correct these distortions and remove noise so that the final interpretation can extract as much useful information as possible.

This chapter examines the traditional non-interactive methods for removing noise by performing a space-variant band-pass filtering step on seismic data. It proposes a number of improved methods based on direct user interaction. To enable experimentation, a space-variant band-pass filter is developed which is capable of being controlled interactively. Several dynamic visualization methods are also designed. To introduce the problem, different data acquisition methods are discussed together with some standard methods for processing the data they obtain.

4.1 Seismic data

4.1.1 Acquisition

Seismic data are constructed from examining and collating echoes in the ground from man-made sources. The basic geometry for a single shot has one acoustic energy source (sender) and many listening devices (receivers), which are evenly spaced in a line as shown in Figure 19.

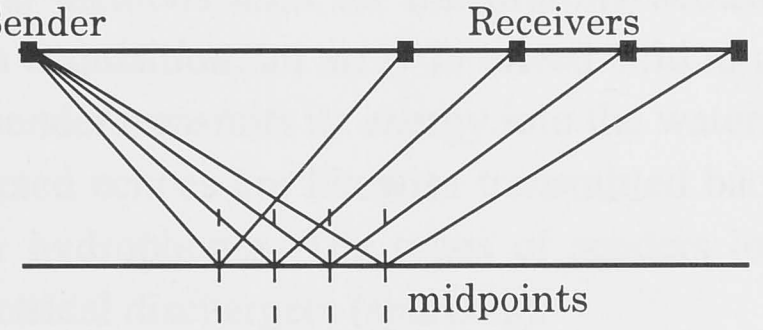
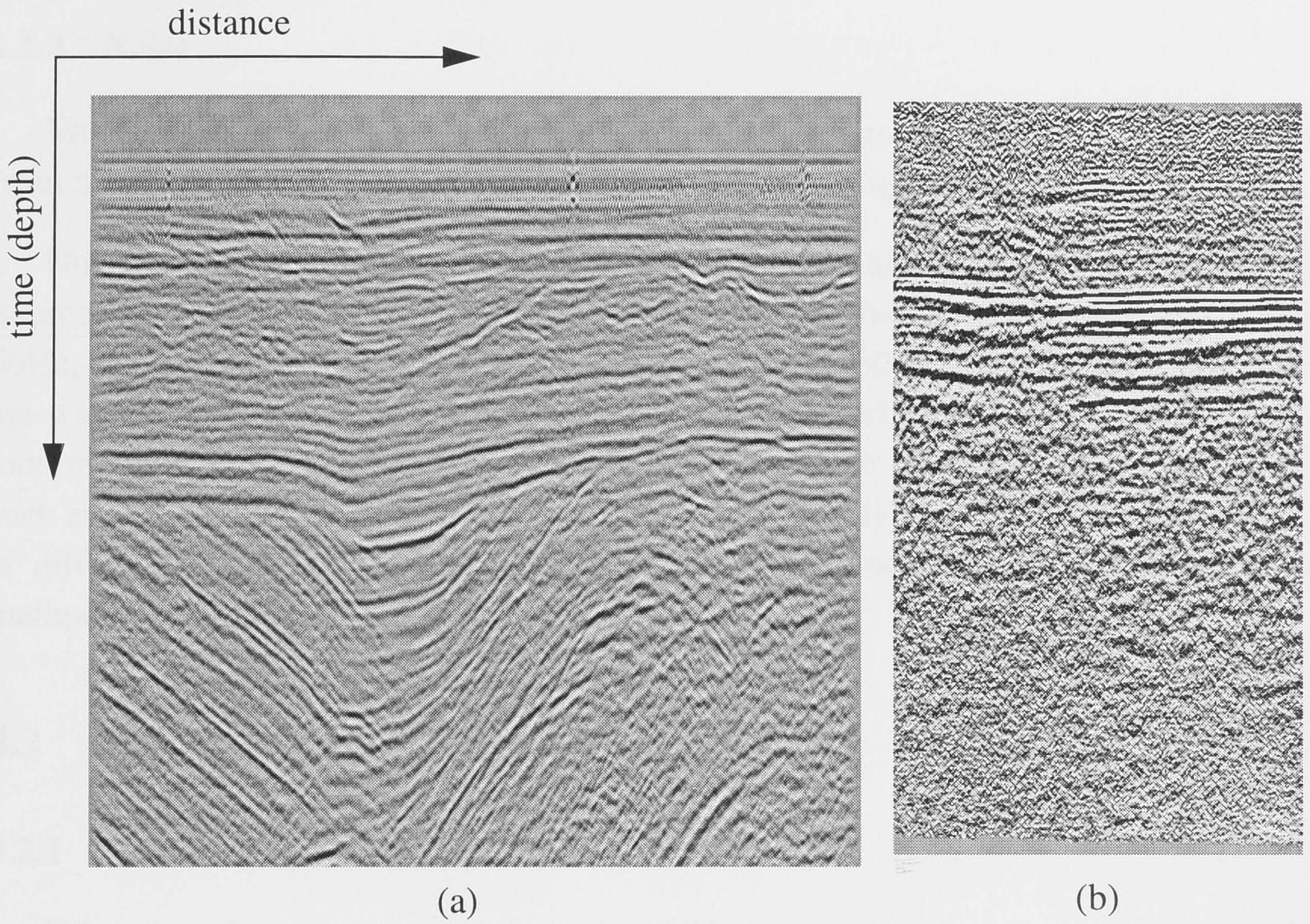


Figure 19 Sender and receiver positions showing a single reflection plane

By moving the line a whole set of signals can be obtained and collated to form a 2D slice of the ground beneath, with "distance" on the horizontal axis and "time" on the vertical axis (time is used instead of distance due to the uncertainty of knowing the

velocity of sound through the different layers of rock). See Figure 20 for two examples of fully processed 2D slices.



(a) (b)

Figure 20 2D seismic slices. (a) marine, (b) land

Ideally the acoustic energy source should create a single impulse spike to allow the geology to be clearly imaged. Since a single impulse is impossible to obtain in practice, a number of different signals are used which only approximate a spike. A deconvolution operator (described later) is used in the processing stage to collapse the actual signal wavelet to obtain results as if an impulse spike had been used.

There are several methods used for transmitting acoustic energy into the earth. In marine seismic data acquisition, an array is towed behind a boat containing the sender and receivers. The sender transmits its energy into the water, which then transmits it into the earth. The reflected echoes are likewise transmitted back into the water where they can be recorded by hydrophones. The types of senders include compressed air guns, water guns, and electrical dischargers (sparkers).

In land-based seismic exploration senders and receivers have to be hand-placed in the ground, often in drill holes to get to the bedrock beneath the top soil. Acoustic energy can be created by explosives, or by ground vibration (pounding the ground with a machine similar to a pile driver). The echoes are recorded by geophones.

A detailed description of these methods is beyond the scope of this chapter. See [Hatton et al., 1986] and [Yilmaz, 1987] for further explanation.

4.1.2 Noise

Much of the processing of seismic data is aimed at removing noise. Noise can come from many sources including multiple reflections of the original source signal.

Noise can be divided into "coherent" and "random" ambient noise for processing. Sources of coherent noise include powerlines (marine-based and land-based) at 50-60Hz, cable movement through water (marine), ground roll, and guided waves in either the water or geology. Random noise can come from poorly mounted geophones, wind motion, wave motion in the water, instrument noise, other boats, and heavy machinery such as compressors in seismic vessels. For most of these types of noise, the noise signal is difficult to measure or calculate. Therefore data processing must use empirical methods to remove it.

4.2 Data processing

4.2.1 General processing steps

When raw data are returned from the field, it must be processed to ascertain the information returned. In this section the major processing steps are briefly mentioned.

The initial pre-processing includes demultiplexing, reformatting, editing of the data files, gain recovery, and some coherent noise rejection. Gain recovery aims to restore the lost signal strength by scaling. As the source signal penetrates deeper into the earth, its strength, and thus the strength of its reflections decreases. This decrease is primarily due to energy lost by reflections at various layers, signal spread, and energy being absorbed by the rock and liquid pockets. Gain recovery restores this loss in signal strength by scaling the data values. In the ideal case it scales the data so that the signal strength is uniform across the image. Unfortunately, since signal attenuation can not be known precisely, a suitably chosen function must approximate. Typically an exponential signal strength decay is assumed.

The major data processing steps are deconvolution, stacking, and migration [Yilmaz, 1987]. Each stage itself can take many processing steps to complete, and these vary depending on the source of the data and the geophysicist processing it.

The deconvolution stage is designed to increase temporal resolution of the data by collapsing the original signal wavelet into an impulse spike. Weiner filters are commonly used for this task and require that the original source signal wavelet be known. This is usually recorded or taken from the known characteristics of the source device.

Up to and including the deconvolution stage, the data being processes is still the 1D signals (trace) taken from the receivers. Because of the spacing chosen for the receivers, there is redundant information between consecutive shots. The midpoint between a source and receiver is the position where the signal received by the receiver is reflected from the source. Between consecutive shots there may be many source and receiver pairs with the same midpoint position, see Figure 19. Common mid point (CMP) gathering collects together the traces with the same midpoint. A normal move out (NMO) operation normalizes each receiver's trace to appear as if it were taken coincident with the source. Then after some amplitude scaling, the traces in the CMP gather can be stacked together, or statistically averaged. This averaging has the effect of improving the S/N ratio by $\sqrt{N}$, where N is the number of traces in the CMP gather.

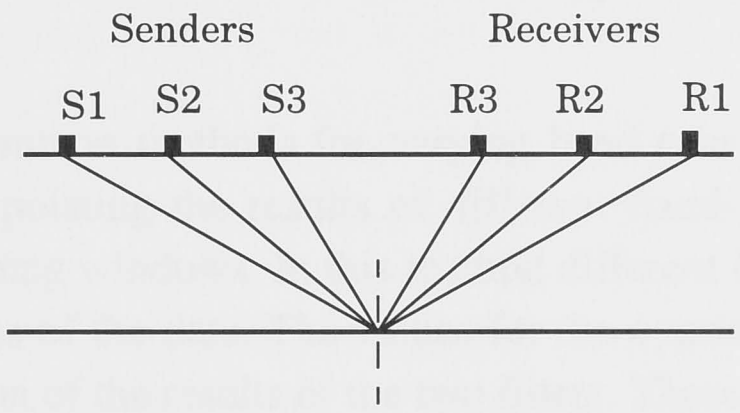


Figure 21 Common midpoint positions

Finally, migration collapses refractions. Before migration and after stacking, a "time-varying" band-pass filtering step (the filter varies along the time axis) is usually performed to remove noise. It is this space-variant filtering, which is described in the next section, which we will focus on in greater detail.

4.2.2 Space-variant (time-variant) band-pass filtering

Why filter

The band-pass filtering step attempts to clean up the trace data by removing noise. In the spectrum of the data there are usually some bands where the signal-to-noise (S/N) ratio is high and other bands where it is low. By removing those bands where the S/N ratio is low the overall S/N ratio of the data can be improved. However, some of the original signal is lost in this process; there is always a balance between noise removal and signal loss. In most seismic data the S/N ratio is low at very high frequencies due to ambient noise, and at low frequencies due to ground roll. Thus a band-pass filter is appropriate.

As the source signal penetrates the earth, its high frequencies become attenuated more rapidly than its low frequencies. Therefore the usable portion of the signal and its

S/N ratio shifts to lower frequencies with depth (time). The band-pass filter to improve the S/N ratio should also change with depth of the signal for the best results.

How

The selection of band-pass filters for removing noise in seismic data is a subjective process. Geophysicists use their knowledge and experience to select filters to achieve the best balance between noise removal and signal preservation.

The most common method uses a series of pre-filtered panels to show the geophysicist the frequency content of the data. From examination of various frequency bands, they can determine where noise starts to dominate the signal and design appropriate band-pass filters. These filters can be applied, evaluated, and redesigned as required.

The two most common methods for varying band-pass filtering with depth (time) rely on linearly interpolating the results of different fixed-bandwidth filters. The first method uses overlapping windows. In this method different band-pass filters are applied to overlapping regions of the data. The values for the overlapped regions are computed as a linear combination of the results of the two filters. These merge zones can be chosen in relatively unimportant regions of the data.

The second method uses a continual merging of windows so that each point is a combination of several filters. Figure 22 shows both the overlapping windows (a), and continuous merging (b) methods.

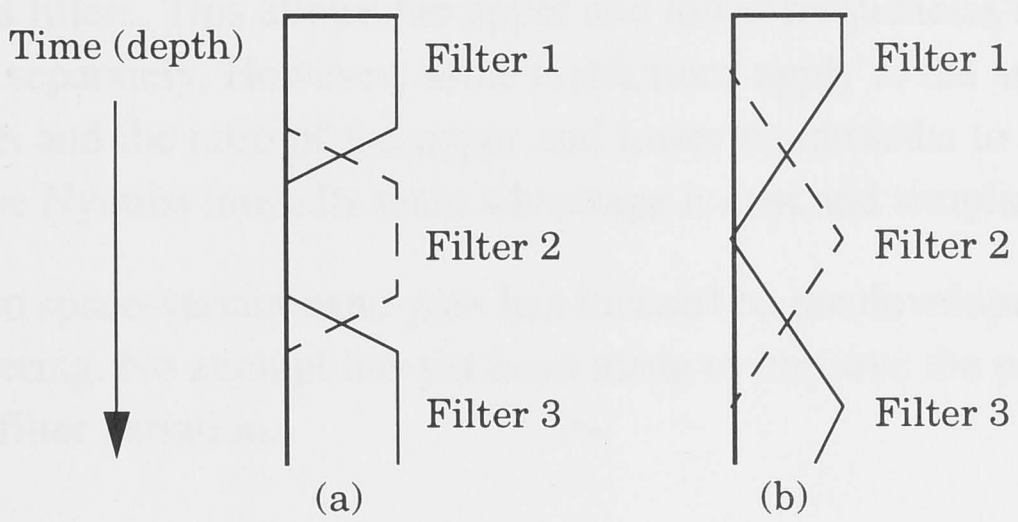


Figure 22 Methods for time-varying band-pass filtering. (a) Overlapping window (b) continuous merging

Recent work has developed improved methods for creating continuously variable band-pass filters.

[Pann and Shin, 1976] have developed a class of convolutional space-variant filters. These are conceptually implemented by a pre-application of a space-variant frequency

shifting operator before a standard space-invariant band-pass filter, followed by a frequency restoration operator. This approach has two main limitations: its bandwidth is fixed, and it only supports a linear variation in filter frequency.

[Stein and Bartley, 1983] use a recursive continuously variable Butterworth filter, whose coefficients are calculated at key points and linearly interpolated at intermediate points to improve efficiency. Band-pass filtering is achieved either through a cascade of low-pass and high-pass Butterworth filters, or by a direct band-pass filter. The direct method requires fewer coefficients to be interpolated and has better filter characteristics, however, the cascade approach gives greater flexibility for varying the characteristics of the filter. The filter is limited to linear variation, which is its major limitation. Some errors are also introduced by the interpolation of filter coefficients.

[Shon and Yamamoto, 1992] use a similar idea to that used by [Pann and Shin, 1976]. They apply a complex operator that has the effect of first shifting the signal's frequency spectrum towards the negative axis and multiplying it by the unit step function to remove the lower frequencies (a high-pass filter). A further shift and then multiplication by the inverse step function removes the lower frequencies (low-pass filter), and finally shifting back to the original frequency location. This filter has a boxcar like frequency spectrum and can be varied continuously.

[Park and Black, 1995] implement a space-variant band-pass filter by scaling a reference spatial filter kernel. Scaling the width and height of the kernel has the effect of scaling and shifting the band-pass filter's cutoff frequencies. The bandwidth of this filter is a constant in octaves, with the upper and lower cutoff frequencies dependent on one another. Park and Black increase the flexibility of the filter by applying a cascade of two scaled band-pass filters. This allows the upper and lower frequencies (or the bandwidth) to be controlled separately. However, some restrictions apply to the amount of variation of the bandwidth and the ratio of the upper and lower bandwidths to prevent folding of frequencies at the Nyquist limit. Its main advantage is cost and simplicity.

Research into space-variant band-pass has focused on the development of new filters and types of filtering. No attempt has yet been made to improve the process of selecting and controlling filter variation.

4.3 Interactive band-pass filtering

One problem with the filter panel based method for selecting and specifying band-pass filters is that it is a static process. The filter panels are pre-computed images that show fixed information, and the process of selecting filters although iterative, is non-interactive. A dynamic method could provide a more powerful way for selecting band-pass filters and learning about the signal and noise content of the data. Thus the aim is to develop an interactive band-pass filtering method that solves these problems.

There are four areas of space-variant band-pass filtering that need to be addressed:

1. the type and form of the space-variant band-pass filter
2. the type and form of the filter variation
3. the filter interaction methods
4. the visualization required to understand and control filter variation.

4.3.1 Space-variant band-pass filtering

The requirements for the band-pass filtering are

- continuously variable with independent control of upper and lower frequencies
- fast enough to support interaction
- sufficient quality not to introduce artifacts which could affect data interpretation
- supports a variety of different filter types.

Since none of the filtering methods reviewed in the literature satisfies these requirements sufficiently for our purposes, a new approach has been developed. A key feature of this approach is that the filter only varies in one direction, and thus the same filter kernel can be used for an entire row of the data; the variation of the filter over the data is shown in Figure 23. Because the data sizes are usually large, the overhead of directly computing the filter kernel once for each row is small compared to the cost of filtering a row of the image.

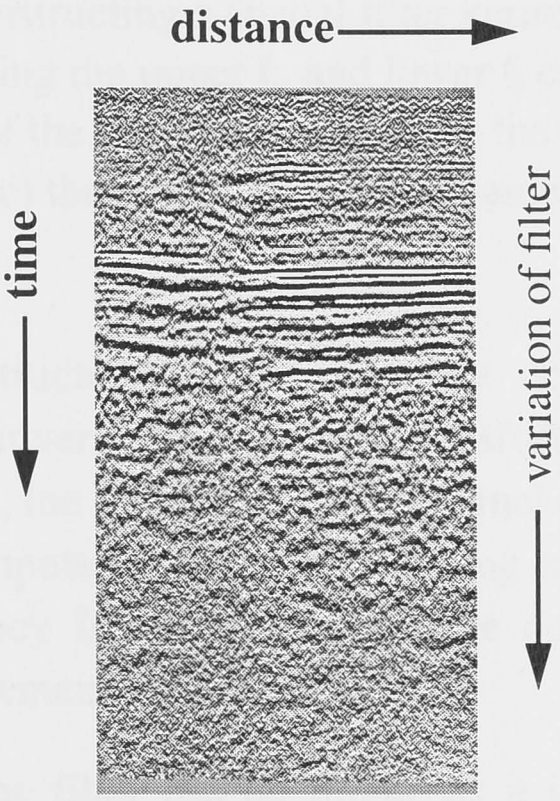


Figure 23 Variation of band-pass filter over seismic data. Data values with the same "time" will have the same band-pass filtering

The steps for the direct filtering method are:

1. the general frequency characteristics of the filter are specified by the user in terms of transition region size and/or slope

2. for each step in the time axis, the convolutional spatial filter kernel is generated by constructing the required filter in the frequency domain and inverse Fourier transforming it (this process is shown in Figure 24)

3. the filter kernel is applied in the column direction for each value in the current row. The spatial filtering is performed using a look-up table approach similar to the one described in [Feibush et al., 1980] [Ward and Cok, 1989].

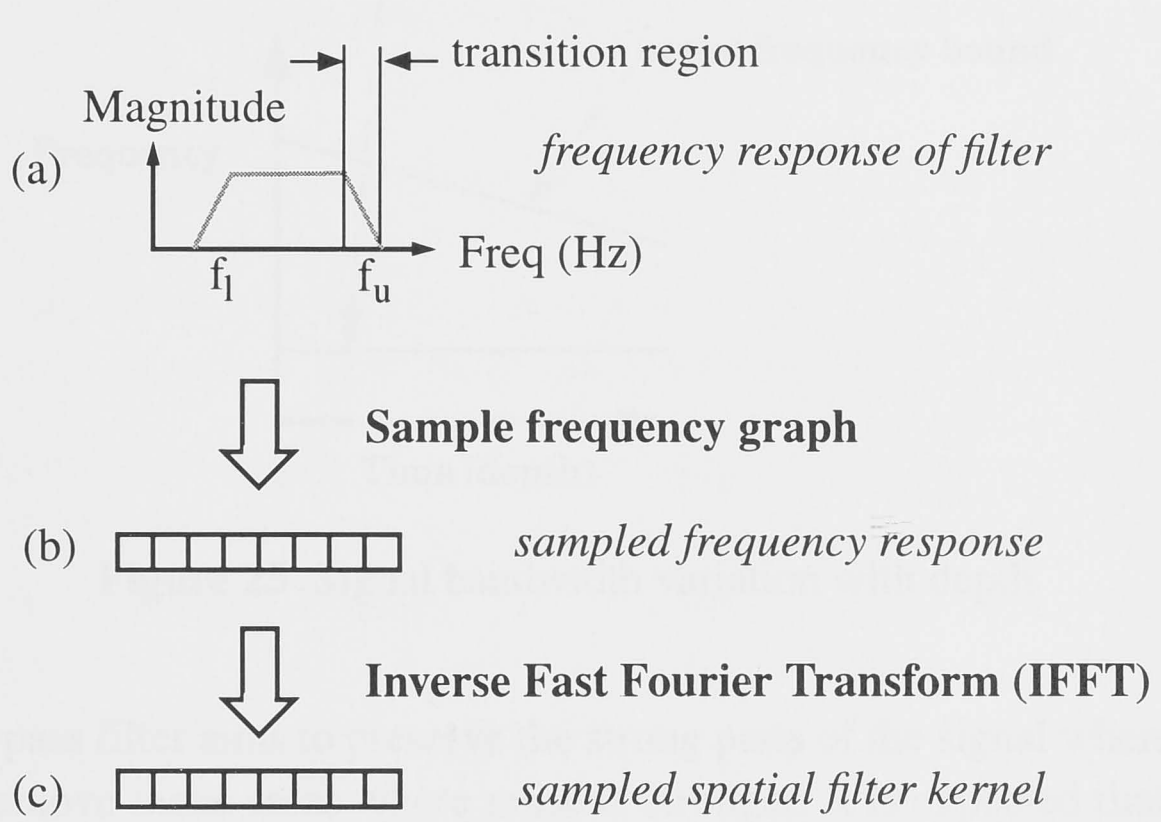


Figure 24 Constructing a spatial filter kernel, (a) the frequency response is constructed using the upper f_u and lower f_l cutoff frequencies together with the properties of the transition region, (b) the frequency response is sampled, (c) the spatial kernel is obtained using the IFFT.

The filter is constructed in the frequency domain as a discrete data set, and transformed using the inverse Fast Fourier Transform (FFT). By changing the sampling density of this function, the size of the spatial kernel can be lengthened or shortened. The filter kernel can be computed at different sampling resolutions by changing the sampling density of the frequency filter. This allows the cost vs. accuracy of the filter to be balanced for the requirements of the problem.

The variation of the filter has no limits, as it is computed for each location. The overhead of computing the filter does not affect the performance significantly – the filter is significantly faster when tested against the operator scaling method of Park and Black.

The algorithm is also highly parallelizable. Different processors can be used to calculated the filter kernel for each row and perform its filtering.

4.3.2 Filter variation

Once the geophysicist has a good idea of the signal/noise properties of the data, they then have to select the band-pass filters for removing noise. As previously discussed, the S/N ratio in different frequency bands changes with depth (time) since the higher frequencies of the source signal are attenuated more rapidly. The original signal's frequency will thus form a wedge shape similar to that shown in Figure 25.

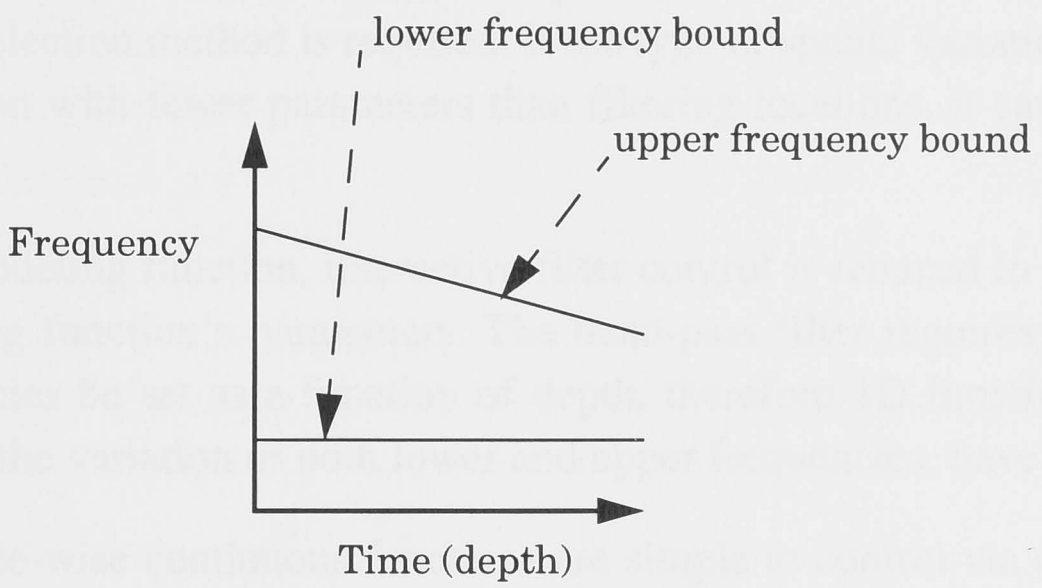


Figure 25 Signal bandwidth variation with depth

The band-pass filter aims to preserve the strong parts of the signal where noise is less evident and remove those areas where noise is stronger. It is expected that the signal is strongest in the middle frequencies (also since noise tends to be in either the high or lower frequencies). The change in the band-pass filter's frequencies would possibly look like that shown in Figure 26.

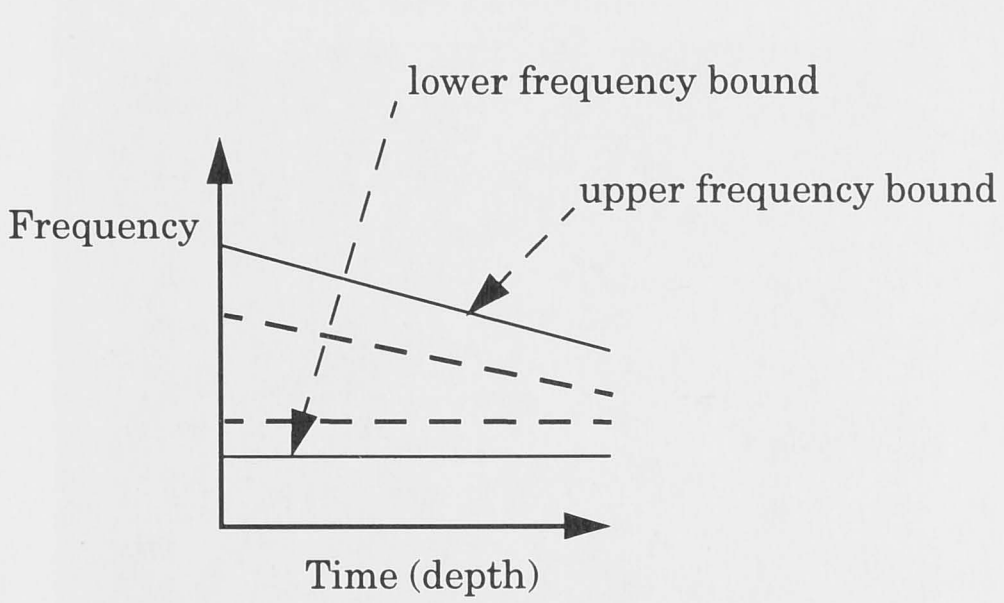


Figure 26 Signal and filter bandwidth variation with depth

4.3.3 Filter interaction

The goal of interactive filter selection is to give the geophysicist direct control over the filter's parameters. With spatially invariant filters, the geophysicist can adjust the filter's global parameters while viewing the effects of interaction. The interactive feedback loop promotes fast interpretation of filter properties, and the effect of each parameter.

The problem becomes more complex when the filter has space-variant behaviour such as for the band-pass filtering. The user has to select the filter for every location. This is impractical since most data sets require greater than a thousand locations, and thus indirect selection method is required. If the type of spatial variation can be modeled using a function with fewer parameters than filtering locations, it can provide a means for control.

Using a modeling function, interactive filter control is reduced to interactive control of the modeling function's parameters. The band-pass filter requires that its upper and lower frequencies be set as a function of depth, therefore 1D functions, which can be used to model the variation of both lower and upper frequencies, have been explored.

Linear piece-wise continuous functions are simple to control via control points, and can be used to produce a variation similar to traditional systems. Spline functions provide greater flexibility than linear functions, and are continuous over the first and second derivatives.

Both these user controllable functions have been embedded into an OpenInventor™ ExaminerViewer structure. The interface for manipulating the upper and lower frequencies using the piecewise linear function is shown in Figure 27.

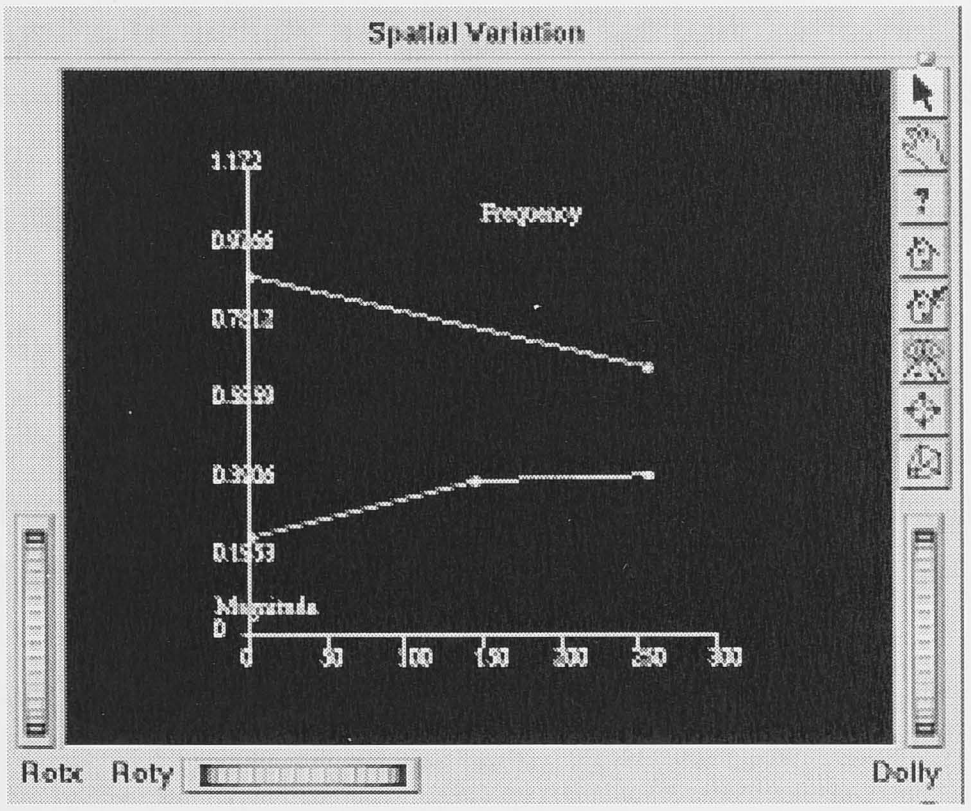


Figure 27 User interface for piecewise-linear function

The upper and lower functions control the upper and lower frequencies respectively. Along the horizontal axis is the variation of the function over the depth axis of the seismic image. Along the vertical axis is the cutoff frequencies of the filter.

The 1D functions are controlled by adding and moving control points along their lengths using the mouse. The interface restricts the movement of the control points to ensure the upper cutoff frequency is always greater than the lower cutoff frequency. Control points are limited to move between their adjacent neighbours. The control points at either end are only free to move in the vertical direction.

4.3.4 Visualization

The display of pre-filtered panels of data aims to give the geophysicist an empirical feel for the data. If enough information is provided, they can make a good judgement about which areas do and do not have good signal-to-noise characteristics.

A typical display may contain panels showing a 10Hz range of the spectrum from 0Hz to the Nyquist limit; producing 10 or more panels. Visual examination of these panels can be time consuming since differentiating between noise and signal is often difficult.

Dynamic display of data can make hard to see features more visible, fuzzy objects appear sharper, and generally allow faster interpretation of the data [Gershon, 1992], [Gershon, 1994]. There are a number of dynamic methods which can be employed effectively in this problem which will maintain the information in a form with which the geophysicist is familiar.

A frequency sliding panel is one form of dynamic display that can be applied to this problem. Following the same lines as the filtered panels, a single panel is provided that can be moved along the frequency axis. Moving the panel allows the whole frequency spectrum to be examined in the same manner as the fixed filtered panels, but the dynamic change of the data, as the panel is moved, increases the power of the visualization. Features jump into and out of focus as the frequencies covered by the panel change, allowing the geophysicist to gain a feel for the data's properties. Figure 28 shows graphically the sliding frequency window paradigm.

By allowing either the lower or upper frequency to be fixed, and the other to be moved, the panel has a variable frequency width which adds an extra dimension to the dynamic behaviour of the display.

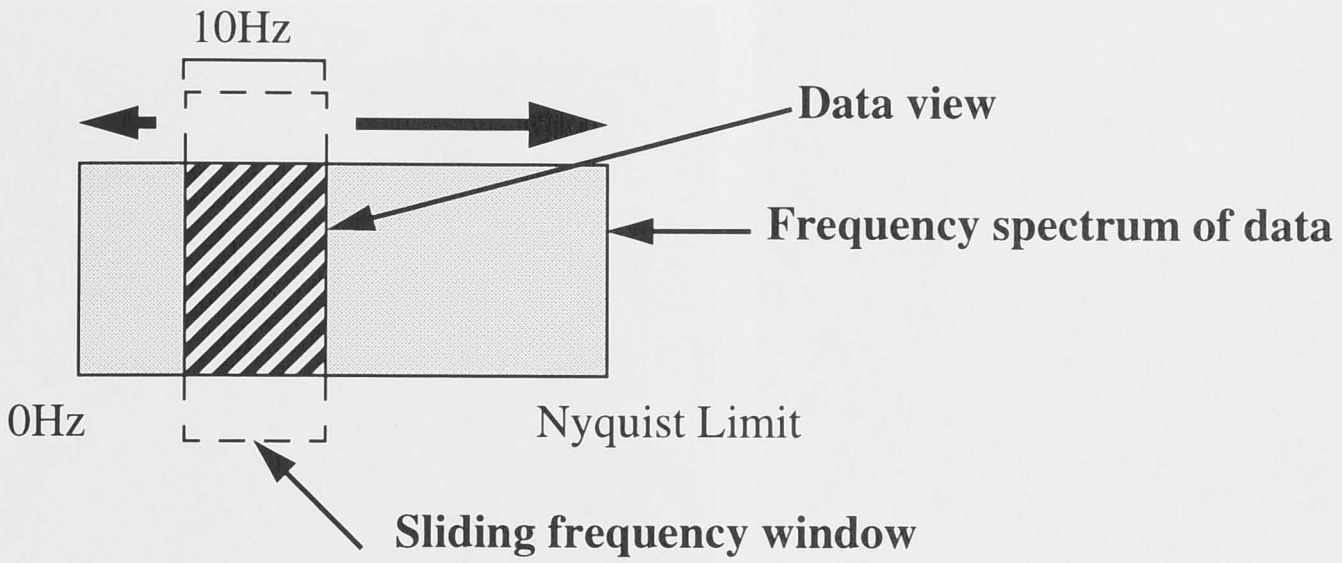


Figure 28 Sliding frequency window paradigm

4.4 Results

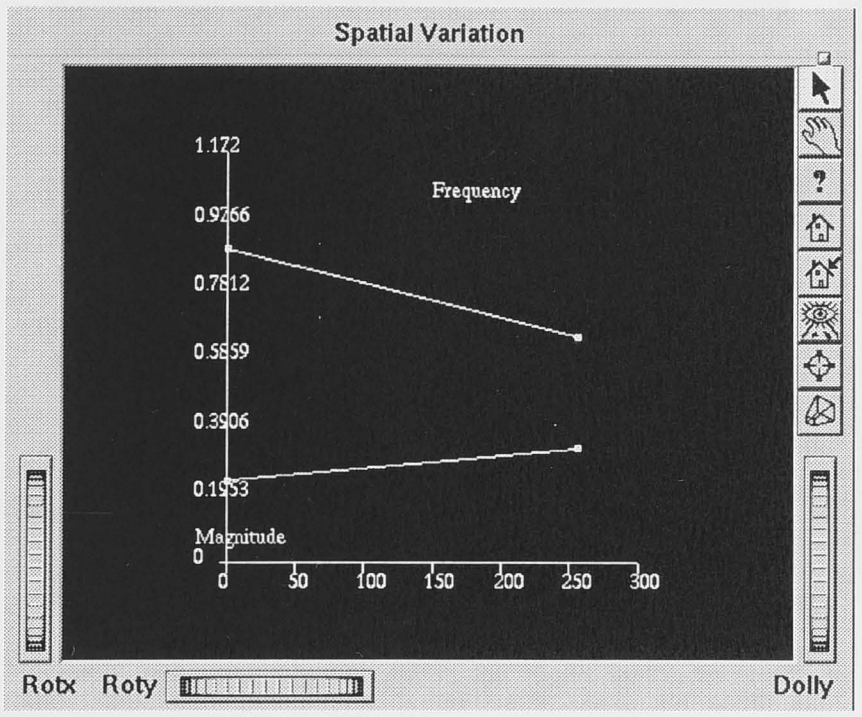
The interactive graph for controlling the filter is easy to use, but time consuming when specifying complicated filter variations. There is no noticeable difference in the "ease of use" between the piecewise-linear and the spline functions.

The results of varying the filter with interaction can be seen in the synthetic images, though only the broad variation is noticeable without close examination. Magnifying sections of the image may aid in allowing the results of fine interaction to be seen more easily. It is found that piecewise linear interaction gave an adequate level of control over the filter variation to the level that is noticeable.

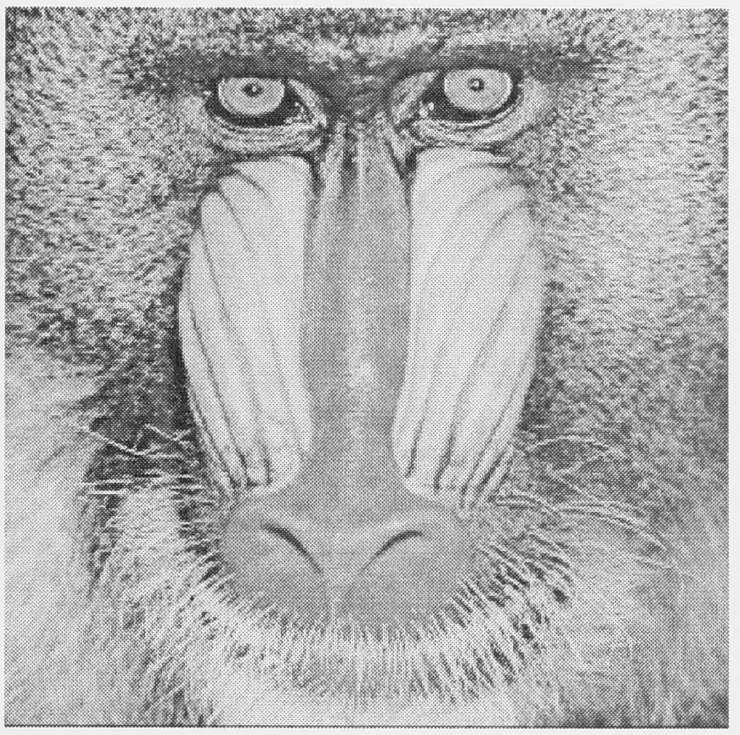
Interaction was sufficiently fast to see cause and effect on images of size 512x512 pixels. Interaction is less than a second, which is adequate, however faster filtering would improve the process.

The evaluation of the space-variant band-pass filtering method, and the user interface for controlling it, have been performed on a Silicon Graphics Indigo Impact Workstation. Experiments were performed on synthetic images which our non-geophysicists users are familiar with, which allowed the effects of the filter to be understood; unlike a practising geophysicist, they would have less chance understanding the action of a band-pass filter on seismic data. The main test image, namely the mandrill, is a common test image in the image processing community with a good spread of frequencies.

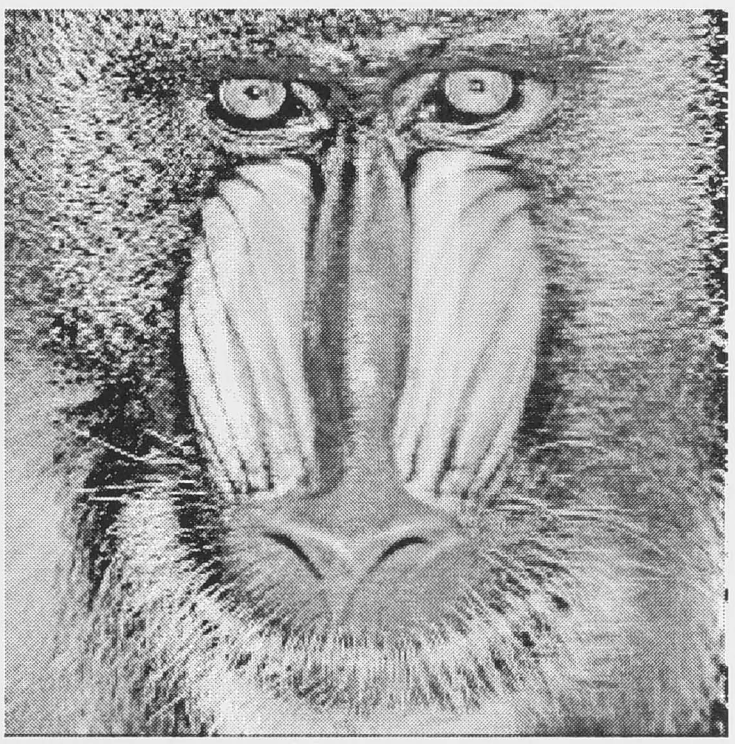
Figure 29 shows an example of controlling the variation of a band-pass filter over the mandrill image. Figure 29.(a) shows a filter variation specified by the control interface, (b) shows the original mandrill image, and (c) shows the image after the space-variant band-pass filtering.



(a)



(b)

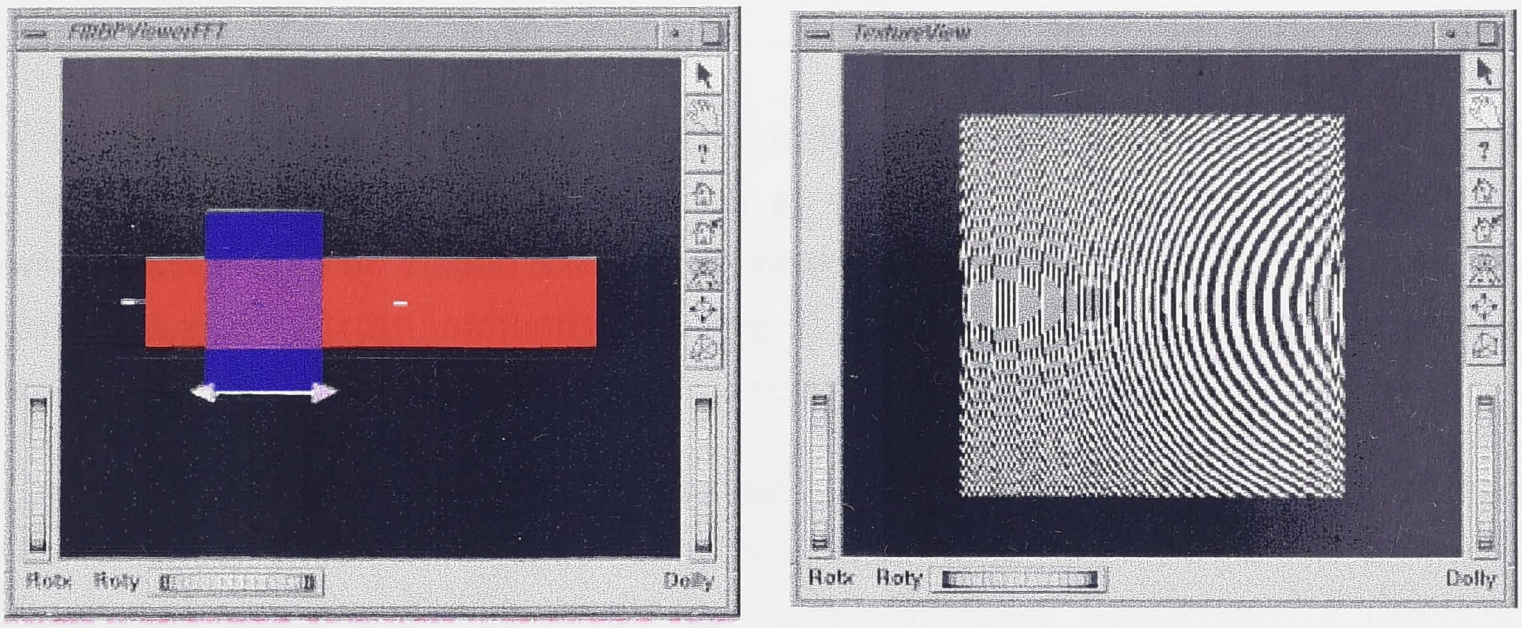


(c)

Figure 29 (a) Interactive graph for controlling the variation of filter bandwidth, (b) original image, (c) filtered image

A frequency sliding window display has been tested for viewing different image frequencies dynamically. A simple prototype interface and panel display, which allows the upper and lower cutoff frequencies to be controlled independently or together, is shown in Figure 30. A 2D sine function image is used to test the display.

The interface proved to be easy to use, and effective in showing the frequency content of the test image.



(a) (b)

Figure 30 Filter panel display prototype. (a) interface, (b) dynamic band-pass filtered panel

4.5 Summary

In this chapter an interactive space-variant band-pass filtering technique for improving the signal-to-noise ratio of seismic data has been presented. A space-variant band-pass filter has been developed whose upper and lower cutoff frequencies are continuously variable and whose cost and quality can be balanced by the user. Direct interactive methods for controlling this filter's upper and lower cutoff frequencies have been designed, together with some dynamic visualization methods for understanding the frequency content of the data. Experimentation with these methods shows success in allowing the user to control the filter variation over synthetic images.

5 VISUALIZING DIGITAL RASTER MAPS

With the current explosion in data generation and storage causing increasingly larger data sets to be available, operations such as geometric scaling become important in allowing such sets to be visualized. Digital raster maps (DRM) are one such type of data that benefits greatly from interactive scaling. It allows whole maps to be interpreted at a glance at low resolution, as well as smaller regions at higher resolution.

Geometrically down-scaling digital raster maps (DRM) requires low-pass filtering to prevent aliasing. Choice of this filter greatly affects the quality and general properties of the final map image and thus should be made with care. Space-variant filtering can improve image quality, but a well chosen space-invariant filter can also produce acceptable results quickly. A subjective experiment is designed for the selection of a suitable perceptually pleasing space-invariant filter for down-scaling DRM, when visual quality rather than numerical accuracy is required. Using this method, the two parameter cubic convolution filter with parameters (-2, 1.7) improves the readability of text, roads, and other critical map features at low resolutions over the more traditionally used most numerically accurate filter (0, 0.5).

Image filtering is a costly operation and difficult to compute at interactive rates for large images. Serial algorithms scale poorly with data size, so parallel implementation is necessary to support a more general framework for interactively filtering large data sets. To improve the efficiency and speed of parallel filtering algorithms, a software clustering scheme, which works in the reverse of traditional virtualization schemes is developed. The clustering scheme increases the speed of filtering algorithms when the number of processors is larger than the grain size of the problem. It uses several real processors to perform the task of one virtual processor, but is limited in the types of algorithms it can handle. The computation and memory savings of clustering is presented and compared against non-clustered algorithms for geometrical scaling. Both a parallel 2D algorithm and a two-pass separable 1D algorithm are used for the comparison.

5.1 Chapter outline

Digital raster maps form an imposing source of data, whether derived from digitized paper maps or sensed imagery. In this chapter we focus on how best to efficiently manipulate such data, and how best to visualize the information.

First, the selection of a filter for preserving critical map details when geometric scaling of raster maps is examined. To this end, a set of perceptual tests are performed to evaluate the quality of a particular set of filtering functions. Second, methods for speeding up geometric scaling operations by exploiting parallel computing are explored.

Before going further, the characteristics of digital raster maps and the features which make them different from general raster images are described, and the theory behind geometric scaling is summarized.

5.2 Digital raster maps

We define a DRM as any image-based map that can be derived from scanning or digitizing paper maps, topographical maps, contour maps, and street maps. They can also be created directly from the real world via sensing methods such as satellite and aerial photography, laser scans, and synthetic aperture radar. In most cases they have a tendency to be large, which creates difficulties for their display and for navigating within them.

DRM derived from scanning contain a mixture of smoothly varying texture interspersed with features such are region boundaries, roads, and labels. Edges around these features tend to be sharp with good contrast.

DRM created by direct sensing methods have features which may be unclear, blurred, or incomplete due to the sampling and/or any filtering performed during acquisition. There may be aliasing of high frequencies due to the sampling rate, noise induced by the sensing method, and geometrical distortions. Most will have had some post-processing steps performed to correct for the noise and distortion added by the sensing device and to improve data representation.

5.3 Geometrical scaling

By visualizing an entire map, an observer is often able to see structures which are not visible within smaller regions. It may thus be critical to preserve important features as much as possible when scaling down so that useful information is not lost.

Scaling, like all geometrical operations, is based on resampling and filtering. The quality and characteristics of a scaled image depend on the quality and characteristics of the filtering technique used in the resampling step. The better the quality of the filtering in the scale, the more accurately a user will be able to discern details at lower resolutions, and the easier navigation through the data set becomes.

Space-variant filtering can be used to improve the scaling of maps by using different filters on different types of image features. Typically map features consist of large areas where the data varies very little, combined with annotations, and sharp physical features such as text, roads, and boundaries between regions. The annotations and sharp features can be modeled as discontinuities in what is otherwise low-frequency data. For most applications, when manipulating these raster maps, it is important that the information

contained in the discontinuities be preserved in priority over the rest of the data. Given this, scaling algorithms, whether space-variant or space-invariant, should use filtering techniques that enhance discontinuities over other features.

Geometric scaling involves three steps for enlarging details, and four steps for reducing it. For enlarging detail, it first requires the digital image function to be reconstructed from the data samples into a continuous function. The scaling transformation is applied to the continuous function, and then the continuous function is resampled. For reducing detail, a low-pass filtering of the reconstructed continuous function may be required.

In practice, the image function is never fully reconstructed between data samples. Reconstruction is only performed at the new sample positions. The reconstruction, low-pass filtering and sampling steps are generally performed as a single filtering operation [Wolberg, 1990]. Figure 31 shows a flow diagram of the resampling operation.

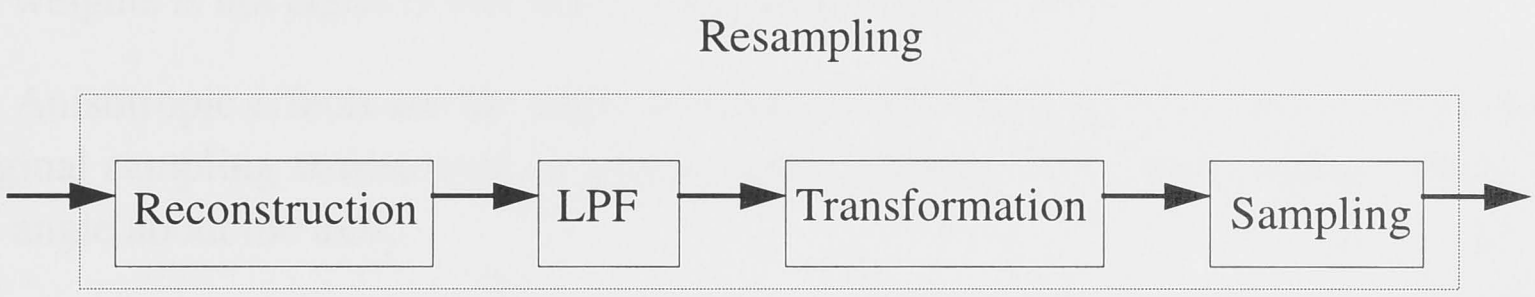


Figure 31 Resampling step

5.4 Perceptual evaluation of scaling filters

Central to any geometric transformations is the reconstruction of discrete data into a continuous representation. Reconstruction is performed using filtering techniques and the selection of good filtering techniques must be based on data characteristics, types of transformations, and the desired properties of results. Evaluating the goodness of filters is not an easy task, particularly when perceptual criteria are involved. This is exactly what we will attempt to do; perceptual filter evaluation through scaling transformations. First the filtering operation is analytically described, then possible filtering artifacts or distortions are characterized.

Many filtering techniques consist of convolving the image samples with a filter kernel function. The filter kernel function produces new output samples by summing together the weighted contribution of a finite number of source sample points. If the original discrete function is $f(x)$, and the filter kernel function $h(x)$, then the filtered function $g(x)$ is determined by:

$$g(x) = \sum_{v=-\infty}^{\infty} f(v)h(v-x)$$

In the continuous domain a convolution integral is used instead of summation. Since $h(x)$ is of finite size, the limits of v can be restricted to the values over which $h(x)$ is defined. An inherent problem with convolution style filters is how to handle the edges of an image where the filter kernel extends over the edge. Some common methods used include setting extra values to zero, extrapolating the data near the edge, and replicating the data at the edge. For simplicity, we have adopted the latter of these approaches.

When performing geometrical transformations over DRM, the reconstruction and subsequent resampling steps are often responsible for introducing visible distortion in the output. Several types of distortion have been described, these include sample-frequency ripple, anisotropic effects, ringing, blurring, and aliasing [Schreiber and Troxel, 1985].

Sample-frequency ripple is shown as periodically varying intensity values in regions that were of a constant intensity before filtering. It is caused by filters where the sum of the weights is not equal to one [Mitchell and Netravali, 1988].

Anisotropic effects are the angle dependent behavior of a filter, and often reveal the original sampling structure of an image. It is caused by filters whose shape varies with the angle about the axis.

Ringing is a periodic rippling that is radiated out from edge features within an image. It is caused by large negative side lobes of the spatial filter, which corresponds to a sharp cut-off in the frequency domain. The sharp cut-off causes frequencies just above the cut-off to be poorly attenuated. The deeper the lobes, the heavier the ringing, and multiple negative lobes produces multiple rings.

Blurring is the loss of image detail or smoothing out of sharp features. It is caused by filters with high attenuation of frequencies in the pass-band.

Aliasing is commonly shown as jagged edges or moire patterns. It is caused by filters with poor attenuation of frequencies in the stop-band. The high frequencies which are not attenuated properly in the stop-band appear as low frequencies after resampling.

Distortion free reconstruction is theoretically possible via the optimal sinc function, but when truncated for practical computation, it introduces artifacts.

Traditionally, with the influence of signal processing on image processing, filter techniques have been evaluated using their frequency responses [Park and Schowengerdt, 1982], [Fraser, 1989a]. The better the filter, the fewer frequencies are present in the stop-band, and the less attenuation of frequencies in the pass-band. Image reconstruction differs from signal reconstruction in that some of the types of distortion avoided in signal processing may have visually pleasing effects in images, [Schreiber and Troxel, 1985], [Parker et al., 1983], [Brown, 1969]. As such, image processing

filters need to be judged by their perceptual effects, as well as by their frequency response.

Ringing is one distortion effect that can enhance the appearance of an image. A single "ring" at an edge can sharpen the appearance of that edge, and give an image a cleaner look. Too much ringing, or multiple rings from an edge generally degrades the image. The effects of ringing on appearance of sharpness is examined in [Brown, 1969].

Digital raster maps differ from digital raster images in that our ability to interpret the information within them is more important than fidelity to the original image. Thus, a filter that enhances the readability of a map is more desirable than one that preserves accuracy with respect to the original map (the original map often doesn't represent the accuracy of the original data very well).

In this work we will perceptually evaluate a specific class of space-invariant filters for geometric scaling operations.

5.4.1 Perceptual evaluation

To perceptually evaluate filters for specific applications, criteria must first be established by which the filters are to be judged. In Chapters 3 & 4, interactive adjustment of filter parameters is used to determine the appropriate variation and the amount of filtering to apply to specific images. The interaction allows the user to isolate and judge filter characteristics. Several perceptual experiments performed recently ([Mitchell and Netravali, 1988], [Schreiber and Troxel, 1985]), described in Chapter 2, Section 2.2.2, use direct image comparison methods to evaluate the results of different resampling filters. The image comparison methods used are time consuming and difficult when the differences are small. To address this problem, we focus on developing an image comparison method that takes advantage of the properties of the human visual system. Using this, we develop a discrete search-based method for selecting a resampling filter.

The human visual system is much more sensitive to motion than judging absolute intensity differences; it has preattentive processes that are dedicated to the detection of moving objects [Gershon, 1992]. We take advantage of this property for image comparison by making differences between images show themselves as motion. From experimentation, we found that by temporally flipping between spatially aligned images, differences in the images are seen as motion (much like between the differences between two consecutive frames of a movie film). Important for the method to be successful is that the images are shown at the same spatial location (as shown in Figure 32), and that the flip is performed fast enough to prevent flickering (in a single screen refresh).

To compare two images, the user flips back and forth between them. At each flip the user's eyes are drawn to the areas where motion occurs. By focusing on these areas and

flipping, the user is quickly able to determine the differences between the images, and to evaluate their subjective quality. We found it possible to find and evaluate many differences between images which looked indistinguishable using side-by-side comparison methods.

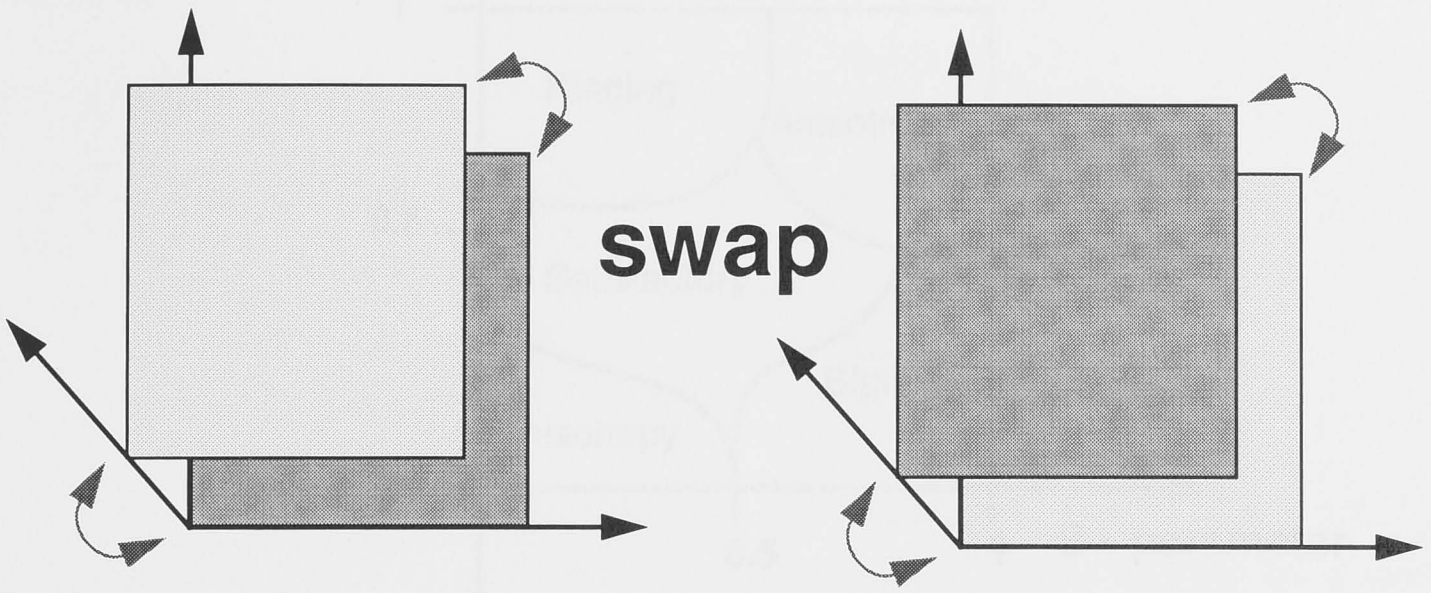


Figure 32 Flipping between images

The search method itself heavily depends on the properties of the filters being evaluated. For independent filters, an exhaustive search method is needed, but when there are similarities, more intelligent methods can be employed. In the experiment we chose to search the parameter space of the two-parameter cubic convolution filter. This filter provides a good quality/cost balance, and has flexibility of shape and filtering characteristics. The equations for this cubic filter and its two parameters B and C follows.

$$h(x) = \begin{cases} \left(\frac{1}{6}\right)((12-9B-6C)|x|^3 + (-18+12B+6C)|x|^2 + (6-2B)) & \text{if}(|x|<1) \\ \left(\frac{1}{6}\right)((-B-6C)|x|^3 + (6B+30C)|x|^2 + (-12B-48C)|x| + (8B+24C)) & \text{if}(1 \le |x| < 2) \\ 0 \qquad otherwise \end{cases}$$

Since [Mitchell and Netravali, 1988] had determined that there is a perceptual ordering of filter properties over the parameter space (shown in Figure 33) of the two-parameter cubic filter, we elected to use a binary style search in two dimensions. We first divide the parameter space into four quadrants. Taking one image from each quadrant we compare them against each other. Selecting one image selects its quadrant in the search space and discards the others. The selected space is then divided into four quadrants and the process is repeated until a sufficiently small area of the search space remains, or until images cannot be differentiated and compared. By using overlapping quadrants a greater search robustness is achieved. Figure 34 shows an example search over a 5x5 matrix of images.

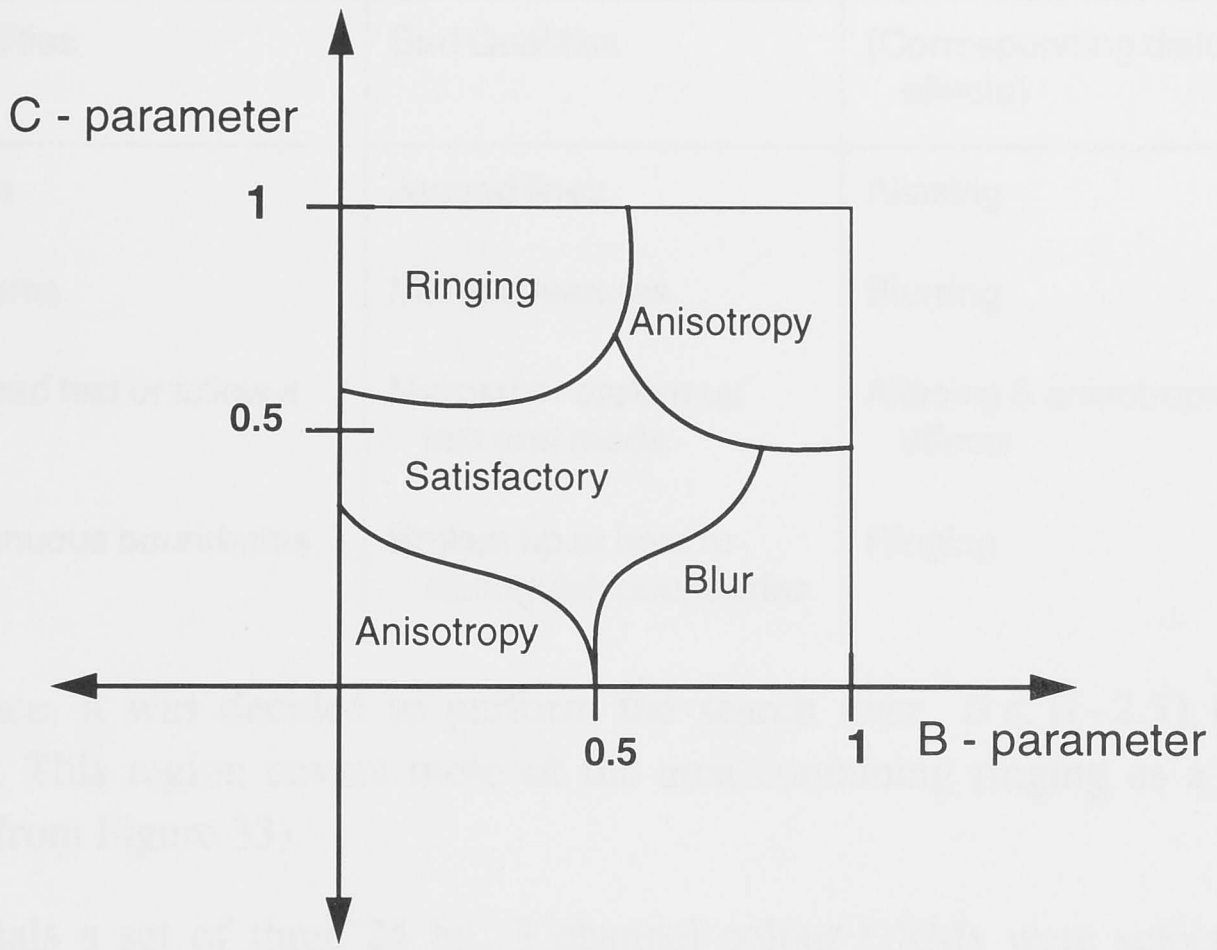


Figure 33 Division of parameter space into perceptual region by Mitchell & Netravali

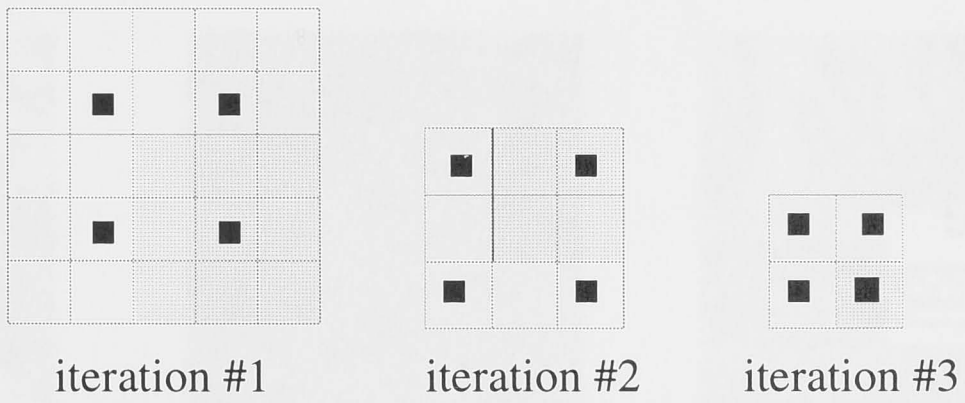


Figure 34 Example of three search iterations. Note the overlap between the search space, which provide extra robustness for searching (i.e a poor first selection can be corrected).

5.4.2 Experiment

In the experiment, comparison criteria (see Table 1) were given to each observer, and it was stressed that the purpose of a map is to present information about a geographic region. The best filtered map can be defined as the map that best conveys such information.

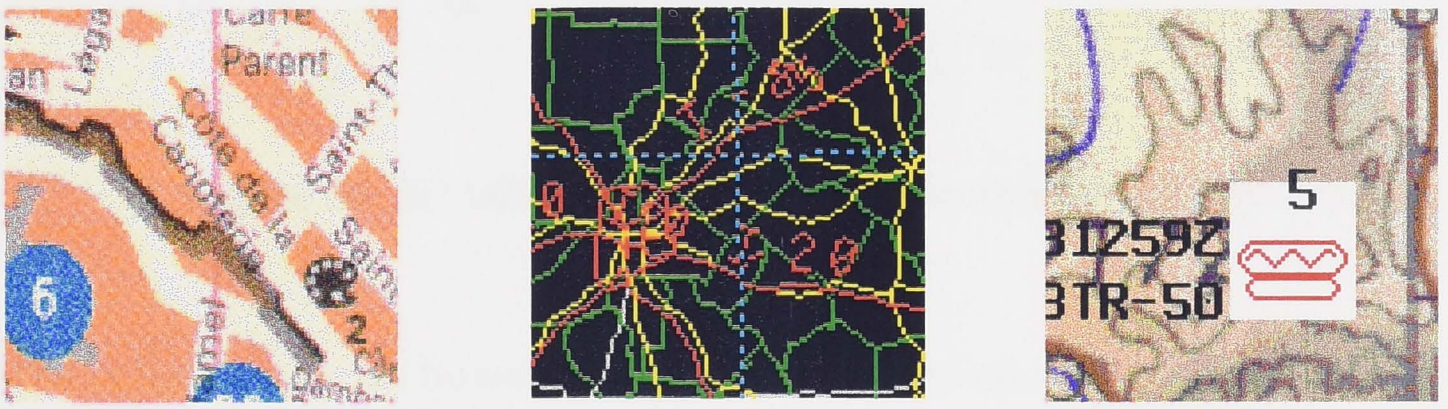
In the experiment performed in [Mitchell and Netravali, 1988], the parameter space classified was $B \subset [0, 1]$ and $C \subset [0, 1]$. Following initial exploration of the

Table 1. Comparison criterion

Good Qualities	Bad Qualities	(Corresponding distortion effects)
Sharp lines	Jagged lines	Aliasing
Clear features	Merged features	Blurring
Ability to read text or follow a road	Merged or broken up text and roads	Aliasing & anisotropic effects
Clear continuous boundaries	Broken up or hard to distinguish boundaries	Ringing

parameter space, it was decided to perform the search over $B \subset [(-2.5), 0.5]$ and $C \subset [0, 2.5]$. This region covers more of the area containing ringing as a dominant visual effect (from Figure 33).

For the trials a set of three 24 bit, 3 channel colour DRMs were selected which represent a range of map types. Portions of each are shown in Figure 35. The maps are examples of a scanned paper map, a computer generated map, and a combination scanned/computer map. They contain examples of angled text and other annotations, roads, region boundaries, contour lines, and computer symbols. The experiment searches the filter parameter space for each image at scales of 0.6 and 0.35.



(a) (b) (c)

Figure 35 a). Scanned paper map. b) Rasterized vector road map. c) Shaded contour map with computer added symbols.

Scaled images were created by interpolating the RGB channels individually using the two-parameter cubic filter and recombining. Improved interpolation of the colours could have been obtained by first converting to a perceptually uniform colour space.

The test environment was designed to ensure that all the trials had identical conditions. The tests were made in a closed room with diffuse fluorescent lighting. The screen was orientated such as to avoid light reflections with the observer sitting perpendicular to the monitor. A test system was constructed that allowed the observers to

freely swap between any of the four comparison images, thereby allowing each image to be directly compared against any other. When an image is selected out of the four, the system automatically descends the search tree, and presents the next four images to be compared.

To obtain good, statistically reliable results, 19 volunteer test subjects were used. The subjects were all postgraduate students and professors from the Vision and Digital Systems laboratory at University Laval.

5.4.3 Results

Figure 36 shows a tabular listing and a three dimensional plot of the collective results for all images and both scales. As can be seen, the majority of the selections were made in the top left corner, with the corner image receiving the most number of selections.

C \ B	-2.5	-1.75	-1	-0.25	0.5
2.5	29	19	12	4	7
1.875	7	8	5	3	3
1.25	2	1	5	1	0
0.625	2	2	2	0	0
0	0	0	0	0	0

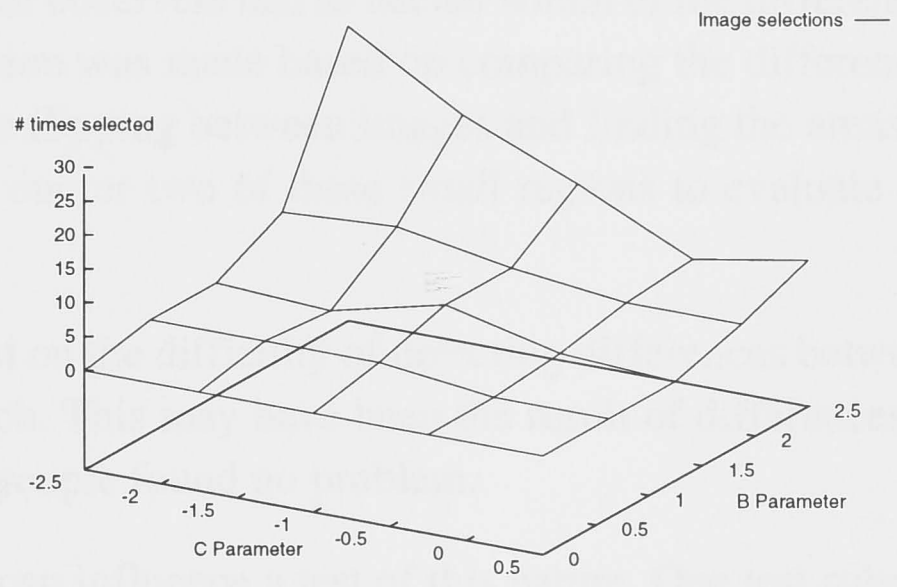


Figure 36 2D table, and 3D plot of collective experimental results.

The collective results however fail to show the variation between different maps and scale factors. The selections made for maps A and C were very similar with only a small standard deviations in their results. Map B selections contained much great variation, which was primarily due to the subjective difficulty in determining the type of distortion/ enhancement which best improved the image. Observers commented on the relative difficulty in deciding for map B which was the best image.

The mean parameter values for all images and scale factors from the experiment was chosen as a general filter for scaling digital raster maps. This is the two-parameter cubic (B, C) = (-2, 1.7).

5.4.4 Discussion

The speed at which observers were able to make comparisons, and the high consistency of the experimental results validated the experimental method and the use of subjective filter selection. That the most selected filter lay on the edge of the parameter space searched indicates that a larger, or different space should really have been searched.

Observers commented that when comparing some sets of images, the choice was very subjective. This was more often the case for the second test image, when there was often no obvious choice of which was the better image. Examining the results shows that statistically the choice was not as arbitrary as observers may believe because of the consistency of the results.

Observers commented that before they compared sets of images, they first had to decide for themselves what constituted a better image. The experimental setup allows differences in the images to be seen, but observers had to decide which of the differences gave the better image. Often the decision was made based on comparing the differences of very small features. Observers were flipping between images and finding the areas of greatest differences, then focusing in one or two of these small regions to evaluate the differences.

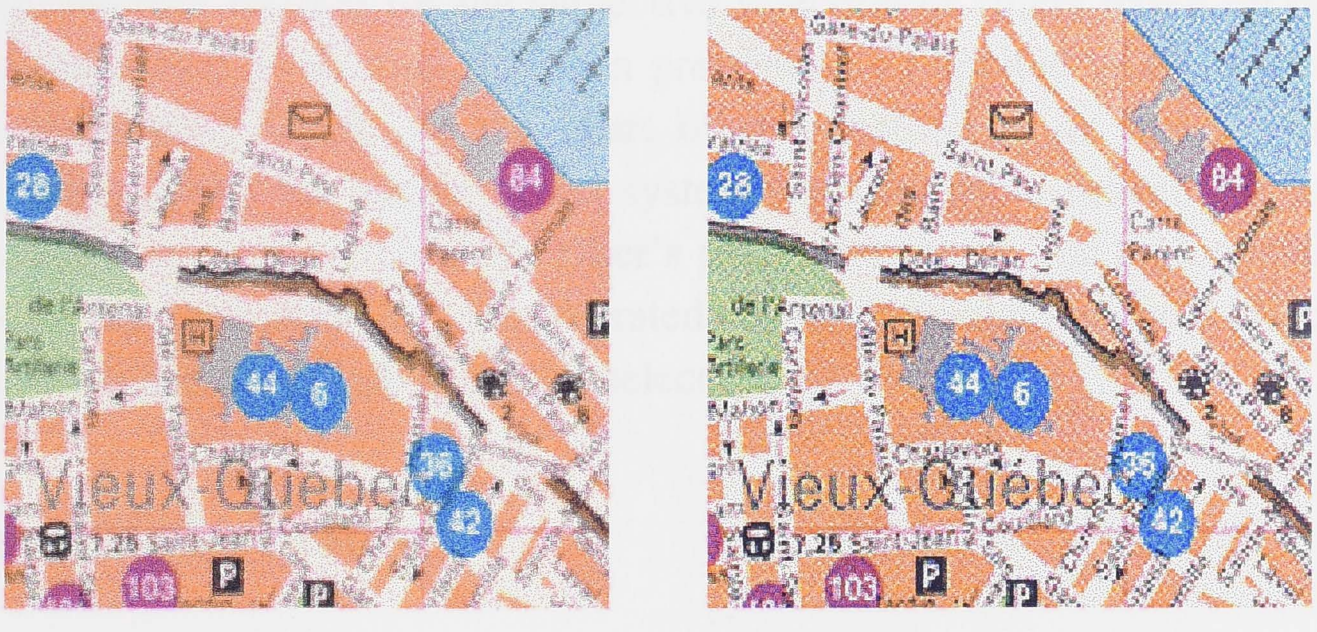
A number of observers commented on the difficulty of detecting differences between images in the 3rd iteration of the search. This may have been the result of differences in the quality of eyesight, because most people found no problem.

Color blindness is one factor that can influence a test of this nature. One test subject admitted to being partially color blind.

To examine the selected filter, it was compared with the most numerically accurate two-parameter cubic filter (B, C) = (0, 0.5) [Keys, 1981], [Reichenback and Park, 1989]. Figure 37 shows the first test image scaled by 0.6 with each of the filters.

The (0, 0.5) filtered image is a smoother image which has lost its sharp edges through the low-pass filtering properties of the filter. The (-2, 1.7) image is sharper, with the text and annotations standing out more clearly with little loss in intensity, but contains some aliasing. In the textured regions, noticeable aliasing is present, but does not degrade the information in the map.

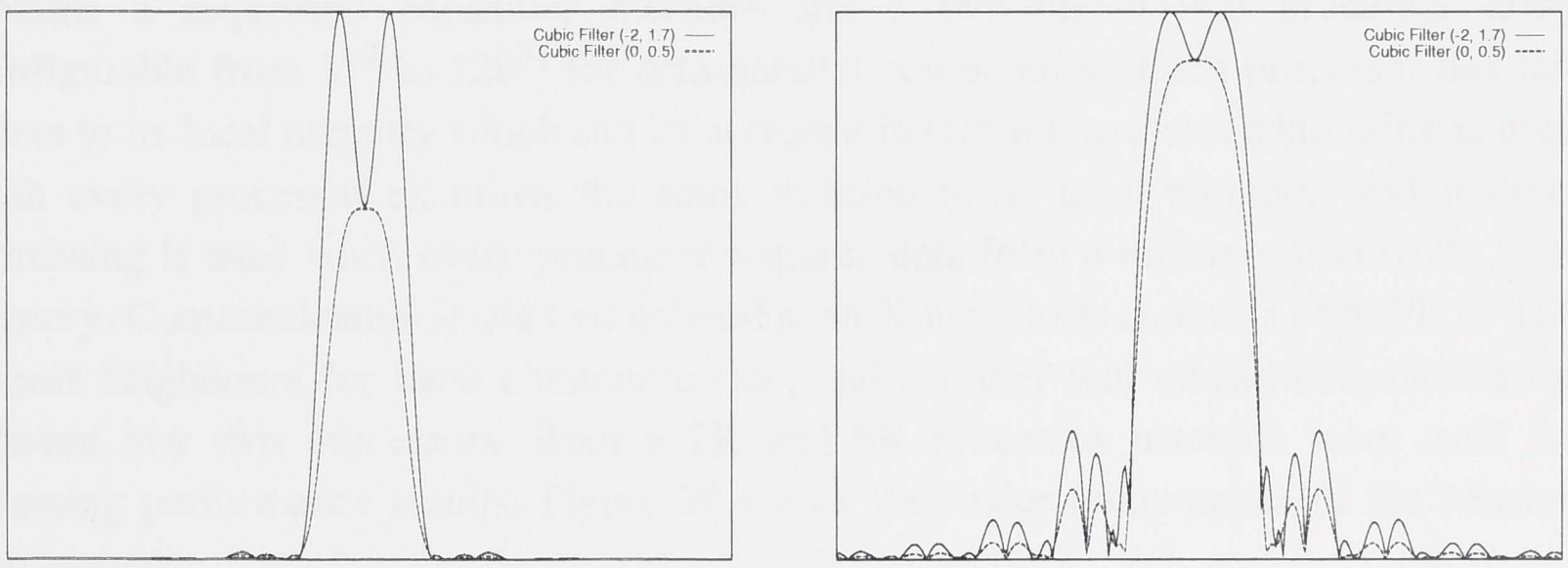
Comparing the frequency response of the two filters, the magnitude of the frequency response is shown plotted on both a linear and log scale in Figure 38. The linear plot best shows the filter's pass-band characteristics, while the log plot is better for stop-band characteristics. The steeper cutoff between pass and stop-bands of the (-2, 1.7) filter over the (0,0.5) filter allows better preservation of higher frequencies in the image. The trade off is an amplification of frequencies just below the cutoff in the pass-band. This



(a) (b)

Figure 37 a) Test image 1 scaled using (0, 0.5) filter and b) Test image 1 scaled using (-2, 1.7) filter.

amplification, combined with the sharper cutoff, sharpens edge features by adding ringing around them which increases their intensity. In [Parker et al., 1983], it is pointed out that a small amount of amplification in frequencies below the cutoff is appealing to the eye, but we found with digital raster maps that a significant amplification of those frequencies can be desirable.



(a) (b)

Figure 38 (a) Linear plot of the frequency response of the subjectively selected cubic filter (-2, 1.7), and the most numerically accurate cubic filter (0, 0.5). (b) Log plot of the frequency response of the cubic filters (-2, 1.7) and (0, 0.5).

A steeper slope between pass and stop-bands results in more frequencies being passed in the stop-band. This leads to aliasing of structures containing high frequencies, but this appears not to be a problem for the class of digital raster map images.

5.4.5 Experimental conclusion

An experimental method for the subjective selection of filters based on perceptual rather than numerical evaluation has been presented and applied to the application of geometrically scaling DRM. At the heart of this method is an image comparison technique that exploits the human visual system's sensitivity to movement, and a two dimensional binary style search of the filter's parameter space. The high consistency of the results for the application has demonstrated the experimental method's effectiveness, and the need for perceptually based filter selection.

5.5 Parallel algorithms

In order to obtain the interactive rates required for navigating through different scales of large digital raster maps, a fast machine is required with high memory/processor bandwidths. Massively parallel machines best suit these criteria, and are well matched to operating on large data sets.

Two algorithms are described for scaling large data sets. One algorithm exploits the separability of the scale operation by performing two passes with a 1D filter, the other uses a single pass with a 2D filter. For both algorithms, a lookup table is used to store a sampled version of the filter kernel (see chapter 3).

Both algorithms have been implemented on the MasPar MP-1. The MP-1 computer contains a sequential controller machine and a two-dimensional processor array (configurable from 32^2 to 128^2) for data-parallel computation. Each processor has fast access to its local memory which can be accessed in two ways; direct addressing is used when every processor examines the same location in its local memory, and indirect addressing is used when every processor requires data from a different part of its local memory. Communication is via two networks: an X-net which connects each PE to its 8 nearest neighbours for local communication, and a router that allows communication between any two processors. Both a 2K and 8K processor machine were used for obtaining performance results. Figure 39 shows the major components of the MasPar architecture.

Some filtering issues of the scale operation are first examined, followed by a description of a software clustering scheme that improves performance when the number of processors is greater than the grain size of the problem. Then both the 1D and 2D scale algorithms are described and evaluated.

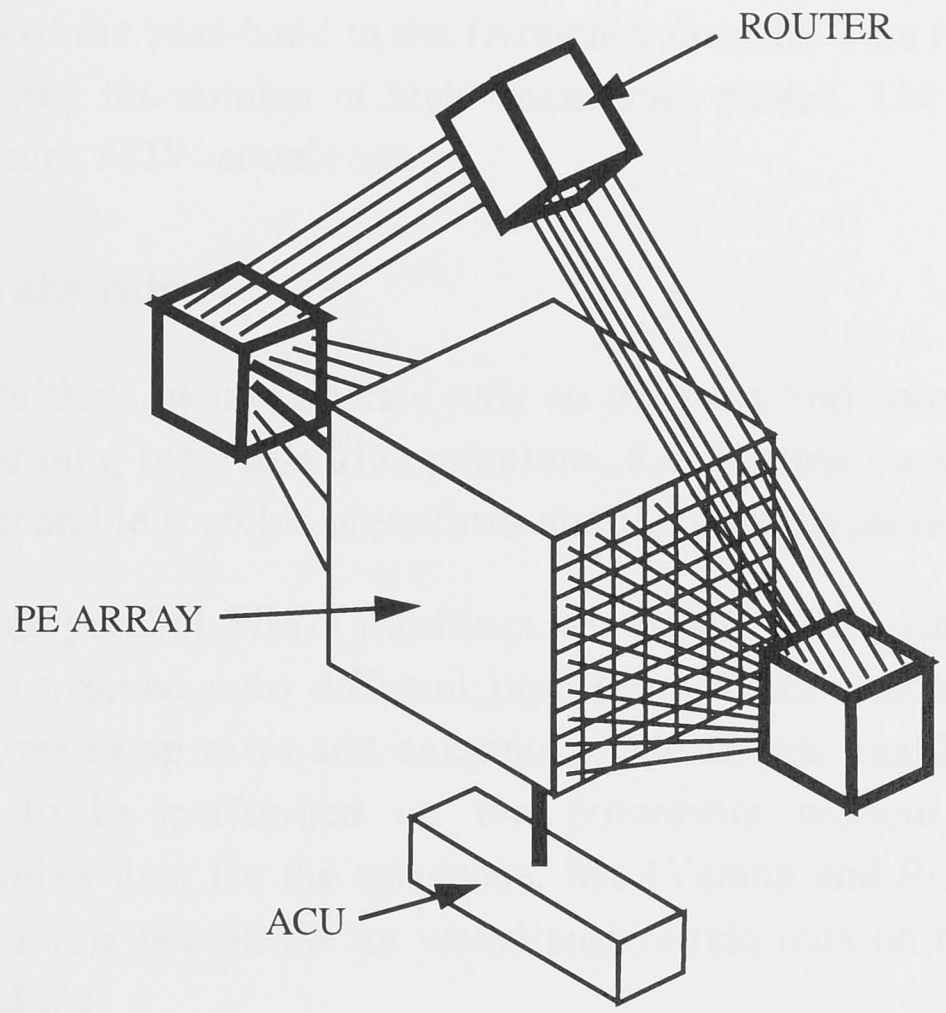


Figure 39 MasPar architecture

5.5.1 Change in filter size with scale

For an infinite sinc, Figure 40 shows the relationship between the dimensions of an ideal 1D filter in the spatial domain, and the band-pass characteristics in the frequency domain. The filter passes all those frequencies between +W and -W, and cuts off all others. The normalized height of the pass-band in the frequency domain is one, so that frequencies in this range are neither amplified or attenuated, i.e A/2W = 1 [Wolberg, 1990]. Finite impulse response filters have a similar relationship, but often have undesirable attenuation of frequencies in the pass-band, and leakage of frequencies in the stop-band.

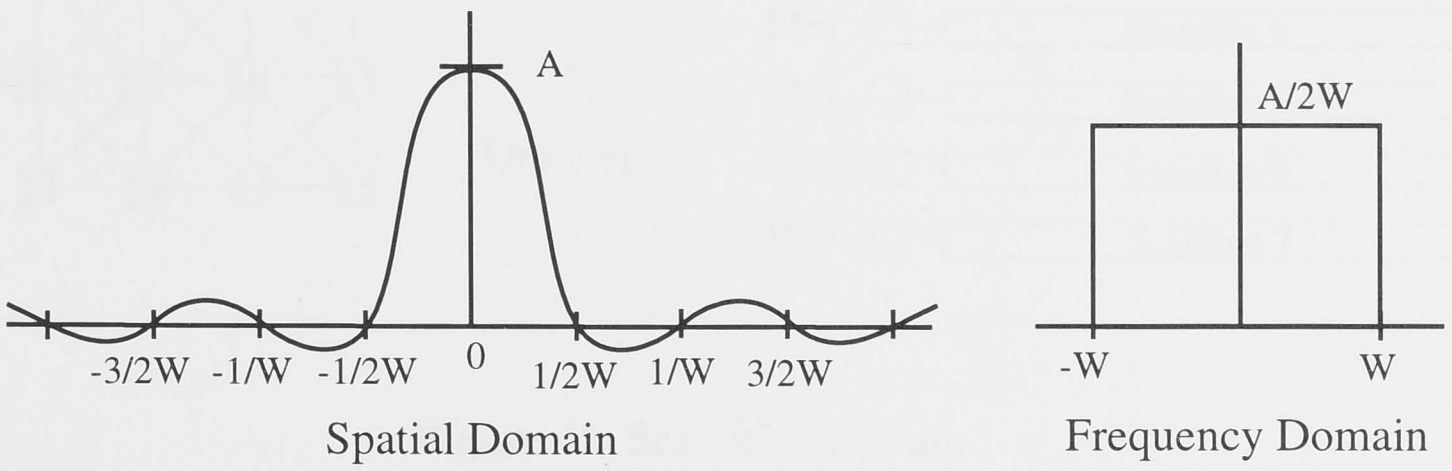


Figure 40 Ideal low-pass filter in both the spatial and frequency domains.

When down-scaling an image, the filter is scaled horizontally in the spatial domain to reduce the width of the pass-band in the frequency domain. This increases the low-pass filtering by reducing the number of high frequencies passed. The height of the filter is also scaled to ensure A/2W equals one.

5.5.2 Scanline algorithms

Scanline algorithms, which operate only on the rows and columns of an image, are used when performing separable 2D operations. See [Catmull and Smith, 1980] for an explanation of separable scanline algorithms and the filtering issues involved.

Massively data-parallel SIMD machines are ideally suited for scanline operations. Scanlines can be mapped onto different processors of the machine and processed in parallel. Transposes swap rows and columns of the image, enabling both column and row operations to be performed on the processors without any inter-processor communication other than for the transpose. See [Vezina and Robertson, 1992a] for a description of scanline operations for visualizing terrain data on the MasPar massively data-parallel machine.

Processors in massively parallel machines can be connected in meshes, hypercubes, trees, and other types of networks. Scanline mappings on such machines have one scanline per processor, usually with adjacent scanlines on adjacent processors. Unraveling the architecture and visualizing the processors as belonging to a linear array, as shown in Figure 41, allows operations on scanline mappings to be more easily visualized and described. Each row lies along the memory axis, and each column along the processor access.

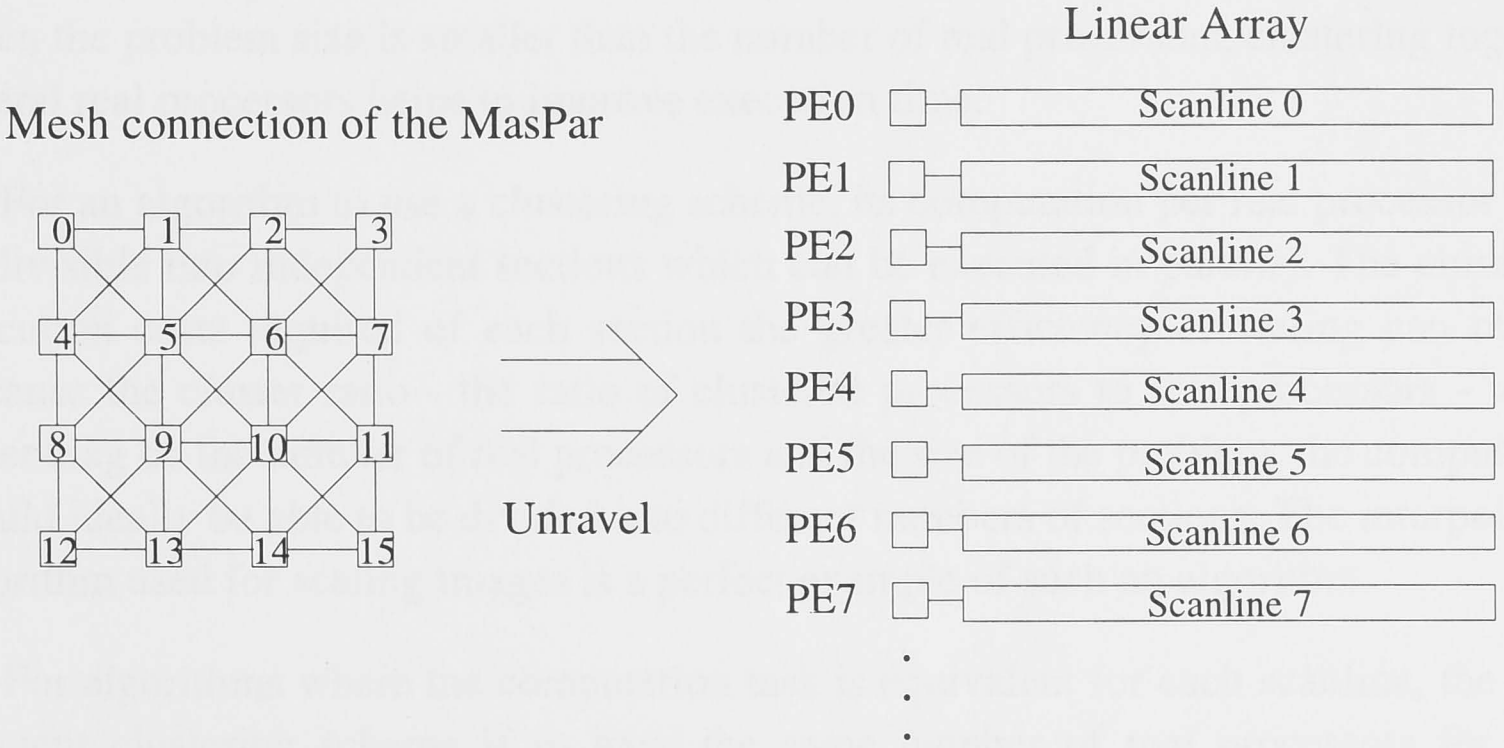


Figure 41 Scanline mapping

Virtualization schemes allow scanline algorithms to operate when the number of image rows exceeds the number of processors. They work by having each real processor emulate one or more 'virtual' processors. This emulation can be performed directly in hardware or simulated in software. When the grain size of a problem is greater than the number of real processors, virtualization allows the algorithm to be executed. See [Hillis, 1985] for a discussion of virtualization. The greater the number of virtual processors a real processor emulates, the slower execution time becomes. There is a linear trade off between speed, and number of processors emulated. Available memory per virtual process also linearly decreases with the number of processors emulated.

Depending on the number of virtual processors required, different real processors may have to emulate different numbers of virtual processors. Some systems, such as that used by the CM2's C*, only allow the number of virtual processors to be an integer multiple of real processors. Virtualization can be a transparent process, or explicitly handled by each program.

5.5.3 Processor clustering - reverse virtualization

When the data size is small, most data parallel algorithms become inefficient in their use of the available processors on a machine. For scanline algorithms, when the number of processors is greater than the number of scanlines, the extra processors are not used. For example, performing a scanline algorithm on a 1K image with an 8K processor machine, 7K processors remain idle.

We develop a clustering scheme for massively data-parallel machines that is the reverse of virtualization. Instead of using real processors to emulate more virtual processors, real processors are clustered together to emulate fewer virtual processors. When the problem size is smaller than the number of real processors, clustering together several real processors helps to improve execution time.

For an algorithm to use a clustering scheme, its computation per real processor must be divisible into independent sections which can be executed in parallel. The closer the execution costs required of each section the greater efficiency clustering can obtain. Because the cluster ratio - the ratio of clustered processors to real processors - varies depending on the number of real processors and the size of the problem, the computation should ideally be able to be divided into different numbers of sections. The interpolation algorithm used for scaling images is a perfect example of such an algorithm.

For algorithms where the computation task is equivalent for each scanline, the most efficient clustering scheme is to have the same number of real processors for each cluster. In algorithms where the computation task varies for different scanlines and is known in advance, a more flexible clustering approach where different clusters have different numbers of real processors can be used. The aim of the clustering scheme is to

keep the computation load balanced over the processors. The case of equal processors per cluster is examined.

A clustered algorithm requires two extra steps over a conventional one. It requires the data to be first mapped over the extra real processors, and requires each real processor to compute its part of the problem.

For communication efficiency, adjacent processors are clustered together. For scanline algorithms, this requires the original scanlines to be mapped so that one processor per cluster contains a unique scanline. The scanline is then copied over the rest of the real processors that make up the cluster.

A standard scanline mapping on a massively data-parallel machine has scanline N mapped onto processor N. For clusters, if there are R times more processors than scanlines, then scanline N is mapped onto processor RN. The case where $R = 2$ is shown in Figure 42.

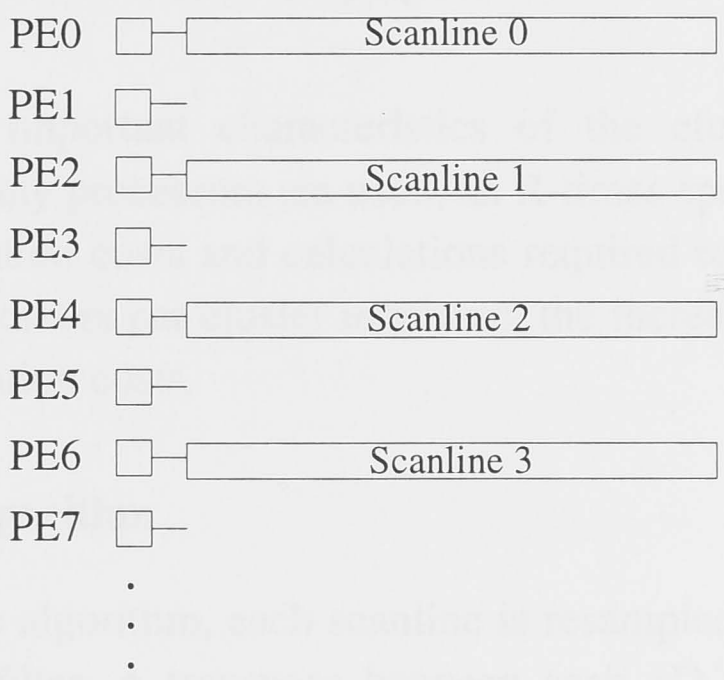


Figure 42 Scanline mapping for clusters. First stage

The original task of the single processor is divided up amongst the R processors of the virtual processor, and the data required by each real processor copied to it. For the scale operation, the output scanline is divided into R equal sections, and copied across the real processors in each cluster. This is shown in Figure 42 for the case where R=2. Because neighboring processors are grouped to form a cluster, fast local communications can be used for intra-cluster data movement and communication.

To enable the 1D filter to cover the data points around each new sample position, there is an overlap of the original scanline data required by each processor (Figure 42). The overlap size equals the filter width, and varies with scale factor. Once each processor has calculated its output section of the scanline, the results can be merged together onto a single processor of the cluster, forming the mapping shown in Figure 42. This data mapping can be passed on to subsequent operations that use the same

clustering ratio, or the data can be mapped back to the standard scanline mapping, as shown in Figure 41, ready to be passed to another scanline operation.

Figure 43 Scanline mapping for clusters. Second stage

There are several important characteristics of the clustering method to note. Although R times as many processors are used, an R-times speed up is not obtained due to the extra communication costs and calculations required to set up the clustering. As the number of real processors per cluster increases, the increases in speed diminish due to increased communication costs.

5.5.4 Two-pass 1D algorithm

To test the 1D scale algorithm, each scanline is resampled in each dimension using the cubic convolution filter. A transpose between each 1D pass maps the dimension being processed into the memory of the processors.

The steps of the two-pass 1D scanline scale operation are shown in Figure 44. The example image consists of four scanlines, and has each scanline mapped onto a different processor. The first step scales the original image in the horizontal direction. A transpose then swaps the dimensions, and another horizontal scale completes the operation. A final transpose brings the image back into its original orientation. The scale shown shrinks the details in the image, but keeps the output size the same as the input size. In practice, for large images the output size would be much smaller than the input. A pan function would be used to allow different portions of the image to be seen. This is achieved by first cropping the image so that only data points required to compute the output image are used. This reduces the data movement requirements of the algorithm, and thus its communication costs. An example of cropping, for the case of scaling up, is shown in Figure 45.

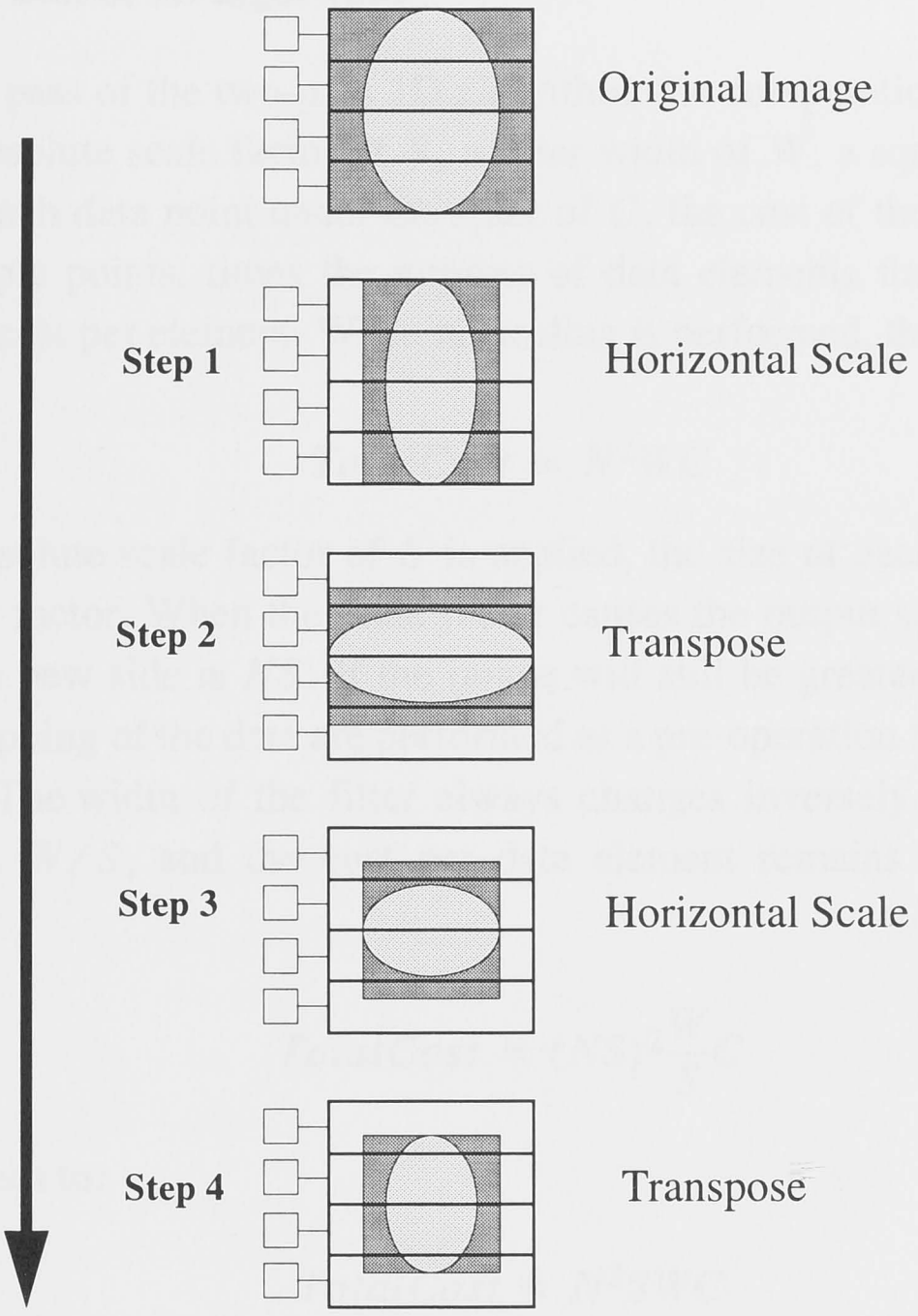


Figure 44 Two pass 1D scale operation.

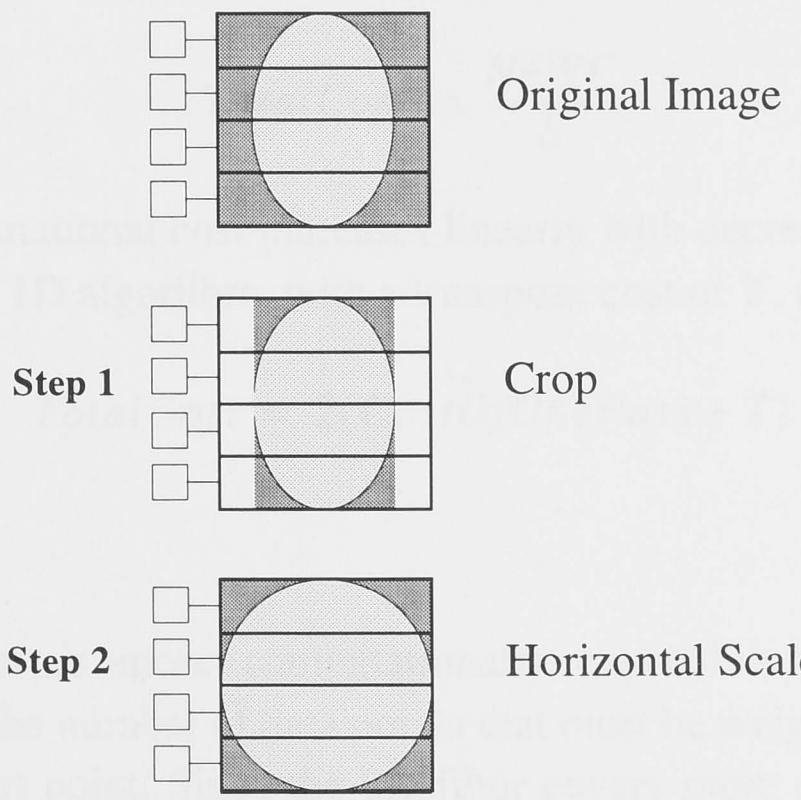


Figure 45 Cropping before horizontal scale. Scale up.

Computational cost of 1D algorithm

For a single pass of the two-pass 1D algorithm, the computational cost is computed. Assuming an absolute scale factor of S, a filter width of W, a square image of side N, and a cost per each data point under the filter of C, the cost of the filter step equals the number of sample points, times the number of data elements that are covered by the filter, times the cost per element. When no scaling is performed, the filtering cost is:

$$TotalCost = N^2WC$$

When an absolute scale factor of S is applied, the size of each side of the image is changed by that factor. When the scale factor causes the output side to be less than the display size, the new side is NS. If the image will still be greater than the display size after scaling, clipping of the data are performed as a pre-operation to keep the output side constant at N. The width of the filter always changes inversely proportionally to the scale factor, i.e W/S, and the cost per data element remains the same. Total cost becomes:

$$TotalCost = (NS)^2\frac{W}{S}C$$

Which reduces to:

$$TotalCost = N^2SWC$$

For the case where the output size is smaller than the display size, the computation cost decreases linearly as the scale decreases. Where the output has to be clipped to the display size, computational cost is

$$TotalCost = \frac{N^2WC}{S}$$

In this case, computational cost increases linearly with decreases in scale factor. For the complete two pass 1D algorithm, with a transpose cost of T, cost equals

$$TotalCost = 2(CostOfOnePass + T)$$

5.5.5 2D algorithm

The 2D algorithm is more computationally expensive than the two-pass 1D algorithm because of the number of data points that must be weighted by the filter kernel to produce each output point. Since the 2D filter covers more data points than the 1D filter, each processor needs more data if it is to produce an equivalent number of output points. The calculations for determining the filter weights for each data point is also more expensive because of the extra dimension.

We map the problem onto processors by assigning the task of producing each output scanline to each processor (clustered or normal depending on data size). The data required by each processor to produce its scanline is then copied to it. This method is chosen because of its simplicity, and compatibility with scanline based methods. Its communication overheads are also small compared with other methods. Once the data are initially mapped, no further communication is required until the results are merged. An example of how the data are mapped to each processor for both clustered and non-clustered versions of this algorithm is shown in Figure 46.

One inherent problem with 2D algorithm for scaling is the relationship between scale factor and the size of filter. In the 1D algorithm, the number of elements covered by the filter increases inversely proportionally with the scale factor. In 2D, the number of elements increases inversely proportionally with the square of the scale factor. As the image is scaled down, the number of data points covered by the filter rapidly increases, and the amount of data that must be copied between processors increases. Depending on the size of the image and the scale factor, the available memory may be exhausted on each processor if a straight scanline mapping is used. Clustering alleviates this effect. The higher the scale, the fewer the number of output scanlines, and the fewer the number of points in these. With fewer scanlines, a greater clustering ratio can be used, and so the memory usage remains stable.

Memory usage in relation to scale factor

Assume a scale factor of S, a filter width and height of W, a square image of side N with B bytes per data element, and a parallel machine with P processors. In a standard scanline mapping, we have the N^2B bytes of the image spread over the processors with NB bytes per processor. If we scale the image by S, then the filter width and height become W/S. For each processor to produce its output scanline, it will need at most $\lfloor W/S \rfloor$ scanlines of the original image on each processor to have all the data points that lie under the filter. In this case, each processor gets $\lfloor W/S \rfloor NB$ bytes of data. As scale factor decreases, memory used per PE increases linearly.

If clustering is used, then we have $\lfloor W/S \rfloor NB$ bytes per cluster, which is spread over the processors in the cluster.

The number of processors in a cluster equals the total number of processors P, divided by the number of output scanlines of the scale NS, which is:

$$\left\lfloor \frac{P}{NS} \right\rfloor$$

Therefore the amount of memory used per processor in a cluster approximately equals the total memory used by the cluster, divided by number of processors per cluster.

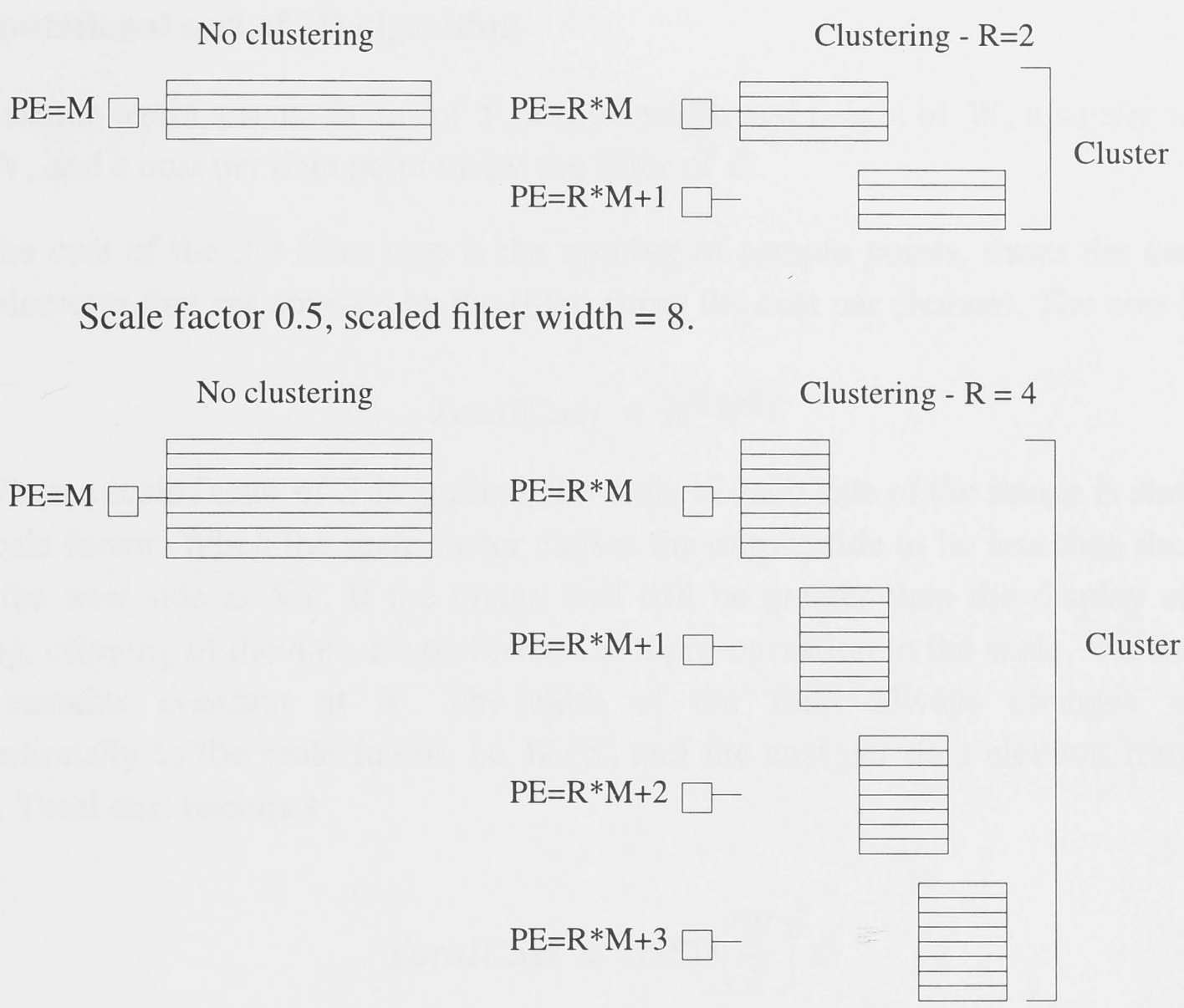


Figure 46 Mapping of data onto processors for 2D scale. An example of the scanlines mapped to a particular processor or cluster at two different scale

$$\frac{\left\lfloor \frac{W}{S} \right\rfloor NB}{\left\lfloor \frac{P}{NS} \right\rfloor}$$

If we approximate the size of the filter and the number of processors per cluster as rational values, this reduces to:

$$\frac{WN^2B}{P}$$

which is independent of the scale factor. This value is approximate because the data used by a cluster is not just divided among the processors of the cluster. There is an overlap the size of the filter width for each scanline segment between adjacent processors in the cluster. Total memory of this overlap equals $(W/S)^2B$ bytes, which increases inversely proportionally with the square of the scale factor. Because the filter size is generally small to begin with, this increasing overlap only becomes important with large images and small

scale factors.

Computational cost of 2D algorithm

Assume again a scale factor of S, a filter width and height of W, a square image of side N, and a cost per data point under the filter of C.

The cost of the 2D filter step is the number of sample points, times the number of data elements that are covered by the filter, times the cost per element. The cost is:

$$TotalCost = N^2W^2C$$

When a scale factor of S is applied, the scale of each side of the image is changed by the scale factor. When the scale factor causes the output side to be less than the display size, the new side is NS. If the image will still be greater than the display size after scaling, clipping of the data are performed as a pre-operation to the scale, and the output side remains constant at N. The width of the filter always changes inversely proportionally to the scale factor, i.e W/S, and the cost per data element remains the same. Total cost becomes

$$TotalCost = (NS)^2\left(\frac{W}{S}\right)^2C$$

which reduces to:

$$TotalCost = N^2W^2C$$

For the case where the output size is smaller than the display size, computation cost decreases linearly with the scale factor. Where the output has to be clipped to the display size, the computational cost is

$$TotalCost = N^2\left(\frac{W}{S}\right)^2C$$

The total computation cost remains constant as scale varies, but communication costs rise with decreasing image scale, and more data has to be moved to allow the 2D filters to cover more data points as they grow larger. Using clustering, the smaller the scale the fewer the number of output scanlines, and the greater the number of processors working on each output scanline. With more processors working on a problem of constant computation cost, the total execution time should reduce. However, because there is greater data movement, communication costs dominate the total execution time when scale factors are small.

For total cost, where the output side remains fixed, computation cost increases inversely proportionally with the square of the scale factor, but because N is less, communication costs should be proportionally less.

5.5.6 Theoretical comparison of 1D and 2D algorithms

For a fixed sized output, the cost of the 2D algorithm is W/S times more expensive than the 1D algorithm. When the whole image is scaled and the output size varies, the 2D algorithm is only W times more expensive than the 1D algorithm. Actual execution times of the two algorithms will be strongly affected by the communication costs, and the cost of the transpose used between passes of the 1D algorithm.

5.5.7 Results

Figures 47, 48, and 49 show the execution times for three different image sizes over a range of scale factors. When scaling a 512x512 image, the 2D algorithm takes less time than the 1D algorithm for most scale factors. For this image size, the transpose overhead of the 1D algorithm causes its execution time to be above that of the 2D algorithm. The 2D algorithm is less stable however. For small scale factors there is a rapid increase in execution time with decrease in scale factor.

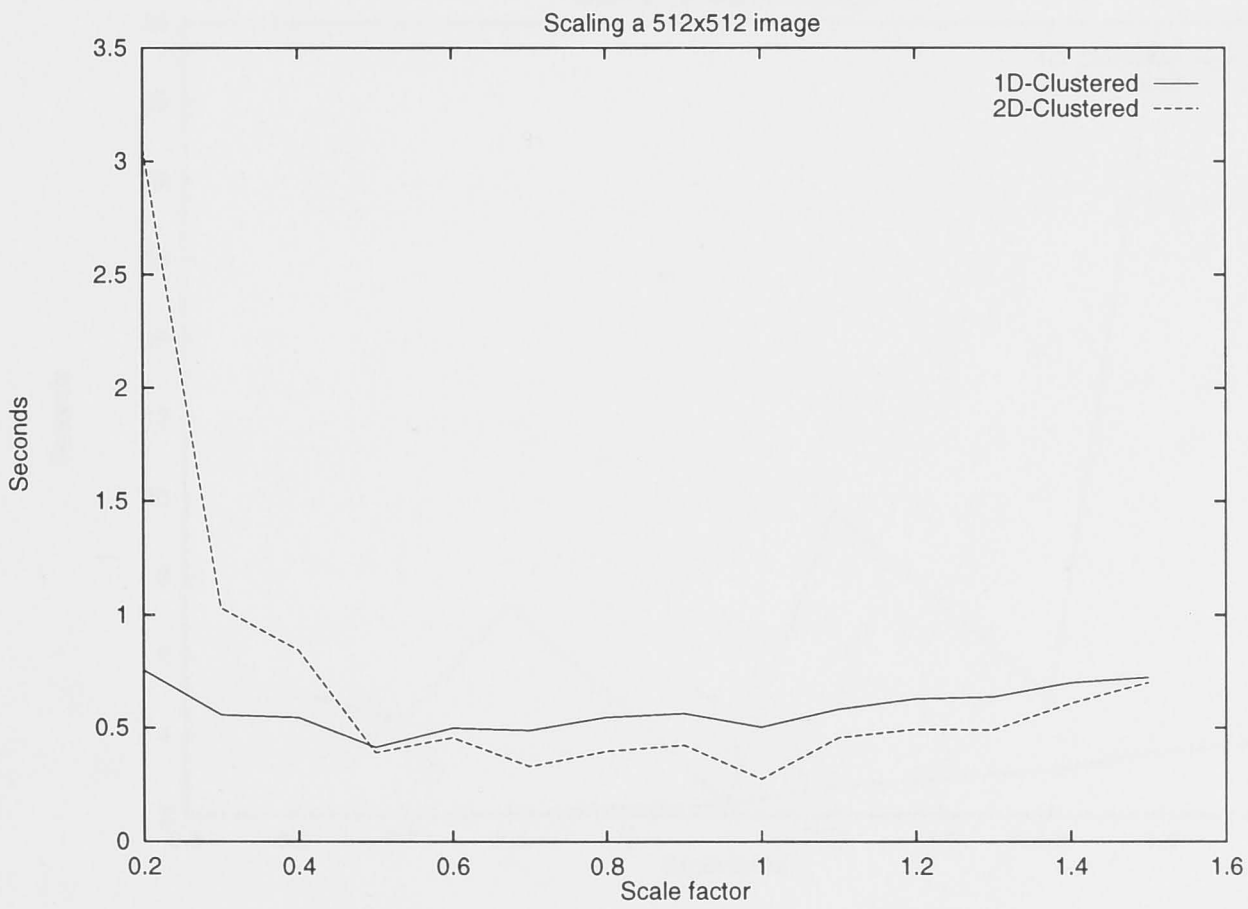


Figure 47 Execution time for scaling 512x512 image

For a 1024x1024 image, the 1D algorithm performs better on average than the 2D algorithm. The 2D algorithm is erratic, with times above and below that of the 1D algorithm. The unstable nature of the 2D algorithm is most likely due to communication

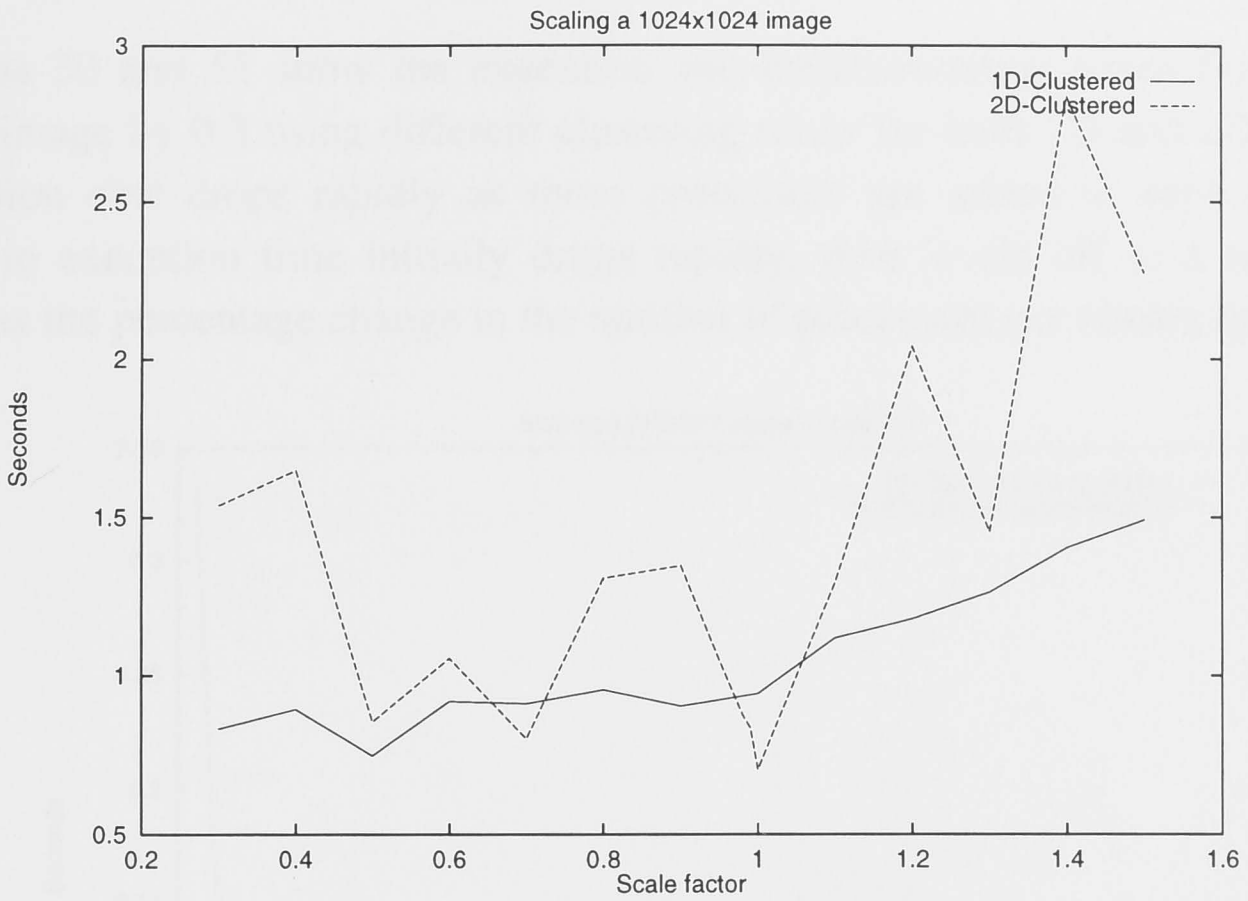


Figure 48 Execution time for scaling 1024x1024 image

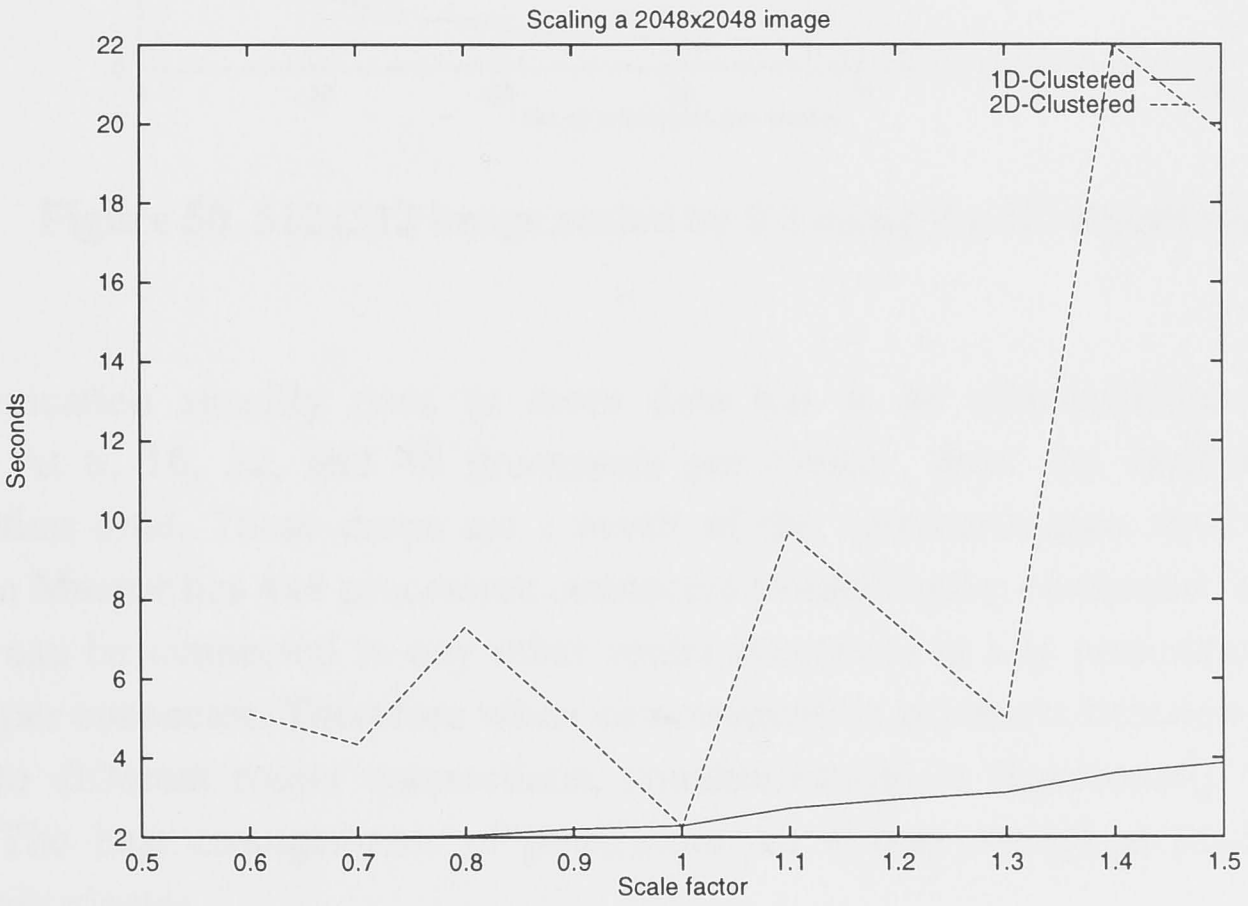


Figure 49 Execution time for 2048x2048 image

times varying. With different scale factors, different processors within the machine are assigned to each cluster. Depending on the relative position of the processors within each cluster, and the location of the clusters, the cost of communication can vary significantly.

For the 2048x2048 image, the results for the 2D algorithm become even more varied, and are substantially larger than that for the 1D algorithm for most scale factors.

Figures 50 and 51 show the execution and communication times for scaling the 512x512 image by 0.3 using different clustering ratios for both 1D and 2D algorithms. Computation cost drops rapidly as more processors are added to each cluster. The decrease in execution time initially drops rapidly, then levels off to a more gradual decrease as the percentage change in the number of processors per cluster decreases.

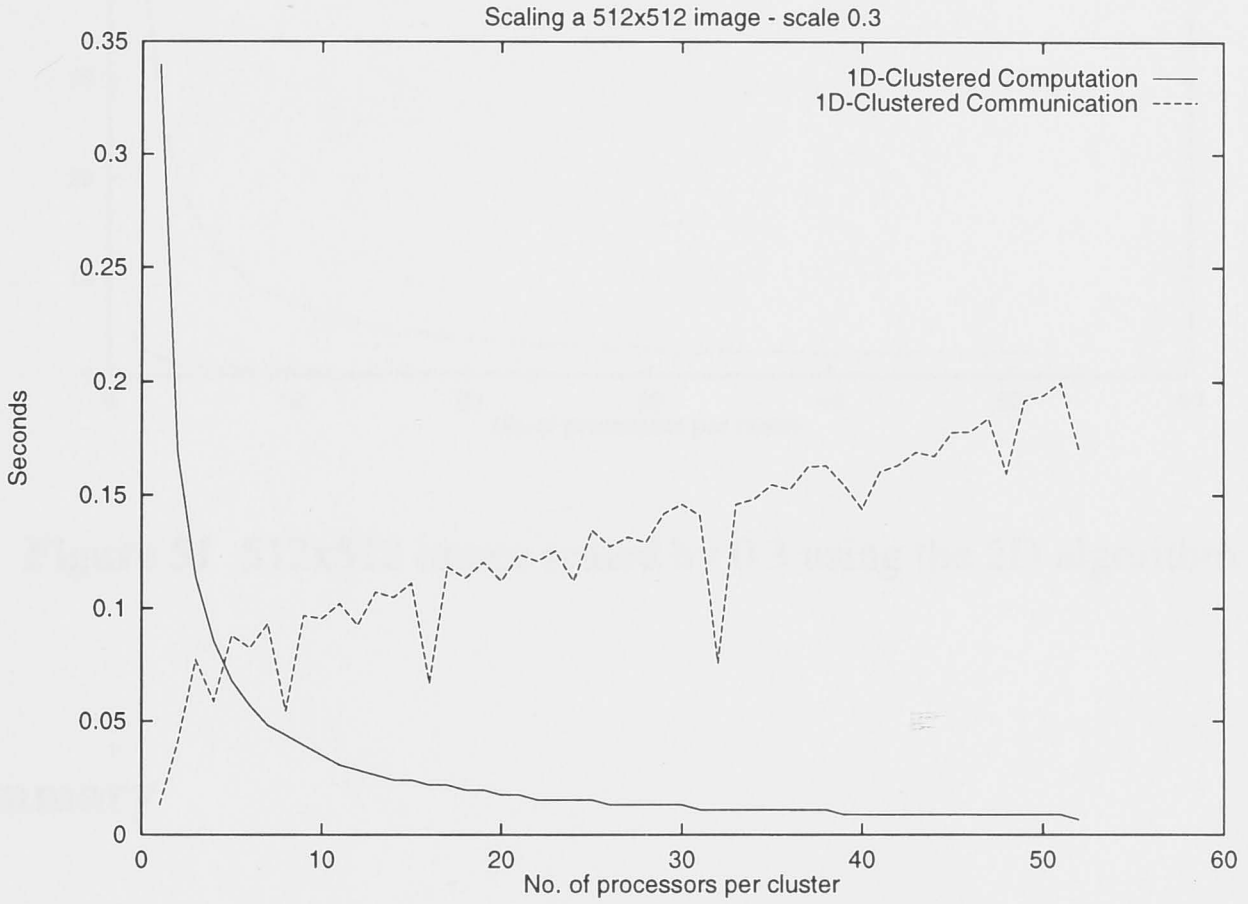


Figure 50 512x512 image scaled by 0.3 using the 1D algorithm

Communication steadily rises as more data has to be distributed among more processors. At 8, 16, 32, and 48 processors per cluster, there are sudden drops in communication time. These drops are a result of the communication structure of the Maspar. The Maspar has 4x4 processors connected to each router connector. Each router connection can be connected to any other router connector at any particular time, one connection per connector. Therefore when communication is always between processors connected to different router connections, communication is significantly faster than otherwise. The best arrangements of processors per cluster occurs at multiples of 8 processors per cluster.

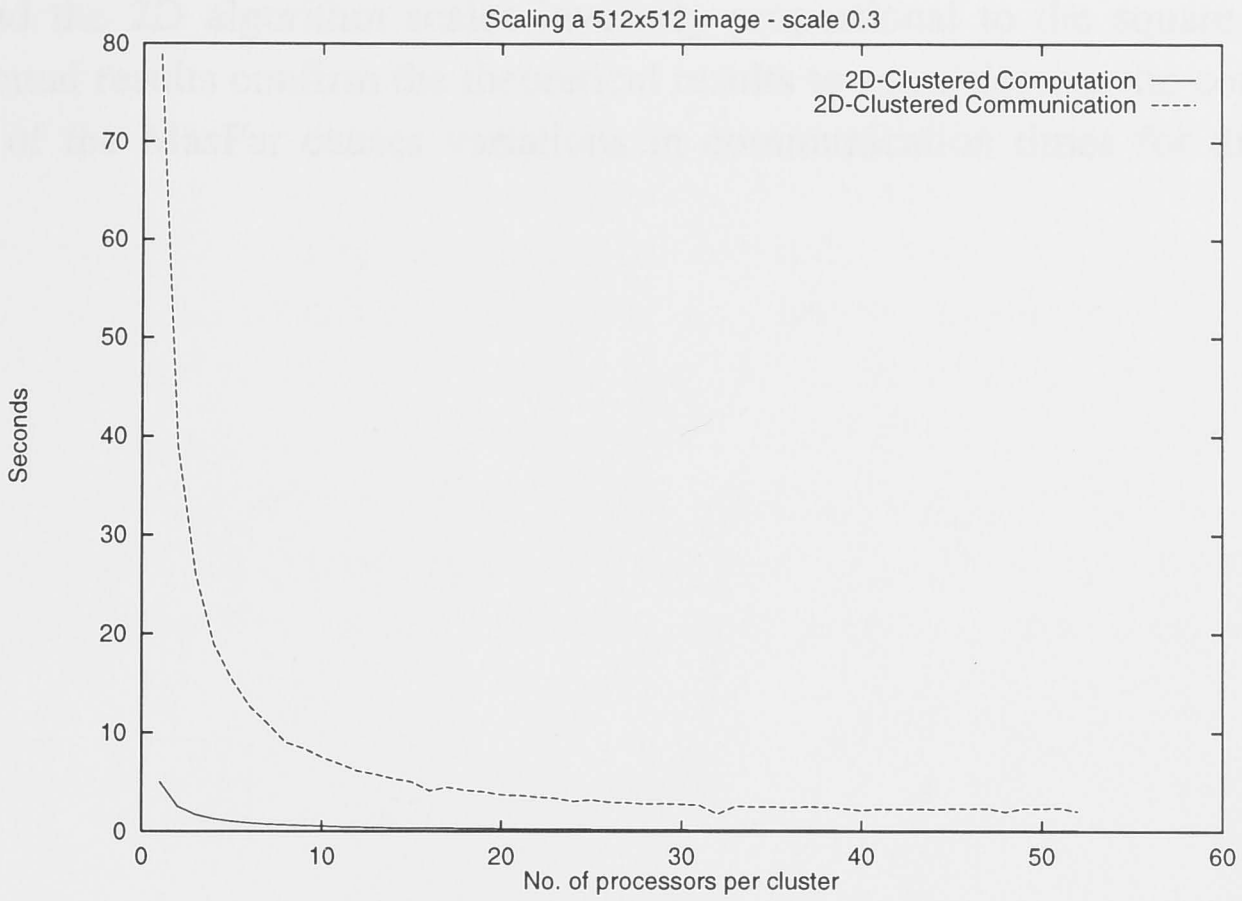


Figure 51 512x512 image scaled by 0.3 using the 2D algorithm

5.6 Summary

This chapter has addressed filtering issues associated with geometrically scaling for visualizing digital raster maps (DRM). The importance of evaluating reconstruction filters using both subjective and objective methods has been argued, and a perceptual experiment described for subjective selection of a filter for scaling DRM. Space-variant filters have been considered for highlighting critical features such as text, roads, and other discontinuity information, but it is found that a fast space-invariant filter provides adequate enhancement of critical features without too much degradation of other features.

The perceptual experiment has led to a novel filter evaluation method based on flipping spatially aligned images, to cause differences to be seen as motion. The experimental results point towards a filter with high edge sharpening characteristics, which enhances the DRM during scaling.

To address the speed requirements necessary for supporting a more general framework for interactively filtering large DRM, a parallel clustering scheme has been described. This scheme increases the speed of parallel filtering algorithms by making groups of processors perform the computation of one processor. To demonstrate the power of clustering, two parallel algorithms have been described: a two-pass 1D filter based algorithm, and a single-pass 2D filter based algorithm. Analysis of the clustered

scaling algorithms reveals that the 1D algorithm scales linearly with decrease in scale factor, and the 2D algorithm scales inversely proportional to the square of the scale factor. Actual results confirm the theoretical results to some degree; the communication structure of the MasPar causes variations in communication times for different scale factors.

6 INTERACTIVE STEERING OF IMAGE FILTERING

In this chapter we show how interactive visualization can support a general framework for space-variant filtering of images. The framework exploits image masks to carry reference information, such as spatial frequency content or discontinuity maps, from which required filter kernel characteristics can be determined. Reloadable filter kernels are realized within a parallel tool-kit, where parallelism up to the number of image scanlines is supported. The framework is general for multi-dimensional images for linear separable filters. Filter kernel images, carrying current or historical information about the space-variant filter applied to the data, are visualized to help enable the expertise of specialists or interpreters by providing direct feedback.

6.1 Space-variant filtering approach

In many space-variant filtering operations it is desirable to preserve a record of the amount and type of filtering performed on each image. Often when a cascade of filtering operations have been performed on an image, the user is left with little idea about what filtering has been performed on each pixel. Hence, the fidelity or accuracy of the final image is unknown. Most spatial filter kernels are approximations to some ideal filter and thus introduce errors at each filtering step. A cascade of such filtering operations may compound the errors introduced by each step.

6.1.1 Using masks to carry filter information

Here we take the approach of developing a space-variant filtering framework for image-based algorithms where fidelity filtering issues are important. The framework simplifies and standardizes control of the spatial variation of filters over images, and provides mechanisms of feedback that show the amount and type of filtering performed by single and cascaded filtering operations. The type of filter used, and the method of filtering, are largely independent of the framework structure but their interface must conform to its specified format. The framework provides two models for space-variant filtering, one allowing the spatial variation of the filter to be pre-computed, the other computing it as part of the filtering operation and generating a mask to show the filter variation. Separate masks are used to carry filter operation information and filter history information. A filter mask for a particular image is an identically sized image where each pixel holds information about the filtering on the corresponding image pixel (filter masks are developed further in Chapter 8). Figure 52 shows the two models for space-variant filtering provided by the framework.

The filter operation mask can be used to specify how much filtering is to be, or was performed on each input pixel. The filter operation mask can be used for either

controlling a space-variant filtering operation, or showing what filtering an operation has done. The filter history mask is similar in form to the operation mask, but shows how much filtering has been performed on each output pixel from a series of filtering operations. For some types of space-variant filtering operations, such as types of geometrical distortion or warps [Wolberg, 1990], the variation of the filter over the image is based on modelling parameters. In these cases filtering is best driven directly from the model parameters, and the operation mask that defines the amount of filtering per output pixel can be generated as the operation runs. Visualization of this mask gives the user feedback on how the operation is working, how much filtering is being performed, and if aliasing is introduced. This model of filter operation is shown in Figure 52(a). For other types of space-variant filter operations, the filter variation is derived from data properties, or specified directly through user interaction. For this model of operation the mask is generated outside the filter operation and then used to control it. The operation mask provides a clear interface between the filtering operation, and the data properties or user driving it. This model of a filtering operation is shown in Figure 52(b).

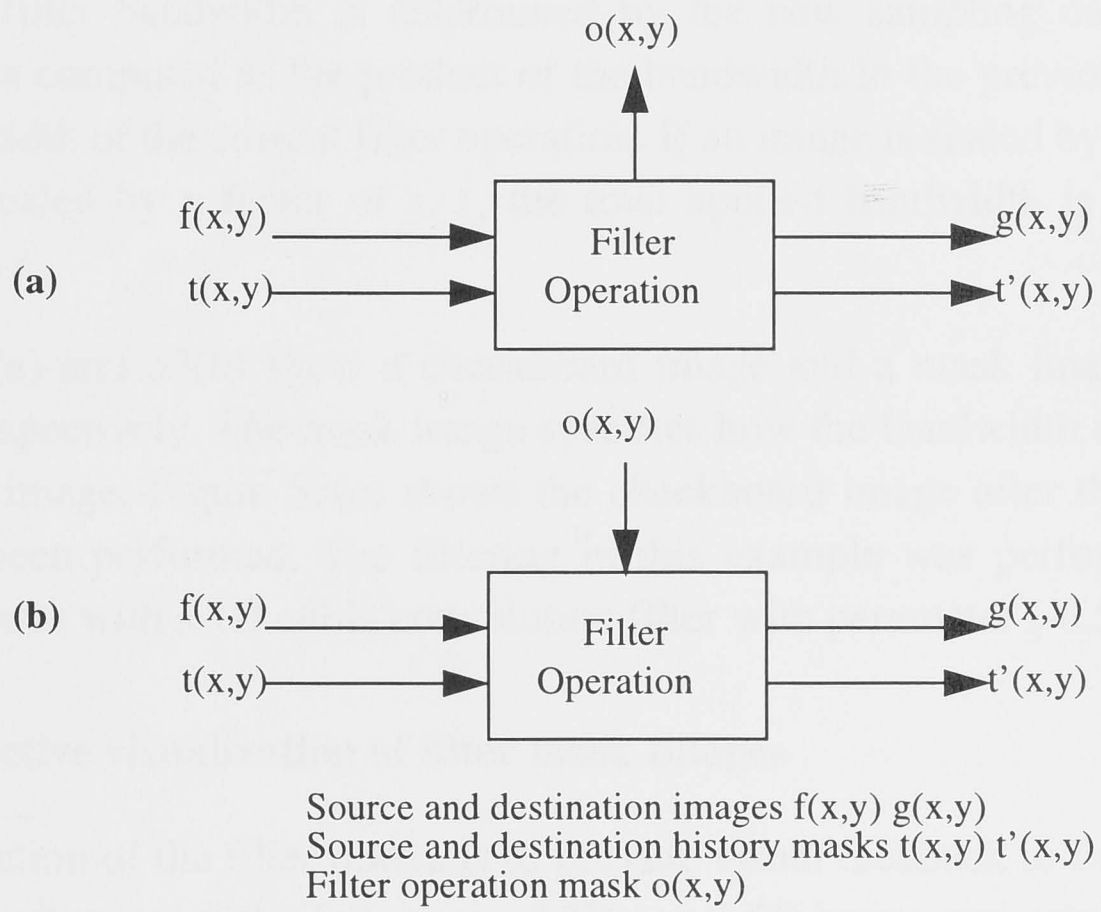


Figure 52 Filter operation models. a) Filter operation mask generated by filtering operation. b) filter operation mask driving filter operation.

The filter history mask contains a history of the filtering performed on an image. If the filtering is performed only on the original image grid (in-place filtering), the history mask will be a composition of the filter operation masks for all the filtering steps. If an operation warps the image and resamples it to a new grid however, the history mask has to be transformed and resampled with the image, before the filtering due to the current operation can be added. When several filtering operations are performed on an image the history mask must be successively passed through each operation and updated

progressively. Interactive visualization of the history mask allows a series of filtering operations to be evaluated and an indication of the fidelity of the final image obtained.

The format for representing filter kernels in the filter masks is a compromise between efficiency and generality. In theory the representation should be general enough to cope with any type of finite impulse response filter shape but in practice this would require an unwieldy representation. We have decided to use a simple mask format, restricting all filtering operations in a cascade to a single type of filter, which means we need only store the variation of bandwidth or the amplitude of the filter. The exact size and shape of the filter used at any spatial location is derived from a priori knowledge of the filter together with the bandwidth information. The type of filter for a single operation or cascade of operations can be selected by a user but the same type of filter must be used for each operation in a cascade. In addition to simplifying the requirement for preserving mask values, this restriction also simplifies the compilation of filter history masks. Since we are only storing the bandwidth of the filter over the image, the history mask values will be proportional to the total bandwidth of the filters applied to the image, and calculated according to the type of filter operation. For example, in a series of a spatial warps where filter bandwidth is determined by the new sampling density, the filter history mask is computed as the product of the bandwidth in the previous history mask and the bandwidth of the current filter operation. If an image is scaled by a factor of $1/2$, then further scaled by a factor of $1/3$, the total applied bandwidth is proportional to $(1/2 \times 1/3) = 1/6$.

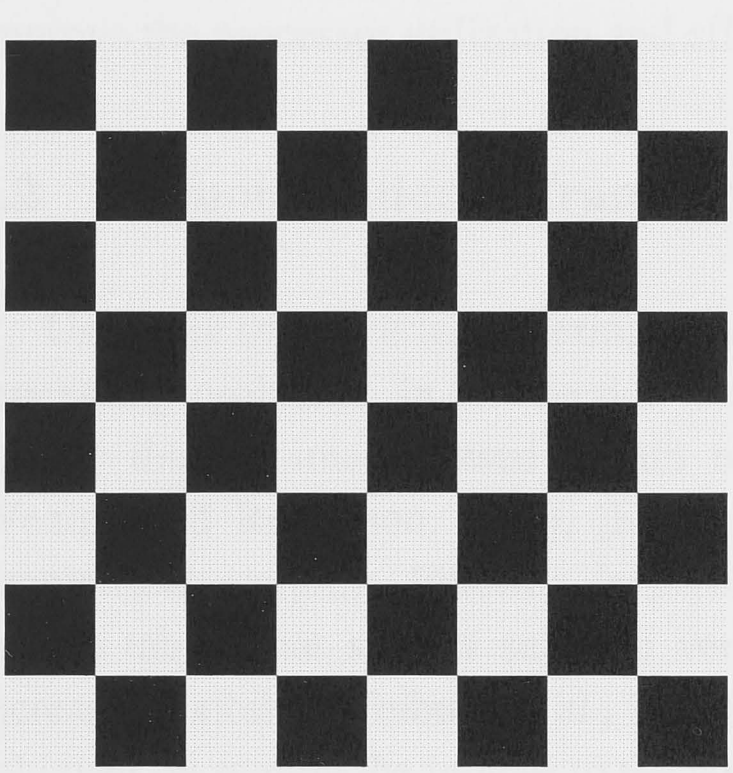
Figure 53(a) and 53(b) show a checkboard image and a mask image containing a hemisphere respectively. The mask image specifies how the bandwidth of the filter is to vary over the image. Figure 53(c) shows the checkboard image after the space-variant filtering has been performed. The filtering in this example was performed using two orthogonal passes with a 1D cubic convolution filter with parameter (-0.5).

6.1.2 Interactive visualization of filter mask images

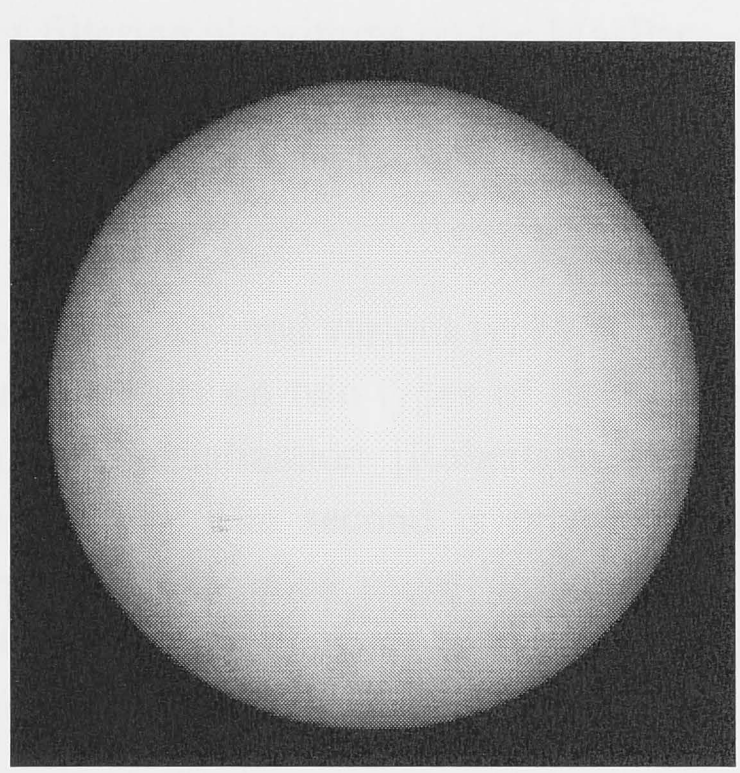
A key function of the filter masks is to provide visual feedback to the user. Because the mask is another image the full range of 2D and 2.5D image presentation options can be used.

With many filtering operations, interactive tuning of filter parameters and operation parameters can be desirable [Anderson and Netravali, 1976]. Automatic determination of the best space-variant filtering for visual quality is difficult because of the likely range of viewing conditions and the complexity of the human visual system. Providing the user with a means of controlling the filtering, even if it is only to tune up an initial computed guess, is a more practical approach.

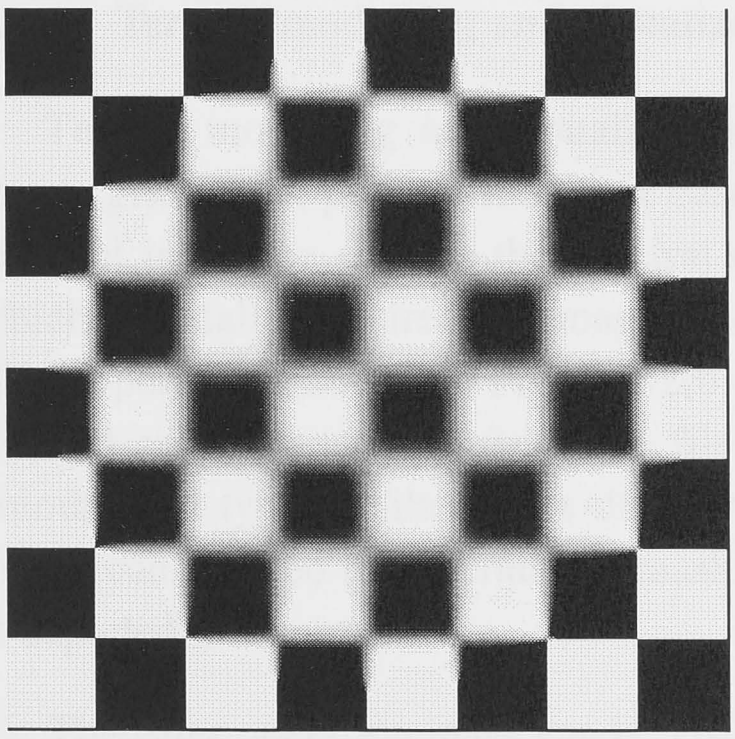
We provide three main forms of interaction based on developments in Chapter 3. The first two are global shifting and global scaling, which act on all the values in the filter operation mask. The values for the filter operation mask can be initially derived from properties of the data or operation. These values can provide a starting point from which the user can then start interacting. The third form of interaction, lookup table manipulation, allows more detailed control of filter variation over the image. A computed guess determines the initial amount of filtering per pixel, but lookup table manipulation allows these individual values to be adjusted. Our approach is to allow the user to interactively 'play' with the lookup table, using convenient graphical tools, and observe the results. The aim is to achieve intuitive hand-eye coordination to enable the full set of skills an observer develops from real world experience [Robertson, 1991].



(a) Checkboard image



(b) Hemisphere filter mask



(c) Blurred checkboard image

Figure 53 Checkboard blurred according to a hemisphere filter mask

6.1.3 Massively parallel implementation

Filtering is a costly operation when performed on large raster images of millions of pixels. Space-variant filtering is even more so, especially in such operations as geometric warps where filter sizes can grow very large. Massively parallel computer architectures can provide close to real time performance on these operations and can be scalable with the problem size [Vezina and Robertson, 1992a]. With many filtering operations interactive user tuning of the parameters of both the filters and the operations are essential for producing the best image for human observation. In this work we have used an 8k processor massively parallel computer, the MasPar MP1, to achieve interactive or near interactive performance to illustrate the benefits of interaction with filter parameters. The virtualization scheme, developed in Chapter 5, Section 5.5, helps to improve the processor utilization and efficiency of filtering algorithms on the MasPar.

6.2 Demonstration of framework

Using the space-variant filtering framework we have explored various types of image processing problems. In this section we summarize results on two types of tasks. The first is that of image based terrain modelling and visualization [Robertson, 1987], [Vezina and Robertson, 1991], [Kaba and Peters, 1993]. The second is image-based geometrical warping. Filtering is performed for both examples using pre-computed lookup tables that store a sampled version of the filter function, to reduce cost and increase flexibility [Feibush et al., 1980], [Ward and Cok, 1989]. Variation of the filter is achieved by scaling the indices to the table.

6.2.1 Terrain modelling and visualization

Terrain modelling and visualization is characterized by filtering on the original image grid. Each pixel is modified based either on properties of the data, or on some external influence such as the spectral or spatial characteristics of the imaging system which originally acquired the image. These types of operations are best modelled by the filter operation model of Figure 52(b). A filter operation mask is derived by some method and then used to drive the filtering over the image. For many operations the amount and type of filtering is directly data dependent. In these cases analysis of the image can be used to generate the mask which in turn drives the filtering. This scenario is depicted in Figure 54.

The filter operation mask can be visualized to gain a better understanding of both the data and the filtering operation, and as feedback to the user for interaction with the filtering. The user either controls the parameters for mask generation or manipulates the mask image before it drives the filtering. This approach is illustrated using a simple test image with added noise. This noisy image is blurred or low-pass filtered in a space-

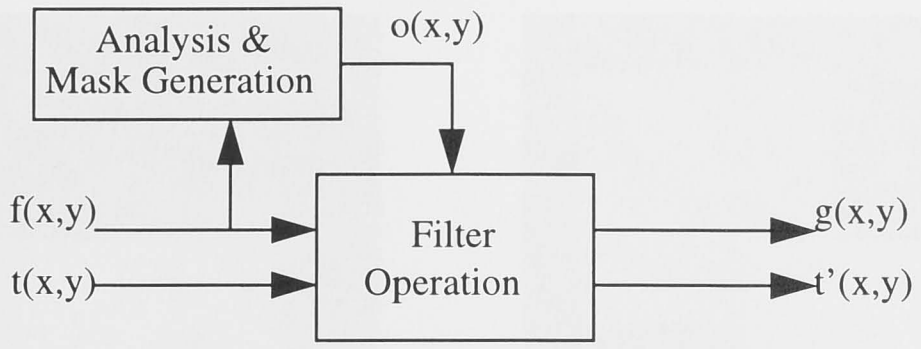


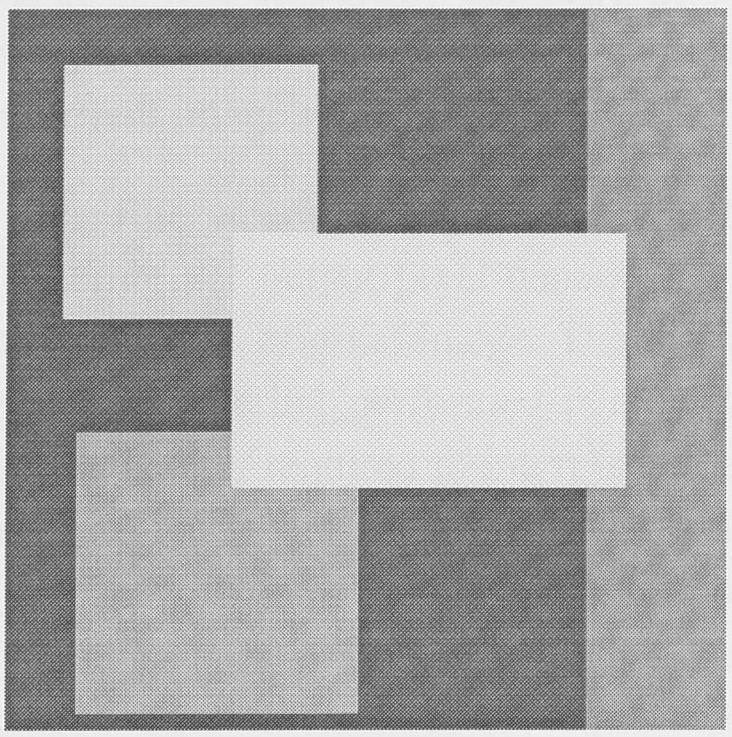
Figure 54 Filter operation template with mask generated from image properties.

variant way to reduce the visual effects of the noise while attempting to preserve the sharpness of edges [Wahl, 1987]. It uses low-pass filtering in relatively flat regions to remove the effects of noise, while in regions containing edges the filter has a higher cut-off frequency, Figure 55(a) and 55(b) show respectively the original image and the image after noise was added. Figure 55(c) shows a filter operation mask derived from the noisy image by a simple gradient and threshold method. Figure 55(d) shows the resulting space-variant low-pass filtered image.

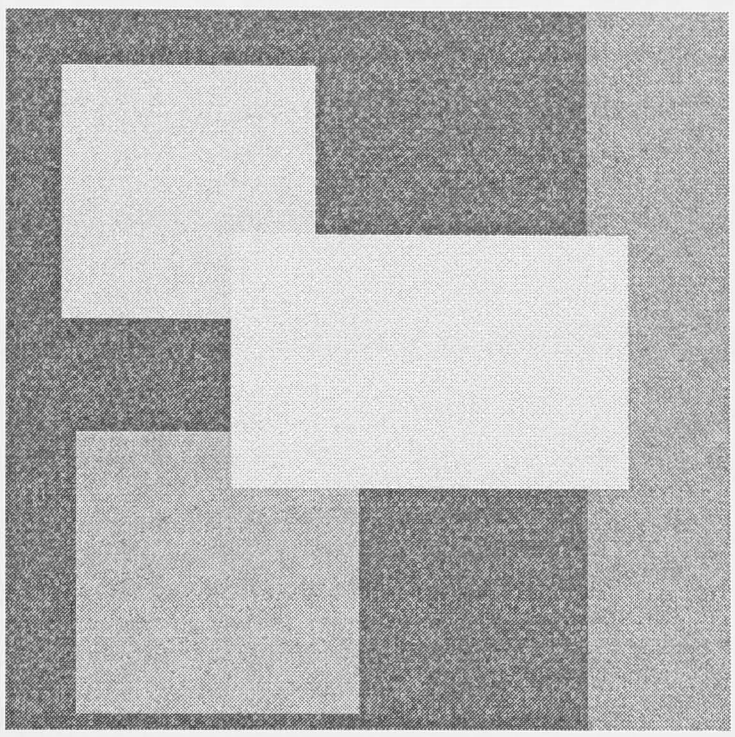
Clearly there are better methods for removing noise while preserving edges than this method, but it does show clearly how the filtering model can be used. It would be a simple task to add user control over the thresholding and/or scaling of the filter operation mask to allow control over the amount and spatial variance of smoothing, which would allow adjustment of the filtering for perceived visual quality.

A practical application of this is in digital terrain modelling as described in chapter 3. Relief shading of digital terrain data provides cues that aid in the perception of depth and solidity, which in turn allows the structural features of the data to be better interpreted and understood [Zhou, 1992]. Terrain data are generally obtained through direct measurement, often calculated from stereo pairs of images, and may be degraded in several ways. Noise can be introduced into the data during the acquisition, transmission, and storage stages. Quantization errors can also arise from mapping a continuous variable to a set of discrete values giving rise to artifacts often known as terracing [Jain, 1989].

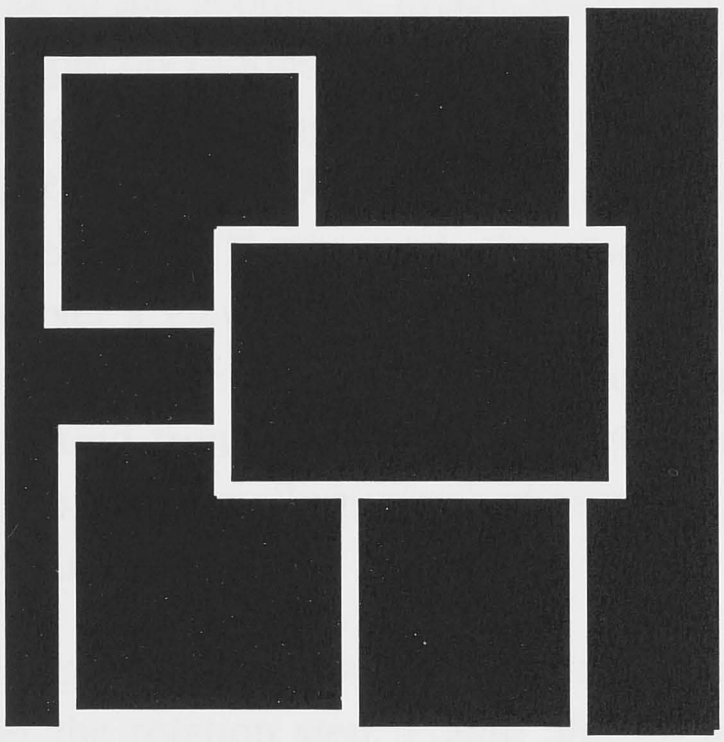
To reduce the effects of noise and quantization artifacts for image shading we perform a space-variant smoothing operation that is inversely proportional to the image gradient and which is tuned through user interaction. The examples in Chapter 3, Section 3.5 demonstrate how filter control masks are generated, visualized, and interactively applied to control the space-variant smoothing for this problem.



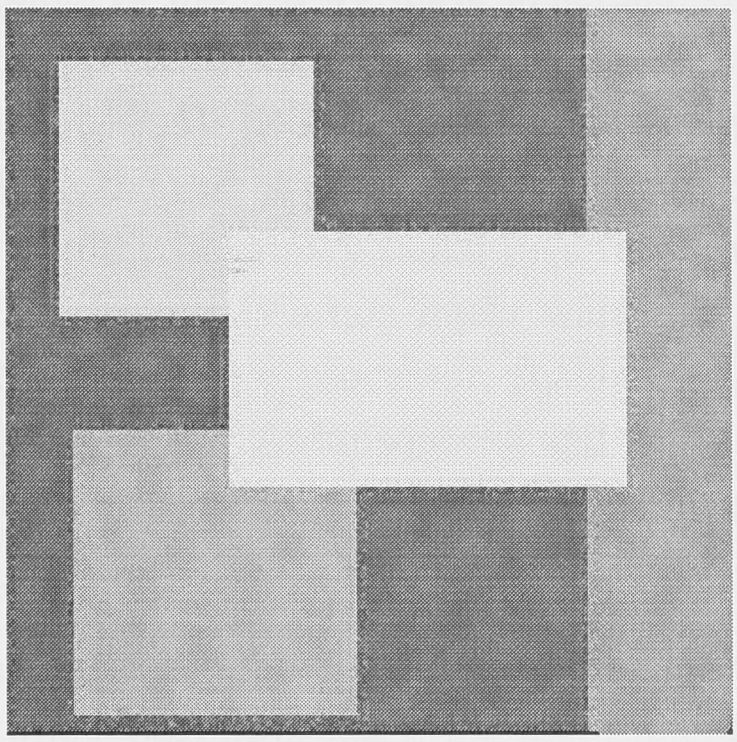
(a) Patches image



(b) Patches image with added noise



(c) Filter control mask



(d) Smoothed image with preserved edges

Figure 55 Improving the visual quality of a noisy image using a space-variant smoothing filter

6.2.2 Image-based geometrical distortion (warping)

The second application we use to illustrate the approach is image-based geometrical warping. Warping requires the image to be reconstructed, then warped and resampled to a new grid [Wolberg, 1990], [Fraser et al., 1985]. The image reconstruction, warping, and resampling is generally performed as a single operation. When performing an affine geometric image warp, space-invariant filters can be used to prevent aliasing. For non-affine image warps space-variant filters are required if aliasing is to be prevented and data integrity preserved.

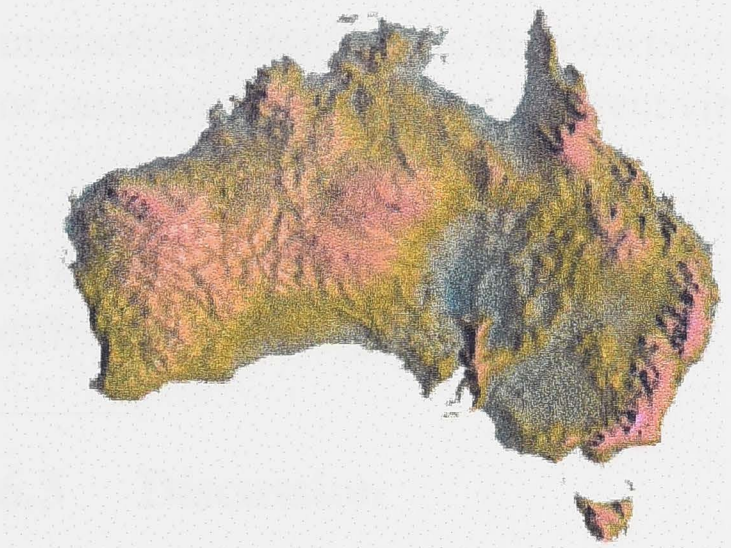
When the image is warped, the history mask must follow the image and be similarly warped if it is to portray correctly the amount of filtering per output pixel. Once the history mask has been warped, it can then be updated with the filtering information of the current image warp.

The filter operation mask can be created as part of the warping process as the amount of filtering per output pixel depends on the sampling density (or amount of stretch or squeeze) of the original image. The mask will show, after the warping is completed, the amount of filtering performed per output pixel. If the image is locally sub-sampled, filtering is only required to interpolate between existing values. If on the other hand the image is locally super-sampled, then additional low-pass filtering is needed to filter out any frequencies above the Nyquist limit of the new sample grid to prevent aliasing. Thus, depending on the definition of the algorithm, the amount of filtering is best determined from the algorithm if it includes options, or during its execution if it does not.

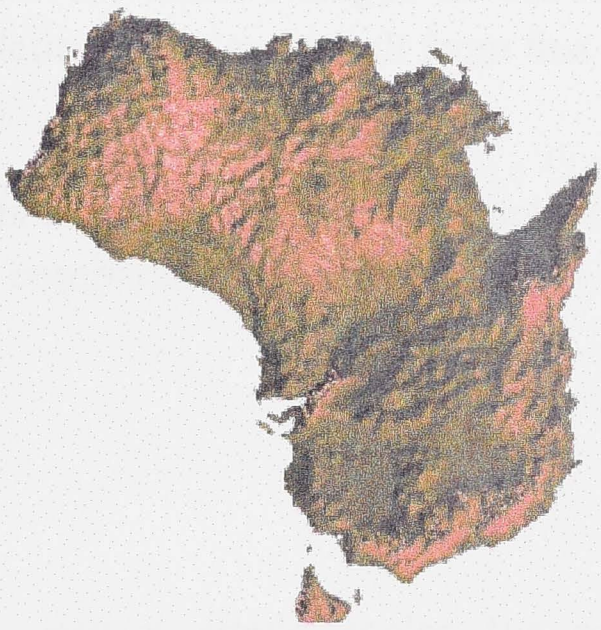
The geometric warping example shows both the relationship of the filter operation mask to the geometric image warp, and an example of the history mask generated for multiple operations. The warp is one component of a multi-stage algorithm to perform perspective viewing of surfaces [Tsui and Fletcher, 1995], [Robertson, 1987], [Vezina and Robertson, 1991]. An image rotation first aligns the user's viewpoint with the front of the image. The image is then "squeezed" to map the imaginary radial rays from the observer's viewpoint onto parallel scanlines. This squeeze is a non-linear warp. On a massively parallel machine, with each scanline mapped to a processor, scanlines are processed from front to back to generate a visibility mask. An unsqueeze and reverse rotation are usually performed on the visibility mask to return it to the original image orientation.

Figure 56(a) shows a digital terrain height image, and Figure 56(b) shows this image after the rotation step. Figure 56(c) shows the filter operation mask for this rotation. Since rotation is an affine mapping the filtering is space-invariant and the filter operation mask contains the same value for all destination pixels containing data. The squeeze operation, however, is not affine and space-variant filtering is required. The variation of the filtering for the squeeze is only along the vertical axis with filtering constant along the rows of the image. Figure 56(d) shows the image after the squeeze, and Figure 56(e) shows the filter operation mask that was generated by it. The mask illustrates clearly the variation of filtering described above. Note that the constant steps in the mask are caused by its quantization for storage and display, and are thus an artifact of an interim storage format only. The filtering across the image in fact varies continuously along the vertical axis. Figure 56(f) shows the history mask for the cascaded rotate and squeeze operations.

The rotation used is the [Catmull and Smith, 1980] multi-pass algorithm which can suffer from aliasing due to intermediate stage sub-sampling. The history filter mask values thus depend on rotation angle; larger angles produce larger history mask values because of increased possibility of aliasing artifacts. This can be used by the user to



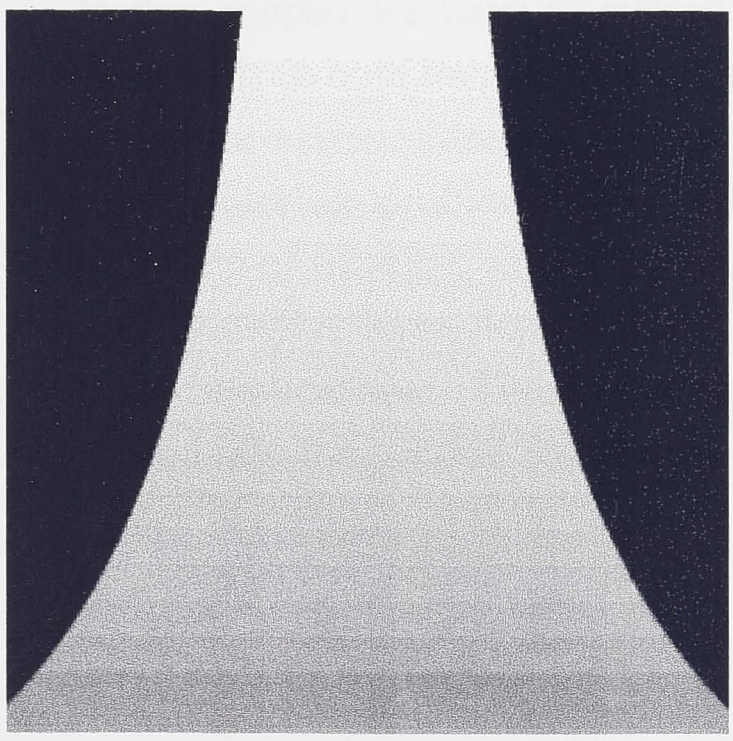
(a) Terrain data image



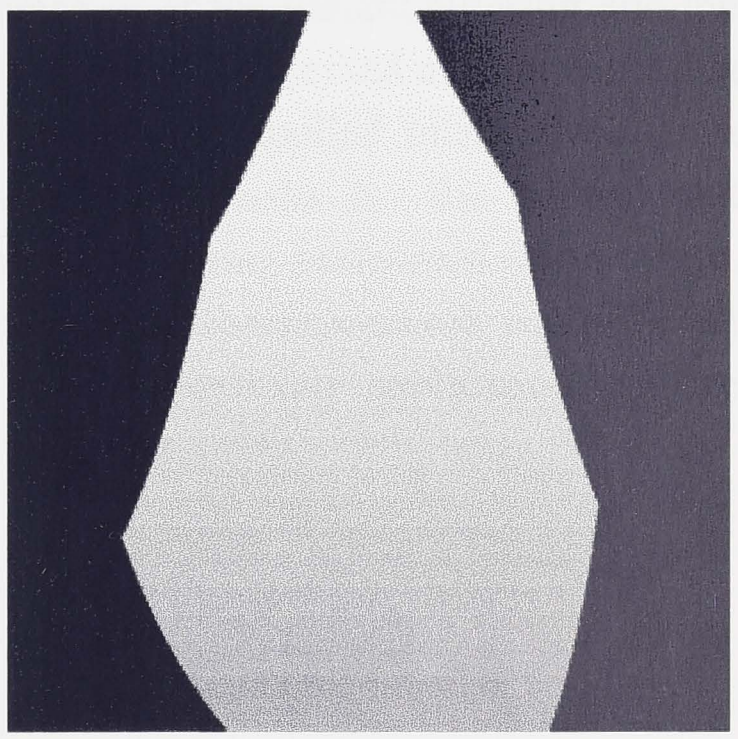
(b) Rotated image

(c) Filter operation mask for rotation

(d) Squeezed, rotated image



(e) Filter operation mask for squeeze



(f) Filter history mask for rotation and squeeze

Figure 56 Rotation and squeeze warping of a terrain image

evaluate the fidelity of the data. The squeeze operation similarly produces different levels of low-pass filtering depending on viewing angle and distance of the viewpoint from the image. Visualization of the mask, and how the mask changes as the user interactively manipulates the viewpoint, shows the fidelity of the end result from this set of cascaded operations for different viewing positions.

Interaction with the filtering for these operations is currently limited to the selection of the shape of the low-pass filter used. This allows comparison of different filters for different types of operations.

6.3 Discussion

The initial filter mask format used contains only a single scalar for each corresponding image pixel, to represent filter bandwidth information. This scalar value can be used to store a different filter property such as shape parameter (i.e for cubic or gaussian filters) or filter type where a number of different types are available. One extension to the single value mask is the addition of data discontinuity information. Data discontinuities arising from data edges, and from missing data, can be handled using methods such as filter kernels with built in extrapolation [Vezina and Robertson, 1992a]. Extending the filter masks to contain multiple values per image pixel allows a wider range of filter properties to be represented and increases the range of filter applications which can use the framework. The complexity and cost of visualization, interaction and filter history composition also increases.

6.4 Summary

In this chapter we have discussed the importance of space-variant filtering and the need for tools to both increase and improve its use. We have presented a framework that allows space-variant filters to be generated, modified, and applied interactively to a range of filtering problems. The framework uses filter masks for the control and visualization of filtering, as well as the retention of total filtering performed on an image. Several examples have demonstrated the effectiveness of the framework for different types of applications. These examples show how an existing application can be embedded into the parallel framework in a generic manner.

7 A WORKBENCH FOR SPACE-VARIANT IMAGE FILTERING

In this chapter a software workbench that simplifies the task of implementing, controlling, and visualizing space-variant image filters is developed.

A filter's behaviour over an image is dictated by the parameters that control it. The values of each parameter can be data, geometric, algorithmic, or user dependent. We call this the parameter's source-dependence. Parameters can also vary over any number of image dimensions (a spatially-invariant parameter has dimensionality of 0). We call this the parameter's dimensionality-dependence.

Using the parameter dependence classification scheme as a base, the software workbench provides tools that allow visualization of filter properties, and where appropriate, interactive user control.

A median filter is a simple example of a data dependent (adaptive) filter. We make explicit the components of data analysis and filtering, and use it to show how filter properties can be visualized. A space-variant band-pass filter, used in seismic data processing, shows how user interaction can be incorporated into the workbench. Finally, a simple geometric warp shows how geometric (and algorithmic) dependent filters benefit.

In Chapter 6 and [Moore and Robertson, 1995], two dimensional image masks are used to carry reference information to control the variation of low-pass smoothing filters, and filter history information to determine image fidelity after multiple operations. For each pixel in the source data, a control image pixel specifies the amount of low-pass filtering to be performed. Control images have been visualized to show the filter's spatial variation.

In this chapter we derive a more general form of the control image which is not restricted to two dimensions and has a wider application. We then describe a more general framework, based on these control images, for implementing and controlling space-variant filters. Finally we present three examples to illustrate how the workbench can be used for different types of problems.

7.1 Basis for design

In order to support a large number of filter types, we first examine the fundamental process of space-variant filtering, and derive from it some elements common to all filters. We then exploit these common elements in the design of the workbench.

In this work we limit ourselves to spatial domain filtering. We represent the general form of a spatial domain space-variant filter by the function $t_x(f, x)$, which depends on its location $\boldsymbol{x}$, and filters the input data f at position $\boldsymbol{x}$. For a convolutional space-variant filter, for example, the function $\boldsymbol{t_x}$ equals the convolution of a space-variant filter kernel h_x with the input data.

$$t_x(f, x) = \int_{-\infty}^{\infty} f(u) h_x(x - u) du$$

For all space-variant filters, the filter function t_x can be re-written as a kernel function t'_m which depends on M parameters $\boldsymbol{m}$, which in turn depend on the function's position plus some addition information $\boldsymbol{o}$. We represent the relationship between the parameters $\boldsymbol{m}$ and their source via the control functions $\boldsymbol{d}$. This is shown:

$$t_x(f, x) = t'_m(f, x) \text{ where } m_i = d_i(x, o_i)$$

The function d_i is a control function which computes the parameter m_i for position $\mathbf{x}$, with additional information $\boldsymbol{o_i}$. The exact source and type of $\boldsymbol{o_i}$ depends on the properties of the filter parameter. For a given filter function t_x, it may be possible to derive several different t'_m functions.

By making M equal to the number of data coordinates and each function d_i return the i'th coordinate, the two forms of the filter function become identical. However, a simpler form of t'_m can usually be found which makes the $\boldsymbol{d}$ functions and control parameters m_i more useful.

The number and form of the $\boldsymbol{o_i}$ parameters, together with the dependence of the $\mathbf{x}$ coordinates, can be used to classify each control function, and thus classify the filter function's parameters. We classify the parameters according to two forms of dependence, the dimensionality-dependence and the source-dependence. The dimensionality-dependence of a parameter is the number of source data dimensions on which the parameter value depends. A global parameter, which does not depend on any coordinates, is dimensionally independent or has a dimensional-dependence of zero. A parameter that varies in only one direction has a dimensional-dependence of one, and so on. The source-dependence describes what the $\boldsymbol{o_i}$ parameters depend on. We classify the sources into four categories: data dependence, algorithm or system dependence, geometric dependence, and user dependence.

7.1.1 Parameter dependencies

Data dependent filter control parameters are those whose spatial variation is derived from the source data. Filters with data dependent control parameters are also known as adaptive filters, as their action adapts to data characteristics. Example data dependent filtering problems include noise removal, edge and feature enhancement, and missing

data interpolation/extrapolation. The flow of information from the source data, into a data dependent filter control function and the parameter from the control function into the filter, is shown in Figure 57.

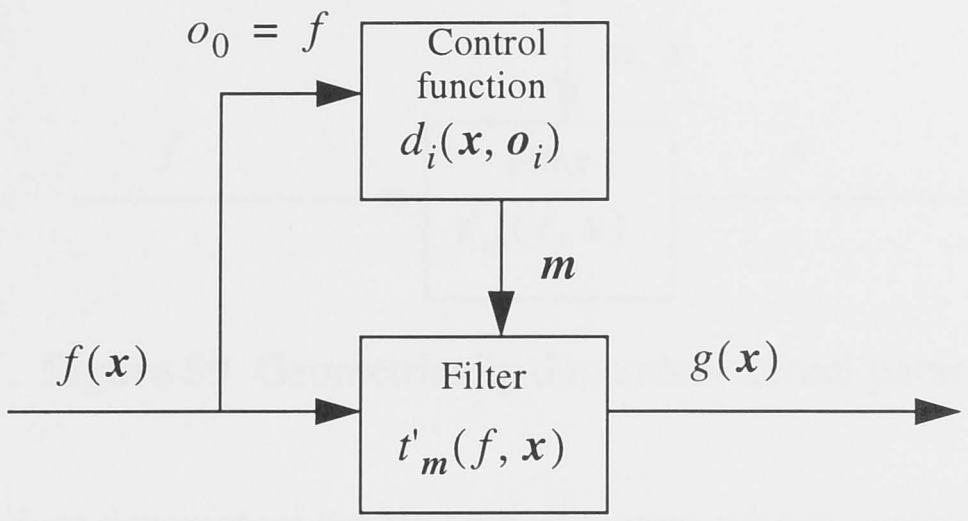


Figure 57 Data dependent kernel parameter control function

Algorithm or system dependent parameters derive their values directly from the algorithm controlling them, or from the parameters of the system of which the filtering forms a part. Figure 58 shows the control function embedded into a larger system. The system may or may not have its own inputs, and the filter may or may not be contained within the system as well. We show it as outside the system.

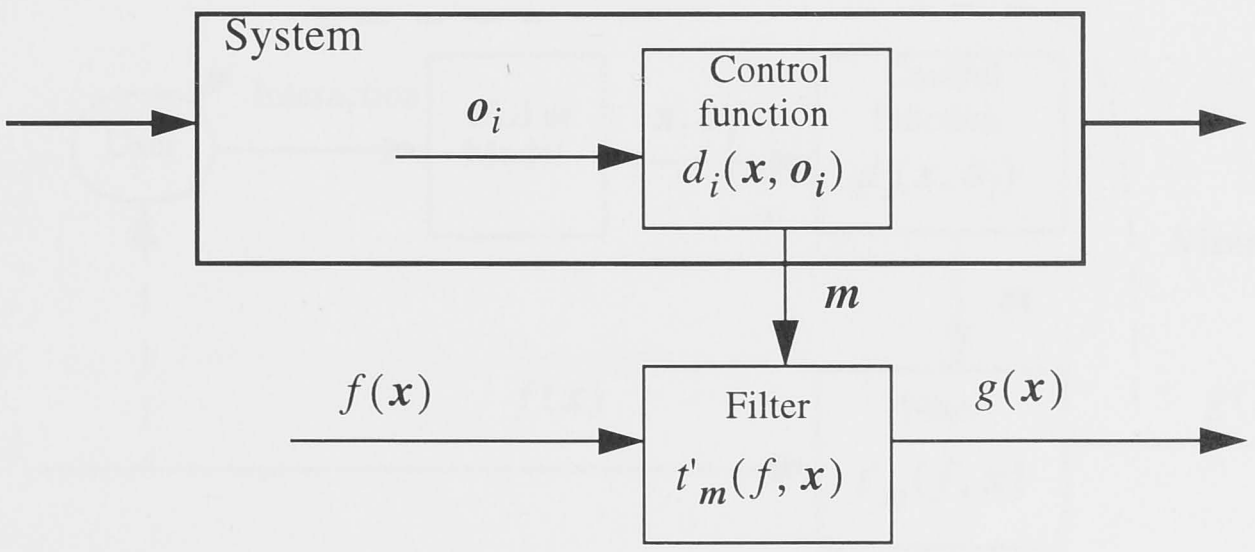


Figure 58 Algorithm dependent kernel parameter control

Geometry dependent parameters are those parameters whose spatial variation can be defined geometrically. The variation can depend on the geometry of the data capturing system, spatial transformation, data tiling, or any other geometrically derived process. Specific examples include coordinate resampling, image registration (warping), and texture mapping. The filter parameters include a coordinate position as well as the filter controlling parameters. See Figure 59.

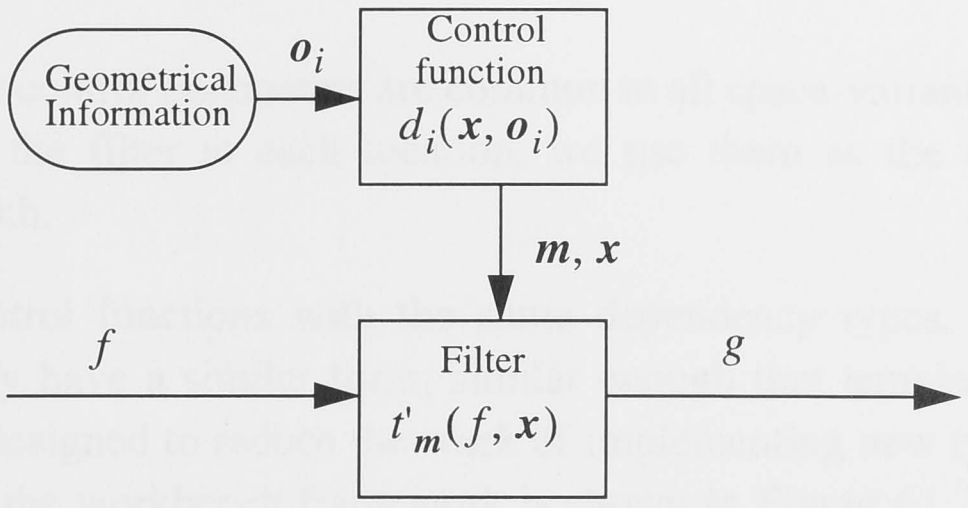


Figure 59 Geometrically dependent kernel parameter

User dependent parameters are those parameters whose variation is controlled by the user. For some applications, user dependent parameters are data dependent parameters, where the user adapts the filter to the data. For others, they are geometrically dependent parameters, where the user controls the geometry. User dependent control parameters have the potential to be applied to a wide range of problems, but existing use has been limited mainly to data enhancement problems where the results are optimized for human viewing. For the user to interact, there must be a method for interaction with some form of feedback. The interaction and visualization loop is shown in Figure 60.

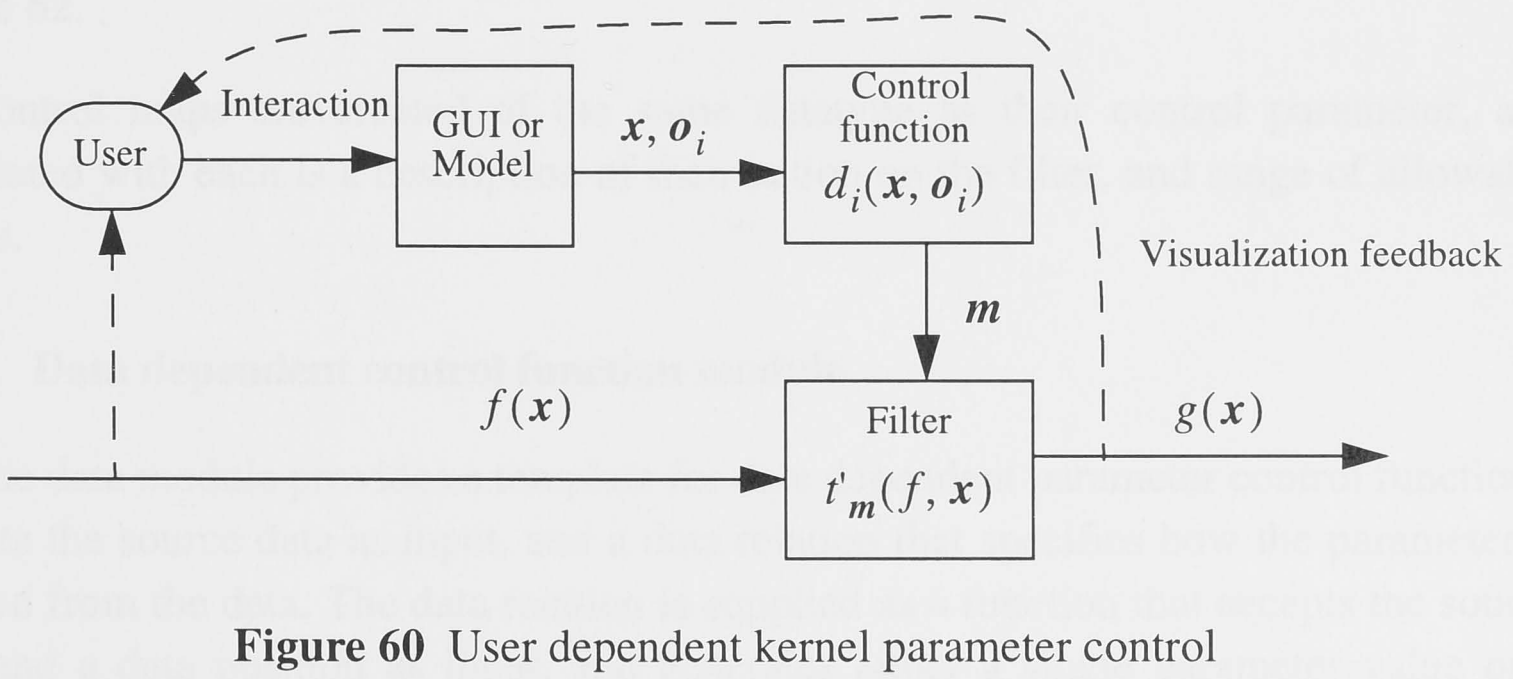


Figure 60 User dependent kernel parameter control

Control parameters are not limited to one type of dependence; it frequently occurs that a parameter has multiple dependencies.

7.2 Architecture

Since the filter control parameters are common to all space-variant filters, and dictate the behaviour of the filter at each location, we use them as the foundation for the software workbench.

Parameter control functions with the same dependency types, even for different filters, will usually have a similar form; similar enough that templates and supporting functions can be designed to reduce the work of implementing new types of filters. The basic structure of the workbench framework is shown in Figure 61. There are modules for each of the source-dependency types of the filter parameter control functions (the object-oriented design of the workbench is given in Appendix A). Each control function produces a parameter value for each coordinate in the source data. Taken together, the parameter values from a function form an image, which can be used as a lookup table or map function for that parameter. We call this image a parameter control map.

Parameter control maps have a list of dimensions over which they vary, and when filtering higher order data sets, the map's values are replicated over those dimensions in the source data for which they are not defined. For example, if we are using 3D source data with (X, Y, Z) coordinates, and a 2D parameter control map image defined over the coordinates (X, Z), then the parameter is constant along the Y axis for each (X, Z), and equals the value defined in the parameter map accessed by (X, Z). This is shown in Figure 62.

Control maps are created of the same datatype as their control parameter, and associated with each is a description of their action on the filter, and range of allowable values.

7.2.1 Data dependent control function module

The data module provides a template for data dependent parameter control functions. It takes the source data as input, and a data relation that specifies how the parameter is derived from the data. The data relation is supplied as a function that accepts the source data and a data position as input, and generates either a single parameter value or a complete parameter control map. More specific templates can be constructed that accept data relations that look at sliding windows over the data.

7.2.2 Algorithm dependent control function module

The algorithm or system module is a place holder that provides an interface for applications to directly control the filter function's parameter values.

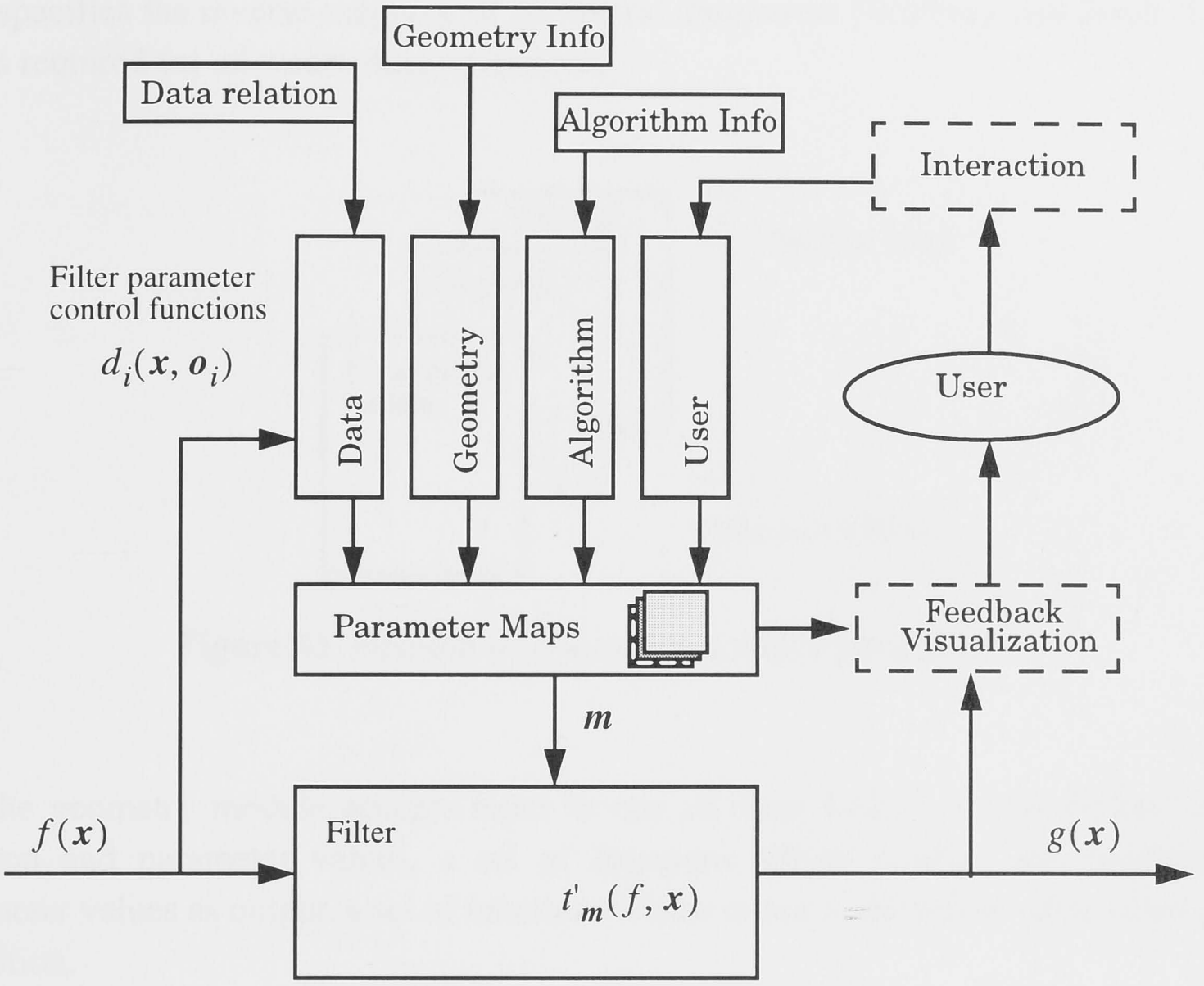


Figure 61 Architecture for workbench

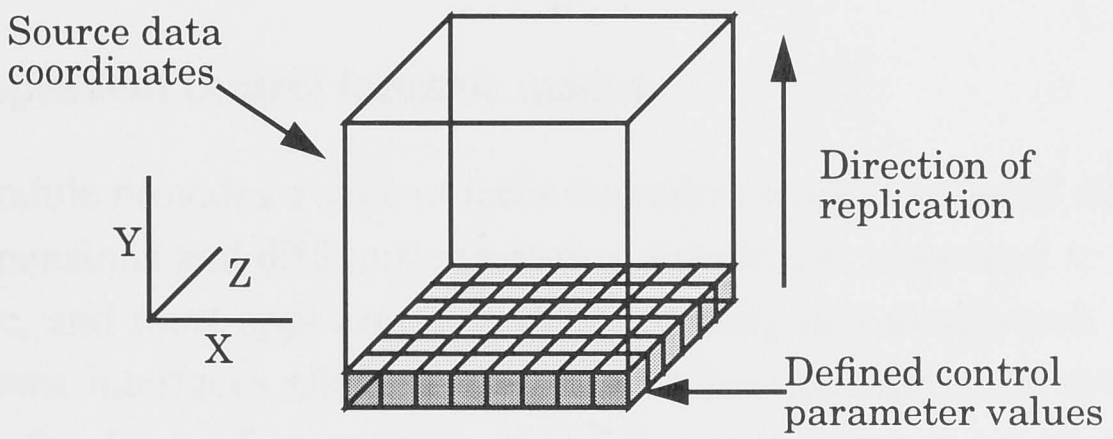


Figure 62 Replication of parameter map values in those dimensions for which they are not defined.

7.2.3 Geometry dependent control function module

The geometry module supports geometry dependent parameter control functions. These require both a position and a value to specify the nature of the filtering, and therefore produce a position map as well as a parameter control map. A position map contains the locations at which to filter the source data. Unlike a parameter map, a

position map must be the same dimensions as the destination data. The position map is usually made up of separate planes for each pixel dimension, see Figure 63. The position map specifies the inverse mapping of geometric transforms [Wolberg and Boult, 1989], and is required for all resampling operations.

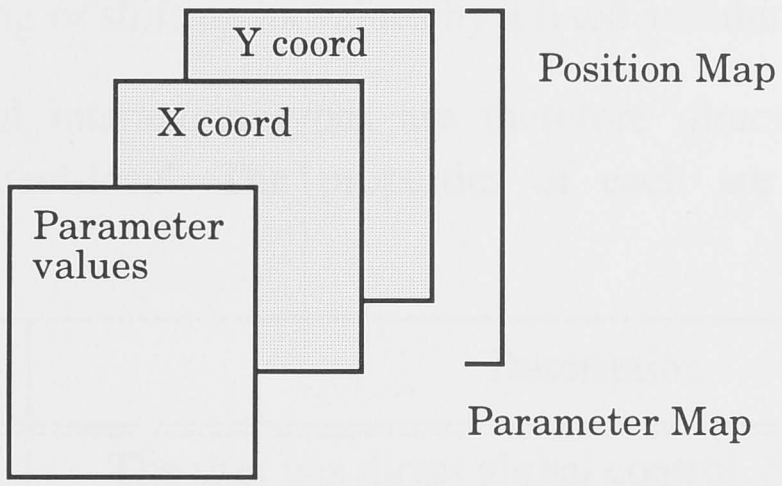


Figure 63 Position map associated with a parameter

The geometry module accepts input in one of three forms: an enumerated list of position and parameter values, a set of functions which produce the position and parameter values as output, a set of functions which return a parameter value when given a position.

The geometry module creates at least one position map when there are geometry dependent parameters. It can associate one map per parameter, or a single map common to all the geometry parameters.

7.2.4 User dependent control function module

The user module provides a suite of tools that allow a user to control filter parameters of different dimensions and different semantics. Interaction tools tend to be application domain specific, and most application developers using this workbench would need to develop their own interfaces aligned with the familiar paradigms of their domain. To help determine the form of interaction required, we describe a set of categories into which most forms of interaction can be classified.

Interaction

We classify user control of parameter maps according to what is controlled. Control by the user is either by direct methods or indirect. By either method, the user can control parameters globally over the image, or local to a region.

Direct control allows a user to control the individual values in the parameter map. Indirect control gives the user control over a secondary process or algorithm that in turn

controls the parameter map values. Indirect interaction may be used to provide better semantics for interaction then direct interaction.

Local control alters spatial variation. It allows individual values or regions of pixels in the parameter map to be controlled. Global control changes the values in a parameter map as a whole. For a parameter map with an existing spatial variation, global control may be as simple as scaling or shifting its values by a fixed amount.

The four fundamental interaction types are therefore direct-global, direct-local, indirect-global, and indirect-local. The properties of each are summarized in the following table.

How	What	Description
Direct	Global	The user has direct global control. Standard options include scaling and shifting the values in the parameter control map.
	Local	The user has direct local control. Can alter the spatial variation of the values in the control map. Rather then controlling the exact data value for each data value, a user controllable function may be used.
Indirect	Global	The user controls the parameters to a process that globally alters the values of the parameter map.
	Local	The user controls the parameters to a process that controls the local variation of the parameter map.

Next to the fundamental interaction types, the dimensionality of the parameter map plays the greatest part in determining the form of interaction. Global interaction tools can easily be made general over many dimensional types, but local ones cannot. The form of local interaction has to be matched to the dimensionality of the data. We provide simple tools for basic manipulation of zero, one, and two dimensional parameters. For zero dimensional, we provide sliders for controlling scale and offset of values. For one dimensional we provide interactive graphs to manipulate piece-wise linear and spline functions. With two dimensional parameters we are currently experimenting with manipulating spline surfaces. Figure 64 shows one of the prototype interfaces for this. Thin-plate splines have been shown to be useful for editing raster images [Lui and Bone, 1995], [Lui and Bone, 1996]. Quadratic B-splines surfaces can be created by specifying sets of surface points [Pham, 1989] and then rasterized. Both spline types can be embedded into the workbench for controlling filter variation.

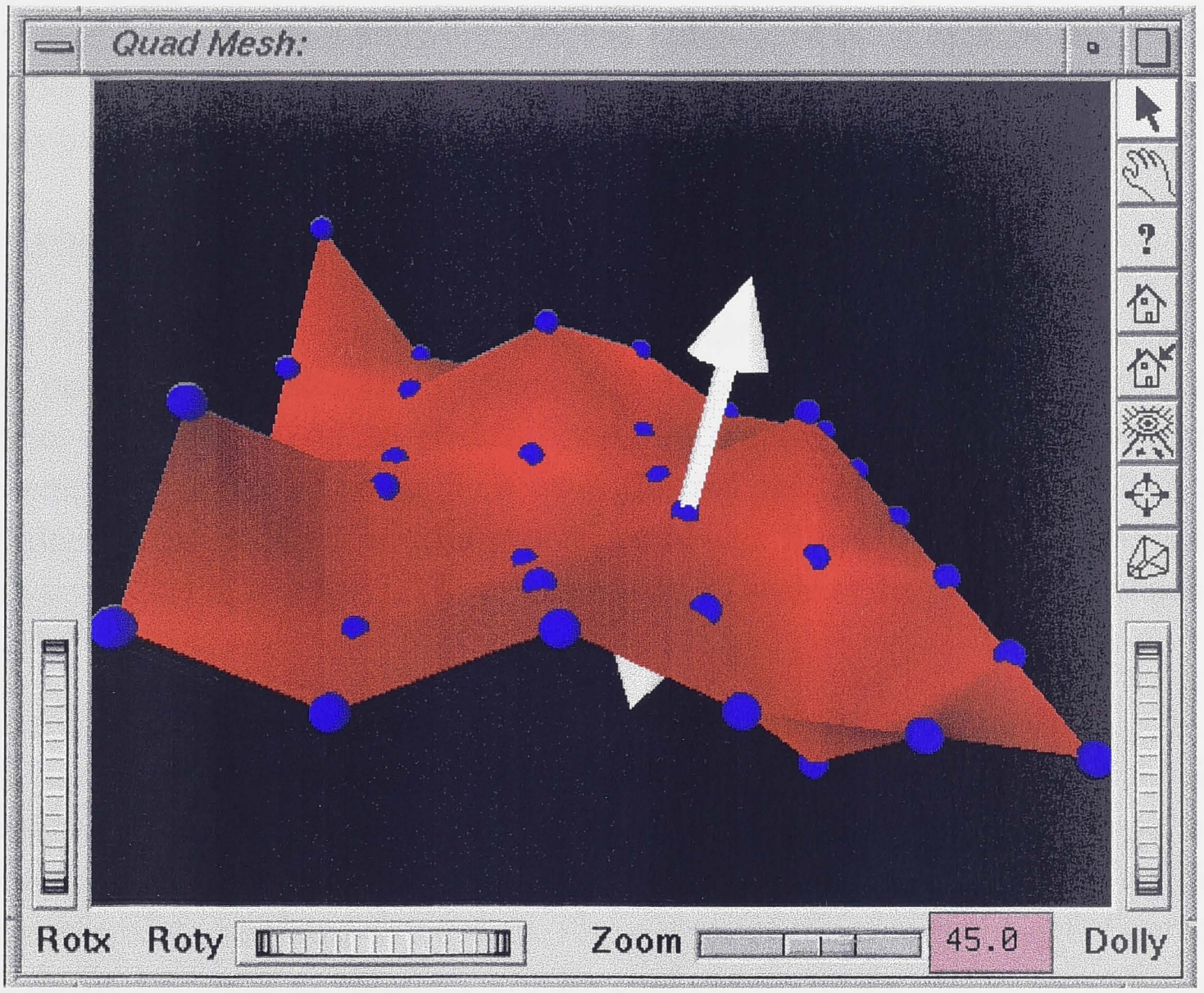


Figure 64 Prototype interface for controlling of two dimensional parameters using spline surfaces

Visualization

The filter visualization has to display enough information so that the user can evaluate the results of the filtering. This is especially important when the user has some control over the filtering parameters.

The parameter control maps form the foundation of visualization in the workbench. Parameter maps contain sufficient information about filter variation for the filter to perform filtering, and therefore must contain enough information for the user to understand it. The semantics of the values in the maps depend of the filter function, and therefore any interpretation of a map visualization much be tied to the specific problem. Since the parameter control maps are images, the full range of image visualization techniques can be applied. For example, intensity images, shading, surface perspective, false coloring, and so forth.

Viewing the filtered image can also provide significant information, especially if filtering can be performed in real-time with user interaction. In this case, user interaction can aid the user in quickly gaining an understanding of filter characteristics.

The range of visualization methods that can be applied is extensive, and depend greatly on the type of filter. This limits the support in this area to a few generic methods. The visualization module merely provides the interface to the parameter maps to allow applications to write their own visualizations, together with some straightforward intensity displays and false coloring; as will be shown later in some examples.

7.2.5 Filtering

The filtering module takes input from the control modules, either as a complete control map image, or element by element. It filters the source data according to the parameter control maps and any position maps. For a given application, a developer may use a template module or create their own. Custom written functions usually provide faster filtering for applications than general ones. The filtering module supports filters which vary in one of two ways. Those which vary by scale or size, and those which vary by shape.

Filters which vary by scale or size have a fixed kernel shape or function whose scale or size varies over the image. Filters whose kernel is warped or distorted by simple functions, such as those used in some geometrical transformations, are also considered in this category. Kernels whose size or scale varies are common in resampling filters, variable low, high, and band-pass filters. These types of filters have a single filter kernel which forms the basis of filtering. We provide two methods for implementing these filters. The first uses spatial lookup tables to store the filter function. The indices and entries of these tables are scaled to perform the variation [Feibush et al., 1980]. The second method stores a series of basis functions in a pyramid structure, and different levels and basis functions are summed to construct the filter kernel [Fournier and Fiume, 1988].

Filters which vary by shape or function are harder to deal with than simple scale or size changes. We only deal with those that can be simplified using one of the following methods.

- Filters that can be constructed from a combination of several basis functions. By combining the basis functions in different ways, different filter kernels can be constructed. In some cases they are simply variations of a basic filter kernel [Park and Schowengerdt, 1983], in other cases they can form directional filters of varying direction [Freeman and Adelson, 1991]. These types of filters form a large flexible set.
- Filters which take on only a small number of fixed forms. All forms can be stored and the appropriate kernel selected for filtering each location.
- Filters which use weighted combinations of adjacent samples, and the weights can be computed or selected from a table based on a small number of parameters.

7.3 Examples

To illustrate how the workbench can be used for different types of problems, we run through three examples; a data adaptive filter example, a user dependent example, and a geometric dependent example.

7.3.1 Median filter

The median filter is a common data adaptive filter used for removing impulse or spike noise from images. It replaces each pixel value by the median pixel value in a window of data surrounding it (see Figure 65). For many pixels, their value remains unchanged or is changed by very little, since on smooth slopes the original pixel value lies close to the median value. However, if the pixel is a spike or unusual impulse, it will be replaced by one of its surrounding values. We use the median filter as an example because it is a well known, relatively simple, data-dependent filter.

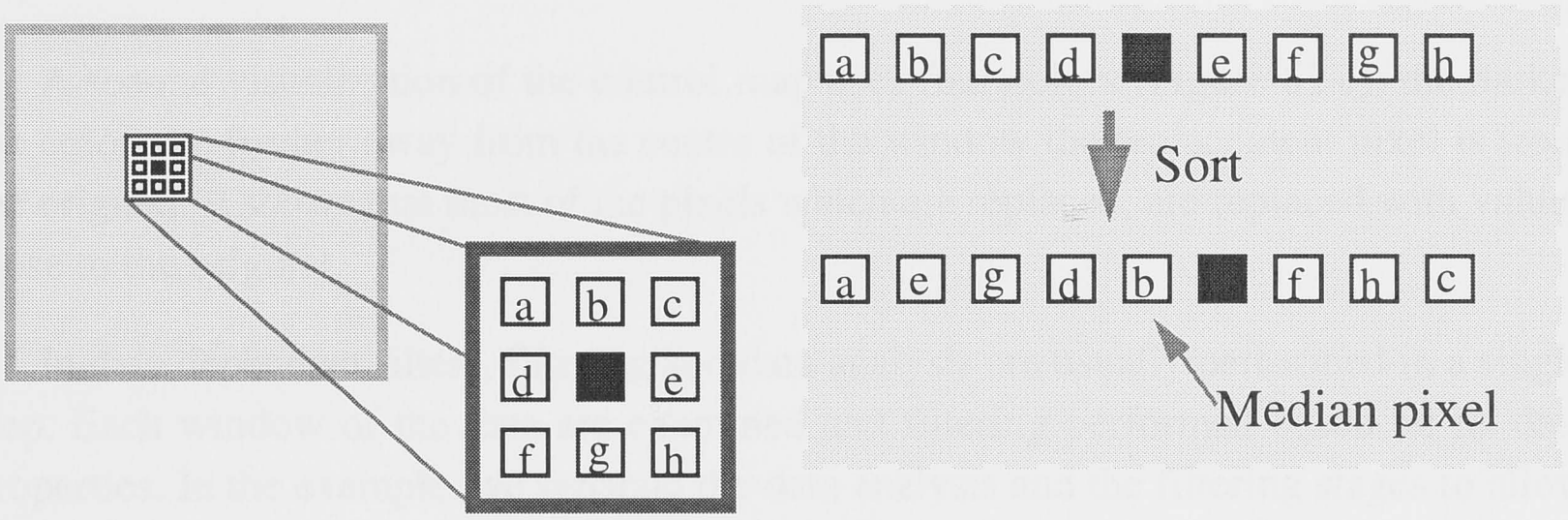


Figure 65 Median filter algorithm

To implement a median filter in the workbench, we need to form a filter function t_m, a control function d_0 which depends on the data, and determine the form of the parameter m_0. The choice of d_0 should allow the parameter m_0 to convey the most information about filter variation to enable it to be visualized. We chose a d_0 function which calculates the location of the median pixel value within the current window. The filter function t_m depends on this location parameter, and replaces the current pixel value with the pixel value of a nearby pixel based on that location parameter. We show this operation in Figure 66.

To demonstrate this form of the median filter, we de-noise an aerial photo using a 7x7 window median filter. The original photo is shown in Figure 67.a. The result of median filtering it is shown in Figure 67.d. To visualize the results, we first view the parameter control map in two colors (Figure 67.b); white for pixels with their original

value, black for pixels that have changed. The colors with respect to the filter window are shown in the top left corner. While only a slight improvement can be seen in the final image, the map shows that quite a few pixels had their values changed by the filter. This occurs noticeably at edges in the image where step changes in intensity are less distinct.

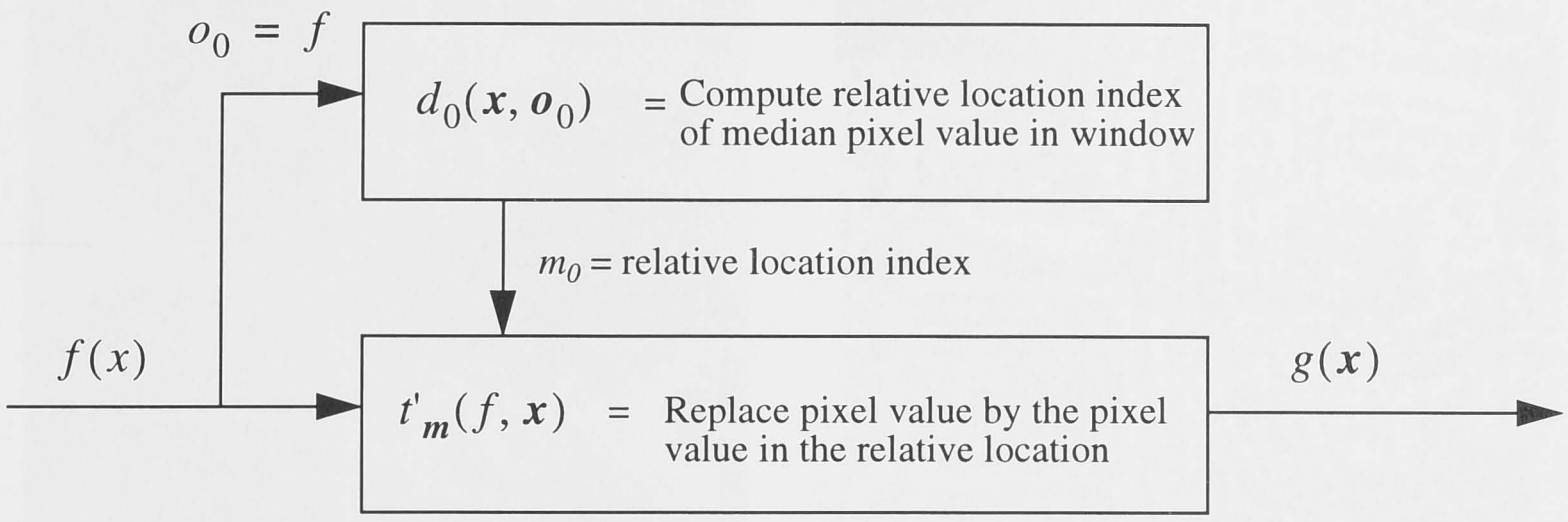


Figure 66 Median filter implementation in workbench

A second visualization of the control map uses four colors (Figure 67.c), the darker the color, the further away from the centre of the window the replacement pixel is from the original. It shows that most of the pixels which are replaced, are replaced with values close to the original.

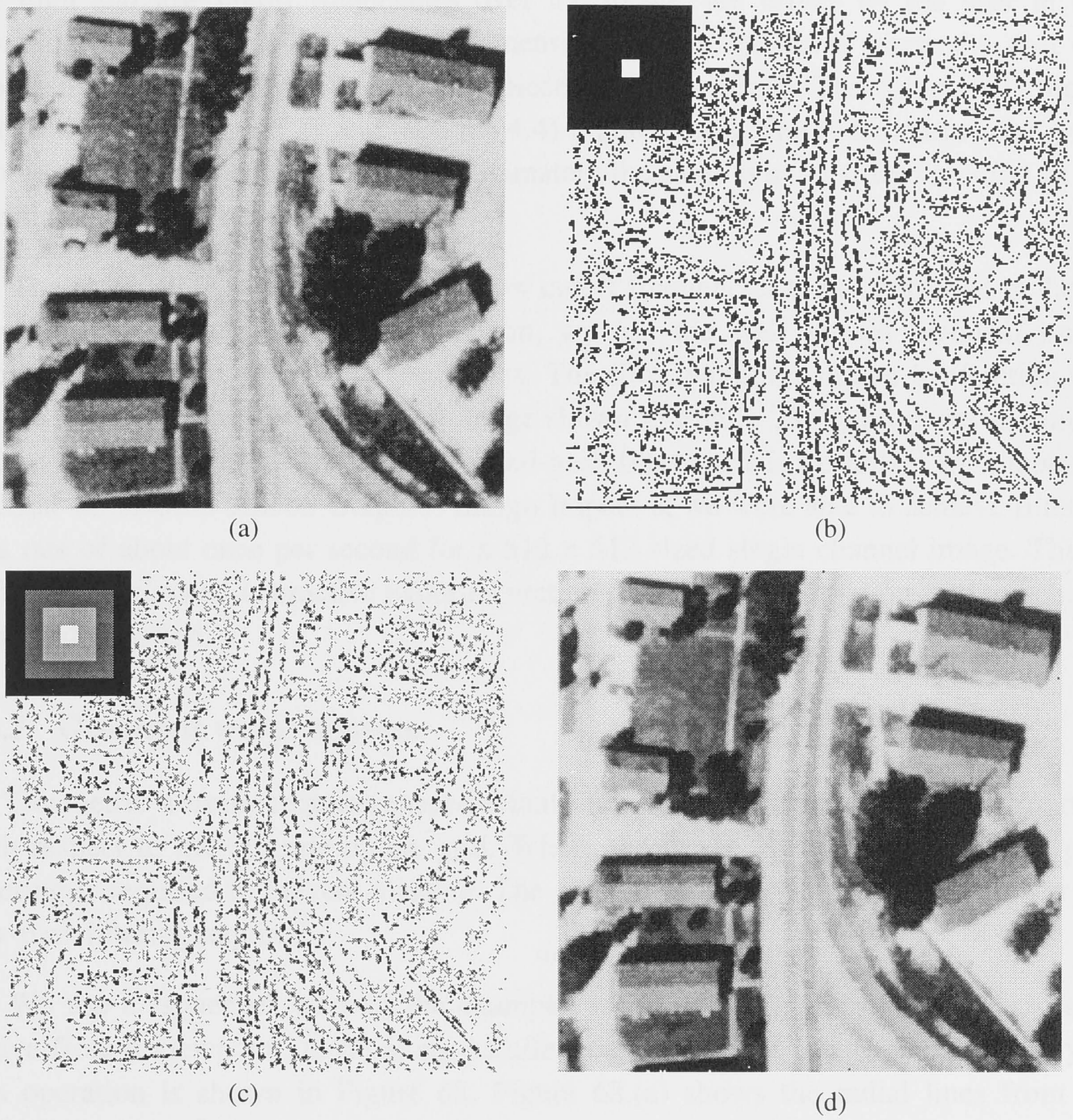
In data dependent filters, filtering and data analysis are usually performed in a single step. Each window of the data are examined and filtering performed based on its data properties. In the example, we separate the data analysis and the filtering stages to allow the filter variation to be examined. Obviously more memory is required to store the parameter maps, and the filtering is more costly. However, flexibility is gained because a large range of data dependent filters can be implemented merely by changing the data analysis phase – the filtering stage remains the same. In addition, it promotes better understanding of the filtering process through the display of variation information. In cases where the sensitivity or degree of adaptiveness of the filter can be controlled by the user, this understanding is essential for control.

7.3.2 Time-variant band-pass filter for seismic data processing

Interpretation of reflection seismic data and the process of removing noise from it is a highly subjective process. Geophysicists aim to achieve a good balance between noise removal and signal preservation to improve the interpretation of the data (Chapter 4).

While noise can be prevalent throughout the entire frequency spectrum, the reflected signal's spectrum is usually limited to a narrow band. To improve the signal-to-noise ratio of the signal, a band-pass filter is often used to remove noise outside the reflected signal's band (see Chapter 4). Because the energy absorbed by rock layers is frequency

dependent, higher frequencies are absorbed faster, and the frequency content of the signal changes with depth. Thus the band-pass filter used should vary with depth.



(a) (b) (c) (d)

Figure 67 Result of median filtering and visualization. (a) original image, (b) parameter control map with 2 colors, (c) parameter control map with 4 colors, (d) filtered image.

We implement the depth dependent band-pass filter in the workbench as a space-variant filter whose variation is only in one direction. $t_{\boldsymbol{m}}(f, \boldsymbol{x})$ is a band-pass filter which depends on three parameters m_0, m_1, and m_2. m_0 is the filter kernel function which is global over the image, and which is stored in an array. m_1 and m_2 are the low and high cutoff frequencies respectively of the filter over the image. They vary in only one direction, and so have a dimensional-dependence of one.

The $d_0(\boldsymbol{x}, \mathbf{o})$ function for the filter kernel can be designed so that the user can experiment with different basic filter types, to apply over the entire image. **o** parameters, for example, could be transition zone characteristics. $d_1(\boldsymbol{x}, \mathbf{o})$ and $d_2(\boldsymbol{x}, \mathbf{o})$, which control the high and low cutoff frequencies over the image, are driven by the user in one direction. We use an interactive two dimensional graph interface, where the user can move any number of control points on a piecewise linear graph. This interface is shown in (Section 4.3.3) Figure 27 and (Section 4.4) Figure 29.(a). The two dimensional graph should allow geophysicists to get approximately the same type of filter variation as the traditional method, but interactively.

The graph allows both the high and low cutoff frequencies to be controlled in the X-direction. To experiment with interaction, we use the well known mandrill image because of its good spread of frequencies. The filter variation shown in (Section 4.4) Figure 29.(a) applied to the mandrill image shown in Figure 29.(b) produces the image shown in Figure 29.(c). The variable band-pass filter used is described in Chapter 4, Section 4.3.1. On a Silicon Graphics Indigo Impact 2, we were able to achieve filtering at a rate of about once per second for a 512 x 512 sized single channel image. This is sufficient to allowed interactive experimentation with the filter variation, and to achieve a desired variation.

7.3.3 Geometric warp - squeeze

Geometric warping requires space-variant filtering, if the sample spacing is irregular with respect to the original image grid. Where the image is super-sampled, low-pass filtering is required to prevent aliasing. The greater the super-sampling, the greater the low-pass filtering required.

We use a "squeeze" warp as the example, which squeezes lines of sight across an image from a viewpoint, to lie along parallel scanlines of the image. The geometry of this operation is shown in Figure 68. Figure 68.(a) shows the radial lines from the viewpoint spreading out over the image. Figure 68.(b) shows where these lines end up on the final image; as parallel scanlines. Figure 68.(c) shows the original image grid and Figure 68.d the grid after the image has been warped.

The filter kernel function t_m for the resampling filter is a low-pass filter which depends a single parameter m_0, which is the bandwidth of the filter. We supply the geometry module with a function that computes the required bandwidth of the filter at each location, given the list of sampling coordinates. This function calculates the local super-sampling at each new coordinate, and determines the filter bandwidth to prevent aliasing at that position. We simplify the filtering by keeping the bandwidth of the filter the same the filter in both the X and Y directions.

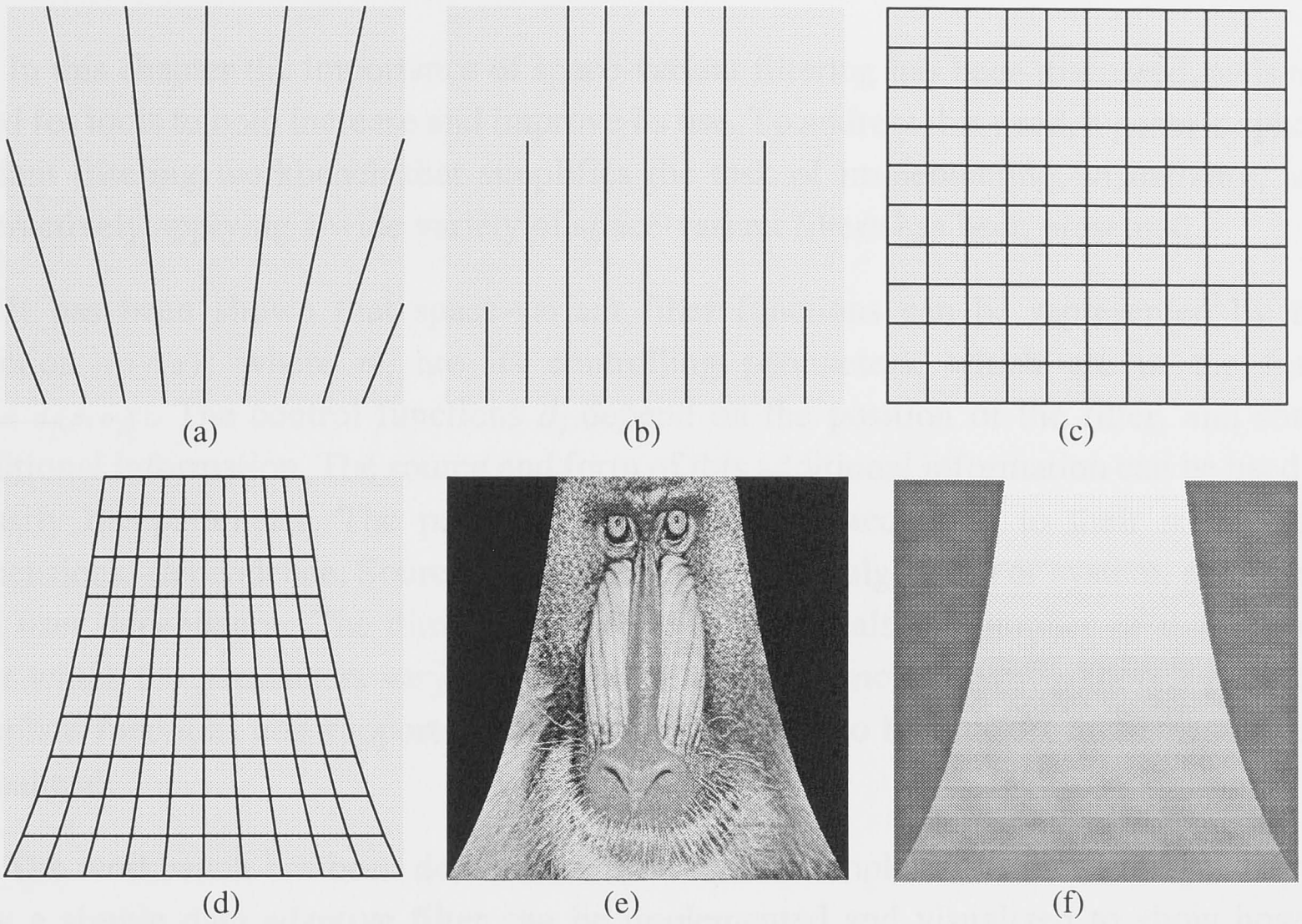


Figure 68 Turn fan lines (a) into parallel lines (b), the data plane (c), warped data plane (d) by the squeeze operation, (e) a squeezed image, (f) a bandwidth control image

To perform the warp, the squeeze algorithm computes the new sample positions of the image and passes them to the geometry module. The geometry module computes the filter bandwidth, then passes on the bandwidth parameter control map, together with the X and Y coordinate maps, to the resampling filter. The resampling filter then performs the filtering.

View of the bandwidth parameter map shows where in the final image information has been lost from the original. The lower the bandwidth in the map, the more information has been lost from the original image. This information can be important, especially after multi-stage operations, to understanding the fidelity of the data. An example is shown in Figure 68.(e) and (f). In the bandwidth control image, the lighter the color, the more low-pass filtering was performed on the image, and therefore the more bandwidth has been lost from the original image.

7.4 Summary

In this chapter the importance of space-variant filtering has been discussed, as is the need for tools to both increase and improve its use. To address this need, a generic space-variant filtering workbench that simplifies the task of implementing, visualizing, and interactively applying a wide variety of space-variant filters has been presented.

It has been shown that space-variant filter functions can be represented by the function $t_x(f, \boldsymbol{x})$, where m_i are its controlling parameters, which are in the form $m_i = d_i(\boldsymbol{x}, o_i)$. The control functions d_i depend on the position of the filter, and some additional information. The source and form of this additional information can be used to classify the parameter. The parameters are classified according to their source and dimensional-dependence. Source-dependencies are data, algorithm or system, geometry, and user dependence. The dimensional dependence equals the number of dimensions over which the parameters vary. Based on these dependencies, the workbench provides template functions and supporting tools to allow filters to be quickly implemented and visualized.

The workbench has been demonstrated on three examples. The first example shows how a simple data adaptive filter can be implemented and visualized to show how it changes the data. The second example shows how user interaction can be incorporated into the filtering process. The third example shows how a geometrically driven resampling filter can be applied, and how an idea of the fidelity of the original image could be gained through visualization of the bandwidth of the resampling filter.

8 CONCLUSIONS

8.1 Summary

This thesis has opened up the research area of interactive space-variant image filtering and sought to solve some of its fundamental problems. Initial work focuses on solving small well defined problems to develop and test ideas. These ideas were then generalized and applied to the design of a framework that supports a wider range of problems. The culmination of this work is an object-oriented software workbench for aiding future research and development into space-variant filtering applications; its design simplifies the task of developing interactive space-variant image filtering applications.

Since interactive space-variant filtering is a new research area, background work consisted of extracting relevant results from a broad range of research fields including interactive space-invariant filtering, subjective filter evaluation, human perception and visualization, filter evaluation and comparison, and space-variant filtering. This background work has been presented in Chapter 2.

In Chapter 3, an interactive space-variant smoothing filter has been developed for improving the subjective quality of shaded terrain images which suffer from quantization noise giving rise to terracing artifacts. The filter used is a convolutional low-pass filter whose kernel is stored in a lookup table. Variation of the low-pass filtering is achieved by scaling the indices of the lookup table during the discrete convolution.

User control over the smoothing is performed using three techniques. The first two techniques, global shifting and global scaling of smoothing, provide space-invariant control over image filtering. The third technique, based on interaction with a lookup table, allows control over the amount of smoothing applied to image regions with different characteristics or statistics (we used image gradient to differentiate regions). Initial variation of the smoothing filter is calculated as the inverse of the gradient value at each location, and is tuned through user interaction. The interactive control over image filtering provided by these techniques is sufficient to allow the user to achieve a better result then can be obtained by a standard uniform filtering techique.

Chapter 4 has examined the problem of controlling the band-pass filter used for improving the signal-to-noise ratio of reflection seismic data. Traditional methods provided only limited feedback and control over filtering. We describe a direct method for specifying and interacting with the one dimensional variation of the upper and lower cut-off frequencies of the band-pass filter over the two dimensional data. This method, which is based on direct manipulation of a 2D graph, supports the same types of variation as the traditional method. Experimentation with synthetic images verified that filter variation can be successfully controlled using this technique. This work

demonstrates the feasibility of direct interactive control and specification of the variation of space-variant filtering.

Chapter 5 has addressed the more general problem of visualizing digital raster maps (DRM) using geometric scaling, focusing on the selection of an interpolation filter which improves the subjective quality of the scaled image, and parallel algorithms for fast scaling. A perceptual experiment that was performed used a novel method for the subjective selection of a filter based on a powerful image comparison technique. The image comparison technique allows small differences between images to be seen as motion when flipping between two spatially aligned images.

To address the real time speed requirements of scaling large DRM, a software clustering scheme for speeding up algorithms on massively parallel machines is described. This method, which is limited in the types of algorithms it supports, uses several real processors to perform the work of one virtual processor, and so works in reverse of traditional virtualization schemes.

Chapter 6 showed how interactive visualization can support a framework for space-variant image filtering. The framework exploits image masks to carry reference information, such as the spatial frequency content or discontinuity maps, from which required filter kernel characteristics can be determined. Reloadable filter kernels are implemented within a parallel tool-kit, where parallelism up to the number of image scanlines is supported. The framework is general for multi-dimensional images for linear separable filters. Filter kernel images, carrying current or historical information about the space-variant filter applied to the data, are visualized to help enable the expertise of specialists or interpreters by providing direct feedback.

Chapter 7 derives a more general form of the reference mask called a parameter control map, which is not restricted to two dimensions and has wider application. A classification scheme is defined for these maps based on filter dependence information. Maps are classified as having one or more dependencies: data, system, geometric, or user dependence. Using the parameter control maps and their dependencies as a base, a general workbench for implementing and controlling space-variant filters is developed. The use of this workbench is successfully demonstrated by three examples with different dependency types.

8.2 Achievements and limitations

8.2.1 Subjective filter evaluation

The proposed subjective filter evaluation method based on image flipping has proved to be a powerful tool for comparing and evaluating the results of different filters. With it, it was possible to accurately determine where and how the results of filtering differ, especially when those differences are small. While it is capable of being applied to many filtering problems, its benefits will be limited for some.

8.2.2 Interactive filter control

The interactive filter steering interfaces developed for controlling the smoothing of shaded images (Chapter 3), and the band-pass filtering over seismic data (Chapter 4), has showed that interactive control of space-variant image filtering is possible. Further refinement will improve their performance as these interfaces can be developed still further.

Control of filter variation using the reference masks of Chapter 6, or the more general parameter control maps of Chapter 7, simplifies the process of filter control greatly, providing a uniform interface for filtering of images and facilitating the construction of general tools. The exact form of these control images depends on the filter parameter they are representing and the application of the filter; the form of the control image is characterized by its datatype and the semantic meaning of its values. Because the form is application dependent, tools for manipulating the images also have some application dependence.

8.2.3 Generic workbench

The proposed generic space-variant filtering workbench simplifies the task of implementing and applying space-variant image filters. It does this by supporting features common to all space-variant filters based on the generic derived form. The variety of different filters is large, and because they have many different features, it is impractical to provide strong support for all. Common properties between filters, such as found in the control parameters, has allowed us to make the framework more general.

The workbench's classification system for filters, based on parameter dependencies, is one of its important contributions. It defines four major classes for filters based on the types of sources of filter parameter values. Mixed classes result when a filter has multiple parameters with different dependencies.

The definition of the filter classes characterizes the shared behaviour between filters in that class. This shared behaviour is exploited for the development of tools that can be applied to filters in that class.

The object-oriented design of the workbench which is described in Chapter 7, and whose class diagrams are included in Appendix A, has been prototyped on a Silicon Graphics Indigo Impact workstation, and ported to a DEC Alpha workstation. Its implementation in C++ makes it easily portable amongst a range of different architectures.

8.2.4 Future research

In this thesis we have explored as virgin territory the research area of interactive space-variant image filtering, and have sought to map the area and make it accessible to the user. The software workbench for space-variant filtering has been developed not only for the specific studies in the thesis, but to support future research in interactive space-variant image filtering; more specifically into the techniques for controlling two dimensional and higher variation, and in the development of space-variant filtering algorithms which are flexible, controllable, and understandable.

Future work should focus on integrating these tools more fully with the real application problems, to test the workbench and conclusions developed in this thesis over an extended domain of problems and tasks.

9 REFERENCES

Abramson, S. B. and Schowengerdt, R. A. (1993). Evaluation of edge-preserving smoothing filters for digital image mapping. *ISPRS Journal of Photogrammetry and Remote Sensing*, 48(2):2–17.

Anderson, G. L. and Netravali, A. N. (1976). Image restoration based on a subjective criterion. *IEEE Transactions on Systems, Man, and Cybernetics*, smc-6(12):845–853.

Boult, T. E. and Wolberg, G. (1993). Local image reconstruction and subpixel restoration algorithms. *CVGIP: Graphical models and image processing*, 55(1):63–77.

Bracewell, R. N. (1995). Private correspondance.

Brailean, J. C., Sullivan, B. J., Chen, C.-T., and Giger, M. L. (1991). Evaluating the EM algorithm for image processing using a human visual fidelity criterion. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 2957 – 2960.

Breene, L. and Bryant, J. (1993). Image warping by scanline operations. *Computers and Graphics*, 17(2):127–130. Parallel Algorithms and Architectures; Neural Networks.

Brown, E. F. (1969). Television: The subjective effects of filter ringing transients. *Journal of the SMPTE*, 78(4):249–255.

Carlbom, I. (1993). Optimal filter design for volume reconstruction and visualization. pages 54–61.

Catmull, E. and Smith, A. R. (1980). 3D transformations of images in scanline order. *Computer Graphics*, pages 279 – 285.

Crow, F. C. (1984). Summed-area tables for texture mapping. *Computer Graphics*, 18(3):207–212.

Fant, K. M. (1986). A nonaliasing, real-time spatial transform technique. *IEEE Computer Graphics and Applications*, pages 71–80.

Feibush, E. A., Levoy, M., and Cook, R. L. (1980). Synthetic texturing using digital filters. *Computer Graphics*, 14(3):294–301.

Foley, J. D., van Dam, A., Feiner, S. K., and Hughes, J. F. (1991). *Computer Graphics: Principles and Practise*, volume 2 of *The Systems Programming Series*. Addison-Wesley Publishing Company.

Fonseca, L. M. G., Prasad, G. S. S. D., and Mascarenhas, N. D. A. (1993). Combined interpolation-restoration of landsat images through FIR filter-design techniques. *International Journal of Remote Sensing*, 14(13):2547–2561.

Fournier, A. and Fiume, E. (1988). Constant-time filtering with space-variant kernels. *Computer Graphics*, 22(4):11.

Fraser, D. (1987). A conceptual image intensity surface and the sampling theorem. *The Australian Computer Journal*, 19:119–125.

Fraser, D. (1989a). Comparison at high spatial frequencies of two-pass and one-pass image geometric transformation algorithms. *Computer Vision, Graphics, and Image Processing*, 46:267–283.

Fraser, D. (1989b). Interpolation by the FFT revisited - an experimental investigation. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-37:665–675.

Fraser, D., He, H., and Schowengerdt, R. A. (1996). High-fidelity image warping for serial and parallel processing. In *Proc. ICIP'96, IEEE International conference on Image processing*, pages 719–722.

Fraser, D. and Schowengerdt, R. (1994). Avoidance of additional aliasing in multipass image rotations. *IEEE Trans. on Image Processing*, 3:721–735.

Fraser, D., Schowengerdt, R. A., and Briggs, I. (1985). Rectification of multichannel images in mass storage using image transposition. *Computer Vision, Graphics, and Image Processing*, 29:23 – 26.

Freeman, W. T. and Adelson, E. H. (1991). The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Inteliigence*, 13(9):891 – 906.

Gershon, N. (1994). *Scientific Visualization*, chapter 8. From perception to visualization, pages 129 –138. Academic Press.

Gershon, N. D. (1992). Visualization of fuzzy data using generalized animation. In *Proceedings of IEEE Visualization 92*, pages 268–273.

Greene, N. and Heckbert, P. S. (1986). Creating raster omnimax images from multiple perspective views using the elliptal weighted average filter. *IEEE Computer Graphics and Applications*.

Hall, E. L. (1979). *Computer Image Processing and Recognition*. Academic Press.

Haralick, R. M. (1984). Digital step edges from zero crossings of second directional derivatives. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-6(1):58–68.

Hatton, L., Worthington, M. H., and Makin, J. (1986). *Seismic Data Processing: Theory and Practice*. Blackwell Scientific Publications.

Heckbert, P. S. (1986a). Filtering by repeated integration. *Computer Graphics*, 20(4):315 – 321.

Heckbert, P. S. (1986b). Survey of texture mapping. *IEEE Computer Graphics and Applications*, pages 56–67.

Hillis, D. W. (1985). *The connection machine*. The MIT Press.

Hou, H. S. and Andrews, H. C. (1978). Cubic splines for image interpolation and digital filtering. *IEEE Trans. Acoustics, Speech, and Signal Processing*, ASSP-26(6):508–517.

Huck, F. O., Alter-Gartenberg, R., and Rahman, Z. (1991). Image gathering and digital restoration for fidelity and visual quality. *CVIP: Graphical Models and Image Processing*, 53(1):71–84.

Jain, A. K. (1989). *Fundamentals of Digital Image Processing*. Prentice-Hall.

John T. Hooks, J., Marinsen, G. J., and Devarajan, V. (1993). On 3-D real-time perspective generation from a multiresolution photo-mosaic data base. *CVGIP: Graphical Models and Image Processing*, 55(5):333 – 345.

Kaba, J. and Peters, J. (1993). A pyramid-based approach to interactive terrain visualization. In *Visualization 93*, page 67.

Keightley, D., Tsui, K., Lilleyman, J., and Moore, K. (1993). A software framework for an application-driven parallel image processing and display system (pipads). In *SPIE Conference on Recent advances in Sensors, Radiometric Calibration, and Processing of Remotely Sensed Data*, pages 368–379.

Kelly, P. and Keller, M. (1992). *Visual Cues*. IEEE Computer Society Press.

Keys, Robert, G. (1981). Cubic convolution interpolation for digital image processing. *IEEE Trans. Acoustics, Speech, and Signal Processing*, ASSP-29(6):1153–1160.

King, M. A., Glick, S. J., Penney, B. C., Schwinger, R. B., and Doherty, P. W. (1987). Interactive visual optimization of spect prereconstruction filtering. *The Journal of Nuclear Medicine*, 28(7):1192 – 1198.

Kishimoto, K., Onaga, K., and Nakamae, E. (1987). Theoretical assessments of mean square errors of antialiasing filters. *Computer Vision, Graphics, and Image Processing*, 37:428–437.

Lee, J.-S. (1980). Digital image enhancement and noise filtering by use of local statistics. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, (2):165–168.

Livingstone, M. S. (1988). Art, illusion and the visual system. *Scientific American*, pages 68 – 75.

Lorenz, H., Richter, G. M., Capaccioli, M., and Longo, G. (1993). Adaptive filtering in astronomical image processing. *Astronomy and Astrophysics*, 277(1):321–330.

Lui, A. K. and Bone, D. (1995). Novel approach to interactive editing of raster based data. In *Proceedings of SPIE Conference: Visual Data Exploration and Analysis II*.

Lui, A. K. and Bone, D. (1996). Graphical raster based data editors. In *Proceedings of the 19th Australasian Computer Science Conference*.

Machiraju, R. and Yagel, R. (1996). Reconstruction error characterization and control: A sampling theory approach. *IEEE Transactions on visualization and computer graphics*, 2(3):364–378.

Maeland, E. (1988). On the comparison of interpolation methods. *IEEE Trans. Medical Imaging*, 7(3):213–217.

Mallat, S. G. (1989). A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(7):674–693.

Marr, D. (1982). *Vision*. Freeman.

Marschner, S. R. and Lobb, R. J. (1994). An evaluation of reconstruction filters for volume rendering. In *Visualation 94*, pages 100–107.

Mastin, G. A. (1985). Adaptive filters for digital image noise smoothing: An evaluation. *Computer Vision, Graphics, and Image Processing*, 31:103–121.

McCormick, B. H., Defanti, T. A., and Brown, M. D. (1987). Visualization in scientific computing. *Computer Graphics*, 21(6).

Miller, T. R., Hagge, R. J., Wallis, J. W., and Sampthkumaran, K. S. (1988). Interactive digital filtering of gated cardiac studies during cine display. *IEEE Transactions on Medical Imaging*, 7(3):188–192.

Miller, T. R. and Rollins, E. S. (1985). A practical method of image enhancement by interactive digital filtering. *The Journal of Nuclear Medicine*, 26(9):1075–1079.

Miller, T. R., Sampathkumaran, K. S., and King, M. A. (1983). Rapid digital filtering. *The Journal of Nuclear Medicine*, 24(7):625–627.

Mitchell, D. P. (1987). Generating antialiased images at low sampling densities. *Computer Graphics*, 21(4):65 –69.

Mitchell, D. P. and Netravali, A. N. (1988). Reconstruction filters in computer graphics. *Computer Graphics*, 22(8):221 – 228.

Moore, K. W. and Robertson, P. K. (1995). Interactive steering of image filtering using visualization of space-variant filter kernels. In *SPIE Conference on Visual Data Exploration and Analysis 2*.

Moore, K. W. and Tsui, K. (1997). A software framework for visualizing space-variant image filters. In *SPIE Conference on Visual Data Exploration and Analysis 4*.

Moore, K. W. and Vézina, G. (1995). Enhanced geometric scaling of digital raster maps. In *Digital Image Computing:. Techniques and Applications*. The Australian Pattern Recognition Society.

Nickerson, K. and Haykin, S. (1989). Scan conversion of radar images. *IEEE Transactions on Aerospace and Electronic Systems*, 25(2):166 – 175.

Ozkan, M. K., Tekalp, A. M., and Sezan, M. I. (1994). POCS-based restoration of space-varying blurred images. *IEEE Transactions on Image Processing*, 3(4):450 –454.

Panda, D. P. (1978). Nonlinear smoothing of pictures. *Computer Graphics and Image Processing*, 8:250 – 270.

Pann, K. and Shin, Y. (1976). A class of convolutional time-varying filters. *Geophysics*, 41(1):28 – 43.

Park, C. and Black, R. A. (1995). Simple time-variant, band-pass filtering by operator scaling. *Geophysics*, 60(5):1527 – 1535.

Park, S. K. and Schowengerdt, R. A. (1982). Image sampling, reconstruction, and the effect of sample-scene phasing. *Appl. Opt*, 21:3142–3151.

Park, S. K. and Schowengerdt, R. A. (1983). Image reconstruction by parametric cubic convolution. *Computer Vision Graphics Image Process*, 23:258–272.

Parker, J. A., Kenyon, R. V., and Troxel, D. E. (1983). Comparison of interpolating methods for image resampling. *IEEE Trans. Medical Imaging*, 2(1):31–39.

Pham, B. (1989). Quadratic b-splines for automatic curve and surface fitting. *Computers & Graphics*, 13(4):471–475.

Potmesil, M. and Chakravarty, I. (1983). Modeling motion blur in computer-generated images. *Computer Graphics*, 17(3):389–399.

Pratt, W. K. (1991). *Digital image processing*. John Wiley & Sons, 2nd edition.

Ran, X. and Farvardin, N. (1995). A perceptually motivated three-component image model - part i: Description of the model. *IEEE Transactions on Image Processing*, 4(4):401 – 415.

Reeves, S. J. (1994). Optimal space-varying regularization in iterative image restoration. *IEEE Transactions on Image Processing*, 3(3).

Reichenbach, S. E. and Park, S. K. (1991). Small convolution kernels for high-fidelity image restoration. *IEEE Trans. Signal Processing*, 39(10):2263–2274.

Reichenback, S. E. and Park, S. K. (1989). Two-parameter cubic convolution for image reconstruction. *Proc. SPIE Visual Communications and Image Processing*, 1199:833–840.

Rhea, W. J. (1991). A new technique for enlargement and reconstruction of digital sensor imagery. *Internation Journal of Remote Sensing*, 12(3):627–634.

Rifman, S. S. and McKinnon, D. M. (1974). Evalution of digital correction techniques for erts images–final report. *Report 20634600-TU-00, TRW Systems*.

Robertson, P. K (1989). Spatial transformations for rapid scan-line surface shadowing. *IEEE Computer Graphics and Applications*, 9(3):30–38.

Robertson, P., Keightley, D., Kravis, S., Lilleyman, J., Tsui, K., Moore, K., Fulton, N., Hobbs, T., Boesen, B., Kim, I., Vance, C., Schroder, H., Spray, A., Lie, K., Bray, P., Moore, M., and Leu, G. (1992). PIPADS: a vertically integrated parallel image processing and display system. In *Fifth Australian Supercomputing Conference*.

Robertson, P. K. (1987). Fast perspective views of images using one-dimensional operations. *IEEE Computer Graphics and Applications*, 7(2):47–56.

Robertson, P. K. (1991). A methodology for choosing data representations. *IEEE Computer Graphics and Applications*, pages 56–67.

Robertson, P. K. (1993). Interactive visualization. In *DICTA-93*, pages 27–34.

Robertson, P. K., Fletcher, P. A., and Gunn, C. (1993). Interactive relief shading on graphics workstations. In Tang, Z., editor, *Proceedings of the Third International Conference on CAD and CG*, volume 1, pages 75–79. Int. Acad. Publishers.

Sawchuk, A. A. (1974). Space-variant image restoration by coordinate transformations. *Journal of the Optical Society of America*, 64(2):138 – 144.

Scher, A., Velasco, F. R. D., and Rosenfeld, A. (1980). Some new image smoothing techniques. *IEEE Transactions on Systems, Man, and Cybernetics*, smc-10(3):153–158.

Scheuer, T. and Oldenburg, D. W. (1988). Aspects of time-variant filtering. *Geophysics*, 53(11):1399 – 1409.

Schowengerdt, R., Park, S. K., and Gray, R. (1984). Topics in the two-dimensional sampling and reconstruction of images. *Int. Journal of Remote Sensing*, 5(2):333–347.

Schreiber, W. F. and Troxel, D. E. (1985). Transformation between continuous and discrete representations of images: A perceptual approach. *IEEE Trans. Pattern Analysis and Machine Intelligence*, PAMI-7(2):178–186.

Shannon, C. E. (1949). Communications in the presence of noise. *Proc. IRE*, 17(3):1–11.

Shon, H. and Yamamoto, T. (1992). Simple data processing procedures for seismic section noise reduction. *Geophysics*, 57(8):1064 – 1067.

Stein, R. A. and Bartley, N. R. (1983). Continuously time-variable recursive digital band-pass filters for seismic signal processing. *Geophysics*, 48(6):702 – 712.

Thorpe, G. and Fraser, D. (1996). Wide area imaging through the atmosphere. In *Conference on Optics in Atmospheric Propagation, Adaptive systems, and lidar techniques for remote sensing*, pages 188–197.

Thorpe, G., Fraser, D., Elphick, R., and Lambert, A. (1995a). Automatic image-sequence registration. In *Proc IVCNZ'95*, pages 107–112.

Thorpe, G., Fraser, D., and Lambert, A. (1995b). Modified hierachical adaptive pixel matching algorithm. In Lovell, B., editor, *DICTA95*.

Trussell, H. J. and Fogel, S. (1992). Identification and restoration of spatially variant motion blurs in sequential images. *IEEE Transactions on Image Processing*, 1(1):123 – 126.

Trussell, H. J. and Hunt, B. R. (1978a). Image restoration of space-variant blurs by sectioned methods. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-26(6):608 – 609.

Trussell, H. J. and Hunt, B. R. (1978b). Sectioned methods for image restoration. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-26(2):157 – 164.

Tsui, K. and Fletcher, P. (1995). Fast surface perspective algorithms. Technical Report TR-HJ-95-01, CSIRO Divison of Information Technology, PO Box 664, Canberra, ACT 2601, Australia.

Vézina, G. and Robertson, P. K. (1991). Terrain perspectives on massively parallel simd computers. In Patrikalakis, N. M., editor, *Scientific Visualization of Physical Phenomena*, pages 163–188. Springer-Verlag.

Vézina, G. and Robertson, P. K. (1992a). Data-parallel visualisation using multi-dimensional transformations. In *Frontiers of Massively Parallel Computation*, pages 230–236.

Vézina, G. and Robertson, P. K. (1992b). Resampling techniques for image processing and visualisation on massively parallel architectures. Technical Report TR-HJ-92-05, CSIRO Divison of Information Technology, PO Box 664, Canberra, ACT 2601, Australia.

Wahl, F. M. (1987). *Digital Image Signal Processing*. Artech House Inc.

Wang, D. C. C. and Vagnucci, A. H. (1981). Gradient inverse weighted smoothing scheme and the evaluation of its performance. *Computer Graphics and Image Processing*, 15:167–181.

Wanger, L. R., Ferwerda, J. A., and Gennberg, D. P. (1992). Perceiving spatial relationships in computer-generated images. *IEEE Computer Graphics and Applications*, pages 44–58.

Ward, J. and Cok, D. R. (1989). Resampling algorithms for image resizing and rotation. *Proc. SPIE Digital Image Processing Applications*, 1075:260–269.

Weiman, C. F. R. (1980). Continuous anti-aliased rotation and zoom of raster images. *Computer Graphics*, 14(3):286–293.

Williams, L. (1983). Pyramidal parametrics. *Computer Graphics*, 17(3):1–11.

Wolberg, G. (1990). *Digital Image Warping*. IEEE Computer Society Press, 1st edition.

Wolberg, G. and Boult, T. E. (1989). Separable image warping with spatial lookup tables. *Computer Graphics*, 23(3):369–378.

Yilmaz, O. (1987). *Seismic data processing*. Society of Exploration Geophysicists.

Zhou, Q. (1992). Relief shading using digital elevation models. *Computers and Geosciences*, 18(8):1035–1045.

10 APPENDIX A - Overview of the workbench's toplevel object-oriented software design

The following sections provide an overview of the object-oriented software design of the filtering workbench. The software described has been implemented and is available from the author.

10.1 Model overview

The main components of the workbench's software are the parameter controllers, parameter control maps (PCMs or simply parameters), filter module, source data, and destination data. The interaction of these components in the system is shown in Figure 69. The filter module can take any number of parameters depending on the type of filter, with each parameter controlled by a controller. The filter module in the diagram takes two parameters maps.

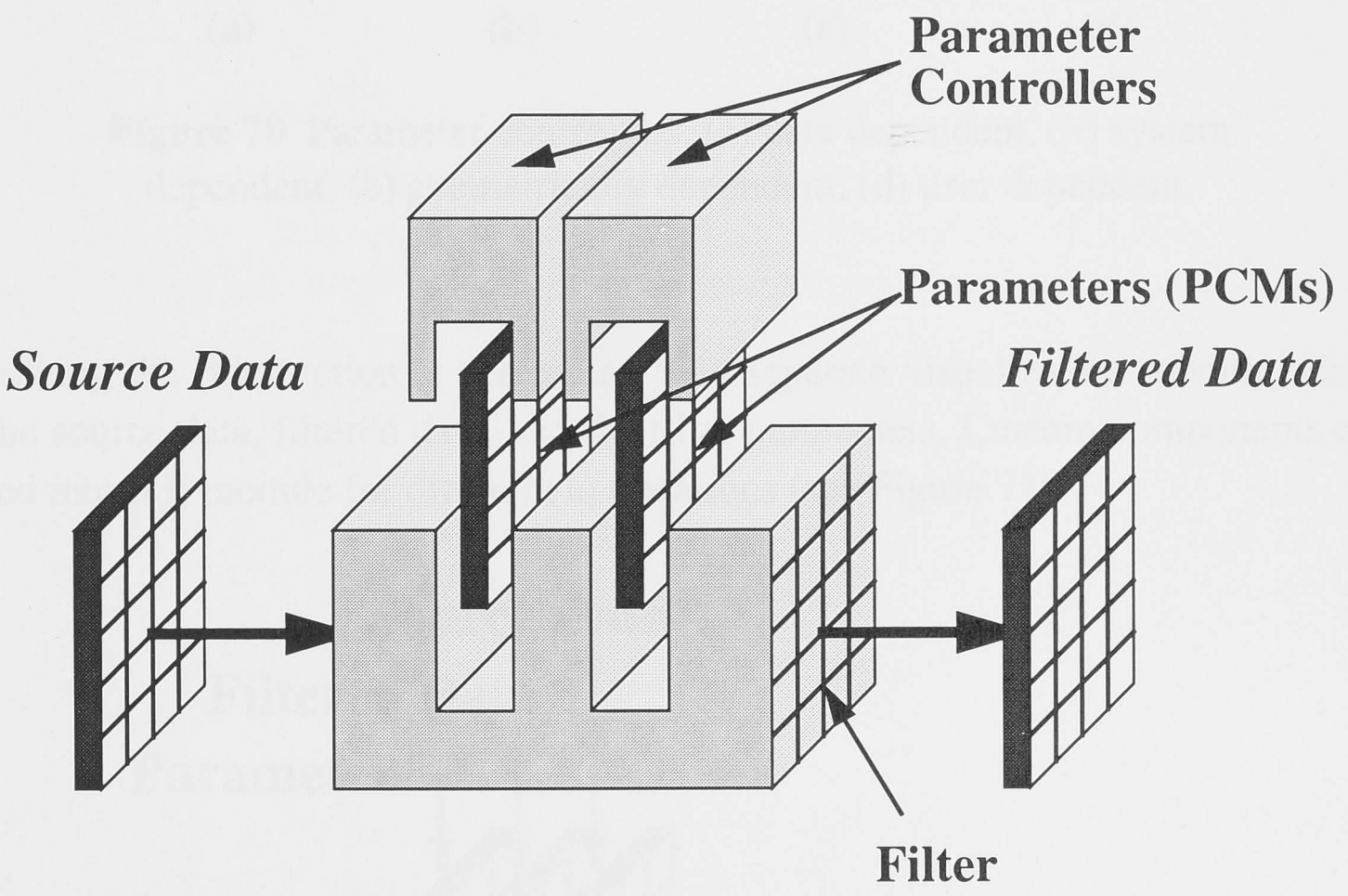


Figure 69 Workbench components

The system provides four different types of parameter controllers: data dependent, system dependent, geometrically dependent, and user dependent. Each can be customized for a particular application by plugging in different modules. The types of parameter controllers is shown in Figure 70.

The data dependent parameter controller takes a data relation and the source data to generate parameter values. The system dependent parameter controller provides

connectors to connect directly into the host system; the host system is responsible for generating the parameter values. The geometrically dependent controller provides an interface to allow the parameter values to be specified via geometric information. The user dependent controller provides a graphical user interface (GUI) to allow the user to control parameter variation.

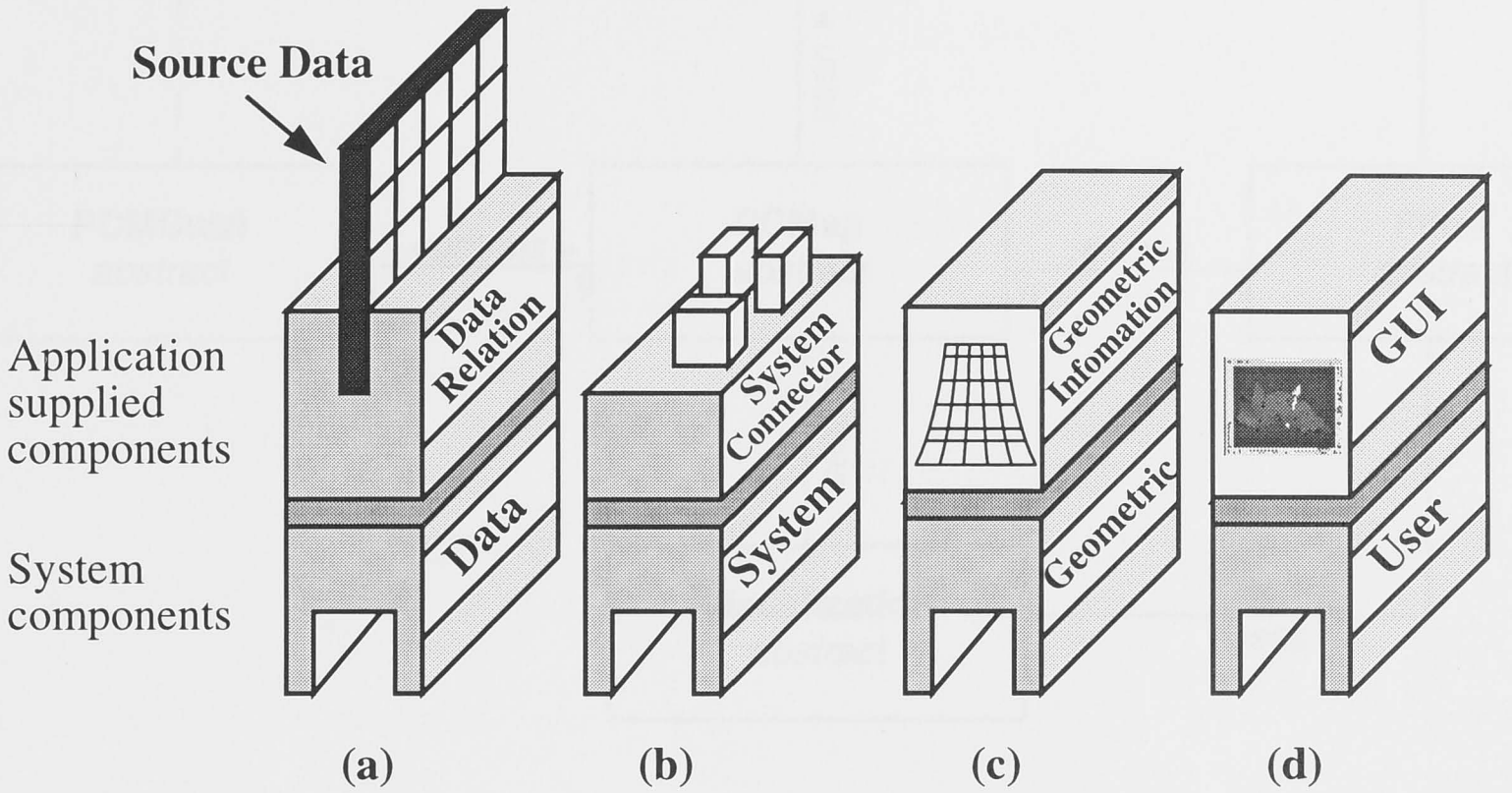


Figure 70 Parameter controllers. (a) data dependent, (b) system dependent, (c) geometrically dependent, (d) user dependent.

Visualizing the filter action is performed by a separate visualization module which accepts the source data, filtered data, and the filter parameters. Custom components can be plugged into this module for different applications (see Figure 71).

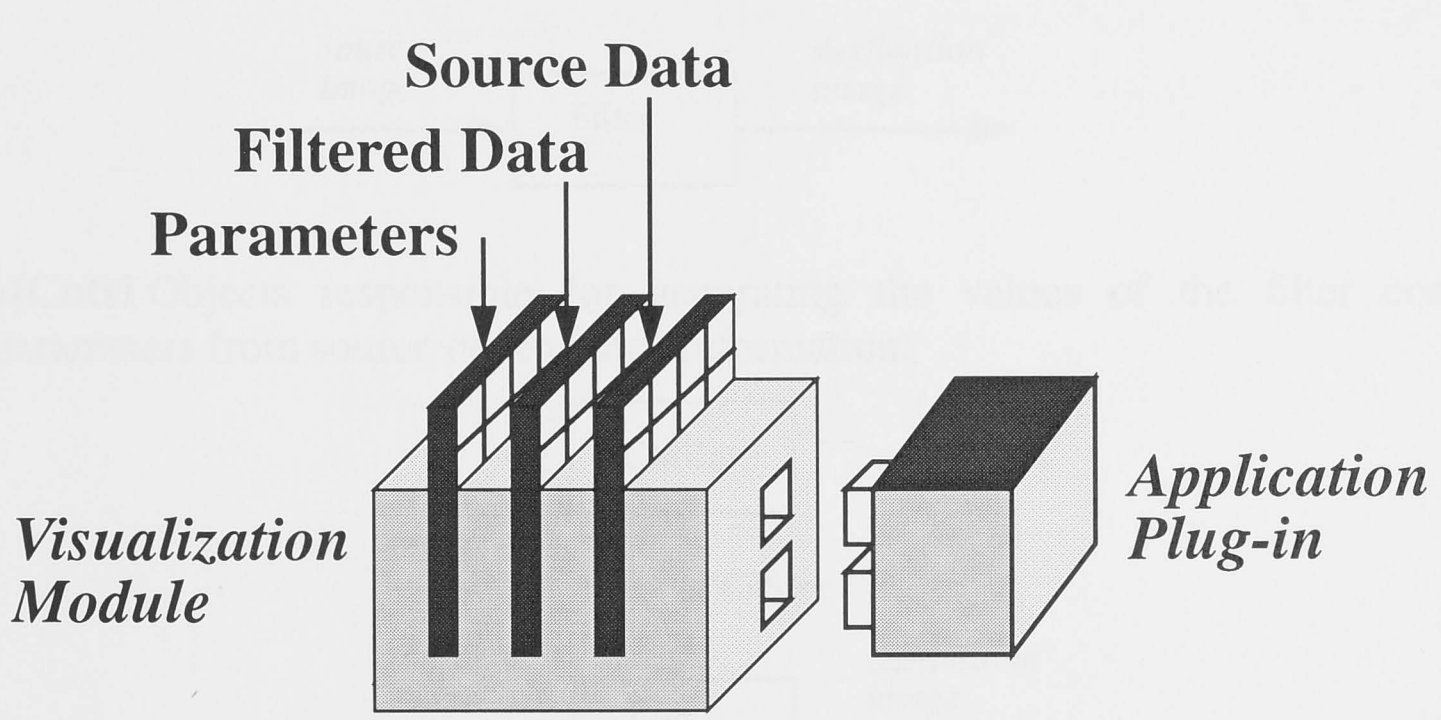
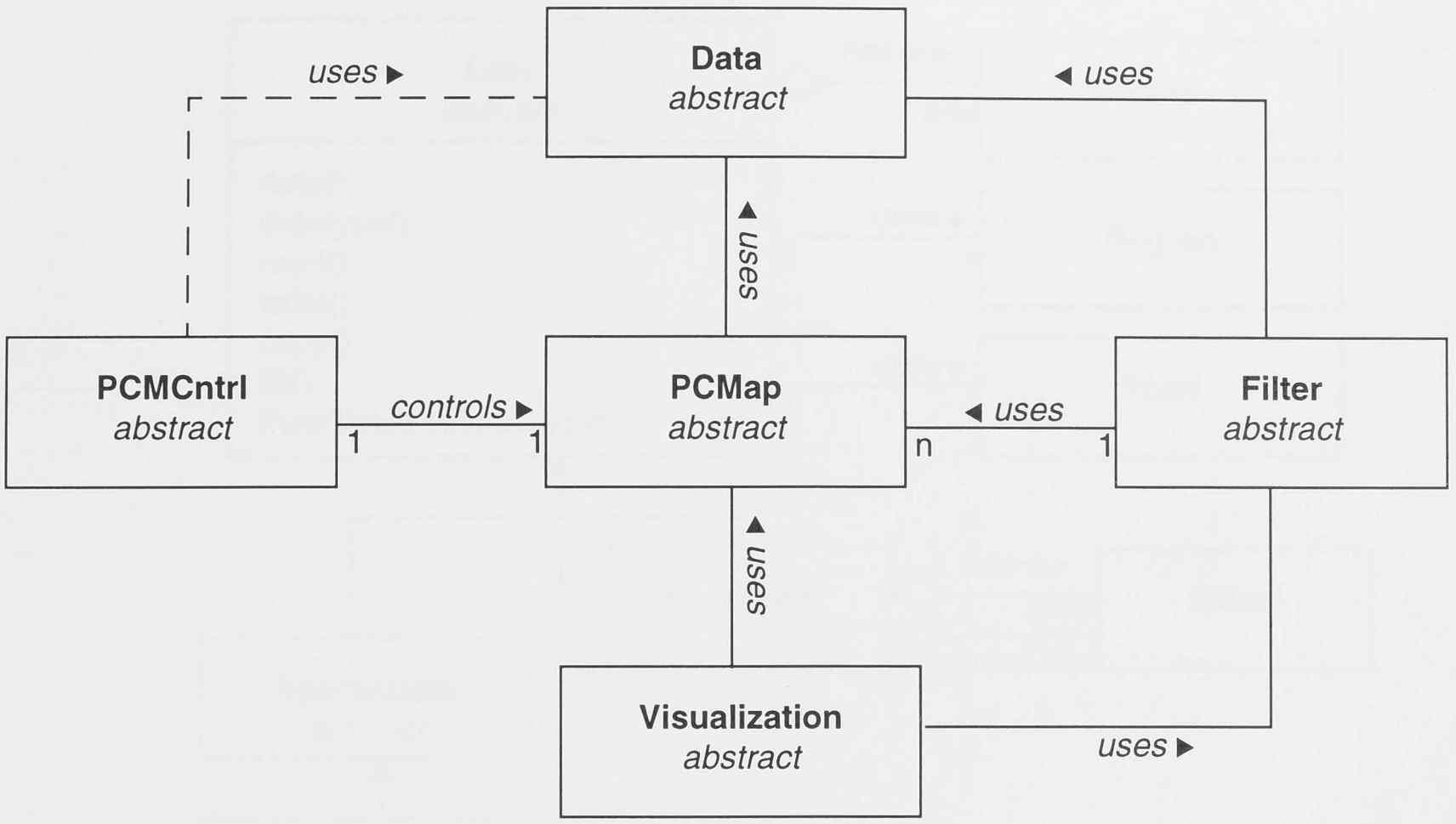


Figure 71 Visualization module

10.2 Class overview

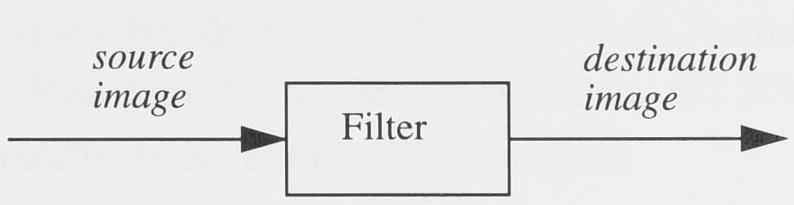


There are five major classes which define the workbench's operation:

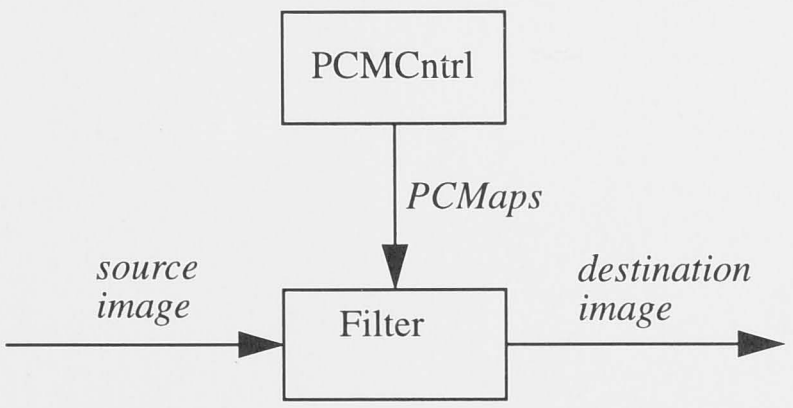
Data Describes the form and structure of the input data. Maintains information about its size, dimensions, data type, and storage structure. The input data includes both the image to be filtered and the filter control parameters.

PCMap Wrapper for the filter parameter control map's data. Uses the Data object to store its values and adds information about its dependencies, and a change flag with a list of regions that have been modified.

Filter The object which performs the filtering.

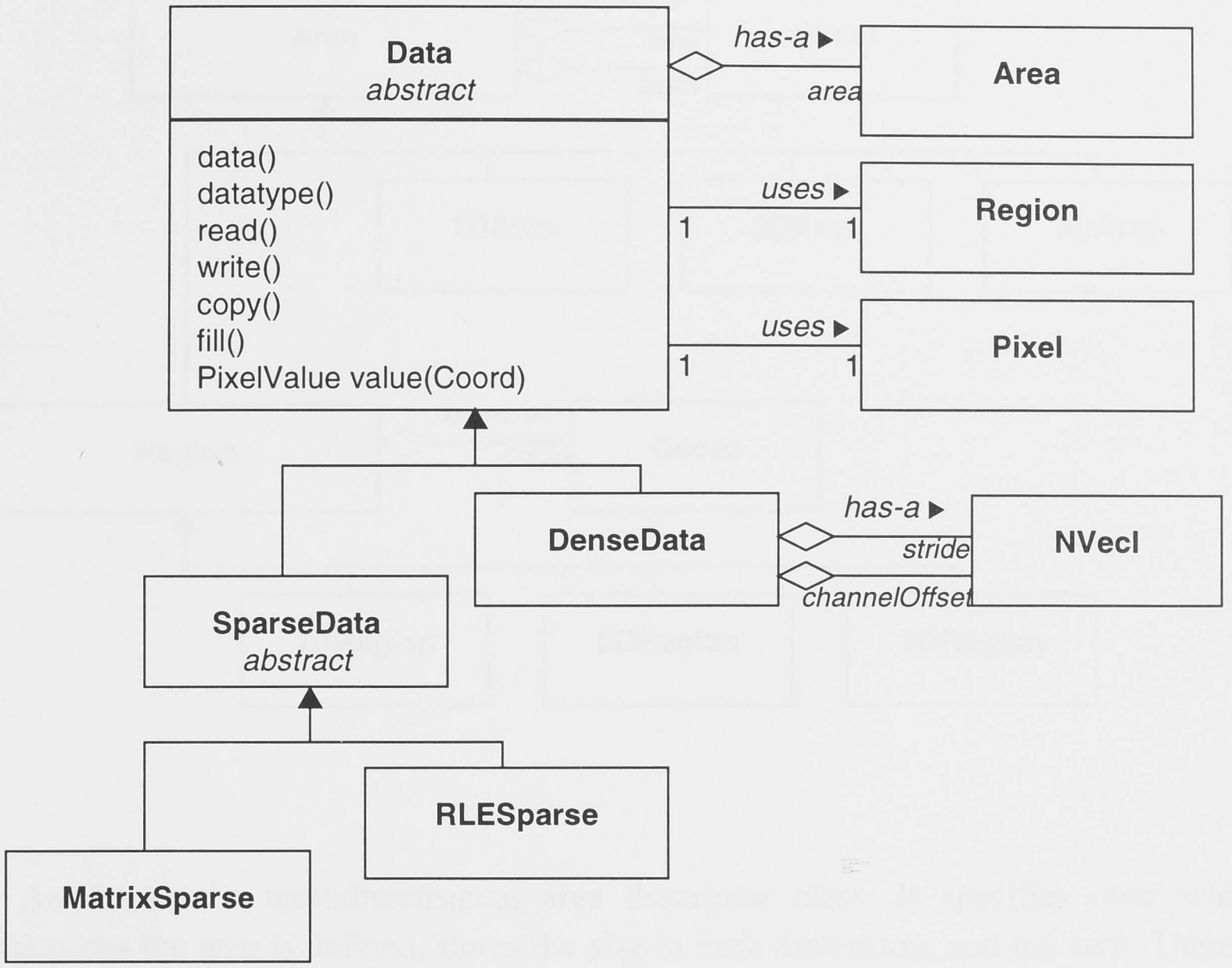


PCMCntrl Objects responsible for generating the values of the filter control parameters from source dependence information.



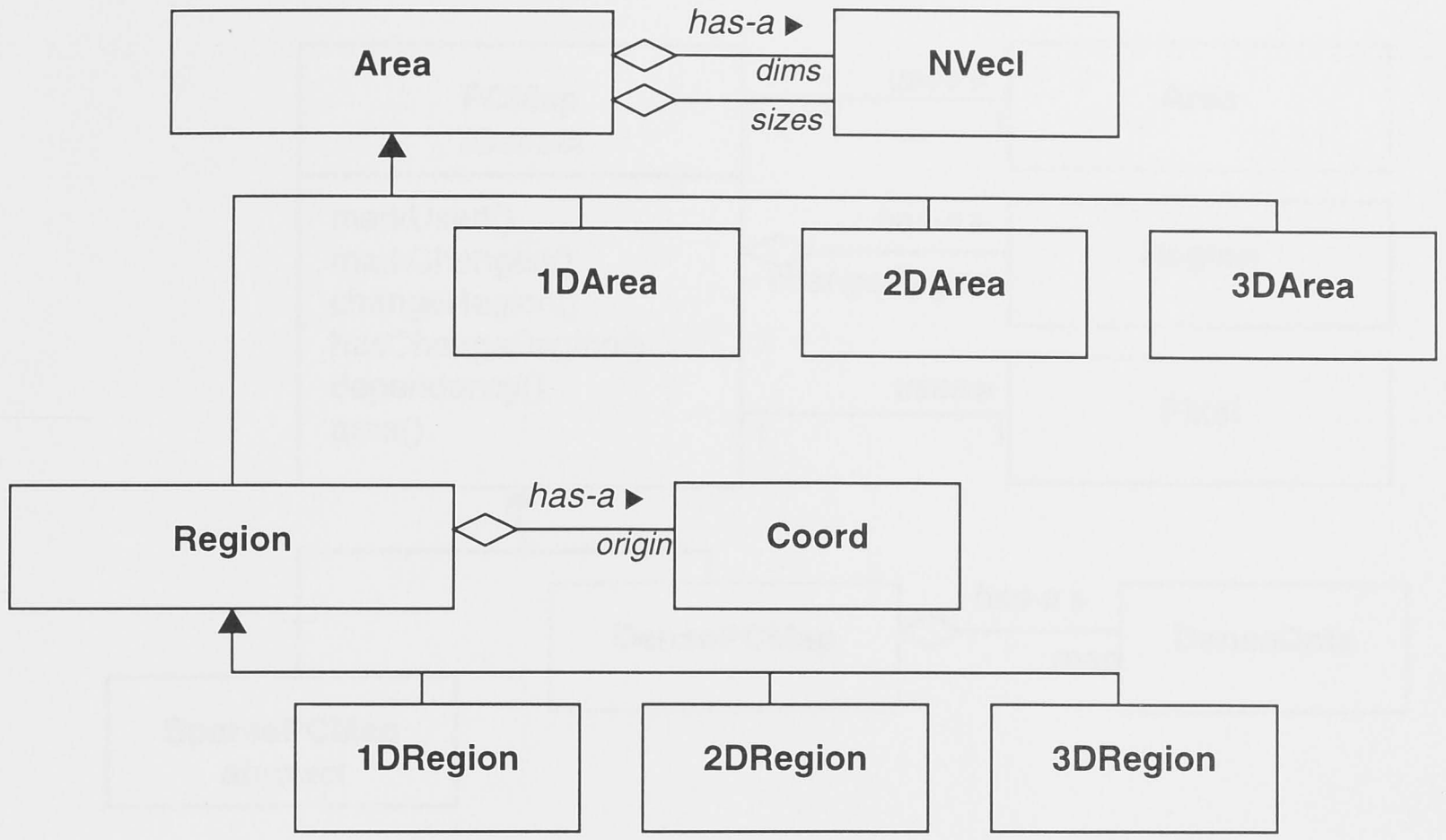
Visualization Is responsible for the visualization of the filtering. Uses the PCMaps, source data, and filtered data.

10.3 Data



The Data class provides an interface to a data buffer. Member functions allow the data to be accessed, read, written, filled, and copied. The Data class is an abstract class with two defined derived classes: SparseData and DenseData. SparsedData objects provide special functions for dealing with sparse data formats. There are two defined sparse data formats: Run length encoded sparse data (RLESparse), and MatrixSparse. DenseData objects handle dense data storage formats.

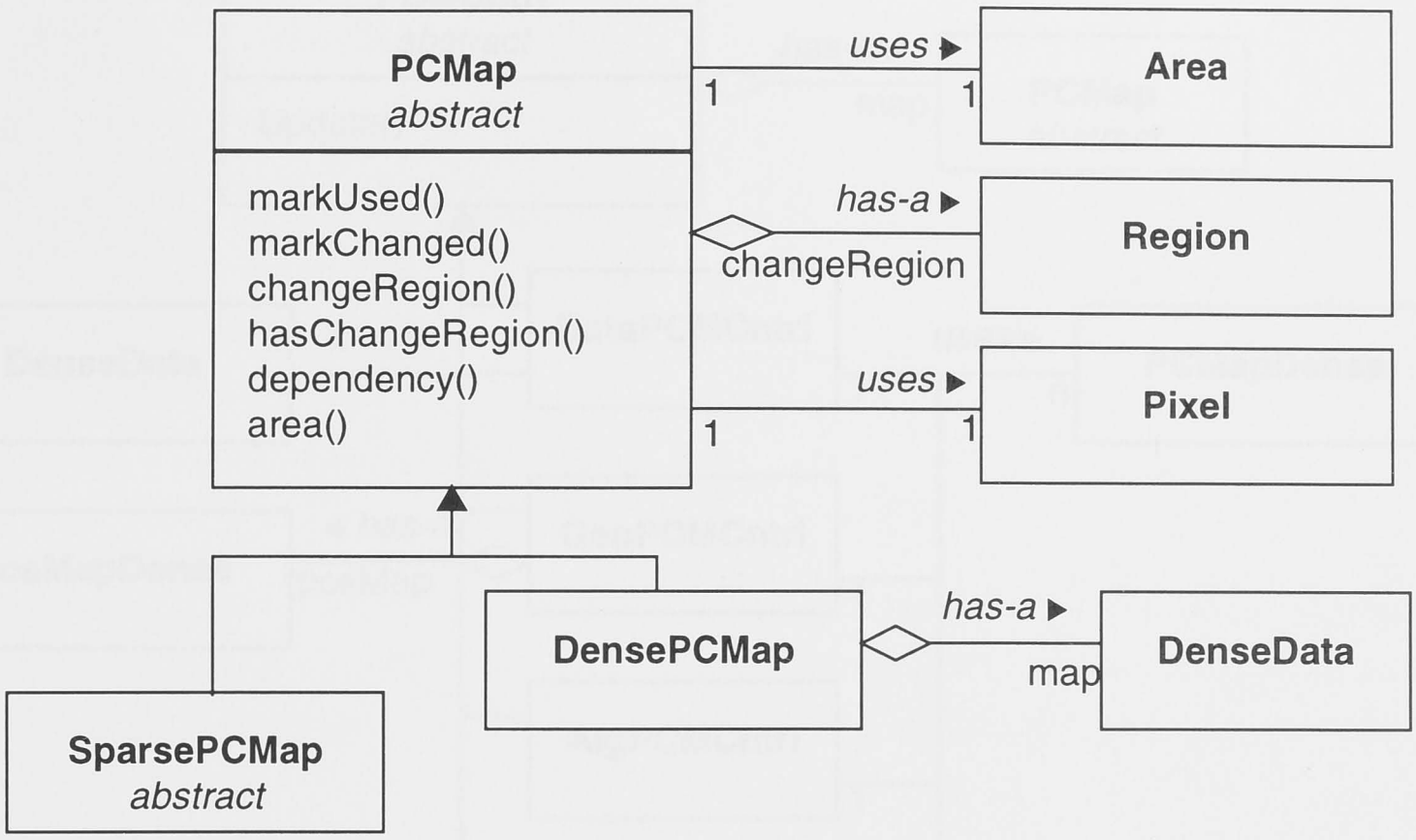
10.3.1 Area & region



An Area is a multidimensional area descriptor class. It specifies over which dimensions the area is defined, stores the size in each dimension, and the area. There is no origin associated with an area. The number of dimensions is fixed at the time of construction. Sizes of the area in each dimension can be set directly by specifying a size.

A Region is a multidimensional region descriptor class derived from the Area class. It specifies over which dimensions the region is defined, stores the size in each dimension, and the origin of the region. Example: A 2D region may be defined over dimension 0 and 2 (dimension 1 is not defined). This is a 2D region that can form part of a 3-dimensional or N-dimensional space. dim(0)=0 and dim(1) = 2. note the dimension orders could have been reversed. dimsize(0) = 20, dimsize(1) = undefined or 0, and dimsize(2) = 20. The number of dimensions is fixed at the time of construction. Sizes of the region in each dimension can be set directly by specifying a size. An origin value gives the location of the region. Alternatively a minimum and maximum value can be set for each dimension.

10.4 Parameter control map

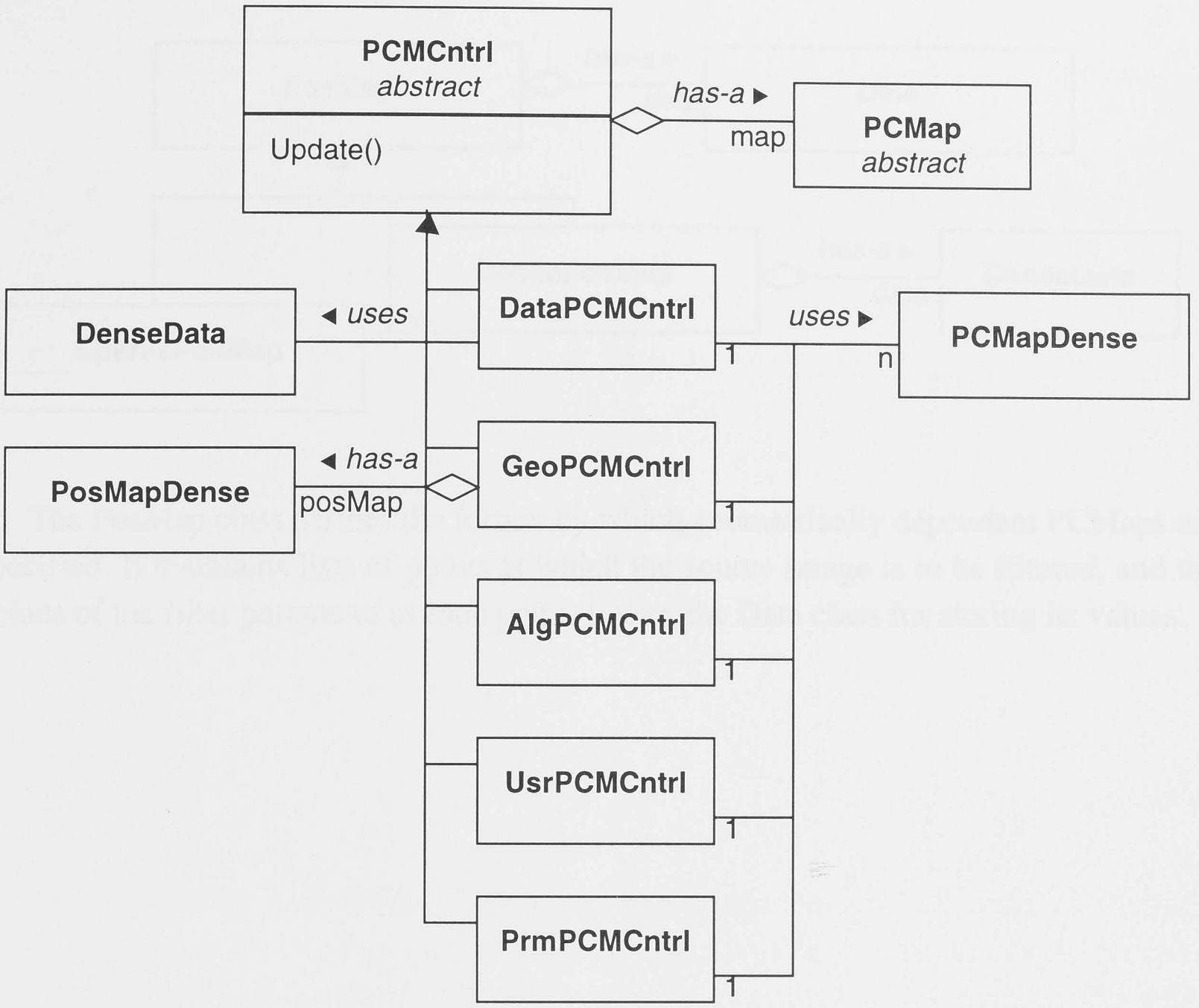


PCMap is an abstract class for parameter control maps (PCMs). Parameter control maps are used to control the filter's operation over an image. They can be scalar, or any number of dimensions. A 1-dimensional PCM specifies variation over a filter parameter over one dimension. If this map is used to control a filter over a 2-dimensional image, then variation will be uniform over the unspecified dimension.

PCMap's have a change flag and region specifier, which indicate whether the map has been modified since it was last used. If the data within the map is modified, the markChanged() function should be called. The markUsed() function can be called by the objects which use the map to say that the data has been used. The changeRegion() indicates which region of the image has been changed. Variation constraints may limit the way in which the PCMap's values are allowed to vary.

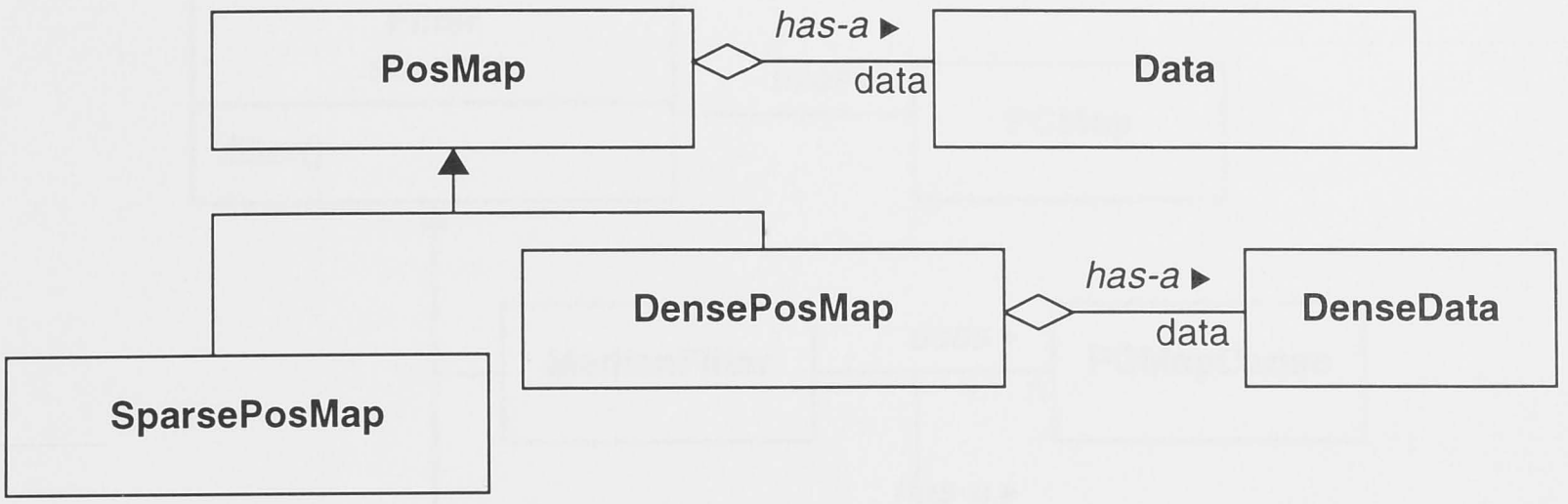
PCMapDense is a dense data instantiation of a PCMap, and SparsePCMap is a sparse instantiation.

10.5 Parameter control map controllers



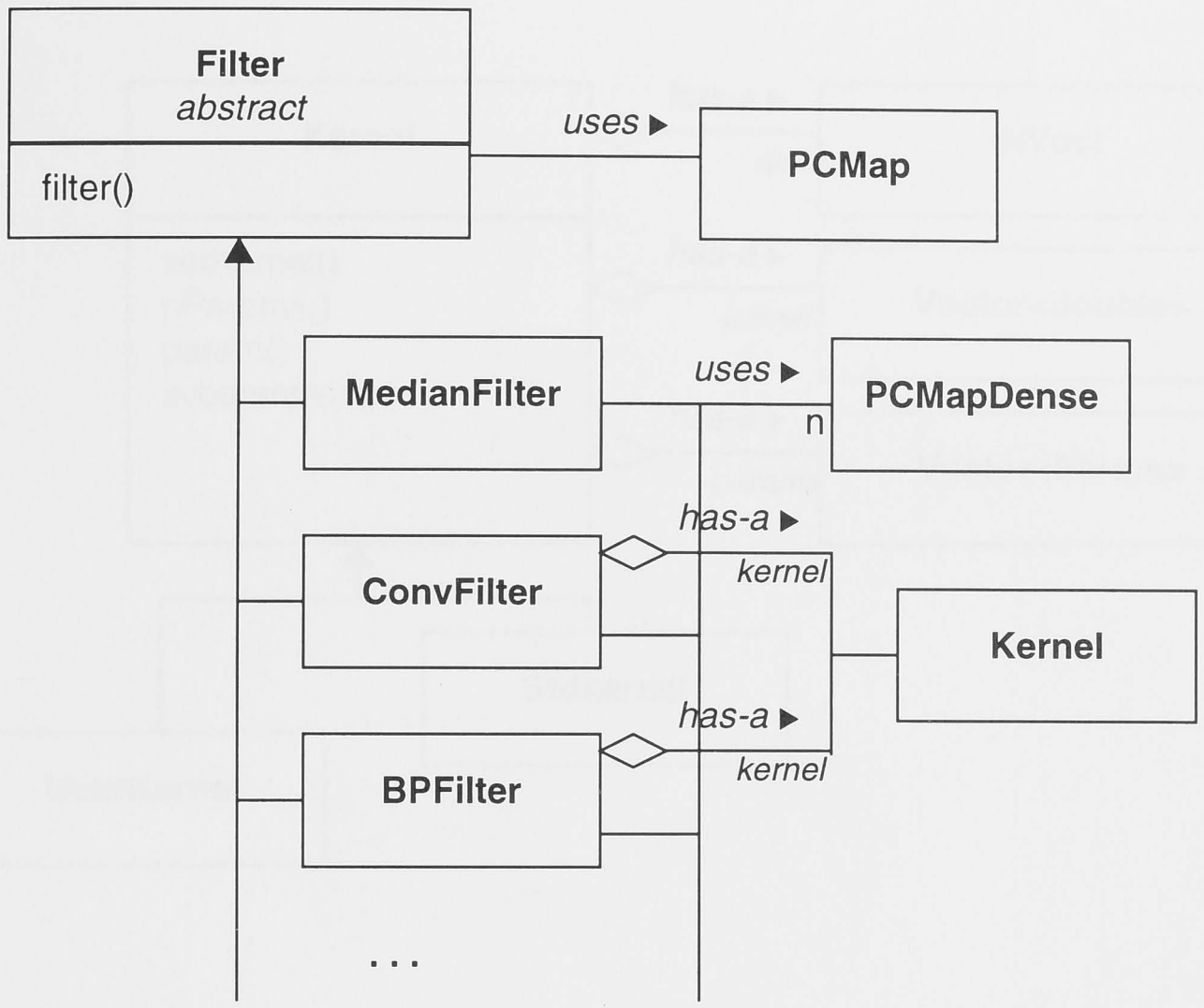
The PCMCntrl base class provides access functions to the PCMap, change flags, and information on the dependency type. It also defines the update() function which asks the controller to perform any outstanding functions and then recalculate the PCMap if necessary. The update() function of the controllers is called by the host application before applying the filtering. There are derived classes for each of the dependency types.

10.5.1 Position map for geometry dependent controllers



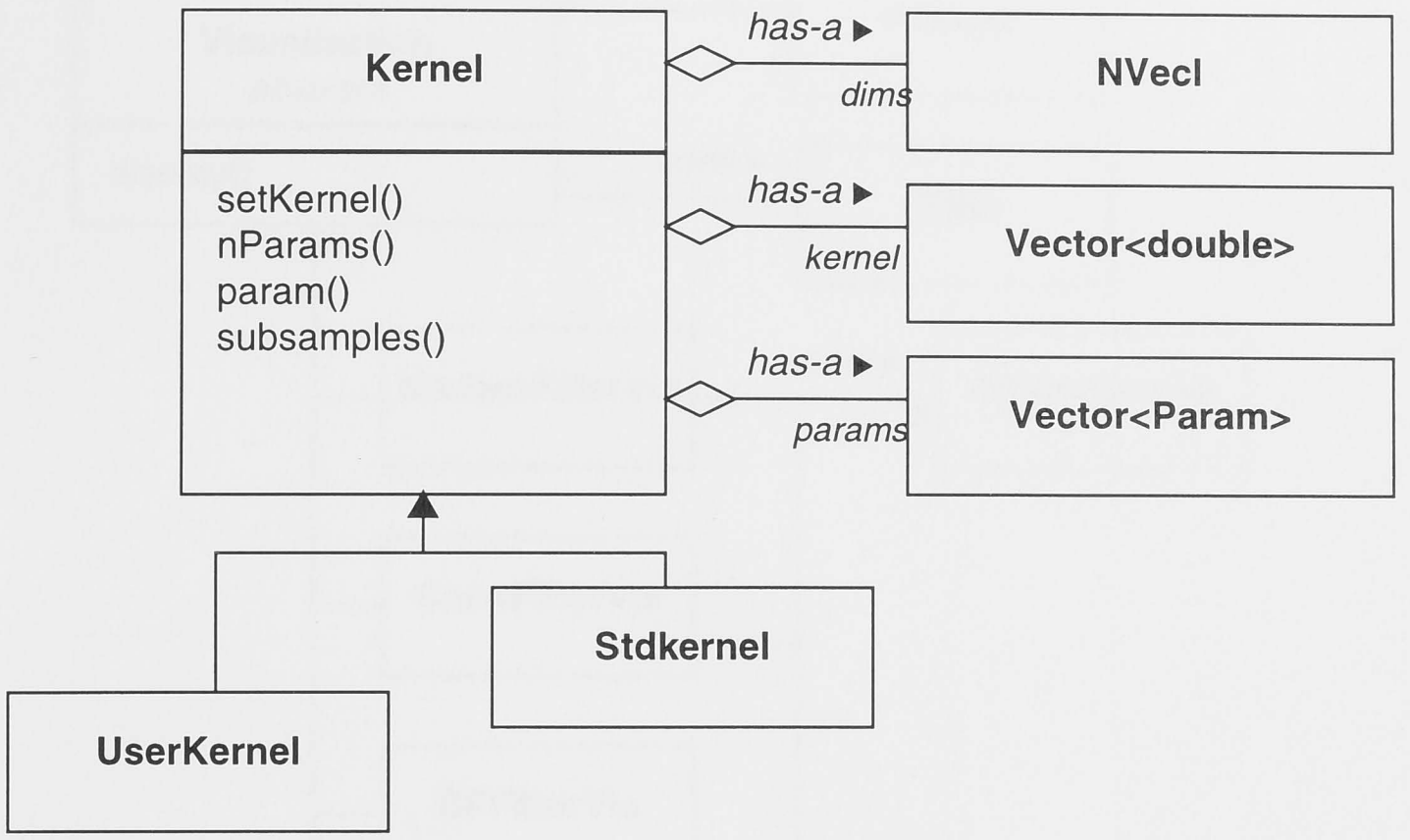
The PosMap class defines the format by which geometrically dependent PCMaps are specified. It maintains lists of points at which the source image is to be filtered, and the values of the filter parameter at each point. It uses the Data class for storing its values.

10.6 Filter



The Filter class is an abstract base class for classes that perform image filtering. It specifies the interface that applications can apply to a variety of filters. Derived classes may need additional functions for initialization and parameters for control. The Filter class defines the filter() function for performing filtering on images based on its PCMs. Functions are also provided for determining the type and number of controlling parameters.

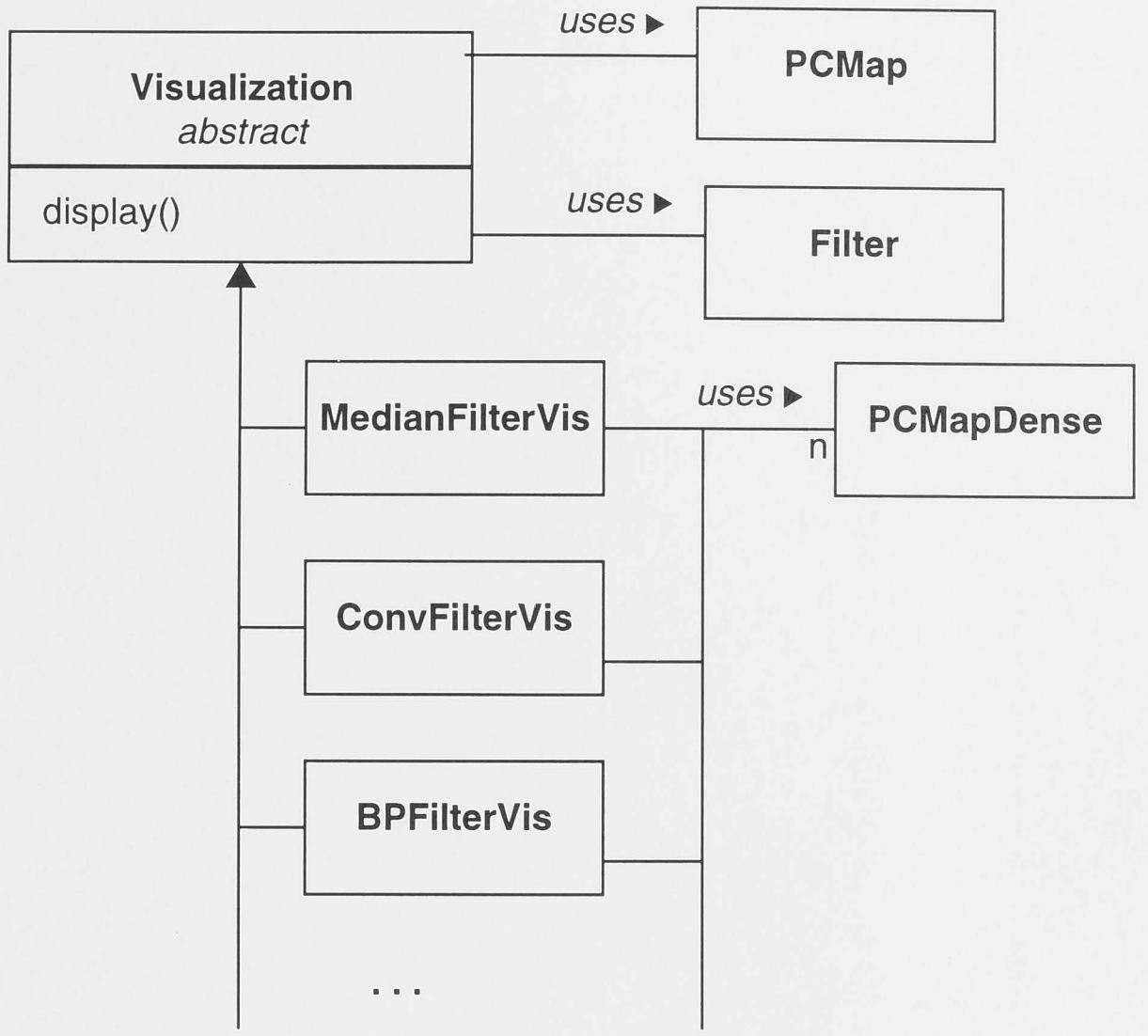
10.6.1 Filter kernel



A Kernel is a sampled filter function used for weighting samples when filtering an image. The kernel's dimensionality is not limited, it can be a 1-dimensional kernel used for 1-dimensional filtering or separable 2-dimensional filtering. It can be a 2-dimensional filter, or even an N-dimensional filter. A Kernel can be created empty, in which case the dimensionality and size of each dimension must be set using the setKernel() function; before this the Kernel is undefined. Or else the Kernel can be initialized to a certain size. The size of each dimension is specified in terms of pixel coverage. The actual number of samples per dimension equals the number of subsamples per pixel times the dimension size.

A Kernel shape can often be controlled by parameters. In the base class there are no parameters, but derived classes may define several. The functions nParams(), and param() allow these parameters to be accessed.

10.7 Visualization



The Vis class is an abstract base class for filter visualization classes. It defines the interface by which host applications can control the visualization of different filter types. The display() function updates internal information and the filter visualization. Host applications may provide window pointers to Vis object for them to draw in. The current implementation of the Vis objects uses the OpenInventorTM 3D graphics toolkit for drawing graphics.