Model Centred Approach to Scientific Visualization

Bahari Belaton[1] **and Ken Brodlie**[2]

[1]School of Computer Sciences, Universiti Sains Malaysia 11800 Penang, Malaysia
[2]School of Computing, University of Leeds, Leeds LS2 9JT, United Kingdom
{bahari@cs.usm.my, kwb@comp.leeds.ac.uk}

ABSTRACT

One of the crucial requirements for a Scientific Visualization system is to produce reliable and accurate results. There are many possible sources of errors that could jeopardise these efforts, such as measurement or simulation errors in the pre-analysis stage, or errors generated in the visualization process itself. Our focus here is to control errors introduced during the visualization processes. To this end we propose a conceptual model for visualization known as the Model Centred Approach (MCA). This new paradigm separates the modelling and viewing processes in visualization, and this provides the opportunity to consistently utilise a single modelling function throughout the visualization process. Results show that consistent visualizations are produced by our approach when compared to conventional methods.

Keywords: scientific visualization, conceptual model

1 Introduction

Scientists of various disciplines face the challenge of correctly interpreting data collected from experiments or produced by simulations. Usually, these datasets are huge and complex in nature, hence the application of conventional analysis tools is either ineffective or unsuitable. Scientific Visualization helps tackle this problem by mapping the raw data set to a graphical form, which can be effectively processed by our visual senses. It has also been shown that visualization can promote data exploration - to uncover new phenomena that are unseen in the raw data.

However, together with the opportunities of visualization, there lurk hidden dangers. In simulation and measurement, there are well established processes for controlling error, and the scientist will typically understand the degree of reliability that can be assigned to a dataset. However the process of turning data into visualization is much less understood. As a result there is a tendency to place greater trust in a visualization than may be warranted. The difficulty with data visualization, as Fred Brooks has observed, is that there is typically nothing to compare against - unlike photorealistic rendering, for example, where the real world scene can be used as the basis of comparison. There is a responsibility therefore to derive measures of accuracy and reliability in visualization. This can be used to give confidence measures for a visualization, as is routinely done in statistical work.

It is not easy to quantify errors in visualization. Important work has been done by Pang *et al* [Pang96], in highlighting different points at which errors can occur in proceeding from data to image. They make a special study of flow visualization. In particle tracing, errors occur when integrating the differential equations which describe the path. Lopes and Brodlie [Lopes98a] suggest ways in which these errors may be displayed, to give an indication of the trust which may be placed in the particle trace. In contouring, Lopes and Brodlie [Lopes98b] use similar techniques to illustrate the reliability of a contour line.

These efforts, however, have tended to focus on a particular visualization technique, aiming to augment the visualization with error information. Our approach in this paper is to address the problem at a higher level. We re-visit the classical dataflow model for the visualization pipeline, suggested by Haber and McNabb [Haber90]. Rather than have data flow through a pipeline, we propose a different paradigm: a two-stage process in which firstly an empirical model of the data is created as a function of the independent variables; and secondly a visualization of this empirical model is produced. By having this single model at the heart of the process as a unique reference, we promote greater consistency throughout the visualization process and a greater understanding of the operations applied in going from data to picture. We call the new paradigm the *model-centred approach*.

This paper is organized as follows: section 2 introduces the concept of accuracy and discusses it in the context of dataflow systems. Then in section 3, we present our proposed model for scientific visualization; this is then followed in section 5 by a description of a case study in which we explore and evaluate the new paradigm. Section 6 discusses how the ideas can be incorporated within existing visualization systems, and finally, in section 7 we conclude our findings.

2 Reliability and Accuracy in the Dataflow Model

The classical model for scientific visualization systems is the dataflow model. In this model, visualization is considered as a sequence of transformations that converts raw data into a rendered image or animation. Following Haber and McNabb [Haber90], these transformations can be categorized into three major groups - data enrichment/enhancement, visualization mapping and rendering (see Figure 1).

A detailed inspection of each stage of the dataflow model will reveal that artefacts or errors are often unintentionally introduced during the visualization process. The two most common sources of errors stem from (i) *implicit* assumptions about the raw data, and (ii) *inconsistent* usage of modelling functions in the dataflow model.

In the data enrichment/enhancement stage, raw data is normally in discrete form and needs to be filtered to remove noise, or converted to a format suitable for later processing - for example, taking scattered data and returning values on a regular grid. This preparation will involve building an empirical model of what we believe the underlying behaviour of the data to be. In doing this we make *implicit* assumptions about the data characteristics - for example we may assume linear behaviour between data points. We have to make these assumptions because typically there is no analytical model available. However it has to be *recognised* as an important source of error in visualization - even if it is inevitable. The dataflow model fails to recognise this crucial step in an explicit manner, since the empirical model is 'hidden' in the data enrichment stage and invisible to the user. The output from the data enrichment stage is simply a refined set of data (for example, re-evaluated on a different mesh), with no record of the modelling function used.

Similar implicit assumptions also occur in the mapping stage - one example is the original Marching Cubes [Loren87] method for isosurface construction. The 'holes' artefacts (see [Durst88, Niels90]) occur because there is an implicit assumption of linear behaviour on the edges, but no assumed model on the faces, or in the interior of the cube. An *ad hoc* decision is used to join up the edge intersection points to form a triangular representation of the isosurface - and this *ad hoc* decision can cause holes when adjacent cubes are handled in different ways. A better approach is to create an empirical model of the data behaviour throughout the volume, and to use that model consistently in the creation of the isosurface. This is done for example by Natarajan [Naraj94] and by Lopes [Lopes99], who both use a trilinear interpolant as basis of the interior triangulation.

Thus we find that individual modules in a dataflow pipeline often make implicit assumptions about the underlying data behaviour. More seriously, in any particular pipeline, different modules may make different implicit assumptions, and hence assume different underlying models of the data, and this can lead to *inconsistencies*. This is clearly unsatisfactory.

We illustrate the problem with a simple example of contouring from scattered height data. Figure 2 shows the typical visualization pipeline that would be used in a dataflow-based system. The pipeline begins with the input of the scattered data set. The contouring module requires gridded data, and so an `Interpolation` module is used to achieve this. From the scattered data, it will build an empirical model of the underlying function from which the discrete samples are assumed to have been taken. It will then output values of that model on a regular grid. Note that the modelling function is internal to the module.

The gridded height data then forms the main input to the `Contour` module which will generate isolines of equal height. To do this, once again an empirical model of the underlying function is again created, but this time from the gridded data. There is no guarantee that the two models used by `Interpolation` and `Contour` are the same. The output is a set of points lying on the isoline. Once again the modelling function is internal to the module.

Finally, the `Render` module generates an image from the isoline points. Again a model is created - typically the isoline is assumed to be linear between the points.

In visualization we are attempting to recreate reality from data sampled at a set of points. The examples just given show that this recreation step is often confused because different assumptions about underlying behaviour are made at different points in the pipeline; and moreover these assumptions are usually implicit and therefore not understood by the user. This is the motivation for the work of this paper.

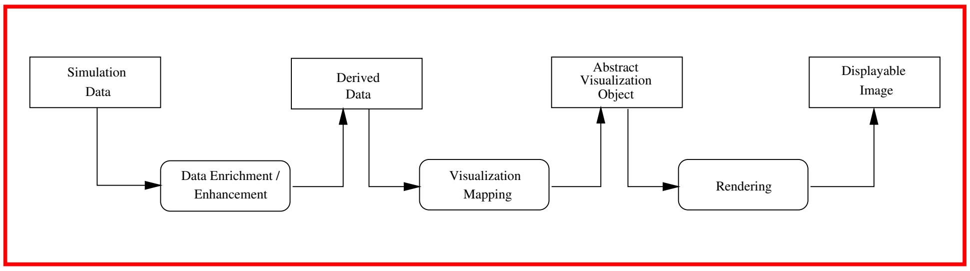

Figure 1: Dataflow Model for Scientific Visualization

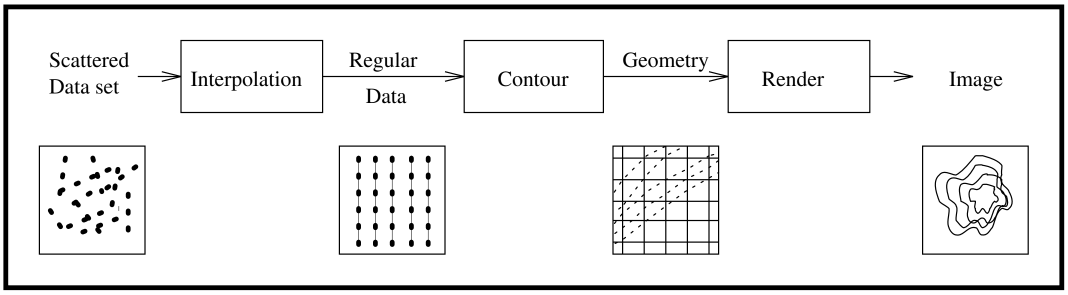

Figure 2: Visualization pipeline for contouring scattered data.

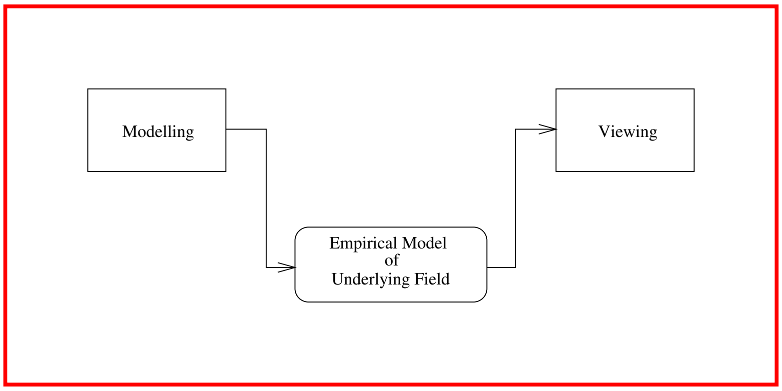

Figure 3: Model-centred approach.

3 Model Centred Approach

The fundamental idea of the model–based approach is that an *empirical model* of the underlying behaviour of the data set should be considered the central focus of the visualization process. Rather than conceal the modelling function(s) within modules of a pipeline, we expose a single empirical model as an interface between two stages of visualization. These two stages are *modelling* and *viewing*. We term this new paradigm the Model–Centred Approach (MCA) and it is shown in Figure 3.

The modelling component is responsible for building the empirical model from the data set. This modelling may be interpolation, or approximation, to describe the general behaviour between sample points. It is critical to the success of a visualization. Typically we *know* the behaviour at the sample points: we have the numbers. What we are interested in is understanding the behaviour *between* the data points. The modelling component is where we make our prediction of what this behaviour is. This modelling component essentially replaces the filtering stage in the data flow model. The key difference is that in our MCA the modelling component produces an expression which is defined everywhere in the domain, not just at sample points. This expression can be evaluated to generate sample data at any resolution whereas in the filtering stage a discrete form of data is produced. A major advantage of our approach is in the visualization of numerical simulations: if the simulation creates a model as part of its solution, this model can be 'inherited' directly rather than having to be evaluated at a discrete set of points, losing information in the process.

The viewing component is responsible for choosing a graphical representation of the model created by the modelling component. Haber and McNabb refer to this as an Abstract Visualization Object (AVO). The difference now is that these abstractions can be based on complete, rather than partial information. Thus for 3D isosurfacing, we work from an implicit function definition

$$f(x, y, z) = c \tag{1}$$

generated by the modelling component. Similarly, for particle tracing, we work from a velocity function rather than velocity data - this removes the error-generating interpolation step within the numerical integration of the particle path. The viewing component is also responsible for display on the graphics device - thus the viewing component essentially replaces both the mapping and rendering stages of the traditional Haber-McNabb pipeline.

4 Potential Benefits of the Model-Based Approach in Visualization

The distinctive feature of MCA is the identification and separation of the modelling process in the visualization. The new architecture *explicitly* decouples the modelling operation from other processes, and considers it as a fundamental process in its own right. This separation provides a different way of doing visualization and offers the following potential benefits :

1. It provides a better conceptual model for visualization.
2. It facilitates the importing of the model from a simulation process, and hence to more accurate results.
3. It allows more opportunity for scientists to experiment with different modelling function(s) when working with measured data.
4. In some circumstances, it provides a form of data compression by representing the model in a mathematical formula which can be stored compactly.

The following paragraphs look at these benefits in more detail.

Better conceptual model

The two stage architecture of MCA provides a better conceptual model of visualization for scientists and engineers. Rather than highlighting the extended steps of image production processes which are more suitable to graphics experts than scientists, MCA identifies two stages - *modelling* and *viewing* which are conceptually distinct. It acknowledges the importance of the underlying field from which the given data is only a sample, and encourages the scientist to think carefully about how they wish to predict behaviour between data points. It promotes greater accuracy and consistency because there is only one modelling process, not several. It hides the detail of the picture production pipeline in a single viewing component and stresses the requirement to base the visual construction from the functional representation created earlier in the modelling component.

Simulation and visualization

In certain applications, there will already exist an empirical model. For example in numerical simulations, the solution of partial differential equations will typically be defined as a model, not as data. Of course, it may not be particularly easy to export the model -

but there has to date been little pressure on computational scientists to solve this problem, since the interface to analysis software has always been in the form of data. The MCA approach, by explicitly identifying the model construction as a separate process, allows a model to be imported directly and passed directly to the viewing component. This is not possible in traditional data flow systems, with the result that certain valuable information can be lost as the data is reduced from a continuous model to discrete sample data. As described before, the model reconstructed from this sample data is implicit, ill-defined and often inconsistent with the model in the numerical simulation.

Measured data and visualization

The importing of a model is only suitable for cases when the model of the underlying field is known, such as in simulation. In the case of data gathered through experiment or measured/captured by scanning/sensor devices, the model of the underlying field is unknown. This is the case in many real world applications such as medicine, chemistry, environmental studies, etc. The explicit handling of this process in the MCA modelling component gives more opportunity to the scientist to experiment with various modelling methods. This contrasts interestingly with the current paradigm of the data flow model where scientists are offered a variety of mapping techniques, but typically little freedom to vary the modelling function. Because modelling and mapping are intertwined, a scientist needs to modify the mapping module as a whole if they want to experiment with a different modelling method from the one currently embedded in the module. In contrast, because MCA treats modelling as a process in its own right, new modelling methods can be added independently from the existing mapping modules.

Data compression

The model constructed in MCA's modelling component can act as a form of data compression. This is especially true for approximation rather than interpolation. It can be more efficient and economical to manipulate a model that estimates the relationships, behaviours and characteristics of the data set, compared to processing a large data set itself (a well known problem in dataflow environments). A good example is in surface fitting where we are approximating a large 2D dataset with say a bicubic spline. The model consists of coefficients and knots of the spline, giving good compression over the raw data.

5 Case Study

For a simple case study of the MCA method, we look at the example of contour plotting in 2D. The case study extends in an obvious way to isosurfacing in 3D and indeed further examples are can be found in the thesis of one of the authors [Belat95] (including the surface fitting example mentioned above).

We shall use the case study to explore the following aspects:

1. We look at the situation where a model is created externally, say in a simulation process, and a visualization using contouring is required. We contrast the use of MCA - where the model may be directly imported, with the use of a conventional approach - where data is exported from the simulation and then passed to the contourer. We see the greater *accuracy* which comes from using the MCA method.
2. We look next at the situation where data is to be visualized - rather than a model. We show the *inconsistency* that can occur in conventional contouring - where scattered data is first interpolated using method `A` to get values on a regular grid, and then contoured from the grid using interpolation method `B`. By contrast, MCA uses one method throughout.

Accuracy in visualizing a simulation model

For simplicity, we take our simulation model as the 'four peak' function defined as:

$$z(x, y) = f(x) * f(y), \quad x, y = 0...1 \tag{2}$$

$$f(t) = \exp^{-\frac{1}{2}\left(\frac{t-0.3}{0.07}\right)^2} + \frac{1}{2}\exp^{-\frac{1}{2}\left(\frac{t-0.7}{0.12}\right)^2} \tag{3}$$

In the Model–Centred Approach, we can import this model directly into a contouring routine that operates on an implicit function approach - here we have used the routine `j06gff()` from the NAG Graphical Library [NAG]. This contours from a function supplied as a Fortran subroutine. It works by evaluating the function on a mesh in order to detect presence of a contour, and to give a starting point. It then tracks the contour through the region in small steps, evaluating the function as it goes.

In the conventional approach, we have to first evaluate the function on a regular grid of a predefined size, and pass this data to a grid-based contouring routine. Here we have used `FARBE-2D`, a contouring routine developed by Preusser [Preus89], and available freely from the ACM Transaction of Mathematics archive. Internally (hence implicitly) the routine creates a bicubic patch for every rectangle of the mesh. This bicubic polynomial is used to track contour lines within each rectangle.

The difference between the two approaches becomes clear when we zoom in on an area of interest. Figure 4 shows three sequences of snap-shots that demonstrate the greater accuracy of the MCA method on zooming. In each snap-shot of a sequence we zoom in on the area shown to plot contours in more detail in the succeeding snapshot. In each case the functional model is used by `j06gff()` to define the contours. Throughout all levels of detail, accuracy is maintained.

By contrast, the conventional approach using `FARBE-2D` on an initial grid size of `40x40` is shown in Figure 5. The same zooming operations are applied, but each successive snapshot of a sequence is necessarily based on less data than its predecessor (the grid size on which the plot is based is shown underneath each snapshot). The result is that the piecewise bicubic generated by `FARBE-2D` steadily becomes less and less accurate. Indeed we reach a stage with a `2x2` grid where we cannot zoom any further because we have reached the minimum grid size.

This case study shows the advantage of the model–based approach when a simulation model is available. The key difference is the point at which *discretisation* is carried out: in the conventional approach, the first step is to discretise the model at some fixed resolution (as we have seen that gives problems on later zooming); in the model–based approach, the discretisation is postponed until as late as possible in the processing pipeline.

Consistency in visualizing data

We now look at the problem of *inconsistent* usage of functional models in a visualization pipeline - again using contouring as our example. In this case, however, we suppose we are given a set of scattered, or unstructured, 2D data, rather than a model. The results are illustrated in Figure 6.

The conventional approach is to use an interpolation method (here the Renka and Cline C^1 interpolation method [Renka84], as implemented in the NAG Library routine `e01saf` [NAG]) to derive a regular grid dataset, which can then be passed to a contouring routine (here `FARBE-2D`). The results are shown in the top right of Figure 6 for a range of different grid sizes.

In the MCA, the Renka-Cline interpolation method `e01saf` provides the unique model for the contouring method. A mesh is still needed, but only to detect the existence of contours, not their position. Two pictures are shown at the bottom of Figure 6 - on the left, a tracking step of 0.5 is used in `j06gff()`; on the right, greater accuracy is achieved with a smaller step-size. The mesh size is shown - an advantage of the finer mesh size is the improved location of the contour intersection with the boundary, and the greater chance of detecting small closed contours (as can be seen in the Figure).

Although the results are superficially quite similar, the MCA with Renka-Cline and `j06gff()` does give a more consistent sequence of images. A key aspect again is the discretisation step: the lower the grid resolution, the more that the conventional approach relies on the bicubic approximation within `FARBE-2D`, rather than reflecting the Renka-Cline interpolant. By contrast, reassuringly the MCA will always reflect the true Renka-Cline interpolant.

6 Using MCA within a Modular Visualization Environment

In this section we investigate how the MCA concept might be incorporated within existing visualization systems. This is motivated by pragmatism. One cannot expect established users to abandon their ‘favourite’ visualization system. If we can incorporate our ideas within existing systems, we have the potential of reaching a large user population. Thus we look to see how easy it is to extend current systems to support the idea of an explicit modelling step. By exposing the modelling step, it should be possible to allow a user to try different modelling functions.

A popular class of system is the Modular Visualization Environment, or MVE. These are based on the data flow reference model of Haber and McNabb. The systems provide a library of predefined modules to carry out different parts of the pipeline: data enrichment, mapping and rendering. Visual programming is used to select modules from the library and combine them in a data flow network. Each module takes data in, processes it, and outputs results downstream. The systems are also extensible in that users can add their own modules. Examples of MVEs include IRIS Explorer [NAG], AVS [AVS] and IBM Data Explorer [IBM].

We have developed a simple prototype using IRIS Explorer. The major difficulty to overcome is that only data, not functions, may flow from module to module. Hence the modelling functions used in data enrichment and mapping modules are never exposed. The simplest way to incorporate MCA is to build a new module which incorporates both data enrichment and mapping into a single unit. The modelling function becomes a parameter of the module, and can be specified via the user interface allowing the user to experiment with different choices. By merging the two steps in one module, we force the use of a unique modelling function. Thus the consistency property of the MCA approach is realised. Equally, the scientist is still able to exploit the many useful utilities from IRIS Explorer - render module, data readers, colour maps and so on.

7 Conclusion

The primary contribution of this work is the development of a new framework for scientific visualization systems based on the concept of the model–centred approach. This replaces the conventional dataflow pipeline with a two-stage approach: the first stage creates an explicit model of the data; the second stage provides a view of the model. We claim this gives a clearer conceptual basis to the scientist, emphasising the central position that modelling plays in building a representation of the underlying phenomenon that is being studied.

It promotes accuracy and consistency: accuracy through allowing an external simulation model to be directly used in visualization, rather than being first discretised; consistency through the use of a unique model of the data throughout the process.

Finally we have shown how the principle of MCA can be accommodated within a conventional modular visualization environment.

REFERENCES

[AVS] See the AVS web site - http://www.avs.com.

[Belat95] Bahari Belaton. *A Model Based Approach to Scientific Visualization*. PhD thesis, School of Computer Studies, University of Leeds, September 1995.

[Durst88] M. Durst. Additional reference to marching cube. *Computer Graphics*, 22(2):72–73, April 1988.

[Haber90] Robert B. Haber and D. A. McNabb. Visualization idioms: A conceptual model for scientific visualization systems. In Gregory M. Nielson, B. Shriver, and L. J. Rosenblum, editors, *Visualization in Scientific Computing*, pages 74–93. IEEE Comp. Society Press, 1990.

[IBM] See the IBM Open Visualization Data Explorer web site - http://www.research.ibm.com/dx.

[Lopes98a] Adriano Lopes and Ken Brodlie. Accuracy in 3d particle tracing. In Hans-Christian Hege and Konrad Poltheir, editors, *Mathematical Visualization*, pages 329–341. Springer-Verlag, Heidelberg, 1998.

[Lopes98b] Adriano Lopes and Ken Brodlie. Accuracy in contour drawing. In *Eurographics UK Conference Proceedings*, pages 301–311. University of Leeds, March, 1998.

[Lopes99] Adriano Lopes. *Accuracy in Scientific Visualization*. PhD thesis, School of Computer Studies, University of Leeds, 1999.

[Loren87] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface reconstruction algorithm. *Computer Graphics*, 21(4):163–169, 1987.

[NAG] See the nag web site - http://www.nag.co.uk.

[Naraj94] B Narajan. On generating topologically consistent isosurfaces from uniform samples. *The Visual Computer*, 11:52–62, 1994.

[Niels90] Gregory M. Nielson and B. Hamann. The asymptotic decider: Resolving the ambiguity in marching cube. In *Proceedings of Visualization'90*, pages 83–91. IEEE Computer Society Press, Los Alamitos, 1990.

[Pang96] Alex T. Pang, Craig M. Wittenbrink, and Suresh K. Lodha. Approaches to uncertainty visualization. In R. Yagel and G. Nielson, editors, *Proceedings Visualization '96*, pages 249–254. ACM Press, 1996.

[Preus89] Albrech Preusser. Algoritm 671 farbe-e-2d: Fill area with bicubics on rectangles - a contour plot program. *ACM Transactions on Mathematical Software*, 15(1):79–89, March 1989.

[Renka84] R.L. Renka and A.K. Cline. A triangle-based c^1 interpolation method. *Rocky Mountain Journal of Mathematics*, 14:223–237, 1984.

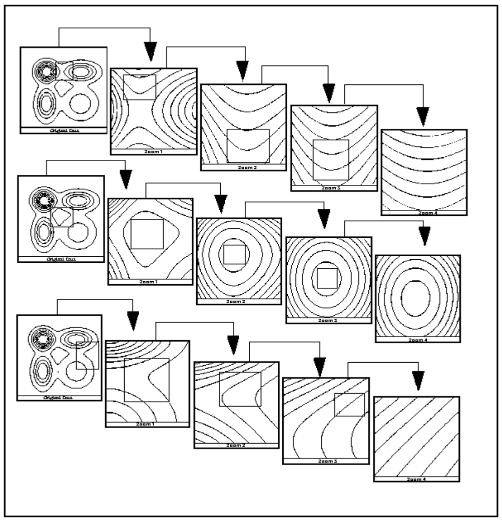

Figure 4: The snap shots of four peak function as zoomed using MCA system.

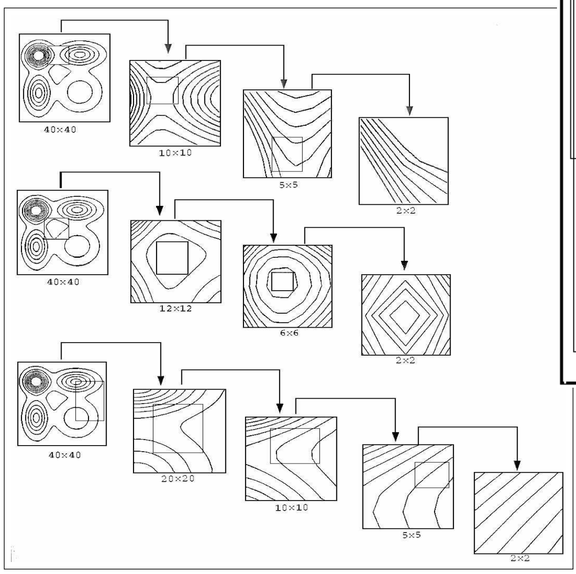

Figure 5: The snap shots of four peak function as zoomed using FARBE-2D system.

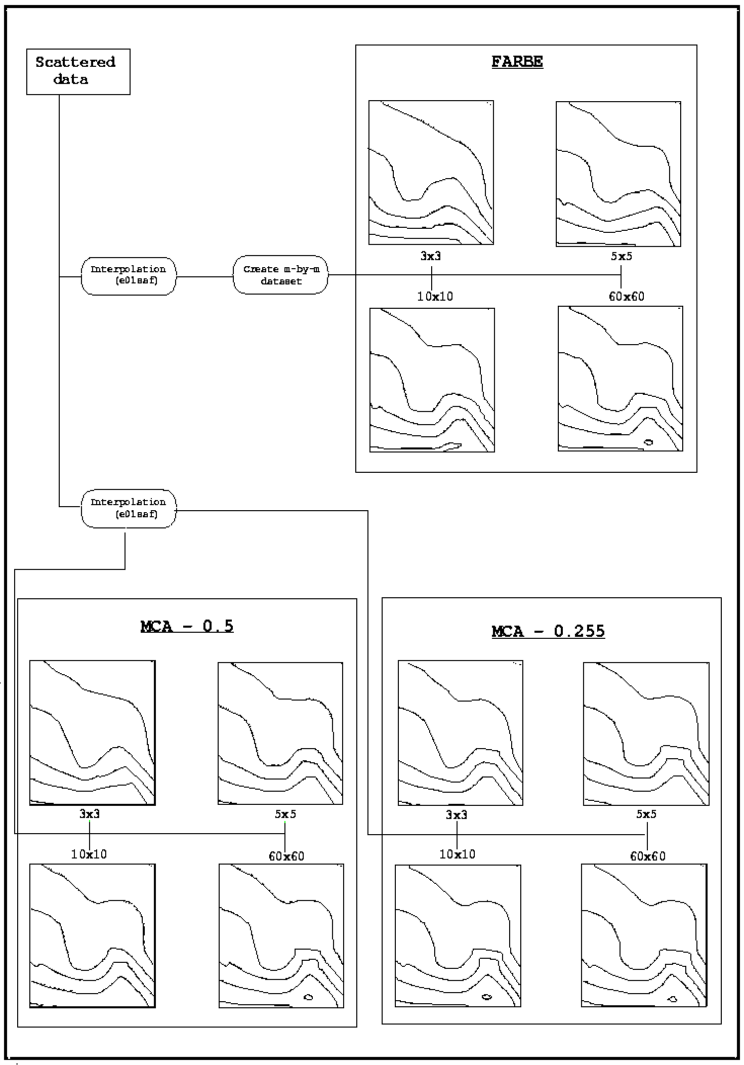

Figure 6: Consistency results.